OSTIS-2012
(Open Semantic Technologies for Intelligent Systems)

**УДК 004.822:514**

# ГРАФОДИНАМИЧЕСКИЕ МОДЕЛИ ПАРАЛЛЕЛЬНОЙ ОБРАБОТКИ ЗНАНИЙ: ПРИНЦИПЫ ПОСТРОЕНИЯ, РЕАЛИЗАЦИИ И ПРОЕКТИРОВАНИЯ

Голенков В. В., Гулякина Н.А.

[*] *Белорусский государственный университет информатики и радиоэлектроники, г. Минск, Республика Беларусь*

golen@bsuir.by
guliakina@bsuir.by

Рассматриваются принципы построения технологии проектирования интеллектуальных систем, ориентированной на семантическое представление знаний, расширение контингента разработчиков и сокращение сроков проектирования.
**Ключевые слова:** интеллектуальная система, технология проектирования интеллектуальных систем, семантическая сеть, язык семантических сетей, алфавит семантических сетей, ключевые узлы семантических сетей, бинарная семантическая сеть, предметная область.

## ВВЕДЕНИЕ

Мы исходим из того, что основным результатом исследований в области искусственного интеллекта являются не столько разработка конкретных интеллектуальных систем, обладающих высокими интеллектуальными возможностями, сколько **разработка технологий** позволяющих быстро и в большом количестве порождать самые разнообразные интеллектуальные системы, имеющие большую практическую ценность. Очевидно, что составляющими таких технологий являются:

- формальная теория интеллектуальных систем (формальное описание того, как они устроены);
- методы проектирования интеллектуальных систем;
- инструментальные средства (средства автоматизации проектирования интеллектуальных систем);
- средства информационной поддержки (информационного обслуживания) разработчиков интеллектуальных систем;
- средства компьютерной поддержки управления коллективной разработкой интеллектуальных систем.

Современные технологии проектирования интеллектуальных систем имеют ряд недостатков:

- технологии искусственного интеллекта не ориентированы на широкий круг разработчиков интеллектуальных систем и, следовательно, не получили массового распространения;
- велики сроки разработки интеллектуальных систем и велика трудоемкость их сопровождения;
- высока степень зависимости технологий искусственного интеллекта от платформ, на которых они реализованы, что приводит к существенным изменениям технологий при переходе на новые платформы;
- для эффективной реализации даже существующих моделей представления знаний и моделей решения трудно формализуемых задач современные компьютеры оказываются плохо приспособленными, что требует разработки принципиально новых компьютеров;
- современно состояние в области проектирования интеллектуальных компьютерных систем представляет собой "вавилонское столпотворение" самых различных моделей, методов, средств, платформ;
- отсутствуют подходы, позволяющие на некоторой универсальной основе интегрировать научные и практические результаты в области искусственного интеллекта, что порождает высокую степень дублирования результатов. В частности, высока трудоемкость интеграции различных моделей представления информации, моделей обработки информации, моделей решения задач и, следовательно, различных интеллектуальных компьютерных систем.

Искусственный интеллект является междисциплинарной научной дисциплиной. Этим обусловлен большой её потенциал, так как на стыках научных направлений рождаются сильные результаты. Но этим же обусловлены и большие трудности, так как развитие искусственного интеллекта требует глубокого взаимопонимания и

сотрудничества исследователей, имеющих разный стиль мышления, разный подход к объекту и предмету исследования, разный менталитет, разные целевые установки и традиции. Современный этап развития искусственного интеллекта остро нуждается в преодолении указанных трудностей.

Важнейшей задачей искусственного интеллекта в настоящее время является построение общей комплексной теории интеллектуальных систем, в рамках которой бы сочетались самые разные направления искусственного интеллекта – и теория представления знаний, и теория решения задач (в том числе различные исчисления, эвристики, стратегии), и теория программ (процедурных, декларативных, параллельных, последовательных), и архитектуры интеллектуальных систем, в том числе детализированные до уровня аппаратной поддержки, и теория интеллектуальных пользовательских интерфейсов, и компьютерная лингвистика.

Сейчас эпицентром развития искусственного интеллекта является не столько разработка отдельных его направлений, сколько их глубокая **семантическая интеграция**, целью которой должна быть не только общая теория интеллектуальных систем, но и общая, доступная технология их комплексного проектирования.

В основе предлагаемого нами подхода к созданию технологии проектирования интеллектуальных систем, направленного на устранение выше указанных недостатков, лежат следующие принципы.

## Принцип 1. Использовать опыт наиболее продвинутых технологий

В первую очередь, имеется в виду технология проектирования микросхем, которая за последнее время обеспечила существенное сокращение времени и повышение качества разработок,

(1) благодаря созданию языковых средств формального описания проектируемых микросхем на разных уровнях детализации,

(2) благодаря четкому разделению процесса разработки формальных описаний микросхем и процесса их реализации по заданным формальным описаниям,

(3) благодаря созданию мощных и доступных библиотек формальных описаний типовых (многократно используемых) компонентов микросхем.

Для того, чтобы аналогичным образом построить технологию проектирования интеллектуальных систем, необходимо

(1) создать языковые средства полного унифицированного формального описания интеллектуальных систем,

(2) четко отделить разработку полного унифицированного формального описания проектируемой интеллектуальной системы от разработки различных вариантов интерпретации таких формальных описаний интеллектуальных систем,

(3) создать библиотеку формальных описаний типовых (многократно используемых) компонентов интеллектуальных систем. Но для того, чтобы такая библиотека была создана, необходимо обеспечить **интегрируемость** (семантическую совместимость) указанных компонентов интеллектуальных систем.

## Принцип 2. Графодинамические модели:

В качестве формальной основы проектируемых интеллектуальных систем использовать **графодинамические модели обработки информации**.

Графодинамическая модель обработки информации трактует процесс обработки информации как процесс преобразования графовой структуры, в ходе которого меняется не только состояние элементов этой графовой структуры, но и конфигурация этой структуры (появляются или удаляются её вершины, а также связи между ними). Заметим, что для создания графодинамических моделей обработки информации недостаточно тех видов графовых структур, которые в настоящее время исследуются в теории графов. Нам потребуется не только увеличение числа компонентов инцидентных ребру (т. е. переход от ребра к **гиперребру**), но и увеличение числа компонентов инцидентных дуге (т. е. переход от дуги к **гипердуге**, которая по сути является графовой трактовкой многоместного кортежа). Нам потребуются не только ребра, гиперребра, дуги, гипердуги, компонентами которых являются вершины графовой структуры, но и ребра, гиперребра, дуги, гипердуги, компонентами которых являются другие ребра, гиперребра, дуги и гипердуги. Нам потребуются такие связующие элементы графовых структур, которые задают целые фрагменты (подструктуры) заданной графовой структуры, в состав которых входят соответствующие вершины, ребра, гиперребра, дуги, гипердуги [Попков, 1986].

Приведем общее определение **графовой структуры,** на основе которого можно строить практически полезные графодинамические модели обработки информации. Графовая структура $G$ задается пятеркой $< V, C, K, M, I >$, где

$V$ – множество **вершин** (первичных элементов, терминальных элементов);

$C$ – множество **связующих элементов** графовой структуры, каждый из которых задает некоторый фрагмент графовой структуры;

$K$ – множество **ключевых вершин** графовой структуры, каждая из которых задает некоторый класс эквивалентных (однотипных) в определенном смысле элементов графовой структуры ($K \subset V$);

$M$ – множество **меток** элементов (алфавит элементов) графовой структуры, каждая из которых задает некоторый базовый класс эквивалентных в определенном смысле элементов графовой структуры. К таким классам элементов, в частности, относятся:
- класс всех вершин графовой структуры;
- класс всех связующих элементов графовой структуры;
- класс всех ключевых вершин графовой структуры;
- класс всех меток графовой структуры;
- класс всех используемых в графовой структуре отношений инцидентности элементов графовой структуры.

$I$ – множество используемых в графовой структуре **отношений инцидентности** элементов. Все эти отношения инцидентности являются бинарными ориентированными отношениями. Среди этих отношений выделим:
- отношения инцидентности вершин. Примером такого отношения является последовательность символов в строке символов;
- отношения инцидентности, каждая пара которых связывает связующий элемент графовой структуры с элементом (компонентом) того фрагмента, который задается этим связующим элементом. Подчеркнем, что компонентами связующего элемента могут быть элементы графовой структуры любого вида (вершины, связующие элементы, метки, отношения инцидентности). Подчеркнем также, что компоненты одного и того же связующего элемента могут выполнять разные роли в рамках соответствующего фрагмента графовой структуры, который задается связующим элементом. Указанные роли соответствуют разным отношениям инцидентности, входящим во множество $I$;
- отношения инцидентности, каждая пара которых связывает ключевую вершину графовой структуры с тем элементом графовой структуры, который входит в класс элементов, задаваемый этой ключевой вершиной. Подчеркнем, что элементами, инцидентными ключевой вершине, могут быть элементы графовой структуры любого вида;
- отношения инцидентности, каждая пара которых связывает метку графовой структуры с тем элементом графовой структуры, который имеет указанную метку. Подчеркнем, что элементами, инцидентными метке, могут быть элементы графовой структуры любого вида.

Заметим, что связь каждого отношения инцидентности (каждого элемента множества $I$) с соответствующими парами инцидентности и связь каждой пары инцидентности с элементами графовой структуры, соединяемыми этой парой инцидентности, можно условно считать неявно задаваемыми связями инцидентности более низкого уровня.

Заметим также, что каждую графовую структуру $G$ мы будем трактовать как множество всех элементов, входящих в её состав: $G = ( V \cup C \cup K \cup M \cup I )$. Таким образом, в число элементов графовой структуры входят все её вершины (в том числе ключевые), связующие элементы, метки и отношения инцидентности.

Заметим также, что множество связующих элементов графовой структуры можно разбить на:
- множество **связок** (простых связующих элементов);
- множество **подструктур**, каждая из которых задает фрагмент графовой структуры, в состав которого входят такие связующие элементы, которые связывают элементы графовой структуры, входящие в указанный фрагмент.

В свою очередь, по признаку ориентированности множество связок можно разбить на:
- множество **ориентированных связок**, некоторые компоненты которых выполняют в рамках этих связок разные роли;
- множество **неориентированных связок**, все компоненты которых выполняют в рамках этих отношений связок одинаковые роли.

Важным частным случаем ориентированной связки является **кортеж**. Кортеж задает такое подмножество элементов графовой структуры, в котором роли всех элементов пронумерованы. То есть в рамках указанного подмножества имеется элемент, который в этом подмножестве выполняет роль первого элемента (первого компонента), имеется элемент, который в этом подмножестве выполняет роль второго элемента (второго компонента) и т. д.

Если в графовой структуре имеются кортежи, то в число её отношений инцидентности должны входить следующие отношения:
- быть первым компонентом;
- быть вторым компонентом;
- быть третьим компонентом;
- и т. д.

По количеству компонентов множество связок можно разбить на:
- множество **унарных связок** (одноместных, однокомпонентных);
- множество **бинарных связок** (двухместных, двухкомпонентных);
- множество **многокомпонентных связок** (многоместных), имеющих более двух компонентов.

Неориентированные бинарные связки будем называть **ребрами**, ориентированные – **дугами**. Неориентированные многокомпонентные связки

будем называть **гиперребрами**, а ориентированные – **гипердугами**.

Множество связующих элементов графовой структуры можно трактовать как подмножество **шкалы множеств**, заданной над множеством ($V \cup M \cup I$).

Шкала множеств ($H$) над указанным множеством определяется рекурсивно:

(1) $H \supset ( V \cup M \cup I )$

(2) если $hj1, hj2, ..., hjn \in H$

то $\{ hj1, hj2, ..., hjn \} \in H$

/* т. е. любое множество, состоящее из любых элементов шкалы множеств, само также является одним из элементов шкалы множеств */

(3) если $hj1, hj2, ..., hjn \in H;\ ij1, ij2, ..., ije \in I$

то множество $hj = \{ hj1, hj2, ..., hjn \}$, между которым и множеством $\{ ij1, ij2, ..., ije \}$ задано произвольное соответствие, определяющее то или иное распределение ролей между элементами множества $hj$, также является элементом шкалы (т. е. $hj \in H$)

Какие вопросы ассоциируются с рассмотрением графодинамической модели (графодинамической парадигмы) обработки информации:

- Можно ли на основе теории графов построить **универсальную абстрактную модель обработки информации**, которая могла бы конкурировать с абстрактной машиной фон Неймана, лежащей в основе традиционных компьютерных систем;
- Нужно ли это делать, какими преимуществами эта модель обладает по сравнению с абстрактной машиной фон Неймана. Какими преимуществами обладают системы, создаваемые на основе этой модели. Что принципиально нового дает графодинамическая парадигма обработки информации;
- Есть ли к этому предпосылки;
- Можно ли различные модели решения задач (в том числе различные логические исчисления) формально описать в виде графодинамических моделей обработки информации и можно ли обеспечить совместимость (интегрируемость) таких графодинамических моделей.

Интерес к графодинамическим моделям обработки информации имеет достаточно длительную историю. Для подтверждения этого достаточно отметить:

- Предложенное А.Н. Колмогоровым уточнение понятия алгоритм [Колмогоров, 1958];
- Работы школы М.А. Айзермана по графодинамике [Айзерман, 1988];
- Исследования по графовым грамматикам [Петров, 1987];
- Исследования по теории программирования и CASE-технологиям [Касьянов, 2003];
- Разработка параллельных моделей обработки информации [Котов, 1966];
- Предложенные В.Б. Борщевым и М.В. Хомяковым клубные системы и вегетативная машина [Борщев, 1983].

Для разработки графодинамических моделей обработки информации необходимо рассматривать графовую структуру с позиций семиотики и трактовать её как **знаковую структуру** (текст), представляющую собой систему взаимосвязанных знаков. Такая трактовка графовых структур позволяет "вдохнуть" семантику в теорию графов.

Действительно, почему тексты обязательно должны быть линейными (т. е. цепочками символов). Но, как только мы введем понятие **графового языка** (т. е. языка, текстами которого являются в общем случае графовые структуры различной конфигурации), возникают следующие вопросы:

- В чем преимущество графовых языков по сравнению с традиционными линейными языками, текстами которых являются цепочки (строки) символов;
- Можно ли построить **универсальный графовый язык**, обеспечивающий представление информации (знаний) любого семантического вида;
- Можно ли в универсальном графовом языке сделать так, чтобы множество всех меток, используемых во всех графовых структурах, являющихся текстами универсального графового языка, было конечным.

Говоря о графовых языках, следует подчеркнуть то, что графовые структуры, являющиеся текстами таких языков, представляют собой абстрактные математические структуры, не уточняющие (не детализирующие) способ их материального представления (например, способ кодирования в компьютерной памяти, способ графического изображения, ориентированного на человеческое восприятие). То есть графовая структура как абстрактный математический объект и её, например, графическое представление – это принципиально разные вещи. Из этого, в частности, следует, что каждому графовому языку может соответствовать несколько языков, использующих разные способы представления (кодирования, изображения) текстов этого графового языка.

Накопленный опыт развития и применения теории графов и все полученные в ней результаты становятся хорошим математическим фундаментом для разработки различных графовых языков и различных графодинамических моделей обработки информации, а также для создания теории таких языков и моделей. На стыке теории графов и семиотики может появиться очень интересный раздел семиотики – **графовая семиотика**.

## Принцип 3. Параллельные асинхронные графодинамические модели:

В качестве формальной основы проектируемых интеллектуальных систем использовать графодинамические модели специального вида, ориентированные на **параллельную и асинхронную** обработку информации.

Почему акцентируется внимание на **параллельных** графодинамических моделях. Во-первых, потому, что без организации параллельной обработки информации невозможно рассчитывать на необходимую производительность подавляющего числа практически полезных интеллектуальных систем. Во-вторых, потому, что целый ряд исследований [Котов, 1966] показал перспективность создания параллельных моделей обработки информации именно на основе графодинамического подхода.

Почему отдается предпочтение **асинхронному** варианту управления обработкой информации. Потому, что асинхронные модели обработки информации являются более гибкими, их легче интегрировать и наращивать новыми функциональными возможностями.

Графодинамическая модель параллельной асинхронной обработки информации, которую будем также называть **графодинамической параллельной асинхронной машиной**, трактуется нами как абстрактная **многоагентная система**, состоящая из:

- абстрактной **графодинамической памяти**, в которой хранятся обрабатываемые графовые структуры;
- **коллектива агентов**, работающих над общей для них графодинамической памятью и обменивающихся информацией только через эту память (в т. ч. и для координации своих действий).

**Графодинамическая память** носит реконфигурируемый, структурно перестраиваемый характер, поскольку процесс обработки графовых структур в конечном счете сводится к генерации и удалению различных элементов графовых структур, а также к генерации и удалению пар инцидентности между этими элементами. Другими словами, процесс обработки информации в графодинамической памяти сводится не только к изменению состояния элементов памяти, но и к изменению конфигурации связей между ними.

**Агенты**, работающие над общей графодинамической памятью, делятся на три вида:

- внутренние агенты, каждый из которых реагирует на определенного вида ситуации или события в графодинамической памяти и осуществляет изменение состояния графодинамической памяти, соответствующее своему функциональному назначению;
- рецепторные агенты, каждый из которых реагирует на определенные события во внешней среде и осуществляет первичное отражение этих событий в графодинамической памяти;
- эффекторные агенты, каждый из которых реагирует на определенного вида команды, формируемые внутренними агентами в графодинамической памяти, и осуществляет соответствующее изменение материального (физического) состояния интеллектуальной системы, которое определенным образом влияет на изменение её внешней среды.

Агенты могут работать параллельно, если одновременно возникают условия инициирования агентов.

Асинхронность деятельности внутренних агентов заключается в том, что наличие условия инициирования агента ещё не означает начала его работы. То есть время реакции каждого внутреннего агента в известной мере субъективно и достаточно произвольно. В этом смысле указанные агенты обладают:

- свободой выбора момента начала реакции на условие инициирования;
- свободой выбора последовательности обработки условий инициирования, если в текущий момент таких условий возникло несколько.

Для обеспечения эффективного взаимодействия агентов, работающих над общей графодинамической памятью, наряду с представляемой им свободой, необходима разработка таких правил их поведения, которые гарантируют безопасность и производительность каждого из них. В конечном счете, эти правила сводятся к двум положениям:

- позаботься о своей безопасности, точнее, об обеспечении безопасного выполнения своей задачи;
- не навреди другим агентам (помни о том, что ты не один, не создавай для других "аварийных" ситуаций).

Для обеспечения безопасного выполнения своей задачи агент блокирует некоторые элементы графовой структуры, которая хранится в общей графодинамической памяти. Блокировка – это запрет, установленный заданным агентом и адресованный другим агентам, на выполнение тех или иных действий над заданным элементом хранимой графовой структуры. Таким образом, существует несколько видов таких блокировок. Приведем некоторые из них:

- запрет на удаление заданного элемента графовой структуры;
- запрет на удаление всех элементов хранимой графовой структуры, инцидентных заданному (блокируемому) элементу;
- запрет на удаление заданного вида элементов хранимой графовой структуры, которые связаны с блокируемым элементом выходящей из него (или входящей в него) парой инцидентности,

принадлежащей заданному отношению инцидентности;

- запрет на генерацию заданного вида элементов в хранимой графовой структуре, которые связаны с блокируемым элементом выходящей из него (или входящей в него) парой инцидентности, принадлежащей заданному отношению инцидентности.

Приведем некоторые правила поведения агента, работающего в коллективе агентов над общей графодинамической памятью:

- не нарушать блокировочные запреты, сформированные другими агентами;
- самому заблокировать тот фрагмент обрабатываемой графовой структуры, целостность которого необходимо сохранить до завершения своей работы;
- не "жадничать" – не блокировать больше, чем надо;
- снимать свои блокировки как можно быстрее, как только в них отпадает необходимость (т. е. желательно это делать до завершения своей работы);
- удалять сгенерированные для своей работы вспомогательные структуры (информационные "леса") как можно быстрее, как только в них отпадает необходимость (т. е. желательно убирать информационный "мусор" по мере возможности до завершения своей работы);
- поиск фрагментов хранимой графовой структуры, являющихся условиями инициирования агента осуществлять поэтапно, начиная с поиска тех частей этих условий, которые реже появляются в памяти (это необходимо для того, чтобы скорее установить факт отсутствия условий инициирования).

В случае возникновения **конфликтов** между агентами используются внутренние агенты специального вида, реагирующие на возникновение таких конфликтов и обеспечивающие их разрешение (такие агенты будем называть метаагентами-судьями).

Заметим, что для организации своей деятельности над графодинамической памятью каждый агент "опирается" на соответствующее ему семейство постоянно присутствующих в памяти (резидентных) элементов хранимой в памяти графовой структуры. Указанные элемениты будем называть ключевыми элементами агентов. Очевидно, что такие элементы соответствуют константам программ, описывающих поведение агентов.

Сложность комплексного перехода на графодинамическую парадигму параллельной асинхронной обработки информации определяется исключительно психологическими обстоятельствами – это непривычно и, следовательно, боязно. Но накопленный человечеством опыт по созданию компьютерных систем и, в частности, интеллектуальных систем позволяет этот переход сделать достаточно быстро, так как многие проблемы, возникающие при реализации и применении графодинамических моделей, имеют достаточно близкие аналоги в традиционных компьютерных системах, но многие из них могут быть решены значительно проще и элегантнее. Кроме того, в результате перехода к графодинамическим моделям "обнажается" целый ряд проблем, которые ранее просто были не видны. Эпицентром такого перехода является формализация семантики и разработка семантически совместимых языковых средств представления различных видов знаний.

## Принцип 4. Семантические модели представления и обработки знаний:

В качестве формальной основы проектируемых интеллектуальных систем, в качестве основы абстрактных логико-семантических моделей интеллектуальных систем, использовать графодинамические модели специального вида – семантические модели представления и обработки знаний, в основе которых лежат **семантические сети** [Кузнецов, 1986], [Лозовский, 1984], [Плесневич, 1982], [Скороходько, 1989], [Шенк, 1980], [Sowa, 2008].

Фактически, речь идет о создании формальных средств описания семантики различных видов знаний и формальных средств описания обработки знаний на семантическом уровне.

**Семантическая сеть** – это графовая структура $G$, задаваемая пятеркой $< V, C, K, M, I >$ и удовлетворяющая следующим требованиям, которые дополняют свойства множеств $V, C, K, M, I$, указанные в определении графовой структуры:

- каждая вершина $vj \in V$ является знаком одного из объектов, описываемых семантической сетью;
- каждая ключевая вершина $kj \in K$ является знаком соответствующего класса элементов графовой структуры $G$;
- каждая метка $mj \in M$ также является знаком соответствующего класса элементов графовой структуры $G$;
- каждая пара инцидентности, принадлежащая любому отношению инцидентности $ij \in I$, является парой принадлежности, связывающей знак некоторого множества элементов семантической сети с одним из этих элементов;
- в семантической сети вершины $vj, vk \in V$ могут быть инцидентны друг другу, но только в том случае, если по крайней мере одна из них (например, $vk$) является ключевой ($vk \in K$), а вторая является вершиной, принадлежащей множеству, обозначаемому ключевой вершиной $vk$ (т.е. $vj \in vk$);
- каждый элемент множества $I$ ($ij \in I$) является

знаком некоторого подмножества отношения принадлежности, задающего определенную роль, выполняемую соответствующими элементами семантической сети в рамках соответствующих множеств таких элементов. Указанные подмножества отношения принадлежности будем называть **ролевыми отношениями**;

- каждый связующий элемент $cj \in C$ является знаком некоторого фрагмента графовой структуры $G$, а точнее знаком некоторого подмножества множества всех элементов графовой структуры $G$;
- среди элементов графовой структуры $G$ нет пар, синонимичных друг другу знаков, т.е. знаков, обозначающих один и тот же объект (одну и ту же сущность) – либо один и тот же внешний описываемый объект, либо одно и то же множество элементов графовой структуры;
- среди элементов графовой структуры $G$ нет омонимичных знаков, которые в разных контекстах, в разных обстоятельствах могут обозначать разные сущности.

Следовательно, все (!) элементы (атомарные фрагменты) семантической сети являются знаками различных сущностей (объектов). Такими сущностями могут быть всевозможные внешние описываемые объекты, а также различные множества, состоящие их элементов (атомарных фрагментов) этой же семантической сети.

Таким образом, семантическая сеть – это абстрактная знаковая конструкция "рафинированного вида", в которой нет ничего кроме знаков и инцидентности этих знаков. В частности, в семантической сети отсутствуют элементарные незначащие фрагменты (символы), имена описываемых объектов, слова, из которых эти имена состоят, всевозможные разделители и ограничители, обеспечивающие структуризацию текста. В отличие от текстов традиционного вида, семантическая сеть имеет в общем случае нелинейный характер, поскольку каждый элемент семантической сети может быть инцидентен более чем двум другим элементам.

Семантическую сеть можно трактовать как абстрактный текст, который является семантическим инвариантом соответствующего максимального множества семантически эквивалентных текстов, принадлежащих всевозможным языкам.

На основе понятия семантической сети вводится понятие языка семантических сетей в заданном алфавите и с заданным набором ключевых узлов.

Семантические сети как модели представления знаний известны давно. Но, в отличие от фреймовых, продукционных и логических моделей, для семантических сетей не были разработаны достаточно удобные и практически используемые языки представления знаний, достаточно удобные языки программирования, специально ориентированные на обработку семантических сетей. И, как следствие этого, не были созданы широко используемые комплексные технологии проектирования интеллектуальных систем, в основе которых лежат семантические сети. Причин тому много. Одна из них – это не совсем привычный, нетрадиционный характер таких моделей и возникший на этой основе миф о сложности их реализации. Если такие графодинамические семантические модели реализовывать "в лоб" и без ориентации на последующую аппаратную поддержку, то, конечно, это будет неэффективно. Но жизнь берет свое. И на фоне бурного развития микроэлектронных технологий подобного рода мифы выглядят все менее и менее убедительными. Более того, развитие Internet-технологий привело к необходимости формализации семантики информации, обрабатываемой в сети Internet, что вызывало бурное развитие целого направления – Semantic Web.

Чем же хороши семантические сети и в чем достоинство семантических моделей обработки информации:

- Представление знаний в виде семантических сетей позволяет существенно упростить процедуру интеграции знаний и свести эту процедуру к выявлению и склеиванию синонимичных элементов интегрируемых семантических сетей;
- Специфика обработки баз знаний заключается в том, что порождаемые (генерируемые) новые фрагменты знаний необходимо не просто построить, но и погрузить, интегрировать в текущее состояние базы знаний, т.к. в этих порождаемых фрагментах знаний могут появиться знаки, синонимичные тем, которые уже присутствуют в текущем состоянии базы знаний. Таким образом, процедура интеграции порождаемых фрагментов обрабатываемой базы знаний является процедурой, постоянно используемой в ходе обработки знаний. Следовательно, представление знаний в виде семантических сетей, благодаря упрощению процедуры интеграции знаний, позволяет упростить не только ввод новых знаний из вне, но и интеграцию в состав текущего состояния базы знаний новых знаний, порождаемых в ходе решения задач;
- База знаний интеллектуальной системы, представленная в виде корректно построенной семантической сети, полностью исключает дублирование информации в рамках такой базы знаний – каждая информация, представленная соответствующим фрагментом семантической сети, должна находиться в рамках этой семантической сети там и только там, где она должна находиться, и нигде больше;
- Представление знаний в виде семантических сетей позволяет существенно упростить процедуру ассоциативного доступа к различным

видам фрагментов хранимой базы знаний, а также существенно расширить типологию запросов (вопросов) к базе знаний;

- Семантические модели обработки знаний не просто хорошо приспособлены к поддержке параллельной асинхронной обработки информации, но и обеспечивают обмен информацией через общую графодинамическую память между различными параллельно (одновременно) протекающими (выполняемыми) процессами, что может существенно ускорить каждый из этих процессов. Примером такого взаимодействия параллельно протекающих процессов является одновременная реализация разных стратегий и тактик, направленных на поиск пути решения заданной нетривиальной задачи;
- С помощью семантических моделей представления и обработки знаний можно проинтерпретировать все известные виды моделей представления обработки знаний (фреймовые, продукционные, логические), а также все известные модели решения задач различного вида и все известные модели рассуждений. Это дает возможность рассматривать перечисленные модели не как альтернативные, а как дополняющие друг друга модели, которые могут сосуществовать в разных сочетаниях в разных интеллектуальных системах.

Семантическая модель обработки знаний представляет собой абстрактную многоагентную систему, состоящую из **абстрактной семантической памяти** в которой хранятся семантические сети, и из множества агентов, ориентированных на обработку семантических сетей, хранимых в указанной семантической памяти.

Семантическую память можно трактовать как абстрактную семантическую модель памяти интеллектуальной системы.

Семейство абстрактных агентов, работающих над семантической памятью вместе с этой семантической памятью можно трактовать как **семантическую модель решения задач**, используемую в соответствующей интеллектуальной системе, или как операционную семантику этой интеллектуальной системы. Подчеркнем, что семантическую модель обработки информации можно построить для любой компьютерной системы (как для интеллектуальной системы, так и для компьютерной системы традиционного вида), обеспечивая, тем самым, семантическую совместимость (на абстрактном уровне) не только интеллектуальных систем, но и компьютерных систем любого уровня интеллектуальности.

Всю семантическую сеть (максимальную семантическую сеть), хранимую в семантической памяти абстрактной логико-семантической модели интеллектуальной системы, будем называть **абстрактной семантической моделью базы знаний** этой интеллектуальной системы.

**База знаний** должна содержать в себе всю информацию, необходимую агентам, работающим над семантической памятью, для организации коллективной деятельности по решению задач, с которыми должна справляться интеллектуальная система (сюда, в том числе входит и описание блокировок, задаваемых разными процессами в семантической памяти).

Семантическая модель базы знаний интеллектуальной системы – это, образно говоря, формальная трактовка "семантического пространства" в котором "живет" эта интеллектуальная система, а, точнее, такого фрагмента указанного "семантического пространства", который в текущий момент указанной интеллектуальной системе известен.

В целом логико-семантическая модель интеллектуальной системы включает в себя:

- семантическую модель базы знаний этой интеллектуальной системы;
- семантическую машину обработки знаний этой интеллектуальной системы, которая, в свою очередь, состоит из:
  - семантической памяти;
  - коллектива агентов над семантической памятью.

## Принцип 5. Унификация семантического представления знаний:

Определить структуру унифицированных семантических сетей, обеспечивающих представление самых различных видов знаний.

Это предполагает разработку соответствующего стандарта, выделяющего из всего многообразия абстрактных языков семантических сетей определенный базовый универсальный язык семантических сетей, который мы назвали SC-кодом (Semantic Computer code).

Для того, чтобы перейти от семантических сетей произвольного вида к текстам SC-кода уточним направления такого перехода, т.е. задаваемые нами критерии качества разрабатываемого языка семантических сетей (SC-кода). К таким критериям мы отнесем:

(1) Переход от семантических сетей, имеющий унарные и многокомпонентные (многоместные) связки, к семантическим сетям, имеющим только бинарные связки. Такие сети будем называть бинарными семантическими сетями [Плесневич, 2008], [Карабеков и др., 2008];

(2) Минимизация алфавита, т.е. минимизация числа меток, используемых в семантических сетях;

(3) Универсальность разрабатываемого языка, т.е. возможность представления любых знаний в виде текстов этого языка.

Семантическую сеть *Gb* задаваемую пятеркой, *< Vb, Cb, Kb, Mb, Ib >* будем называть **бинарной семантической сетью** в том и только в том случае, если:

- все связующие элементы, входящие во множество *Cb*, являются бинарными связками, каждая из которых имеет только два компонента, которыми могут быть элементы множества *Vb*, элементы множества *Cb* и элементы множества *Kb*;
- *Ib*={*i1, i2*}, где *i1* – бинарное ориентированное отношение инцидентности, связывающее бинарную связку с её компонентом; *i2* – бинарное ориентированное отношение инцидентности, связывающее бинарную ориентированную связку с ее вторым компонентом. Очевидно, что $i1 \subset i2$.

Для любой семантической сети *G,* задаваемой пятеркой *< V, C, K, M, I >,* можно построить семантически эквивалентную ей бинарную семантическую *Gb*, задаваемую пятеркой *< Vb, Cb, Kb, Mb, Ib >* следующим образом:

- во множество *Vb* включаются все вершины (*V*) семантической сети *G* и все небинарные связующие элементы (*Cn*) семантической сети *G* ($Cn \subset G$);
- *Ib* = {*i1, i2*} – это отношения инцидентности связывающие бинарные связки (в том числе из множества *C*) с их компонентами;
- $Mb = ( M \cup \{ re \} )$. Если в алфавит *M* не входила метка отношения принадлежности (*re*), то в алфавит *Mb* она дополнительно вводится;
- $Kb = K \cup (I \setminus ( Ib \cup \{ re \} ) )$. Т.е. знаки отношений инцидентности, связывающих (в рамках *G*) небинарные связующие элементы с их компонентами, становятся в рамках *Gb* ключевыми узлами;
- во множество *Cb* включаются (1) все бинарные связки семантической сети *G*, (2) все ориентированные пары отношения принадлежности (*re*), связывающие ключевые вершины семантической сети *G* с инцидентными им элементами семантической сети, (3) все ориентированные пары всех отношений инцидентности ( *I* ), связывающих небинарные связующие элементы (из множества *C*) с их компонентами, (4) все пары принадлежности, явно связывающие ключевые узлы семантической сети *Gb*, соответствующие различным отношениям инцидентности ( *I* ) семантической сети *G*, с парами принадлежности которые явно связывают небинарные связующие элементы с их компонентами, выполняющими в рамках этих связующих элементов роли, обозначаемые указанными отношениями инцидентности.

Нетрудно заметить, что приведение семантической сети к бинарному виду приводит также к минимизации числа отношений инцидентности.

Нетрудно также заметить, что все пары инцидентности, связывающие небинарные связующие элементы семантической сети *G* с их компонентами, в бинарной семантической сети *Cb* "превращаются" в бинарные ориентированные связки, которые принадлежат отношению принадлежности и которые, следовательно, должны быть помечены меткой *re* ($re \in Mb$), являющейся знаком отношения принадлежности.

Таким образом, каждый небинарный связующий элемент семантической сети *G* в семантической сети *Gb* трактуется как множество связываемых им элементов семантической сети, связь которого с его элементами представляется явно – не в виде пар инцидентности, а в виде дополнительно вводимых связок принадлежности. Эти связки принадлежности, в свою очередь, могут быть вторыми компонентами других связок принадлежности, связывающих указанные связки с ключевыми узлами, обозначающими различные роли компонентов небинарных связующих элементов исходной семантической сети.

Бинарные связки легко изображать графически (в виде линий, каждая из которых соединяет графические изображения двух связываемых элементов семантической сети). Использование только бинарных связок существенно упрощает машинное кодирование семантических сетей и, в частности, упрощает разработку специальной памяти для хранения семантических сетей.

В основе **минимизации алфавита** элементов лежит следующее свойство семантических сетей. Метки, входящие в состав алфавита элементов семантической сети, и ключевые узлы этой семантической сети семантически эквивалентны в том смысле, что просто являются синтаксически различными способами выделения (задания) различных классов элементов семантической сети. При этом заметим, что в отличие от алфавита символов линейного текста, все элементы алфавита (все метки) семантической сети, как и все ее ключевые узлы, имеют семантическую интерпретацию на описываемой предметной области. Таким образом, метки элементов семантической сети без какого-либо изменения семантики этой семантической сети можно "превращать" (преобразовывать) в ее ключевые узлы. При этом семантическая интерпретация каждого такого ключевого узла будет совпадать с семантикой соответствующей преобразованной метки.

Если мы имеем дело с корректно (правильно) построенной семантической сетью и записанной в некотором языке семантических сетей, то перевод этой семантической сети на любой другой язык семантических сетей не требует больших усилий, т.к. имеет место большое сходство синтаксического и семантического устройства всех языков семантических сетей. Фактически эти языки отличаются своими алфавитами, элементы которых, как было показано выше, легко преобразуются в ключевые узлы семантических сетей.

Итак, число меток семантической сети можно уменьшать ценой расширения множества её ключевых узлов. Вопрос в том, до какого предела это можно делать и как выглядит минимальный алфавит универсального языка семантических сетей. Универсальным языком семантических сетей будем называть такое множество семантических сетей, элементами которых являются семантические сети, представляющие любую информацию о любой описываемой предметной области.

Из сказанного ранее следует, это в состав минимального алфавита универсального языка семантических сетей, по крайней мере, должна входить метка *re*, обозначающая ***отношение принадлежности***. Без этой метки невозможно описать связи ключевых узлов семантической сети с элементами обозначаемых ими классов, а также невозможно осуществить переход к бинарным семантическим сетям.

Текст, принадлежащий SC-коду, т.е. ***sc-текст*** (sc-структура, sc-конструкция) является семантической сетью частного вида, имеющей следующие особенности:

- все связки sc-текстов являются бинарными связками, которые будем называть ***sc-коннекторами***. Неориентированные sc-коннекторы будем называть ***sc-ребрами***, ориентированные – ***sc-дугами***.

- множество меток элементов sc-текстов (алфавит sc-элементов, алфавит SC-кода) включает в себя:
  - ***метку sc-узлов*** (вершин sc-текстов);
  - ***метку sc-ребер***;
  - ***метку sc-дуг общего вида***;
  - ***метку sc-дуг принадлежности***;
  - ***метку sc-дуг основного вида***.

- множество отношений инцидентности элементов sc-текстов состоит из двух следующих отношений:
  - *быть* ***компонентом sc-коннектора*** (sc-ребра или sc-дуги);
  - *быть* ***вторым компонентом sc-дуги***;

- множество ключевых узлов sc-кода (ключевых узлов sc-текстов) вместе с метками sc-элементов задает базовую семантическую типологию sc-элементов, т.е. базовую онтологию SC-кода.

К числу ключевых узлов SC-кода, определяющих разбиение множества sc-элементов по признаку **константности**, относятся:

- ключевой sc-узел, обозначающий множество всевозможных константных sc-элементов (***sc-констант***). Каждая sc-константа является обозначением некоторого конкретного фиксированного объекта;

- ключевой sc-узел, обозначающий множество всевозможных переменных sc-элементов (***sc-переменных***), каждая из которых обозначает некоторый произвольный, нефиксированный объект из некоторого дополнительно уточняемого множества объектов. Используются sc-переменные в логических формулах (в т.ч. в высказываниях), в программах (в обобщенных описаниях способов решения различных классов задач), в формулировках вопросов.

К числу ключевых узлов SC-кода, определяющих разбиение множества sc-элементов по **структурному признаку**, относятся:

- ключевой sc-узел, обозначающий множество всевозможных ***sc-коннекторов***, т.е. атомарных связок sc-элементов. Более детальное разбиение множества sc-коннекторов по структурному признаку осуществляется с помощью меток sc-элементов (на ***sc-ребра***, ***sc-дуги общего вида***, ***sc-дуги принадлежности***, ***sc-дуги основного вида***);

- ключевой sc-узел, обозначающий множество всевозможных sc-узлов, каждый из которых обозначает некоторое связующее множество sc-элементов. Указанные sc-элементы будем называть ***связующими sc-узлами***. К числу ключевых узлов SC-кода, определяющих более детальную структурную типологию связующих sc-узлов, относятся:
  - ключевой sc-узел, обозначающий множество всевозможных sc-узлов, каждый из которых обозначает некоторую неатомарную связь между sc-элементами, т.е. связь, не являющуюся sc-коннектором. Такие sc-узлы будем называть ***неатомарными sc-связками***. Более детальная структурная типология неатомарных sc-связок задается такими ключевыми узлами SC-кода, как быть ***унарной sc-связкой***, быть ***бинарной неатомарной sc-связкой***, быть ***многокомпонентной sc-связкой***, быть ***ориентированной неатомарной sc-связкой***, быть ***неориентированной неатомарной sc-связкой***;

  - ключевой sc-узел, обозначающий множество всевозможных sc-узлов, каждый из которых обозначает некоторую структуру из sc-элементов. Такие структуры будем называть обозначениями sc-структур или просто ***sc-структурами***.

- ключевой sc-узел, обозначающий множество всевозможных sc-узлов, каждый из которых обозначает некоторый класс sc-элементов. Такие sc-узлы будем называть обозначениями sc-понятий или, просто, ***sc-понятиями***. Более детальная структурная типология sc-понятий задается следующими ключевыми узлами SC-кода: быть ***отношением*** (классом однотипных связок), быть ***бинарным отношением***, быть ***унарным отношением***, быть ***многоместным отношением***, быть ***ориентированным отношением***, быть ***неориентированным отношением***, быть ***ролевым отношением*** (т.е. отношением, которое является подмножеством отношения принадлежности), быть ***классом структур***, быть ***классом терминальных sc-узлов*** (первичных sc-узлов, которые не являются обозначениями множеств sc-элементов).

- ключевой sc-узел, обозначающий множество всевозможных sc-узлов, каждый из которых обозначает некоторый объект, который не является множеством sc-элементов. Такие sc-узлы будем называть ***терминальными sc-узлами*** (первичными sc-узлами). Более детальная структурная типология терминальных sc-узлов задается следующими ключевыми узлами sc-кода:
  - быть ***предметным sc-узлом***, каждый из которых обозначает некоторый реальный (материальный, физический) или вымышленный внешний объект некоторой предметной области;
  - быть ***sc-ссылкой***, каждая из которых обозначает либо определенный файл, который можно просматривать или в котором закодирована в определенном формате некоторая внешняя, инородная для SC-кода информационная конструкция, либо некоторую компьютерную систему, с которой можно взаимодействовать;
  - быть ***терминальным элементом шкалы или шаблона*** (это sc-элементы, для которых трудно установить обозначаемые ими объекты, поскольку эти sc-узлы просто являются терминальными элементами каких-либо шкал, шаблонов, типовых структур, с которыми устанавливаются соответствия, с которыми сравниваются, сопоставляются различные объекты и структуры).

К числу ключевых узлов SC-кода, уточняющих семантику sc-дуг принадлежности, относятся:

- ключевые узлы SC-кода, определяющие разбиение множества sc-дуг принадлежности по признаку **позитивности**:
  - быть ***sc-дугой позитивной принадлежности***;
  - быть ***sc-дугой негативной принадлежности***;
  - быть ***sc-дугой нечеткой принадлежности*** (т.е. sc-дугой позитивность или негативность которой в текущий момент не установлена);
- ключевые узлы SC-кода, определяющие разбиение множества sc-дуг принадлежности по признаку **стационарности**:
  - быть ***sc-дугой стационарной принадлежности***, семантический тип которой является постоянным, не изменяющимся во времени;
  - быть ***sc-дугой нестационарной принадлежности***, семантический тип которой изменяется во времени:

Заметим, что ***sc-дуги основного вида***, которые выделяются с помощью соответствующей метки, семантически трактуются как ***sc-дуги позитивной стационарной принадлежности***.

Перечислим основные особенности и достоинства SC-кода.

Унифицированные семантические сети (sc-тексты) – это абстрактная семантическая модель знаний, являющаяся инвариантом различных способов представления и кодирования этих же знаний (в том числе и самих семантических сетей). Наличие такого инварианта необходимо для решения проблемы интеграции самых различных видов знаний. На основе унифицированных семантических сетей можно строить семантические модели различных компьютерных систем и решить проблему интеграции таких систем.

SC-код является абстрактным языком в том смысле, что способ изображения (материализации) его текстов не уточняется. Следовательно, можно разрабатывать различные графические уточнения SC-кода (например, SCg-код), различные варианты изображения sc-текстов в виде строк символов (например, SCs-код), различные варианты машинного представления sc-конструкций в адресной памяти традиционных компьютеров, а также в специальной структурно перестраиваемой ассоциативной памяти будущих компьютеров, ориентированных на обработку баз знаний.

Все sc-элементы, кроме терминальных sc-узлов, являются обозначениями множеств, состоящих из sc-элементов (множеств sc-элементов). Такие sc-элементы будем называть ***вторичными sc-элементами***. Из этого следует то, что SC-код имеет базовую теоретико-множественную семантическую интерпретацию.

SC-код представляет собой достаточно простой компьютерный код семантических сетей, который является не "инородным" представлением семантических сетей, а их представлением тоже в виде семантических сетей, но максимально простого вида – с минимальным алфавитом и с бинарными связками.

SC-код ориентирован на представление информации в компьютерной памяти и может рассматриваться как основа модели структурно перестраиваемой ассоциативной памяти будущих компьютеров, ориентированных на обработку семантических сетей. Т.е. SC-код можно рассматривать как универсальную основу машинного кодирования знаний в памяти будущих компьютеров, ориентированных на обработку семантических сетей. В такой памяти биты и байты "уступят место" sc-дугам, sc-ребрам и sc-узлам.

На базе SC-кода можно создавать целое семейство совместимых специализированных языков, ориентированных на представление самых разных видов знаний (логических формул и высказываний, программ, вопросов, поведенческих целей, различных видов моделей динамических систем и т.п.), таким образом, чтобы тексты всех этих специализированных языков полностью соответствовали SC-коду (т.е. были sc-текстами). Такие специализированные языки, общим носителем которых является SC-код, будем называть ***sc-языками***. Каждый sc-язык определяется своим расширением множества ключевых узлов SC-кода.

SC-код представляет собой ядро **универсального открытого языка семантических сетей**, являющегося результатом интеграции всевозможных языков семантических сетей, построенных на основе SC-кода, и задаваемого:

- фиксированным алфавитом (алфавитом SC-кода);
- постоянно расширяемым (открытым) семейством ключевых узлов, в состав которого входят все ключевые узлы всех интегрируемых языков.

SC-код представляет собой **единство языка и метаязыка**. Так, например, в виде sc-конструкций можно описать синтаксис, семантику и онтологию SC-кода. С формальной точки зрения SC-код можно трактовать как метаязык базовой семантической спецификации sc-элементов с помощью специального набора ключевых узлов SC-кода.

Единство языка и метаязыка в SC-коде проявляется в том числе и на самом низком уровне – на уровне sc-дуг принадлежности < *si*, *ei* >, в которых сам sc-элемент *ei*, а не обозначаемый им объект, является элементом множества, обозначаемого sc-узлом *si*.

SC-код позволяет описать структуру любой информационной конструкции, не принадлежащей SC-коду, на любом уровне (на любом этапе синтаксического и семантического анализа). В частности, первичную синтаксическую структуру любой информационной конструкции можно представить в виде изоморфной sc-конструкции. Следовательно, SC-код может быть использован в качестве метаязыка для описания любого внешнего языка, т.е. языка, тексты которого не являются sc-конструкциями.

SC-код хорошо приспособлен к использованию в условиях так называемых **не-факторов** - нестационарности, неточности, противоречивости, неактуальности знаний, а также неполноты знаний (нечеткости, несформированности множеств, несформированности внешних информационных конструкций) [Нариньяни, 1994].

Информационные конструкции SC-кода (sc-конструкции) легко **визуализируются**.

## Принцип 6. Унификация структуризации баз знаний:

Трактовать семантическую структуру **базы знаний** интеллектуальной системы как отражение иерархической системы взаимосвязанных друг с другом **предметных областей**, представляемых в базе знаний. Это предполагает

(1) уточнение понятия предметной области;
(2) разработку языковых средств описания структуры предметных областей с помощью унифицированных семантических сетей;
(3) разработку языковых средств описания типологии предметных областей и различных видов связей между ними.

Структуризация базы знаний, выделение в ней различных связанных между собой подструктур необходимы по целому ряду причин. В частности, это необходимо для дидактических целей (человеку усваивающему некоторые знания, желательно иметь, своего рода оглавление или "карту" этих знаний, что позволяет планировать их усвоение и рассматривать их с различной степенью детализации), а также для организации распределения работ по проектированию баз знаний (когда разным исполнителям поручается разработка разных фрагментов базы знаний, имеющих достаточно четкие границы).

Таким образом, база знаний рассматривается нами как система взаимосвязанных между собой интегрируемых структур, которые будем называть **фрагментами базы знаний**. Связи между фрагментами базы знаний могут быть самыми различными. Каждый фрагмент и вся база знаний в целом может иметь несколько вариантов декомпозиции на подфрагменты (частные фрагменты).

По стурктурно-семантическому принципу можно выделить следующие типы фрагментов баз знаний:

- база фактов некоторой предметной области, которую, сокращенно, будем называть просто **предметной областью** и которая представляет собой результат интеграции всех известных в текущий момент фактографических высказываний, являющихся истинными для указанной предметной области;
- **иерархическая система нескольких предметных областей**, которые нецелесообразно объединять (интегрировать) в одну предметную область, так как в результате этого получается "сборная солянка";
- **семантическая окрестность заданного объекта**;
- **связная семантическая окрестность конечного множества заданных объектов**, представляющая сравнение (сравнительный анализ) и связи объектов из заданного конечного множества, т. е. описывающая сходства (аналогии), отличия заданных объектов, а также "близкие" связи между ними.

При структуризации базы знаний некоторым её фрагментам приписывается статус **разделов базы знаний**, которые именуются, нумеруются и входят в состав (оглавление) базы знаний.

Рассмотрим формальное уточнение понятия **предметной области** с помощью SC-кода. Если в рамках **sc-модели базы знаний** явно вводится некоторая предметная область, то она трактуется как некоторая sc-структура, для которой в базе знаний явно вводится обозначающий её sc-узел, который, в свою очередь, связывается входящей в него sc-дугой основного вида с sc-узлом, обозначающим класс sc-структур, являющихся предметными областями. После этого в указанной sc-структуре необходимо явно задать роли

некоторых узлов, входящих в состав этой sc-структуры. К числу таких ролей относятся:

- быть ***максимальным классом исследуемых объектов***, т. е. множеством всех исследуемых объектов и только их. В каждой предметной области существует только один ключевой узел, выполняющий такую роль;
- быть ***классом исследуемых объектов***. Каждая предметная область может иметь любое число таких классов;
- быть ***классом вторичных объектов,*** построенных на основе исследуемых;
- быть ***классом вспомогательных объектов***, через связи с которыми описываются некоторые характеристики исследуемых объектов;
- быть ***отношением, каждая связка которого связывает только исследуемые объекты*** или вторичные объекты, построенные на основе исследуемых;
- быть ***отношением, каждая связка которого связывает исследуемые объекты со вспомогательными***.

Такое явное указание ролей ключевых элементов предметных областей есть не что иное, как их семантическая спецификация, уточняющая то, какие объекты описываются (исследуются) в данной предметной области, и о каких характеристиках, связях исследуемых объектов в данной предметной области идет речь.

Можно говорить о достаточно богатой типологии предметных областей. В частности, можно выделить следующие классы предметных областей:

- предметная область, описывающая теоретико-множественные характеристики и связи заданного семейства объектов. Такие предметные области, в частности, могут быть онтологиями других предметных областей;
- терминологическая сеть заданного фрагмента базы знаний;
- текст формальной теории, описывающей свойства и закономерности заданной предметной области. Классами объектов исследования такой предметной метаобласти являются: (1) класс логических формул и, в частности, высказываний интерпретируемых на заданной предметной области, (2) класс элементов заданной предметной области, используемых в качестве констант в указанных логических формулах, (3) класс переменных, используемых в указанных логических формулах и возможными значениями которых являются соответствующие элементы заданной предметной области;
- логическая система понятий, описываемых в заданной формальной теории. Эта предметная метаобласть выделяет класс понятий, не определяемых в заданной формальной теории, и связывает каждое определяемое понятие с теми понятиями, на основе которых оно определяется;
- логическая система утверждений заданной формальной теории. Эта предметная мтеаобласть выделяет класс аксиом для заданной формальной теории, каждой теореме ставит в соответствие одно из её доказательств (основное доказательство) и связывает каждую теорему со всеми теми утверждениями и определениями, которые используются в основном доказательстве этой теоремы;
- логическая система фрагментов баз знаний, связывающая каждый фрагмент базы знаний с теми фрагментами, в которых (1) даются определения понятий, используемых в заданном фрагменте и (2) вводятся и доказываются используемые утверждения. На основании такой логической системы строятся различные варианты последовательности изучения (прочтения) разделов баз знаний.

Заметим, что некоторым из перечисленных классов предметных областей может соответствовать одинаковый (унифицированный, фиксируемый) набор используемых в них ключевых понятий. Унификация (стандартизация) таких наборов понятий является важнейшим средством более глубокой семантической совместимости (интегрируемости) различных фрагментов базы знаний. Результатом такой унификации фактически является разработка средств SC-кода, ориентированных на представление предметных областей соответствующего класса. Такие языковые средства будем называть специализированным **sc-языком**.

К числу таких специализированных sc-языков можно отнести:

- **Теоретико-множественный sc-язык**, обеспечивающий описание теоретико-множественных характеристик и связей заданного семейства объектов. С помощью такого языка, в частности, могут быть представлены предметные области, являющиеся теоретико-множественными онтологиями других предметных областей;
- **Терминологический sc-язык**, обеспечивающий построение терминологических сетей;
- **Логический sc-язык,** обеспечивающий построение sc-текстов формальных теорий.

Таким образом, **SC-код** является ядром целого семейства самых различных **sc-языков**, ориентированных на описание различных классов предметных областей, в каждый из которых входят предметные области с разными множествами исследуемых объектов, но с одинаковыми предметами исследования.

Каждому такому специализированному sc-языку ставится в соответствие **множество ключевых узлов**, обозначающих различные классы

исследуемых объектов, различные отношения и алгебраические операции, заданные на множестве исследуемых объектов.

SC-язык, являющийся объединением всевозможных специализированных sc-языков будем называть **языком SCK** (Semantic Code Knowledge). Этот язык рассматривается нами как интегрированный язык представления знаний. Язык SCK является открытым (расширяемым) языком, поскольку его всегда можно пополнить новым sc-языком, описывающим структуры нового вида предметных областей.

Построение **семантической структуры базы знаний** интеллектуальной системы требует не только явного представления спецификаций каждой описываемой предметной области в виде sc-текста, но и явного описания всевозможных связей между этими предметными областями.

Переходя к рассмотрению отношений, заданных на множестве предметных областей, мы фактически переходим к некоторой предметной метаобласти, объектами исследования которой являются всевозможные предметные области (в том числе и сама эта предметная метаобласть).

Обобщая понятия гомоморфизма и изоморфизма алгебраических систем, можно говорить о гомоморфизме и изоморфизме предметных областей, что дает хорошую основу для выявления глубоких нетривиальных аналогий между предметными областями.

Различные предметные области могут пересекаться. То есть элементы одной предметной области могут быть также и элементами другой предметной области. При этом возможны самые различные варианты такого пересечения. Это может быть строгое пересечение, строгое включение. Общие элементы пересекающихся предметных областей могут в рамках этих областей выполнять как одинаковые, так и разные роли. Так, например, первичные элементы одной предметной области могут входить в состав другой предметной области в качестве вторичных элементов, в качестве ключевых элементов. Объекты исследования одной предметной области могут входить в состав другой предметной области и в качестве вспомогательных элементов.

В качестве примеров отношений, заданных на множестве предметных областей рассмотрим несколько вариантов выделения частных предметных областей:

- выделение частной предметной области, на основе выделения подмножества из максимального класса исследуемых объектов. Таким способом из предметной области геометрии, объектами исследования которой являются геометрические точки, фигуры и семейства фигур, можно выделить (1) предметную область планиметрии, изучающую планарные фигуры и планарные семейства фигур (т. е. семейства фигур, лежащих в одной плоскости) и (2) предметную область стереометрии, которая изучает непланарные фигуры и непланарные семейства фигур, которые могут состоять как из непланарных, так и из планарных фигур;
- выделение частной предметной области, на основе выделения подмножества из семейства классов исследуемых объектов. Таким способом из предметной области, изучающей треугольники, можно выделить (1) предметную область, изучающую остроугольные, тупоугольные и прямоугольные треугольники, а также (2) предметную область, изучающую равносторонние, разносторонние и равнобедренные треугольники;
- выделение частной предметной области, на основе выделения подмножества из семейства отношений, заданных на исследуемых объектах. Таким способом из геометрической предметной области можно выделить (1) предметную область, объектами исследования которой являются геометрические фигуры, а предметом исследования – их числовые характеристики и (2) предметную область, объектами исследования которой являются геометрические фигуры, а предметом исследования – различные виды их конгруэнтности (движений).

Предметная область позволяет рассматривать исследуемые объекты на разных уровнях детализации. Детализацию рассмотрения исследуемых объектов можно осуществлять как в рамках исходной (заданной) предметной области, расширяя эту предметную область в соответствующих направлениях, а можно переходить к системе самостоятельных, но связанных между собой предметных областей.

Первым и важнейшим этапом проектирования базы знаний является уточнение структуры описываемой предметной области или нескольких взаимосвязанных предметных областей. Уточнение такой структуры – это, прежде всего, уточнение класса исследуемых объектов, уточнение предмета исследования, уточнение всего семейства ключевых узлов семантической сети, представляющей предметную область. При этом для заданного класса исследуемых объектов и заданного предмета исследования можно построить более качественную и менее качественную предметную область.

Рассмотрим еще один тип фрагментов баз знаний – **семантические окрестности**. В общем случае семантическая окрестность заданного объекта – это описание некоторых, числовых характеристик, свойств и связей заданного объекта. Частными видами семантических окрестностей являются:

- описание характеристик, свойств или связей заданного объекта, однозначно (!) определяющих (устанавливающих) заданный объект (для

понятий – это определение или любое другое высказывание определяющего типа);

- полная (интегрированная) семантическая окрестность заданного объекта, содержащая все известные в текущий момент сведения об этом объекте в рамках заданного раздела базы знаний или в рамках всего текущего состояния базы знаний;
- описание числовых характеристик (параметров, признаков) заданного объекта;
- описание свойств заданного объекта (это те истинные нефактографические высказывания , в которых знак этого объекта используется в качестве константы);
- описание теоретико-множественных связей заданного объекта с другими объектами;
- описание разноязычных терминов, иероглифов, пиктограмм, используемых для внешней идентификации заданного объекта (в т. ч. и описание происхождения этих идентификаторов);
- перечень отношений, соответствующих заданному объекту (отношений, в области определения которых заданный объект входит в качестве элемента; отношений, в области определения которых заданный объект входит в качестве подмножества; отношений, для областей определения которых заданный объект является надмножеством; отношений, область определения каждого из которых строго пересекается с заданным объектом).

В **sc-модели базы знаний** каждая семантическая окрестность представляется в виде соответствующего sc-текста. Для явного введения (задания) этой семантической окрестности в рамках sc-модели базы знаний необходимо:

(1) явно ввести sc-узел, обозначающий эту семантическую окрестность;

(2) явно связать введенный sc-узел sc-дугами основного вида со всеми элементами обозначаемого им sc-текста;

(3) с помощью ролевого отношения ***быть центром семантической окрестности*** явно указать центральный элемент семантической окрестности;

(4) явно связать введенный sc-узел sc-дугой основного вида со специальным sc-узлом, обозначающим ***класс sc-структур, являющихся семантическими окрестностями***.

После этого можно описывать различные характеристики (в частности, типологию) явно введенной семантической окрестности, а также различные связи этой семантической окрестности с другими фрагментами базы знаний.

Технология проектирования унифицированных семантических моделей баз знаний рассмотрена в работе [Ивашенко, 2012].

## Принцип 7. Графовые языки программирования:

Для описания способов решения задач и поведения агентов над общей графодинамической памятью использовать **графовые языки программирования**, которые ориентированы на обработку унифицированных семантических сетей и программы которых сами являются унифицированными семантическими сетями.

Если все используемые в интеллектуальной системе графовые языки программирования привести к общему унифицированному стандарту – к SC-коду (это требует представления в виде sc-текстов не только самих программ, но и обрабатываемых ими данных), то можно достаточно эффективно решать проблему формализации семантической совместимости программ, написанных не только на одном, но и на разных языках программирования.

В традиционных языках программирования синтаксическая структура и семантика хранящихся в памяти обрабатываемых данных отдается на откуп программисту, в результате чего это делается по принципу "кто во что горазд". О какой же семантической совместимости программ после этого можно говорить. В этом смысле традиционные языки программирования "хромают на одну ногу".

Наряду с применением в интеллектуальных системах множества sc-языков самого различного назначения, востребованным является использование целого семейства совместимых sc-языков программирования, которые могут иметь разный уровень, могут быть последовательными, процедурными и декларативными.

Важнейшей особенностью всех этих языков является использование ассоциативного доступа к обрабатываемым фрагментам хранимого в графодинамической памяти sc-текста.

Операционная семантика каждого такого графового языка программирования (точнее, sc-языка программирования) задается коллективом агентов над общей графодинамической памятью, которые обеспечивают интерпретацию любой программы указанного языка программирования, хранящейся вместе с обрабатываемой информацией в указанной графодинамической памяти.

Программы представленные в виде семантической сети и описывающие обработку семантических сетей, а также соответствующие им языки программирования фактически открывают новую страницу в теории программирования, которую можно назвать семантической теорией программ и языков программирования.

Основным лейтмотивом такой теории должно

быть обеспечение семантической совместимости программ и языков программирования.

## Принцип 8. Унификация формального описания агентов, работающих над семантической памятью:

Из всех используемых в интеллектуальной системе графовых языков программирования (а, точнее, из всех sc-языков программирования) выделить **базовый графовый язык программирования**, ориентированный на описание агентов, работающих над общей графодинамической памятью, в которой хранятся и обрабатываются унифицированные семантические сети.

Выделение базового sc-языка программирования предназначено для унификации формального описания поведения агентов, работающих над общей графодинамической памятью.

Такой базовый sc-язык программирования будем называть ***языком SCP*** (Semantic Code Programming), а написанные на нем программы будем называть ***scp-программами***.

Перечислим основные особенности языка SCP:

- язык SCP относится к классу **графовых языков программирования**;
- язык SCP ориентирован на обработку **унифицированных семантических сетей** (sc-текстов), хранимых в семантической памяти;
- программы языка SCP представляются также в виде унифицированных семантических сетей (sc-текстов), т. е. язык SCP принадлежит классу sc-языков;
- язык SCP ориентирован на описание **параллельной асинхронной обработки** sc-текстов, хранимых в семантической памяти;
- язык SCP использует **ассоциативный доступ** к фрагментам обрабатываемых sc-текстов;
- язык SCP является процедурным языком программирования низкого уровня, предназначенным для описания поведения агентов, работающих над семантической памятью;
- уникальной особенностью языка SCP является то, что на нем можно писать **реконфигурируемые программы**, т. е. программы, которые в процессе своего выполнения могут изменять сами себя (удалять или порождать операторы, корректировать порядок их выполнения и т. п.). Такая особенность языка SCP обусловлена не только тем, что scp-программы и обрабатываемые ими данные хранятся в общей памяти, но и тем, что они принадлежат одному и тому же, базовому языку (SC-коду), имеющему четко заданную семантическую интерпретацию.

Более подробно графовый язык программирования SCP и технология проектирования scp-программ рассмотрены в работах [Голенков и др., 2001], [Гулякина и др., 2012].

## Принцип 9. Унификация семантических моделей обработки знаний:

На основе унифицированных семантических сетей (sc-текстов) уточнить понятие **унифицированной модели обработки информации**, а также понятие унифицированной модели решения задач.

Все указанные абстрактные модели будем называть **sc-моделями обработки знаний** или **sc-машинами**, поскольку в основе их лежит использование SC-кода. Каждая такая модель (sc-машина) представляет собой многоагентную систему, состоящую:

(1) из графодинамической памяти, в которой хранятся и обрабатываются тексты SC-кода – такую память будем называть **sc-памятью**;
(2) из коллектива агентов, работающих над общей для них sc-памятью и взаимодействующих между собой только через эту память – такие агенты будем называть **sc-агентами**.

Очевидно, что sc-модели обработки знаний являются частным, унифицированным видом графодинамических моделей параллельной асинхронной обработки информации.

Каждый sc-агент реагирует на соответствующий ему класс ситуаций и/или событий, происходящих в sc-памяти, и осуществляет определенное преобразование sc-текста, находящегося в семантической окрестности обрабатываемой ситуации и/или события. Типология sc-агентов достаточно богата. В частности, можно выделить следующие классы sc-агентов:

- sc-агенты, обеспечивающие интерпретацию программ различных sc-языков программирования высокого уровня;
- sc-агенты информационного поиска;
- sc-агенты, обеспечивающие реализацию правил логического вывода, соответствующих самым различным логическим исчислениям;
- sc-агенты сведения задач к подзадачам;
- sc-агенты анализа качества хранимой базы знаний, в частности, ее корректности, полноты;
- sc-агенты обнаружения и автоматического склеивания синонимичных sc-элементов;
- sc-агенты автоматического устранения некоторых ошибок в базе знаний;
- sc-агенты удаления информационного мусора (в частности, удаления фрагментов базы знаний, которые редко востребованы и могут быть достаточно легко восстановлены в случае их отсутствия);
- sc-агенты, обеспечивающие трансляцию вводимой информации с различных внешних языков в SC-код;
- sc-агенты, обеспечивающие трансляцию sc-текстов, вводимых пользователю на различные внешние языки;
- рецепторные sc-агенты;
- эффекторные sc-агенты.

В понятии sc-машины набор агентов не задается, т.е. могут существовать разные sc-машины с разным набором sc-агентов. Несколько разных sc-машин можно **интегрировать**. С формальной точки зрения это сделать не очень сложно:

(1) интегрировать sc-текст, описывающий текущее состояние взаимодействия sc-агентов одной sc-машины, с аналогичным sc-текстом другой sc-машины;
(2) полученный интегрированный sc-текст поместить в sc-память интегрированной sc-машины;
(3) в интегрированную sc-машину включить все sc-агенты первой интегрируемой sc-машины и все sc-агенты второй интегрируемой sc-машины.

Более того, одна sc-машина может **интерпретировать** другую. Т. е. при интерпретации sc-машин можно не выходить за пределы класса sc-машин. Для этого необходима разработка целого семейства sc-языков программирования различного уровня. Тексты (программы) всех этих языков должны храниться в sc-памяти, т.е. должны быть семантическими сетями, представленными в SC-коде. Операционная семантика (интерпретация) каждого из этих языков задается определенным набором sc-агентов, процедура выполнения (поведения) каждой из которых описывается программой, написанной на языке более низкого уровня.

В абстрактных sc-машинах можно выделить следующие языки программирования:

- Семейство sc-языков программирования высокого и сверхвысокого уровня (как процедурных, так и непроцедурных). Тексты программ этих языков хранятся в базе знаний интеллектуальной системы и описывают способы решения различных классов задач в соответствующих предметных областях.
- Базовый sc-язык программирования (язык SCP), на котором описываются sc-агенты и интерпретации sc-языков программирования высокого и сверхвысокого уровня, а также sc-операции, обеспечивающие интерпретацию различных логических исчислений, различных моделей интеллектуального решения задач.
- SC-язык программирования, на котором описываются sc-агенты интерпретации базового sc-языка программирования. Фактически, это 1-й язык микропрограмм для **sc-компьютера**, обеспечивающего аппаратную интерпретацию базового sc-языка программирования (языка SCP).
- При необходимости можно ввести 2-й язык микропрограммирования, описывающий интерпретацию 1-го и т.д.

На основе понятия абстрактной sc-машины можно уточнить понятие унифицированной логико-семантической модели интеллектуальной системы. Такую унифицированную модель интеллектуальной системы будем называть абстрактной **sc-моделью интеллектуальной системы** или, сокращенно, **sc-системой**. Абстрактная sc-система включает в себя:

- интегрированную базу всех (!) знаний, которые необходимы для функционирования интеллектуальной системы и которые представлены в виде интегрированного sc-текста (такую семантическую модель базы знаний будем называть **sc-моделью базы знаний** или sc-текстом базы знаний);
- абстрактную **sc-машину**, в памяти которой хранится указанный sc-текст базы знаний.

Нетрудно заметить, что sc-текст базы знаний интеллектуальной системы является формальным и унифицированным уточнением того, что должна знать проектируемая интеллектуальная система, а sc-машина интеллектуальной системы и, в первую очередь, набор входящих в ее состав sc-агентов является формальным и унифицированным уточнением того, что должна проектируемая интеллектуальная система уметь делать со своими знаниями.

Подчеркнем также, что четкое выделение абстрактного семантического уровня интеллектуальной системы позволяет не только обеспечивать их семантическую совместимость интеллектуальных систем, но и сформировать критерии сравнения интеллектуальных систем по уровню их возможностей. Очевидно, что уровень возможностей интеллектуальной системы определяется качеством (корректностью, полнотой, многообразием) ее знаний и эффективностью ее умений (т.е. эффективностью используемых ее моделей решения задач).

Заметим также, что абстрактную логико-семантическую модель в принципе можно построить для любой компьютерной системы (как для интеллектуальной системы, так и для компьютерной системы традиционного вида), обеспечивая их семантическую совместимость на абстрактном логико-семантическом уровне.

## Принцип 10. Унификация семантических моделей информационного поиска:

На основе унифицированных семантических сетей (т.е. на основе SC-кода) обеспечить построение **унифицированных семантических моделей информационного поиска** (унифицированных семантических моделей ассоциативного доступа к информации, хранимой в семантической памяти).

Ассоциативный доступ – это доступ, основанный не (!) на знании того, где находится искомая (требуемая) информация (в частности, на знании адреса или имени соответствующей области памяти), а на знании того, как искомая информация связана с известной информацией, хранимой в памяти, т.е. на знании некоторой спецификации искомой информации.

Эффективность организации информационного поиска в базе знаний интеллектуальной системы во многом определяет эффективность самой интеллектуальной системы. Это обусловлено тем, что время, затрачиваемое интеллектуальной системой на поиск нужных в текущий момент знаний и навыков, занимает, мягко говоря, не меньше половины времени затрачиваемого на решение задачи в целом.

Унифицированная семантическая модель информационного поиска, которую будем называть **sc-моделью информационного поиска**, включает в себя:

(1) **SC-язык вопросов**, с помощью которого в виде sc-текстов осуществляется описание (спецификация) запрашиваемых (искомых) фрагментов всего интегрированного sc-текста, хранимого в текущий момент в sc-памяти (т.е. sc-текста, который является sc-моделью базы знаний). Тексты, принадлежащие SC-языку вопросов, будем называть ***sc-вопросами***.

(2) **SC-язык оформления ответов**, с помощью которого осуществляется явное выделение sc-текстов, являющихся ответами, и явное описание их связи с явно выделенными sc-текстами, которые представляют вопросы, соответствующий указанным ответам.

(3) Семейство информационно-поисковых sc-агентов, каждый из которых реагирует на соответствующий ему тип sc-вопроса (который при этом должен быть инициирован) и выполняет соответствующую поисковую процедуру в sc-памяти.

Семантическая типология вопросов является предметом отдельного рассмотрения. Приведем фрагмент такой типологии, чтобы проиллюстрировать семантическую мощность sc-языка вопросов.

Прежде всего, по аналогии с логическими формулами множество вопросов разбивается на:

- ***атомарные вопросы***;
- ***неатомарные вопросы***, каждый из которых представляет собой конечное множество вопросов.

Компонентами неатомарного вопроса могут быть как атомарные, так и неатомарные вопросы. При этом, если построить орграф, вершинами которого будут знаки всех вопросов, входящих в состав заданного неатомарного вопроса, а дуги которого будут связывать знаки неатомарных вопросов, входящих в состав заданного неатомарного вопроса, с их компонентами, то этот орграф будет деревом, все конечные вершины которого являются знаками атомарных вопросов. Частным видом неатомарного вопроса является конъюнктивный вопрос, ответом на который является конъюнкция (интеграция) ответов на все вопросы, являющиеся компонентами этого конъюнктивного вопроса.

Поскольку в общем случае вопросу может соответствовать несколько правильных ответов (т.е. ответов, удовлетворяющих, релевантных, соответствующих заданному вопросу), множество вопросов разбивается на:

- вопросы, запрашивающие все правильные ответы;
- вопросы, запрашивающие один (или по крайней мере один) правильный ответ;
- вопросы, запрашивающие несколько разнообразных правильных ответов;
- вопросы, запрашивающие точно указанное число (большее единицы) правильных ответов.

Специальным видом неатомарных вопросов являются ***сколько-вопросы***, запрашивающие не сами правильные ответы некоторого вопроса (который может быть как атомарным, так и неатомарным, и который является единственным компонентом сколько-вопроса), а количество таких правильных ответов.

Приведем некоторые типы атомарных вопросов:

- ***какой-вопрос атомарного вида***. Каждый такой вопрос запрашивает фрагменты базы знаний, изоморфные заданному **образцу**, который может иметь произвольный размер, произвольную конфигурацию и который может быть представлен не только логической формулой существования в которой квантор существования действует на конъюнкцию атомарных логических формул, но также и логической формулой существования, в которой квантор существования действует на логическую формулу произвольного вида. Суть атомарного *какой-вопроса* заключается в поиске знаков таких объектов, которые заданным образом связаны с другими известными и неизвестными (искомыми) объектами, т.е. в поиске знаков таких объектов, которые удовлетворяют заданным требованиям. На основе *какой-вопросов* атомарного вида строится важный класс неатомарных вопросов ключевыми компонентами которых являются атомарные *какой-вопросы*, а остальными компонентами – вопросы любого вида, в формулировках которых используются переменные, входящие в состав соответствующих ключевых *какой-вопросов*;

- ***запрос всех элементов*** заданного конечного множества (чаще всего – это множество из элементов некоторой структуры);

- ***запрос внешней информационной конструкции***, представленной некоторым файлом в том или ином формате;

- запрос полного текста заданного высказывания;

- ***ли-вопрос***, запрашивающий факт истинности или ложности заданного высказывания в рамках заданной формальной теории;

- ***вопрос выбора альтернатив***, запрашивающий одно или несколько истинных высказываний из заданного множества высказываний;
- ***почему-вопрос***, запрашивающий обоснование (доказательство) истинности заданного высказывания;
- ***что-это-вопрос***, запрашивающий основные сведения об указываемом объекте. Фактически, речь идет о выделении из базы знаний семантической окрестности, "центром" которой является знак указываемого объекта. Таким объектом может быть все, что угодно – понятие, предметная область, формальная теория, высказывание, любая структура, материальный объект.
- ***запрос общих свойств*** объектов, принадлежащих заданному классу;
- ***запрос идентифицирующих признаков*** заданного объекта. Здесь запрашиваются фрагменты базы знаний, каждый из которых однозначно (!) определяет (устанавливает, идентифицирует) заданный объект. Если заданным объектом является понятие, то таким идентифицирующим признаком является либо определение этого понятия, либо соответствующая теорема о необходимости и достаточности;
- ***запрос связей между заданными объектами***;
- ***запрос сравнительного анализа заданных объектов***;
- ***запрос сходств*** заданных объектов (сходства, аналогии – это частный вид связей между объектами);
- ***запрос отличий*** заданных объектов (отличия объектов – это тоже частный вид связей между ними);
- ***запрос плана решения*** заданной конкретной задачи, т.е. плана достижения заданной цели в заданных конкретных условиях;
- ***запрос обобщенного способа решения*** любой задачи из заданного класса задач. Таким обобщенным способом может быть алгоритм, декларативная (непроцедурная) программа, нестрогое предписание (рекомендация);
- ***зачем-вопрос***, запрашивающий то, какой надцели соответствует заданная цель, которая может быть сформулирована как в декларативной, так и в процедурной форме.

Список типов атомарных вопросов можно продолжить, но почти все они будут подтипами (подмножествами) перечисленных типов вопросов. В основе sc-языка вопросов лежит построение онтологии вопросов, в рамках которой четко прописываются все теоретико-множественные (и, в первую очередь, родо-видовые) связи между всеми выделенными типами и подтипами вопросов. При этом в формулировке каждого конкретного sc-вопроса явным образом отражаются иерархия всех типов вопросов, которым принадлежит данный конкретный sc-вопрос. Для этого каждому типу вопросов ставится в соответствие ключевой sc-узел, обозначающий этот тип вопросов.

В заключение заметим, что в **SC-языке оформления ответов** кроме отношения релевантности, связывающего вопросы с правильными на него ответами, используются языковые средства, описывающие качество, полноту ответов. Это вызвано тем, что некоторые типы вопросов предполагают наличие целого множества правильных ответов, но разного качества, с разной степенью полноты.

## Принцип 11. Унификация семантических моделей интеграции знаний и семантических моделей интеграции целых интеллектуальных систем:

На основе унифицированных семантических сетей обеспечить построение **унифицированных семантических моделей интеграции знаний** (понимания знаний) и использовать эти модели (1) как основу процесса приобретения интеллектуальной системой новых знаний как со стороны конечных пользователей, так и со стороны разработчиков; (2) как основу интеграции программ и различных семантических моделей расширения задач; (3) как основу интеграции абстрактных логико-семантических моделей интеллектуальных систем.

Главное свойство интеллектуальной системы – не те интеллектуальные знания и навыки, не те интеллектуальные способности, которые она имеет в текущий момент, а метаспособность приобретать любые необходимые ей новые знания и навыки. А для этого интеллектуальная система, как минимум, должна уметь интегрировать эти приобретаемые знания и навыки. Следовательно, проблема формализации интеграции знаний и навыков является центральной для деятельности интеллектуальных систем.

Принципиальное свойство интеграции двух интеллектуальных систем заключается в следующем. Пусть имеется две интеллектуальные системы *s1*, *s2*, первая из которых способна решать задачи из множества *q1*, в вторая – *q2*. В результате простого соединения этих систем мы получаем систему, которая способна решать задачи из множества ($q1 \cup q2$). Тогда как в результате интеграции мы получаем систему, которая способна решать значительно большее число задач, чем ($q1 \cup q2$). Такое расширение числа решаемых задач происходит за счет тех задач, для решения которых некоторые (но не все) знания и навыки находятся в системе *s1*, а другие – в системе *s2*.

Таким образом, при интеграции интеллектуальных систем происходит приобретение

нового качества "на стыке" интегрируемых систем, когда для решения некоторых задач одна часть необходимых знаний и/или умений находится в одной интегрируемой системе, а другая часть – в другой системе.

Процесс интеграции двух семантических сетей рассмотрим как систему следующих взаимодействующих подпроцессов, некоторые из которых могут выполняться параллельно:

- приведение интегрируемых семантических сетей к унифицированному виду, т.е. представление (запись) их в SC-коде;
- согласование ключевых узлов и онтологий, используемых в интегрируемых sc-текстах. Очевидно, что полностью автоматизировать такое согласование невозможно, поэтому разработчикам интегрируемых фрагментов баз знаний и целых баз знаний необходимо уметь договариваться друг с другом;
- выделение в интегрируемых sc-текстах таких sc-элементов, которые имеют глобальные (уникальные) идентификаторы (внешние имена);
- выделение в интегрируемых sc-текстах sc-элементов, имеющих локальные идентификаторы вместе с областью действии каждого такого идентификатора. Область действия локального идентификатора – это такой фрагмент базы знаний, в рамках которого разные sc-элементы, имеющие этот локальный идентификатор, считаются синонимичными;
- склеивание sc-элементов, имеющих одинаковые глобальные идентификаторы;
- склеивание sc-элементов, имеющих одинаковые локальные идентификаторы, если каждый их этих sc-элементов принадлежит области действия своего локального идентификатора и области действия локального идентификатора другого sc-элемента;
- склеивание sc-элементов на основании однозначности используемых алгебраических операций;
- склеивание sc-элементов на основании логических высказываний о существовании единственности;
- склеивание кратных связок, принадлежащих отношениям:
  - не имеющим кратных связок;
  - имеющих кратные связки, но не для заданных типов компонентов (например, кратные связки принадлежности не могут выходить из знаков канторовских множеств).

Таким образом, интеграция семантических сетей, т.е. процесс погружения (понимания) одной семантической сети в другую – это нетривиальный процесс рассуждений, направленный на выявление пар синонимичных элементов семантической сети на основе определенных знаний, имеющихся в базе знаний интеллектуальной системы.

От унифицированной семантической модели интеграции знаний (точнее, sc-текстов) можно достаточно легко перейти к интеграции sc-моделей интеллектуальных систем, поскольку после интеграции sc-моделей баз знаний интегрируемых интеллектуальных систем интеграция соответствующих им наборов sc-агентов сводится к простому теоретико–множественному объединению указанных множеств sc-агентов.

## Принцип 12. Унификация и интеграция различных семантических моделей решения задач:

Обеспечить в рамках проектируемой интеллектуальной системы использование не только самых различных видов знаний, но и самых **различных моделей и стратегий решения задач**.

Для этого необходимо акцентировать внимание не столько на разработку новых моделей решения задач, сколько на унификацию и интеграцию в рамках проектируемых интеллектуальных систем уже разработанных и хорошо зарекомендовавших себя моделей (дедуктивных, индуктивных, абдуктивных, четких, нечетких, универсальных, специализированных…). Подчеркнем то, что в разных проектируемых интеллектуальных системах могут быть востребованы самые разные сочетания известных моделей и стратегий решения задач. Подавляющее число моделей представления знаний и решения задач не являются альтернативными и дополняют друг друга. Не составляют исключение и такие классы моделей, как фреймовые, логические, продукционные.

Рассмотренное выше понятие **вопроса** и его формализация является основой не только для информационно-поисковых моделей, но и для самых различных моделей решения задач. С точки зрения решателя задач вопрос – это **непроцедурная формулировка информационной цели**, т.е. декларативная формулировка некоторой информационной цели, которая описывает спецификацию (свойства) той информации, которую требуется либо найти, если она уже присутствует в текущем состоянии базы знаний, либо построить (сгенерировать, вывести), если она отсутствует в текущем состоянии памяти. Таким образом, вопрос можно считать описанием целевого (требуемого) состояния обрабатываемой базы знаний (а, точнее, определенного фрагмента этой базы знаний). Вопрос также можно считать одним из видов метазнаний, описывающих (специфицирующих) наше незнание, т.е. наше знание о том, что мы не знаем, но хотели бы знать.

Вопросы могут инициироваться (задаваться) как пользователями, так и самой системой. Это означает, что в процессе обработки информации интеллектуальная система сама себе может задавать (генерировать, порождать) вопросы. Если

инициирован некоторый sc-вопрос, то сначала активизируются соответствующие агенты информационного поиска в "надежде" на то, что запрашиваемый ответ (или ответы) на указанный sc-вопрос уже присутствует в текущем состоянии базы знаний. И только после того, как информационно-поисковые sc-агенты обнаружат отсутствие ответа в текущем состоянии базы знаний, начинается работа решателя задач, направленная на генерацию (построение, порождение, вывод) требуемого ответа.

Кроме вопроса используется также и **процедурная формулировка информационной цели** – это описание (спецификация) некоторого действия, которое требуется выполнить и которое направлено на преобразование (изменение состояния) базы знаний, хранимой в некоторой памяти. Указанное действие, выполняется либо одним sc-агентом (в случае, если это элементарное действие над sc-памятью), либо несколькими sc-агентами и порождает определенное событие (изменение состояния sc-памяти).

Для унификации различных моделей решения задач необходимо уточнить не только понятие **информационной цели**, но и понятие **информационной задачи**. Информационная задача задается (1) формулировкой информационной цели, т.е. описанием того, что требуется, и (2) той хранимой в памяти информацией, которая семантически связана с заданной информационной целью, является контекстом этой информационной цели, т.е. тем, что дано. В пределе, контекстом информационной цели можно считать текущее состояние всей хранимой базы знаний.

Формальное рассмотрение контекстов различных информационных задач требует разработки специальных языковых средств, предназначенных для описания текущего состояния хранимой базы знаний, а, точнее, для описания "грани" между тем, что в текущем состоянии базы знаний известно и тем, что неизвестно. К числу таких языковых средств, в частности, относятся следующие ключевые узлы, являющиеся знаками нестационарных множеств (т.е. множеств, которые в разные моменты времени могут иметь разные элементы):

- быть sc-дугой нечеткой принадлежности (такая sc-дуга связывает sc-узел, обозначающий некоторое множество, с sc-элементом о котором нам в текущий момент времени не известно, принадлежит он указанному множеству или нет);
- быть построенным конечным множеством (у каждого такого множества в текущем состоянии базы знаний известны и явно указаны все его элементы);
- быть построенным высказыванием (для каждого такого высказывания в текущем состоянии базы знаний представлен не только его знак, но и полный текст);
- быть построенной внешней информационной конструкцией (файлом);
- быть аксиоматизированной формальной теорией;
- быть построенным рассуждением (обоснованием, доказательством, решением);
- быть построенной программой;

Более подробно унифицированные семантические модели решения задач и технология их проектирования рассмотрены в работе [Заливако и др., 2012].

## Принцип 13. Унификация визуализации семантических сетей:

В качестве основы организации графического пользовательского интерфейса использовать язык унифицированного визуального представления абстрактных унифицированных семантических сетей в виде, близком к изоморфному.

Указанный язык графического изображения sc-текстов назван SCg-кодом (Semantic Code graphical). Подчеркнем, что следует четко отличать язык абстрактных унифицированных семантических сетей (SC-код), который абстрагируется от того, как должны быть физически представлены узлы и коннекторы текстов этого языка (sc-текстов), и язык графического изображения таких семантических сетей. Т.е. абстрактная семантическая сеть и ее рисунок – принципиально разные вещи.

С помощью SCg-кода осуществляется отображение на экране не только пользовательских сообщений, адресуемых системе, и не только сообщений, адресуемых пользователю, но и всей остальной информации, необходимой для организации работы пользователя (прежде всего – это элементы управления интерфейсом). Такая унификация отображаемой пользователю информации дает возможность организовать взаимодействие пользователя с help-системой точно так же, как и его взаимодействие с основной (предметной) системой.

Трактовка элементов управления пользовательским интерфейсом как элементов отображаемого на экране SCg-текста позволяет:

(1) унифицировать представлений любой информации, отображаемой на экране;

(2) унифицировать способы инициирования различных вопросов, касающихся любой отображаемой на экране информации (в том числе, и элементов управления);

Для того, чтобы четко отделить те средства SCg-кода, которые обусловлены самим SC-кодом, от тех средств, которые обусловлены стремлением повысить наглядность SCg-текстов, введем ядро SCg-кода (или, просто, SCg-ядро), алфавит которого взаимно однозначно соответствует алфавиту SC-кода и, соответственно этому, тексты которого изоморфны (!) семантически эквивалентным текстам SC-кода.

Переход от SCg-ядра к SCg-коду заключается в ослаблении требований, предъявляемых к изображениям семантических сетей, в целях обеспечения удобства для человеческого восприятия. Такое ослабление осуществляется в следующих направлениях: вводится приписывание идентификаторов изображаемых sc-элементов, расширяется алфавит графических примитивов, допускается уникальное изображение некоторых sc-узлов, допускается синонимия sc.g-элементов, но при этом синонимичным элементам должны быть приписаны одинаковые идентификаторы, вводятся специальные графические средства, направленные на повышение наглядности (шинные линии, контуры).

Заметим также, что кроме SCg-кода для внешнего представления абстрактных унифицированных семантических сетей используются также и другие языки:

- SCs-код, обеспечивающий представление унифицированных абстрактных семантических сетей (sc-текстов) в виде, близком к традиционным текстам;
- SCn-код, обеспечивающий гипертекстовое представление абстрактных sc-текстов, предназначенное для оформления исходных текстов баз знаний.

Более подробно различные языки внешнего представления абстрактных sc-текстов вместе с большим количеством примеров рассмотрены в работах [Голенков и др., 2001].

## Принцип 14. Унификация семантических моделей различных пользовательских интерфейсов:

**Пользовательский интерфейс** интеллектуальной системы, построенной на основе предлагаемой технологии, рассматривать как **специализированную интеллектуальную систему**, построенную по той же технологии и предназначенную для **трансляции адресуемых пользователю сообщений** с внутреннего абстрактного семантического языка представления знаний (SC-кода) на тот или иной внешний язык, тексты которого отображаются пользователю в удобном для него виде, а также для **трансляции пользовательских сообщений** с внешнего языка внутренний семантический язык интеллектуальной системы (т.е. в SC-код).

Трактовка пользовательских интерфейсов как интеллектуальных систем и унификация семантических моделей таких систем дает возможность:

(1) унифицировать проектирование пользовательских интерфейсов;

(2) легко наращивать возможности пользовательских интерфейсов;

(3) легко интегрировать пользовательские интерфейсы с предметными (основными) интеллектуальными системами;

(4) неограниченно использовать базу знаний предметных интеллектуальных систем для семантического анализа и понимания вводимой пользователем информации (в частности, естественно-языковых текстов).

Более подробно унифицированные семантические модели пользовательских интерфейсов и технология их проектирования рассмотрены в работе [Корончик, 2012].

## Принцип 15. Библиотека типовых семантически совместимых компонентов интеллектуальных систем и методика модульного проектирования интеллектуальных систем:

В целях ускорения процесса проектирования интеллектуальных систем создать общую библиотеку многократно используемых семантически совместимых компонентов интеллектуальных систем, на основе которой разработать методику модульного (компонентного, сборочного) проектирования интеллектуальных систем.

В указанной библиотеке можно выделить следующие разделы (частные библиотеки):

- библиотека многократно используемых компонентов баз знаний. Прежде всего, в эту библиотеку входят самые различные по содержанию, но семантически совместимые онтологии. Кроме того, сюда относятся различные "джентльменские наборы" знаний, которыми должны владеть "образованные" интеллектуальные системы. К таким знаниям, в частности, относятся базовые знания по арифметике, базовые знаний по теории множеств (каждая интеллектуальная система должна, по крайней мере, отличать элемент заданного множества от его подмножества), базовые знания по теории отношений (каждая интеллектуальная система должна уметь отличать бинарное отношение от многоместного отношения, должна понимать, что такое соответствие), базовые знания по логике (каждая интеллектуальная система должна понимать, что такое теория, высказывание, определение, переменная должна отличать фактографическое высказывание от высказывания, не являющегося фактографическим, должна уметь отличать высказывание от логической формулы, не являющейся высказыванием) и многие другие знания, востребованность которых может быть самой разной;
- библиотека компонентов семантических моделей информационного поиска. Сюда, прежде всего, входят различные информационно-поисковые агенты;

- библиотека компонентов семантических моделей интеграции знаний и машин обработки знаний;
- библиотека интерпретаторов программ, соответствующих различным языкам программирования;
- библиотека различных стратегий решения задач, различных моделей решения задач и агентов, входящих в состав таких моделей;
- библиотека компонентов пользовательских интерфейсов.

Все компоненты, включаемые в состав общей библиотеки компонентов интеллектуальных систем, оформляются как компоненты интеллектуальной собственности (intellectual property), поэтому будем их также называть ip-компонентами.

Особо подчеркнем то, что модульное проектирование интеллектуальных систем возможно только в том случае, если отбор компонентов, включаемых в состав рассмотренной библиотеки, будет осуществляться на основе тщательного анализа качества этих компонентов. Одним из важнейших критериев такого анализа является семантическая совместимость анализируемых компонентов со всеми компонентами, имеющимися в текущей версии библиотеки.

Для обеспечения семантической совместимости таких компонентов интеллектуальных систем, которые являются унифицированными семантическими моделями (sc-моделями знаний, sc-моделями машин обработки знаний, sc-агентов, sc-моделями интеллектуальных подсистем), необходимо (1) согласовать семантику (смысл) всех используемых ключевых узлов и (2) согласовать глобальные идентификаторы ключевых узлов, используемых в разных компонентах. После этого интеграция всех компонентов, входящих в состав библиотеки, и в любых комбинациях осуществляется автоматически, без вмешательства разработчика.

## Принцип 16. Платформенно-независимый характер проектирования интеллектуальных систем:

Для максимальной платформенной независимости технологии обеспечить четкое разделение процесса проектирования формального описания логико-семантической модели разрабатываемой интеллектуальной системы от процесса реализации (интерпретации) этой модели на той или иной платформе.

Подчеркнем при этом следующее. Если каждой интеллектуальной системе соответствует своя уникальная логико-семантическая модель, то каждый интерпретатор абстрактных логико-семантических моделей интеллектуальных систем должен обеспечивать интерпретацию целого класса таких моделей, а в идеале – интерпретацию любой такой модели. Следовательно, разработка указанных интерпретаторов может осуществляться абсолютно независимо от разработки логико-семантических моделей конкретных интеллектуальных систем.

Таким образом, SC-код, обеспечивающий унификацию семантического представления любых знаний, вместе с языком SCP, обеспечивающим унификацию формального описания агентов, работающих над семантической памятью, являясь средством унификации логико-семантических моделей интеллектуальных систем, выполняют в рамках предлагаемой технологии роль, аналогичную той, которую выполняет язык VHDL в современных микроэлектронных технологиях. В лице SC-кода и языка SCP мы имеем стандарт полного (!) формального описания логико-семантических моделей интеллектуальных систем, обеспечивающий независимость проектирования абстрактных логико-семантических моделей конкретных интеллектуальных систем от разработки различных вариантов реализации (различных вариантов их интерпретации на различных платформах). Такой стандарт является своего рода "водоразделом" между полным платформенно-независимым описанием интеллектуальной системы (абстрактной логико-семантической моделью) и платформенно зависимой реализацией (интерпретацией) этой абстрактной модели.

Полностью построенная абстрактная логико-семантическая модель проектируемой интеллектуальной системы:

(1) является открытой, поскольку ее можно легко пополнять новыми знаниями и навыками, интегрируя их в текущую версию модели;
(2) концентрирует внимание на семантические аспекты функционирования интеллектуальной системы и не содержит никаких лишних деталей, обусловленных тем или иным способом ее технической реализации (интерпретации);
(3) является абстрактным инвариантом целого множества самых различных способов ее технологической реализации (в том числе и с помощью принципиально новых компьютеров).

Разработка прототипа интеллектуальной системы завершается разработкой полной sc-модели этой системы, которая записывается в виде исходного текста с использованием таких языковых средств, как SCg-код, SCs-код, SCn-код. После этого разработчик выбирает один из универсальных (!) вариантов интерпретации (реализации) sc-моделей, загружает разработанные им исходные тексты в выбранный интерпретатор и получает прототип, пригодный для опытной эксплуатации и последующего совершенствования.

Если же после этого разработчиков интеллектуальной системы что-то не устраивает в выбранном варианте интерпретации sc-моделей (в частности, производительность), должна существовать достаточно продуманная методика совершенствования выбранного варианта интерпретатора sc-моделей интеллектуальных

систем. Очевидно, что для каждого варианта интерпретации sc-моделей интеллектуальных систем указанная методика будет иметь свои особенности.

Следовательно, нижние уровни детализации проектируемых интеллектуальных систем, в отличие от верхнего (логико-семантического) являются платформенно-зависимыми. Можно говорить о различных модификациях технологии проектирования интеллектуальных систем, соответствующих разным платформам. Напомним при этом, что основная трудоемкость проектирования интеллектуальных систем, полностью определяющая уровень ее возможностей (уровень знаний и навыков) концентрируется именно на 1-ом этапе проектирования – на разработке ее абстрактной логико-семантической модели.

Таким образом, проектирование интеллектуальной системы можно организовать как два следующих самостоятельных процесса, выполняемых одновременно и независимо друг от друга:

(1) Процесс разработки абстрактной унифицированной логико-семантической модели проектируемой интеллектуальной системы;
(2) Процесс совершенствования выбранного интерпретатора абстрактных унифицированных логико-семантических моделей интеллектуальных систем.

Заметим, что сама идея обеспечения кросс-платформенной разработки компьютерных систем путем внедрения формального языка, обеспечивающего описание абстрактных (логических) моделей этих систем не нова. Существует целый ряд кросс-платформенных технологий. Вопрос в том (1) о каком классе разрабатываемых компьютерных систем идет речь, (2) какими свойствами обладают используемые абстрактные модели компьютерных систем, (3) какими достоинствами обладает технология разработки самих этих абстрактных моделей.

Вопросы программной реализации и, в частности, web-ориентированной реализации унифицированных логико-семантических моделей интеллектуальных систем рассмотрены в работе [Колб, 2012].

## Принцип 17. Семантический ассоциативный параллельный компьютер:

Обеспечить возможность реализации унифицированных логико-семантических моделей интеллектуальных систем на **семантических ассоциативных параллельных компьютерах**, специально ориентированных на аппаратную реализацию таких моделей.

Очевидно, что для указанных компьютеров базовый графовый язык программирования (язык SCP) является их ассемблером, т. е. аппаратно интерпретируемым языком программирования.

В связи с проблемой создания компьютеров, ориентированных на обработку знаний, необходимо отметить следующее:

(1) В таких компьютерах принципиально важна поддержка именно параллельной обработки знаний;

(2) Опыт использования параллельных компьютеров показывает, что эффективное их использование предполагающее разработку качественных параллельных программ требует особой профессиональной подготовки и высокой квалификации. Мир параллельного программирования требует особой культуры, особого стиля мышления. Еще более серьезная профессиональная подготовка требуется для разработки параллельных программ, ориентированных на обработку знаний и использующих ассоциативный доступ к обрабатываемой информации;

(3) Уровень развития микроэлектронных технологий сейчас позволяет достаточно быстро реализовывать самые смелые компьютерные архитектуры и модели обработки информации. Необходима только четкая постановка задачи;

(4) Созданию параллельных компьютеров для обработки знаний должно предшествовать создание **технологии** проектирования интеллектуальных систем, в основе которой лежат те модели параллельной обработки знаний, которые будут аппаратно поддерживаться в указанных компьютерах. Иначе мы получим "груду" талантливо сделанного "железа", эффективность использования которого будет, мягко говоря, весьма низкой. Это главная причина неудач такого рода проектов;

(5) Предлагаемая технология проектирования интеллектуальных систем как раз и предполагает последовательное выполнение следующих этапов:
- разработка технологии проектирования абстрактных унифицированных логико-семантических моделей интеллектуальных систем;
- разработка нескольких вариантов программной реализации абстрактных унифицированных логико-семантических моделей интеллектуальных систем, выполненных на современных компьютерах;
- разработка, эксплуатация достаточно большого количества прикладных интеллектуальных систем и совершенствование технологии проектирования интеллектуальных систем на основе приобретенного опыта;

- и только после этого разработка семантического ассоциативного компьютера, появление которого не отменит абсолютно ничего, сделанного ранее. Просто появится ёще один, но уже аппаратный вариант реализации абстрактных унифицированных логико-семантических моделей интеллектуальных систем, применение которого для уже разработанных абстрактных унифицированных логико-семантических моделей самых различных прикладных систем для конечных пользователей этих интеллектуальных систем абсолютно ничего не изменит, кроме существенного повышения быстродействия.

Рассматривая абстрактную sc-машину обработки знаний на самом верхнем уровне, мы не уточняем (не детализируем) "внутреннее устройство" sc-агентов обработки знаний. Разработав язык SCP, мы получили возможность формально описывать (детализировать) поведение sc-агентов обработки знаний. Если трактовать язык SCP как ассемблер семантического ассоциативного компьютера, то проектирование этого компьютера можно рассматривать как формальный переход к sc-машинам более низкого уровня, обеспечивающим интерпретацию sc-машин более высокого уровня. Существенным здесь является то, что при этом мы не выходим за пределы класса абстрактных sc-машин. Просто вводится последовательность sc-языков программирования все более и более низкого уровня, каждый из которого обеспечивает формальное описание sc-агентов, входящих в состав sc-машины, интерпретирующей программы непосредственно предшествующего ему sc-языка программирования более высокого уровня (см. принцип 9). При этом число таких уровней, т. е. число таких специальных sc-языков программирования (которые можно назвать sc-языками микропрограммирования) должно быть столько, сколько необходимо для доведения формального описания sc-машин до такого уровня детализации, который позволяет перейти от соответствующего абстрактного языка микропрограммирования к формальному описанию цифровой аппаратуры на языке VHDL.

Архитектуру аппаратной реализации семантических моделей обработки знаний можно рассматривать как иерархию абстрактных машин, описывающих переход от агентов, имеющих доступ ко всей семантической памяти, к агентам, имеющим доступ только к своей семантической окрестности и, в конечном счете, взаимодействующим только со своими семантическими соседями.

Аппаратная интерпретация абстрактных sc-машин предполагает создание реконфигурируемой памяти с распределенными в ней процессорными элементами. Такую интеграцию памяти и процессора будем называть **процессоро-памятью**. Реконфигурируемость (структурная перестраиваемость) памяти может быть обеспечена коммутационной средой для процессорных элементов. Можно рассматривать целый ряд подходов к реализации реконфигурируемой семантической ассоциативной процессоро-памяти. В частности, процессорным элементам можно ставить в соответствие узлы обрабатываемых унифицированных семантических сетей, а коммутируемым каналам связи между процессорными элементами – коннекторы этой семантической сети. В этом случае текущее состояние конфигурации коммутируемых каналов связи будет полностью соответствовать текущему состоянию конфигурации обрабатываемой семантической сети. Следовательно, память "превращается" из пассивного хранилища байтов в коммутационную среду между процессорными элементами.

## Принцип 18. Встроенные подсистемы интеллектуальных систем, обеспечивающие их эффективную эксплуатацию и эволюцию:

Каждую проектируемую интеллектуальную систему трактовать как **результат интеграции следующих интеллектуальных подсистем**:

- предметной (основной) интеллектуальной системы;
- интеллектуального пользовательского интерфейса;
- интеллектуальной подсистемы адаптивного управления диалогом с конечным пользователем;
- интеллектуальной help-системы для информационного обслуживания и обучения конечных пользователей предметной интеллектуальной системы, которые, начиная работать с системой не обязаны иметь сразу высокую квалификацию;
- интеллектуальные системы управления проектированием интеллектуальной системы, которая координирует деятельность разработчиков предметной интеллектуальной системы [Грибова, 2010];
- интеллектуальные системы управления информационной безопасностью предметной интеллектуальной системы.

Подчеркнем, что для обеспечения интегрируемости (семантической совместимости) перечисленных интеллектуальных систем они должны проектироваться на основе одной и той же технологии.

Таким образом, проектируя каждую интеллектуальную систему, необходимо одновременно проектировать:

- также и подсистему, которая осуществляет информационное обслуживание и обучение конечных пользователей данной интеллектуальной системы, т. е. фактически, является оформлением документации по эксплуатации системы в виде интеллектуальной справочной и обучающей системы. Это существенно расширит контингент конечных

пользователей, повысит эффективность эксплуатации системы и существенно упростит эту эксплуатацию;

- также и подсистему, которая обеспечивает координацию разработчиков проектируемой интеллектуальной системы, поскольку разработка (совершенствование) системы продолжается в ходе её эксплуатации и требует создания специальных методов и компьютерных средств постоянного совершенствования предметной интеллектуальной системы непосредственно в ходе её эксплуатации. Это существенно отодвинет срок её морального старения;

- также и подсистему, обеспечивающую управление информационной безопасности проектируемой интеллектуальной системы.

Если подсистема управления проектированием интеллектуальной системы будет создаваться действительно как интеллектуальная система, интегрируемая с основной (предметной) интеллектуальной системой, то в перспективе она может стать не только координатором деятельности разработчиков, но и самостоятельным субъектом проектирования, способным тестировать, диагностировать, анализировать как основную проектируемую интеллектуальную систему, так и самоё себя.

## Принцип 19. Доступность и открытость технологии:

Обеспечить максимально возможное **расширение контингента разработчиков** интеллектуальных систем, использующих предлагаемую технологию, за счет максимальной доступности этой технологии и открытого характера её развития.

Если технология проектирования интеллектуальных систем ориентируется на широкое, массовое распространение и на интенсивное собственное развитие, опирающееся на накапливаемый опыт её использования, она должна быть доступной и открытой. Это означает:

- свободный доступ к всей документации и основанным средствам автоматизации (компьютерной поддержки) проектирования интеллектуальных систем;

- открытость исходных текстов всех основных средств компьютерной поддержки проектирования интеллектуальных систем, всех основных многократно используемых (типовых) компонентов интеллектуальных систем;

- открытость исходных текстов всех "пилотных" проектов прикладных интеллектуальных систем, выполняющих роль "образцово-показательных" проектов;

- открытый характер организации (project-менеджмента) процесса развития технологии, имеющий форму открытого (open source) проекта, участником которого может быть любой желающий, в том числе, и любой пользователь этой технологии, указывающий на различные ошибки и высказывающий различные пожелания.

Завершая рассмотрение открытого характера предлагаемой технологии, сделаем следующие замечания:

- открытый характер технологии не является препятствием для реализации коммерческих интересов, связанных с этой технологией. Так, например, на коммерческой основе могут создаваться и предоставляться
  (1) самые различные прикладные интеллектуальные системы,
  (2) некоторые варианты реализации различных многократно используемых (типовых) компонентов интеллектуальных систем,
  (3) некоторые варианты реализации интерпретатора абстрактных семантических логико-семантических моделей интеллектуальных систем, в частности, различные варианты построения семантических ассоциативных параллельных компьютеров;

- открытый характер технологии, при грамотном использовании фактора её открытости, способствует обеспечению информационной безопасности, как самой технологии, так и прикладных интеллектуальных систем, созданных на её основе;

- открытый характер предлагаемой технологии проектирования интеллектуальных систем может быть эффективно реализован только на базе **технологии облачных вычислений**, в рамках которой вся предлагаемая технология проектирования интеллектуальных систем рассматривается как некий Internet-сервис [Грибова и др., 2011]

## Принцип 20. Эволюционная методика проектирования:

Использовать **методику поэтапного эволюционного проектирования** интеллектуальных систем.

Указанная методика предполагает:

- быстрое проектирование;
- скорейшее введение в эксплуатацию первых версий проектируемой системы с минимальными, но практически полезными возможностями;
- эволюционное поэтапное совершенствование проектируемой интеллектуальной системы путем её расширения новыми знаниями и навыками непосредственно в ходе эксплуатации интеллектуальной системы и активным привлечением её конечных пользователей.

С формальной точки зрения проектирование унифицированной логико-семантической модели (sc-модели) интеллектуальной системы в конечном счете сводится к проектированию sc-модели **базы знаний** этой интеллектуальной системы, поскольку scp-программы, описывающие поведение sc-агентов, можно рассматривать как часть базы

знаний. Таким образом, проектируемая база знаний включает в себя:

- базу знаний предметной (основной) интеллектуальной системы;
- тексты всех scp-программ, описывающих поведение sc-агентов;
- текст документации, представленный в виде базы знаний интеллектуальной help-системы, обеспечивающей всестороннее информационное обслуживание пользователей проектируемой интеллектуальной системы.

Начальный этап проектирования базы знаний интеллектуальной системы – это уточнение иерархической системы предметных областей, которые должны быть описаны в проектируемой базы знаний. Каждой такой предметной области ставится в соответствие определенный раздел проектируемой базы знаний. Среди выделенных разделов проектируемой базы знаний имеются разделы, которые делятся (декомпозируются) на подразделы, а также атомарные (недекопозируемые) разделы. Далее процесс проектирования всей базы знаний сводится к проектированию каждого её атомарного раздела с последующей их интеграцией в единую базу знаний.

В целом начальную стадию проектирования всей интеллектуальной системы на основе предлагаемой технологии условно разбить на следующие четыре этапа:

(1) Разработка 1-й версии интеллектуальной системы, которая включает в себя:
   - 1-ю версию её базы знаний;
   - типовое ядро интеллектуальной информационно-поисковой машины, которое входит в состав библиотеки многократно используемых компонентов интеллектуальных систем;
   - типовое ядро интеллектуального решателя, которое входит в состав библиотеки многократно используемых компонентов интеллектуальных систем;
   - типовое ядро пользовательского интерфейса, которое входит в состав библиотеки многократно используемых компонентов интеллектуальных систем.

   Разработанная 1-я версия интеллектуальной системы уже обладает определенной целостностью, её можно тестировать и запускать в предварительную опытную эксплуатацию.

(2) Разработка 2-й версии интеллектуальной системы, которая включает в себя:
   - 2-ю версию её базы знаний;
   - 1-ю версию её информационно-поисковой машины;
   - типовое ядро её интеллектуального решателя;
   - типовое ядро её пользовательского интерфейса;

(3) Разработка 3-й версии интеллектуальной системы, включающей в себя:
   - 3-ю версию её базы знаний;
   - 2-ю версию её информационно-поисковой машины;
   - 1-ю версию её интеллектуального решателя;
   - типовое ядро её пользовательского интерфейса;

(4) Разработка 4-й версии интеллектуальной систем, включающей в себя:
   - 4-ю версию её базы знаний;
   - 3-ю версию её информационно-поисковой машины;
   - 2-ю версию её интеллектуального решателя;
   - 1-ю версию её пользовательского интерфейса.

Дальнейшее развитие проектируемой интеллектуальной системы может акцентировать внимание на самых разных направлениях, приоритетность которых определяется самим приложением.

Более подробно методика эволюционного коллективного проектирования унифицированных семантических моделей интеллектуальных систем, на основе содержательной структуризации знаний (см. принцип 6), описана в работе [Давыденко, 2012]

## Принцип 21. Реализация предлагаемой технологии в виде интеллектуальной метасистемы:

Реализовать предлагаемую технологию как **интеллектуальную метасистему ориентированную на поддержку проектирования интеллектуальных систем**, построенную по тем же самым принципам (т. е. по той же технологии), что и интеллектуальные системы, разрабатываемые на её основе.

Указанная интеллектуальная система должна включать в себя:

- теорию (принципы построения) проектируемых интеллектуальных систем, которая входит в состав базы знаний метасистемы;
- библиотеку типовых многократно используемых компонентов (ip-компонентов) интеллектуальных систем, которая входит в состав базы знаний рассматриваемой метасистемы;
- средства автоматизации синтеза, анализа и имитационного моделирования проектируемых интеллектуальных систем и их компонентов (это подсистема интеллектуальной метасистемы, ориентированная на решение задач проектирования интеллектуальных систем);
- интеллектуальную help-систему, являющуюся подсистемой рассматриваемой интеллектуальной метасистемы ориентированной на информационное обслуживание и обучение разработчиков интеллектуальных систем;
- методику проектирования интеллектуальных систем, которая оформляется как часть базы знаний метасистемы;

- методику обучения проектированию интеллектуальной системы, которая также является частью базы знаний метасистемы;
- интеллектуальную подсистему управления проектированием самой метасистемы;
- интеллектуальную подсистему управления информационной безопасностью метасистемы;
- семейство различных вариантов реализации интерпретаторов унифицированных абстрактных логико-семантических моделей интеллектуальных систем.

Учитывая рассматриваемые выше принципы построения предлагаемой нами технологии, она названа Открытой Семантической Технологией проектирования Интеллектуальных Систем (Open Semantic Technology for Intelligent Systems – OSTIS). Можно также её было бы назвать **SC-технологией**, поскольку основой этой технологии является SC-код. Соответственно этому, интеллектуальную метасистему, ориентированную на поддержку проектирования интеллектуальных систем, будем называть **метасистемой OSTIS**.

В интеллектуальной метасистеме OSTIS можно выделить целый ряд подсистем, ориентированных на поддержку проектирования различных компонентов интеллектуальных систем, таких, как:
- базы знаний и различные фрагменты баз знаний (онтологии, формальные теории, программы);
- информационно-поисковые машины, машины интеграции знаний, решатели задач;
- пользовательские интерфейсы (графические, естественно-языковые, мультимодальные).

В интеллектуальной метасистеме OSTIS можно также выделить семейство интеллектуальных подсистем, ориентированных на поддержку проектирования различных классов интеллектуальных система, таких, как:
- интеллектуальные справочные системы (системы информационного обслуживания);
- интеллектуальные обучающие системы (имеющие подсистемы интеллектуального управления обучением);
- интеллектуальные help-системы для пользователей различных компьютерных систем;
- интеллектуальные системы автоматизированного проектирования;
- интеллектуальные системы управления проектами.

## Заключение

В предлагаемой технологии OSTIS существенным являются не сами рассмотренные выше принципы, некоторые из которых выглядят очевидными и бесспорными, а весь целостный комплекс этих принципов и их максимально возможная согласованность.

Ключевыми проблемами, решение которых лежит в основе предлагаемой технологии являются:
- обеспечение семантической совместимости (интегрируемости) различных моделей представления и обработки знаний;
- создание общей теории абстрактных семантических моделей интеллектуальных систем, не противопоставляя, а интегрируя самые различные подходы;
- обеспечение максимальной возможной независимости интеллектуальных систем от многообразия вариантов и платформ их технической реализации (в т. ч. и от будущих компьютеров, специально ориентированных на аппаратную поддержку обработки знаний).

Таблица 1 – Стандарты технологии OSTIS

<table>
<tr><td colspan="4">sc-модели интеллектуальных систем:<br>• sc-модели пользовательских интерфейсов;<br>• sc-модели help-систем;<br>• sc-модели подсистем управления проектами;<br>• sc-модели систем поддержки проектирования;<br>• sc-модели подсистем управления информационной безопасностью</td></tr>
<tr><td colspan="4">sc-модели решателей задач</td></tr>
<tr><td colspan="2">sc-модели информационного поиска</td><td colspan="2">sc-модели интеграции знаний</td></tr>
<tr><td colspan="4">sc-модели баз знаний</td></tr>
<tr><td colspan="2">язык SCK</td><td colspan="2">унифицированные идентификаторы sc-элементов</td></tr>
<tr><td>SC-язык онтологий</td><td>sc-языки целей, вопросов и задач</td><td>Логический sc-язык</td><td>sc-языки программирования</td></tr>
<tr><td colspan="4">Язык SCP</td></tr>
<tr><td colspan="4">scp-машина</td></tr>
<tr><td>SC-код</td><td>SCg-код</td><td>SCs-код</td><td>SCn-код</td></tr>
<tr><td colspan="2">программные интерпретаторы scp-машины</td><td colspan="2" rowspan="2">scp-компьютеры</td></tr>
<tr><td>Internet</td><td>Локальные платформы</td></tr>
</table>

## Библиографический список


**[Айзерман и др., 1988]** Айзерман, М.А. Динамический подход к анализу структур, описываемых графами (основы графодинамики) / М. А. Айзерман, Л. А. Гусев, С. В. Петров, И. М. Смирнова, Л. А. Тененбаум // Исследования по теории структур. - М. : Наука, 1988. - С. 5-76.

**[Бениаминов, 1988]** Бениаминов, Е.М. Основания категорного подхода к представлению знаний. Категорные средства / Е. М. Бениаминов // Изв. АН СССР. Техн. кибернет. - 1988. - N 2. – С. 21-33.

**[Борщев, 1983]** Борщев, В.Б. Схемы на клубных системах и вегетативная машина / В. Б. Борщев // Семиотика и информатика. - 1983. - Вып. 22. - с. 3-44.

**[Вагин и др., 2008]** Вагин, В.Н. Достоверный и правдоподобный вывод в интеллектуальных системах / В. Н. Вагин [и др.]. - М. : ФИЗМАТЛИТ, 2008. - 712 с.

**[Гаврилова и др., 2000]** Гаврилова, Т.А. Базы знаний интеллектуальных систем / Т. А. Гаврилова, В.Ф. Хорошевский. – СПб.: Питер, 2000.

**[Гастев, 1975]** Гастев, Ю.А. Гомоморфизмы и модели. Логико-алгебраические аспекты моделирования / Ю.А. Гастев. - М.: Наука, 1975.

**[Гладун, 1994]** Гладун, В.П. Процессы формирования новых знаний / В. П. Гладун. – София : Педагог, 1994.

**[Голенков и др, 2001]** Голенков, В.В. Представление и обработка знаний в графодинамических ассоциативных машинах /В. В. Голенков [и др.] – Мн. : БГУИР, 2001.

[**Грибова и др., 2011**] Грибова, В. В. Автоматизация разработки пользовательских интерфейсов с динамическими данными / В. В. Грибова, Н. Н. Черкезишвили // Материалы

международной научно-технической конференции «Открытые семантические технологии проектирования интеллектуальных систем» – Минск, 2011. –С. 287-292

**[Гуляева, 1989]** Гуляева, Д.М. Решение прикладных задач на расширенных семантических сетях. / Д. М. Гуляева // Математическое обеспечение ЭВМ и систем программирования. - М., 1989.

**[Гулякина, 2012]** Гулякина, Н. А. Языки и технологии программирования, ориентированные на обработку семантических сетей / Н. А. Гулякина, О. В. Пивоварчик, Д. А. Лазуркин // Материалы международной научно-технической конференции «Открытые семантические технологии проектирования интеллектуальных систем» – Минск, 2012. –С. 221-228

**[Гусаков и др., 1981]** Гусаков, В.Я. Динамические алгебраические системы как математическая модель банка данных / В.Я. Гусаков, С. М. Гусакова // Семиотика и информатика. - 1981. - Вып. 17. - С. 43-52.

**[Давыденко, 2012]** Давыденко, И. Т. Комплексная методика проектирования семантических моделей интеллектуальных справочных систем / И. Т. Давыденко // Материалы международной научно-технической конференции «Открытые семантические технологии проектирования интеллектуальных систем» – Минск, 2012. –С. 457-466

**[Евгенев, 2008]** Евгенев, Г.Б. Технология создания многоагентных прикладных систем / Г. Б. Евгенев // Одиннадцатая национальная конференция по искусственному интеллекту : Труды конференции. Т.2. – М., 2008. – С. 306-312.

**[Епифанов, 1984]** Епифанов, М.Е. Индуктивное обобщение в ассоциативных сетях / М. Е. Епифанов // Известия АН СССР. Техническая кибернетика. №5, 1984.- С. 132-146.

**[Ефимова и др., 1988]** Ефимова, С.М. Поиск в базах знаний, опирающихся на модель П-графов, и его аппаратная реализация на основе метода $M^3$ / С. М. Ефимова, Е.В. Суворова. – М : Вычислительный центр АН СССР, 1988.

**[Загорулько, 1988]** Загорулько, Ю.А. Технология конструирования средств обработки знаний на основе семантических сетей. Средства спецификации и настройки / Ю. А. Загорулько. - Новосибирск, 1988.

**[Заливако и др., 2012]** Заливако, С. С. Семантическая технология компонентного проектирования интеллектуальных решателей задач / С. С. Заливако, Д. В. Шункевич // Материалы международной научно-технической конференции «Открытые семантические технологии проектирования интеллектуальных систем» – Минск, 2012. –С. 297-314.

**[Ивашенко, 2012]** Ивашенко, В. П. Семантические модели и средства интеграции и отладки баз знаний / В. П. Ивашенко // Материалы международной научно-технической конференции «Открытые семантические технологии проектирования интеллектуальных систем» – Минск, 2012. –С. 193-204

**[Калиниченко, 1983]** Калиниченко, Л.А. Методы и средства интеграции неоднородных баз данных / Л. А. Калиниченко. - М.: Наука, 1983.

**[Кандрашина и др., 1989]** Кандрашина, Е.Ю. Представление знаний о времени и пространстве в интеллектуальных системах / Е. Ю. Кандрашина, Л. В. Литвинцева, Д. А. Поспелов - М.: Наука, 1989.

**[Карабеков и др., 2008]** Карабеков, Б.А. Система «Бинарная Модель Знаний» как инструмент для концептуального моделирования бизнес-процессов // Одиннадцатая национальная конференция по искусственному интеллекту : Труды конференции. Т.2. – М., 2008. – С. 282-291.

**[Касьянов, 2003]** Касьянов, В.Н. Графы в программировании: обработка, визуализация и применение/ В. Н. Касьянов, В. А. Евстигнеев // BHV–Санкт-Петербург, 2003.–1104 с.

**[Клещев, 1986]** Клещев, А. С. Семантические порождающие модели. Общая точка зрения на фреймы и продукции в экспертных системах / А. С. Клещев. - Владивосток, 1986.

**[Колб, 2012]** Колб, Д. Г. Web-ориентированная реализация семантических моделей интеллектуальных систем / Д. Г. Колб // Материалы международной научно-технической конференции «Открытые семантические технологии проектирования интеллектуальных систем» – Минск, 2012. –С. 111-122

**[Колмогоров, 1958]** Колмогоров, А.Н. К определению алгоритма / А. Н. Колмогоров // Успехи математических наук. - 1958. - Т.13. - N 4(82). - С. 3-28.

**[Корончик, 2012]** Корончик, Д. Н. Семантические модели мультимодальных пользовательских интерфейсов и семантическая технология их проектирования / Д. Н. Корончик // Материалы международной научно-технической конференции «Открытые семантические технологии проектирования интеллектуальных систем» – Минск, 2012. –С. 339-346

**[Котов и др., 1966]** Котов, В.Е. Асинхронные вычислительные процессы над общей памятью / В. Е. Котов, А. С. Нариньяни // Кибернетика. - 1966. - N 3. - С. 64-71.

**[Кузнецов В.Е., 1989]** Кузнецов, В.Е. Представление в ЭВМ неформальных процедур / В. Е. Кузнецов. - М.: Наука, 1989.

**[Кузнецов И.П., 1986]** Кузнецов, И.П. Семантические представления / И.П. Кузнецов. – М : Наука, 1986.

**[Лозовский, 1984]** Лозовский, В.С. Семантические сети / В. С. Лозовский // Представление знаний в человеко-машинных и робототехнических системах. – М. : ВИНИТИ, 1984. – С. 84-121.

**[Любарский, 1980]** Любарский, Ю.Я. Интеллектуальные информационные системы / Ю.Я. Любарский. – М : Наука, 1980.

**[Люгер, 2003]** Люгер, Дж.Ф. Искусственный интеллект: стратегии и методы решения сложных проблем / Дж.Ф. Люгер. – М. : Вильямс, 2003.

**[Мальцев, 1970]** Мальцев, А. И. Алгебраические системы / А. И. Мальцев. – М.: Наука, 1970.

**[Марковский, 1997]** Марковский, А. В. Анализ структуры знаковых ориентированных графов / А. В. Марковский. // Известия РАН : Теория и системы управления. - 1997. - №5.

**[Мартынов, 1977]** Мартынов, В. В. Универсальный семантический код / В. В. Мартынов. – Минск : Наука и техника, 1977.

**[Мельчук, 1974]** Мельчук, И.А. Опыт теории лингвистических моделей «Смысл-Текст». Семантика, синтаксис/ И. А. Мельчук. – М. : Наука, 1974.

**[Месарович и др., 1978]** Месарович, М. Общая теория систем: математические основы / М. Месарович, Я. Тахакара. – М. : Мир, 1978.

**[Молокова, 1992]** Молокова, О.С. Методология анализа предметных знаний / О. С. Молокова. // Новости искусственного интеллекта. - 1992. – № 3. - С.11-60.

**[Нариньяни, 1994]** Нариньяни, А.С. НЕ-факторы и инженерия знаний: от наивной формализации к естественной программатике / А. С. Нариньяни //КИИ-94. Сборник трудов Национальной конференции с международным участием по ИИ. «Искусственный интеллект-94»; в 2-х т. – Т. 1. – Тверь : АИИ, 1994.- С. 9-18.

**[Осипов, 1990]** Осипов, Г.С. Построение моделей предметных областей. Неординарные семантические сети / Г. С. Осипов. // Известия АН СССР. Техническая кибернетика. – 1990. - № 5.

**[Петров, 1978]** Петров, С.В. Графовые грамматики и автоматы (обзор) / С. В. Петров. // Автоматика и телемеханика. - 1978. - N 7. - С. 116-136.

**[Петрушкин, 1992]** Петрушкин, В.А. Экспертно-обучающие системы / В. А. Петрушкин. – Киев : Наукова думка. – 1992.

**[Плесневич, 1982]** Плесневич, Г.С. Представление знаний в ассоциативных сетях / Г. С. Плесневич // Изв. АН СССР. Техн. кибернет. - 1982. – N 5. - с.6-22.

**[Плесневич, 2008]** Плесневич, Г.С. Бинарные модели знаний / Г. С. Плесневич // Труды Международных научно-технических конференций «Интеллектуальные системы» (AIS'08) и «Интеллектуальные САПР» (CAD-2008). Научное издание в 4-х томах. – М : Физматлит, 2008, Т.2. – С. 424 – 135-146.

**[Попков, 1986]** Попков, В.К. Гиперсети и их характеристики связности / В. К. Попков. // Исследования по прикладной теории графов. - Новосибирск: Наука, 1986. - С. 25-58.

**[Поспелов, 1986а]** Поспелов, Д.А. Представление знаний. Опыт системного анализа / Д. А. Поспелов. // Системные

исследования. Методологические проблемы. Ежегодник. - М.: Наука, 1986. - с. 83-102.

**[Поспелов, 1986b]** Поспелов, Д.А. Ситуационное управление. Теория и практика / Д. А. Поспелов. – М : Наука, 1986.

**[Рабинович, 1995]** Рабинович, З.Л. О концепции машинного интеллекта и ее развитии / З. Л. Рабинович // Кибернетика и системный анализ. – 1995. – N2. –С.163-173.

**[Рассел, 2006]** Рассел, С. Искусственный интеллект: современный подход / С. Рассел, П. Норвиг. – М. : Вильямс, 2006.

**[Резанов, 1989]** Резанов, С.Н. Об одном методе обобщения на семантических сетях в системе управления энергообъединением / С. Н. Резанов // Изв. АН СССР. Техн. кибернет. - 1989. - № 5. - С. 55-62.

**[Рубашкин, 1989]** Рубашкин, В.Ш. Представление и анализ смысла в интеллектуальных информационных системах / В. Ш. Рубашкин. – М.: Наука, 1989.

**[Рыбина, 2010]** Рыбина, Г.В. Основы построения интеллектуальных систем: учеб. пособ./ Г.В. Рыбина. – М. : Финансы и статистика, 2010.

**[Сапатый, 1983]** Сапатый, П.С. Об эффективности структурной реализации операций над семантическими сетями / П. С. Сапатый // Техн. кибернет. - 1983. - № 5. - С. 128-134.

**[Семенов, 1980]** Семенов, В.В. Семантические фреймовые сети как модели предметной области для САПР САУ / В. В. Сапатый // Представление знаний в системах искусственного интеллекта. - М.: МДНТП, 1980. - С. 117-122.

**[Скороходько, 1989]** Скороходько, Э.Ф. Семантические сети и автоматическая обработка текста. / Э. Ф. Скороходько. – Киев: Наук. думка, 1983.

**[Скрэгг, 1983]** Скрэгг, Г. Семантические сети как модели памяти / Г. Скрэгг // Новое в зарубежной лингвистике. – Вып. 12. – М. : Радуга, 1983. – С. 228-271.

**[Соловьев, 1990]** Соловьев, В.А. Формирование на семантической сети понятий и суждений с помощью рассуждений по аналогии / В. А. Соловьев // II Всесоюзная конференция "Искусственный интеллект-90". Секционные и стендовые доклады. - Минск, 1990. - Т. 1. - С. 166-169.

**[Тузов, 1984]** Тузов, В.А. Математическая модель языка / В. А. Тузов. - Л.: Изд-во ленингр. ун-та, 1984.

**[Тузов, 1986]** Тузов, В.А. О формализации понятия задачи / В. А. Тузов. - М.: Наука, 1986. - С. 73-83.

**[Тыугу, 1989]** Тыугу, Э.Х. Интеграция знаний / Э. Х. Тыугу // Изв. АН СССР. Техн. кибернет. - 1989. - № 5. - с. 3-13.

**[Финн, 2008]** Финн, В.К. Многозначные логики и их применения / ред. В. К. Финн – М.:ЛКИ, 2008. – Т.1, Т.2

**[Уварова, 1987]** Уварова, Т. Г. Формальное описание операционного языка для семантических сетей. / Т. Г. Уварова, Л. Л. Лифшиц - М.: ВЦ АН СССР, 1987.

**[Хельбиг, 1980]** Хельбиг, Г. Семантическое представление знаний в вопросно-ответной системе FAS-80 / Г. Хельбиг. // Представление знаний и моделирование процессов понимания. - Новосибирск, 1980. - С. 97-123.

**[Хендрикс, 1975]** Хендрикс, Г. О расширении применяемости семантических сетей введением разбиений / Г. О. Хендрикс // Труды IV Международной объединенной конференции по искусственному интеллекту. - М., 1975. - Т. 1. - С. 190-206.

**[Хорошевский, 2008]** Хорошевский, В. Ф. Пространства знаний в сети Интернет и Semantic Web (Часть 1) / В. Ф. Хорошевский. // Искусственный интеллект и принятие решений, 2008, №1. – С.80-97.

**[Цаленко, 1989]** Цаленко, М.Ш. Моделирование семантики в базах данных. / М. Ш. Цаленко. – М. : Наука, 1989.

**[Шенк, 1980]** Шенк, Р. Обработка концептуальной информации / Р. Шенк. – Москва: Энергия, 1980.

**[Шрейдер, 1971]** Шрейдер, Ю.А. Системы и модели / Ю. А. Шрейдер, А. А. Шаров. – М.: Радио и связь, 1982.

**[Шуберт, 1979]** Шуберт, Л. Усиление выразительной мощности семантических сетей / Л. Шуберт // Кибернетический сборник. Новая серия. - 1979. - Вып. 16. - С. 171-212.

**[OSTIS, 2011]** Открытая семантическая технология проектирования интеллектуальных систем [Электронный ресурс]. – 2011. - Режим доступа: http://ostis.net. – Дата доступа: 20.11.2011

**[Russell et al., 1995]** Russell, S. Artificial Intelligence. A Modern Approach / S Russell, P Norvig. - New Jersey : Prentice Hall, – 1995.

**[Sowa, 2008]** Sowa, J. Conceptual Graphs/ John F. Sowa, F. van Harmelen, V. Lifschitz, B. Porter// eds., Handbook of Knowledge Representation, Elsevier, 2008, pp. 213-237

**[Wooldridge et al., 1994]** Wooldridge, M. Agent Theories, Architectures and Languages: A Survey / M. Wooldridge, N. Jennings // Intelligent Agents. Languages. Amsterdam : Springer Verlag, Augast, – 1994. – P. 3-39.

## GRAPHODYNAMICAL MODELS OF PARALLEL KNOWLEDGE PROCESSING

Golenkov V. V., Guliakina N. A.

*Belarusian State University of Informatics and Radioelectronics, Minsk, Republic of Belarus*

golen@bsuir.by

guliakina@bsuir.by

The principles of construction technology of designing intelligent systems which are oriented on semantic representation of knowledge, expansion of the number of developers and shortening time of design are considered.

### INTRODUCTION

Current state in the designing of computer systems reminds Babel of various approaches, models, methods, tools and platforms. Artificial intelligence technologies are not targeted to a wide range of developers of intelligent systems and therefore did not get mass distribution.

### MAIN PART

We suggest the following approach, which are aimed to eliminate the above drawbacks:

- orientation on semantic knowledge representation, which completely abstracts from the peculiarities of the technical implementation of intelligent systems;
- development of the semantic-unified and easily integrable computer systems models in the first stage of their design;
- modular designing based on the libraries of the typical reusable components of intelligent systems;
- gradual evolutionary design based on a basis rapid prototyping;
- fully compatible designing tools with the developed system - the tools are built as intelligent systems and are based on the same principles

### CONCLUSION

The paper presents the key provisions of the one of the possible approaches to the mass systems development with different levels of intelligence. The results presented in the paper are being tested within the open source project OSTIS (http://www.ostis.net) bounds. This work is supported by Belorussian and Russian foundation for basic research.