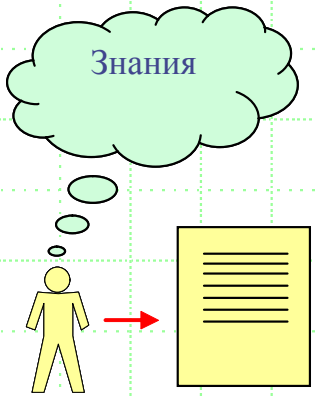
Выхованец В.С.

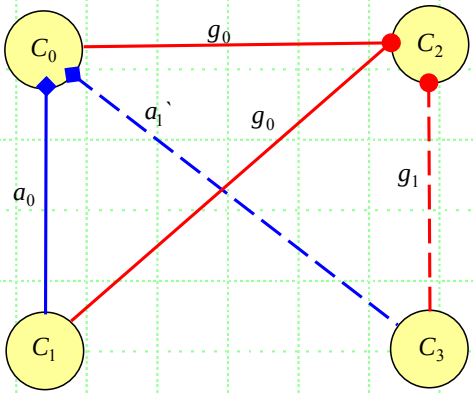
Семантически замкнутая онтология

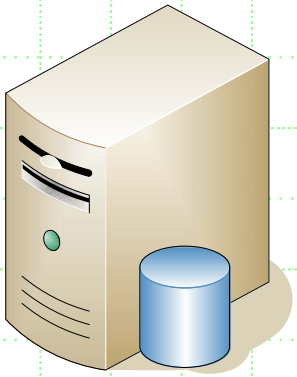
(http://valery.vykhovanets.ru)

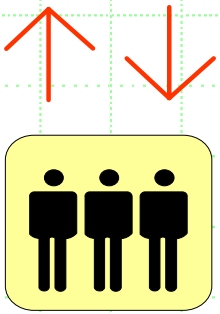
План доклада

- ***Постановка задачи***
  - Природа знания
  - Языки онтологии
  - Проблема семантики
  - Суть подхода
- ***Понятийный анализ***
  - Формализм
  - Абстрагирование
  - Понятийная структура
  - Модель понятия
- ***Контекстная технология***
  - Метаязык
  - Предметный язык
  - Понятийная модель
  - Семантическая индукция
- ***Демонстрационный пример***
  - Система управления лифтом
  - Анализ эффективности
  - Заключение









Природа знаний

Знание[1] – проверенный на практике и утвержденный некоторой последовательностью умозаключений результат субъективного познания, отраженный в сознании в виде понятий и суждений.

Приобретение знаний осуществляется в результате педагогического процесса, самообразования и жизненного опыта.

Представление знаний – отчуждение знания от его носителя во внешней (объективной) форме.

Объективация знаний[2] – признание знаний, выраженных в некоторой внешней форме объективно истинными и социально значимым.

---

[1] Данильян О.Г., Панова Н.И. Современный словарь по общественным наукам. – М.: Эксмо-Пресс, 2005 г.

[2] Бердяев Н.А. Творчество и объективация. – М.: Экономпресс, 2000.

Представление знаний

Представление знаний может быть основано на использовании семиотической системы, состоящей из двух языков: предметного языка и метаязыка.

Предметный язык служит для непосредственного выражения знаний в виде текстов, рассматриваемых как последовательность знаков.

Метаязык предназначен для выражения знаний о предметном языке, которые необходимы для правильной структурной и смысловой интерпретации текстов на предметном языке.

Естественный язык является одновременно как универсальным предметным языком, так метаязыком.

Текст – форма выражения знаний на естественном или искусственном языке, представленная в виде последовательности знаков на некотором носителе данных.

Онтология

Господствующая познавательная парадигма – для описания любой прикладной области знания первичными являются понятия и связи между ними.

Онтология – общее описание (эксплицитная спецификация) некоторой области знания в виде базовых понятий этой области, связей между понятиями и суждений о понятиях и их связях[1].

$$O =< N, R, F >$$

N – понятия (концепты);

R – отношения между понятиями, $R \subset N^k$. ← разметка

F – функции интерпретации вида $N^i \times R^j \to N$. ← нагрузка

Концептуальная схема онтологии – семантическая сеть, заданная множеством понятий и отношений между ними, $S =< N, R >$.

[1] Гаврилова Т. А., Хорошевский В. Ф. Базы знаний интеллектуальных систем. СПб.: Питер, 2000.

Языки онтологии

Глобальная семантическая сеть (Semantic Web)[1] – явное описание на некотором языке смысла терминов, неявно определенных концептуальной схемой.

- язык ***RDF*** (Resource Definition Framework) выражает самые простые предикатные отношения в виде троек объект-свойство-значение.
- язык ***OWL*** (Web Ontology Language) дополнительно включает средства для классификации знаков (классы, подклассы, свойства, ограничения).

Стандарт ***IDEF5***[2] включает язык для описания элементов онтологии EL, а зависимости между элементами задаются в графической форме на языке SL диаграммами классификаций, композиционными схемами, схемами взаимосвязей и диаграммами состояния объектов.

[1] Davies J., Fensel D., Bussler C., Studer R. The Semantic Web Research And Applications: First European Semantic Web Symposium. Heraklion, 2004.

[2] Стандарт онтологического исследования IDEF5 (www.idef.com/idef5.html)

Описание семантики

Основная проблема онтологического подхода – формальное описание семантики понятий, которое устанавливает соответствие между сущностями из мира смыслов их реализации в мире знаков.

Определение семантики основано на выделении типологии смыслов – семантических категорий, через которые и посредством которых на некотором семантическом языке определяется более сложный смысл.

Известны следующие ***модели*** описания семантики[1]:

- ***грамматические модели*** (метод атрибутных грамматик, императивный метод);
- ***аппликативные модели*** (аксиоматический метод, методы денотационный и функциональной семантики);
- ***модели спецификаций*** (метод спецификаций, алгебраический метод).

[1] Пратт Т., Зелковиц М. Языки программирования: разработка и реализация. СПб.: Питер, 2002.

Семантический разрыв

1. Не существует универсального общепринятого подхода для формального описания семантики формальных языков.

2. Ни один из известных методов не оказался приемлемым: императивный метод порождает плохо читаемые описания, денотационный метод сложен для понимания, аксиоматический метод труден в реализации, и т.д.

3. Не определяется прагматика понятий. Учет специфики прикладной области осуществляется, например, через создание многоуровневой онтологии (онтология верхнего уровня, предметная онтология, онтология задач, прикладная онтология), что приводит к проблемам их взаимной стыковки и верификации.

Вывод: Известные методы описания семантики порождают трудно устранимый семантический разрыв между содержательными представлениями относительно предметной области и теми средствами, которые служат для выражения смысла этих представлений.

Постановка задачи

1. Для формализации знаний в некоторой прикладной области будем строить специализированную семиотическую систему, состоящую из универсального *метаязыка* и специализированного *предметного языка*.

2. Данные, выражающие формализуемые знания, будем *сопровождать* описанием предметного языка, который был использован для представления этих знаний.

3. *Синтаксис* предметного языка зададим метаязыковыми средствами.

4. Описание *семантики* предметного языка выполним на самом предметном языке, т.е. предметный язык будет одновременно и специализированным семантическим языком.

5. Первичные семантические категории определим низкоуровневыми средствами целевой вычислительной платформы.

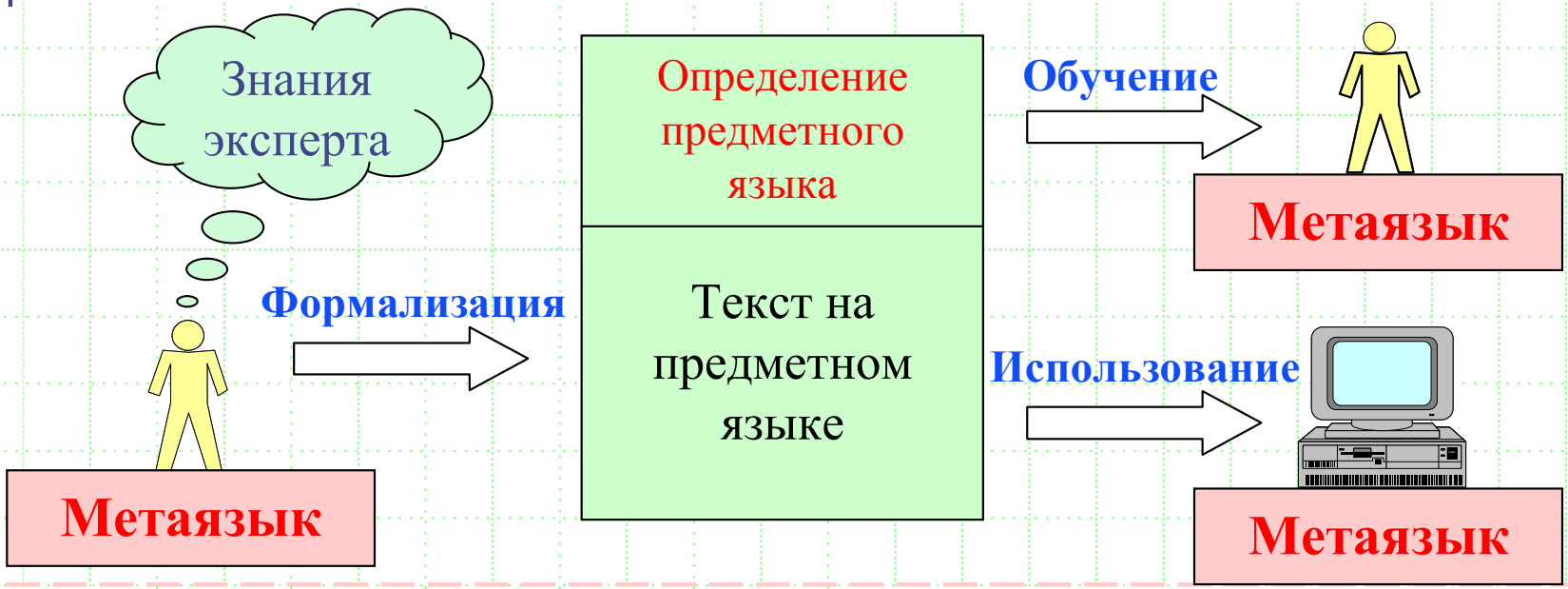
Суть предлагаемого подхода

1. ***Онтология:*** понятийная структура, определяющая способы образования (абстрагирования) понятий.

2. ***Синтаксическое замыкание:*** синтаксис предметного языка задается в виде правил выражения понятий в тексте.

3. ***Семантическое замыкание:*** семантика каждого правила выражения понятий описывается на предметном языке.

4. ***Прагматическое замыкание:*** прагматика понятий задается в виде пополняемого множества именованных семантик.



Основные определения

Проблемная область – совокупность предметной области и решаемых в ней задач (проблем).

Предметная область – фрагмент действительности, представляемый совокупностью принадлежащих ему сущностей.

Сущность – устойчивое представление о выделенной и уникально идентифицируемой части предметной области.

Признак – именованная сущность, характеризуемая множеством своих проявлений (значений) и имеющая проблемную интерпретацию (семантическую роль).

Понятие – именованное множество сущностей, объединенных на основе общности своих признаков.

Формализм понятия[1]

$$N = \begin{cases} \operatorname{shm} C = (P^0, P^1, \ldots, P^{n-1}); \\ \operatorname{int} C = \{(P_j^0, P_j^1, \ldots, P_j^{n-1}) \mid j = \overline{0, m-1}\}; \\ \operatorname{ext} C = \{C_0, C_1, \ldots, C_{u-1}\}. \end{cases}$$

Имя N выражает семантическое значение понятия C.

Схема shm C – набор признаков P^0, P^1, …, P^{n-1}, на которых понятие C определено.

Интенсионал int C – наборы значений взаимосвязанных признаков $(P_j^0, P_j^1, \ldots, P_j^{n-1})$, позволяющим отличать сущности, принадлежащие понятию C от других сущностей.

Экстенсионал ext C – множество сущностей C_k, принадлежащих понятию C.

[1] Макетирование, проектирование и реализация диалоговых информационных систем / Под ред. Е.И. Ломако. М.: Финансы и статистика, 1993.

Фрактальность понятия

$$N_E = \begin{cases} \mathrm{shm}\, E = (E); \\ \mathrm{int}\, E = \{(E)\}; \\ \mathrm{ext}\, E = \{E\}, \end{cases}$$

$$N_P = \begin{cases} \mathrm{shm}\, P = (P); \\ \mathrm{int}\, P = \{(P_0), (P_1), \ldots, (P_{u-1})\}; \\ \mathrm{ext}\, P = \{P_0, P_1, \ldots, P_{u-1}\}, \end{cases}$$

$$N_C = \begin{cases} \mathrm{shm}\, C = (N^0, N^1, \ldots, N^{n-1}); \\ \mathrm{int}\, C = \{(N_j^0, N_j^1, \ldots, N_j^{n-1})\}; \\ \mathrm{ext}\, C = \{C_0, C_1, \ldots, C_{u-1}\}, \end{cases}$$

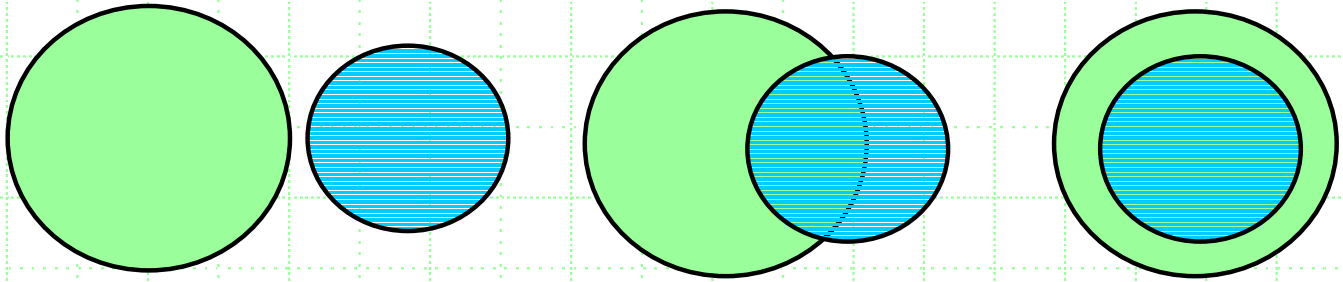
Сущность (essence) – понятие, имеющее экстенсионал и схему, которые равны этому понятию.

Признак (property) – понятие, имеющее схему, совпадающую с самим понятием.

Понятие (notion) – именованное множество понятий, имеющих одну и ту же подсхему, являющуюся схемой образуемого понятия .

Образование понятий

Пространство признаков[1]



а) независимость **б) дифференциация** **в) интеграция**

Абстрагирование – форма мышления, при которой образуются новые понятия на основе выделения общих и существенных признаков абстрагируемых понятий. Абстрагирование связано со способностью субъекта к обобщению и ассоциации.

Известны следующие абстракции[2]:

– обобщение-специализация (типизация-конкретизация);

– ассоциация-индивидуализация (агрегация-декомпозиция).

[1] Smith B., Mulligan K. Framework for Formal Ontology // Topoi. – 1983. - Vol. 2. P. 73-85.

[2] On Conceptual modeling / Ed. By M. Brodie, J. Mylopoulos, J. Schmid. – New York: Springer Verlag, 1984.

Абстракции обобщения

Обобщение – порождение понятия на основе *пересечения* схем обобщаемых понятий и *расширенного* объединения их экстенсионалов.

Специализация – из понятия выделяют *схожие* с ним понятия.

Типизация – порождение понятия на основе *пересечения* схем типизируемых понятий и *точного* объединения экстенсионалов.

Конкретизация – из понятия выделяют *эквивалентные* ему понятия.

$$\begin{cases} \operatorname{shm} C_G = \bigcap\limits_{j=0}^{m-1} \operatorname{shm} C_j; \\ \operatorname{int} C_G \supseteq \bigcup\limits_{j=0}^{m-1} \operatorname{int} C_j \\ \operatorname{ext} C_G \supseteq \bigcup\limits_{j=0}^{m-1} \operatorname{ext} C_j. \end{cases}$$

$$\begin{cases} \operatorname{shm} C_T = \bigcap\limits_{j=0}^{m-1} \operatorname{shm} C_j, \\ \operatorname{int} C_T = \bigcup\limits_{j=0}^{m-1} \operatorname{int} C_j \\ \operatorname{ext} C_T = \bigcup\limits_{j=0}^{m-1} \operatorname{ext} C_j; \\ \operatorname{key} C_T \subseteq \operatorname{shm} C_T. \end{cases}$$

Абстракции ассоциации

Ассоциация – порождение понятия на основе *объединения* схем ассоциируемых понятий и *ограниченного* декартова произведения их экстенсионалов.

$$\begin{cases} \operatorname{shm} C_B = \bigcup\limits_{j=0}^{m-1} \operatorname{shm} C_j, \\ \operatorname{int} C_B \subseteq \times\limits_{j=0}^{m-1} \operatorname{int} C_j \\ \operatorname{ext} C_B \subseteq \times\limits_{j=0}^{m-1} \operatorname{ext} C_j; \\ \operatorname{link} C_B \subseteq \operatorname{shm} C_B. \end{cases}$$

Индивидуализация – из понятия выделяют *связанные* им понятия.

Агрегация – порождение понятия на основе *объединения* схем агрегируемых понятий и *полного* декартова произведения их экстенсионалов.

$$\begin{cases} \operatorname{shm} C_A = \bigcup\limits_{j=0}^{m-1} \operatorname{shm} C_j; \\ \operatorname{int} C_A = \times\limits_{j=0}^{m-1} \operatorname{int} C_j \\ \operatorname{ext} C_A = \times\limits_{j=0}^{m-1} \operatorname{ext} C_j. \end{cases}$$

Декомпозиция – из понятия выделяют *содержащиеся* в нем понятия.

Понятийная структура

Понятийная структура $S = \langle C, G, T, A, B \rangle$ - множество понятий, на которых заданы способы их образования (абстрагирования):

$C = \{C_0, C_1, \ldots, C_{n-1}\}$ – носитель понятийной структуры;

$G = \{g_0, g_1, \ldots, g_{n_g-1}\}$ – отображения обобщения (generalization);

$T = \{t_0, t_1, \ldots, t_{n_t-1}\}$ – отображения типизации (typification);

$A = \{a_0, a_1, \ldots, a_{n_a-1}\}$ – отображения агрегации (aggregation);

$B = \{b_0, b_1, \ldots, b_{n_b-1}\}$ – отображения ассоциации (association);

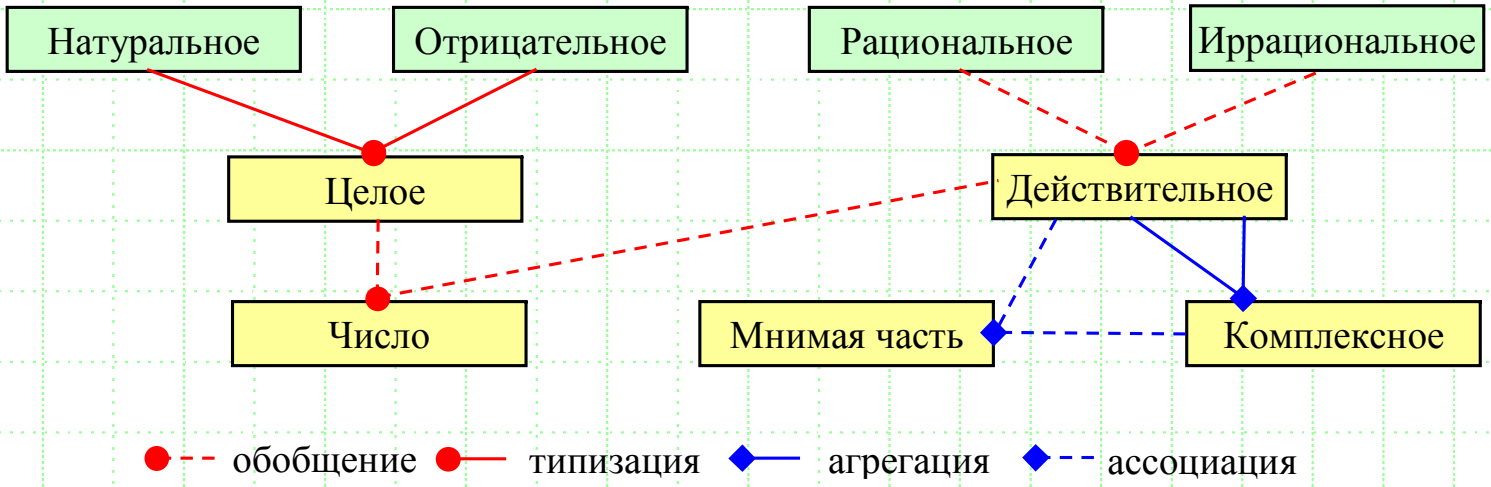


Схема понятия

Схема понятия – набор простых понятий (понятий-признаков), на которых понятие может быть определено.

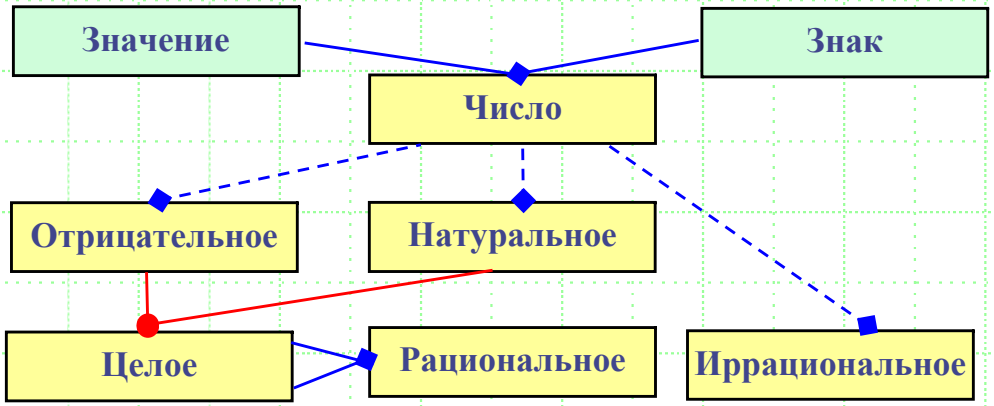
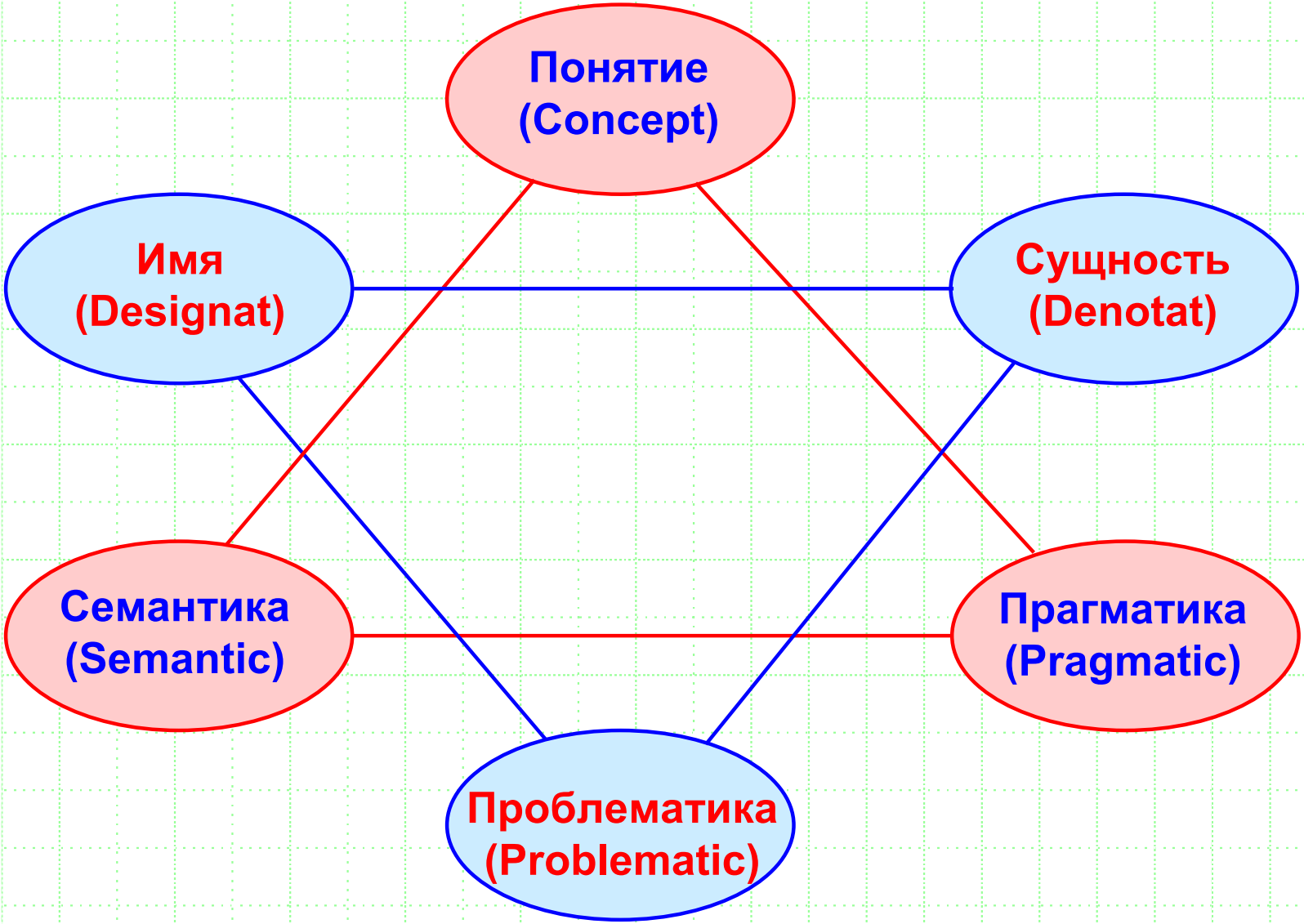


Схема «Рациональное» равна:
(«Знак», «Значение»,
«Знак», «Значение»).

Вычисление схем:

- схема простого понятия равна этому понятию;
- схема понятия, полученного при дифференциации равна пересечению схем дифференцируемых понятий;
- схема понятия, полученного при интеграции равна объединению схем интегрируемых понятий;
- схема понятия, полученного при дифференциации и интеграции равна подмножеству объединения схем интегрируемых понятий, являющемуся пересечением схем дифференцируемых понятий.

Модель понятия



Понятийный анализ

Понятийный анализ – методика построения и верификации понятийной структуры проблемной области путем:

- выявления существенных признаков у сущностей предметной области с учетом некоторой активной проблематики;
- сопоставления сущностей и определение их общих и различающихся признаков;
- образования новых или определение уже существующих понятий на основе интеграции и дифференциации понятий;
- создания понятийной структуры предметной области в виде отображений одних понятий в другие;
- уточнения способа абстрагирования понятий (обобщение-типизация, ассоциация-агрегация);
- вычисления схем понятий и задания ключей – для типизации, и связей – для ассоциации;
- тождественных преобразований понятийной структуры и проверки ее на полноту и непротиворечивость.

Контекстная технология

Контекстная технология обработки знаний :

- выделение проблемной области (проблематика);
- понятийный анализ (понятийная структура);
- описание предметного языка (понятийная модель);
- ситуационное описание (суждения);
- описание решения (умозаключения).

Программа = Модель + Решение задачи.

Модель = Структура + Синтаксис + Семантика.

на метаязыке ***на предметном языке***

Грамматика метаязыка[1]

1	program	→	model [situation] [program]
2	model	→	essences [model]
3	essences	→	differenciation ***notion*** [integration] [intension]
4	differenciation	→	'(' [notions] ')'
5	integration	→	'(' [notions] ')'
6	notions	→	**notion** [alias] [notions]
7	intension	→	sentence [intension]
8	sentence	→	syntax semantic
9	syntax	→	item [syntax]
10	item	→	**notion** [alias] | lexeme [alias]
11	alias	→	'`' ***terms*** '`'
12	lexeme	→	term | pattern
13	term	→	'/' [***terms***] '/'
14	pattern	→	'//' [***terms***] '//'
15	semantic	→	pragmatic [semantic]
16	pragmatic	→	[***aspect***] '{' [text] '}'
17	situation	→	[**aspect**] '<' [text] '>'
18	text	→	phrase [text]
19	phrase	→	**terms** | [**aspect**] '{' text '}'

[1] Грамматика представлена в металингвистической форме Бэкуса-Наура, где терминальные знаки заключены в одинарные кавычки, а нетерминальные обозначены словами. Необязательные части правил приведены в квадратных скобках. ***Курсивом*** выделенные места определения терминальных и нетерминальных знаков предметного языка, а **жирным** шрифтом – их использование в правилах грамматики.

Пример. Исчисление высказываний

Понятийная модель

```
() Variable
      ″[A-Za-z][A-Za-z0-9]*″ {…}
() Constant
      'false' { asm{ mov eax, 0; push eax } }
      'true' { asm{ mov eax, -1; push eax } }
(Variable) Logic
      Variable { asm{ pop ebx; mov eax, [ebx]; push eax } }
      Integer { asm{ pop eax; cmp eax, 0; je lab; mov eax, -1; lab: push eax } }
      '(' Boolean ')' { }
(Constant Logic) Negation
      'not' Logic { asm{ pop eax; not eax; push eax } }
(Negation) Conjunction
      Negation 'and' Negation { asm{ pop eax; pop edx; and eax, edx; push eax } }
(Conjunction) Disjunction
      Conjunction `a` 'or' Conjunction `b` { not a and not b }
(Disjunction) Boolean
      Disjunction `a` 'imp' Disjunction `b` { not a or b }

<(not x or y) and z>       fuzzy < (not x or y) and z >
```

Ситуационное описание

Семантическая индукция

Семантическая индукция состоит в определении семантических категорий по мере необходимости, в процессе определения предметного языка и средствами этого языка.

База индукции – первичные семантические категории, которые непосредственно реализуются целевой вычислительной платформой и декларируются на метаязыке перед использованием.

Предположение индукции – все ранее определенные семантические категории, заданные в виде прагматик понятий и выражающиеся фрагментами текста, которые построены по правилам, описываемым синтаксисом предложений.

Индуктивный переход – описание прагматики нового или уже существующего предложения на уже определенном до этого предметном языке.

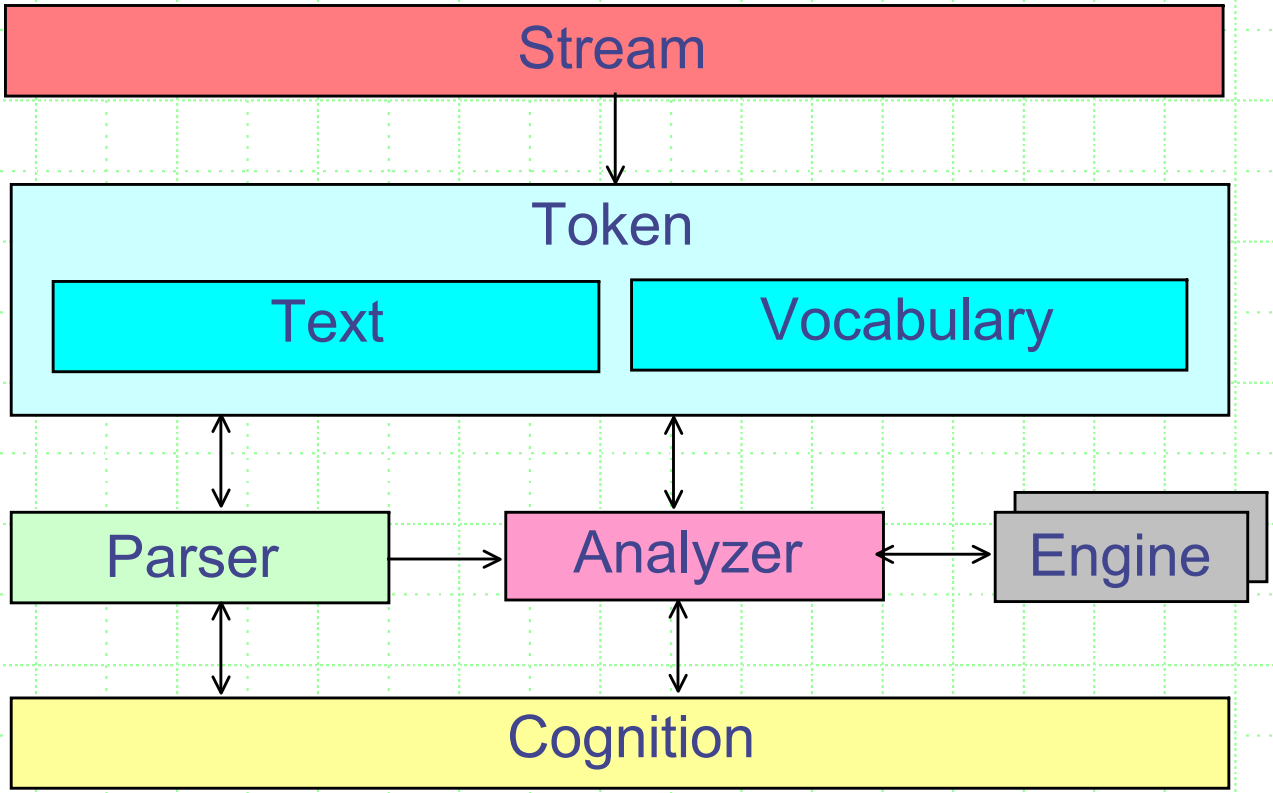
Заключение индукции – определение новой семантической категории.

Синтаксически управляемый перевод

```
(String) Expression ()
    ″[0-9]+″ `n`
        { n } dif{ ‘0’ }
    'x'
        { 'x' } dif { '1' }
    '(' Expression ')' `exp`
        { '(' & exp & ')' } dif { '(' & dif { exp } & ')' }
    'sin' '(' Expression `exp` ')'
        { 'sin(' & exp & ')' } dif { 'cos(' & exp & ')*(' & dif { exp } & ')' }
    'cos' '(' Expression `exp` ')'
        { 'cos(' & exp & ')' } dif { '–sin(' & exp & ')*(' & dif { exp } & ')' }
    '–' Expression `exp`
        { '–' & exp } dif { '–' & dif { exp } }
    Expression `exp1` '*' Expression `exp2`
        { exp1 & '*' & exp2 }
        dif { '('& exp1 & '*' & dif { exp2 } & '+' & dif { exp1 } & '*' & exp2 & ')' }
    Expression `exp1` ″+ | –″ `oper` Expression `exp2`
        { exp1 & oper & exp2 } dif { '(' & dif { exp1 } & oper & dif { exp2 } & ')' }
```

```
<dif { sin(5*cos(x)) – x*x }>
(cos(5*cos(x))*(5* – sin(x)*(1) + 0*cos(x)*(1))–(x*1 + 1*x)
equ{…} →  -5*(cos(5*cos(x))*sin(x))–2*x
```

Система контекстного программирования



Понятийные модели:

- определяемые;
- дополняемые;
- библиотечные;
- общедоступные.

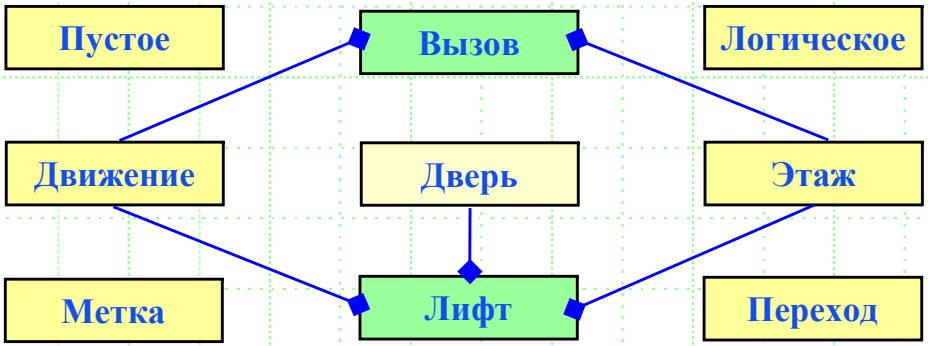
Целевые платформы:

- контроллеры, процессоры;
- виртуальные машины;
- операционные системы;
- системы программирования.

Задача управления лифтом

Дано многоэтажное здание с одним лифтом. На каждом этаже – кнопки для вызова лифта на движение вверх и вниз. В кабине имеется панель с кнопками для перемещения на этажи.

Разработать систему управления лифтом.



1. Ожидание вызова

Если вызовов нет, то ожидать вызов (1). Если вызов со второго этажа, то перейти к открытию дверей (2), иначе – на принятие решения о движении (4).

2. Открытие дверей

Открыть двери. Если дверь открылась, то перейти на принятие решения о движении (4), иначе повторить открытие двери (2).

3. Закрытие дверей

Закрыть дверь. Если дверь не закрылась, то повторить закрытие двери (3), иначе перейти к определению направления движения (4).

4. Направление движения

Продолжение движения. Если лифт двигался вверх (вниз) и имеются вызовы на движение вверх (вниз), то начать движение вверх (5) (вниз 6). ***Изменение направления***. Если вызовов на движение вверх (вниз) нет, но есть вызовы на движение вниз (вверх), то начать движение вниз (6) (вверх 5). ***Возврат в начало***. Если вызовов нет и при этом лифт выше (ниже) второго этажа, то начать движение вниз (6) (вверх 5), иначе перейти в состояние ожидания (1).

5. Движение вверх

Начать движение вверх. Если при проходе этажа имеется вызов на движение вверх, то остановить лифт и открыть дверь (2), иначе продолжить движение вверх (5).

6. Движение вниз

Начать движение вниз. Если при проходе этажа имеется вызов для движения вниз, то остановить лифт и открыть дверь (2), иначе продолжить движение вниз (6).

Понятийная модель и ситуация

```
( ) Движение ( )
      ″[Дд]вижение″ { }
( ) Дверь ( )
      ″[Дд]верь″ { }
( ) Этаж ( )
      'этаж' ″[0-9]″ { }
      'этаж' 'вызова' ″|вверх|вниз″ { }
      'этаж' 'лифта' { }
( ) Вызов (Этаж Движение)
      'вызов' { }
( ) Лифт (Этаж Движение Дверь)
      'лифт' { }
( ) Метка ( )
      ″[А-Яа-я][А-Яа-я0-9]*″ { }
( ) Переход ( )
      Метка { }
( ) Логическое ( )
      Этаж ″выше|ниже|равен″ Этаж { }
      Вызов ″вверх|вниз|нет″ { }
      Дверь ″открыта|закрыта″ { }
      'не ' Логическое { }
( ) ( )
      '. ' { }
      Метка ': ' { }
      Движение ″вверх|вниз|останов″ { }
      Дверь ″открыть|закрыть″ { }
      'Если ' Логическое ', ' 'то ' Переход { }
      'Если ' Логическое ', ' 'то ' Переход ', '
                              'иначе ' Переход { }
```

```
Ожидание:
      Если вызов нет, то Ожидание.
      Если этаж вызова равен этаж 2,
            то Открыть, иначе Решение.
Открыть:
      Дверь открыть.
      Если дверь открыта, то Решение,
            иначе Открыть.
Закрыть:
      Дверь закрыть.
      Если дверь закрыта, то Решение,
            иначе Закрыть.
Решение:
      Если лифт вверх и вызов вверх, то Вверх.
      Если лифт вниз и Вызов вниз, то Вниз.
      Если не вызов вверх и вызов вниз, то Вниз.
      Если не вызов вниз и вызов вверх, то Вверх.
      Если вызов нет и этаж лифта выше этаж 2,
            то Вниз.
      Если вызов нет и этаж лифта ниже этаж 2,
            то Вверх, иначе Ожидание.
Вверх:
      Движение вверх.
      Если этаж лифта равен этаж вызова вверх,
            то Открыть, иначе Вверх.
Вниз:
      Движение вниз.
      Если этаж лифта равен этаж вызова вниз,
            то Открыть, иначе Вниз.
```

Анализ эффективности

Длина исходного текста, написанного экспертом на естественном языке, равна 1545 знакам, причем этот текст не содержит определение своего синтаксиса и не раскрывает семантику терминов и понятий, которые в нем использованы.

Длина понятийной модели и ситуационного описания равна 1606 знакам, причем понятийная модель содержит описание синтаксиса, но не содержит описание семантики.

Предварительная эффективность реализации системы управления лифтом – 1606/1545=1,04.

Итоговая эффективность. Если описание семантики будет реализовано с той же эффективностью, что и общее описание, то эффективность всего решения существенно не измениться.

Вывод: семантический разрыв между текстом исходного описания и текстом программы отсутствует.

Заключение

1. Предложена модель семантически замкнутой онтологии, состоящая из понятийной структуры предметной области и специализированной семиотической системы.

2. Разработана методология понятийного анализа, позволяющая получить понятийную структуру предметной области и выполнить ее верификацию.

3. Решена задача определения семантики формальных систем на основе метода семантической индукции.

4. Разработана технология контекстной обработки знаний, устраняющая семантический разрыв между содержательными представлениями относительно предметной области и средствами их формальной спецификации.

5. Использование контекстной технологии позволяет повысить надежность и качество информационных систем, основанных на обработке знаний.

Понятийный и объектный анализ[1]

Сущность (единичное понятие): – обладает уникальностью; – различается признаками; – выражает смысл.	**Объект**: – обладает идентичностью; – имеет состояние; – проявляет поведение.
Признак (элементарное понятие): отличает одну сущность от другой; имеет имя, домен и семантическую роль	**Свойство** (атрибут): принимает различные значения и характеризует состояние объекта.
Понятие: множество сущностей, образованное на основе абстрагирования.	**Класс**: множество объектов, имеющих общую структуру и общее поведение.
Спецификация понятия: – имя (уникальность), схема (признаки); – интенсионал (содержание); – экстенсионал (состав).	**Спецификация класса**: – имя (идентичность); – свойства (состояние); – методы (поведение).
Абстрагирование понятий: – обобщение (есть некоторый); – агрегация (есть часть); – ассоциация (есть участник); – типизация (есть экземпляр).	**Взаимосвязь классов**: – общее и частное (наследование); – целое и часть (агрегация); – зависимость (ассоциация).

[1] Дополнительный слайд, не демонстрировался.

Понятийный и объектный анализ[1]

Обобщение (расширенная типизация): объединение сущностей дифференцируемых понятий.	**Наследование**: наследуемый класс повторяет структуру и поведение базового класса.
Агрегация: соединение сущностей интегрируемых понятий.	**Агрегация**: отношения целого и части, приводящие к иерархии объектов.
Ассоциация (ограниченная агрегация): связывание сущностей интегрируемых понятий.	**Ассоциация**: зависимость классов, обеспечивающая переход между объектами этих классов.
Типизация: идентификация сущностей дифференцируемых понятий.	**Виртуальные классы**: объект базового класса замещается объектами различных классов.
Понятийная структура: множество понятий с отображениями абстрагирования.	**Диаграммы**: иерархия классов, диаграммы состояний и поведения объектов.

[1] Дополнительный слайд, не демонстрировался.

Грамматическая модель[3]

Грамматическая модель определения семантики языков основана на добавлении расширений к грамматике, определяющей этот язык.

В ***императивном*** (операционном)[1] методе семантика конструкций языка определяется в виде команд абстрактной машины, а смысл выполняемых действий выражается в изменении состояния этой машины.

В методе ***атрибутных грамматик*** [2] с каждым понятием языка связывается атрибут, а правила грамматики дополняется выражениями, позволяющими вычислить значения атрибутов до и после применения этих правил.

---

[1] Specification Case Studies in RAISE / Ed. Van H.D., George C., Janowski T., Moore R. // In Formal Approaches to Computing and Information Technology. Springer, 2002.

[2] Кнут Д. Семантика контекстно-свободных языков // В сб.: Семантика языков программирования. М.: Мир, 1980.

[3] Дополнительный слайд, не демонстрировался.

Аппликативная модель[3]

Аппликативная модель основана на конструировании определений функций, которые связываются с каждой распознанной конструкцией языка.

Аксиоматический[1] (дедуктивный, продукционный) метод базируется на исчислении предикатов, где результат вычисления описывается через взаимосвязь переменных до и после применения конструкций языка.

Денотационный[2] (математический) метод основывается на функциональных вычислениях, в которых языковые конструкции отображаются в однозначные математические объекты, выражающие семантику этих конструкций.

---

[1] Hoare C. A. An axiomatic basis for computer programming // Communications ACM. 1969. Vol. 12, No 10. PP. 576-583.

[2] Stoy J. E. Denotational semantics: the Scott-Strachey approach to programming language theory. MIT Press. 1977.

[3] Дополнительный слайд, не демонстрировался.

Модель спецификаций[3]

Модель спецификаций основана на определении отношений между конструкциями языка. Если некоторый текст реализует эти отношения, то он считается семантически корректным.

Метод ***спецификаций***[1] основан на задании аксиом, которые должны выполнятся для семантически корректной программы и для каждой конструкции языка определяются выражения, имеющие место при применении этой конструкции.

Алгебраический[2] метод основан на отображении конструкций языка в объекты, описываемые посредством аппарата многоосновных алгебр (таблицы и записи баз данных), а смысл текста определяется объектами, полученными при анализе текста.

---

[1] Пратт Т., Зелковиц М. Языки программирования: разработка и реализация. СПб.: Питер, 2002.

[2] Замулин А. В. Алгебраическая семантика императивного языка программирования // Программирование. 2003. № 6. С. 51-64

[3] Дополнительный слайд, не демонстрировался.

Атрибутные грамматики[1]

```
(Number) Integer ()
    ″[0-9]″ `digit` { digit }
    ″[0-9]″ `digit` Integer `int` { int * 10 + digit }
(Float) Fraction ()
    ″[0-9]″ `digit` { digit }
    Fraction `frac` ″[0-9]″ `digit` { digit + frac / 10 }
() Fixed (Integer Fraction)
    Integer `int` '.' Fraction `frac` { int + frac / 10 }
```

В отличие от атрибутных грамматик, не имеющих средств задания ***порядка вычисления атрибутов***, последовательность вычисления сущностей в примере определена выразительными средствами самой модели.

Для правильного вычисления ***синтезируемого*** атрибута int понятие Integer определено как подлежащее грамматическому разбору слева направо. Для вычисления ***наследуемого*** атрибута frac понятие Fraction подвергается разбору справа налево.

[1] Дополнительный слайд, не демонстрировался.

Система управления[1]

Система управления лифтом – параллельный процесс в многозадачной операционной системе.

```
() Лифт ()    \#  (дополнение)
      'лифт'
            {
                  \# Описания функционирования
                  \# лифта
            }

() ()
      'Запустить' Лифт
            {
                  \# Создать параллельный процесс и
                  \# передать ей для исполнения императив
                  \# предложения 'лифт'
            }
```

Дополним модель путем переноса ситуационного описания (см. 35) в прагматику предложения понятия Лифт.

Для запуска системы управления определим предложение, инициирующее создание параллельного процесса.

```
< Запустить лифт >
```

[1] Дополнительный слайд, не демонстрировался.