ТЕМА 1. МЕТОДИКА ПРОЕКТИРОВАНИЯ ЭКСПЕРТНЫХ СИСТЕМ НА БАЗЕ ИНСТРУМЕНТАЛЬНЫХ СРЕДСТВ СЕМЕЙСТВА ЭКСПЕРТ-ЭКО

1.1. Назначение комплекса

Комплекс представляет собой совокупность программных средств, предназначенных для создания и использования экспертных систем, решающих задачи в статических проблемных областях.

Комплекс может быть использован для создания экспертных систем диагностики технических, биологических объектов, вывода эвристических оценок риска или надежности (в строительстве, медицине и т.д.), качественного прогнозирования, обучения.

Средствами комплекса экспертные системы могут создаваться непосредственно высококвалифицированными специалистами в области приложения (экспертами), не обладающими навыками программирования. В сложных случаях они могут привлекать специалистов в области создания экспертных систем (инженеров по знаниям).

Экспертные системы, создаваемые средствами комплекса, позволяют решать конкретные прикладные задачи, а также объяснять, ЗАЧЕМ во время решения пользователю задается тот или иной вопрос и КАК получен результат.

Комплекс работает в диалоговом режиме. Использование подсказок и управление диалогом с помощью меню обеспечивает доступность средств комплекса для пользователя-непрограммиста.

Комплекс имеет средства для организации взаимодействия с внешним программным окружением.

1.2. Использование комплекса для разработки экспертных систем

Программы комплекса представляет собой настраиваемую "оболочку" экспертной системы.

"Оболочками" называются такие инструментальные средства, в которых реализован некоторый стандартный способ представления знаний

(т.е. реализован некоторый язык представления знаний) и стандартизован механизм вывода решений. Подобная стандартизация позволяет значительно снизить трудоемкость разработки экспертных систем, так как "оболочка" содержит все программные средства, необходимые для создания и использования этих систем. "Оболочки" включают в себя следующие подсистемы: средства приобретения знаний, средства решения задач (ведения консультации), средства объяснения и ведения диалога с пользователем, а также базу знаний (возможно, пустую). Разработка экспертной системы с помощью "оболочки" сводится к введению в базу средствами приобретения знаний описания знаний эксперта о проблемной области и порядке решения задач, выраженного на языке представления знаний "оболочки". Решение конкретных задач осуществляется средствами ведения консультации путем интерпретации содержимого базы знаний с учетом данных о конкретной ситуации в проблемной области (исходных данных консультации).

Средствами комплекса реализованы два режима работы: режим приобретения знаний и режим консультации (режим решения задач). Оба режима обеспечивают диалоговое взаимодействие с пользователем. Режим консультации обеспечивает объяснение получаемых решений.

Функционирование комплекса в различных фазах существования экспертных систем

Выделяют следующие фазы существования экспертных систем: создание, использование и модификацию. Программы комплекса используются во всех фазах. В фазе создания (модификации) с помощью средств комплекса разработчик экспертной системы может вводить в базу знаний (модифицировать) описание знаний эксперта, выраженное на языке представления знаний. Кроме того, он может отлаживать базу знаний, решая с ее помощью тестовые задачи и устраняя найденные ошибки. При традиционном подходе к разработке программ фазе создания экспертной системы соответствуют этапы алгоритмизации, программирования и

отладки. В фазе использования экспертной системы программы комплекса применяются для решения задач из проблемной области.

В фазе создания (модификации) экспертной системыпользователями комплекса являются:

- эксперты, т.е. специалисты в той области знаний, задачи которой должна решать экспертная система;
- инженеры по знаниям, т.е. специалисты по проектированию экспертных систем.

Эксперт должен построить адекватную модель проблемной области, т.е. описать саму область и указать порядок решения задач данного типа.

Инженер по знаниям должен: помочь эксперту выявить знания, необходимые для функционирования экспертной системы; выразить выявленные знания на языке представления знаний; при необходимости выделить и запрограммировать традиционными средствами некоторые нестандартные функции, не предусмотренные в комплексе.

Разработанное описание модели проблемной области, выраженное на языке представления знаний, вводится в базу знаний в режиме приобретения знаний и преобразуется во внутреннее представление модели (в дальнейшем по тексту - модель). Ввод описания может осуществляться по частям, при этом проверяется синтаксическая корректность вводимых фрагментов.

После того как описание модели введено полностью, проверяется целостность модели. Если в ходе проверки ошибки не обнаружены, полученная модель помечается как корректная (т.е. готовая к работе), после чего она может быть использована в режиме консультации. При обнаружении ошибок сообщения о них заносятся в протокол проверки, а модель помечается как неготовая к работе. Ошибки могут быть исправлены путем коррекции описания модели в режиме приобретения знаний.

Экспертная система считается созданной, если в базе знаний получена соответствующая корректная модель.

Допускается создание иерархических экспертных систем, включающих несколько взаимосвязанных моделей. Каждая модель предназначена для решения некоторой частной подзадачи в проблемной области, причем одна из них - управляющая, или главная. Все модели оформляются в виде отдельных файлов данных. Единственным ограничением на количество моделей в базе знаний является объем доступной памяти на внешнем носителе информации.

Решение задачи в проблемной области будет осуществляться программами комплекса в режиме консультации на основе:

- одной или нескольких взаимосвязанных моделей, находящихся в базе знаний и представляющих знания эксперта о методах решения задач данного типа (например, модель для диагностики сердечно-сосудистых заболеваний);
- исходных данных, описывающих конкретную ситуацию в проблемной области, для которой пользователь хочет получить результат.

Чтобы получить решение задачи, необходимо в режиме консультации выбрать в базе знаний соответствующую модель (или главную модель), а затем в ответ на вопросы программ комплекса ввести те исходные данные, в которых возникает необходимость в ходе вывода решения.

Для проверки адекватности модели необходимо провести ее тестирование в режиме консультации. В ходе тестирования экспертная система должна решить достаточное количество контрольных задач. Результаты ее работы следует сравнить с решениями, предложенными экспертом. Кроме того, с помощью средств объяснения эксперт должен проверить правильность рассуждений системы в ходе вывода решений. Если эксперт не удовлетворен решениями или рассуждениями системы, он может с помощью средств объяснения локализовать ошибки, а затем исправить их в режиме приобретения знаний.

В фазе использования экспертной системы с программами комплекса взаимодействуют конечные пользователи, т.е. те специалисты, для которых создавалась система. В этой фазе комплекс работает только в режиме консультации.

Создаваемые средствами комплекса экспертные системы могут использоваться как неспециалистами, так и специалистами в областях приложения этих систем. В первом случае система дает пользователю совет, который тот не может получить сам. Во втором случае использование системы ускоряет процесс получения результата и (или) освобождает специалиста от рутинной работы.

В данной фазе средства объяснения обеспечивают доверие пользователя к полученным экспертной системой результатам, а также демонстрируют ему применяемые экспертами методы решения задач, которые можно использовать в обучении.

1.3. Представление знаний экспертов

База знаний может содержать несколько моделей, предназначенных для поддержания работы экспертных систем в различных областях приложения. Каждая модель включает описание проблемной области и знания о порядке решения задач (сценарий консультации).

Описание проблемной области состоит из описаний атрибутов и правил вывода. Атрибуты используются для описания состояния проблемной области, например атрибуты "возраст", "диагноз" и т.д. Описание проблемной области должно содержать описания всех атрибутов, которые будут использоваться при решении задачи. Описание атрибута включает область определения этого атрибута, а также некоторую лингвистическую информацию, необходимую программам комплекса для ведения диалога с конечным пользователем во время консультации. Комплекс ориентирован на работу со статическими проблемными областями (значения атрибутов не

изменяются в ходе решения задачи), в которых проблемная область может быть описана с помощью априорно заданного набора атрибутов (не допускается динамическое создание атрибутов во время решения задачи).

Средства комплекса позволяют представлять качественные и количественные характеристики проблемной области. Высказывания типа "А есть В", где А - атрибут, представляющий качественную характеристику, а В - элемент области определения этого атрибута, называются утверждениями о состоянии проблемной области. Например, высказывание "диагноз - острый бронхит" является утверждением в предметной области, если в ней определен атрибут "диагноз", одним из возможных значений которого является "острый бронхит". Решение задачи сводится к получению значений некоторых целевых атрибутов (например, "диагноз" или "метод обезболивания"), и/или определению истинности некоторых целевых утверждений.

Значения атрибутов и сведения об истинности утверждений о состоянии проблемной области либо запрашиваются в ходе консультации у конечного пользователя (исходные данные), либо вычисляются по правилам вывода на основе исходных данных.

Сценарий консультации указывает, значения каких атрибутов должны быть получены в результате консультации, в каком порядке их следует выводить, каким образом сообщать о полученных результатах и т.д. Сценарий и правила вывода образуют процедурные знания, причем сценарий выполняет роль метазнаний по отношению к правилам. Описания атрибутов и исходные данные образуют декларативные знания о проблемной области.

Программные средства комплекса позволяют работать с неточно определенными (неопределенными) знаниями о качественных характеристиках проблемной области. Неточная определенность декларативных знаний представляется с помощью коэффициентов определенности, характеризующих степень уверенности в истинности утверждений о состоянии проблемной области. Неточная определенность

правил может представляться с использованием формул нечеткой логики, формулы Байеса, а также произвольных эвристических формул, задаваемых экспертами.

Средствами комплекса можно создавать такие системы, которые получают все возможные решения коэффициентов определенности.

Коэффициент определенности является действительным числом с точностью до двух знаков после запятой и принимает значения из диапазона от -5.00 до 5.00.

В комплексе реализован диалоговый ввод знаний с семантико-синтаксическим контролем знаний, проверкой их целостности, а также компиляция знаний, повышающая эффективность работы комплекса в режиме консультации.

Решение задач осуществляется в режиме диалога конечного пользователя с программными средствами комплекса. В ходе решения задачи делается попытка вывести значения тех атрибутов, которые отмечены в сценарии как целевые, а также выполняются другие описанные в сценарии действия (выдаются сообщения о результатах и т.д.). При выводе значений атрибутов используется метод поиска решения " в глубину" с последующим выводом значений от данных.

В ходе решения задач пользователь может получить объяснения, для чего потребовались запрашиваемые у него данные и как получено решение задачи.

Средства комплекса могут проводить трассировку хода решения задачи, что облегчает отладку моделей.

Кроме того, программы комплекса могут обмениваться данными с внешними программами.

1.4. Структура комплекса

Комплекс состоит из основных (ОПС) и вспомогательных программных средств. Основные программные средства состоят из

взаимосвязанных программных компонент, использование которых в режимах консультации и приобретении знаний обеспечивается автоматически. Вспомогательные средства реализованы в виде автономных программ, применение которых при создании экспертных систем не является обязательным. В то же время, использование этих средств позволяет улучшить качество создаваемых систем или облегчить процесс их создания.

ОПС поставляются в двух вариантах: основном и учебном. Основной вариант предназначен для создания промышленных экспертных систем. Учебный - для обучения работе с комплексом на примере "игрушечных" экспертных систем небольшого объема.

Структура основных программных средств комплекса включает:

- Редактор, обеспечивающий работу в режиме приобретения знаний;
- Консультатор, обеспечивающий работу в режиме консультации;
- Загрузчик, обеспечивающий переключение режимов;
- Помощь, которая позволяет пользователю оперативно получать доступ к справочной информации в соответствии с текущим контекстом;

Базу знаний, которая может быть пустой (например, перед началом эксплуатации комплекса);

В процессе функционирования комплекса Редактор и Консультатор обмениваются управлением и данными. Обмен управлением производится при помощи Загрузчика, обмен данными осуществляется через Базу знаний. Помощь может быть вызвана в процессе диалога пользователя с Редактором и Консультатором в момент, когда управление находится у пользователя.

В основном варианте ОПС имеется также пакетный Консультатор (П-Консультатор), обеспечивающий решение задач в пакетном режиме.

1.5. Структура моделей проблемной области

Модель включает в себя набор атрибутов, определяющих множество допустимых состояний проблемной области, правила вывода, описывающие знания о методах решения задач, и сценарий консультации.

Для представления количественных характеристик проблемной области используются числовые атрибуты - переменные, определяемые на числовых интервалах. Например, возраст сотрудника может задаваться с помощью числового атрибута "возраст", определенного на интервале от 18 до 90 . Числовой атрибут во время решения задачи может иметь только одно значение.

Значения числовых атрибутов - действительные числа, содержащие не более двух знаков после десятичной точки и не превышающие по абсолютной величине 64000.00.

Качественные характеристики проблемной области описываются символьными атрибутами, которые определяются на конечных множествах значений, задаваемых перечислением. Например, рекомендуемый для местного обезболивания лекарственный препарат можно представить в виде символьного атрибута "анестетик", определенного на следующем множестве: "новокаин", "тримекаин", "лидокаин" и т.д. Высказывания типа "А есть В", где А - символьный атрибут, а В - элемент области определения этого атрибута, называются утверждениями. Значением утверждения является коэффициент его определенности. Вывод символьного атрибута, т.е. вычисление его значения, состоит в определении коэффициентов определенности утверждений, образованных из всех возможных значений этого атрибута. Допускаются множественные значения символьного атрибута, т.е. возможно сосуществование в какой-либо ситуации нескольких его значений. Например, символьный атрибут "иняз", задающий иностранные языки, которыми владеет специалист, может иметь одновременно несколько точно определенных значений.

Коэффициенты определенности утверждений - это действительные числа, принимающие значения от -5.00 до 5.00. Коэффициенту определенности D(H) утверждения H можно дать следующую содержательную интерпретацию:

если точно известно, что H истинно, то D(H) = 5.00;

если точно известно, что H ложно, то D(H) = -5.00;

если H может быть с одинаковой уверенностью истинно или ложно, то D(H) = 0.00;

если H скорее истинно, чем ложно, то 0.00 < D(H) < 5.00, причем D(H) тем больше, чем больше уверенность в истинности H;

если H скорее ложно, чем истинно, то -5.00 < D(H) < 0.00, причем D(H) тем меньше, чем больше уверенность в ложности H.

Утверждения и числовые атрибуты называются целями, асимвольные атрибуты, представляющие собой множества утверждений,называются сложными целями. Значения целей определяются с помощьюправил вывода. Правило вывода указывает, каким образом можно получить значение цели по значениям других атрибутов и утверждений, называемых подцелями правила. В зависимости от типа цели (простая или сложная) правила делятся на простые и сложные.

Правила могут иметь условия применимости - выражения, вычисляемые в момент обращения к правилу. Правило применяется только в том случае, если выполнено условие его применимости. Условие применимости считается выполненным, если значение соответствующего выражения больше 0.

Для определения значения одной цели разработчик экспертной системы может задавать несколько правил, образующих в модели упорядоченный список правил вывода данной цели. Порядок правил в списке отражает порядок их рассмотрения во время решения задачи.

Сложные правила преобразуются в результате компиляции в совокупности простых, выводящих значения всех утверждений о целевых

символьных атрибутах, соответствующих сложным правилам. Особенность таких простых правил состоит в том, что применение одного из них влечет за собой применение остальных.

Сеть вывода строится следующим образом. При вводе новой простой цели (нового числового атрибута или нового значения символьного атрибута) в сеть вывода вводится вершина, соответствующая этой цели.

При вводе простого правила, т.е. правила вывода значения простой цели, в сеть добавляется вершина, соответствующая этому правилу, и дуги, представляющие связи между простой целью и правилом, между правилом и его подцелями. При вводе сложного правила из него создается нескольких простых правил, предназначенных для вывода значений утверждений о сложной цели вводимого правила. Простые правила добавляются описанным ранее образом в сеть вывода.

В правилах представлены рекомендации относительно целесообразности направления пациента в стационар для проведения хирургического вмешательства под общим обезболиванием. Если травматичность предполагаемого вмешательства высокая, а риск высокий, то рекомендуется проведение общего обезболивания в стационаре. Если риск не является высоким или если травматичность вмешательства низкая, то в проведении общего обезболивания в условиях стационара нет необходимости. Риск считается высоким, если пациент страдает тяжелой формой бронхиальной астмы либо если ранее он перенес анафилактический шок.

Для задания модели проблемной области разработчик должен указать, какие вершины сети вывода являются целевыми в данной модели, какие сообщения необходимо выдать пользователю экспертной системы в зависимости от полученных результатов, а также другую информацию о порядке ведения консультации. Знания этого типа описываются в сценарии, являющимся обязательным компонентом модели. Сценарий представляет собой последовательность предложений, снабженных условиями

применимости. Каждое предложение указывает, какое действие должна выполнить экспертная система в случае применимости этого предложения. Возможно выполнение следующих действий:

- вывести значение цели;
- выдать сообщение пользователю;
- выдать сообщение внешней программе;
- сбросить выведенные результаты;
- перейти к выполнению другого предложения;
- принять информацию от внешней программы;
- внешней программе;
- передать информацию о результатах решения
- создать контрольную точку консультации;
- загрузить контрольную точку;
- обратиться к подмодели, решающей некоторую частную подзадачу, передать ей параметры и получить выведенные в подмодели результаты;
- закончить консультацию с сообщением (СТОП).

1.6. Стратегия решения задач

В начале решения задачи выбирается первое предложение сценария, проверяется условие его применимости и, если условие вы полнено, выполняется указанное в нем действие. Если в ходе проверки условия возникает потребность в значении некоторой цели, анализ условия приостанавливается, и требуемое значение выводится из сети вывода. Далее проверяется условие текущего предложения. Затем осуществляется переход к следующему предложению и т.д., пока не будет обнаружено действие СТОП или пока не будет исчерпан сценарий.

При выводе значения цели делается попытка в соответствии с описанием проблемной области найти в сети вывода путь от вершин, представляющих исходные данные консультации (т.е. значения, которые

вводятся пользователем в ответ на вопросы), к вершине, соответствующей рассматриваемой цели. Этот путь должен обладать следующим свой ством: условия применимости правил, соответствующих входящим в путь дугам, должны быть выполнены. При поиске пути используется метод поиска "в глубину" с последующим выводом значений от данных.

Алгоритм вывода значения простой цели состоит в следующем. Из списка правил вывода данной цели выбирается первое правило и делается попытка его применить (при этом проверяется условие его применимости, как это делается для предложений сценария). Если условие выполнилось, правило применяется. Если ни одно из правил не удалось применить, а цель - числовой атрибут, то в качестве значения этого атрибута принимается значение по умолчанию (в том случае, если оно указано разработчиком в описании атрибута), иначе цель помечается как невыводимая.

В том случае, когда оказалось сразу несколько применимых правил, используется первое применимое правило. Значения целей вычисляются один раз и не могут быть изменены иначе как по команде пользователя или с помощью предложения сценария СБРОС.

Исходные цели консультации задаются в сценарии; исходными являются все цели, упоминаемые в предложениях сценария. Остальные цели рассматриваются тогда, когда их значения понадобятся для применения каких-либо правил.

В режиме консультации пользователь может наблюдать ход рассуждения программы комплекса при поиске пути в сети вывода с помощью средств объяснения и трассировки.

1.7. Язык представления знаний

В описании языка представления знаний использованы следующие метасимволы и обозначения:

::= - символы, отделяющие левую часть определения от правой;

<> - скобки, в которые заключаются метапонятия; - символ, разделяющий альтернативы;

[] - скобки, в которые заключается необязательный фрагмент определяемой конструкции;

{} - скобки, обозначающие повторение (ноль или более раз) заключенной в скобки конструкции;

' - апостроф для выделения в языковых конструкциях символов языка представления знаний, совпадающих с метасимволами.

Числовые константы, допустимые в языке представления знаний, - это положительные и отрицательные числа с десятичной точкой и точностью до двух знаков после десятичной точки, не превышающие по абсолютной величине 64000.00. В описании синтаксиса языка представления знаний числовые константы обозначены метапонятием <число>.

Числовые константы, представляющие коэффициенты определенности, являются числовыми константами, не превышающими по абсолютной величине 5.00. В описании синтаксиса они обозначаются метапонятием <коэффициент определенности.

Числовые и символьные атрибуты, а также значения символьных атрибутов, идентифицируются с помощью имен. Именем может быть любая последовательность, состоящая не более чем из 30 символов, средикоторых могут быть:

- латинские прописные и строчные буквы;
- русские прописные и строчные буквы;
- цифры;
- символы: (подчеркивание), #. Имя не должно начинаться с цифры.

Это имя используется в описании модели как идентификатор, но с ним не всегда удобно работать в фазе использования системы при формировании сообщений пользователю, когда может возникнуть необходимость в развернутом имени атрибута, раскрывающем его семантику. Развернутые имена, представляющие лингвистические знания об атрибутах, задаются в

описаниях атрибутов. Длина развернутого имени не должна превышать 120 символов. Например, значение символьного атрибута "метод обезболивания" может иметь развернутое имя "местное обезболивание с премедикацией седуксеном и антигистаминным препаратом". В этом случае в сообщениях, выдаваемых пользователю в ходе консультации, будет использоваться это развернутое имя.

В описании синтаксиса языка представления знаний имена и развернутые имена определяются метапонятиями <имя> и <развернутое имя> соответственно .

Употребляемые в конструкциях языка представления знаний тексты (например, тексты комментариев)- это произвольные последовательности символов ограниченной длины, не содержащие символов-кавычек ("). Эти тексты обозначаются в описании синтаксиса как <текст N>, где N - число, указывающее максимально допустимую длину текста. В примерах конструкций, приведенных в настоящем документе, тексты и развернутые имена заключены в кавычки.

Для представления лингвистических знаний о коэффициентах определенности утверждений и значениях числовых атрибутов используются специальные конструкции, называемые шкалами. Шкалы представляют собой списки равенств вида: "<текст 40>=<число>", ставящих в соответствие лингвистическим (текстовым) описаниям некоторые числовые значения. Равенства в списке разделяются запятыми. Каждая шкала, описанная в модели относится к некоторому числовому атрибуту или утверждению. Например, лингвистические знания о коэффициенте определенности некоторого утверждения могут быть представлены с помощью шкалы:

"да=5, возможно=2.5, да или нет=0, маловероятно=-2.5, нет=-5".

Все числовые константы, упоминаемые в шкале, не должны выходить за пределы диапазона допустимых значений числового атрибута (если шкала относится к числовому атрибуту) или диапазона от -5.00 до 5.00 (если шкала

относится к утверждению). Шкалы могут задаваться как в описаниях атрибутов и утверждений, так и в текстах вопросов к ним.

Шкалы применяются при формировании текстов вопросов и сообщений конечному пользователю.

Шкала используется в вопросе в том случае, если она задана в тексте вопроса или в описании утверждения или атрибута, о котором задается вопрос. При этом из лингвистических описаний, содержащихся в шкале, формируется меню, и конечному пользователю предлагается выбрать одно из этих описаний. В качестве ответа на вопрос принимается то числовое значение, которое соответствует в шкале выбранному лингвистическому описанию. Например, если будет задан вопрос с приведенной выше шкалой, и пользователь выберет ответ "возможно", то в качестве ответа на вопрос будет принято значение 2.50.

Использование шкалы при формировании сообщения о вычисленном коэффициенте определенности утверждения или значении числового атрибута осуществляется в том случае, если в описании этого утверждения или числового атрибута задана шкала. При этом выдается то лингвистическое описание, которому соответствует число, наименее отличающееся от вычисленного значения. Например, при использовании приведенной выше шкалы, для коэффициента определенности 3.0 будет выбрано и помещено в текст сообщения лингвистическое описание "возможно".

В описании синтаксиса языка представления знаний шкалы определяются метапонятием <шкала>.

Формирование текстов вопросов и сообщений осуществляется с использованием шаблонов. Шаблон представляет собой текст, в который могут подставляться сведения об атрибутах и их значениях. Позиции для подстановки отмечаются в шаблоне следующим образом:

^^ - для коэффициентов определенности утверждений и значений числовых атрибутов (или соответствующих им лингвистических описаний, описанных в шкалах);

^s - для имен символьных атрибутов;

^v - для имен значений символьных атрибутов;

^g - для имен простых целей;

^y - для имен утверждений и неразвернутых имен числовых атрибутов.

При заполнении шаблонов предпочтение всегда отдается развернутым именам, если они указаны, и лингвистическим описаниям, если заданы соответствующие шкалы.

В описании синтаксиса языка представления знаний шаблоны определяются метапонятием <шаблон N>, где N - число, указывающее максимально допустимую длину текста шаблона.

Описание модели предметной области включает в себя комментарий к модели, способ разрешения конфликтов, описание атрибутов, правила вывода, а также сценарий консультации.

Синтаксис описания модели имеет следующий вид:

<описание модели> ::= [<комментарий к модели>]

<описание атрибута> { <описание атрибута> }

<описание правила вывода> { <описание правила вывода> }

<предложение сценария> { <предложение сценария> }

<комментарий к модели> ::= <текст 160>

Комментарий к модели представляет собой произвольный текст, длина которого не должна превышать 160 символов.

Для представления количественных и качественных характе

ристик проблемной области используются соответственно числовые и символьные атрибуты.

<описание атрибута> ::= <описание числового атрибута> |

<описание символьного атрибута>

Числовые атрибуты представляют собой величины, принимающие значения из некоторого диапазона. Значения числовых атрибутов считаются точно известными. Аналогом числового атрибута в традицион ном программировании является переменная вещественного типа, значе ние которой может быть получено только один раз и в дальнейшем не изменяется. Значения числовых атрибутов задаются с точностью до двух знаков после десятичной точки:

<описание числового атрибута> ::=

ЧИСЛОВОЙ АТРИБУТ <имя числового атрибута>

[<развернутое имя числового атрибута>]

[$<шкала>]

ОТ <число> ДО <число>

[ПО УМОЛЧАНИЮ <число>]

<имя числового атрибута> ::= <имя>

<развернутое имя числового атрибута> ::= <развернутое имя>

Задание развернутого имени атрибута не является обязательным. Если разработчик не укажет это имя, в сообщениях будет использоваться имя атрибута.

Описание шкалы начинается со специального символа '$'. Задание шкалы также не является обязательным. Если шкала есть, то при задании вопросов (выдаче сообщений) о значении числового атрибута значения будут запрашиваться (выдаваться) не в числовой форме, а терминах лингвистических описаний, упоминаемых в шкале.

Область допустимых значений числового атрибута описывается в виде диапазона, который задается двумя числовыми константами, представляющими его нижнюю и верхнюю границы. Область допустимых значений не должна выходить за пределы диапазона от -64000.00 до 64000.00.

Разработчик экспертной системы может задать значение числового атрибута по умолчанию (не обязательно). Это значение не должно быть

выходить за пределы области допустимых значений атрибута "ОТ <число> ДО <число>". Как указано значение по умолчанию принимается при решении задачи в качестве значения числового атрибута в том случае, когда его не удалось вывести другим способом.

Пример описания числового атрибута:

ЧИСЛОВОЙ АТРИБУТ возраст

"возраст пациента"

ОТ 15 ДО 100

ПО УМОЛЧАНИЮ

Символьные атрибуты принимают значения из конечного множества значений, задаваемого перечислением в описании атрибутов:

<описание символьного атрибута> ::=

СИМВОЛЬНЫЙ АТРИБУТ <имя символьного атрибута>

[<развернутое имя символьного атрибута>]

[$ [<шкала>]]

[ШАБЛОН <шаблон утверждений>]

<описание значения> { <описание значения> }

<имя символьного атрибута> ::= <имя>

<развернутое имя символьного атрибута> ::= <развернутое имя>

<шаблон утверждений> ::= <шаблон 180>

<описание значения> ::= ЗНАЧЕНИЕ <имя значения>

[<развернутое имя значения>]

[$ [<шкала>]]

[ИМЯ УТВЕРЖДЕНИЯ <имя>]

[ПО УМОЛЧАНИЮ <коэффициент определенности по умолчанию>]

<имя значения> ::= <имя>

<развернутое имя значения> ::= <развернутое имя>

<коэффициент определенности по умолчанию> ::=

<коэффициент определенности>

Задание развернутого имени атрибута не является обязательным. Если разработчик не укажет это имя, в сообщениях будет использоваться имя атрибута.

В тех случаях, когда всем значениям символьного атрибута соответствует одна и та же шкала, для сокращения записи эту шкалу можно указать один раз в описании символьного атрибута. Если указан только специальный символ '$', то будет использована шкала по умолчанию:

"Да=5, Скорее всего=4, Возможно=2, Да или нет=0, Наверное нет=-2, Маловероятно=-4, Нет=-5".

Шкала, задаваемая в описании значения символьного атрибута, относится только к данному значению. Как и в случае символьного атрибута, можно указать только символ '$', при этом будет использована шкала по умолчанию. Если указаны шкалы как для символьного атрибута, так и для некоторого его значения, то при работе с этим значением шкала для символьного атрибута игнорируется.

Шаблон утверждений используется при формировании сообщений о значениях данного атрибута. В шаблон могут подставляться: имя данного атрибута; имя того значения, сообщение о котором формируется. Если в модели не описан шаблон утверждения, используется стандартный шаблон "^s - ^v".

Имена утверждений могут использоваться в правилах и предложениях сценария как сокращенные ссылки на утверждения вида: "символьный атрибут <имя символьного атрибута> имеет значение <имя значения>". Полная ссылка на утверждение представляет собой пару имен (имя символьного атрибута и имя значения) в квадратных скобках, разделенных точкой.

Коэффициент определенности по умолчанию (или априорный коэффициент определенности) задается числовой константой в диапазоне от -5.00 до 5.00. Если коэффициент по умолчанию не указан в описании, он

будет считаться равным 0, что соответствует неопределенному значению истинности утверждения до начала консультации.

Пример описания символьного атрибута:

СИМВОЛЬНЫЙ АТРИБУТ рекомендация

$Да=5, Возможно=2, Да или нет=0, Маловероятно=-2, Нет=-5

ШАБЛОН " рекомендуемый лекарственный препарат - ^v"

ЗНАЧЕНИЕ нитросорбид

ИМЯ УТВЕРЖДЕНИЯ нитро1

ПО УМОЛЧАНИЮ 4.0

ЗНАЧЕНИЕ нитронг

ИМЯ УТВЕРЖДЕНИЯ нитро2

ПО УМОЛЧАНИЮ 0.0

В приведенном примере имя утверждения "нитро1" ссылается на утверждение с текстом "рекомендуемый лекарственный препарат - нитросорбид" и значением по умолчанию, равным 4.0. Полная ссылка на утверждение "нитро1" имеет вид: [рекомендация.нитросорбид].

Арифметические и логические выражения

Для описания действий правил, а также условий применимости правил и предложений сценариев используются арифметические и логические выражения.

Арифметические выражения используются как в правилах, для задания формул, по которым следует вычислять значения числовых атрибутов и утверждений, так и в логических выражениях.

Арифметические выражения имеют следующий синтаксис:

<арифметическое выражение> ::= <число>| <имя простой цели> |

<арифметическое выражение> + <арифметическое выражение>

<арифметическое выражение> - <арифметическое выражение>

<арифметическое выражение> * <арифметическое выражение>

<арифметическое выражение> / <арифметическое выражение>
<арифметическое выражение> % <арифметическое выражение>
(<арифметическое выражение>) | -<арифметическое выражение> | <имя арифметической функции> ({ <арифметическое выражение>, }
<арифметическое выражение>)

<имя простой цели> ::= <имя числового атрибута>| <ссылка на утверждение>

<имя числового атрибута> ::= <имя>

<ссылка на утверждение> ::= <имя утверждения>|
'['<имя символьного атрибута>.<имя значения>']'

<имя утверждения> ::= <имя>

<имя символьного атрибута> ::= <имя>

<имя значения> ::= <имя>

арифметической функции> ::= abs | exp | sq | lg | inth | intl | sin | cos | tg | ctg | asin | atg | rad | grad | <имя вспомогательной функции>

<имя вспомогательной функции> ::= more_cert | less_cert

Арифметические выражения строятся стандартным образом с помощью умножения скобок (*), деления (/), , а также следующих и операций сложения (+), вычитания (-) получения остатка от деления нацело (%) арифметических функций:

abs(X) - абсолютное значение X;

exp(X,Y) - число X в степени Y;

sq(X) – квадратный корень из X;

lg(X) – десятичный логарифм X;

inth(X) – округление X в большую сторону;

intl(X) - округление X в меньшую сторону;

sin(X) - синус X(X в радианах);

cos(X) - косинус X(X в радианах);

tg(X) - тангенс X(X в радианах);

ctg(X) - котангенс X(X в радианах);

asin(X) - арксинус X(X в радианах);

atg(X) - арктангенс X(X в радианах);

rad(X) - получение значения в радианах (X в градусах);

grad(X) - получение значения в градусах (X в радианах);

Определены следующие вспомогательные функции:

more_cert(X,A) - количество значений символьного атрибута A, коэффициенты определенности которых не ниже значения X;

less_cert(X,A) - количество значений символьного атрибута A, коэффициенты определенности которых не выше значения X, где значение X задается либо числовой константой, либо числовым атрибутом. Эти функции удобно использовать в тех случаях, когда необходимо оценить выведенные результаты. Например, при решении задачи диагностики можно определить, удалось ли выявить хотя бы несколько диагнозов с достаточной, с точки зрения эксперта, степенью уверенности: если поставлен хотя бы один диагноз, т.е. more_cert(4.0,диагноз) > 0 или все диагнозы отвергнуты, т.е. less_cert(-4.0,диагноз) = N, где N - число возможных диагнозов, то можно закончить решение задачи; иначе необходимо провести более подробный анализ с привлечением дополнительных данных. Другой распространенный случай использования этих функций - оценка результатов тестирования: сколько тестов дали положительный или отрицательный результат.

В арифметических выражениях можно использовать числовые константы и простые цели, т.е. числовые атрибуты и утверждения. Ссылаться на числовые атрибуты можно по именам (не развернутым), на утверждения - по именам утверждений или по парам вида: [<имя символьного атрибута>.<имя значения>].

Арифметические выражения вычисляются слева направо. Порядком выполнения операций можно управлять с помощью круглых скобок.

Пример арифметического выражения:

вес_пациента - (рост_пациента_в_см - 100.00)

В данном арифметическом выражении, исходя из значений числовых атрибутов "вес_пациента" и "рост_пациента_в_см", вычисляется величина избыточного веса пациента.

Логические выражения используются для записи высказываний о состоянии проблемной области - условий применимости правил вывода и предложений сценария. Значением логического выражения является коэффициент определенности соответствующего ему высказывания.

Логические выражения имеют следующий синтаксис:

<логическое выражение> ::= <коэффициент определенности> |

<имя простой цели> |

<арифметическое выражение> <отношение> <арифметическое выражение> |

<логическое выражение> '|' <логическое выражение> |

<логическое выражение> '||' <логическое выражение> |

<логическое выражение> & <логическое выражение> |

<логическое выражение> && <логическое выражение> |

(<логическое выражение>) | ~(<логическое выражение>) |

<имя логической функции> (<список параметров>)

(<логическое выражение>) ~<логическое выражение>

<отношение> ::= '=' | '>' | '<'

<имя логической функции> ::= AF | EF | AT | ET | UNC | UN | unc | un | eq |

gte | lte | <имя L-S функции>

<имя L-S функции> ::= L_ll | L_rr | L_lr | S_ll | S_rr | S_lr

<список параметров> ::= <имя символьного атрибута> |

<имя простой цели> | <арифметическое выражение>

{ , <арифметическое выражение> }

Логические выражения строятся из высказываний о проблемной области, значениями которых являются коэффициенты определенности, с помощью операций: конъюнкции (&), дизъюнкции (|) и отрицания (~).

В логических выражениях могут быть использованы операции: && и || (конъюнкция и дизъюнкция с принудительным завершением). Эти операции выполняются так же, как & и | соответственно, с единственным отличием: если первый операнд неположителен (в случае &) или положителен (в случае |), то второй операнд не рассматривается, и в качестве результата принимается значение первого операнда.

Как следует из приведенных определений, высказывания о проблемной области могут быть представлены:

- утверждениями;
- обращениями к логическим функциям;
- арифметическими отношениями: равно (=), больше (>) и меньше
- логическими выражениями.

Кроме того, можно задавать коэффициенты определенности с помощью числовых констант, числовых атрибутов и арифметических выражений, если их значения не выходят за пределы диапазона от -5.00 до +5.00.

При вычислении значений логических выражений используются следующие формулы:

D(H1 & H2) = min[D(H1), D(H2)]; (7)

D(H1 | H2) = max[D(H1), D(H2)]; (8)

D(~H) = -D(H), (9)

где H, H1 и H2 - высказывания о проблемной области (утверждения, обращения к логическим функциям и т.д.);

D(H) - коэффициент определенности высказывания H (т.е. его значение).

Параметры логических функций определены следующим образом:

- функции AF, EF, AT, ET, UN, UNC - имя символьного атрибута;
- функции un и unc - имя простой цели;
- параметры функций eq, gte, lte, а также L-S функций задаются с помощью арифметических выражений.

Для сокращения записи логических выражений в языке представления знаний определены 4 стандартные логические функции, вычисляющие коэффициенты определенности следующих высказываний:

AF(A) - все утверждения о всех возможных значениях атрибута A ложны (All False);

EF(A) - среди всех утверждений A есть ложные (Exist False);

AT(A) - все утверждения о всех возможных значениях символьного атрибута A истинны (All True);

ET(A) - среди всех утверждений о значениях символьного атрибута A есть истинные (Exist True).

Перечисленные логические функции могут быть представлены следую образом:

AF(A) = ~a1 & ~a2 & … & ~aN,

EF(A) = ~a1 | ~a2 | … | ~aN,

AT(A) = a1 & a2 & … & aN,

ET(A) = a1 | a2 & … | aN,

где a1, a2, . aN - утверждения о значениях символьного атрибута A. Следует отметить, что в функциях ET и EF рассматриваются только те из всех возможных утверждений об атрибуте A, коэффициенты определенности которых удалось вывести. Для вычисления значений функций AT и AF необходимо рассмотрение всех возможных утверждений.

L-S функции используются при описании нечетких понятий. Каждая L-S функция предоставляет эксперту стандартные средства для вычисления того коэффициента определенности, с которым значение числового атрибута удовлетворяет нечеткому понятию. Таким образом, эти функции обеспечивают параметрическое задание функций принадлежности нечетких понятий в терминах коэффициентов определенности.

Пример логического выражения:

(возраст > 40) & (бронх_астма > 2) & анафилактический_шок

При значении числового атрибута "возраст", равном 44, при коэффициенте определенности утверждения о наличии бронхиальной астмы, равном 3, и коэффициенте определенности утверждения об анафилактическом шоке, равном 4, получаем в качестве значения данного логического выражения 4.

Арифметические и логические выражения не должны содержать более 500 элементов (т.е. знаков операций, имен, запятых и круглых скобок).

С помощью правил вывода разработчик экспертной системы указывает, каким образом должны вычисляться значения атрибутов и утверждений. Числовой атрибут или утверждение, значение которого вычисляется с помощью некоторого правила, называется целью этого правила. Числовые атрибуты и утверждения являются простыми целями, а символьные атрибуты (т.е. множества утверждений) - сложными. В зависимости от целей все правила подразделяются на простые и сложные.

Для вычисления значения простой цели можно указывать несколько правил, которые образуют упорядоченный список правил вывода данной цели. Правила помещаются в список в том порядке, в каком они описаны в модели. Список правил используется в режиме консультации при попытке вывести значение данной цели.

Сложные правила в результате компиляции знаний преобразуются в совокупности простых, предназначенных для вывода утверждений о значениях соответствующего символьного атрибута. Особенность этих простых правил состоит в том, что применение одного из них в ходе решения задачи влечет за собой применение всех остальных.

В языке представления знаний определены следующие типы правил:

- простой вопрос;
- сложный альтернативный вопрос;
- сложный дистрибутивный вопрос;
- арифметическое правило;
- логическое правило;

- байесовское правило.

Правила-вопросы позволяют запрашивать данные о конкретной ситуации в проблемной области - исходные данные консультации. Возможны два способа ввода этих данных: посредством задания вопроса пользователю или обмена информацией с внешними программами. Выбор способа ввода данных определяется разработчиком ЭС.

Простой вопрос позволяет получать либо значение числового атрибута, либо коэффициент определенности отдельного утверждения. Сложный вопрос позволяет получить распределение коэффициентов определенности по всем возможным значениям символьного атрибута. Альтернативный вопрос используется в тех случаях, когда известно, что символьный атрибут имеет точно одно значение из множества возможных значений. Дистрибутивный вопрос используется в тех случаях, когда символьный атрибут может иметь одновременно несколько или ни одного значений.

Арифметические, логические и байесовские правила относятся к простым правилам.

Арифметические правила предназначены для вычисления значений числовых атрибутов, а также для получения коэффициентов определенности утверждений по эвристическим формулам, предложенным экспертами.

Логические правила предназначены для вычисления коэффициентов определенности утверждений по формулам нечеткой логики. Они применяются только в случае выполнения условия применимости, при этом значение соответствующего логического выражения становится значением утверждения - цели правила.

Байесовские правила применяются для вычисления коэффициентов определенности тех утверждений, об истинности которых можно судить по выполнению ряда факторов (симптомов), имеющих разную значимость.

Синтаксис описания правила имеет следующий вид:

<описание правила> ::=

ПРАВИЛО <имя правила> [<комментарий>]

ЦЕЛЬ <имя цели>

ЕСЛИ [<логическое выражение>]

ТО <действие правила> правила>

<имя правила> ::= <имя простого вопроса>|

<имя сложного альтернативного вопроса> |

<имя сложного дистрибутивного вопроса> |

<имя арифметического правила> |

<имя логического правила> |

<имя байесовского правила>

<имя простого вопроса> ::= Q <текст 19> | П <текст 19>

<имя сложного альтернативного вопроса> ::= A <текст 19> | А <текст 19>

<имя сложного дистрибутивного вопроса> ::= D <текст 19> | Д <текст 19>

<имя арифметического правила> ::= N <текст 19> | Р <текст 19>

<имя логического правила> ::= L <текст 19> | Л <текст 19>

<имя байесовского правила> ::= B <текст 19> | Б <текст 19>

<комментарий> ::= <текст 300>

<имя цели> ::= <имя символьного атрибута> | <имя простой цели>

Именами правил могут быть произвольные последовательности сим волов длиной до 20 байт. Имена правил должны быть уникальными. Пер вый символ имени идентифицирует тип правила.

Комментарии к правилам - это произвольные тексты длиной до 300 байт, которые используются в режиме консультации для формирования объяснений.

Целью простого правила может быть числовой атрибут или утверждение, сложного - символьный атрибут.

Условие применимости правила - это логическое выражение, значение которого вычисляется в момент обращения к правилу.

Действие правила указывает, каким образом должно вычисляться значение цели правила в том случае, когда правило применимо. Вид действия правила определяется типом правила.

В случае простого вопроса действие правила имеет вид:

<действие правила> ::= [<шаблон>] [$ [<шкала>] | @ <текст 299>

Действием для правила типа "простой вопрос" является текст сообщения пользователю или внешней программе. Текст сообщения пользователю формируется с помощью шаблона, в который могут подставляться: имя символьного атрибута, имя значения этого атрибута и имя утверждения (составленного с помощью шаблона утверждений для данного атрибута), если вопрос относится к утверждению; имя числового атрибута, если вопрос относится к числовому атрибуту. Если текст не указан, то в режиме консультации он формулируется автоматически. Стандартный текст вопроса о значении числового атрибута имеет вид: "Каково значение атрибута ^g ?", При построении вопроса об утверждении текст вопроса имеет вид "Насколько Вы уверены в том, что ^g ?", В простом вопросе можно указать шкалу, в соответствии с которой пользователь может вводить ответ не в числовой форме, а с помощью некоторых лингвистических описаний (см. п. 5.1.5). Шкала отделяется от текста основного вопроса символом '$'. В случае утверждений может быть использована стандартная шкала: для этого достаточно указать символ '$' без задания самой шкалы. Если для утверждения или числового атрибута, о котором задается вопрос, уже определена шкала, она будет проигнорирована.

Если для ответа на вопрос требуется вызвать внешнюю программу, в качестве первого символа текста сообщения следует поставить специальный символ '@', тогда остальной текст будет интерпретироваться как командная строка.

Примеры правил типа простой вопрос:

ПРАВИЛО Q1

Возраст указывается во время консультации"

ЦЕЛЬ возраст

ЕСЛИ

ТО Каков возраст пациента ?

При применении данного правила для вычисления значения цели (числового атрибута) "возраст", пользователю будет задан вопрос "Каков возраст пациента ? (Введите число в диапазоне от 15.00 до 100.00)". Если пользователь введет число в указанном диапазоне, то это число станет значением атрибута "возраст"; если пользователь сообщит, что не знает ответа, будет считаться, что данное правило не удалось применить.

ПРАВИЛО Q2

ЦЕЛЬ риск

ЕСЛИ

ТО Оцените степень риска предполагаемого вмешательства:

$высокая=5, средняя=2.5, невысокая=0, низкая=-2.5

При применении данного правила пользователю будет задан вопрос "Оцените степень риска предполагаемого вмешательства:" и предложено меню, включающее следующие варианты ответа: высокая, средняя, невысокая и низкая. Если, например, пользователь выберет второй вариант, то коэффициенту определенности утверждения с именем "риск" будет присвоено значение 2.5.

ПРАВИЛО П_зона

ЦЕЛЬ далеко

ЕСЛИ

ТО @gq CGA way.zon

При применении данного правила будет выдана команда операционной системы "gq CGA way.zon", после выполнения которой "ответ" должен находиться в файле EKO.ANS в символьном виде, например, "V:*:4.5".

В случае сложного альтернативного или дистрибутивного вопроса действие правила имеет вид:

<действие правила> ::= [<шаблон>] [$ [<шкала>] | @ <текст 299>

Действие для правил типа "сложный альтернативный вопрос" и "сложный дистрибутивный вопрос" определяются так же, как для простых вопросов. Если шаблон сообщения пользователю опущен, то используется стандартный шаблон: "Каково значение атрибута ^s ?". Шкалы могут использоваться только в дистрибутивных вопросах. Помимо текста вопроса пользователю во время консультации будет выдан список развернутых имен значений атрибута. Если вопрос альтернативный, пользователю предлагается выбрать одно из значений. После того как пользователь укажет выбранный вариант, соответствующее утверждение получает значение 5.00, а утверждения обо всех остальных значениях символьного атрибута - значение -5.00. Если вопрос дистрибутивный, пользователю предлагается задать коэффициенты определенности для значений из списка. Если пользователь не укажет коэффициенты для некоторых значений, будет сделана попытка вывести значения соответствующих утверждений (когда в них будет возникать потребность) с помощью других правил вывода, если они есть.

Передача данных от внешней программы в ЭС производится через файл EKO.ANS в соответствии с требованиями.

Примеры правил типа сложный вопрос (альтернативный и дистрибутивный):

```
ПРАВИЛО     A_зона
ЦЕЛЬ        место
ЕСЛИ
ТО          @gq CGA c.zon
```

При применении данного правила будет выдана команда операционной системы "gq CGA c.zon", после выполнения которой "ответ" должен находиться в файле EKO.ANS в символьном виде. Например, если атрибут "место" имеет значения "далеко" и "близко", а файл EKO.ANS содержит "V:2", то утверждение [место.далеко] получит значение -5.00, а [место.близко] - значение 5.00.

```
ПРАВИЛО     D1
```

"Ввод информации о проявлениях аллергии"

ЦЕЛЬ проявления_аллергии

ЕСЛИ

ТО Наблюдались ли раньше какие-либо проявления аллергии ? $ никогда=-4, редко=1, часто=3, есть заболевания=5

Попытка применить данное правило будет делаться в том случае, когда возникнет потребность в коэффициенте определенности какого-либо из утверждений о значении символьного атрибута "проявления_аллергии". При этом на экран будет выдан текст вопроса, список развернутых имен значений данного символьного атрибута, а также шкала для ответа. Если для утверждений о символьном атрибуте "проявления_аллергии" уже определены шкалы, они будут проигнорированы.

Для каждого значения пользователь должен ввести ответ с помощью шкалы и/или в виде числового коэффициента определенности. Если пользователь не сможет указать коэффициенты для некоторых значений, то, в случае возникновения необходимости в них, будут использованы другие правила (если они есть в модели).

В случае арифметического правила действие имеет вид:

<действие правила> ::= <арифметическое выражение>

Действием для арифметического правила должно быть арифметическое выражение, значение которого, в случае когда правило применимо и все упоминаемые в выражении подцели выведены, присваивается цели правила. Если вычисленная с помощью данного выражения величина выходит за пределы допустимого для цели диапазона, в качестве значения этой цели принимается соответствующая граница диапазона.

Примеры арифметических правил:

ПРАВИЛО назначение дозы

"Учет избыточного веса пациента при назначении дозы лекарственного препарата X"

ЦЕЛЬ доза_X

ЕСЛИ

ТО вес / 10 - (вес / 100 - норма * рост)

Применение этого правила для вычисления значения числового атрибута "доза_X", приведет к выводу значений числовых атрибутов "вес", "норма" и "рост", после чего вычислится значение арифметического выражения из действия правила. Полученное значение присваивается целевому числовому атрибуту "доза_X".

ПРАВИЛО Nr1

"Вычисление коэффициента определенности утверждения о том, что сотрудник живет далеко от работы, по эвристической формуле"

ЦЕЛЬ [место_работы.расположено_далеко]

ЕСЛИ

ТО 0.2 * время_на_дорогу - 9

Здесь "время на дорогу" - числовой атрибут, указывающий время на дорогу в минутах. При применении данного правила вычислится значение этого числового атрибута, а затем - значение арифметического выражения из действия правила. Это значение будет рассматриваться как коэффициент определенности утверждения о том, что место работы расположено далеко от дома сотрудника.

В случае логического правила действие имеет вид:

<действие правила> ::= [<арифметическое выражение>]

Действием в логических правилах может быть арифметическое выражение. Значение этого выражения не должно превышать по абсолютной величине 1.00. Это выражение называется коэффициентом определенности правила и используется в случае отсутствия полной уверенности эксперта в надежности данного правила.

Задание действия логического правила не обязательно. Если разработчик опускает действие, то по умолчанию коэффициент определенности правила принимается равным 1.0. Если в результате вычисления значения действия правила получается недопустимая величина, то в качестве коэффициента определенности правила принимается:

1.00 - в случае положительного результата;

-1.00 - в случае отрицательного результата.

В отличие от всех остальных правил задание условия применимости для логических правил является обязательным.

Применение логического правила осуществляется следующим образом: вычисляется условие применимости, и если оно положительно, то его значение, умноженное на коэффициент определенности правила, принимается в качестве значения цели правила; в противном случае правило считается неприменимым.

Пример логического правила:

ПРАВИЛО	L1 "Как правило, у больных, перенесших анафилактический шок, ранее бывали другие проявления аллергии; поэтому если в данном случае других проявлений не было, то с большой уверенностью можно считать, что обмороки при проведении местной анестезии не связаны с анафилактическим шоком"
ЦЕЛЬ	анафилактический_шок
ЕСЛИ	[осложнения.обмороки] & AF (проявления_аллергии)
ТО	0.90

Здесь "проявление_аллергии" - имя символьного атрибута, принимающего следующие значения: "бронхиальная_астма", "отек_Квинке", "дерматит", "зуд_кожи". При применении данного правила, выведется значение утверждения о наличии осложнений в виде обмороков, а также

утверждений обо всех возможных проявлениях аллергии, а затем вычислится значение логического выражения из условия применимости. Если оно положительно, то правило применимо; полученное значение, умноженное на 0.90, будет рассматриваться как коэффициент определенности того, что больной действительно перенес анафилактический шок.

В случае байесовского правила действие имеет вид:

<действие правила>:= <описание фактора> {<описание фактора>}

<описание фактора> := <ссылка на утверждение> <разделитель>
<вес> <разделитель> <вес> <разделитель>

<разделитель> ::= <пробелы> | <запятая>

<вес> ::= <число> | <имя числового атрибута>

Действие байесовского правила задается в виде таблицы с тремя столбцами. В первом столбце перечисляются утверждения-факторы, коэффициенты определенности которых влияют на коэффициенты определенности целевого утверждения.

Во втором столбце указываются подтверждающие, а в третьем - опровергающие веса факторов. Подтверждающий вес утверждения-фактора должен быть равен такому коэффициенту определенности, который, по мнению эксперта, следует приписать целевому утверждению в случае истинности утверждения-фактора. Опровергающий вес равен коэффициенту, приписываемому целевому утверждению экспертом в случае ложности утверждения-фактора.

При работе с байесовскими правилами коэффициент определенности D(H) высказывания H связывается с вероятностью того, что это высказывание истинно в рассматриваемой ситуации в проблемной области, следующим образом:

$$D(H) = 10\,P(H) - 5,$$

где D(H) - коэффициент определенности H;

P(H) вероятность истинности H.

Коэффициент определенности целевого утверждения байесовского правила вычисляется следующим образом. Если подтверждающий вес некоторого фактора равен 5.0 (-5.0), т.е. наличие фактора точно свидетельствует в пользу (против) целевого утверждения, а коэффициент определенности этого фактора, вычисленный в ходе решения задачи, равен 5.0, то целевое утверждение считается точно истинным (точно ложным). Если опровергающий вес некоторого фактора равен 5.0 (-5.0), т.е. отсутствие фактора свидетельствует в пользу (против) целевого утверждения, а вычисленный коэффициент определенности этого фактора равен -5.0, то целевое утверждение считается точно истинным (точно ложным). В противном случае вычисляются величины, представляющие свидетельства факторов в пользу или против целевого утверждения, а затем, для получения коэффициента определенности целевого утверждения, происходит объединение полученных свидетельств. При вычислении свидетельств используются следующие формулы:

$$De(Hi) = \begin{cases} Danp(G) + \dfrac{Ai - Danp(G)}{5 - Danp(Hi)} * (D(Hi) - Danp(Hi)), \\ \qquad\qquad \text{если } D(Hi) >= Danp(Hi); \\ Danp(G) + \dfrac{Danp(G) - Bi}{5 + Danp(Hi)} * (D(Hi) - Danp(Hi)), \\ \qquad\qquad \text{если } D(Hi) < Danp(Hi), \end{cases}$$

где De(Hi) – свидетельство i-го фактора;

Danp(G) - априорное значение коэффициента определенности целевого утверждения;

Danp(Hi) - априорное значение коэффициента определенности фактора Hi;

D(Hi) - текущее значение коэффициента определенности i-го фактора;

Ai - подтверждающий вес фактора i-го фактора;

Bi - опровергающий вес фактора i-го фактора.

Свидетельства объединяются следующим образом:

$$D(G/H1,H2,\ldots,HM) = 5 * \frac{De - 1}{De + 1},$$

где

$$De = \frac{5 + Danp(G)}{5 - Danp(G)} * I(H1) * I(H2) * \ldots * I(HM);$$

$$I(Hi) = \frac{5 - Danp(G)}{5 + Danp(G)} * \frac{5 + De(Hi)}{5 - De(Hi)}.$$

Пример байесовского правила:

ПРАВИЛО	"B1"		
	"Вычисление коэффициента определенности гипотезы об остеомиелите челюстей"		
ЦЕЛЬ	остеомиелит_челюстей		
ЕСЛИ			
ТО	сильные_боли	2.00	-1.00
	сопутствующие_заболевания	тяжесть	0.00
	симптом_Венсана	4.50	-3.50
	обширное_воспаление	4.00	-4.00

Пусть априорный коэффициент определенности утверждения "остеомиелит_челюстей" равен 0, а выведенное в ходе консультации значение числового атрибута "тяжесть" равно 2.50.

Пусть также удалось установить, что

D (сильные_боли) = 3.00;

D (сопутствующие_заболевания) = 5.00;

D (симптом_Венсана) = 0.00;

D (обширное_воспаление) = -5.00.

Предположим, что в модели указан априорный коэффициент определенности утверждения "сильные_боли", равный 0.00. В этом случае :

$$De(\text{сильные\_боли}) = 0.00 + \frac{2.00 - 0.00}{5 - 0.00} * (3.00 - 0.00) = 1.20;$$

$$De(\text{сопутствующие\_заболевания}) = 0.00 + \frac{2.50 - 0.00}{5 - 0.00} * (5.00 - 0.00) = 2.50;$$

$$De(\text{симптом\_Венсана}) = 0.00 + \frac{4.50 - 0.00}{5 - 0.00} * (0.00 - 0.00) = 0.00;$$

$$De(\text{обширное\_воспаление}) = 0.00 + \frac{0.00 - (-4.00)}{5 - 0.00} * (-5.00 - 0.00) = -4.00;$$

$$I(\text{сильные\_боли}) = \frac{5 - 0.00}{5 + 0.00} * \frac{5 + 1.20}{5 - 1.20} = 1.632;$$

$$I(\text{сопутствующие\_заболевания}) = \frac{5 - 0.00}{5 + 0.00} * \frac{5 + 2.50}{5 - 2.50} = 3.00;$$

$$I(\text{симптом\_Венсана}) = \frac{5 - 0.00}{5 + 0.00} * \frac{5 + 0.00}{5 - 0.00} = 1.00;$$

$$I(\text{обширное\_воспаление}) = \frac{5 - 0.00}{5 + 0.00} * \frac{5 - 4.00}{5 + 4.00} = 0.111;$$

$$De = 1.632 * 3.00 * 1.00 * 0.111 = 0.543;$$

$$D(\text{остеомиелит\_челюстей}) = 5 * \frac{0.543 - 1.00}{0.543 + 1.00} = -1.48.$$

Описание сценария консультации

Сценарий консультации представляет собой последовательность предложений, определяющую порядок действий при решении задачи.

Синтаксис описания предложения сценария имеет следующий вид:

<предложение сценария> ::=

<имя предложения> <имя действия> <параметры действия>

[ЕСЛИ <условие применимости предложения>]

[<комментарий>]

<имя предложения> ::= <имя>

<имя действия> ::= ЦЕЛЬ | РЕЗУЛЬТАТ | СООБЩЕНИЕ | ПЕРЕХОД |
СБРОС | СБРОС_ВЫВОДА | СТОП | ПРИЕМ | ВОЗВРАТ |
ВЫПОЛНИТЬ | ЗАДАЧА | АРГУМЕНТЫ |
ЗАГРУЗИТЬ_ТОЧКУ | СДЕЛАТЬ_ТОЧКУ | ВЫХОД |
СДЕЛАТЬ_ТОЧКУ_ВЫХОД | НАСТРОЙКА

<условие применимости предложения> ::= <логическое выражение> |
<ссылка на условие предыдущего действия>
<ссылка на условие предыдущего действия> ::= "-"
<комментарий> ::= <текст 300>

Именем предложения может быть любая последовательность, содержащая до 20 символов, без пробелов. Например, в качестве имени можно использовать номер предложения.

Условие применимости можно задавать либо так же, как в правилах либо в виде ссылки на условие предыдущего действия. В первом случае проверка условия применимости осуществляется аналогично проверке условий в правилах. Во втором случае действие выполняется только тогда, когда выполнилось условие применимости предыдущего предложения, при этом считается, что условие применимости данного предложения также выполнилось.

Вид параметров действия зависит от типа действия.

В шаблонах сообщений, используемых в сценарии, предусмотрена дополнительная возможность подстановки имени наиболее вероятного значения символьного атрибута: из всех значений выбирается то, у которого коэффициент определенности наибольший, а если таких значений несколько, то выбирается первый по порядку. Для задания такой ссылки необходимо в символах, отмечающих позицию для подстановки, поставить знак '!' (восклицательный знак) после '^', а в списке имен для подстановки указать имя символьного атрибута.

Параметры действия ЦЕЛЬ определяются следующим образом:

<параметры действия ЦЕЛЬ> ::= { <имя цели> , } <имя цели>

Параметры действия ЦЕЛЬ задают в виде списка имен, разделенных запятыми или пробелами, исходные цели консультации, а также порядок их рассмотрения. Параметрами могут быть утверждения, числовые и символьные атрибуты.

Пример предложения сценария:

3 ЦЕЛЬ ДИАГНОЗ

ЕСЛИ [задача.диагностика]

"Если требуется решить задачу диагностики, вывести значения символьного атрибута ДИАГНОЗ"

При выполнении данного предложения в сценарии сначала будет определено значение утверждения [задача.диагностика], и, если оно окажется положительным, будет выводиться значение символьного атрибута ДИАГНОЗ, т.е. будут определяться коэффициенты определенности утверждений обо всех значениях данного атрибута.

Параметры действия РЕЗУЛЬТАТ определяются следующим образом:

<параметры действия РЕЗУЛЬТАТ> ::= [<шаблон 1600>,]<уровень>,

{ <имя цели> , } <имя цели>

<уровень> ::= <коэффициент определенности | <имя простой цели>

<имя цели> ::= <имя простой цели> | <имя символьного атрибута>

Действие РЕЗУЛЬТАТ задает список целей, сообщения о значении которых должны быть выданы на экран, а также уровень выдачи результатов. Сообщения будут выдаваться только о тех утверждениях, коэффициенты определенности которых не ниже уровня выдачи. Сообщения выдаются в виде списка развернутых имен целей и их значений. Если параметры действия содержат символьный атрибут, то утверждения о данном атрибуте при выдаче упорядочиваются по убыванию коэффициентов определенности. Утверждения о значениях символьного атрибута строятся исходя из шаблона утверждений и развернутых имен.

Сообщения о результате могут выдаваться по шаблону выдачи, задаваемому в качестве первого (необязательного) параметра. Если шаблон выдачи не указан, используется стандартный: ¤wb^g: ^^¤ .

Перед выдачей сообщения просматривается список целей-параметров выводятся те из них, значения которых еще не были получены.

Пример предложения сценария:

1 РЕЗУЛЬТАТ -3.00, анестетик

"Выбор средства обезболивания"

При выполнении этого действия будут вычислены значения символьного атрибута "анестетик" (предложение является первым в сценарии, поэтому никакие значения еще не выведены), а затем на экран будут выданы сообщения о тех значениях атрибута "анестетик", коэффициенты определенности которых не ниже -3.00. Сообщения будут упорядочены по убыванию их коэффициентов определенности. Например, если выведено:

[анестетик.новокаин] = -5.00,

[анестетик.тримекаин] = 5.00,

[анестетик.лидокаин] = 2.00,

и шаблон утверждений для символьного атрибута "анестетик" имеет вид "рекомендуемый анестетик - ^v", причем развернутые имена значений атрибута в модели не указаны, то на экран будет выведено следующее:

рекомендуемый анестетик - тримекаин: 5.00

рекомендуемый анестетик - лидокаин : 2.00

Это же предложение можно задать с шаблоном выдачи:

1 РЕЗУЛЬТАТ "с определенностью ^^ рекомендуется: ^v", -3, анестетик "Выбор средства обезболивания"

В этом случае на экран выведется:

с определенностью 5.00 рекомендуется: тримекаин

с определенностью 2.00 рекомендуется: лидокаин

Параметры действия СООБЩЕНИЕ определяются следующим образом:

<параметры действия СООБЩЕНИЕ> ::= <шаблон 1600>{,<имя простой цели>}

Действие СООБЩЕНИЕ, как и действие РЕЗУЛЬТАТ, предназначено для описания сообщений конечному пользователю, однако в действиях

данного типа разработчик может указывать произвольный текст сообщения. Текст задается с помощью шаблона.

Первым параметром действия является шаблон сообщения, за ним должны быть указаны простые цели в том порядке, в каком их значения должны подставляться в сообщение. Если к моменту выполнения действия какие-либо из требуемых значений не будут известны, то сначала будет произведен вывод их значений и только после этого будет выдано сообщение. Если для каких-либо простых целей в списке параметров определены шкалы, то выдача их значений будет осуществляться в символьном виде в соответствии с этими шкалами.

Если требуется вывести сообщение не на экран монитора, а передать внешней программе (через файл EKO.MSG), то перед текстом сообщения ставится специальный символ '$'.

Если после выдачи текста сообщения требуется приостановить консультацию (сделать паузу), то перед текстом сообщения ставится специальный символ '@'. Эта возможность используется для того, чтобы обратить на сообщение внимание пользователя экспертной системы. В этом случае для продолжения консультации следует нажать любую клавишу (Все сказанное справедливо и по отношению к действиям СТОП и РЕЗУЛЬТАТ).

Пример предложения сценария:

5 СООБЩЕНИЕ

"Для проведения местной анестезии (уверенность ^^) можно использовать

^!v (с уверенностью ^!^)",

[метод обезболивания.местная анестезия], анестетик, анестетик

ЕСЛИ [метод обезболивания.местная анестезия]

"Сообщение о наиболее подходящем анестетике для проведения местной анестезии"

При выполнении данного действия будет выведено, если это не было сделано раньше, значение утверждения о возможности проведения местной

анестезии и, если оно окажется положительным, будет выполнен переход к определению значений утверждений о возможности применения в рассматриваемой ситуации всех описанных в системе местных анестетиков (например, тримекаина и лидокаина). Пусть значение утверждения о возможности местной анестезии будет равно 2.00, а значения утверждений о возможности применения тримекаина и лидокаина -5.00 и 5.00 соответственно. Тогда на экран будет выдано следующее сообщение:

"Для проведения местной анестезии (уверенность 2.00) можно использовать

лидокаин (с уверенностью 5.00)" .

Параметр действия ПЕРЕХОД определяется следующим образом:

<параметры действия ПЕРЕХОД> ::= <имя предложения>

Действие ПЕРЕХОД позволяет осуществлять условный переход в сценарии. Параметром действия является имя предложения, к которому необходимо перейти.

Пример предложения сценария:

11 ПЕРЕХОД 18

ЕСЛИ ~[метод обезболивания.общее обезболивание]

При выполнении данного предложения будет определено значение утверждения о возможности проведения общего обезболивания, и если оно окажется отрицательным, будет сделан переход к предложению с именем 18 (в примере имена предложений совпадают с их номерами, что необязательно).

Параметр действия СБРОС определяется следующим образом:

<параметры действия СБРОС> ::= [{ <имя цели> , } <имя цели>]

Действие СБРОС предназначено для отмены тех значений, которые указаны в качестве параметров этого действия. Если параметры не указаны, происходит отмена всех значений.

Пример фрагмента сценария (повторение консультации с новыми исходными данными):

```
Повторить    СБРОС
             ЕСЛИ      повторить_решение
Цикл         ПЕРЕХОД   начало
             ЕСЛИ      -"-
```

При выполнении предложения "повторить" будет выведено значение утверждения о необходимости повторить решение задачи (для рассмотрения новой ситуации). Если значение этого утверждения положительно, произойдет отмена всех значений, полученных во время консультации. Затем будет выполняться предложение сценария с именем "начало".

Действие СБРОС ВЫВОДА предназначено для отмены всех значений, кроме тех, которые были введены пользователем в ответ на вопросы. Это действие не требует параметров.

Пример фрагмента сценария (повторение консультации с изменением одного исходного данного):

```
Повторить    СБРОС_ВЫВОДА
             ЕСЛИ      повторить решение
Отмена       СБРОС     возраст
             ЕСЛИ      -"-
Цикл         ПЕРЕХОД   начало
             ЕСЛИ      -"-
```

В том случае, когда надо повторить решение, будут отменены все значения, за исключением исходных данных консультации, а также значение исходного данного "возраст", после чего будет повторено решение задачи; это позволит определить влияние возраста пациента на результат консультации.

Параметры действия СТОП определяются следующим образом:

<параметры действия СТОП> ::= [<шаблон 1600> {, <имя простой цели> }]

Действие СТОП предназначено для принудительного прекращения решения задачи. В качестве параметров действия можно задать описание заключительного сообщения. Параметры данного действия задаются так же, как и в случае действия СООБЩЕНИЕ. Если параметры не заданы, заключительное сообщение не выдается. Решение задачи заканчивается также тогда, когда достигнут конец сценария (нет следующего предложения).

Пример предложения сценария:

```
18      СТОП
        ЕСЛИ        ~повторить_решение
```

Параметры действия ПРИЕМ определяются следующим образом:

<параметры действия ПРИЕМ> ::= <имя файла>[,<нижняя граница>,
<верхняя граница>]

<нижняя граница> ::= <число>

<верхняя граница>::= <число>

Действие ПРИЕМ предназначено для приема исходных данных из файла входных данных, имя которого задается в качестве параметра действия. Файл входных данных может быть подготовлен внешней программой или вручную. Информация, содержащаяся в файле входных данных, представляет собой текст. Каждая строка этого текста задает значение, которое следует присвоить некоторой цели. Строка определяется следующим образом:

<строка файла входных данных> ::= <имя простой цели>=<значение>

<значение> ::= <число>

Если диапазон значений коэффициентов во внешнем файле отличается от [-5.0, 5.0] (например, используется диапазон [0.0, 1.0] или [-100.0, 100.0]), то при выполнении действия пропорциональное преобразование коэффициентов может осуществляться автоматически - для этого достаточно

в качестве второго и третьего параметров этих действий указать нижнюю и верхнюю границы диапазона, используемого во внешнем файле.

При выполнении данного действия каждой цели, указанной в строке файла входных данных и к настоящему моменту не выведенной, присваивается соответствующее значение. Если значение выходит за пределы допустимого диапазона для данной цели, оно игнорируется. Строки с неверно заданными именами простых целей не обрабатываются.

Пример предложения сценария:

6 ПРИЕМ "EXAMPLE.INP"

ЕСЛИ [метод_обезболивания.местная_анестезия]

Пример файла входных данных EXAMPLE.INP:

[анестетик.лидокаин] = 5.00

[анестетик.тримекаин] = -3.50

Если во внешнем файле использован диапазон от -100 до 100, то это предложение следует записать следующим образом:

6 ПРИЕМ "EXAMPLE.INP",-100, 100

ЕСЛИ [метод_обезболивания.местная_анестезия]

Пример файла входных данных EXAMPLE.INP:

[анестетик.лидокаин] = 100.00

[анестетик.тримекаин] = -60.00

Параметры действия ВОЗВРАТ определяются следующим обра-

<параметры действия ВОЗВРАТ> ::= <имя файла>, <уровень>
[,<нижняя граница>,
<верхняя граница>],
<имя простой цели>
{,<имя простой цели>}

<нижняя граница> ::= <число>

<верхняя граница>::= <число>

<имя простой цели> ::= <имя числового атрибута>

<ссылка на утверждение>

<имя числового атрибута> ::= <имя>

<ссылка на утверждение> ::= <имя утверждения> |

'['<имя символьного атрибута>.<имя значения>']'

<имя утверждения> ::= <имя>

<имя символьного атрибута> ::= <имя>

<имя значения> ::= <имя>

Действие ВОЗВРАТ аналогично действию РЕЗУЛЬТАТ. Разница заключается в следующем:

1) результаты решения задачи выдаются не на экран дисплея, а в файл выходных данных, имя которого задается в параметрах действия.

2) вместо развернутых имен целей всегда используются имена целей.

3) если диапазон значений коэффициентов во внешнем файле отличается от [-5.0, 5.0] (например, используется диапазон [0.0, 1.0] или [-100.0, 100.0]), то при выполнении действия пропорциональное преобразование коэффициентов может осуществляться автоматически -для этого достаточно в качестве второго и третьего параметров этих действий указать нижнюю и верхнюю границы диапазона, используемого во внешнем файле.

Файл выходных данных имеет формат, полностью аналогичный формату файла входных данных.

Пример предложения сценария:

10 ВОЗВРАТ "DIAGN.OUT",0,диагноз

Если значениями атрибута "диагноз" являются "аллергия" и "в дистония", и коэффициенты соответствующих утверждений равны 0.80 и 0.50 соответственно, то после выполнения указанного предложения файл DIAGN.OUT будет содержать следующую информацию:

[диагноз.аллергия]=0.80

[диагноз.в дистония]=0.50

Параметры действия ВЫПОЛНИТЬ определяются следующим образом:

<параметры действия ВЫПОЛНИТЬ> ::= <шаблон 1600> [{,<имя простой цели>}]

Действие ВЫПОЛНИТЬ предназначено для выполнения команды операционной системы. Шаблон используется так же, как и в действии СООБЩЕНИЕ.

Пример предложения сценария:

```
1       ВЫПОЛНИТЬ    "dir /p"
        ЕСЛИ
```

Параметры действия ЗАДАЧА определяются следующим образом:

<параметры действия ЗАДАЧА> ::= <имя модели> [{,<простой параметр>}]

<простой параметр> ::= <имя простой цели> | <число>

Действие ЗАДАЧА описывает обращение одной модели к другой для решения некоторой частной задачи, как это имеет место в традиционном программировании при вызове подпрограммы. Первый параметр указывает имя вызываемой модели (к началу консультации эта модель должна находится в Базе знаний). Далее перечисляются простые цели, с помощью которых происходит обмен данными между вызывающей и вызываемой моделями. Параметрами могут быть простые цели и числовые константы. При вызове модели происходит передача не только значений, но и имен (включая развернутые) простых целей, являющихся параметрами действия ЗАДАЧА. Если простая цель является числовым атрибутом, в вызываемую модель передается также значение по умолчанию и диапазон возможных значений. Если простая цель является утверждением, будут переданы имена (простое и развернутое) соответствующего символьного атрибута, а также шаблон утверждения.

Если значение переданной в модель цели не было известно в момент обращения, но оказалось вычисленным к моменту возвращения в вызывающую модель, результат вычисления будет передан в вызывающую модель. Значения целей, вычисленных к моменту обращения, не могут быть изменены в вызванной модели.

Пример фрагмента сценария с вызовом моделей:

```
1    ЦЕЛЬ       стаж 1, возраст 1, стаж 2, возраст 2
2    ЗАДАЧА     "TEST", оценка кандидата 1, стаж 1, возраст 1
3    ЗАДАЧА     "TEST", оценка кандидата 2, стаж 2, возраст 2
```

В данном примере предполагается, что необходимо сделать выбор из двух кандидатов (например, на какую-либо должность) на основании оценок, получаемых ими при прохождении некоторого стандартного теста. Тест описан в модели TEST и использует данные о стаже и возрасте кандидата. При выполнении предложений сценария с именами 2 и 3 будет вызвана модель TEST и переданы данные о стаже и возрасте. Результатами тестирования являются оценки кандидатов, вычисляемые в этой модели.

Параметры действия АРГУМЕНТЫ определяется следующим образом:

<параметры действия АРГУМЕНТЫ> ::=

{ <имя простой цели> , } <имя простой цели>

Данное действие предназначено для описания тех простых целей, которые данная модель принимает при ее вызове из другой модели. Оно должно быть первым в сценарии. При вызове модели перечисленные в действии АРГУМЕНТЫ простые цели заменяются целями из вызывающей модели. Если в подмодель передавалось число, то передается только значение подцели, а имя ее не изменяется. Имена простых целей в подмодели должны быть не короче имен соответствующих целей в вызывающей модели.

Пример фрагмента сценария модели TEST:

```
1    АРГУМЕНТЫ  оценка_кандидата, стаж      , возраст
2    ЦЕЛЬ       оценка кандидата
```

Выполняя данное действие, подмодель заменит имена перечисленных простых целей на имена переданных из модели параметров и примет вычисленные в вызывающей модели значения числовых атрибутов "стаж " и "возраст ".

Параметр действия СДЕЛАТЬ_ТОЧКУ определяется следующим образом:

<параметр действия СДЕЛАТЬ_ТОЧКУ> ::= [<имя файла>]

Текст указывает имя файла, в котором должна быть создана контрольная точка. Если текст опущен, в качестве имени принимается имя текущей модели с расширением ".PNT". При создании контрольной точки происходит запоминание во внешнем файле всех выведенных к текущему моменту значений, включая значения в моделях, вызвавших данную, если такие имеются. Таким образом, в отличие от действия ВОЗВРАТ, удается передать во внешний файл значения из всех выполняемых моделей.

Внешний файл представляет собой последовательность строк, описывающих выполняемые модели. Модели описываются в порядке их вызова: сначала - исходная модель, далее - модель, вызванная из исходной, и т.д., до текущей модели. Описание каждой модели начинается с заголовка, состоящего из двух строк:

<первая строка заголовка> ::= '>'<имя модели>

<вторая строка заголовка> ::= ПРЕДЛОЖЕНИЕ = <номер>

где номер указывает, какое по порядку (начиная с первого) предложение сценария выполняется в модели в момент создания контрольной точки. Остальные строки содержат описания данных, и их структура определяется следующим образом:

<строка описания данных> ::= <имя простой цели>=<значение>

<значение> ::= <число>

Параметр действия ЗАГРУЗИТЬ_ТОЧКУ определяется следующим образом:

<параметр действия ЗАГРУЗИТЬ_ТОЧКУ> ::= [<имя файла>]

Текст указывает имя контрольной точки; если текст опущен, будет загружаться точка с именем текущей модели и расширением ".PNT". Выполнение данного предложения приводит к загрузке из внешнего файла информации о значениях простых целей в модели. Внешний файл может быть либо получен в результате выполнения действия СДЕЛАТЬ ТОЧКУ (при этом он доступен для последующей корректировки пользователем), либо создан вручную. Поэтому все загруженные значения рассматриваются в ходе решения задачи как полученные от пользователя.

Параметр действия ВЫХОД определяется следующим образом:

<параметр действия ВЫХОД> ::= [<число>]

Действие ВЫХОД предназначено для принудительного прекращения консультации. Если не задан параметр действия то после выполнения данного предложения управление получат средства просмотра оперативного протокола консультации. Если параметр задан (какое либо число), то произойдет выход из Консультатора.

Параметр действия СДЕЛАТЬ_ТОЧКУ_ВЫХОД определяется следующим образом:

<параметр действия СДЕЛАТЬ_ТОЧКУ_ВЫХОД> ::= [<имя файла>]

Данное предложение аналогично предложению СДЕЛАТЬ ТОЧКУ и отличается от него только тем, что после создания контрольной точки следует немедленное завершение консультации. Это удобно использовать для описания прерывания консультации (и сохранения всех выведенных результатов) с возможностью возобновления решения задачи с момента прерывания.

Пример.

```
1    ЗАГРУЗИТЬ_ТОЧКУ
     ЕСЛИ продолжить решение
2    ЦЕЛЬ симптомы,рекомендации
3    СООБЩЕНИЕ
     " Для продолжения консультации необходимо проведение
```

следующих лабораторных исследований :"

4 РЕЗУЛЬТАТ "^v",0,рекомендации

5 СДЕЛАТЬ_ТОЧКУ_ВЫХОД ЕСЛИ нет исследований

6 РЕЗУЛЬТАТ -5,диагноз

В данном примере перед началом решения задачи выясняется, является ли данная консультация продолжением прерванной ранее. Если нет, то решение задачи начинается сначала. Когда система выясняет, какие лабораторные исследования должны быть сделаны для продолжения решения, она сообщает о них пользователю. Если результаты исследований уже имеются, решение может быть продолжено; иначе происходит прерывание консультации с запоминанием всех промежуточных данных. После получения результатов исследований консультация может быть продолжена с момента прерывания.

Параметр действия НАСТРОЙКА определяется следующим образом:

<параметр действия НАСТРОЙКА> ::= <шаблон 1600> [{,<имя простой цели>}]

С помощью этого действия разработчик ЭС может задавать реакцию Консультатора на клавиши F9 и F10 в ходе решения задачи.

Параметры действия НАСТРОЙКА аналогичны параметрам действия СООБЩЕНИЕ. В шаблоне сообщения указывается, какая команда DOS должна быть выполнена в том случае, когда пользователь, вместо ответа на вопрос, нажмет клавишу F9 или F10. Описание реакции на каждую клавишу начинается в шаблоне сообщения с первой позиции строки и имеет вид:

F<номер клавиши>: <шаблон команды DOS>

где номер клавиши - число 9 или 10, а шаблон команды содержит не более 160 символов. Для обозначения имени активной цели (т.е. той, к которой задается вопрос) используются символы "^x".

По умолчанию клавиша F9 не используется, а по клавише F10 выполняется команда DOS: EHELP ^x exp lst.pic.

Настройка клавиш осуществляется при каждом выполнении действия НАСТРОЙКА. При выполнении общего сброса или сброса выведенных результатов отменяется текущая настройка клавиш и принимается настройка по умолчанию.

Данное действие удобно использовать для описания контекстно-зависимой помощи в ходе консультации. По одной из клавиш можно получать, например, помощь для начинающего пользователя ЭС, а по другой - сжатую справочную информацию для специалиста.

Пример.

```
1       АРГУМЕНТЫ индекс_объекта, состояние
2       НАСТРОЙКА
            "F9:  ehelp ^x obj ^v.pic
            F10: ht hlp.txt",индекс объекта
```

В примере приведены первые предложения подзадачи, посвященной анализу состояния некоторого типового объекта, индекс которого передается в подзадачу в качестве параметра. К каждому конкретному объекту O1, O2 и т.д. имеется графическая помощь, описание которой содержится в файлах OBJ O1.PIC, OBJ O2.PIC и т.д. Общая помощь по системе имеет вид гипертекста с именем HLP.TXT. После выполнения настройки пользователь получает возможность:

по клавише F9 - получать графическую помощь, относящуюся к рассматриваемому конкретному объекту;

по клавише F10 - получать общую помощь по системе.

Режим приобретения знаний предназначен для ввода описаний моделей в Базу знаний с проведением семантико-синтаксического контроля вводимой информации. Этот режим используется разработчиком в процессе создания и модификации экспертной системы.

В режиме приобретения знаний средства комплекса позволяют осуществлять диалоговый ввод и корректировку Базы знаний с семантико-

синтаксическим контролем вводимой информации, тестировать полученные модели и компилировать введенные знания.

Ввод и коррекция описаний осуществляется с помощью шаблонов ввода, соответствующих основным конструкциям языка представления знаний. Шаблоны ввода изображаются на экране дисплея в виде окон.

В режиме приобретения знаний обеспечивается работа с Базой знаний как с иерархией элементов различных типов. Определеныследующие типы элементов:

- модели;
- предложения сценария
- числовые атрибуты;
- символьные атрибуты;
- значения символьных атрибутов;
- правила.

База знаний представляет собой совокупность различных моделей. Каждая модель включает в себя описания предложений сценария, числовые и символьные атрибуты. Описания символьных атрибутов включают в себя описания значений (т.е. утверждений о состоянии проблемной области). Атрибутам и утверждениям соответствуют правила вывода их значений. Сложные правила соответствуют символьным атрибутам, простые - числовым и утверждениям.

Структура диалога по приобретению знаний отражает синтаксическую структуру Базы знаний. В ходе диалога пользователь может работать на различных уровнях иерархии, при этом состояние диалога и допустимые действия пользователя определяются тем уровнем, на котором ведется работа.

В начальном состоянии диалога пользователю предлагается работа с оглавлением Базы знаний - списком имен и комментариев к моделям, находящимся в Базе знаний. Пользователь может выбрать модель, при этом диалог переходит в состояние работы с моделью в целом. В этом состоянии

пользователь может устанавливать режим разрешения конфликтов, запускать средства проверки целостности или средства распечатки текста модели, отменять сделанные в ходе работы с данной моделью изменения, а также выбирать тип элементов, с которыми он хочет работать. В последнем случае произойдет переход к более низкому уровню иерархии. Например, если пользователь выберет числовые атрибуты, то он перейдет к работе с числовыми атрибутами данной модели. Затем пользователь может спуститься по иерархии и начать работу с правилами вывода значения какого-либо числового атрибута. Потом он можетвернуться к работе с числовыми атрибутами данной модели.

В каждом состоянии диалога (кроме состояния работы с моделью в целом) пользователь может:

- работать со списком элементов текущего уровня, относящихся к одному и тому же элементу предыдущего уровня;
- просматривать и корректировать описание одного элемента.

При работе со списком выводится на экран терминала список имен соответствующих элементов, выделяется текущий элемент и высвечивается текст комментария к нему. Выделение текущего элемента называется установкой внимания на этом элементе. При работе со списком пользователь имеет возможность:

- перемещать, переименовывать или уничтожать текущий элемент;
- создавать новые элементы;
- корректировать текст комментария текущего элемента;
- переходить к работе с описанием текущего элемента;
- переходить к работе со списком элементов более низкого уровня, относящихся к текущему элементу (если они есть);
- перемещать внимание по элементам списка;
- возвращаться к предыдущему уровню иерархии.

Например, при работе с числовыми атрибутами, описанными в модели, пользователь может просматривать список имен этих атрибутов,

переименовывать текущий атрибут, вводить или корректировать его развернутое имя, просматривать и корректировать описание текущего атрибута, переходить к работе со списком правил вывода значения текущего атрибута и т.д.

При работе с описанием отдельного элемента на экран терминала выводится шаблон описания этого элемента, соответствующий синтаксису языка представления знаний. Пользователь может заполнить или откорректировать поля данного шаблона. При вводе описаний элементов осуществляется контроль вводимой информации и в случае обнаружения синтаксических ошибок пользователю выдаются сообщения о них.

На последовательность ввода описаний элементов модели накладывается единственное ограничение, состоящее в том, что в описании вводимого элемента не допускаются ссылки на элементы, описания которых еще не введены. Например, прежде чем ввести предложение сценария с параметром " диагноз", необходимо ввести описание атрибута "диагноз".

Описание модели может вводиться по частям, поэтому при вводе может возникнуть потеря целостности модели. Модель считается целостной, если выполнены следующие условия:

- введен сценарий консультации;
- во всех предложениях сценария определены действия и, если это необходимо, параметры действий;
- в сценарии консультации нет ссылок на несуществующие предложения;
- для каждой простой цели указано хотя бы одно правило вывода ее значения;
- для каждого символьного атрибута указано хотя бы одно значение;
- для всех правил, атрибутов и значений символьных атрибутов указаны имена;
- для всех числовых атрибутов указан диапазон допустимых значений;

- в правилах определены типы и в соответствии с этими типами - условия применимости и действия;
- в правилах и предложениях сценария не упоминаются не описанные в данной модели цели.

Контроль синтаксической корректности арифметических и логических выражений, а также контроль отсутствия в них ссылок на неописанные цели осуществляются непосредственно при вводе выражений в Базу знаний. Остальные условия целостности проверяются отдельно по завершении ввода описания модели с помощью средств тестирования.

В одной модели пользователь может описать не более:

- 500 (30-учебный вариант) утверждений о значениях символьных атрибутов ;
- 100 символьных атрибутов; □
- 300 числовых атрибутов;
- 1000 (30-учебный вариант) правил;
- 100 предложений сценария.

Область данных в описании модели, содержащая тексты имен, сообщений и откомпилированные арифметические и логические выражения, не может превышать 64 тыс. байт.

Режим консультации предназначен для решения задач конечного пользователя на основе моделей в Базе знаний и исходных данных о конкретной ситуации в проблемной области. Этот режим используется разработчиком на этапе тестирования экспертных систем, а также конечным пользователем при решении конкретных задач с помощью разработанных систем.

В режиме консультации обеспечиваются следующие возможности:

- решение конкретной задачи на основе выбранной модели с формиро
- ванием объяснений ЗАЧЕМ в ходе консультации задается тот или иной
- вопрос;

- просмотр информации об описанных в модели целях, в том числе - объяснений, КАК получены значения этих целей, и изменение значений любых целей;
- просмотр информации о правилах в модели;
- сброс значений всех целей либо значений всех выведенных целей (т.е. отмена всех значений, не являющихся исходными данными);
- получение трассы решения задачи;
- запись протокола консультации в файл;
- создание и загрузку контрольных точек.

Решение задачи осуществляется в ходе диалога экспертнойсистемы с пользователем. На экран выдаются сообщения в соответствиисо сценарием консультации, а также задаются вопросы, описанные в применяемых правилах-вопросах. При задании вопроса пользователю выдается краткое поясняющее сообщение о виде ожидаемого ответа.

Значения числовых атрибутов должны вводиться в виде действительных чисел с точностью до двух знаков после десятичной точки и не выходящих за пределы диапазона допустимых значений этих атрибутов. Диапазоны допустимых значений выдаются пользователю в текстах подсказок во время задания вопросов.

Значения утверждений должны вводиться в виде коэффициентов определенности. Если при задании вопроса пользователю предлагается меню, составленное на основе шкалы, то пользователь имеет возможность как ввести числовое значение коэффициента, так и выбрать в меню лингвистическое описание коэффициента.

При задании сложного вопроса пользователю выводится на экран список значений целевого символьного атрибута. Если вопрос альтернативный, то пользователь в соответствии с поясняющим сообщением системы должен выбрать то значение, которое имеет место в рассматриваемой ситуации. Если вопрос дистрибутивный, то пользователь должен указать коэффициенты определенности (в числовом виде или с

помощью меню если имеется шкала), для всех тех значений символьного атрибута, о которых он имеет информацию.

Если пользователь не имеет информации, позволяющей ему ответить на вопрос, он может ввести ответ НЕ ЗНАЮ. Тогда будут применяться другие правила вывода искомого значения, если такие правила есть в модели.

Вместо ответа на вопрос пользователь может ввести команду ЗАЧЕМ. В этом случае консультация будет прервана и пользователю будет сообщено, для вывода какой цели задается вопрос. Если пользователь снова введет команду ЗАЧЕМ, то он получит информацию о том, подцелью какого правила является эта цель, и т.д. Таким образом, многократный ввод команды ЗАЧЕМ позволяет просматривать цели и правила на пути в сети вывода от исходной цели к текущей, т.е. к той, о которой задается вопрос.

Пользователь имеет возможность вместо ответа на вопрос откладывать решение задачи и переходить к работе с моделью в целом. По окончании этой работы он имеет возможность либо вернуться к прерванной консультации, либо завершить ее.

При выдаче информации о целях пользователю сообщаются:

- имена целей;
- типы (числовой атрибут или утверждение);
- развернутые имена;
- значения и их источники - для выведенных целей (в случае утверждений выдаются коэффициенты определенности);
- статус цели (значение неизвестно, цель активна, значение не удалось получить).

Пользователь может изменить значение любой цели. При внесении изменений он должен учитывать возможность нарушения корректности текущего решения, поскольку автоматически не пересматриваются те значения, которые были выведены исходя из измененных целей.

Просмотр источников значений целей обеспечивает объяснение того, КАК был получен тот или иной результат.

При выдаче информации о правилах пользователю сообщаются:

- имена правил;
- комментарии к правилам; цели и подцели правил.

Если пользователь хочет повторить консультацию с другими исходными данными, он может сбросить все значения по команде СБРОС.

Программа обеспечивает сброс либо всех значений, либо только тех, которые не были получены в ответ на вопрос к пользователю, т.е. выведенных значений. С помощью сброса выведенных значений, изменения некоторых исходных данных и повторного решения задачи пользователь может проверять гипотезы типа "что если".

Решение задач может, по желанию пользователя, проводиться с трассировкой консультации. При этом на экране появляется дополнительное окно для трассы, в которое помещаются сообщения о выполняемых действиях сценария, анализируемых целях и рассматриваемых правилах. Трассировка может использоваться как для отладки Базы знаний, так и для получения объяснений о работе экспертной системы.

При решении задач с протоколированием все сообщения, выдаваемые в ходе консультации, и все ответы, вводимые пользователем в ответ на вопросы, будут помещены в текстовой файл, называемый протоколом консультации. Если решение задачи проводится с трассировкой, то в протокол будет помещена и трасса консультации.

Контрольные точки используются для отладки моделей и создаются по команде пользователя во время консультации. При создании точки информация о состоянии всех моделей, загруженных к моменту выдачи команды, запоминается во внешнем файле, имя которого запрашивается у пользователя. В отличие от файлов, создаваемых действием СОЗДАТЬ ТОЧКУ из сценариев моделей, данный файл является двоичным и не должен корректироваться. В дальнейшем пользователь может загрузить контрольную

точку из данного файла и продолжить консультацию. Загрузка может быть успешно осуществлена, если с момента создания точки модели не корректировались и не тестировались.

1.8. Вспомогательные средства комплекса

Программное средство ELK предназначено для подготовки гипертекстов. Оно позволяет значительно облегчить работу с большими объемами текстовой информации, что достигается путем предоставления пользователю простых и удобных средств структурирования текстов, возможности связывания полученных структурных единиц в конструкцию, отражающую семантику текста и поисковых процедур, эффективно реализующих эти связи.

В комплексе программ ЭКО средство ELK имеет следующие применения:

- при помощи ELK создана контекстно-зависимая Помощь, содержащая всю справочную информацию, имеющуюся в документации на комплекс, и доступная пользователю в любой момент в обоих режимах работы ОПС;
- при помощи ELK пользователь может разрабатывать собственные справочные средства для оказания помощи в процессе работы с экспертными системами;
- совместно используя средства ELK и XHT, пользователь может получать отчеты по Базе знаний в виде гипертекстов.

Программное средство XHT представляет собой генератор отчетов по Базе знаний. Оно предназначено для того, чтобы помочь пользователю при изучении, анализе и отладке Базы знаний, созданной с помощью комплекса ЭКО. При помощи XHT могут быть получены отчеты трех типов: текстовые, гипертекстовые и графические.

Текстовый отчет представляет собой твердую копию некоторой модели в Базе знаний и включает распечатку атрибутов, правил и сценария этой модели, а также таблицы перекрестных ссылок. Пользователю предоставляются широкие возможности по управлению содержанием текстовых отчетов.

Гипертекстовый отчет включает ту же информацию, что и текстовый, однако отличается от последнего способом использования. Если работа с текстовым отчетом может производиться без помощи ПЭВМ, то гипертекстовый отчет, как и любой другой гипертекст, существенно использует ПЭВМ для реализации поисковых процедур, поддерживающих связи между его фрагментами.

Графический отчет представляет собой графическое изображение сети вывода некоторой модели в Базе знаний в виде И-ИЛИ графа. Пользователь может распечатать графический отчет или работать с ним непосредственно на ПЭВМ, используя возможность быстрого перехода к гипертекстовому отчету по соответствующей модели.

Для получения гипертекстовых отчетов необходимо совместно использовать средства XHT и ELK.

В состав комплекса программ ЭКО входят два стандартных интерфейса: графический и информационный.

Средства графического интерфейса предназначены для организации диалога с пользователем с помощью графических изображений (рисунков, схем, карт, диаграмм и т.д.).

Преимущества представления информации в виде рисунка по сравнению с текстовым представлением той же информации в некоторых случаях весьма существенны. Например: указать местоположение на плане.

Комплекс программ ЭКО допускает использование рисунков как в качестве иллюстраций, так и для задания вопросов.

Экспертная система, созданная при помощи комплекса ЭКО, может использоваться как подзадача в системах обработки данных, использующих

файловую структуру DOS для хранения данных. Программная и информационная совместимость системы обработки данных и экспертной системы осуществляется через так называемый информационный интерфейс.

Свои функции информационный интерфейс осуществляет при помощи следующих операций:

- получение данных от систем обработки данных;
- интерпретация полученных данных в терминах переменных экспертной системы.

ТЕМА 2. ОСНОВНЫЕ НАПРАВЛЕНИЯ ИССЛЕДОВАНИЙ В ОБЛАСТИ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

Иску́сственный интелле́кт (англ. Artificial intelligence (AI)) — раздел информатики, изучающий возможность обеспечения разумных рассуждений и действий с помощью вычислительных систем и иных искусственных устройств. При этом в большинстве случаев заранее неизвестен алгоритм решения задачи.

Теорией явно не определено, что именно считать необходимыми и достаточными условиями достижения интеллектуальности. Хотя на этот счёт существует ряд гипотез, например, гипотеза Ньюэлла-Саймона. Обычно к реализации интеллектуальных систем подходят именно с точки зрения моделирования человеческой интеллектуальности.

В рамках искусственного интеллекта различают два основных направления:

- символьное (семиотическое, нисходящее) основано на моделировании высокоуровневых процессов мышления человека, на представлении и использовании знаний;
- нейрокибернетическое (нейросетевое, восходящее) основано на моделировании отдельных низкоуровневых структур мозга (нейронов).

Таким образом, задачей искусственного интеллекта является построение компьютерной интеллектуальной системы, которая обладала бы уровнем эффективности решений неформализованных задач, сравнимым с человеческим или превосходящим его.

На данный момент не существует систем искусственного интеллекта, однозначно отвечающих основным задачам, обозначенным выше. Наиболее часто используемые при построении систем искусственного интеллекта парадигмы программирования — функциональное программирование и логическое программирование. От традиционных структурного и объектно-ориентированного подходов к разработке программной логики они отличаются нелинейным выводом решений и низкоуровневыми средствами поддержки анализа и синтеза структур данных.

Можно выделить две научные школы с разными подходами к проблеме ИИ:

→ Конвенционный ИИ

→ Вычислительный ИИ

В **конвенционном ИИ** главным образом используются методы машинного самообучения, основанные на формализме и статистическом анализе.

Методы конвенционного ИИ:

- Экспертные системы: программы, которые действуя по определенным правилам, обрабатывают большое количество информации, и в результате выдают заключение на её основе.
- Рассуждение на основе аналогичных случаев (Case-based reasoning).
- Байесовские сети.
- Поведенческий подход: модульный метод построения систем ИИ, при котором система разбивается на несколько сравнительно автономных программ поведения, которые запускаются в зависимости от изменений внешней среды.

Вычислительный ИИ подразумевает итеративную разработку и обучение (например, подбор параметров в сети связности). Обучение основано на эмпирических данных и ассоциируется с не-символьным ИИ и мягкими вычислениями.

Основные методы:

- Нейронные сети: системы с отличными способностями к распознаванию.
- Нечёткие системы: методики для рассуждений в условиях неопределенности (широко используются в современных промышленных и потребительских системах контроля)
- Эволюционные вычисления: здесь применяются понятия традиционно относящиеся к биологии такие как популяция, мутация и естественный отбор для создания лучших решений задачи. Эти методы делятся на эволюционные алгоритмы (например, генетические алгоритмы) и методы роевого интеллекта (например, муравьиный алгоритм).

В рамках гибридных интеллектуальных систем пытаются объединить два этих направления. Экспертные правила умозаключений могут генерироваться нейронными сетями, а порождающие правила получают с помощью статистического обучения.

Искусственный интеллект — одна из новейших наук, появившихся во второй половине 20-го века. На базе вычислительной техники, математической логики, программирования, психологии, лингвистики, нейрофизиологии и других отраслей знаний. Искусственный интеллект — это образец междисциплинарных исследований, где соединяются профессиональные интересы специалистов разного профиля. Само название новой науки возникло в конце 60-х гг., а в 1969 г. в Вашингтоне (США) состоялась первая Всемирная конференция по искусственному интеллекту.

Известно, что совокупность научных исследований обретает права науки, если выполнены два необходимых условия. У этих исследований должен быть объект изучения, не совпадающий с теми, которые изучают

другие науки. И должны существовать специфические методы исследования этого объекта, отличные от методов других, уже сложившихся наук. Исследования, которые объединяются сейчас термином "искусственный интеллект", имеют свой специфический объект изучения и свои специфические методы. В этой статье мы обоснуем это утверждение.

Когда в конце 40-х — начале 50-х гг. появились ЭВМ, стало ясно, что инженеры и математики создали не просто быстро работающее устройство для вычислений, а нечто более значительное. Оказалось, что с помощью ЭВМ можно решать различные головоломки, логические задачи, играть в шахматы, создавать игровые программы. ЭВМ стали принимать участие в творческих процессах: сочинять музыкальные мелодии, стихотворения и даже сказки. Появились программы для перевода с одного языка на другой, для распознавания образов, доказательства теорем. Это свидетельствовало о том, что с помощью ЭВМ и соответствующих программ можно автоматизировать такие виды человеческой деятельности, которые называются интеллектуальными и считаются доступными лишь человеку.

Несмотря на большое разнообразие не вычислительных программ, созданных к началу 60-х гг., программирование в сфере интеллектуальной деятельности находилось в гораздо худшем положении, чем решение расчетных задач.

Причина очевидна: программирование для задач расчетного характера опиралось на соответствующую теорию — вычислительную математику. На основе этой теории было разработано много методов решения задач. Эти методы стали основой для соответствующих программ. Ничего подобного для не вычислительных задач не было. Любая программа была здесь уникальной, как произведение искусства. Опыт создания таких программ никак не обобщался, их создавать не формализовалось.

Никто не станет отрицать, что, в отличие от искусства, у науки должны быть методы решения задач. С помощью этих методов все однотипные задачи должны решаться единообразным способом. И "набив руку" на

решении задач определенного типа, легко решать новые задачи, относящиеся к тому же типу. Но именно таких методов и не смогли придумать те, кто создавал первые программы не вычислительного характера.

Когда программист создавал программу для игры в шахматы, то он использовал собственные знания о процессе игры. Он вкладывал их в программу, а компьютер лишь механически выполнял эту программу. Можно сказать, что компьютер “не отличал” вычислительные программы от не вычислительных. Он одинаковым образом находил корни квадратного уравнения или писал стихи. В памяти компьютера не было знаний о том, что он на самом деле делает.

Об интеллекте компьютера можно было говорить, если бы он сам, на основании собственных знаний о том, как протекает игра в шахматы и как играют в эту игру люди, сумел составить шахматную программу или синтезировал программу для писания несложных вальсов и маршей.

Не сами процедуры, с помощью которых выполняется та или иная интеллектуальная деятельность, а понимание того, как их создать, как научиться новому виду интеллектуальной деятельности, — вот где скрыто то, что можно назвать интеллектом. Специальные метапроцедуры обучения новым видам интеллектуальной деятельности отличают человека от компьютера.

Следовательно, в создании искусственного интеллекта основной задачей становится реализация машинными средствами тех метапроцедур, которые используются интеллектуальной деятельности человека. Что же это за процедуры?

В психологии мышления есть несколько моделей творческой деятельности.

Одна из них называется лабиринтной. Суть лабиринтной гипотезы, на которой основана лабиринтная модель, состоит в следующем: переход от исходных данных задачи к ее решению лежит через лабиринт возможных альтернативных путей. Не все пути ведут к желаемой цели, многие из них

заводят в тупик, из которого надо уметь возвращаться к тому месту, где потеряно правильное направление. Это напоминает попытки не слишком умелого школьника решить задачу об упрощении алгебраических выражений. Для этой цели на каждом шагу можно применять некоторые стандартные преобразования или придумывать искусственные приемы. Но весьма часто вместо упрощения выражения происходит его усложнение, и возникают тупики, из которых нет выхода. По мнению сторонников лабиринтной модели мышления, решение всякой творческой задачи сводится к целенаправленному поиску в лабиринте альтернативных путей с оценкой успеха после каждого шага.

С лабиринтной моделью связана первая из метапроцедур — целенаправленный поиск в лабиринте возможностей. Программированию этой метапроцедуры соответствуют многочисленные процедуры поиска, основанные на соображениях “здравого смысла” (человеческого опыта решения аналогичных задач). В 60-х гг. было создано немало программ на основе лабиринтной модели, в основном игровых и доказывающих теоремы “в лоб”, без привлечения искусственных приемов. Соответствующее направление в программировании получило название эвристического программирования. Высказывались даже предположения, что целенаправленный поиск в лабиринте возможностей — универсальная процедура, пригодная для решения любых интеллектуальных задач.

Но исследователи отказались от этой идеи, когда столкнулись с задачами, в которых лабиринта возможностей либо не существовало, либо он был слишком велик для метапроцедуры поиска, как, например, при игре в шахматы. Конечно, в этой игре есть лабиринт возможностей — это все мыслимые партии игры. Но как в этом астрономически большом лабиринте найти те партии, которые ведут к выигрышу? Лабиринт столь велик, что никакие мыслимые скорости вычислений не позволят целенаправленно перебрать пути в нем. И все попытки использовать для этого человеческие (эвристики в данном случае профессиональный опыт шахматистов) не дают

пути решения задачи. Поэтому современные шахматные программы уже давно используют не только метапроцедуру целенаправленного поиска, но и другие метапроцедуры, связанные с другими моделями мышления.

Долгие годы в психологии изучалась ассоциативная модель мышления. Основной метапроцедурой этой модели является ассоциативный поиск и ассоциативное рассуждение. Предполагается, что решение неизвестной задачи так или иначе основывается на уже решенных задачах, чем-то похожих на ту, которую надо решить. Новая задача рассматривается как уже известная, хотя и несколько отличающаяся от известной. Поэтому способ ее решения должен быть близок к тому, который когда-то помог решить подобную задачу.

Для этого надо обратиться к памяти и попытаться найти нечто похожее, что ранее уже встречалось. Это и есть ассоциативный поиск. Когда, увидев незнакомого человека, вы стараетесь вспомнить, на кого он похож, реализуется метапроцедура ассоциативного поиска. Но понятие ассоциации в психологии шире, чем просто “похожесть”. Ассоциативные связи могут возникнуть и по контрасту, как противопоставление одного другому, и по смежности, т. е. в силу того, что некоторые явления возникали в рамках одной и той же ситуации или происходили одновременно (или с небольшим сдвигом по времени).

Ассоциативное рассуждение позволяет переносить приемы, использованные ранее, на текущую ситуацию. К сожалению, несмотря на многолетнее изучение ассоциативной модели, не удалось создать стройную теорию ассоциативного поиска и ассоциативного рассуждения. Исключение составляет важный, но частный класс ассоциаций, называемых условными рефлексами. И все же метапроцедура ассоциативного поиска и рассуждения сыграла важную роль: она помогла создать эффективные программы в распознавании образов, в классификационных задачах и в обучении ЭВМ. Но одновременно эта метапроцедура привела к мысли о том, что для ее эффективного использования надо привлечь результаты, полученные в

другой модели мышления, опирающейся на идею внутреннего представления проблемной области, на знания о ее особенностях, закономерностях и процедурах действия в ней.

Это представление о мыслительной деятельности человека обычно называют модельной гипотезой. Согласно ей, мозг человека содержит модель проблемной ситуации, в которой ему надо принять решение. Для решения используются метапроцедуры, оперирующие с совокупностью знаний из той проблемной области, к которой принадлежит данная проблемная ситуация. Например, если проблемная ситуация — переход через улицу с интенсивным движением, то знания, которые могут помочь ее разрешить, касаются способов организации движения транспорта, сигналов светофоров, наличия дорожек для перехода и т. п.

В модельной гипотезе основными метапроцедурами становятся представление знаний, рассуждения, поиск релевантной (связанной с данной проблемной ситуацией) информации в совокупности имеющихся знаний, их пополнение и корректировка. Эти метапроцедуры составляют ядро интеллектуальных возможностей современных программ и программных систем, ориентированных на решение творческих задач. В совокупности с метапроцедурами целенаправленного поиска в лабиринте возможностей, ассоциативного поиска и рассуждения они образуют арсенал интеллектуальных средств, которым располагают современные интеллектуальные системы, часто называемые системами, основанными на знаниях.

Объектом изучения искусственного интеллекта являются метапроцедуры, используемые при решении человеком задач, традиционно называемых интеллектуальными, или творческими. Но если психология мышления изучает эти метапроцедуры применительно к человеку, то искусственный интеллект создает программные (а сейчас уже и программно-аппаратные) модели таких метапроцедур.

Цель исследований в области искусственного интеллекта — создание арсенала метапроцедур, достаточного для того, чтобы ЭВМ (или другие технические системы, например роботы) могли находить по постановкам задач их решения. Иными словами, стали автономными программистами, способными выполнять работу профессиональных программистов-прикладников (создающих программы для решения задач в определенной предметной области). Разумеется, сформулированная цель не исчерпывает всех задач, которые ставит перед собой искусственный интеллект. Это цель ближайшая. Последующие цели связаны с попыткой проникнуть в области мышления человека, которые лежат вне сферы рационального и выразимого словесно (вербально) мышления. Ибо в поиске решения многих задач, особенно сильно отличающихся от ранее решенных, большую роль играет та сфера мышления, которую называют подсознательной, бессознательной, или интуитивной.

Основными методами, используемыми в искусственном интеллекте, являются разного рода программные модели и средства, эксперимент на ЭВМ и теоретические модели. Однако современные ЭВМ уже мало удовлетворяют специалистов по искусственному интеллекту. Они не имеют ничего общего с тем, как устроен человеческий мозг. Поэтому идет интенсивный поиск новых технических структур, способных лучше решать задачи, связанные с интеллектуальными процессами. Сюда относятся исследования по нейроподобным искусственным сетям, попытки построить молекулярные машины, работы в области голографических систем и многое другое.

Существуют несколько основных проблем, изучаемых в искусственном интеллекте.

1. Представление знаний — разработка методов и приемов для формализации и последующего ввода в память интеллектуальной системы знаний из различных проблемных областей, обобщение и классификация накопленных знаний, использование знаний при решении задач.

2. Моделирование рассуждений — изучение и формализация различных схем человеческих умозаключений, используемых в процессе решения разнообразных задач, создание эффективных программ для реализации этих схем в вычислительных машинах.

3. Диалоговые процедуры общения на ее естественном языке, обеспечивающие контакт между интеллектуальной системой и человеком-специалистом в процессе решения задач.

4. Планирование целесообразной деятельности — разработка методов построения программ сложной деятельности на основании тех знаний проблемной области, которые хранятся в интеллектуальной системе.

5. Обучение интеллектуальных систем в процессе их деятельности, создание комплекса среде для накопления и обобщения умений и навыке накапливаемых в таких системах.

Кроме этих проблем исследуются многие другие, составляющие тот задел, на который будут опираться специалисты на следующем витке развития теории искусственного интеллекта.

Некоторые из самых впечатляющих гражданских ИИ систем:

- Deep Blue — победил чемпиона мира по шахматам.
- Mycin — одна из ранних экспертных систем, которая могла диагностировать небольшой набор заболеваний, причем часто так же точно как и доктора.
- 20q — проект, основанный на идеях ИИ, по мотивам классической игры «20 вопросов». Стал сильно популярен после появления в интернете на сайте 20q.net.
- Распознавание речи. Системы такие как ViaVoice способны обслуживать потребителей.
- Роботы в ежегодном турнире RoboCup соревнуются в упрощенной форме футбола.

Искусственный интеллект и теоретические проблемы психологии. Можно выделить две основные линии работ по ИИ. Первая связана

ссовершенствованием самих машин, с повышением "интеллектуальности" искусственных систем. Вторая связана с задачей оптимизации совместной работы "искусственного интеллекта" и собственно интеллектуальных возможностей человека.

Переходя к собственно психологическим проблемам ИИ О.К. Тихомиров выделяет три позиции по вопросу о взаимодействии психологии и искуственного интеллекта:

1) "Мы мало знаем о человеческом разуме, мы хоти его воссоздать, мы делаем это вопреки отсутствию знаний"- эта позиция характерна для многих зарубежных специалистов по ИИ.

2) Вторая позиция сводится к констатации ограниченности результатов исследований интеллектуальной деятельности, проводившихся психологами, социологами и физиологами. В качестве причины указывается отсутствие адекватных методов. Решение видится в воссоздании тех или иных интеллектуальных функций в работе машин. Иными словами, если машина решает задачу ранее решавшуюся человеком, то знания, которые можно подчерпнуть, анализируя эту работу и есть основной материал для построения психологических теорий.

3) Третья позиция характеризуется оценкой исследования в области искусственного интеллекта и психологии как совершенно независимых. В этом случае допускается возможность только потребления, использования психологических знаний в плане психологического обеспечения работ по ИИ.

Закономерно возникает вопрос о влиянии работ по искусственному интеллекту на развитие психологической науки. О.К.Тихомиров [10] выделяет в качестве первого результата - появление новой области психологических исследований, а именно, сравнительные исследования того, как одни и те же задачи решаются человеком и машиной. Кроме того, уже первые работы по искусственному интеллекту показали, что не только область решения задач затрагивается сопоставительными исследованиями,

но и проблема мышления в целом. Возникла потребность в уточнении критериев дифференциации "творческих" и "нетворческих" процессов.

Популярные идеи системного анализа позволили сделать сравнение принципов работы искусственных систем и собственно человеческой деятельности важным эвристическим приемом выделения именно специфического психологического анализа деятельности человека.

В 1963 г. выступая на совещании по философским вопросам физиологии ВНД и психологии, А.Н. Леонтьев сформулировал следующую позицию: машина воспроизводит операции человеческого мышления, и следовательно соотношение "машинного" и "немашинного" есть соотнесение операционального и неоперационального в человеческой деятельности в то время этот вывод был достаточно прогрессивен и выступал против кибернетического редукционизма. Однако в последствии при сравнении операций, из которых слагается работа машины, и операций как единиц деятельности человека выявились существенные различия - в психологическом смысле "операция" отражает способ достижения результатов, процессуальную характеристику, в то время как применительно к машинной работе этот термин используется в логико-математическом смысле (характеризуется результатом).

В работах по искусственному интеллекту постоянно используется термин "цель". Анализ отношения средств к цели А.Ньюэлл и Г.Саймон называют в качестве одной из "эвристик". В психологической теории деятельности "цель" является конституирующим признаком действия в отличии от операций (и деятельности в целом). В то время как в искусственных системах "целью" называют некоторую конечную ситуацию, к которой стремится система. Признаки этой ситуации должны быть четко выявленными и описанными на формальном языке. Цели человеческой деятельности имеют другую природу. Конечная ситуация может по-разному отражаться субъектом: как на понятийном уровне, так и в форме представлений или перцептивного образа. Это отражение может

характеризоваться разной степенью ясности, отчетливости. Кроме того, для человека характерно не просто достижение готовых целей, но и формирование новых.

Также работа систем искусственно интеллекта, характеризуется непросто наличием операций, программ, "целей", а как отмечает О.К.Тихомиров,- оценочными функциями. И у искусственных систем есть своего рода "ценностные ориентации". Но специфику человеческой мотивационно-эмоциональной регуляции деятельности составляет использование не только константных, но и ситуативно возникающих и динамично меняющихся оценок, существенно также различие между словесно-логическими и эмоциональными оценками. В существовании потребностей и мотивов видится различие между человеком и машиной на уровне деятельности. Этот тезис повлек за собой цикл исследований, посвященных анализу специфики человеческой деятельности.

Некоторые считают, что интеллект - умение решать сложные задачи; другие рассматривают его как способность к обучению, обобщению и аналогиям; третьи - как возможность взаимодействия с внешним миром путем общения, восприятия и осознания воспринятого. Тем не менее многие исследователи ИИ склонны принять тест машинного интеллекта, предложенный в начале 50-х годов выдающимся английским математиком и специалистом по вычислительной технике Аланом Тьюрингом. Компьютер можно считать разумным,- утверждал Тьюринг,- если он способен заставить нас поверить, что мы имеем дело не с машиной, а с человеком.

Среди специалистов нет единой точки зрения на то, что входит в область искусственного интеллекта и что в нее не входит. Нет единой точки зрения и на цели исследований, которые могли бы быть отнесены к искусственному интеллекту.

Имеются две основные точки зрения на то, что следовало бы назвать искусственным интеллектом:

1. Нейробионическая. Ее сторонники ставят перед собой цель воспроизвести искусственным образом те процессы, которые протекают в мозгу человека. Этот путь изучения естественного мозга, выявление способов его работы, создания технических средств для повторения биологических структур и протекающих в них процессов.

2. Информационная, доминирующая в искусственном интеллекте. Сторонники информационного подхода считают, что основной целью работ в искусственном интеллекте является не построение технического аналога биологической системы, а создание средств для решения задач, традиционно считающихся интеллектуальными.

Информационная точка зрения в свою очередь неоднородна. В ней можно выделить три направления.

2.1. Часть специалистов считает, можно найти свой способ ее решения на ЭВМ, который даст либо результат, подобный человеческому, либо даже лучший его. Специалисты такого типа неоднократно демонстрировали свое искусство по созданию программ такого рода. Достаточно назвать, например, программы для игры в шахматы, которые играют в эту игру лучше подавляющего большинства людей, проводящих время за шахматной доской. Но делают эти программы совсем не так, как люди.

2.2. Другая часть специалистов считает, что искусственный интеллект должен имитировать не решение отдельных (пусть и весьма творческих) задач, ибо, естественный интеллект человека - это его способность при необходимости обучаться тому или иному виду творческой деятельности. Значит и программы, создаваемые в искусственном интеллекте, должны быть ориентированы не на решение конкретных задач, а на создание для автоматического построения необходимых программ решения конкретных задач, когда в этом возникает необходимость, именно эта группа исследователей сейчас определяет лицо искусственного интеллекта, составляя основную массу специалистов этого профиля.

2.3. Третья часть специалистов – это программисты, чьими руками делают программы для решения задач искусственного интеллекта. Они склонны рассматривать область своей деятельности, как новый виток развития программирования. Они считает, что средства, разрабатываемые для написания программ решения интеллектуальных задач, в конце концов, есть средства, позволяющие по описание задачи на профессиональном естественном языке построить нужную программу на основании тех стандартных программных модулей, которые хранятся в памяти машины.

Все метасредства, которые предлагают те, кто рассматривает искусственный интеллект, как способ разобраться на информационном уровне, какие функции реализует естественный интеллект, когда он решает задачу, программисты видят насквозь призму своей цели – создание интеллектуального программного обеспечения (по существу, комплекса средств, автоматизирующих деятельность самого программиста).

На Рис. 2.1 схематично показаны точки зрения на искусственный интеллект, цифровые индексы соответствуют нейробионической точке зрения и трем возможным информационным точкам зрения. Пунктирные линии характеризуют то, что заимствуется из естественного интеллекта.

На рис. 2.2 показана схема строения искусственного интеллекта, связанная с точками зрения на него. Пятая ветвь связана с различными прикладными проблемами, в которых активно используются результаты, полученные в искусственном интеллекте и его модели и методы.

Схема, приведенная на рис. 2.2 еще слишком крупная, поэтому дадим расшифровку каждого из пяти направлений, начнем с нейробионического направления.

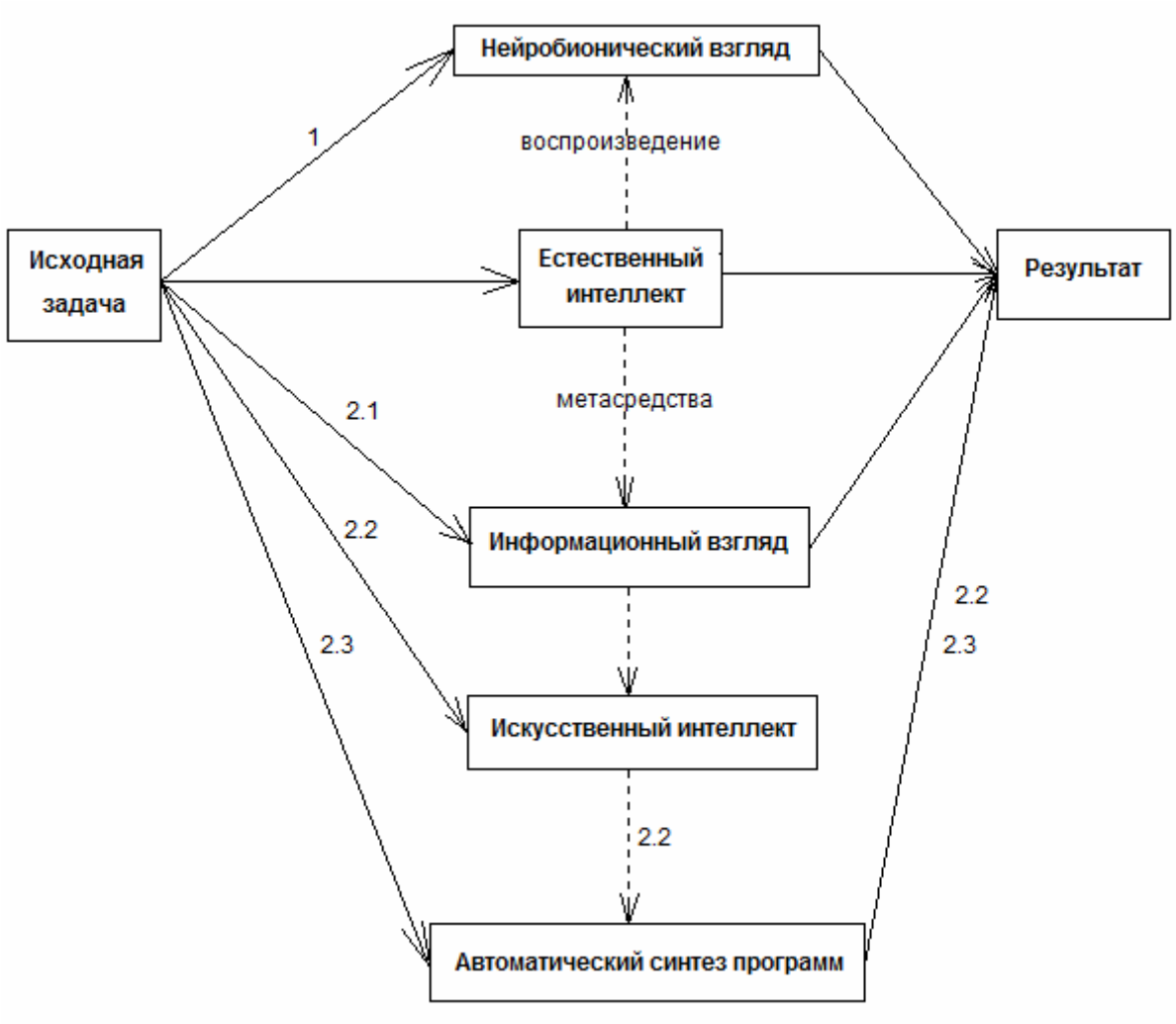

Рисунок 2.1.

В те годы, когда возникали ЭВМ, мало кто предполагал, что они очень быстро вытеснят из вычислительной сферы все остальные вычислительные устройства, Дж. Фон-Неман, с именем которого связана идея архитектуры классической ЭВМ, в те годы интересовался и другой организацией процесса вычислений, использующей аналоги нейроподобных структур, первые модели Формальных нейронов были предложены Маккалоком и Питсом, по сути, эти элементы реализовали пороговую функцию, сигнал на выходе элемента возникал лишь тогда, когда взвешенная сумма разрешающих входных сигналов превышала взвешенную сумму запрещающих входных сигналов более чем на величину, определяемую значением порога элемента. Варьируя значения весов и порога можно было добиться нужного

срабатывания Формального нейрона. Объединенные в сети, такие нейроны представлялись весьма мощным способом реализации различных процедур.

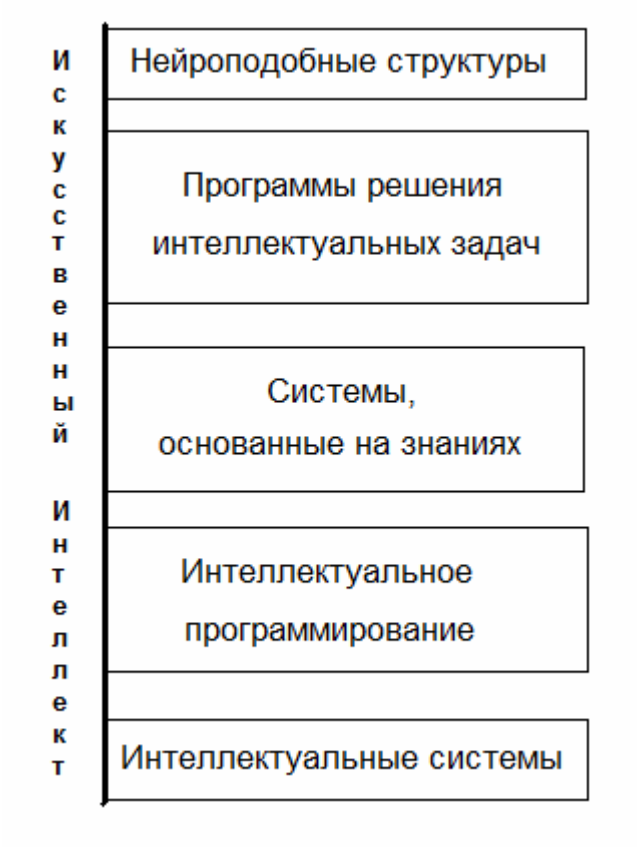

Рисунок 2.2.

Одним из наиболее известных нейробионических устройств был персептрон.

В середине 1958 г. Ф. Розенблатом была предложена модель электронного устройства, названного им перцептроном, которое должно было бы имитировать процессы человеческого мышления. Перцептрон должен был передавать сигналы от "глаза", составленного из фотоэлементов, в блоки электромеханических ячеек памяти, которые оценивали относительную величину электрических сигналов. Эти ячейки соединялись между собой случайным образом в соответствии с господствующей тогда теорией, согласно которой мозг воспринимает новую информацию и реагирует на нее через систему случайных связей между нейронами. Два года спустя была продемонстрирована первая

действующая машина "Марк-1", которая могла научится распознавать некоторые из букв, написанных на карточках, которые подносили к его "глазам", напоминающие кинокамеры. Перцептрон Розенблата оказался наивысшим достижением "восходящего", или нейромодельного метода создания искусственого интеллекта. Чтобы научить перцептрон способности строить догадки на основе исходных предпосылок, в нем предусматривалась некая элементарная разновидность автономной работы или "самопрограммирования". При распознании той или иной буквы одни ее элементы или группы элементов оказываются гораздо более существеуными, чем другие. Перцептрон мог научаться выделять такие характерные особенности буквы полуавтоматически, своего рода методом проб и ошибок, напоминающим процесс обучения. Однако возможности перцептрона были ограниченными: машина не могла надежно распознавать частично закрытые буквы, а также буквы иного размера или рисунка, нежели те, которые использовались на этапе ее обучения.

Ведущие представители так называемого "нисходящего метода" специализировались, в отличие от представителей "восходящего метода", в составлении для цифровых компьютеров общего назначения программ решения задач, требующих от людей значительного интеллекта, например для игры в шахматы или поиска математических доказательств. К числу защитников "нисходящего метода" относились Марвин Минский и Сеймур Пейперт, профессора Массачусетского технологического института. Минский начал свою карьеру исследователя ИИ сторонником "восходящего метода" и в 1951 г. построил обучающуюся сеть на вакуумных электронных лампах. Однако вскоре к к моменту создания перцептрона он перешел в противоположный лагерь. В соавторстве с южно-африканским математиком Пейпертом, с которым его познакомил Маккаллох, он написал книгу "Перцептроны", где математически доказывалось, что перцептроны, подобные розенблатовсим, принципиально не в состоянии выполнять многие из тех функций, которые предсказывал им Розенблат. Минский

утверждал, что, не говоря о роли работающих под диктовку машинисток, подвижных роботов или машин, способных читать, слушать и понимать прочитанное или услышанное, перцептроны никогда не обретут даже умения распознавать предмет частично заслоненный другим.

Нельзя сказать, что появившаяся в 1969 г. эта критическая работа покончила с кибернетикой. Она лишь переместила интерес аспирантов и субсидии правительственных организаций США, традиционно финансирующих исследования по ИИ, на другое направление исследований - "нисходящий метод".

Интерес к кибернетике в последнее время возродился, так как сторонники "нисходящего метода" столкнулись со столь же неодолимыми трудностями. Сам Минский публично выразил сожаление, что его выступление нанесло урон концепции перцептронов, заявив, что , согласно его теперешним представлениям, для реального прорыва вперед в создании разумных машин потребуется устройство, во многом похожее на перцептрон. Но в основном ИИ стал синонимом нисходящего подхода, который выражался в составлении все более сложных программ для компьютеров, моделирующих сложную деятельность человеческого мозга.

Перцептрон породил целое семейство конструкций, в основе которых лежала идея первоначального устройства Розенблата.

Метод, который лежал в основе функционирования персептрона, похож на те приемы, которые используются в распознавании образов. Это научное направление весьма близко соприкасается с исследованиями по искусственному интеллекту. Строго говоря, нет никаких оснований не включать его в состав нового научного направления, во всяком случае, нет особых возражений. Но, традиционно, возникшее гораздо ранее направление, связанное с распознаванием образов, существует отдельно. Хотя во многих пограничных вопросах эти две области научных исследований перекрываются. Например, в методах формирования решающих правил при

обучении на примерах и контрпримерах, как это происходит в персептронах, или в задачах анализа зрительных сцен.

Дальнейшие исследования в области нейробионических устройств или по пути увеличения числа слоев из Формальных нейронов, изменения и усложнения способа функционирования нейронов и построения решающего правила. Параллельно развивалась теория персептронов. Но два обстоятельства затормозили эти работы. Очень быстро при решении практических задач распознавания стало понятно, что возможности устройств типа персептронов ограничены. Например, они не могли разгадать изображение, являющееся комбинацией двух ранее персептрону известных, на составляющие, это заставляло рассматривать подобную комбинацию как новое изображение. С другой стороны, Минский М. и Пейперт С. Доказали ряд теорем о персептронах, в которых обосновали их принципиальную ограниченность, отсутствие новых идей нейробионических устройств, в течение десятка лет не давало повода для развития этих исследований, но успехи микроэлектроники последних лет, сделавшие возможным создания большого числа нейроподобных элементов в малом объеме, вновь возродило надежды сторонников этого подхода, появились нейрокомпьютеры, в которых процесс решения задачи развертывается на сети искусственных нейронов, этот процесс может включать в себя множество параллельно и асинхронно протекающих подпроцессов, что сулит высокую эффективность решения задач на нейрокомпьютерах. Беда состоит только в том, что пока неизвестны регулярные приемы программирования решения задач для ЭВМ такой архитектуры.

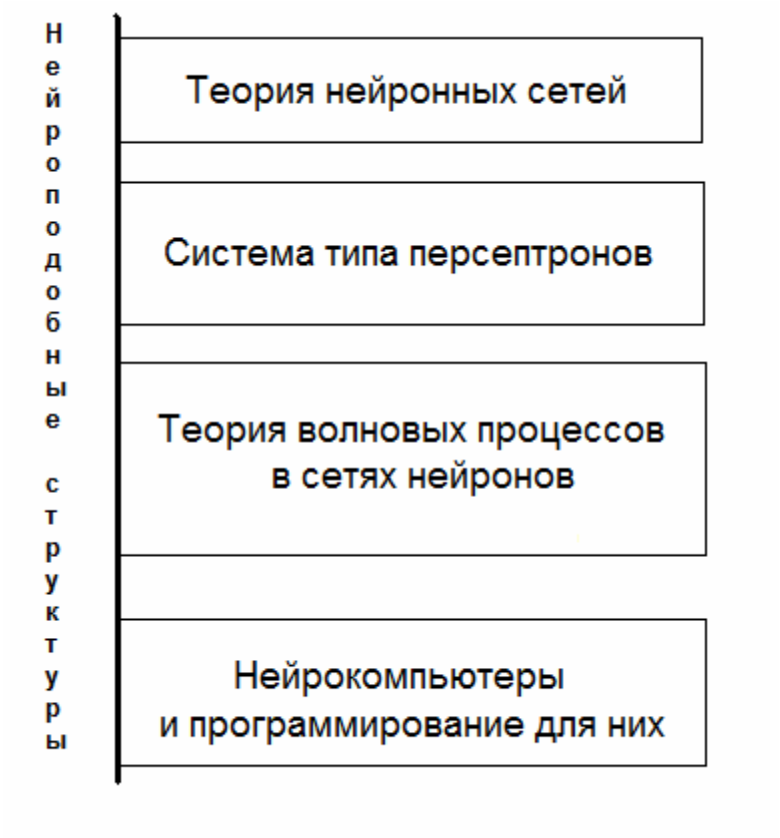

Рисунок 2.3.

На рис. 2.3 показана схема, характеризующая структуру нейробионического направления в искусственном интеллекте.

Программы для решения интеллектуальных задач могут быть разделены на несколько групп, определяемых типом задач, решаемых этими программами.

Первую группу составляют игровые программы, Они в свою очередь делятся на две подгруппы: человеческие игры и компьютерные игры. Дальнейшее деление этой группы программ показано на рис. 2.4. Особенностью всех программ для имитации человеческих игр является большая роль поисковых процедур, поиск лучшего или локально лучшего хода требует в сложных играх типа шахмат просмотра большого числа вариантов, недаром шахматные программы являются специальным тестом для проверки эффективности поисковых процедур.

Интересно отметить, что именно поисковые процедуры казались на первом этапе развития работ по интеллектуальным программам той

метапроцедурой, с помощью которой можно будет решать все интеллектуальные задачи. Первая программа, которая обобщила эту идею, называлась «Общий решатель задач». В этой программе, созданной А. Ньюэллом, Дж. Шоу и Г, Саймоном, поиск с локальными критериями успеха был основной процедурой. Решение всех задач, по мысли авторов программы, могло быть сведено к поиску пути в лабиринте альтернативных возможностей. И хотя эти надежды не оправдались, цикл подобных исследований оказался весьма полезным. Были созданы достаточно эффективные процедуры поиска, используемые специалистами по искусственному интеллекту не только при решении игровых задач, но и во многих других областях. Например, при планировании целесообразной деятельности в интеллектуальных системах.

Переборные игры составляют, по-видимому, большинство во множестве распространенных среди людей игр. Существенно меньшую часть составляют топологические игры, в которых необходимо учитывать не только дерево игры, задаваемое возможными последовательностями кодов противников, но и структурой самой позиции, как целого. Примером такой игры может служить Го. В этой игре оценка позиции не может быть сведена, как, например, в шахматах, к описанию множества фигур и их расположения на игровом поле. Для Го важно не конкретное расположение камней по тем или иным полям, а те конфигурации, которые не образуют на плоскости игрового поля. Программирование таких игр требует создания в памяти ЭВМ эталонных образов тех или иных областей, занятых камнями противников. А это куда более сложная и до конца пока не решенная задача, нежели организация поиска по дереву альтернативных возможностей.

Стохастические игры возникают тогда, когда в процессе игры возникают вероятностные шаги или очередная ситуация формируется при участии некоторого вероятностного механизма. С программированием таких игр (например, карточной игры в очко) связано развитие методов правдоподобного оценивания вариантов, получившего в искусственном

интеллекте заметное использование, во всех таких ситуациях важно уметь пересчитать оценку правдоподобия результирующей ситуации после выбора определенного хода с учетом оценок правдоподобия текущей ситуации и выбора противника.

К стохастическим играм примыкают и игры с неполной информацией, когда при принятии решения необходимо как-то оценивать недостающую информацию. Эти приемы постоянно используются при обращении к содержимому памяти в интеллектуальных системах, когда в ней отсутствует нужная информация, что является почти стандартной ситуацией при функционировании таких систем в сложных предметных областях.

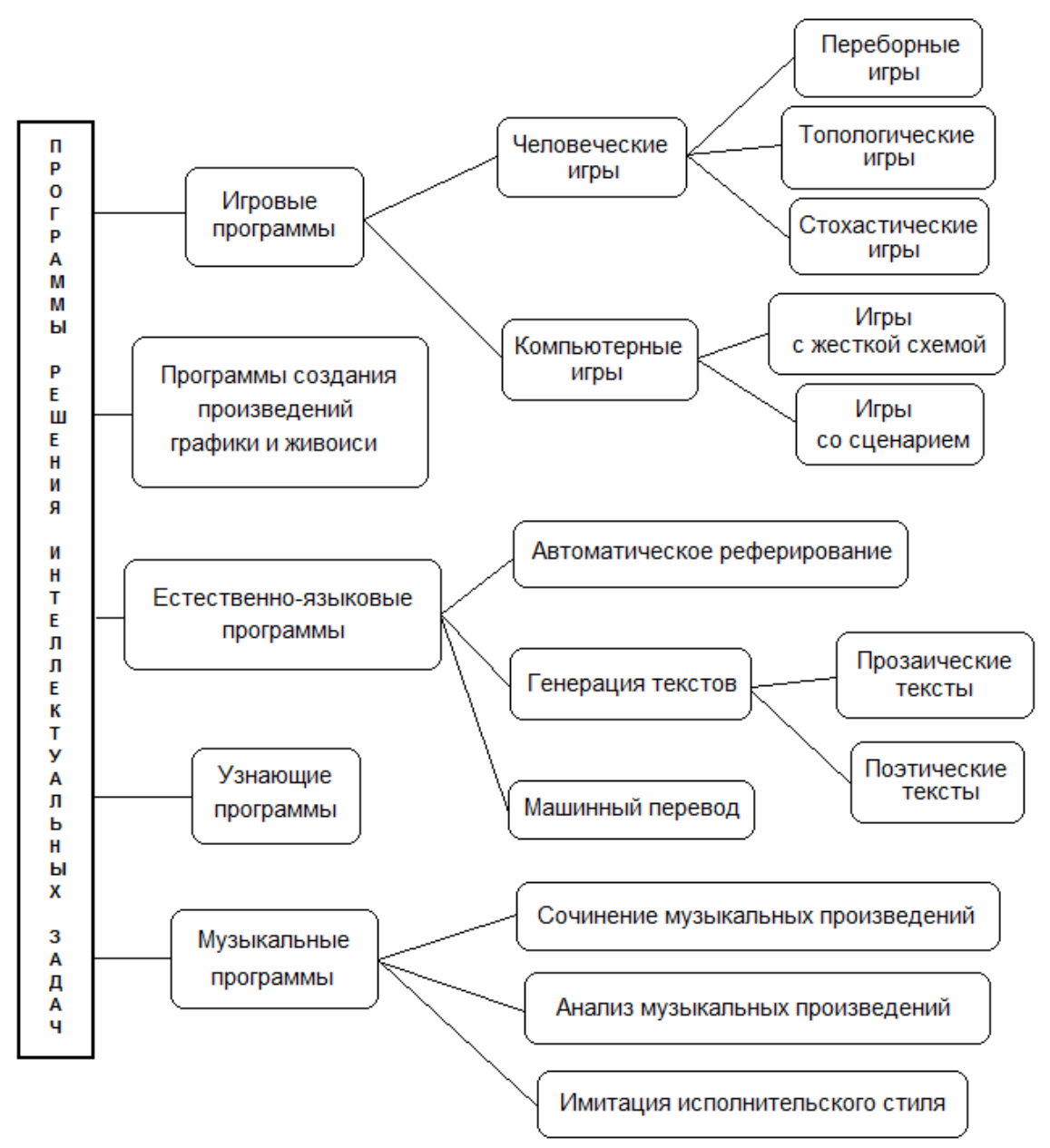

Рисунок 2.4.

Компьютерные игры, получившие в последнее время столь широкое распространение, вообще говоря, не относятся традиционно к работам по искусственному интеллекту. Хотя эта ситуация столь же случайна, как и ситуация с распознаванием образов Конечно, игры с жесткой схемой, в которых "интеллекта" практически нет, не представляют для работ по искусственному интеллекту интереса, но сценарные игры представляют для рассматриваемой области науки прямой интерес. В них используются

сценарии развития игры, движение по которым определяется обоими партнерами. Эти же принципы используются и в таких типичных для искусственного интеллекта задачах, как организация диалога интеллектуальной системы с пользователем на ограниченном естественном языке. Интересны сценарии и для планирования целесообразной деятельности в интеллектуальных работах и других системах искусственного интеллекта.

С самого начала появления ЭВМ стали создаваться программы для машинного перевода и автоматического реферирования текстов. Создание этих программ оказало значительное влияние на развитие искусственного интеллекта, заложило основы тех работ, которые были непосредственно связаны с естественно-языковым общением пользователей с интеллектуальными системами, в системах машинного перевода были разработаны модели и методы, позволяющие автоматически проводить морфологический синтаксический и во многом семантический анализ фраз естественного языка, нащупаны приемы анализа связного текста. Все эти результаты активно используются при обработке естественно-языковых текстов в интеллектуальных системах. В работах по автоматическому реферированию были заложены основы понимания общей структуры текста, как целого, от идеи "что говорится" был сделан переход к идее "о чем говорится". Это позволило на более высоком уровне создавать программы генерации, текстов.

Если первые программы такого вида основывались на жестких моделях порождения или вероятностных механизмах, то более поздние программы генерации текстов стали опираться на идеи сценариев, а также на приемы, наработанные в программах по автоматическому реферированию. Сейчас качество прозаических текстов, создаваемых с помощью ЭВМ, достаточно интересно, если тексты имеют жесткую внутреннюю структуру, определяемую их назначением. Таковы, например, волшебные сказки, в основе которых лежит жесткий сценарий поведения действующих лиц,

таковы хроникальные заметки или документы. Но созданы и достаточно любопытные программы, порождающие поэтические тексты, в которых наблюдается иная крайность - почти полное отсутствие смысловой структуры при достаточно жесткой структуре для формы.

Музыкальные программы, пожалуй, наиболее известны широкой публике, так как первые опыты по созданию таких программ сразу дали весьма обнадеживающие результаты. Этот успех связан опять-таки с наличием, с одной стороны, жестких правил при построении мелодии, а с другой стороны, во многом вероятностными моделями, порождающими остальные элементы музыкального произведения. Менее известны широкой публике программы, ориентированные на музыковедов, в которых имитируются стили исполнения или исследуется "анатомия" музыкальных произведений и процесса их сочинения. Однако, весь комплекс музыкальных программ, хотя и не оказал прямого влияния на работы по искусственному интеллекту, оказался полезным для Формирования общего взгляда на природу творческих процессов и их моделирования.

Узнающие программы зародились в недрах исследований по распознаванию образов. Но как уже говорилось» многие из них оказали значительное влияние на идеи, характерные для работ по созданию интеллектуальных систем, особенно при создании обучающих систем. При их создании были найдены методы оценивания похожести одних объектов на другие, заложены основы рассуждений по аналогии и ассоциации, использования обучающих последовательностей примеров и контрпримеров, все это вошло в Фонд методов, которыми пользуется специалист по искусственному интеллекту.

Несколько особняком стоят программы, с помощью которых создаются машинные произведения в области графики и живописи. Эти исследования связаны, в основном, с созданием специальных программных и в меньшей мере аппаратных средств для устройств графического вывода. Но косвенно

эти программы оказывают влияние на те разделы искусственного интеллекта, которые связаны с использованием зрительных образов при решении задач.

Интеллектуальные системы — это сложные программно-аппаратные комплексы, обязательно включающие в свой состав ЭВМ. Чтобы ввести знания о предметной области в память ЭВМ, необходимо представить их в такой форме, которая была бы понятна машине. Иными словами, знания надо записать на языке, понятном ЭВМ, как понятны ей записи на языках программирования.

Для этого существуют специальные языки представления знаний. Их можно разделить на типы по тем формальным моделям представления знаний, которые лежат в их основе.

Таких формальных моделей три:

- Логическая.
- Сетевая (семантические сети).
- Продукционная.

Общая структура этого направления показана на рис. 2.5.

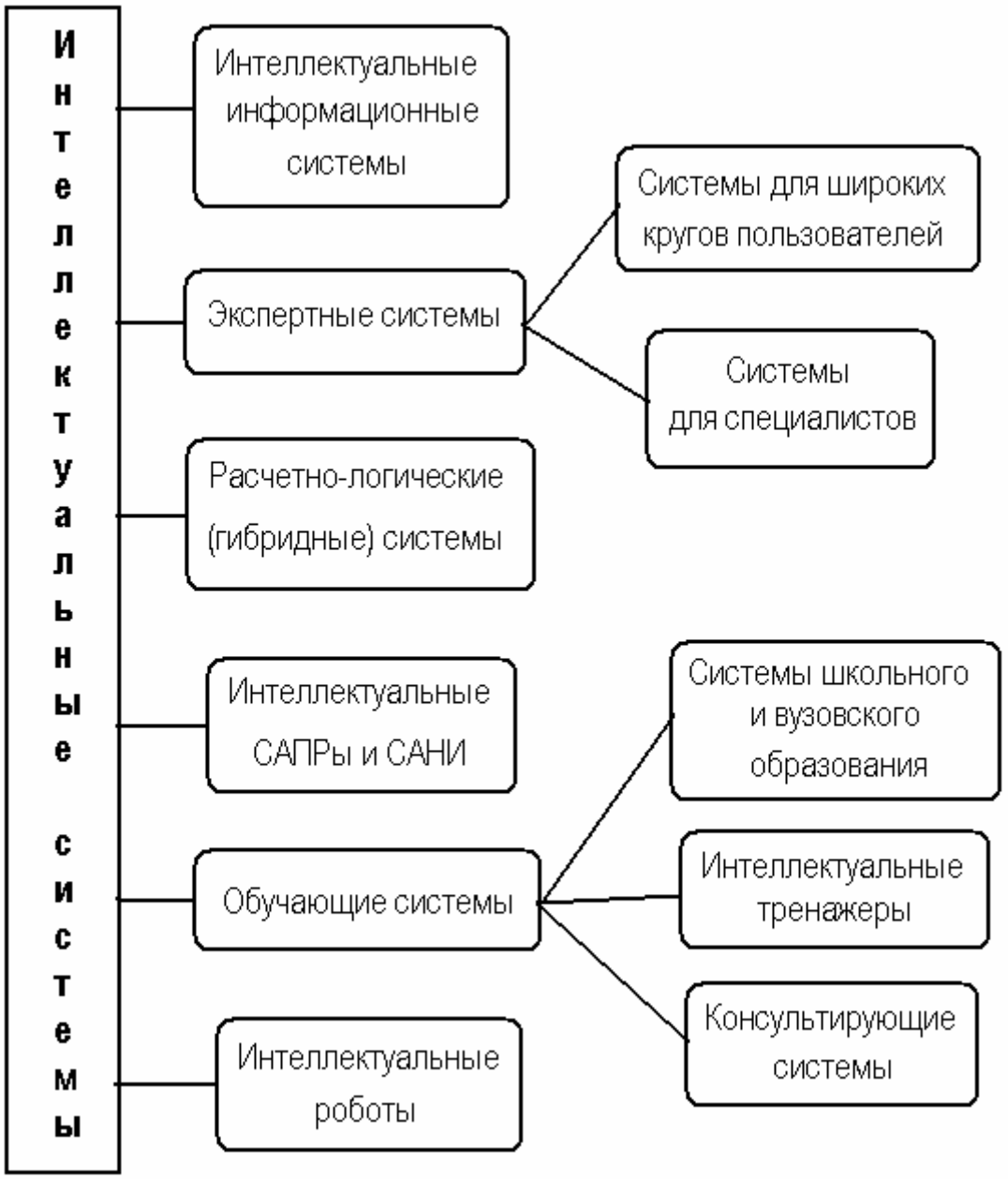

Рисунок 2.5

Интеллектуальные системы обладают рядом особенностей, отличающих их от систем, создававшихся до развития работ по искусственному интеллекту. Общая структура интеллектуальной системы показана на рис. 2.6 кроме обычной ЭВМ со всеми ее вычислительными и логическими возможностями интеллектуальная система включает в себя еще ряд блоков, они могут быть аппаратно выполнены независимо от ЭВМ или имитироваться теми средствами, которые имеются в ЭВМ там, где это возможно (ясно, например, что сенсоры и эффекторы не могут быть смоделированы на ЭВМ, если ими надо пользоваться в реальной среде).

Охарактеризуем кратко блоки интеллектуальной системы, интеллектуальный интерфейс включает в себя средства общения между интеллектуальной системой и пользователем. В его состав входят лингвистический процессор, обеспечивающий ввод, вывод и понимание текстов на ограниченном профессиональном естественном языке, система ввода-вывода речи, система восприятия зрительных образов и, наконец, сенсоры, способные воспринимать и первично обрабатывать различные по своей природе сигналы, поступающие от внешнего мира (радиосигналы, электрические измерительные сигналы, сигналы телекоммуникации и т.п.). На выходе интеллектуального интерфейса возникает информация, согласованная по Форме с принятой в интеллектуальной системе модели представления информации, которая имеет определенный маркер, свидетельствующий о том, куда эта информация должна быть направлена на внутренний вход интеллектуального интерфейса, эта информация перекодируется в нужную Форму и выдается потребителям.

Таким образом, интеллектуальный интерфейс сам по себе есть весьма сложное устройство (комплекс программных систем), что позволяет рассматривать его в виде специализированного процессора общения, в его состав в конкретной реализации могут входить, конечно, не все части, показанные на рис.3.1, но в любом случае его функции остаются, в частности, в ЭВМ новых поколений, которые должны быть весьма просты в использовании, наличие интеллектуального интерфейса, включающего в себя лингвистический процессор, а также и систему анализа и синтеза речи, является обязательным.

Следующий блок интеллектуальной системы - решатель, организует всю работу системы, он является основным блоком по пересылке информации и ее переработке. В реальных интеллектуальных системах решатель не обязан сосредотачивать все свои функции в одном Физическом блоке. Такая абсолютная централизация может отрицательно сказаться на

эффективности работы системы. Поэтому схема на рис. 3.2 отражает не физический состав системы, а ее функциональную схему.

В решателе в рабочих полях памяти хранится вся промежуточная информация, информация, которая хранится лишь определенное небольшое время, служебная информация и т.д. Решатель, как это видно из рисунка, может выполнять роль интерфейса между интеллектуальной системой и ЭВМ, на которой происходят все необходимые вычисления, либо представлять собою специализированную ЭВМ. Поэтому наличие ЭВМ, указанной внутри квадратика, символизирующего решатель, не является обязательным.

В базе знаний хранится вся необходимая для решения задач информация о предметной области, в текущие Фактические данные, отражающие ее состояние и предысторию, хранятся в базе данных. Для организации целенаправленного Функционирования интеллектуальной системы служит система планирования, в которой хранятся априорно заложенная в систему совокупность целей, а также запоминаются новые цели, полученные с помощью системы обучения. Система обучения, кроме того, участвует в формировании новых знаний за счет анализа результатов взаимодействия интеллектуальной системы с внешней средой.

Системы объяснения и доверия служат для обоснования полученных решений, если получатель интересуется этим, Об объяснениях уже говорилось, система доверия или обоснования решения привлекает информацию из базы знаний, которая непосредственно не участвовала в Формировании решения, но заключает в себе дополнительные аргументы в пользу полученного решения, Наконец, система когнитивной графики помогает пользователю, работающему в интерактивном режиме, в решении задач.

Если эту общую структуру интеллектуальной системы соотнести с типами интеллектуальных систем, перечисленных на рис. 2.5, то можно составить таблицу, в которой указаны те обязательные блоки из общей

структуры, приведенной на рис. 2.6, которые присутствуют в структуре интеллектуальной системы данного типа, эта таблица выглядит следующим образом.

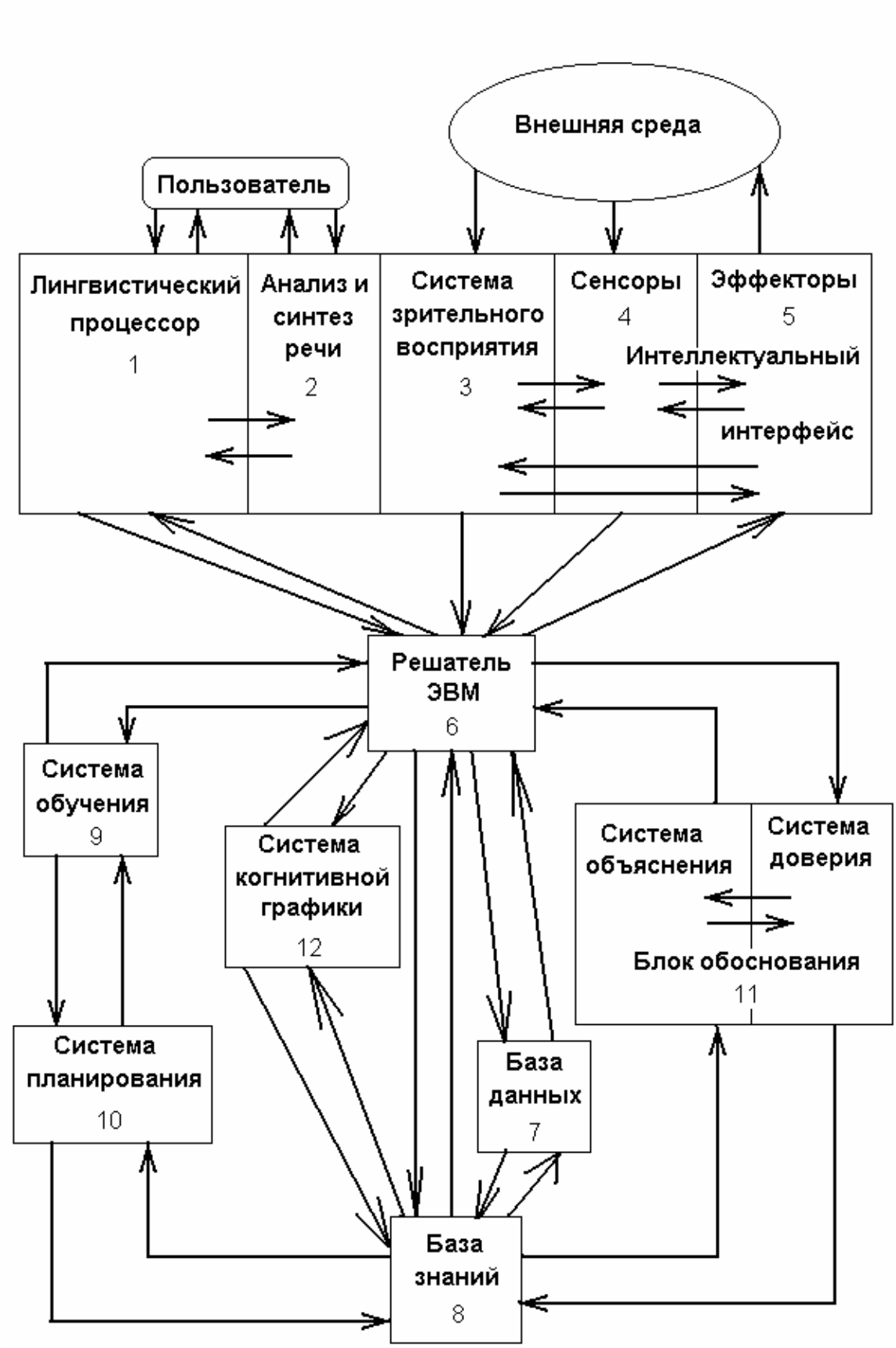

Рисунок 2.6

Перейдем теперь к рассмотрению классов интеллектуальных систем. Экспертные системы получили в последнее десятилетие широкую известность, они первыми из интеллектуальных систем входят в наш быт, и, по-видимому, именно их влияние в обыденкой жизни будет наиболее ощутимым. Свое название экспертные системы получили в связи с тем, что основу знаний, хранящихся в их памяти, составляют сведения, приобретенные от экспертов-профессионалов в тех или иных предметных областях, в них входит стандартное ядро любой интеллектуальной системы (лингвистический процессор, решатель, база данных и база знаний), а также система обоснования, являющаяся обязательной составляющей экспертных систем, во всяком случае, именно в экспертных системах впервые возникла функция объяснения, а затем и Функции доверия.

Экспертные системы бывают двух типов. К первому типу относятся системы, задачей которых является подъем общего уровня профессионализма в некоторой предметной области (медицине, геологии, юриспруденции и др.). Такие системы выступают, как правило, в роли помощников-консультантов, в сложных случаях рядовой специалист может обратиться к ним за помощью, ибо в базе знаний экспертной системы хранится "на все случаи жизни", вещественный опыт ведущих специалистов в этой области, а также информация, извлеченная из различных источников знаний. Такие экспертные системы могут играть роль простого приказчика, диагностировать совместно с человеком те или иные ситуации, осуществлять прогноз и регноз (поиск причин) и выполнять многие другие функции. Обычно базы знаний систем такого типа являются замкнутыми, решатель имеет встроенные функции, которыми управляет некоторый сценарий общения с пользователем, а работа самого пользователя не требует от него никаких специальных знаний.

Это системы массивные, доступные всем специалистам. Экспертные системы такого типа должна быть недорогими, и современная

промышленность интеллектуальных систем готовится к тому, чтобы сделать экспертные системы этого типа бытовым прибором широкого распространения»

Второй тип экспертных систем рассчитан на специалистов достаточно высокой квалификации. Это системы-помощники, берущие на себя вспомогательную, роль в деятельности специалиста. Они могут снабжать его разнообразной информацией, делать технические расчеты и выводы подсказывать специалисту альтернативные решения и т.д. Такие системы играют роль специфического инструмента, помогающего специалисту сделать свою деятельность более эффективной. Будучи включенными, в систему передачи данных, такие экспертные системы становятся для пользователя способом провести консультации с коллегами, получить интересующую информацию из хранилищ, расположенных в других местах, подключиться к мощным ЭВМ, когда возникает необходимость в проведении сложных расчетов.

Информационные системы начали строить еще до появления ЭВМ. С их появлением информационные системы стали строиться на их основе. Но только развитие работ в области искусственного интеллекта сделало реальным создание информационных систем, удовлетворяющих пользователей с точки зрения комфортности общения с ними. В таких системах интеллектуальный интерфейс обеспечивает пользователю возможность входа в систему с запросами на обычном профессиональном языке, а наличие базы знаний и решателя позволяет в информационных системах такого уровня обеспечивать поиск ответа и тогда, когда прямого ответа на его в базе данных нет. Активное развитие интеллектуальных информационных систем обеспечивает основы перехода к информационному обществу.

Появление методов решения задач, характерных для искусственного интеллекта, не отменило всего того, что накопила за долгие годы вычислительная математика. Многие задачи экономического планирования,

моделирования социальных процессов, управления сложными организационно-техническими системами и многие другие виды задач требуют совместного использования численных моделей и моделей, опирающихся на качественные рассуждения специалистов, так возникают гибридные системы, в которых при решении задач чередуются этапы чистых вычислений и логических рассуждений, связанных с выбором дальнейшего пути решения по результатам, полученным к этому моменту. Системы такого типа называют расчетно-логическими, их появление знаменует переход всех систем, используемых при решении самых различных задач прогнозирования, планирования и управления, на новый уровень. Пользователи таких систем обычно решают свои задачи не в одиночку, а целыми коллективами. Все члены этого коллектива, решая свои задачи, должны взаимодействовать с другими участниками» Расчетно-логические системы дают им такую возможность даже в том случае, когда участники находятся в различных местах. Электронная связь (почта) через ЭВМ и каналы передачи данных позволяет им обмениваться необходимой информацией, а интеллектуальные интерфейсы дают этот обмен по форме ничем не отличающимся от обычного общений специалистов между собой.

Системы автоматизации проектирования (САПРы) и системы автоматизации научных исследований (САНИ) уже занимают значительное место в нашей инженерной и научной деятельности, Появление в них ядра интеллектуальной системы позволит, как и в случае расчетно-логических систем значительно повысить уровень, принимаемых коллективных и индивидуальных решений, и обеспечит для специалистов куда более удобный способ общения с САПРами и САНИ, чем это было до сих пор. Во всяком случае, интеллектуализация систем таких типов сейчас идет весьма быстро.

Робототехника, создавшая роботов первых двух поколений, работавших по жестким программам, подошла к третьему поколению интеллектуальным роботам. Роботы этого поколения должны действовать в

динамической среде, когда их действия заранее не могут быть заданы жесткими схемами. Такие роботы должны уметь оценивать текущую ситуацию, классифицировать ее и планировать свою деятельность в соответствии с теми глобальными целями и задачами, которые были априорно заложены в их память. Способные действовать автономно, эти работы могут работать в тех средах, где пребывание человека опасно или невозможно, а также выполнять такие технологические операции, которые человеку недоступны.

Последний класс интеллектуальных систем, известный сегодня в искусственном интеллекте, это обучающие системы. Такие системы также возникли еще до появления работ в области искусственного интеллекта, но только методы, разработанные в новом направлении науки, позволили сделать такие системы эффективными. Прежде всего, это касается систем для индивидуального и группового обучения в школе или в высших учебных заведениях. Уже накоплен немалый опыт использования таких обучающих систем. И стало ясно, что в девяностых годах доля обучающих систем в человеческом образовании будет непрерывно возрастать.

Особый класс обучающих систем составляют интеллектуальные тренажеры. Они соединяют в себе обычный тренажер с системой, имитирующей деятельность инструктора. Такие тренажеры должны резко повысить качество профессиональной подготовки людей в различных сферах человеческой деятельности.

Консультирующие системы, обеспечивающие индивидуальную консультацию по широкому кругу проблем, интересующих людей различных социальных групп, также могут быть отнесены к обучающим системам, так как в процессе общения с ними люди приобретают новую информацию и новые знания, а, следовательно, обучаются.

Вот те основные направления работ, которые образуют в совокупности научное направление, которое называется искусственный интеллект. Конечно, сказанное не более чем весьма поверхностное описание той

вершины айсберга искусственного интеллекта, которая видна с первого взгляда. Все остальные, куда большее, находится вне уровня, на котором можно описывать науку, не вдаваясь в дебри используемой в ней терминологии, приемов и методов.

В практику человеческой деятельности интеллектуальные системы уже внедряются. Это и наиболее известные широкому кругу специалистов экспертные системы, передающие опыт более и подготовленных специалистов менее подготовленным интеллектуальные информационные системы (например, системы машинного перевода), и интеллектуальные роботы, другие системы, имеющие полное право называться интеллектуальными. Без таких систем современный научно-технический прогресс уже невозможен.

Любой вид человеческой деятельности так или иначе связан с рассуждением. В таком разделе науки, как искусственный интеллект, этот термин используется в его общечеловеческом смысле, как понятие естественного языка, нестрогое и неоднозначное. Но есть один вид рассуждений, являющихся объектом изучения специальной науки — логики, который определяется строго научно.

ТЕМА № 3. ПРЕДСТАВЛЕНИЕ ЗНАНИЙ И ВЫВОД НА ЗНАНИЯХ В ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМАХ

Третье основное направление в искусственном интеллекте образует его фундамент. Именно здесь создается теория данного научного направления, решаются основные проблемы, связанные с центральным объектом изучения искусственного интеллекта - знаниями.

На рис. 3.1 показана структура этого направления. Всякая предметная (проблемная) область деятельности может быть описана в виде некоторой совокупности сведений о структуре этой области, основных ее

характеристиках, процессах, протекающих в ней, а также о способах решения, возникающих в ней задач. Все эти сведения образуют знания о предметной области, при использовании интеллектуальных систем для решения задач в данной предметной области необходимо собрать о ней и создать концептуальную модель этой области. Источниками знаний могут быть документы статьи, книги, Фотографии, киносъемка и многое другое. Из этих источников надо извлечь содержащиеся в них знания, этот процесс может оказаться достаточно трудным, ибо надо заранее оценить важность и нужность тех или иных знаний для работы интеллектуальной системы. Специалисты, которые занимаются всеми вопросами, связанными со знаниями, теперь называются инженерами по знаниям или инженерами знаний, эта новая профессия порождена развитием искусственного интеллекта.

В области извлечения знаний можно выделить два основных направления; Формализации качественных знаний и интеграция знаний. Первое направление связано с созданием разнообразных методов, позволяющих переходить от знаний, выраженных в текстовой Форме к их аналогам, пригодным для ввода в память интеллектуальной системы. В связи с этой проблемой развивались не только традиционные методы обработки экспериментальных данных, но и совершенно новое направление, получившее название нечеткой математики. Возникновение этого направления связано с именем американского специалиста Л. Заде. Нечеткая математика и ее методы оказали существенное влияние на многие области искусственного интеллекта и, в частности, на весь комплекс проблем, связанным с представлением и переработкой качественной информации.

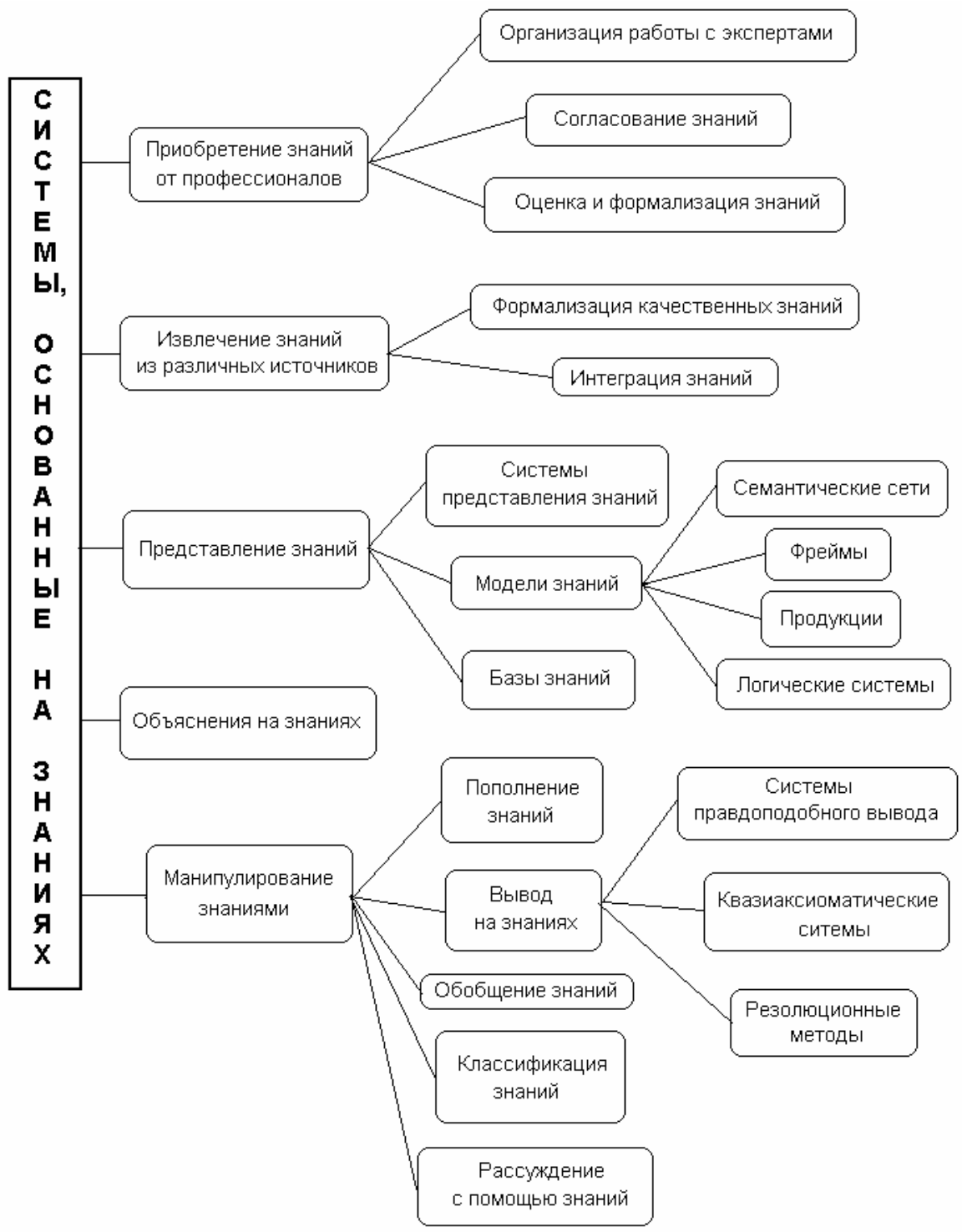

Рисунок 3.1

Когда инженер по знаниям получает знания из различных источников, он должен интегрировать их в некоторую взаимосвязанную и непротиворечивую систему знаний о предметной области. Проблема интеграции знаний пока еще не стоит столь остро, но уже ясно, что без ее решения вряд ли будет возможно создавать представление о предметной области, обладающее теми же богатыми нюансами, которыми оно обладает

для специалиста, знаний, содержащихся в источниках информации, отчужденных от специалиста, как правило, недостаточно.

Значительная часть профессионального опыта остается вне этих источников, в головах у профессионалов, не могущих словесно их выразить. Такие знания часто называют профессиональным умением или интуицией, для того, чтобы приобрести такие знания, нужны специальные приемы и методы. Они используются в инструментальных системах по приобретению знаний, создание которых одна из современных задач инженерии знаний.

Полученные от экспертов знания нужно оценить с точки зрения их соответствия ранее накопленным знаниям и формализовать их для ввода в память интеллектуальной системы. Кроме того, знания, полученные от различных экспертов, надо еще согласовать между собой. Нередки случаи, когда эти знания оказываются внешне несовместимыми и даже противоречивыми. Инженер по знаниям должен путем опроса экспертов устранить эти противоречия.

Следующая большая проблема, изучаемая в искусственном интеллекте - это представление знаний в памяти системы. Для этого, разрабатываются разнообразные модели представления знаний. В настоящее время в интеллектуальных системах используются четыре основных модели знаний.

Первая модель, возможно, наиболее близка к тому, как представляются знания в текстах на естественном языке. В ее основе лежит идея о том, что вся необходимая информация может быть описана как совокупность троек вида; (aRb), где a и b два объекта или понятия, а R - двоичное отношение между ними. Такая модель графически может представляться в виде сети, в которой вершинам соответствуют объекты или понятия, а дугам отношения между ними. Дуги помечены именами соответствующих отношений. Такая модель носит название семантической сети. Впервые такое представление знаний под названием язык синтагматических цепей было, по-видимому, использовано в работах по ситуационному управлению, получивших развитие в СССР в середине шестидесятых годов.

Семантические сети в зависимости от характера отношений, допустимых в них, имеют различную природу. В ситуационном управлении эти отношения, в основном, описывали временные, пространственные и каузальные связи между объектами, а также результаты воздействий на объекты со стороны управляющей системы. В системах планирования и автоматического синтеза программ эти отношения являются связями типа "цель-средство" или "цель-подцель", в классифицирующих системах отношения передают связи по включению объемов понятий (типа "род-вид", "класс-элемент" и т.п.). Распространены так называемые функциональные семантические сети, в которых дуги характеризуют связи вида: "аргумент-Функция". Такие сети используются в качестве моделей вычислительных процессов или моделей функционирования дискретных устройств.

Таким образом, семантические сети - модель весьма широкого назначения. Теория семантических сетей еще не завершена, что привлекает к ним внимание многих специалистов, работающих в искусственном интеллекте. При различных синтаксических ограничениях на структуру семантической сети возникают более жесткие типы представления» Например, реляционные представления, характерные для реляционных баз данных, или клаузальные представления в логике, получившие широкое распространение в машинных методах логического вывода или в языках логического программирования типа языка Пролог.

Семантическая сеть — это модель, используемая для представления знаний в интеллектуальных системах.

Семантическая сеть состоит из вершин и дуг, которые соединяют эти вершины. Вершины соответствуют понятиям, фактам, событиям, процессам и другим элементам описания предметной области. А дуги определяют отношения между этими элементами. Для примера на рисунке приведены две семантические сети. Первая сеть соответствует тексту: "В центре комнаты стоит стол. Слева от него окно. У стола глубокое, удобное кресло. Недалеко от него столик с телефоном. Вторая сеть описывает набор процедур,

необходимых для покупки мороженого в кафе. Если в вершинах первой сети находятся некоторые объекты (стол, комната и т. д.), то в вершинах второй сети названы процедуры. Дуги во второй сети не именованы, так как все они тут имеют одинаковый смысл: "перейти к процедуре".

С помощью семантических сетей можно представлять самые разнообразные знания, но они остались, скорее, теоретической моделью для представления знаний. На практике вместо семантических сетей часто используют продукции и фреймы.

На рис. 3.2 показано сетевое представление совокупности знаний (семантическая сеть), зафиксированных в этом тексте. Понятия и объекты, встречающиеся в тексте, представлены в виде вершин сети, а отношения — в виде дуг, связывающих соответствующие вершины.

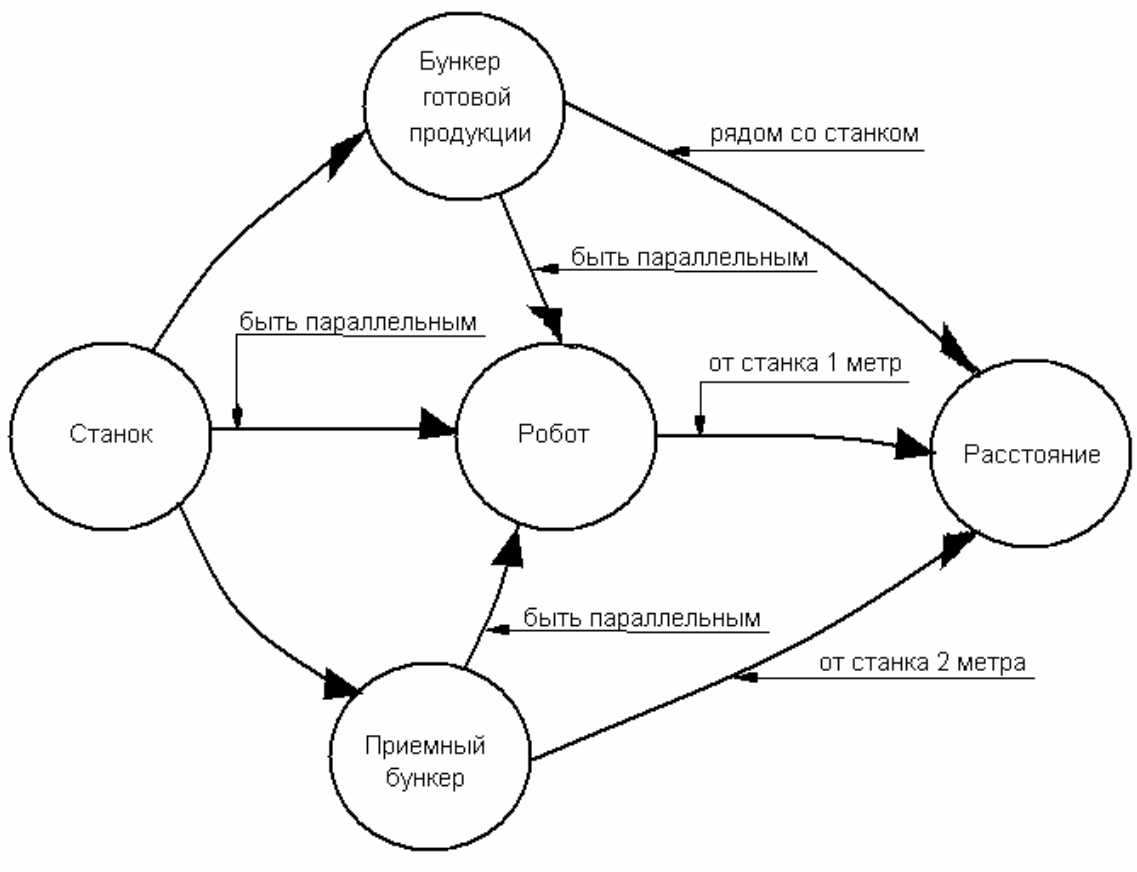

Рисунок 3.2

Известно, что любой текст, описывающий конкретные ситуации в реальном мире, всегда можно представить в виде совокупности

взаимосвязанных понятий. Причем число базовых отношений не может быть бесконечным (оно заведомо меньше 300); все остальные отношения выражаются через базовые в виде их комбинаций. Эта гипотеза служит основой утверждения о том, что семантические сети являются универсальным средством для представления знаний в интеллектуальных системах.

Семантические сети являются весьма мощным средством представления знаний. Однако для них характерны неоднозначность представлений знаний и неоднородность связей. И при автоматизации процесса использования и представления знаний такая неоднозначность и неоднородность заметно усложняют процессы, протекающие в интеллектуальных системах. Поэтому вполне естественно желание как-то унифицировать форму представлений знаний, сделать ее максимально однородной. Одним из способов решения этой задачи в искусственном интеллекте послужил переход к специальному представлению вершин в сети и унификация связей между вершинами.

Все понятия, входящие в сеть, описываются в виде фреймов.

Фрейм — это минимально возможное описание сущности какого-либо явления, события, ситуации, процесса или объекта. (Минимально возможное означает, что при дальнейшем упрощении описания теряется его полнота, оно перестает определять ту единицу знаний, для которой оно предназначено.)

Фрейм имеет почти однородную структуру и состоит из стандартных единиц, называемых слотами. Каждая такая единица — слот — содержит название и свое значение. В качестве примера рассмотрим фрейм для понятия "взятие":

"Взятие":

(Субъект, XI);

(Объект, X2);

(Место, X3);

(Время, X4);

(Условие, X5).

В этом фрейме указаны имена слотов (субъект, объект и т. д.), но вместо их значений стоят переменные (XI, X2 и т. д.). Такой фрейм называется фреймом-прототипом, или протофреймом.

Протофреймы хранят знание о самом понятии. Например, понятие "взять" связано с наличием слотов с указанными именами. Взятие осуществляет XI в месте X3 во время X4, если выполнено условие X5. Берет XI нечто, обозначенное как X2. Подставляя вместо всех переменных конкретные значения, получим конкретный факт-описание:

"Взятие":

(Субъект, Робот);

(Объект, Деталь);

(Место, Приемный бункер);

(Время, X4);

(Условие, В бункере есть деталь, а у робота ее нет).

Такие конкретные описания представляют собой знания, которые совпадают с тем, что в вычислительной технике называют данными. В искусственном интеллекте фреймы, в которых означены все основные слоты (они каким-либо образом помечаются в описании фрейма), называются фреймами-экземплярами, или экзофреймами. В нашем примере, наверное, основными для фрейма "взятие" можно считать слоты с именами "субъект" и "объект". Поскольку в состав фрейма могут входить слоты с именами действий, то фреймы годятся для представления как декларативных, так и процедурных знаний.

Чтобы представить семантическую сеть в виде совокупности фреймов, надо уметь представлять отношения между вершинами сети. Для этого также используются слоты фреймов. Эти слоты могут иметь имена вида "Связь У", где У есть имя того отношения (его тип), которое устанавливает данный фрейм-вершина с другим фреймом-вершиной.

Заметим также, что в качестве значения слота может выступать новый фрейм, что позволяет на множестве фреймов осуществлять иерархическую классификацию. Это очень удобное свойство фреймов, так как человеческие знания, как правило, упорядочены по общности.

В некотором смысле фреймовые представления знаний, широко распространенные в искусственном интеллекте, также являются видом семантических сетей, для перехода к которому надо удовлетворить ряд ограничений синтаксического характера. С понятием «фрейм» в искусственном интеллекте произошла некоторая трансформация смысла. Это понятие было введено в научный оборот М. Минским, который под Фреймом некоторого объекта или явления понимал то его минимальное описание, которое содержит всю существенную информацию об этом объекте или явлении, и обладает тем свойством, что удаление из описания любой его части приводит к потере существенной информации, без которой описание объекта или явления не может быть достаточным для их идентификации.

Однако позже в работах по представлению знаний требование минимальности описания перестали соблюдать и под фреймами стали понимать структуры вида: <имя фрейма (Множество слотов)>. Каждый слот есть пара вида: (Имя слота; Значение слота). Допускается, чтобы слот сам был фреймом. Тогда в качестве значений слота выступает множество слотов. Другими возможностями для заполнения слотов могут быть константы, переменные, любые допустимые выражения в выбранной модели знаний, ссылки на другие слоты и фреймы и т.п. Таким образом, Фрейм представляет собой достаточно гибкую конструкцию, позволяющую отображать в памяти интеллектуальной системы разнообразные знания.

Две другие распространенные модели знаний опираются на классическую логическую модель вывода. Это либо логические исчисления, типа исчисления предикатов и его расширений, либо системы продукций, т.е. правил вида; "Если А, то Б", задающих элементарные шаги преобразований и умозаключений. Эти две модели знаний отличаются явно выраженной

процедурной формой. Поэтому часто говорят, что они описывают процедурные знания, а модели знаний, опирающиеся на семантические сети - описывают декларативные знания. Оба вида знаний могут сосуществовать друг с другом, например, в качестве значений некоторых слотов во Фрейме могут выступать продукции. Именно такие смешанные представления оказываются сейчас в центре внимания исследователей, так как они сулят наиболее хорошие перспективы по представлению знаний.

Третья модель представления знаний носит название ***продукционной***. Основу модели составляют системы продукций. Каждая продукция в наиболее общем виде записывается как стандартное выражение следующего вида:

"Имя продукции":

Имя сферы;

Предусловие;

Условие для ядра;

Если Л, то В;

Постусловие.

Основная часть продукции — ее ядро имеет вид:

"Если Л, то В", где Л и В могут иметь разные значения.

Остальные элементы, образующие продукцию, носят вспомогательный характер. В наиболее простом виде продукция может состоять лишь из имени (например, ее порядкового номера в системе продукций) и ядра. Рассмотрим несколько примеров ядра.

"Если сверкает молния, то гремит гром".

"Если в доме вспыхнул пожар, то вызывайте по телефону 101 пожарную команду".

"Если в путеводителе указано, что в городе есть театр, то надо пойти туда".

Первый пример иллюстрирует тот случай, когда ядро продукции описывает причинно-следственную связь явлений Л и В. Во втором примере

A и B представляют собой некоторые действия. В третьем примере A — это некоторые знания, а B — действие. Возможны и другие варианты ядра продукции. Таким образом, при помощи ядер можно представлять весьма разнообразные знания.

Условие для ядра определяет те необходимые предпосылки, при выполнении которых надо проверять наличие или истинность A в ядре продукции.

Имя сферы указывает ту предметную область, к которой относятся знания, зафиксированные в данной продукции. В интеллектуальной системе может храниться совокупность знаний (ее называют базой знаний), относящихся к разным областям (например, знания о различных заболеваниях человека или знания из различных разделов математики). Ясно, что если в данный момент решается задача из области физики твердого тела или из геометрии треугольника, то надо использовать знания, относящиеся именно к этой области. Сферы и выделяют такие подобласти знаний.

Когда речь шла о различных A и B в ядрах продукций, то практически было показано, что в такой форме можно представлять как декларативные знания, так и процедурные, хотя сама форма продукций весьма удобна для задания именно процедурных знаний.

Логическая модель представляет собой формальную систему — некоторое логическое исчисление, как правило, исчисление предикатов первого порядка (моделирование рассуждений). Все знания о предметной области описываются в виде формул этого исчисления или правил вывода. Описание в виде формул дает возможность представить декларативные знания, а правила вывода — процедурные знания. Рассмотрим в качестве примера знание: "Когда температура в печи достигает 120°C и прошло менее 30 мин с момента включения печи, давление не может превосходить критическое. Если с момента включения печи прошло более 30 мин, то необходимо открыть вентиль №2". Логическая модель представления этого знания имеет вид:

P(p= 120) T(t<30) ->(D<Dкр);

P(p= 120) T(t>30) =>F(№2).

В этой записи использованы следующие обозначения:

P(p=120)—предикат, становящийся истинным, когда температура достигает 120°;

T(t < 30) — предикат, остающийся истинным в течение 30 мин с начала процесса;

T(t > 30) — предикат, становящийся истинным по истечении 30 мин с начала процесса;

D < Dкр — утверждение о том, что давление ниже критического;

F(№2) — команда открыть вентиль №2.

Кроме того, в этих записях использованы типовые логические связки конъюнкции (V), импликации (=>) и логического следования (=>).

Первая строчка в записи представляет декларативные знания, а вторая — процедурные.

Языки представления знаний логического типа широко использовались на ранних стадиях развития интеллектуальных систем, но вскоре были вытеснены (или, во всяком случае, сильно потеснены) языками других типов. Объясняется это громоздкостью записей, опирающихся на классические логические исчисления. При формировании таких записей легко допустить ошибки, а поиск их очень сложен. Отсутствие наглядности, удобочитаемости (особенно для тех, чья деятельность не связана с точными науками) затрудняло распространение языков такого типа.

Куда более наглядными оказались языки, опирающиеся на ***сетевую модель*** представления знаний. В основе такой модели лежит идея о том, что любые знания можно представить в виде совокупности объектов (понятий) и связей (отношений) между ними. Рассмотрим, например, текст, содержащий некоторые декларативные знания: "Слева от станка расположен приемный бункер. Расстояние до него равно 2 м. Справа от станка — бункер готовой

продукции. Он находится рядом со станком. Робот перемещается параллельно станку и бункерам на расстоянии 1 м.".

Перечисленные модели знаний возникли в искусственном интеллекте как бы насильственно, они не опираются на аналоги структур для представления знаний, которыми пользуются люди. Это связано с плохой изученностью форм представления знаний у человека.

Рассмотренные модели представления знаний широко используются в современных интеллектуальных системах и прежде всего в экспертных системах. Каждая из форм представлений знаний может служить основой для создания языка программирования, ориентированного на работу со знаниями. Такими, например, языками являются язык ФРЛ, основанный на фреймовых представлениях, и язык Пролог, опирающийся на модель представления в виде продукций. Однако разные модели представления знаний имеют свои преимущества и свои недостатки. Языки представления знаний, основанные на них, наследуют эти преимущества и недостатки. Поэтому в 80-х гг. наметилась тенденция создавать комбинированные языки представления знаний. Чаще всего комбинируются фреймовые и продукционные модели.

Достаточно богатая предметная область содержит большое количество декларативных и процедурных знаний.

Создание баз знаний большого размера — дело весьма сложное. Ведь необходимо не только накапливать знания, представляя их выбранным способом, но и проверять полноту знаний и их непротиворечивость. Источниками знаний могут быть книги, документы, изобразительная продукция, устные тексты, получаемые от специалистов, и т. п. Эти различные источники знаний надо уметь объединять между собой, что приводит к сложным, интегрированным базам знаний. Отдельные базы знаний, территориально разнесенные между собой, могут совместно использоваться при решении задач. Так возникают распределенные базы знаний, образуются сложные по конфигурации сети баз знаний. Уже сейчас

такие сети хранения и обработки знаний становятся не только общенациональными, но и международными, доступными любому специалисту.

Стремительное развитие баз знаний и процедур, связанных с их заполнением, ведением и использованием, породило новую профессию, получившую название инженер по знаниям. Инженер по знаниям — специалист высокого класса, владеющий системным программированием и методами искусственного интеллекта. Без инженеров по знаниям практически нельзя создать интеллектуальную систему. Как и программисты, инженеры по знаниям становятся необходимой фигурой в эпоху новых информационных технологий.

В интеллектуальных системах для хранения и использования знаний создаются специальные представления знаний, включающие в свой состав всю совокупность процедур, необходимых для записи знаний, извлечения их из памяти и поддержки хранилища знаний в рабочем состоянии. Системы представления знаний часто оформляются как базы знаний, являющиеся естественным развитием баз данных. Теория баз знаний составляет заметную часть современного искусственного интеллекта. Именно в них сосредотачиваются сейчас все основные процедуры манипулирования знаниями.

Среди этих процедур можно, прежде всего, отметить процедуры пополнения знаний. Все человеческие знания, содержащиеся в текстах, таковы, что они принципиально не полны, воспринимая тексты, мы как он пополняем их за счет той информации, которая нам известна и которая имеет отношения к данному тексту. Аналогичные процедуры должны происходить и в базы знаний, новые знания, поступающие в них, должны вместе с теми сведениями, которые уже были ранее записаны в базу, сформировать расширение поступивших знаний. Это и есть процедуры пополнения знаний. Среди этих процедур особое место занимает псевдофизические логики

(времени, пространства, действий и т.п.), которые, опираясь на законы внешнего мира, пополняют поступающую в базы знаний информацию.

Знания, как и у человека, в интеллектуальных системах хранятся небессистемно. Они образуют некоторые упорядоченные структуры, что облегчает поиск нужных знаний и поддержание работоспособности баз знаний, для этого используются различные классифицирующие процедуры, типы классификаций могут быть весьма различными. Это могут быть родовидовые классификации, классификации типа "часть-целое" или ситуативные классификации, когда в одно множество объединяются все те знания, которые релевантны некоторой типовой ситуации, В этой области исследования по искусственному интеллекту тесно соприкасаются с теорией классификации, давно существующей, как некоторая самостоятельная ветвь науки.

В процессе классификации часто происходит абстрагирование от отдельных элементов описаний, от отдельных фрагментов знаний об объектах или явлениях. Это приводит к появлению обобщенных знаний. Обобщение может идти на несколько шагов, что приводит, в конце концов, к абстрактным знаниям, для которых нет прямого прообраза во внешнем мире. Манипулирование абстрактными знаниями повышает интеллектуальные возможности систем, делая эти манипуляции весьма общими по своим свойствам и результатам. Обобщение знаний и Формирование понятий в системах искусственного интеллекта - одно из активно развивающихся направлений, в котором работает немало специалистов.

Особое место занимают процедуры, связанные с выводом на знаниях, получениям на основании имеющихся знаний новых знаний. Вывод на знаниях зависит от той модели, которая используется для их представления. Если в качестве представления используются логические системы или продукции, то вывод на знаниях становится весьма близок к стандартному логическому выводу. Это же происходит при представлении знаний в клаузальной Форме. Во всех случаях в интеллектуальных системах

используются методы вывода, опирающиеся на идеи метода резолюций или на идею обратного вывода С. Ю. Маслова (как на языке Пролог при клаузальной Форме представления).

Но простое заимствование идей и методов математической логики, под знаком чего происходило развитие работ в искусственном интеллекте в семидесятых годах, не привело к сколь-нибудь значительным результатам. Основное отличие баз знаний и баз данных интеллектуальных систем от тех объектов, с которыми имеет дело формальная логическая система, это их открытость, возможность появления в памяти интеллектуальной системы новых файлов, новых сведений приводит к тому, что начинает разрушаться принцип монотонности, лежащий в основе функционирования всех систем, изучаемых математической логикой, согласно этому принципу, если некоторое утверждение выводится в данной системе, то никакие дополнительные сведения не могут изменить этот факт. В открытых системах это не так. Новые сведения могут изменить ситуацию, и сделанный ранее вывод может стать неверным.

Немонотонность вывода в открытых системах вызывает немалые трудности при организации вывода на знаниях. В последнее десятилетие сторонники логических методов в искусственном интеллекте делают попытки построить новые логические системы, в рамках которых можно было бы обеспечить немонотонный вывод, но на этом пути больше трудностей, чем результатов, И дело не только не в монотонности вывода. По сути, системы, с помощью которых представляются знания о предметных областях, не являются строго аксиоматическими, как классические логические исчисления, в этих последних аксиомы описывают извечные истины, верные для любых предметных областей, А в интеллектуальных системах каждая предметная область использует свои, специфические, верные только в ней утверждения, поэтому и системы, которые возникают при таких условиях, следует называть квазиаксиоматическими. В таких

системах вполне возможна смена исходных аксиом в процессе длительного вывода, и» как следствие этого, изменение этого вывода.

И, наконец, еще одна особенность вывода на знаниях, доставляющая немало забот исследователям, занятым Формированием решений в интеллектуальных системах. Это неполнота сведений о предметной области и протекающих в ней процессах, неточность входной информации, не совсем полная уверенность в квазиаксиомах. А это означает, что выводы в интеллектуальных системах носят не абсолютно достоверный характер, как в традиционных логических системах, а приближенный, правдоподобный характер. Такие выводы требуют развитого аппарата вычисления оценок правдоподобия и методов оперирования с ними. Подобные исследования сейчас в искусственном интеллекте развиваются широким Фронтом, по сути, рождается новая теория вывода, в которую лишь как небольшая часть входит достоверный вывод, изучавшийся в течение многие десятилетий логиками.

В интеллектуальных системах специалисты стремятся отразить основные особенности человеческих рассуждений, опыт тех специалистов, которые обладают профессиональными умениями, пока не полностью доступными искусственным системам, вывод: всего лишь одна из форм того, как человек приходит к нужным ему заключениям, поэтому бурно развивается та область, которую в искусственном интеллекте называют моделированием человеческих рассуждений.

К ним относятся аргументация на основе имеющихся знаний, рассуждения по аналогии и ассоциации, оправдание заключения в системе имеющихся прагматических ценностей и многое другое, чем люди пользуются в своей практике. Привнесение всех этих приемов в интеллектуальные системы, без сомнения, сделает их рассуждения более гибкими, успешными и человечными.

Когда некто высказывает свое мнение, то для того, чтобы согласиться или не согласиться с ним, необходимо знать те же основания, которые лежат в его основе. Если эти основания неизвестны, то имеется возможность

попросить оппонента объяснить, как он пришел к своему мнению. Аналогичная функция возникла и в интеллектуальных системах. Поскольку они принимают свои решения, опираясь на знания, которые могут быть неизвестны пользователю, решающему свою задачу с помощью интеллектуальной системы, то он может усомниться в правильности полученного решения. Интеллектуальная система должна обладать средствами, которые могут сформировать пользователю необходимые объяснения. Объяснения могут быть различного типа. Они могут касаться самого процесса получения решения оснований, которые были для этого использованы, способов отсечения альтернативных вариантов и т.п. Все это требует развитой теории объяснения, что стимулирует сейчас активность исследований в этом направлении.

ТЕМА № 4. ТЕХНОЛОГИЯ И ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА ПРОЕКТИРОВАНИЯ ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ

Общая структура исследований в области интеллектуального программирования показана на рис. 4.1.

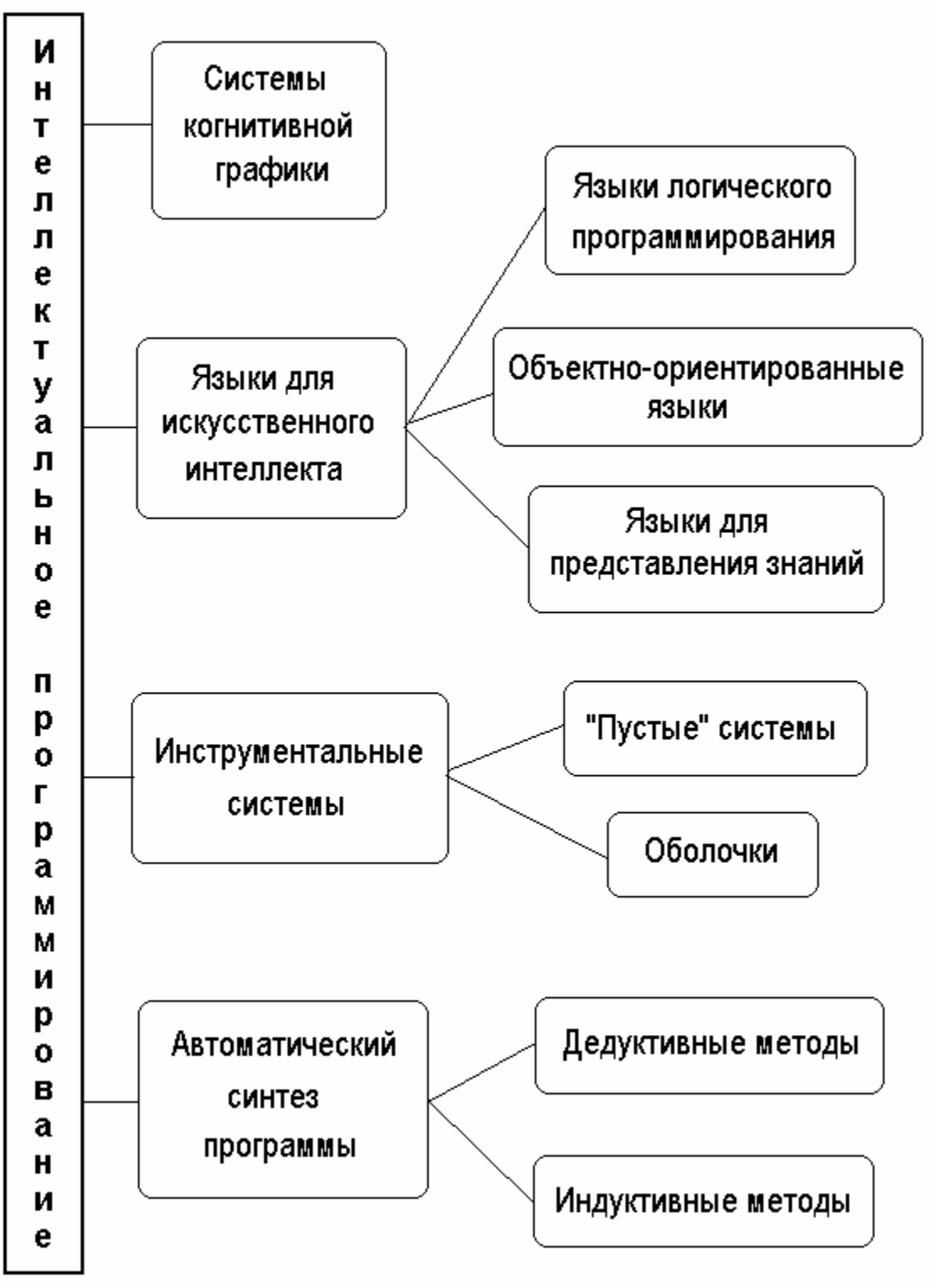

Рисунок 4.1

Среди огромного количества языков программирования, созданных для различных целей, в работах по искусственному интеллекту используется мизерная часть. В подавляющем большинстве случаев из ранее известных языков используется Лисп и все его многочисленные версии. Куда меньше

используется Паскаль или Си, Зато именно искусственный интеллект породил пролог (в самом начале он был стимулирован к жизни работами в области машинного перевода, но Р. Ковальский понял, что в идее клаузальных логических выражений и в процедурах вывода кроется куда более общая идея). Вместе с прологом родилось и некоторое семейство языков программирования, в основе которых лежит идея логического вывода. На сегодняшний день все специалисты в области искусственного интеллекта за небольшим исключением работают на Лиспе или на Прологе.

Языки логического программирования расцветали вместе с идеей, что логический вывод есть основа всех процессов решения задач в искусственном интеллекте (расцвет Лиспа был связан с предшествующим этапом развития искусственного интеллекта, когда считалось, что основой всех процессов решения задач являются хорошо организованный перебор и поиск). Сейчас начинает казаться, что на смену языкам типа Пролога должны придти языки программирования, в которых основной конструкцией является объект и его свойства и признаки. Такие объектно-ориентированные языки уже появились, наиболее известным их представителем является Смолток. Парадигма, связанная с языками такого типа состоит в том, что решение задач может мыслиться, как манипулирование с понятиями, обобщающими объекты с помощью которые описываются предметные области.

Особняком стоят языки для представления знаний. Таковы языки, ориентированные на Фреймы KL-1, KRL_, FRL_, или язык Пилот, ориентированный на модель знаний в виде продукций. Пока таких языков немного и они не достигли еще уровня, на котором можно конкурировать с мощным программным окружением таких языков, как Лисп или Пролог.

Весьма не быстро развиваются работы, связанные с автоматическим программированием. Обе ветви этих работ: создание нужных программ из готовых модулей по специфицированному описанию исходной задачи на основании некоторой дедуктивной системы, осуществляющей синтез путем

процедуры, напоминающей логический вывод (программа как бы "извлекается" из траектории вывода) и индуктивный способ генерирования программ на основе обучения на множестве примеров, пока дали достаточно скромные результаты. Но энтузиасты рассчитывают, что трудности преодолимы.

Наиболее бурно развиваются работы в области создания инструментальных систем, предназначенных для быстрого проектирования и разработки самых разнообразных интеллектуальных систем. Общая идея тут состоит в том, чтобы создать некоторую систему-прототип, затратив на ее создание достаточно много усилий. Но затем использовать для решения задач в конкретной предметной области.

Если в системе-прототипе заранее зафиксированы все средства заполнения базы знаний и манипулирования знаниями в ней, но сама база знаний не заполнена, то такая инструментальная система называется "пустой". Чтобы "настроить" ее на некоторую предметную область, инженер по знаниям должен, используя готовую форму представления знаний, ввести в базу знаний всю необходимую информацию о предметной области. После этого система-прототип превращается в готовую интеллектуальную систему.

Сейчас интерес к "пустым" системам резко уменьшился. Оказалось, что даже для однотипных предметных областей переход из одной области к другой может потребовать модификации тех или иных средств для манипулирования знаниями, а иногда и Формы представления знаний. Поэтому основные усилия разработчиков сейчас направлены на создание систем-оболочек, В таких инструментальных системах имеется возможность при переходе к конкретной системе варьировать в достаточно широких пределах Форму представления знаний и способы манипулирования ими. Конечно, такая гибкость требует больших затрат на создание системы-оболочки. Но эти затраты оправданы, В этой области уже получены' интересные результаты, и именно здесь программисты, специализирующиеся

в области интеллектуальных программных систем, ожидают большого прорыва в интеллектуальное программирование завтрашнего дня.

Системы когнитивной графики - новое направление в интеллектуальном программировании. В персональных ЭВМ имеются достаточно развитые графические средства, но они как бы оторваны от остальных средств, существуют автономно. Но одна из центральных идей искусственного интеллекта - это идея о том, что суть самого феномена интеллекта состоит в совместной работе двух систем переработки информации, зрительной, создающей образную картину мира, и символической, способной к абстрактному мышлению, к оперированию с понятиями, интегрирующими образы внешнего мира. Возможность перехода от зрительной картины к ее текстовому (символическому) описанию, и от текста к некоторой зрительной картине, могущей быть представительницей этого текста, составляют, по-видимому, основу того, что называется мышлением.

Символьная система изучена достаточно хорошо. Успехи искусственного интеллекта определяются тем, что символьная система достаточно хорошо изучена и промоделирована в искусственных системах. Со зрительной системой дело обстоит хуже. Мы пока еще не слишком много знаем о том, как хранятся зрительные образы в памяти человека, как они обрабатываются и, что самое главное, как они соотносятся с текстами, им соответствующими.

Когнитивная графика как раз и занимается приемами соотнесения текстов и зрительных картин через общее представление знаний, интегрирующих тексты и зрительные образы. Это направление в искусственном интеллекте признается всеми, как весьма перспективное. Его развитие даст новый виток развития наших представлений о способах решения задач, стимулирующих развитие систем параллельной обработки, создаст предпосылки перехода к новой технологии решения задач.

Пока в этом направлении делаются первые шаги, появились программы, которые занимаются оживлением картин на экране дисплея, первые программы, в которых оживление происходит не на основании жестких процедур, как при создании мультипликации, а в соответствии с некоторыми текстами на ограниченном естественном языке, вводимых в систему. Но исследования в этом направлении неуклонно расширяются, и специалисты считают, что в девяностых годах в этой области будут получены новые фундаментальные результаты.

Типичная статическая ЭС состоит из следующих основных компонентов (рис. 4.2):

- ☑ решателя (интерпретатора);
- ☑ рабочей памяти (РП), называемой также базой данных (БД);
- ☑ базы знаний (БЗ);
- ☑ компонентов приобретения знаний;
- ☑ объяснительного компонента;
- ☑ диалогового компонента.

База данных (рабочая память) предназначена для хранения исходных и промежуточных данных решаемой в текущий момент задачи. Этот термин совпадает по названию, но не по смыслу с термином, используемым в информационно-поисковых системах (ИПС) и системах управления базами данных (СУБД) для обозначения всех данных (в первую очередь долгосрочных), хранимых в системе.

База знаний (БЗ) в ЭС предназначена для хранения долгосрочных данных, описывающих рассматриваемую область (а не текущих данных), и правил, описывающих целесообразные преобразования данных этой области.

Решатель, используя исходные данные из рабочей памяти и знания из БЗ, формирует такую последовательность правил, которые, будучи примененными к исходным данным, приводят к решению задачи.

Компонент приобретения знаний автоматизирует процесс наполнения ЭС знаниями, осуществляемый пользователем-экспертом.

Объяснительный компонент объясняет, как система получила решение задачи (или почему она не получила решение) и какие знания она при этом использовала, что облегчает эксперту тестирование системы и повышает доверие пользователя к полученному результату.

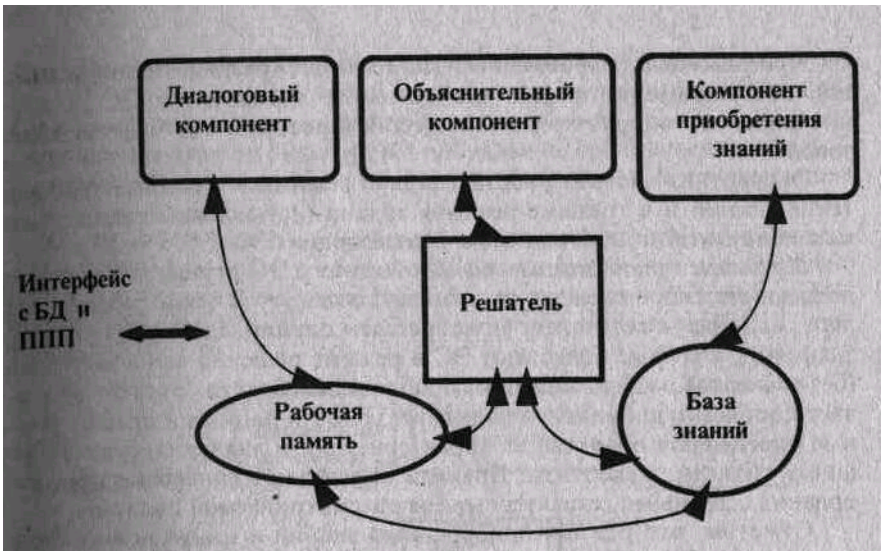

Рисунок. 4.2. Структура статической ЭС

Диалоговый компонент ориентирован на организацию дружественного общения с пользователем как в ходе решения задач, так и в процессе приобретения знаний и объяснения результатов работы.

В разработке ЭС участвуют представители следующих специальностей:

- эксперт в проблемной области, задачи которой будет решать ЭС;
- инженер по знаниям - специалист по разработке ЭС (используемые им технологию, методы называют технологией (методами) инженерии знаний);
- программист по разработке инструментальных средств (ИС), предназначенных для ускорения разработки ЭС.

Необходимо отметить, что отсутствие среди участников разработки инженеров по знаниям (т. е. их замена программистами) либо приводит к неудаче процесс создания ЭС, либо значительно удлиняет его.

Эксперт определяет знания (данные и правила), характеризующие проблемную область, обеспечивает полноту и правильность введенных в ЭС знаний.

Инженер по знаниям помогает эксперту выявить и структурировать знания, необходимые для работы ЭС; осуществляет выбор того ИС, которое наиболее подходит для данной проблемной области, и определяет способ представления знаний в этом ИС; выделяет и программирует (традиционными средствами) стандартные функции (типичные для данной проблемной области), которые будут использоваться в правилах, вводимых экспертом.

Программист разрабатывает ИС (если ИС разрабатывается заново), содержащее в пределе все основные компоненты ЭС, и осуществляет его сопряжение с той средой, в которой оно будет использовано.

Экспертная система работает в двух режимах: режиме приобретения знаний и в режиме решения задачи (называемом также режимом консультации или режимом использования ЭС).

В режиме приобретения знаний общение с ЭС осуществляет (через посредничество инженера по знаниям) эксперт. В этом режиме эксперт, используя компонент приобретения знаний, наполняет систему знаниями, которые позволяют ЭС в режиме решения самостоятельно (без эксперта) решать задачи из проблемной области. Эксперт описывает проблемную область в виде совокупности данных и правил. Данные определяют объекты, их характеристики и значения, существующие в области экспертизы. Правила определяют способы манипулирования с данными, характерные для рассматриваемой области.

Отметим, что режиму приобретения знаний в традиционном подходе к разработке программ соответствуют этапы алгоритмизации,

программирования и отладки, выполняемые программистом. Таким образом, в отличие от традиционного подхода в случае ЭС разработку программ осуществляет не программист, а эксперт (с помощью ЭС), не владеющий программированием.

В режиме консультации общение с ЭС осуществляет конечный пользователь, которого интересует результат и (или) способ его получения. Необходимо отметить, что в зависимости от назначения ЭС пользователь может не быть специалистом в данной проблемной области (в этом случае он обращается к ЭС за результатом, не умея получить его сам), или быть специалистом (в этом случае пользователь может сам получить результат, но он обращается к ЭС с целью либо ускорить процесс получения результата, либо возложить на ЭС рутинную работу). Следует подчеркнуть, что термин "пользователь" является многозначным, так как использовать ЭС кроме конечного пользователя может и эксперт, и инженер по знаниям, и программист. Поэтому когда хотят подчеркнуть, что речь идет о том, для кого делалась ЭС, используют термин "конечный пользователь".

В режиме консультации данные о задаче пользователя после обработки их диалоговым компонентом поступают в рабочую память. Решатель на основе входных данных из рабочей памяти, общих данных о проблемной области и правил из БЗ формирует решение задачи. Подчеркнем, что в отличие от традиционных программ ЭС при решении задачи не только исполняет предписанную последовательность операции, но и предварительно формирует ее. Если реакция системы не понятна пользователю, то он может потребовать объяснения "Почему система задает тот или иной вопрос?", "как ответ, собираемый системой, получен?".

Структуру, приведенную на рис. 4.2, называют структурой статической ЭС. ЭС данного типа используются в тех приложениях, где можно не учитывать изменения окружающего мира, происходящие за время решения задачи. Первые ЭС, получившие практическое использование, были статическими. Они нашли применение в широком классе приложений.

Из общих соображений понятно, что существует огромный класс приложений, в которых требуется учитывать динамику, т. е. изменения, происходящие в окружающем мире за время исполнения приложения. На рис. 4.3 показано, что в архитектуру динамической ЭС по сравнению со статической ЭС вводятся два компонента: подсистем; моделирования внешнего мира и подсистема связи с внешним окружением.

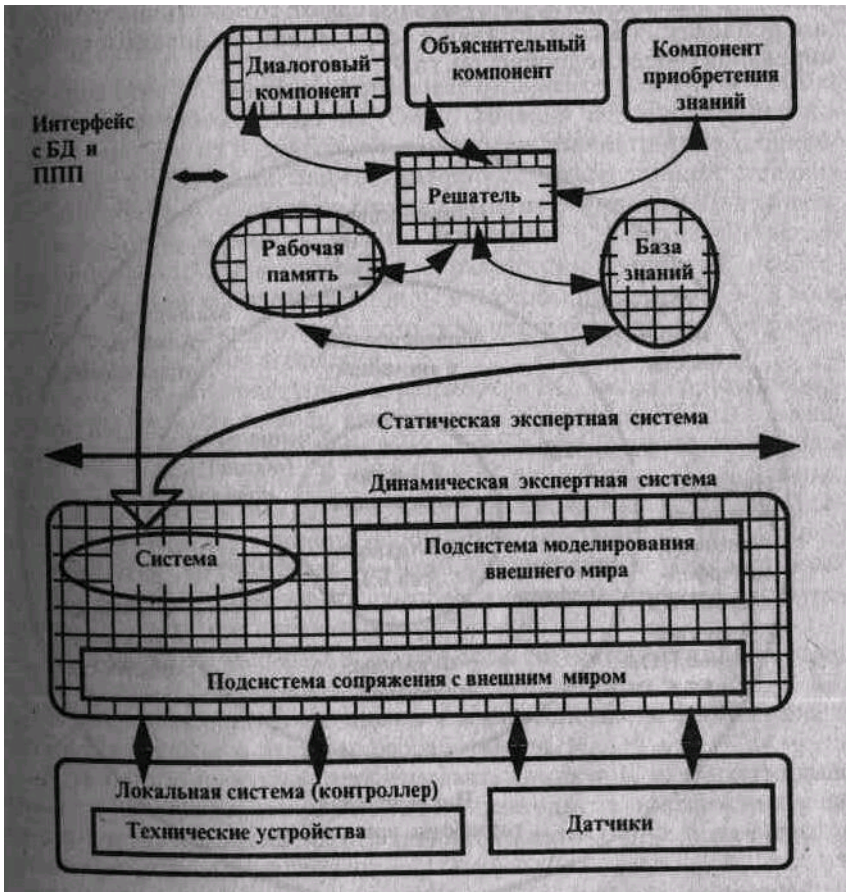

Рисунок 4.3. Архитектура статических и динамических ЭС (компоненты, подвергающиеся изменениям, заштрихованы)

Последняя осуществляет связи с внешним миром через систему датчиков и контроллеров. Кроме того, традиционные компоненты статической ЭС (база знаний и машина вывода) претерпеваю; существенные изменения, чтобы отразить временную логику происходящих в реальном мире событий (подробнее см. гл.9 из [14]).

Подчеркнем, что структура ЭС, представленная на рис. 3.9 и 3.10 отражает только компоненты (функции), и многое остается "за кадром". На

рис. 4.4 приведена обобщенная структура современного инструментального средства (ИС) для создания динамических ЭС, содержащая кроме основных компонентов те возможности, которые позволяют создавать интегрированные приложения в соответствии с современной технологией.

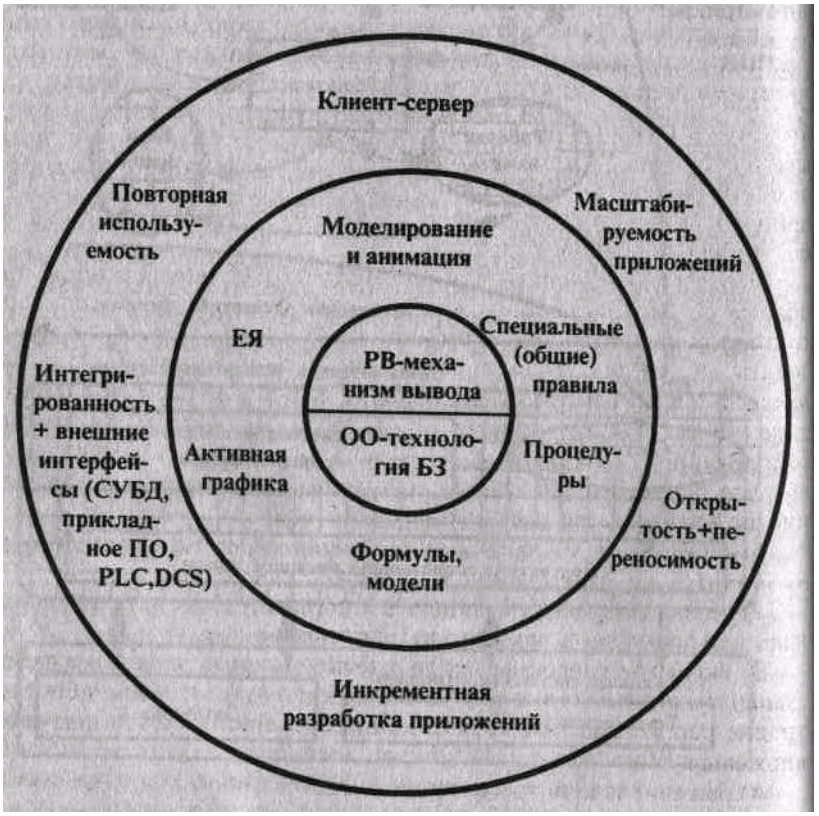

Рисунок 4.4. Структура современных инструментальных средств.

В основе ИС лежат объектно-ориентированная база знаний (00-технология БЗ) и механизм вывода, способный оперировать с правилами в которых явным образом отражено время (РВ - механизм вывода). Во внутреннем кольце расположены компоненты, обеспечивающие моделирование, анимацию, активную графику, механизм общих правил и т.д. Во внешнем кольце отражены технологии и требования, обязательные в современных ИС для создания ЭС.

Разработка ЭС имеет существенные отличия от разработки обычного программного продукта. Опыт создания ЭС показал, что использование при их разработке методологии, принятой в традиционном программировании, либо чрезмерно затягивает процесс создания ЭС, либо вообще приводит к

отрицательному результату. Дело в том, что неформализованность задач, решаемых ЭС, отсутствие завершенной теории ЭС и методологии их разработки приводят к необходимости модифицировать принципы и способы построения ЭС в ходе процесса разработки по мере того, как увеличивается знание, разработчиков о проблемной области.

Перед тем как приступить к разработке ЭС, инженер по знаниям должен рассмотреть вопрос, следует ли разрабатывать ЭС для данного приложения. В обобщенном виде ответ может быть таким: использовать ЭС следует только тогда, когда разработка ЭС возможна, оправдана и методы инженерии знаний соответствуют решаемой задаче. Ниже будут уточнены использованные понятия "возможно", "оправдано", "соответствует".

Чтобы разработка ЭС была возможной для данного приложения, необходимо одновременное выполнение по крайней мере следующих требований:

1) существуют эксперты в данной области, которые решают задачу значительно лучше, чем начинающие специалисты;

2) эксперты сходятся в оценке предлагаемого решения, иначе нельзя будет оценить качество разработанной ЭС;

3) эксперты способны вербализовать (выразить на естественном языке) и объяснить используемые ими методы, в противном случае трудно рассчитывать на то, что знания экспертов будут "извлечены" и вложены в ЭС;

4) решение задачи требует только рассуждений, а не действий;

5) задача не должна быть слишком трудной (т.е. ее решение должно занимать у эксперта несколько часов или дней, а не недель);

6) задача хотя и не должна быть выражена в формальном виде, но все же должна относиться к достаточно "понятной" и структурированной области, т.е. должны быть выделены основные понятия, от ношения и известные (хотя бы эксперту) способы получения решении задачи;

7) решение задачи не должно в значительной степени использовать "здравый смысл" (т.е. широкий спектр общих сведений о мире и способе его функционирования, которые знает и умеет использовав любой нормальный человек), так как подобные знания пока не удается (в достаточном количестве) вложить в системы искусственного интеллекта.

Использование ЭС в данном приложении может быть возможно, но не оправдано.

Применение ЭС может быть оправдано одним из следующих факторов:

- решение задачи принесет значительный эффект, например экономический;
- использование человека-эксперта невозможно либо из-за недостаточного количества экспертов, либо из-за необходимости выполнять экспертизу одновременно в различных местах;
- использование ЭС целесообразно в тех случаях, когда при передаче информации эксперту происходит недопустимая потеря времени или информации;
- использование ЭС целесообразно при необходимости решать задачу в окружении, враждебном для человека.

Приложение соответствует методам ЭС, если решаемая задача обладает совокупностью следующих характеристик:

1) задача может быть естественным образом решена посредством манипуляции с символами (т.е. с помощью символических рассуждений), а не манипуляций с числами, как принято в математических методах и в традиционном программировании;

2) задача должна иметь эвристическую, а не алгоритмическую природу, т.е. ее решение должно требовать применения эвристических правил. Задачи, которые могут быть гарантированно решены (с соблюдением заданных ограничений) с помощью некоторых формальных процедур, не подходят для применения ЭС;

3) задача должна быть достаточно сложна, чтобы оправдать затраты на разработку ЭС. Однако она не должна быть чрезмерно сложной (решение занимает у эксперта часы, а не недели), чтобы ЭС могла ее решать;

4) задача должна быть достаточно узкой, чтобы решаться методами ЭС, и практически значимой.

При разработке ЭС, как правило, используется концепция "быстрого прототипа". Суть этой концепции состоит в том, что разработчики не пытаются сразу построить конечный продукт. На начальном этапе они создают прототип (прототипы) ЭС. Прототипы должны удовлетворять двум противоречивым требованиям: с одной стороны они должны решать типичные задачи конкретного приложения а с другой - время и трудоемкость их разработки должны быть весьма незначительны, чтобы можно было максимально запараллелить процесс накопления и отладки знаний (осуществляемый экспертом) с процессом выбора (разработки) программных средств (осуществляемым инженером по знаниям и программистом). Для удовлетворения указанным требованиям, как правило, при создании прототипа используются разнообразные средства, ускоряющие процесс проектирования.

Прототип должен продемонстрировать пригодность методов инженерии знаний для данного приложения. В случае успеха эксперт с помощью инженера по знаниям расширяет знания прототипа о проблемной области. При неудаче может потребоваться разработка нового прототипа или разработчики могут прийти к выводу о непригодности методов ЭС для данного приложения. По мере увеличения знаний прототип может достигнуть такого состояния, когда он успешно решает все задачи данного приложения. Преобразование прототипа ЭС в конечный продукт обычно приводит к перепрограммированию ЭС на языках низкого уровня, обеспечивающих как увеличение быстродействия ЭС, так и уменьшение требуемой памяти. Трудоемкость и время создания ЭС в значительной степени зависят от типа используемого инструментария.

В ходе работ по созданию ЭС сложилась определенная технология их разработки, включающая шесть следующих этапов (рис. 4.5):

- идентификацию,
- концептуализацию,
- формализацию,
- выполнение,
- тестирование,
- опытную эксплуатацию.

На этапе идентификации определяются задачи, которые подлежат решению, выявляются цели разработки, определяются эксперты и типы пользователей.

На этапе концептуализации проводится содержательный анализ проблемной области, выявляются используемые понятия и их взаимосвязи, определяются методы решения задач.

На этапе формализации выбираются ИС и определяются способы представления всех видов знаний, формализуются основные понятия, определяются способы интерпретации знаний, моделируется работа системы, оценивается адекватность целям системы зафиксированных понятий, методов решений, средств представления и манипулирования знаниями.

На этапе выполнения осуществляется наполнение экспертом базы знаний. В связи с тем, что основой ЭС являются знания, данный этап является наиболее важным и наиболее трудоемким этапом разработки. Процесс приобретения знаний разделяют на извлечение знаний из эксперта, организацию знаний, обеспечивающую эффективную работу системы, и представление знаний в виде, понятном ЭС. Процесс приобретения знаний осуществляется инженером по знаниям на основе анализа деятельности эксперта по решению реальных задач.

Рисунок 4.5. Технология разработки ЭС.

На этапе тестирования эксперт (и инженер по знаниям) в интерактивном режиме с использованием диалоговых и объяснительных средств системы проверяет компетентность ЭС. Процесс тестировании продолжается до тех пор, пока эксперт не решит, что система достигла требуемого уровня компетентности.

На этапе опытной эксплуатации проверяется пригодность ЭС дли конечных пользователей. По результатам этого этапа может потребоваться существенная модификация ЭС.

Процесс создания ЭС не сводится к строгой последовательности перечисленных выше этапов. В ходе разработки приходится неоднократно возвращаться на более ранние этапы и пересматривать принятые там решения.

Если, например, человек обращается к экспертной системе, предназначенной для помощи при организации свободного времени, с вопросом:

"Где в этом городе провести вечер?", то система может спросить его: "А какой вид досуга вы предпочитаете?" Если пользователь сообщит системе, что он предпочитает сходить в дискотеку, то система обратится к

соответствующему сценарию, похожему на тот, что приведен на нашем рисунке. Из этого сценария она извлечет информацию о том, что, прежде всего, надо выяснить наличие дискотеки в городе. После этого система обращается к своей базе знаний и ищет там нужную информацию. Найдя ее, она сообщает человеку адрес и телефон дискотеки. Теперь наступает очередь человека. Он звонит по телефону и узнает, что сегодня дискотека не работает. Эту информацию он вводит в систему. После чего система должна задать ему очередной вопрос о предпочтениях. Если человек называет новый вариант (например, вечер в кино), то система выполняет работу, аналогичную уже описанной. Но человек может воспользоваться и системой "меню", задав вопрос: "А где, вообще, можно провести в этом городе вечер?" В ответ на этот вопрос система предлагает пользователю все "меню" — перечень возможных, описанных в сценарии вариантов времяпрепровождения.

В нашем несложном примере уже видно, как происходит разделение сфер участия человека и машины в решении общей задачи. Но и в общем случае картина выглядит примерно так же. Типовая схема организации интеллектуального пакета прикладных программ представляет из себя следующее: в памяти модулей хранятся готовые программы, с помощью которых можно решать те или иные задачи. Если, например, система предназначена для решения систем линейных алгебраических уравнений, то каждый такой модуль может решить эту задачу своим методом. Один из них, например, решает систему уравнений методом последовательного исключения неизвестных (метод Гаусса), другой для решения той же задачи применяет метод простой итерации, третий — использует итерацию Зейделя и т. д. Пользователь вводит в систему исходную систему уравнений и указывает метод, с помощью которого он хочет решать свою задачу, если такой метод ему известен. Блок, осуществляющий сборку программы, находит нужный модуль в памяти и использует его для поиска решения. Если пользователь не знает, каким именно методом он будет решать задачу, то с

помощью базы знаний система либо сама выбирает автоматически нужный метод, либо выдает пользователю список методов в виде "меню".

Опытный пользователь сможет из этого "меню" выбрать метод решения и указать его системе. Менее опытный может попросить систему дать пояснения к "меню". Эти пояснения могут касаться как сути методов, так и условий их применимости. Постепенно, взаимодействуя с пользователем через интеллектуальный интерфейс, система получает информацию о методе, который будет использоваться при решении. Наконец, возможен случай, когда пользователь совершенно не подготовлен к профессиональному общению с системой. Тогда она сама может выбрать метод решения, опираясь на сценарий, хранящийся в блоке сборки программ.

Убедившись, что выбранный метод (например, метод простой итерации) при достаточно большом числе шагов не дает нужной скорости сходимости, система может автоматически перейти к другому методу.

Решение системы линейных алгебраических уравнений может и не быть самоцелью, а входить в состав некоторой деятельности, осуществляемой совместно человеком и машиной. Например, решение этой задачи может входить в процесс проектирования и расчета какой-либо электрической схемы. Тогда человек рассматривает различные электрические схемы, отличающиеся друг от друга своей структурой и значениями составляющих ее сопротивлений, индуктивностей, емкостей, а система рассчитывает токи и напряжения в узлах этих схем, что позволяет человеку найти наиболее удачное проектное решение.

Именно в системах автоматизированного проектирования наиболее часто используются человеко-машинные процедуры совместного решения задач. Как правило, на долю человека в таких системах падает часть работы, связанная с поиском новых технических решений, а машина помогает ему оценивать эти решения, производя необходимые расчеты.

ТЕМА № 5. МЕТОДЫ ИЗВЛЕЧЕНИЯ ЗНАНИЙ

Рисунок 5.1 иллюстрирует предлагаемую классификацию методов извлечения знаний, в которой используются наиболее употребительные термины, что позволит инженерам по знаниям в зависимости от конкретной задачи и ситуации выбрать подходящий метод.

Из предложенной схемы классификации видно, что основной принцип деления связан с источником знаний.

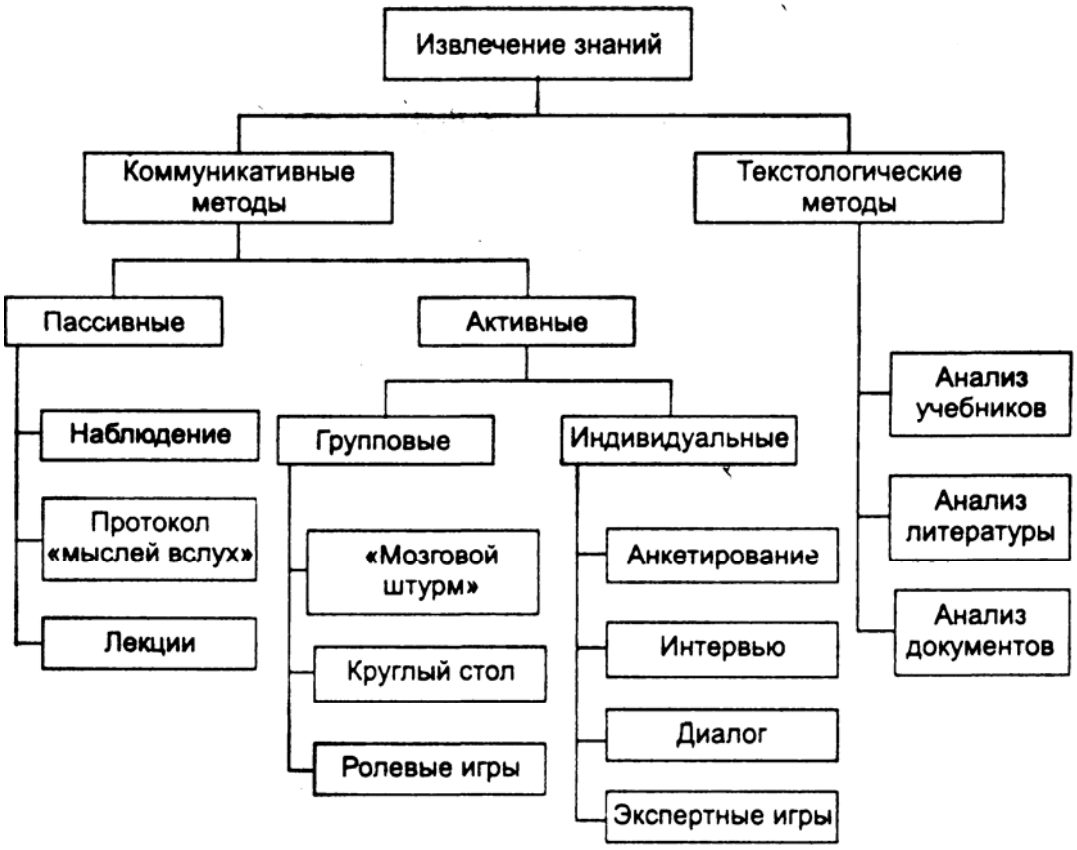

Рис. 5.1. Классификация методов извлечения знаний

Коммуникативные методы извлечения знаний охватывают методы и процедуры контактов инженера по знаниям с непосредственным источником знаний — экспертом, а текстологические включают методы извлечения знаний из документов (методик, пособий, руководств) и специальной литературы (статей, монографий, учебников).

Разделение этих групп методов на верхнем уровне классификации не означает их антагонистичности, обычно инженер по знаниям комбинирует различные методы, например сначала изучает литературу, затем беседует с экспертами, или наоборот.

В свою очередь, коммуникативные методы можно также разделить на две группы: активные и пассивные. *Пассивные* методы подразумевают, что ведущая роль в процедуре извлечения как бы передается эксперту, а инженер по знаниям только протоколирует рассуждения эксперта во время его реальной работы по принятию решений или записывает то, что эксперт считает нужным самостоятельно рассказать в форме лекции. В *активных* методах, напротив, инициатива полностью в руках инженера по знаниям, который активно контактирует с экспертом различными способами — в играх, диалогах, беседах за круглым столом и т. д.

Следует еще раз подчеркнуть, что и активные и пассивные методы могут чередоваться даже в рамках одного сеанса извлечения знаний. Например, если инженер по знаниям застенчив и не имеет большого опыта, то вначале он может использовать пассивные методы, а постепенно, ближе знакомясь с экспертом, захватывать инициативу и переходить «в наступление».

Пассивные методы на первый взгляд достаточно просты, но на самом деле требуют от инженера по знаниям умения четко анализировать поток сознания эксперта и выявлять в нем значимые фрагменты знаний. Отсутствие обратной связи (пассивность инженера по знаниям) значительно ослабляет эффективность этих методов, чем и объясняется их обычно вспомогательная роль при активных методах.

Активные методы можно разделить на две группы в зависимости от числа экспертов, отдающих свои знания. Если их число больше одного, то целесообразно помимо серии индивидуальных контактов с каждым применять и методы групповых обсуждений предметной области. Такие *групповые* методы обычно активизируют мышление участников дискуссий и

позволяют выявлять весьма нетривиальные аспекты их знаний. В свою очередь, *индивидуальные* методы на сегодняшний день остаются ведущими, поскольку столь деликатная процедура, как «отъем знаний», не терпит лишних свидетелей.

Отдельно следует сказать об играх. *Игровые* методы сейчас широко используются в социологии, экономике, менеджменте, педагогике для подготовки руководителей, учителей, врачей и других специалистов. Игра — это особая форма деятельности и творчества, где человек раскрепощается и чувствует себя намного свободнее, чем в обычной трудовой деятельности.

На выбор метода влияют три фактора: личностные особенности инженера по знаниям, личностные особенности эксперта и характеристика предметной области.

Одна из возможных классификаций людей по психологическим характеристикам делит всех на три типа:

• мыслитель (познавательный тип);

• собеседник (эмоционально-коммуникативный тип);

• практик (практический тип).

Мыслители ориентированы на интеллектуальную работу, учебу, теоретические обобщения и обладают такими характеристиками когнитивного стиля, как поле-независимость и рефлексивность.

Собеседники — это общительные, открытые люди, готовые к сотрудничеству.

Практики предпочитают действие разговорам, хорошо реализуют замыслы других, направлены на результативность работы.

Для характеристики предметных областей можно предложить следующую классификацию:

• хорошо документированные;

• средне документированные;

• слабо документированные.

Эта классификация связана с соотношением двух видов знаний Z1 и Z2, где Z1 — это экспертное «личное» знание, а *Z2* — материализованное в книгах «общее» знание в данной конкретной области. Если представить знания Zno предметной области как объединение Z1 и Z2, то есть Zno= *Z1eZ2* то рис. 5.2 наглядно иллюстрирует предложенную классификацию.

Рис. 5.2. Классификация предметных областей

Кроме этого, предметные области можно разделить по критерию структурированности знаний. Под *структурированностью* будем понимать степень теоретического осмысления и выявленности основных закономерностей и принципов, действующих в данной предметной области. И хотя ЭС традиционно применяются в слабо структурированных предметных областях, сейчас наблюдается тенденция расширения сферы внедрения экспертных систем. По степени структурированности знаний предметные области могут быть:

• *хорошо структурированными* — с четкой аксиоматизацией, широким применением математического аппарата, устоявшейся терминологией;

• *средне структурированными* — с определившейся терминологией, развивающейся теорией, явными взаимосвязями между явлениями;

• *слабо структурированными* — с размытыми определениями, богатой эмпирикой, скрытыми взаимосвязями, с большим количеством «белых пятен».

В соответствии с классификацией, представленной на рис. 5.1, рассмотрим подробнее обе разновидности коммуникативных методов: пассивные и активные.

Термин «пассивные» не должен вызывать иллюзий, поскольку он введен как противовес к «активным» методам. В реальности же пассивные методы требуют от инженера по знаниям не меньшей отдачи, чем такие активные методы, как игры и диалог.

Пассивные методы извлечения знаний включают такие методы, где ведущая роль в процедуре извлечения фактически передается эксперту, а инженер по знаниям только фиксирует рассуждения эксперта во время работы по принятию решений.

Согласно классификации (см. рис. 5.1) к этой группе относятся:

- наблюдения;
- анализ протоколов «мыслей вслух»;
- лекции.

Наблюдения

В процессе наблюдений инженер по знаниям находится непосредственно рядом с экспертом во время его профессиональной деятельности или имитации этой деятельности. При подготовке к сеансу извлечения эксперту необходимо объяснить цель наблюдений и попросить максимально комментировать свои действия.

Во время сеанса аналитик записывает все действия эксперта, его реплики и объяснения. Может быть сделана и видеозапись в реальном масштабе времени. Непременное условие этого метода — невмешательство аналитика в работу эксперта хотя бы на первых порах. Именно метод наблюдений является единственно «чистым» методом, исключающим вмешательство инженера по знаниям и навязывание им каких-то своих структур представлений.

Существуют две основные разновидности проведения наблюдений:

• наблюдение за реальным процессом;

• наблюдение за имитацией процесса.

Обычно используются обе разновидности. Сначала инженеру по знаниям полезно наблюдать за реальным процессом, чтобы глубже понять предметную область и отметить все внешние особенности процесса принятия решения. Это необходимо для проектирования эффективного интерфейса пользователя. Ведь будущая ЭС должна работать именно в контексте такого реального производственного процесса. Кроме того, только наблюдение позволит аналитику увидеть предметную область, а, как известно, «лучше один раз увидеть, чем сто раз услышать».

Наблюдение за имитацией процесса проводят обычно также за рабочим местом эксперта, но сам процесс деятельности запускается специально для аналитика. Преимущество этой разновидности в том, что эксперт менее напряжен, чем в первом варианте, когда он работает на «два фронта» — и ведет профессиональную деятельность, и демонстрирует ее.

Наблюдения за имитацией проводят также и в тех случаях, когда наблюдения за реальным процессом по каким-либо причинам невозможны (например, профессиональная этика врача-психиатра может не допускать присутствия постороннего на приеме).

Сеансы наблюдений могут потребовать от инженера по знаниям:

• овладения техникой стенографии для фиксации действий эксперта в реальном масштабе времени;

• ознакомления с методиками хронометража для четкого структурирования производственного процесса по времени;

• развития навыков «чтения по глазам», то есть наблюдательности к жестам, мимике и другим невербальным компонентам общения;

• серьезного предварительного знакомства с предметной областью, так как из-за отсутствия «обратной связи» иногда многое непонятно в действиях экспертов.

Протоколы наблюдений после сеансов в ходе домашней работы тщательно расшифровываются, а затем обсуждаются с экспертом.

Таким образом, наблюдения — один из наиболее распространенных методов извлечения знаний на начальных этапах разработки. Обычно он применяется не самостоятельно, а в совокупности с другими методами.

Анализ протоколов «мыслей вслух»

Протоколирование «мыслей вслух» отличается от наблюдений тем, что эксперта просят не просто прокомментировать свои действия и решения, но и объяснить, как это решение было найдено, то есть продемонстрировать всю цепочку своих рассуждений. Во время рассуждений эксперта все его слова, весь «поток сознания» протоколируется инженером по знаниям, при этом полезно отметить даже паузы и междометия. Иногда этот метод называют «вербальные отчеты».

Вопрос об использовании для этой цели магнитофонов и диктофонов является дискуссионным, поскольку магнитофон иногда парализующе действует на эксперта, разрушая атмосферу доверительности, которая может и должна возникать при непосредственном общении.

Основной трудностью при протоколировании «мыслей вслух» является принципиальная сложность для любого человека объяснить, как он думает. При этом существуют экспериментальные психологические доказательства того факта, что люди не всегда в состоянии достоверно описывать мыслительные процессы. Кроме того, часть знаний, хранящихся в невербальной форме (например, различные процедурные знания типа «как завязывать шнурки»), вообще слабо коррелируют с их словесным описанием. Автор теории фреймов М. Минский считает, что «только как исключение, а не как правило, человек может объяснить то, что он знает». Однако существуют люди, склонные к рефлексии, для которых эта работа является

вполне доступной. Следовательно, такая характеристика когнитивного стиля, как рефлексивность, является для эксперта более чем желательной.

Расшифровка полученных протоколов производится инженером по знаниям самостоятельно с коррекциями на следующих сеансах извлечения знаний. Удачно проведенное протоколирование «мыслей вслух» является одним из наиболее эффективных методов извлечения, поскольку в нем эксперт может проявить себя максимально ярко, он ничем не скован, никто ему не мешает, он как бы свободно парит в потоке собственных умозаключений и рассуждений. Он может здесь блеснуть эрудицией, продемонстрировать глубину своих познаний. Для большого числа экспертов это самый приятный и лестный способ извлечения знаний.

От инженера по знаниям метод «мысли вслух» требует тех же умений, что и метод наблюдений. Обычно «мысли вслух» дополняются потом одним из активных методов для реализации обратной связи между интерпретацией инженера по знаниям и представлениями эксперта.

Лекции

Лекция является самым старым способом передачи знаний. Лекторское искусство издревле очень высоко ценилось во всех областях науки и культуры. Но нас сейчас интересует не столько способность к подготовке и чтению лекций, сколько способность эту лекцию слушать, конспектировать и усваивать. Уже говорилось, что чаще всего экспертов не выбирают, и поэтому учить эксперта чтению лекции инженер по знаниям не сможет. Но если эксперт имеет опыт преподавателя (например, профессор клиники или опытный руководитель производства), то можно воспользоваться таким концентрированным фрагментом знаний, как лекция. В лекции эксперту также предоставлено много степеней свободы для самовыражения; при этом необходимо сформулировать эксперту тему и задачу лекции. Например, тема цикла лекций «Постановка диагноза — воспаление легких», тема конкретной лекции «Рассуждения по анализу рентгенограмм», задача — научить

слушателей по перечисленным экспертом признакам ставить диагноз воспаления легких и делать прогноз. При такой постановке опытный лектор может заранее структурировать свои знания и ход рассуждений. От инженера по знаниям в этой ситуации требуется лишь грамотно законспектировать лекцию и в конце задать необходимые вопросы.

Студенты хорошо знают, что конспекты лекций одного и того же лектора у разных студентов существенно отличаются. Списать конспект лекций просят, как правило, у одного-двух студентов из группы. Люди, умело ведущие конспект, обычно сильные студенты. Обратное не верно. В чем же заключается искусство ведения конспекта? В «помехоустойчивости». Записывать главное, опускать второстепенное, выделять фрагменты знаний (параграфы, под-параграфы), записывать только осмысленные предложения, уметь обобщать.

Хороший вопрос по ходу лекции помогает и лектору и слушателю. Серьезные и глубокие вопросы могут существенно поднять авторитет инженера по знаниям в глазах эксперта.

Опытный лектор знает, что все вопросы можно условно разбить на три группы:

• умные вопросы, углубляющие лекцию;

• глупые вопросы или вопросы не по существу;

• вопросы «на засыпку» или провокационные.

Если инженер по знаниям задает вопросы второго типа, то возможны две реакции. Вежливый эксперт будет разговаривать с таким аналитиком как с ребенком, который сейчас не понимает и все равно ничего уже не поймет. Заносчивый эксперт просто выйдет из контакта, не желая терять время. Если же инженер по знаниям захочет продемонстрировать свою эрудицию вопросами третьего типа, то ничего, кроме раздражения и отчуждения, он, по-видимому, в ответ не получит.

Продолжительность лекции рекомендуется стандартная — от 40 до 50 минут и через 5-10 минут — еще столько же. Курс обычно от двух до пяти лекций.

Метод извлечения знаний в форме лекций, как и все пассивные методы, используют в начале разработки как эффективный способ быстрого погружения инженера по знаниям в предметную область.

В заключение несколько советов, как слушать лекции:

- К лекции подготовьтесь, то есть познакомьтесь с предметной областью.
  .
- Слушайте с максимальным вниманием, для этого: устраните мешающие факторы (скрип двери, шорохи и т. д.); удобно устройтесь; поменьше двигайтесь.
- Учитесь отдыхать во время слушания (например, когда лектор приводит цифры, которые можно взять из справочника).
- Слушайте одновременно и лектора, и самого себя (параллельно с мыслями лектора по ассоциации возникают собственные мысли).
- Слушайте и одновременно записывайте, но записывайте текст сокращенно, используя условные значки (для этого вовсе не следует непременно быть стенографом, достаточно только установить для себя ряд условных значков и ими неизменно пользоваться).
- Расшифруйте записи лекции в тот же день.
- Не спорьте с лектором в процессе лекции.
- Рационально используйте перерывы в лекции для подведения итогов прослушанного.

Сравнительные характеристики пассивных методов извлечения знаний представлены в табл. 5.1.

Таблица 5.1. Сравнительные характеристики пассивных методов извлечения знаний

Пассивный метод	Наблюдения	"Мысли вслух"	Лекции
Достоинства	Отсутствие влияния аналитика и его субъективной позиции. Максимальное приближение аналитика к предметной области.	Свобода самовыражения для эксперта. Обнаженность структур рассуждений. Отсутствие влияния аналитика и его субъективной позиции.	Свобода самовыражения для эксперта. Структурированное изложение. Высокая концентрация. Отсутствие влияния аналитика и его субъективной позиции.
Недостатки	Отсутствие обратной связи. Фрагментарность полученных комментариев	Отсутствие обратной связи. Возможность ухода "в сторону" в рассуждениях эксперта.	"Зашумленность" деталями. Слабая обратная связь. Недостаток хороших лекторов среди экспертов-практиков
Требования к эксперту (типы и основные качества)	Собеседник или мыслитель (способность к вербализации + мыслей + аналитичность + открытость + рефлексивность)	Собеседник или мыслитель (способность к вербализации + мыслей + аналитичность + открытость + рефлексивность)	Мыслитель (лекторские способности)
Требования к аналитику (типы и основные качества)	Мыслитель (наблюдательность + поленезависимость)	Мыслитель или собеседник (контактность + поленезависимость)	Мыслитель (поленезависимость + способность к обобщению)
Характеристика предметной области	Слабо и средне структурированные; слабо и средне документированные	Тоже	Слабо документированные и слабо структурированные

Активные индивидуальные методы

Активные индивидуальные методы извлечения знаний на сегодняшний день наиболее распространенные. В той или иной степени к ним прибегают

при разработке практически любой ЭС. К основным активным методам можно отнести:

• анкетирование;

• интервью;

• свободный диалог;

• игры с экспертом.

Во всех этих методах активную функцию выполняет инженер по знаниям, который пишет сценарий и режиссирует сеансы извлечения знаний. Игры с экспертом существенно отличаются от трех других методов. Три оставшихся метода очень схожи между собой и отличаются лишь по степени свободы, которую может себе позволить инженер по знаниям при проведении сеансов извлечения знаний. Их можно назвать вопросными методами поиска знаний.

Анкетирование

Анкетирование — наиболее жесткий метод, то есть наиболее стандартизированный. В этом случае инженер по знаниям заранее составляет вопросник или анкету, размножает ее и использует для опроса нескольких экспертов. Это основное преимущество анкетирования.

Сама процедура может проводиться двумя способами:

1. Аналитик вслух задает вопросы и сам заполняет анкету по ответам эксперта.

2. Эксперт самостоятельно заполняет анкету после предварительного инструктирования.

Выбор способа зависит от конкретных условий (например, от оформления анкеты, ее понятности, готовности эксперта). Второй способ нам кажется предпочтительнее, так как у эксперта появляется неограниченное время на обдумывание ответов.

Основными факторами, на которые можно существенно повлиять при анкетировании, являются средства общения (в данном случае это вопросник) и ситуация общения.

Вопросник (анкета) заслуживает особого разговора. Существует несколько общих рекомендаций при составлении анкет. Эти рекомендации являются универсальными, то есть не зависят от предметной области:

• Анкета не должна быть монотонной и однообразной, то есть вызывать скуку или усталость. Это достигается вариациями формы вопросов, сменой тематики, вставкой вопросов-шуток и игровых вопросов.

• Анкета должна быть приспособлена к языку экспертов.

• Следует учитывать, что вопросы влияют друг на друга и поэтому последовательность вопросов должна быть строго продумана.

• Желательно стремиться к оптимальной избыточности. Известно, что в анкете всегда много лишних вопросов.

• Анкета должна иметь «хорошие манеры», то есть ее язык должен быть ясным, понятным, предельно вежливым. Методическим мастерством составления анкеты можно овладеть только на практике.

Интервью

Под *интервью* будем понимать специфическую форму общения инженера по знаниям и эксперта, в которой инженер по знаниям задает эксперту серию заранее подготовленных вопросов с целью извлечения знаний о предметной области. Наибольший опыт в проведении интервью накоплен, наверное, в журналистике и социологии. Большинство специалистов этих областей отмечают тем не менее крайнюю недостаточность теоретических и методических исследований по тематике интервьюирования.

Интервью очень близко тому способу анкетирования, когда аналитик сам заполняет анкету, занося туда ответы эксперта. Основное отличие интервью в том, что оно позволяет аналитику опускать ряд вопросов в

зависимости от ситуации, вставлять новые вопросы в анкету, изменять темп, разнообразить ситуацию общения. Кроме этого, у аналитика появляется возможность «взять в плен» эксперта своим обаянием, заинтересовать его самой процедурой и тем самым увеличить эффективность сеанса извлечения.

Вопросы для интервью

Теперь несколько подробнее о центральном звене активных индивидуальных методов — о вопросах. Инженеры по знаниям редко сомневаются в своей способности задавать вопросы. В то время как и в философии и в математике эта проблема обсуждается с давних лет. Существует даже специальная ветвь математической логики — *эротетическая* логика (логика вопросов).

Все вопросительные предложения можно разбить на два типа :

• вопросы *с неопределенностью,* относящейся ко всему предложению («Действительно, введение больших доз антибиотиков может вызвать анафилактический шок?);

• вопросы *с неполной информацией* («При каких условиях необходимо включать кнопку?»), часто начинающиеся со слов «кто», что», «где», «когда» и т. д.

Это разделение можно дополнить классификацией, представленной на рис. 5.3.

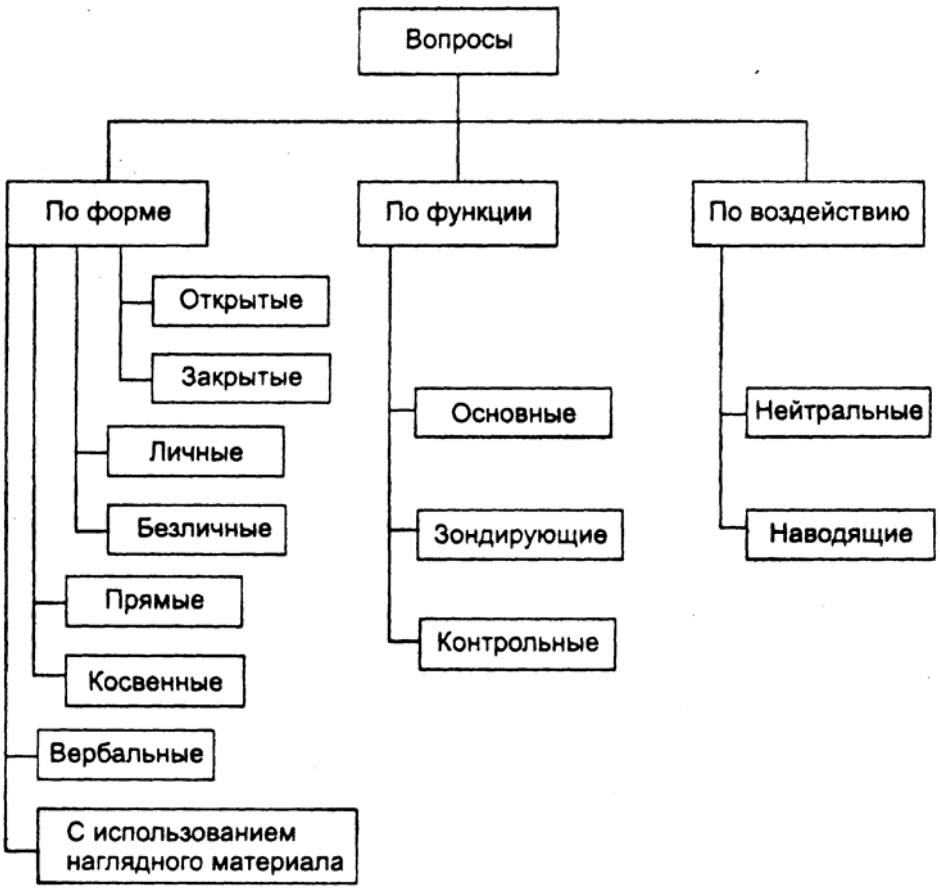

Рис. 5.3. Классификация вопросов

Открытый вопрос называет тему или предмет, оставляя полную свободу эксперту по форме и содержанию ответа («Не могли бы вы рассказать, как лучше сбить высокую температуру у больного с воспалением легких?»).

В *закрытом* вопросе эксперт выбирает ответ из набора предложенных («Укажите, пожалуйста, что вы рекомендуете при ангине: а) антибиотики, б) полоскание, в) компрессы, г) ингаляции»). Закрытые вопросы легче обрабатывать при последующем анализе, но они более опасны, так как «закрывают» ход рассуждений эксперта и «программируют» его ответ в определенном направлении. При составлении сценария интервью полезно чередовать открытые и закрытые вопросы, особенно тщательно продумывать закрытые, поскольку для их составления требуется определенная эрудиция в предметной области.

Личный вопрос касается непосредственно личного индивидуального опыта эксперта («Скажите, пожалуйста, Иван Данилович, в вашей практике

вы применяете вулнузан при фурункулезах?»). Личные вопросы обычно активизируют мышление эксперта, «играют» на его самолюбии, они всегда украшают интервью.

Безличный вопрос направлен на выявление наиболее распространенных и общепринятых закономерностей предметной области («Что влияет на скорость процесса ферментации лизина?»).

При составлении вопросов следует учитывать, что языковые способности эксперта, как правило, ограничены и вследствие скованности, замкнутости, робости он не может сразу высказать свое мнение и предоставить знания, которые от него требуются (даже если предположить, что он их четко для себя формулирует). Поэтому часто при «зажатости» эксперта используют не *прямые* вопросы, которые непосредственно указывают на предмет или тему («Как вы относитесь к методике доктора Сухарева?»), а *косвенные,* которые лишь косвенно указывают на интересующий предмет («Применяете ли вы методику доктора Сухарева? Опишите, пожалуйста, результаты лечения»). Иногда приходится задавать несколько десятков косвенных вопросов вместо одного прямого.

Вербальные вопросы — это традиционные устные вопросы. *Вопросы с использованием наглядного материала* разнообразят интервью и снижают утомляемость эксперта. В этих вопросах используют фотографии, рисунки и карточки. Например, эксперту предлагаются цветные картонные карточки, на которых выписаны признаки заболевания. Затем аналитик просит разложить эти карточки в порядке убывания значимости для постановки диагноза.

Деление вопросов по функции на основные, зондирующие, контрольные связано с тем, что часто *основные* вопросы интервью, направленные на выявление знаний, не срабатывают — эксперт по каким-то причинам уходит в сторону от вопроса, отвечает нечетко.

Тогда аналитик использует *зондирующие* вопросы, которые направляют рассуждения эксперта в нужную сторону. Например, если не сработал основной вопрос:

«Какие параметры определяют момент окончания процесса ферментации лизина?» — аналитик начинает задавать зондирующие вопросы: «Всегда ли процесс ферментации длится 72 часа? А если он заканчивается раньше, как это узнать? Если он продлится больше, то что заставит микробиолога не закончить процесс на 72-м часу?» и т. д.

Контрольные вопросы применяют для проверки достоверности и объективности информации, полученной в интервью ранее («Скажите, пожалуйста, а московская школа психологов так же как вы трактует шкалу К опросника MMPI?» или «Рекомендуете ли вы инъекции АТФ?» (АТФ — препарат, снятый с производства). Контрольные вопросы должны быть «хитро» составлены, чтобы не обидеть эксперта недоверием (для этого используют повторение вопросов в другой форме, уточнения, ссылки на другие источники). «Лучше два раза спросить, чем один раз напутать» (Шолом—Алейхем).

И, наконец, о нейтральных и наводящих вопросах. В принципе интервьюеру (в нашем случае инженеру по знаниям) рекомендуют быть беспристрастным, отсюда и вопросы его должны носить *нейтральный* характер, то есть не должны указывать на отношение интервьюера к данной теме. Напротив, *наводящие* вопросы заставляют респондента (в данном случае эксперта) прислушаться или даже принять во внимание позицию интервьюера. Нейтральный вопрос: «Совпадают ли симптомы кровоизлияния в мозг и сотрясения мозга?» Наводящий вопрос:

«Не правда ли, очень трудно дифференцировать симптомы кровоизлияния в мозг?»

Кроме перечисленных выше, полезно различать и включать в интервью следующие вопросы:

- контактные («ломающие лед» между аналитиком и экспертом);
- буферные (для разграничения отдельных тем интервью);
- оживляющие память экспертов (для реконструкции отдельных случаев из практики);

• «провоцирующие» (для получения спонтанных, неподготовленных ответов).

Вопрос в интервью — это не просто средство общения, но и способ передачи мыслей и позиции аналитика.

Очевидно, что любой вопрос имеет смысл только в контексте. Поэтому вопросы может готовить инженер по знаниям, уже овладевший ключевым набором знаний.

Свободный диалог

Свободный диалог — это метод извлечения знаний в форме беседы инженера по знаниям и эксперта, в которой нет жесткого регламентированного плана и вопросника. Это определение не означает, что к свободному диалогу не надо готовиться. Напротив, внешне свободная и легкая форма этого метода требует высочайшей профессиональной и психологической подготовки. Рисунок 5.4 графически иллюстрирует схему такой подготовки, дополненную в связи со спецификой инженерии знаний. Подготовка занимает разное время в зависимости от степени профессионализма аналитика, но в любом случае она необходима, так как несколько уменьшает вероятность самого нерационального метода — метода проб и ошибок.

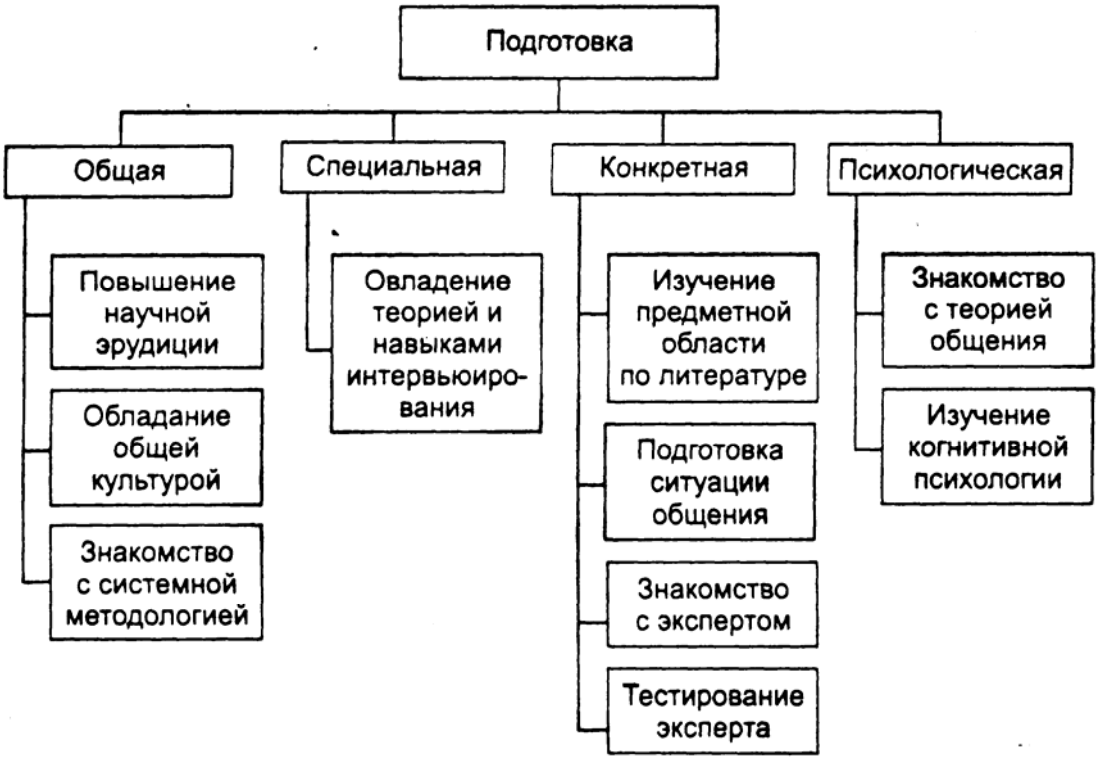

Рис. 5.4. Подготовка к извлечению знаний

Подготовка к диалогу так же, как и к другим активным методам извлечения знаний, включает составление плана проведения сеанса извлечения, в котором необходимо предусмотреть следующие стадии:

1. Начало беседы (знакомство, создание у эксперта «образа» аналитика, объяснение целей и задач работы).

2. Диалог по извлечению знаний.

3. Заключительная стадия (благодарность эксперту, подведение итогов, договор о последующих встречах).

Девизом для инженера по знаниям могут послужить взгляды одного из классиков отечественного литературоведения М. М. Бахтина:

«Диалог — столкновение разных умов, разных истин, несходных культурных позиций, составляющих единый ум, единую истину, общую культуру». «Диалог предполагает:

• уникальность каждого партнера и их принципиальное равенство друг другу;

• различие и оригинальность их точек зрения;

• ориентацию каждого на понимание и на активную интерпретацию его точки зрения партнером;

• ожидание ответа и его предвосхищение в собственном высказывании;

• взаимную дополнительность позиций участников общения, соотнесение которых и является целью диалога».

Сравнительные характеристики активных индивидуальных методов извлечения знаний представлены в табл. 5.2.

Таблица 5.2. Сравнительные характеристики активных индивидуальных методов извлечения.

Активный индивидуальный метод извлечения знаний	**Анкетирование**	**Интервью**	**Свободный диалог**
Достоинства	Возможность стандартизированного опроса нескольких экспертов	Наличие обратной связи (возможность уточнений и разрешения противоречий) для эксперта	Гибкость. Сильная обратная связь изменения.
Активный индивидуальный метод извлечения знаний	Анкетирование не требует особенного напряжения от аналитика во время процедуры анкетирования	Интервью	Свободный диалог. Возможность изменения сценария и формы сеанса
Недостатки	Требует умения и опыта составления анкет. Отсутствие контекста между	Требует значительного времени на подготовку	Требует от аналитика высочайшего напряжения.

	экспертом, нет обратной связи. Вопросы анкеты могут быть неправильно поняты экспертом	вопросов интервью	Отсутствие формальных методик проведения. Трудность протоколирования результатов.
Требования к эксперту (типы и качества)	Практик и мыслитель	Собеседник или мыслитель	Собеседник или мыслитель
Требования к аналитику (типы и качества)	Мыслитель (педантизм в составлении анкет, внимательность)	Собеседник (журналистские навыки, умение слушать)	Собеседник (наблюдательность умение слушать, обаяние)
Характеристика предметной области	Слабо структурированные, слабо и средне документированные	Тоже	Тоже

Активные групповые методы

Основное достоинство групповых методов — это возможность одновременного «поглощения» знаний от нескольких экспертов, взаимодействие которых вносит в этот процесс элемент принципиальной новизны от наложения разных взглядов и позиций.

Поскольку эти методы менее популярны, чем индивидуальные (что связано со сложностью организации), попытаемся описать их подробно.

«Круглый стол»

Метод *круглого стола* (термин заимствован из журналистики) предусматривает обсуждение какой-либо проблемы из выбранной

предметной области, в котором принимают участие с равными правами несколько экспертов. Обычно вначале участники высказываются в определенном порядке, а затем переходят к живой свободной дискуссии. Число участников дискуссии колеблется от трех до пяти—семи.

Большинство общих рекомендаций по извлечению знаний, предложенных ранее, применимо и к данному методу. Однако существует и специфика, связанная с поведением человека в группе.

Во-первых, от инженера по знаниям подготовка «круглого стола» потребует дополнительных усилий: как организационных (место, время, обстановка, минеральная вода, чай, кворум и т. д.), так и психологических (умение вставлять уместные реплики, чувство юмора, память на имена и отчества, способность гасить конфликтные ситуации и т. д.).

Во-вторых, большинство участников будет говорить под воздействием «эффекта фасада» совсем не то, что они сказали бы в другой обстановке, то есть желание произвести впечатление на других экспертов будет существенно «подсвечивать» их высказывания. Этот эффект часто наблюдается на защитах диссертаций. Члены ученого совета спрашивают обычно не то, что им действительно интересно, а то, что демонстрирует их собственную компетентность. Ход беседы за круглым столом удобно записывать на магнитофон, а при расшифровке и анализе результатов учитывать этот эффект, а также взаимные отношения участников.

Задача дискуссии — коллективно, с разных точек зрения, под разными углами исследовать спорные гипотезы предметной области. Обычно эмпирические области богаты таким дискуссионным материалом. Для остроты на «круглый стол» приглашают представителей разных научных направлений и разных поколений, это также уменьшает опасность получения односторонних однобоких знаний. Обмен мнениями по научным вопросам имеет давнюю традицию в истории человечества (античная Греция, Индия). До наших дней дошли литературные памятники обсуждения спорных вопросов (например, Протагор «Искусство спорить», работы софистов),

послужившие первоосновой диалектики — науки вести беседу, спорить, развивать теорию. В самом слове дискуссия (от лат. *discussio* — исследование) содержится указание на то, что это метод научного познания, а не просто споры (для сравнения, полемика — от греч. *polemikos* — воинственный, враждебный).

По ходу дискуссии важно проследить, чтобы слишком эмоциональные и разговорчивые эксперты не подменили тему и чтобы критика позиций друг друга была обоснованной.

«Мозговой штурм»

Активные групповые методы обычно используются в качестве острой приправы при извлечении знаний, сами по себе они не могут служить источником более или менее полного знания. Их применяют как дополнительные к традиционным индивидуальным методам (наблюдения, интервью и т. д.), для активизации мышления и поведения экспертов.

«Мозговой штурм» или «мозговая атака» — один из наиболее распространенных методов раскрепощения и активизации творческого мышления.

Впервые этот метод был использован в 1939 г. в США А. Осборном как способ получения новых идей в условиях запрещения критики. Замечено, что боязнь критики мешает творческому мышлению, поэтому основная идея штурма — это отделение процедуры генерирования идей в замкнутой группе специалистов от процесса анализа и оценки высказанных идей.

Как правило, штурм длится недолго (около 40 минут). Участникам (до 10 человек) предлагается высказывать любые идеи (шутливые, фантастические, ошибочные) на заданную тему (критика запрещена). Обычно высказывается более 50 идей. Регламент до 2 минут на выступление. Самый интересный момент штурма — это наступление пика (ажиотажа), когда идеи начинают «фонтанировать», то есть происходит непроизвольная генерация гипотез участниками. Этот пик имеет теоретическое обоснование в

работах выдающегося швейцарского психолога и психиатра З. Фрейда о бессознательном. При последующем анализе всего лишь 10-15 % идей оказываются разумными, но среди них бывают весьма оригинальные. Оценивает результаты обычно группа экспертов, не участвовавшая в генерации.

Ведущий «мозгового штурма» — инженер по знаниям — должен свободно владеть аудиторией, подобрать активную группу экспертов — «генераторов», не зажимать плохие идеи — они могут служить катализаторами хороших. Искусство ведущего — это искусство задавать вопросы аудитории, «подогревая» генерацию. Вопросы также могут останавливать многословных экспертов и служить способом развития идей других. Основной девиз штурма — «чем больше идей, тем лучше». Фиксация хода сеанса — традиционная (протокол или магнитофон).

Достоинства и недостатки активных групповых методов извлечения знаний представлены в табл. 5.3.

Таблица 5.3. Сравнение активных групповых методов извлечения знаний

Активный групповой метод извлечения знаний	**"Круглый стол"**	**"Мозговой штурм"**
Достоинства	Позволяет получить более объективные фрагменты знаний. Оживляет процедуру извлечения. Позволяет участникам обмениваться знаниями.	Позволяет выявлять глубинные пласты знаний (на уровне бессознательного). Активизирует экспертов. Позволяет получать новое знание (гипотезы).
Недостатки	Требует больших организационных затрат. Отличается сложностью проведения.	Возможен только для новых интересных исследовательских проблем. Не всегда эффективны (довольно низкий процент продуктивных идей)

Требования к эксперту (тип и качества)	Собеседник или мыслитель (искусство полемики)	Мыслитель (креативность, то есть способность к творчеству)
Требования к инженеру по знаниям (тип и качества)	Собеседник (дипломатические способности)	Собеседник или мыслитель (быстрая реакция и чувство юмора)
Характеристика предметной области	Слабо структурированные и слабо документированные с наличием спорных проблем	Слабо структурированные и слабо документированные с наличием перспективных "белых пятен"

Экспертные игры

В настоящее время в психолого-педагогических науках нет развитой теоретической концепции деловых игр и других игровых методов обучения. Тем не менее на практике эти игры широко используются. Под *деловой* игрой чаще всего понимают эксперимент, где участникам предлагается производственная ситуация, а они на основе своего жизненного опыта, своих общих и специальных знаний и представлений принимают решения. Решения анализируются, и вскрываются закономерности мышления участников эксперимента. Именно эта анализирующая часть деловой игры полезна для получения знаний. И если участниками такой игры становятся эксперты, то игра из деловой превращается в экспертную. Из трех основных типов деловых игр (учебных, планово-производственных и исследовательских) к экспертам ближе всего исследовательские, которые используются для анализа систем, проверки правил принятия решений.

Диагностическая игра — это та же деловая игра, но применяемая конкретно для диагностики методов принятия решения в медицине (диагностика методов диагностики). Эти игры возникли при исследовании способов передачи опыта от опытных врачей новичкам. В нашем понимании диагностическая игра — это игра, безусловно, экспертная без всяких

оговорок, только с жестко закрепленной предметной областью — медициной.

Плодотворность моделирования реальных ситуаций в играх подтверждается сегодня практически во всех областях науки и техники. Они развивают логическое мышление, умение быстро принимать решения, вызывают интерес у экспертов.

Экспертные игры бывают:

• индивидуальные;

• групповые.

Экспертные игры можно также классифицировать по использованию оборудования:

• использование специального оборудования;

• применение вычислительной техники.

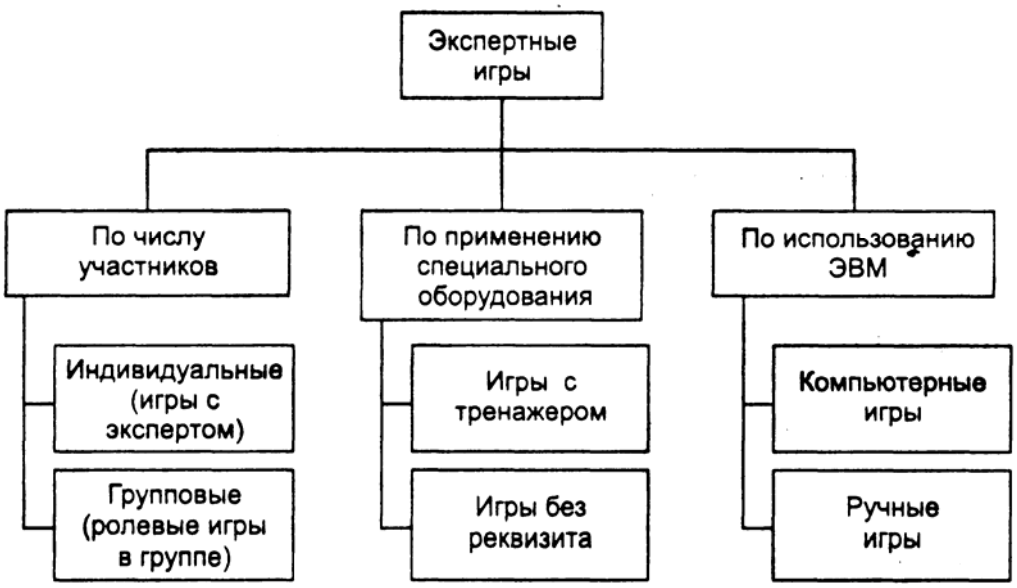

Рис. 5.5. Классификация экспертных игр

Группа *текстологических методов* объединяет методы извлечения знаний, основанные на изучении специальных текстов из учебников, монографий, статей, методик и других носителей профессиональных знаний.

В буквальном смысле текстологические методы не относятся к текстологии — науке, которая родилась в русле филологии с целью критического прочтения литературных текстов, изучения и интерпретации источников с узко прикладной задачей — подготовки текстов к изданию.

Текстологические методы извлечения знаний, безусловно, используя основные положения текстологии, отличаются принципиально от ее методологии, во-первых, характером и природой своих источников (профессиональная специальная литература, а не художественная, живущая по своим особым законам), а во-вторых, жесткой прагматической направленностью извлечения конкретных профессиональных знаний.

Среди методов извлечения знаний эта группа является наименее разработанной, по ней практически нет никакой библиографии, поэтому дальнейшее изложение является как бы введением в методы изучения текстов в том виде, как это представляют авторы.

Задачу извлечения знаний из текстов можно сформулировать как задачу *понимания* и выделения смысла текста.

При этом можно выделить две такие смысловые структуры:

М1 — смысл, который пытался заложить автор, это его модель мира, и *М2* — смысл, который постигает читатель, в данном случае инженер по знаниям (рис. 4.6), в процессе интерпретации I. При этом Г — это словесное одеяние М1, то есть результат вербализации V.

Сложность процесса заключается в принципиальной невозможности совпадения знаний, образующих *М1*, и *М2,* из-за того, что *М1* образуется за счет всей совокупности представлений, потребностей, интересов и опыта автора, лишь малая часть которых находит отражение в тексте *Т.* Соответственно, и *М2* образуется в процессе интерпретации текста *Т* за счет привлечения всей совокупности научного и человеческого багажа читателя.

Таким образом, два инженера по знаниям извлекут из одного Т две различные модели *M*i1 и *M*i2.

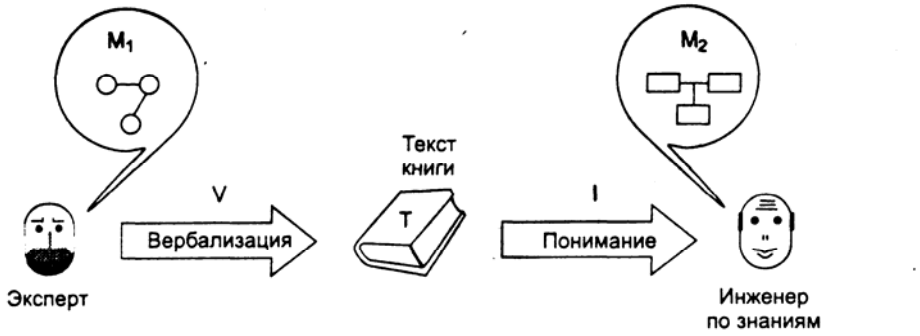

Рис. 5.6. Схема извлечения знаний из специальных текстов

Рис. 4.7. Компоненты научного текста

При извлечении знаний аналитику, интерпретирующему текст, приходится решать задачу декомпозиции этого текста на перечисленные выше компоненты для выделения истинно значимых для реализации базы знаний фрагментов. Сложность интерпретации научных и специальных текстов заключается еще и в том, что любой текст приобретает смысл только в контексте, где под *контекстом* понимается окружение, в которое «погружен» текст.

Основными моментами понимания текста являются:

• Выдвижение предварительной гипотезы о смысле всего текста (предугадывание).

• Определение значений непонятных слов (то есть специальной терминологии).

• Возникновение общей гипотезы о содержании текста (о знаниях).

• Уточнение значения терминов и интерпретация отдельных фрагментов текста под влиянием общей гипотезы (от целого к частям).

• Формирование некоторой смысловой структуры текста за счет установления внутренних связей между отдельными важными (ключевыми) словами и фрагментами, а также за счет образования абстрактных понятий, обобщающих конкретные фрагменты знаний.

• Корректировка общей гипотезы относительно содержащихся в тексте фрагментов знаний (от частей к целому).

• Принятие основной гипотезы, то есть формирование *M*2

Приведем алгоритм извлечения знаний из текста:

1. Составление «базового» списка литературы для ознакомления с предметной областью и чтение по списку.
2. Выбор текста для извлечения знаний.
3. Первое знакомство с текстом (беглое прочтение). Для определения значения незнакомых слов — консультации со специалистами или привлечение справочной литературы.
4. Формирование первой гипотезы о макроструктуре текста.
5. Внимательное прочтение текста с выписыванием ключевых слов и выражений, то есть выделение «смысловых вех» (компрессия текста).
6. Определение связей между ключевыми словами, разработка макроструктуры текста в форме графа или «сжатого» текста (реферата).
7. Формирование поля знаний на основании макроструктуры текста.

ТЕМА 6. СТРУКТУРИРОВАНИЕ ЗНАНИЙ

Поле знаний — это условное неформальное описание основных понятий и взаимосвязей между понятиями предметной области, выявленных из системы знаний эксперта, в виде графа, диаграммы, таблицы или текста.

В качестве простейшего прагматического подхода *к* формированию поля знаний начинающему инженеру по знаниям можно предложить следующий алгоритм для «чайников» (рис. 6.1).

1. Определение входных {X} и выходных {Y} данных. Этот шаг совершенно необходим, так как он определяет направление движения в поле знаний — от X к Y. Кроме того, структура входных и выходных данных существенно влияет на форму и содержание поля знаний. На этом шаге определение может быть достаточно размытым, в дальнейшем оно будет уточняться.

2. Составление словаря терминов и наборов ключевых слов N. На этом шаге проводится текстуальный анализ всех протоколов сеансов извлечения знании и выписываются все значимые слова, обозначающие понятия, явления, процессы, предметы, действия, признаки и т. н. При этом следует попытаться разобраться в значении терминов. Важен осмысленный словарь.

3. Выявление объектов и понятий {A}. Производится «просеивание» словаря N и выбор значимых для принятия решения понятий и их признаков. В идеале на этом шаге образуется полный систематический набор терминов из какой-либо области знаний.

4. Выявление связей между понятиями. Все в мире связано. Но определить, как направлены связи, что ближе, а что дальше, необходимо на этом этапе. Таким образом, строится сеть ассоциаций, где связи только намечены, но пока не поименованы. Например, понятия «день», «ночь», «утро» и «вечер» явно как-то связаны, связаны также и понятия «красный флаг» и «красный галстук», но характер связи тут существенно отличен.

Выявление метапонятий и детализации понятий. Связи, полученные на предыдущем шаге, позволяют инженеру по знаниям структурировать понятия и как выявлять понятия более высокого уровня обобщения (метапонятия), так и детализировать на более низком уровне.

Построение пирамиды знаний. Под пирамидой знаний мы понимаем иерархическую лестницу понятий, подъем по которой означает углубление

понимания и повышения уровня абстракции (обобщенности) понятий. Количество уровней в пирамиде зависит от особенностей предметной области, профессионализма экспертов и инженеров по знаниям. Определение отношений {RA}. Отношения между понятиями выявляются как внутри каждого из уровней пирамиды, так и между уровнями. Фактически на этом шаге даются имена тем связям, которые обнаруживаются на шагах 4 и 5, а также обозначаются причинно-следственные, лингвистические, временные и другие виды отношений.

Определение стратегий принятия решений (S_f). Определение стратегий принятия решения, то есть выявление цепочек рассуждений, связывает все сформированные ранее понятия и отношения в динамическую систему поля знаний. Именно стратегии придают активность знаниям, именно они «перетряхивают» модель M в поиске от X к Y.

Рис. 6.1. Стадии структурирования знаний — алгоритм для «чайников»

Однако на практике при использовании данного алгоритма можно столкнуты с непредвиденными трудностями, связанными с ошибками на стадии извлечения знаний и с особенностями знаний различных предметных областей. Тогда возможно привлечение других, более «прицельных» методов структурирования. При этом на разных этапах схемы (рис. 6.1) возможно использование различных методик.

Основы объектно-структурного анализа (ОСА) были предложены в работах Т.А.Гавриловой. ОСА подразумевает дезагрегацию ПО, как правило, на восемь страт или слоев:

S_1 ЗАЧЕМ-знания Стратегический анализ: назначение и функции системы

S_2 КТО-знания Организационный анализ: коллектив разработчиков системы

S_3 ЧТО-знания Концептуальный анализ: основные концепты, понятийная структура

S_4 КАК-знания Функциональный анализ: гипотезы и модели принятия решения

s_5 ГДЕ-знания Пространственный анализ: окружение, оборудование, коммуникации

s_6 КОГДА-знания Временной анализ: временные параметры и ограничения

s_7 ПОЧЕМУ-знания Каузальный или причинно-следственный анализ: формирование подсистемы объяснений

s_8 СКОЛЬКО-знания Экономический анализ: ресурсы, затраты, прибыль, окупаемость

Разделение стадий извлечения и структурирования знаний является весьма условным, поскольку хороший инженер по знаниям, уже извлекая знания, начинает работу по структурированию и формированию поля знаний, описанному выше.

Однако в настоящее время прослеживается тенденция опережения технологических средств разработки интеллектуальных систем по отношению к их теоретическому обоснованию. Практически сейчас существует пропасть между блестящими, но несколько "постаревшими" математическими основами кибернетики (труды Винера, Эшби, Шеннона, Джорджа, Клира, Йордона, Ляпунова, Глушкова и др.) и современным поколением интеллектуальных систем, которые основаны на парадигме обработки знаний (экспертные системы, лингвистические процессоры, обучающие системы и т. п.).

С одной стороны, это объясняется тем, что с первых шагов наука об искусственном интеллекте (ИИ) была направлена на моделирование слабоформализуемых смысловых задач, в которых не применим традиционный математический аппарат; с другой стороны, ИИ - это ветвь информатики и активно развивается как промышленная индустрия программных средств в условиях жесткой конкуренции, где подчас важнее быстрое внедрение новых идей и подходов, чем их анализ и теоретическая проработка.