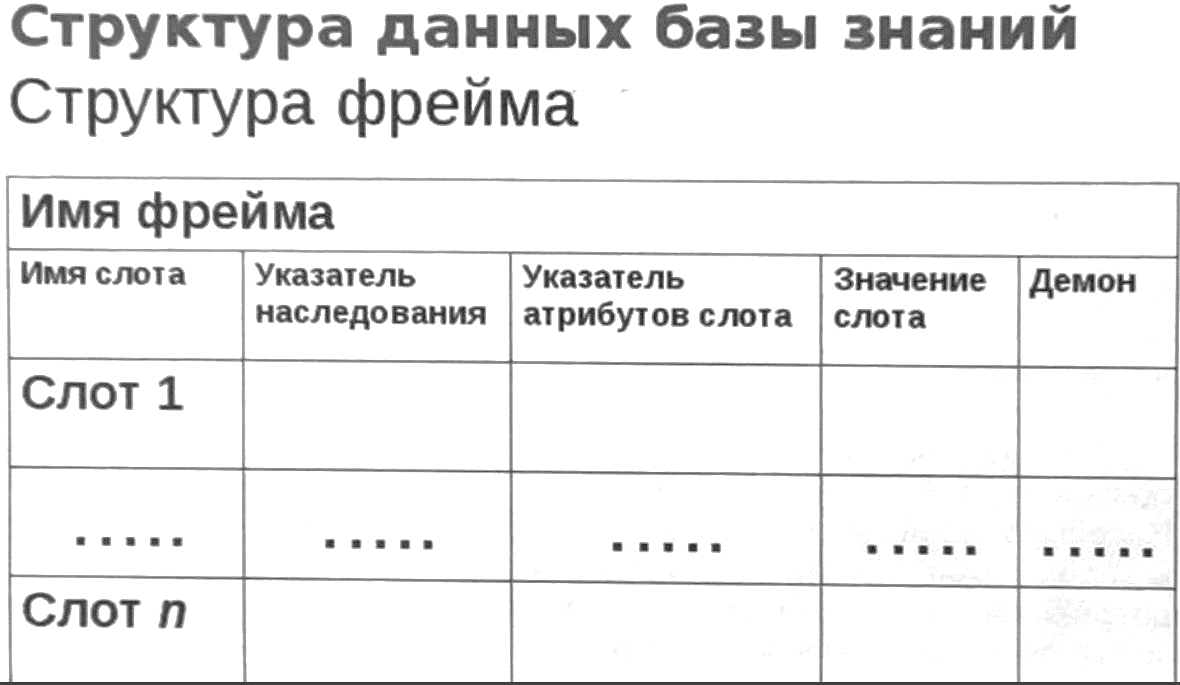
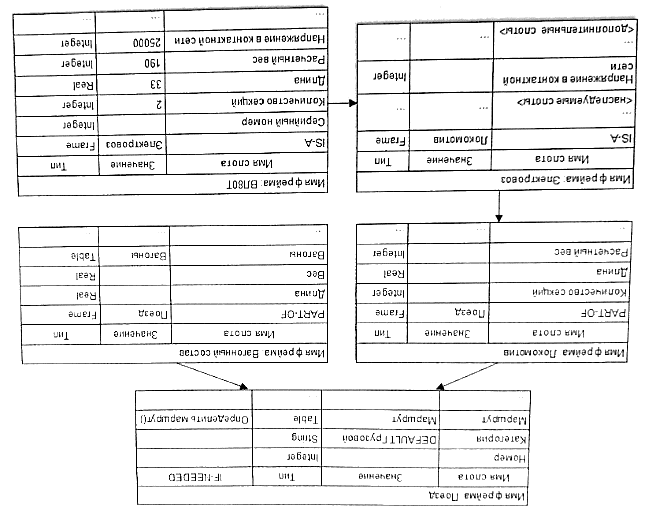
1. **Фреймы.**

**Фрейм** – каркас или рамка – термин был придуман и введён в обращение – Марвин Мински 1975г. **Фрейм** – модель реальной или воображаемой сущности. **Фрейм** – единица представления знаний, запомненная в прошлом, детали которой при необходимости могут быть изменены согласно текущей ситуации. Фреймы бывают: **1). Фрейм образец** – прототип, или протофрейм, – множество сущностей, имеющих общую структуру и поведение. Он выступает шаблоном на основе которого создаются конкретные сущности. **2). Фрейм экземпляр** – это конкретная сущность, которая характеризуется индивидуальностью, т. е. индивидуальным состоянием или поведением. **Индивидуальность** – свойство сущности, с помощью которого её можно отличить от других. Фрейм экземпляр имеет свой уникальный идентификатор. Фрейм экземпляр состоит из слотов, конкретное значение слотов определяет текущее состояние фреймов экземпляров, а набор демонов и присоединённых процедур, а также алгоритмическая реализация определяют поведение фрейма экземпляра. **3).** Есть **фреймы структуры**, которые используются для обозначения объектов и понятий. **4). Фреймы роли.** **5). Фреймы сценарии. 6). Фреймы ситуации.**

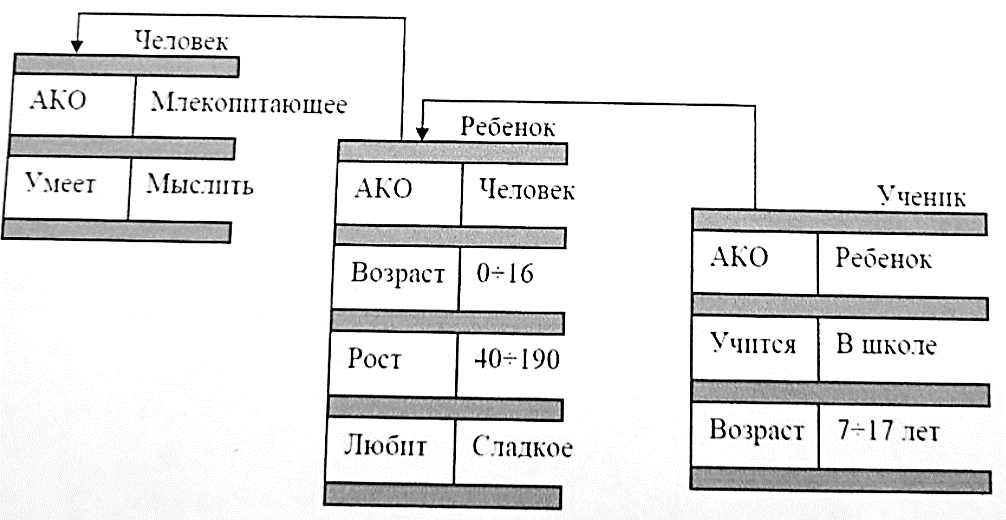
1. **Структура данных БЗ. Структура фрейма.**

  
В общем случае фрейм задаётся следующими параметрами: **1). Имя фрейма** ­– служит для идентификации фрейма и должно быть уникальным. Фрейм представляет собой совокупность слотов. **Слот** – атрибутивная составляющая или декларативные знания и присоединённые процедуры. Присоединенные процедуры – процедурные знания. **2). Имя слота** – идентификатор атрибута сущности, который наделен определенной семантикой. Имя слота должно быть уникальным в пределах фрейма, в качестве имени может выступать произвольный текст или зарезервированное обозначение – системное имя слота. **IS-A** обозначает это, **PART-OF** является частью, **RELATION** относится, **AKO** это есть. Указатели наследования определяют правила заполнения слота. Они характерны только для фреймовых систем иерархического типа. **В конкретных системах указатели могут быть реализованы различными способами: 1).** По умолчанию от фрейма образца. **2).** Через наследование от родительского фрейма указанного в слоте из IS-A или AKO. **3).** По умолчанию или через наследование, но в случае необходимости значение слота может быть изменено, в этом случае используется латинская буква O. **4).** Значения слота для всех фреймов экземпляра одного типа должно быть уникальным, ставится буква U. **5).** Значение слота для всех фреймов экземпляров одного типа должно находиться в пределе указанном в слоте фрейма образца, ставится R. **1).** По формуле указанной в слоте. **2).** Через демон. **3).** Явно из диалога. **4).** Тип Данных (атрибуты) Определяет тип значения слота: фрейм – указатель на фрейм real. Integer – целое число. Boolean – логический тип, text – фрагмент текста, list – список, table – таблица, может быть выражение или присоединённая процедура. **5).** Значения слота должно соответствовать указанному типу данных и условию наследования. **6).** Демон – IF-NEEDED, IF-ADDED, IF-REMOVED, IF-DELETED. Демон IF-NEEDED запускается, если в момент обращения к слоту его значение не было установлено. IF-ADDED запускается при попытке создания фрейма экземпляра. IF-REMOVED или IF-DELETED запускается при попытке удаления значения слота. **7).** Присоединенная процедура – совокупность процедур определяет поведение фреймов, присоединенная процедура демон запускается автоматически или при обращении к этой процедуры из процедуры других фреймов. Процедуры объект управления фреймовой системой.

1. **Пример фреймового представления БЗ.**

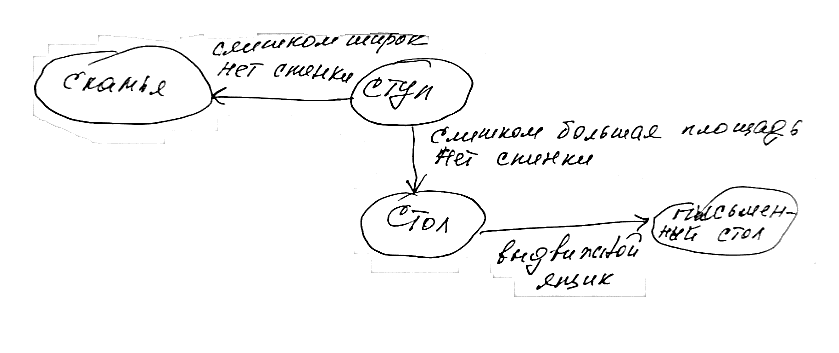


1. **Организация логического вывода на фреймах.**

Наличие иерархической связи позволяет в рамках фреймовой системы организовать логический вывод.   
Пример: Пусть имеется система фреймов.  


**Слот АКО** – системный слот, значением которого является ссылка на родительский фрейм из которого неявно наследуются некоторые свойства фрейма-образца. Вопрос – любит ли ученик сладкое? Ответ – да. Этот вывод сделан на основании наследования свойства фрейма ребёнок, из которого берётся слот любит сладкое. Может ли ученик мыслить? Да. Вывод сделан на основании наследования слота умеет мыслить человека.

1. **Свойства фреймов.**

**1). Иерархическая структура** – фрейм обычно представляет собой иерархическую структуру. Особенности такой структуры заключаются в том, что информация о слотах, которые содержит фрейм верхнего уровня, используется всем фреймами нижнего уровня. **2). Межфреймовые сети** – осуществление выводов во фреймовой системе возможно благодаря соединению в межфреймовые сети фреймов, которые описывают объекты с небольшими различиями с указателями различия. Например:   
  
**3) Значение по умолчанию** – фреймы обладают способностью наследовать значения слотов своих родителей. **4). Отношение абстрактное** ­­­– конкретное, целое – часть. Отношение абстрактное-конкретное – это на верхних уровнях иерархии располагаются абстрактное объекты, а внизу – конкретные, они регламентируются отношениями типа **IS, ARE** или **KIND OF**. **Отношение целое** – часть касается только структурированных объектов и показывает, что объект нижнего уровня является частью объекта высшего уровня.

1. **Преимущества и недостатки фреймов.**

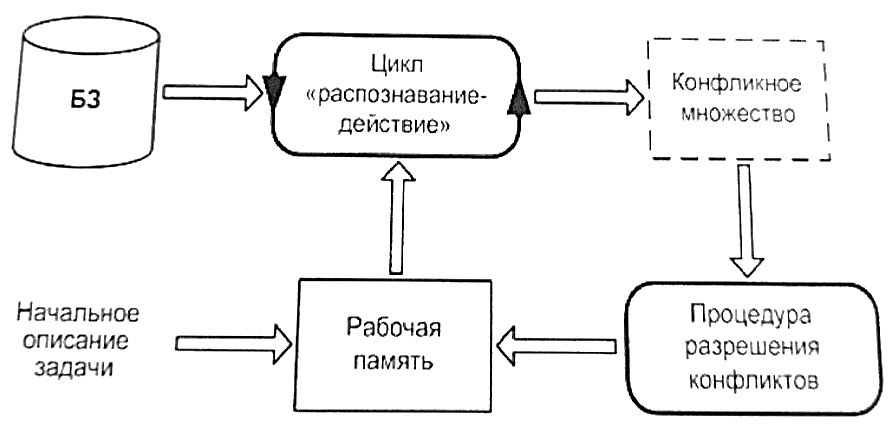
**Преимущества:** 1). Гибкость. 2). Наглядность. 3). Значение может быть вычислено с помощью процедур или получено эвристическими методами. **Недостатки:** 1). Высокая сложность системы в целом. 2). Очень трудно внести какие-то изменения в иерархическую систему. 3). Затруднена обработка исключений.

1. **Общая структура продукционных правил.**

Продукционные модели были предложены Эмилем Леоном Постом в 1943г. Было проведено исследование решений и принятия действий человеком и сделан вывод, что человек использует продукционные правила. **Продукционная модель представления знаний** основана на правилах, позволяющих представить знания в виде предложений. **Два типа предложений:** 1). Если условие, то действие. 2). Если условие A, то действие B, иначе действие C. Но на практике чаще используется первое. В продукционной системе БЗ состоит из **БД** и **Базы Правил.** БД содержит факты, описывающие входные данные и состояние системы. База Правил содержит набор продукционных правил в общем виде (первого типа). Условие и действие могут быть связаны логическими союзами **И** и **ИЛИ**, а также использоваться **НЕ**. Например: Если условие1 или условие2, то действие1 и действие2. На практике очень редко можно встретить правила в которых больше двух условий или действий.

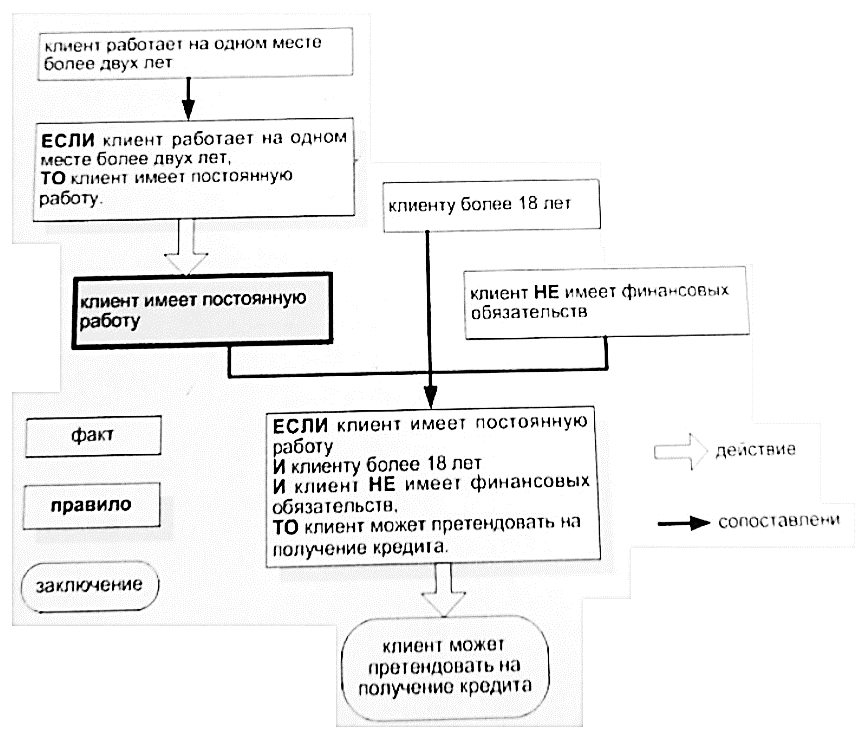
1. **Архитектура продукционной системы.**

БЗ, рабочая память и цикл управления.



Продукционная БЗ состоит из след. ед. **1). Предусловие** – условие срабатывания продукции, оно записывается в виде предиката и выполнимость продукции обозначается как логическая истинна. **2). Условие** – оно разрешает или запрещает выполнение процедур и устанавливает порядок выполнения. **3). Ядро** – состоит из правил. **4). Постусловие** – оно содержит рекомендацию по использованию результатов продукции. Моделирование решения задачи основано на процессе сопоставления с образцом в ходе которого текущее состояние решения сравнивается с имеющимися знаниями для определения дальнейших действий. В управляющем цикле распознавание действий осуществляется сравнение образцов из рабочей памяти с условными частями правил из БЗ. **Допустимые продукции**, т. е. те которые согласованы с текущим сост. рабочей памяти, помещаются в конфликтное множество после этого осуществляется процесс разрешения конфликтов. Разрешение конфликтов производится определенными процедурами цикл распознавания действий происходит до тех пор пока образец рабочей памяти не будет более соответствовать одному из условий рабочей памяти.

1. **Цепочка вызовов.**



Эта цепочка говорит как работает цепочка вывода для определенной продукции. Цепочки вывода имеют две разновидности: **1). Монотонный вывод** – вывод, при котором факты не удаляются из рабочей памяти. **2). Немонотонный вывод** допускает удаление фактов из рабочей памяти. При немонотонном выводе существенную роль играет порядок применения продукционных правил. В процессе вывода решения все правила системы равнозначны между собой и самодостаточны. **Самодостаточность** – это когда всё необходимое для реализации правила содержится в его условии. И ещё одни правила не могут непосредственно вызывать другие правила.

1. **Разрешение конфликтов.**

Порядок активации правил конфликтного множества определяется выбранной стратегией разрешения конфликтов. **Простые стратегии по разрешению конфликтов** основываются на том, что выбирается либо первое, либо последнее правило входящее в список. Выбор первого правила соответствует поиску в ширину, а выбор последнего – в глубину. Во многих продукционных системах чаще применяется поиск в глубину, но это были простейшие стратегии. Кроме этого есть и **более сложные стратегии разрешения конфликтов**. **1). Принцип стопки книг** – основан на идее, что наиболее часто используемая продукция является самой полезной. Если возникло конфликтное множество, то выбирается та продукция у которой наибольшая частота использования. Но в этом случае необходимо вести статистику частоты использования, считать как использовалась та или иная продукция. Управление по принципу стопки книг целесообразно применять если продукции независимы друг от друга. **2). Принцип наиболее длинного условия** – он заключается в том, что из конфликтного множества выбирается та продукция, у которой стало истинным наиболее длинное условие. Этот принцип опирается на соображение здравого смысла, который говорит о том, что правило которое относится к узкому классу ситуации всегда важнее, чем правило относящееся к широкому классу. Управление этим методом целесообразно применять, когда знания и сами продукции хорошо структурированы по привязкам к типовым ситуациям. **3). Принцип метапродукции** – этот принцип основан на идее ввода в систему продукций специальных метапродукций. **4). Принцип классной доски** – зачастую принцип классной доски комбинируется совместно с помощью метапродукций. **5). Принцип приоритетного выбора** – он связан с ведением статических или динамических приоритетов на продукции. Статический приоритет он обычно присваивается продукции сразу и существует всё время. Динамический присваивается в процессе работы по каким-то определенным правилам. **6). Управление по имени** – похожа на стратегию приоритетного выбора, но вместо приоритета задаётся имя продукции. **7). Разбиение на подзадачи**, принцип декомпозиции – декомпозиция даёт хороший результат только для хорошо структурированных областей знаний. В этом случае можно разбить задачу на подзадачи, причем разбиение задачи на подзадачи необходимо делать оптимальным способом.

1. **Направления вывода.**

**1). Вывод на основе данных**. В этом случае процесс решения задачи начинается с исходных фактов, затем применяя допустимые правила, осуществляется переход к новым фактам. И так до тех пор, пока цель не будет достигнута. Этот процесс называется прямой цепочкой вывода. **2). Вывод от цели.** Начинается от одной из допустимых целей и рассматриваются пути, которые ведут к достижению цели. Таким образом определяется последовательность правил, позволяющая найти решение. Процесс повторяется для всех заданных задач цели. Такой способ поиска называют обратной цепочкой вывода. **3).** Существуют системы для которых вывод основывается на сочетании упомянутых выше выводов. Такой комбинированный метод получил название **циклического**. **Прямая цепочка рассуждений используется в следующих случаях:** **1).** Когда на основании имеющихся фактов необходимо определить тип объекта или явления, выдать рекомендацию или поставить диагноз. **2).** Все или большинство данных заданы в пространстве задачи. **3).** Существует большое количество потенциальных целей, но есть только несколько способов представления исходных фактов. **4).** Сформировать цель или гипотезы очень трудно в силу избыточности исходных данных или большого количества конкурирующих гипотиз. **Обратная цепочка рассуждений применяется в следующих случаях:** **1).** Для задач соответствующих процессу проверки гипотиз при решении проблем человеком. **2).** Цель поиска явно присутствует в постановке задачи или может быть легко сформулирована. **3).** Имеется слишком большое число правил, которые на основе исходных фактов продуцируют возрастающее число заключений или целей своевременный отбор цели позволяет отсеять множество тупиковых ветвей. **4).** Исходные данные не приводятся в задаче, но подразумевается, что они могут быть известны или легко получены.

1. **Пример прямого вывода.**

Пусть в базе правил имеются следующие правила: 1). **Прaвило 1** – если двигатель не заводится, то сел аккумулятор. 2) **Правило 2** – если указатель бензина на 0, то двигатель не заводится. Предположим что в рабочую память от пользователя **поступили факты:** 1) фары не горят. 2) бензин на нуле. **Рассмотрим алгоритм прямого вывода:** **1).** Сопоставим факты из рабочей памяти и правила из БП. Правило 1 не срабатывает, а правило 2 срабатывает, т. к. указатель бензина находится на 0, совпадает с условием. **2).** Поскольку правило 2 сработало, то в рабочую память заносится заключения этого правила, а именно двигатель не заводится. **3).** Второй цикл сопоставления фактов в рабочей памяти с образцами правил. Теперь правило 1 сработает, т. к. двигатель не заводится и фары не горят. **4).** В рабочую память заносится действие правила 1, т. е. заключение сел аккумулятор. **5).** Конец работы, т. к. БП исчерпана.

1. **Пример прямого вывода с конфликтным набором.**

Пусть в базе правил имеются следующие правила: 1). **Прaвило 1** – если двигатель не заводится, то сел аккумулятор. 2) **Правило 2** – если указатель бензина на 0, то двигатель не заводится. Теперь допустим, что кроме правила 1 и правила 2, присутствует **правило 3** – если показатель бензина на 0, то нет бензина. В рабочей памяти находятся теже правила, что и в предыдущем примере. Предположим что в рабочую память от пользователя **поступили факты:** 1) фары не горят. 2) бензин на нуле. В результате рабочего цикла может примениться правило 2 и 3, **возникает конфликтная ситуации** и какое правило применять первым. Если выберем правило 2, то в рабочую память добавится двигатель не заводится. Если выберем правило 3, то мы не найдём вывод. Поэтому нужно будет вернуться к правилу 2. В этом случае результат будет достигнут за 3 шага.

1. **Пример обратного вывода.**

Пусть в базе правил имеются следующие правила: 1). **Прaвило 1** – если двигатель не заводится, то сел аккумулятор. 2) **Правило 2** – если указатель бензина на 0, то двигатель не заводится. Предположим что в рабочую память от пользователя **поступили факты**: 1) фары не горят. 2) бензин на нуле. **Алгоритм обратного вывода содержит следующие шаги.** Выдвигается гипотеза. Отыскивается правило, вывод которого соответствует выдвинутой гипотезе, в нашем случае это правило 1. Исследуется возможность применения правила 1, т. е. проверяется может ли оно сработать или нет, но оно сработать не может, потому что в рабочей памяти должны присутствовать факты совпадающие с условием данного правила. Правило 1 не может сработать из-за отсутствия в памяти образца двигатель не заводится, вот этот факт становится новой целью или новой гипотезой. Отыскиваем правило заключение которого соответствует новой гипотезе, это правило 2. Исследуется вохможность применения правила 2, оно срабатывает т. к. в памяти присутствует факт совпадающий с его условием. Правило 2 действует и в рабочую память заносится заключение – двигатель не заводится. Условная часть правила 1 теперь подтверждена фактами, следовательно оно сработает и выдвинутая самая первая гипотеза подтверждается. Конец работы.

1. **Пример обратного вывода с конфликтным набором.**

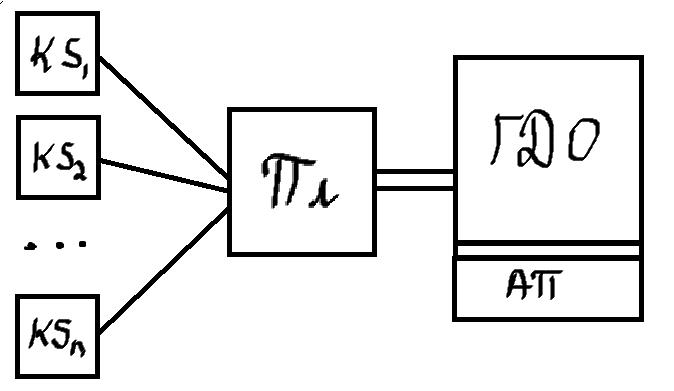
Пусть в базе правил имеются следующие правила: 1). **Прaвило 1** – если двигатель не заводится, то сел аккумулятор. 2) **Правило 2** – если указатель бензина на 0, то двигатель не заводится. Теперь допустим, что кроме правила 1 и правила 2, присутствует **правило 3** – если показатель бензина на 0, то нет бензина. Добавим **правило 4**, если засорился бензонасос, то двигатель не заводится. Предположим что в рабочую память от пользователя **поступили факты**: 1) фары не горят. 2) бензин на нуле. **Алгоритм данного действия:** **1)** Выдвигается гипотеза сел аккумулятор. **2).** Ищем правило заключение которого совпадает с поставленной целью. Это правило 1. **3).** Исследуется возможность применения правила 1, оно не может сработать, выдвигается новая гипотеза, двигатель не заводится. **4).** Поиск правил заключение которых совпадает с новой подцелью, это правило 2, 4. Если выберем правило 2, то дальнейшее совпадет с примером бесконфликтного набора, а если – 4, то оно не сработает, т.к. в рабочей памяти нет образца засорился бензонасос, после этого опять вернёмся к правилу 2. Следует обратить снимание, что правило 3 не связанное с поставленной целью не затрагивалось в процессе вывода, это свидетельство того, что ненужная продукция просто была пропущена.

1. **Предусловия, постусловия.**

**Предусловия** выражают ограничения выполнения которых необходимо для корректной работы программы. **Предусловие** применяется ко всем видам и вызовам программ как внутри класса так и у клиента. Корректная программа никогда не вызовет программу, когда невыполнено предусловие. **Постусловие** выражает свойство или состояние завершающее выполнение программы. **Постусловие** выражает гарантию того, что программы завершат выполнение и приведут её к состоянию с заранее заданными свойствами.

1. **Метод доски объявлений. Архитектура доски объявлений.**

Такая методика **будет эффективна только тогда**, если выполняются следующие требования: **1).** Различные модули правил должны иметь одинаковый синтаксис, одинаковые обозначения, одинаковые имена объектов. Это нужно для того чтобы обеспечивать передачу знаний между модулями. **2).** Должен существовать протокол определения очерёдности выполнения правил, который вступает в силу тогда, когда сразу несколько модулей готовы изменить содержание доски объявлений. С точки зрения организации процесса вычислений в системе можно выделить следующие структурные компоненты, которые показаны на схеме архитектуры доски объявлений.



**ГДО** – глобальная доска объявлений. Сюда записываются исходные данные, а также промежуточные и конечные результаты решения задачи. Внизу есть подраздел АП – это активизированные правила. **Пл** – это планировщик. И **КSi** – это независимые источники знаний. То есть если источник знаний находит на доске объявлений исходные данные необходимые ему, он формирует запрос через планировщик. Планировщик даёт ему доступ к глобальной доске объявлений. Результаты на доску объявлений передаются через планировщик. Как записан результат доступ у данного источника знаний прекращается. Источники знаний общаются между собой только через доску объявлений и не могут непосредственно передавать друг другу или запускать выполнение процедур. Эта архитектура показала **высокую эффективность работы,** **но** модульная организация памяти **требует больших ресурсов**, **поэтому** системы построенные на этом принципе **работают медленно, тем не менее** этот метод **считается очень перспективным**.

1. **Таблица принятия решений.**

Таблица состоит из разделов **условия**, **действия** и **правил**. Таблица строится таким образом, что условия должны принимать состояния **true** или **false**.

Пример: Ввод: a, b, c. Вывод: определить тип треугольника (равносторонний, равнобедренный, разносторонний, не треугольник). Шаг 1 (определить условия): Условие 1: a, b, c образуют треугольник? (Y, N). Условие 2: a = b? (Y, N). Условие 3: a = с? (Y, N). Условие 4: b = c? (Y, N). Шаг 2 (посчитать комбинации): Если не все условия простые ({Y, N}): Кол-во комбинаций = 2 ^ (кол-во условий). Если условия сложные ({A, B, C, D}). Кол-во комбинаций = VC1 \* VC2 \* VC3 \* VCn. N = 2 ^ x = 16. Шаг 3-4 (заполнить комбинации):



1. **Представление знаний семантическими сетями.**

**Семантическая сеть** или смысловая сеть – это модель предметной области, представленная в виде графа, где **вершины** **– это понятия**, а **дуги – отношения между понятиями**. В качестве понятий обычно выступают абстрактные или конкретные объекты. В качестве отношений используется следующие отношения согласно смысловой классификации: 1). **Таксономические отношения** – данный тип отношения определяется отношениями типа **АКО** или **IS, ARE**. **Таксономические отношения** – это класс подкласс экземпляра, или же множество подмножества элемента. 2). **Структурные** – это отношения типа **часть – целое**, **PART OF**. 3). **Родовые** – это предки – потомки. 4). **Производственные** – это начальник – подчиненный. 5) **Функциональные** – определяются глаголами. 6). **Количественные**. 7). **Пространственные**. 8). **Временные**. 9). **Атрибутивные** – имеющие свойства или конкретное значение. 10). **Логические** – **И**, **ИЛИ**, **НЕ**. 11). **Казуальные** (причино-следственные). **Отношения также классифицируют по степени участия:** 1). **Унарные** (рекурсивные) 2). **Бинарные** – связывает два понятия. 3) **N-арное** ­– отношение связывается более двух понятий. Семантические сети используют в экспертных системах для представления знаний. Есть такая система Проспектор и используют в системах распознавания речи для понимания естественного языка.

1. **Классификация семантических сетей.**

**По назначению:** включает несколько типов отношений: 1). **Классифицирующие.** 2). **Функциональные** – вычислительная модель, позволяющая описывать одни вычислительные единицы через другие. 3) *Сценарий*. **Следующая классификационная единица:** 1). **Однородные** – один тип отношений 2). **Неоднородные** – количество типов больше одного. Неоднородные сети самые сложные. Следующая классификационная группа – **по арности.** Последняя **по размеру**: 1). **Для решения конкретных задач**. 2). Семантические сети **отраслевого маштаба.** 3). **Глобальная** семантическая сеть.

1. **Семантические отношения.**

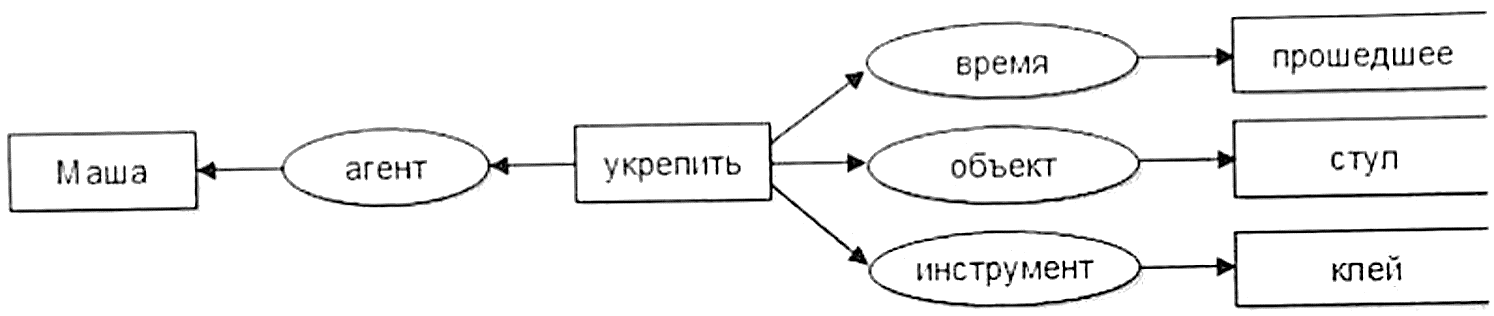
**Основные семантические отношения:** 1). **Иерархические** – отношение типа IS, ARE, AKO, PART OF. 2). **Часть чего-то**. 3). **Вспомогательные** – все остальные.

1. **Достоинства и недостатки семантических сетей.**

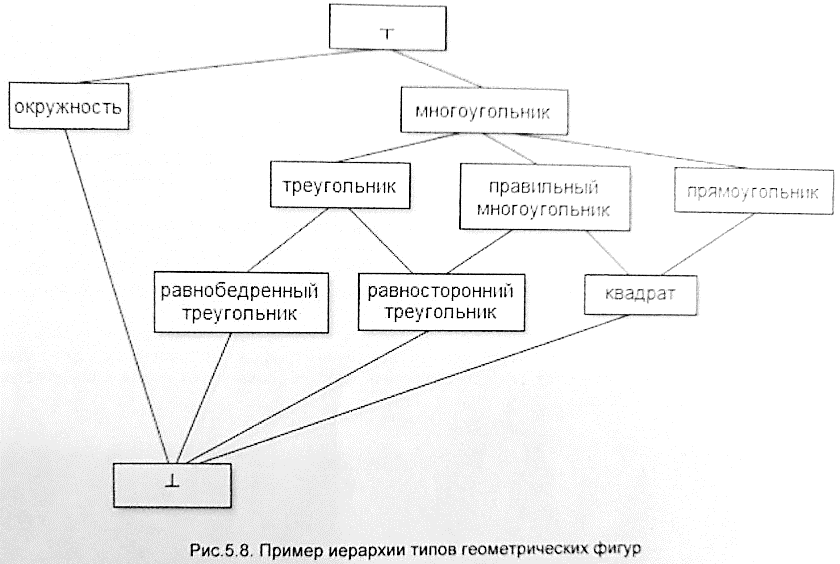
1). **Универсальность** – в принципе с помощью семантической сети можно описать сколь угодно сложную ситуацию. 2). **Наглядность системы знаний**. 3). **Близость структуры сети**, представляющей систему знаний к семантической структуре фраз на естественном языке. 4). **Соответствие** современным представлениям об устройстве **памяти человека. Недостатки семантических сетей:** **1).** Сетевая модель не даёт ясного представления о структуре предметной области. Поэтому формирование и модификация такой модели затруднены. **2)**. Сетевые модели представляют собой пассивные структуры для обработки которых необходим специальный аппарат формального вывода. **3).** Проблема поиска решения в семантической сети сводится к задаче поиска фрагмента сети. Этот фрагмент должен соответствовать подсети, которая отражает поставленный запрос. Это в свою очередь обуславливает сложность поиска решения в семантических сетях. **4).** Представление, использование и модификация знаний при описании системы реального уровня сложности оказывается очень трудоёмкой процедурой. Особенно при наличии множественных отношений между её понятиями.

1. **Концептуальные графы. Иерархия типов.**

**Концептуальный граф** – это двудольный ориентированный граф, состоящий из вершин двух типов, понятий и концептуальных отношений. **Двудольный граф** – это граф множество вершин которого можно разбить на части таким образом, что каждое ребро графа соединяет какую-либо вершин из одной части с какой-то вершиной другой части, но несуществует ребра соединяющего 2 вершины из одой и той же части. **Понятия** в концептуальном графе отображаются **прямоугольниками**, а **отношения** – **элипсами**. Пример: Маша укрепила стул клеем.



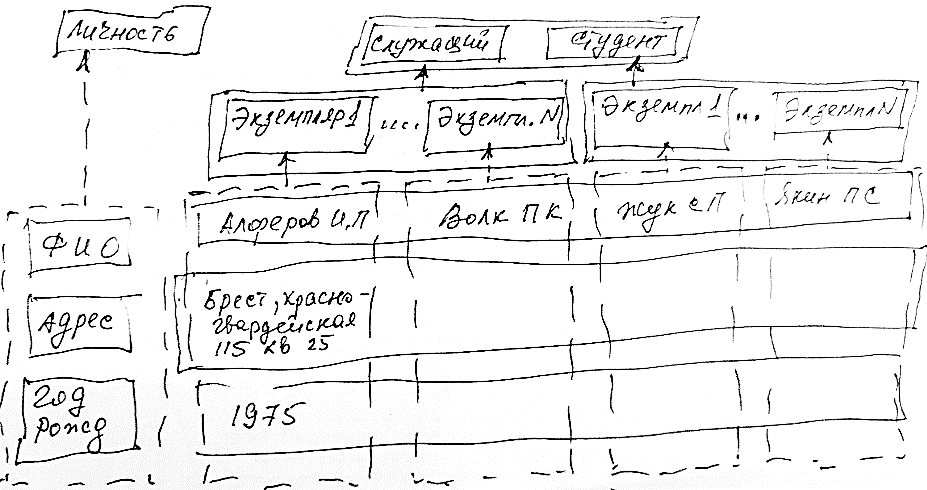
**Иерархия типов.** Отдельнаяиерархия представляется в виде решётки, описывающей таксономические отношения между понятиями, включающие множественное наследование. Для представления иерархии типов в виде решётки в неё включаются **два специальных типа**: 1). **Универсальный тип**, который является супер-типом для всех типов и обозначается T. 2). **Абсурдный тип**, который является подтипом для всех типов и обозначается .



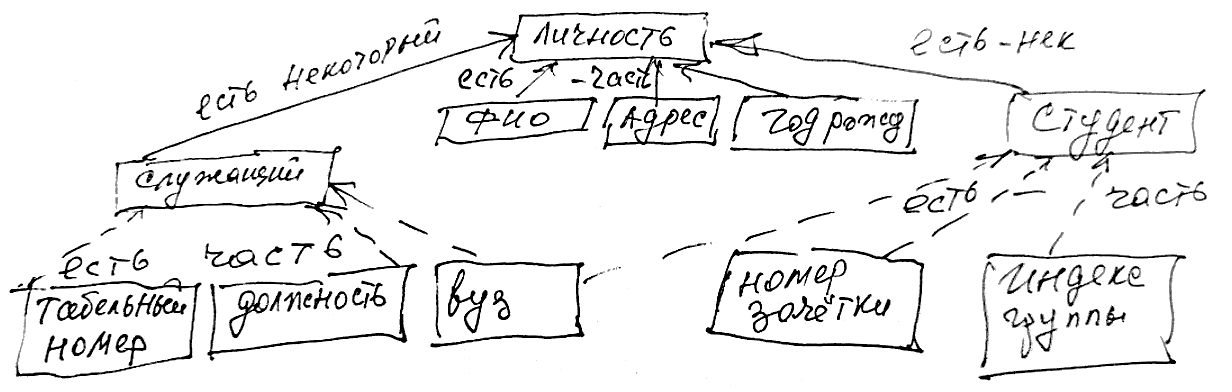
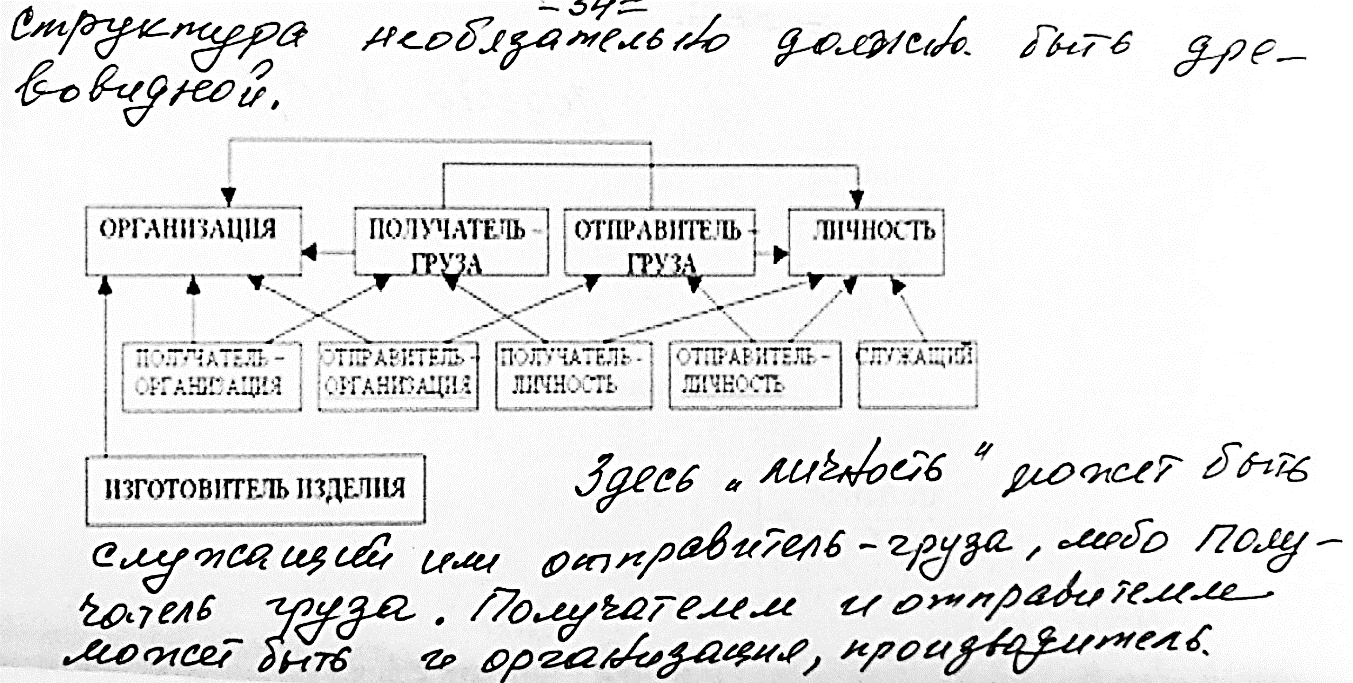
**Вилы операций:** 1). **Копирование**, т. е. создание точной копии какого. 2). **Объединение двух графов в один**, если они имеют общие вершины-понятия. 3). **Ограничения** – это замена вершины понятия графа другой вершиной понятием из другого графа. 4). **Упрощение** – исключение дублирующих отношений или понятий.

1. **Основные концепции реализуемые в семантических сетях.**

Их всего 4: 1). **Концепция одновременного рассмотрения в модели как знака так и типа**. **Знак** – это конкретное значение или конкретный экземпляр рассматриваемого объекта. **Тип** – это класс подобных знаков. 2). **Концепция иерархии типов** – это когда обобщение позволяет соотнести множество знаков или множество типов с одним общим типом. Различают обобщения: знак-тип – это классификация, тип-тип – это и есть обобщение. Процессом обратным процессу классификации является **экземпляризация**, т. е. порождение экземпляров. А процессом обратным процессу обобщения является **специализация**. Агрегация позволяет раскрыть структуру объектов. То есть допускается многократное применение агрегации, т. е. любую структуру можно представить как агрегат базовых типов, а сам агрегат, может служить объектом обобщения базовых типов. Агрегация соотносится с понятиями, есть часть или есть некоторый.



3). **Роли**. Концепция роли задаётся либо древовидной либо сетевой.  
**Древовидная**:

  
**Сетевая**:  


4). **Концепция семантического расстояния**, эта концепция очень широко используется в словарных системах, где одно слово может быть истолковано посредством других слов, а они посредством третьих и т. д. В этом случае используется **мера семантической близости** взаимосвязанных понятий. В качестве меры семантической близости взаимосвязанных понятий, если эти понятия представлены в модели вершинами, то той мерой служит **число дуг от одной вершины к другой**.

1. **Вывод информации в семантических сетях.**

Рассмотрим наиболее распространённый подход **вывода основанный на сопоставлении частей сетевой структуры**. Он заключается в следующем, сначала строится подсеть соответствующая запросу, затем выполняется сопоставление этой подсети с базой знаний. Если управляющая система находит сопоставление это подсети с частью БЗ, то формируется ответ на вопрос.

