OSTIS-2011
(Open Semantic Technologies for Intelligent Systems)

УДК 004.822:514

МНОГОУРОВНЕВАЯ ОНТОЛОГИЧЕСКАЯ СИСТЕМА ДЛЯ ПЛАНИРОВАНИЯ РЕШЕНИЙ ПРИКЛАДНЫХ ЗАДАЧ

О.А. Невзорова (*onevzoro@gmail.com*)
Научно-исследовательский институт "Прикладная семиотика" АН РТ,
г. Казань, Россия

В.Н. Невзоров (*nevzorov@mi.ru)*
Казанский государственный технический университет им. А.Н. Туполева,
г. Казань, Россия

В статье структура многоуровневой онтологической системы, используемой для планирования прикладных лингвистических задач в системе «OntoIntegrator». Рассматриваются представления уровней онтологии задач и онтологии моделей. Обсуждается решение прикладной задачи «Поиск определений в математических текстах».

Ключевые слова: онтологическая система, проектирование прикладных лингвистических задач

Введение

Система "OntoIntegrator" является онтолингвистической научно-исследовательской инструментальной средой для решения прикладных задач, связанных с автоматической обработкой текстов. В системе последовательно реализуется онтолингвистический подход [Невзорова, 2007], интегрирующий следующие основные процессы:

- проектирование решения задач на основе онтологических моделей;
- глубокий лингвистический анализ;
- представление знаний в онтологических моделях и в специализированных базах знаний;
- модульность и наращиваемость инструментальных средств;
- адаптация лингвистических технологий к новым предметным областям.

Система "OntoIntegrator" ориентирована на решение сложных задач обработки текстов в семантическом пространстве знаний, представленных как система онтологических моделей. С одной стороны система онтологических моделей структурирует семантическое пространство, с другой стороны – управляет решением прикладных лингвистических задач. В статье будут рассмотрены концептуальная модель построения иерархической системы онтологических моделей и схемы решения прикладных лингвистических задач в рамках развиваемого онтолингвистического подхода.

1. Организация системы онтологических моделей

Система "OntoIntegrator" содержит в своем составе следующие функциональные подсистемы:

- подсистема "Интегратор";
- подсистема проектирования онтологий "OntoEditor+";
- подсистема "Анализатор текста";
- подсистема ведения внешних лингвистических ресурсов;

- подсистема онтологических моделей.

Подсистема проектирования онтологий "OntoEditor+" [Невзорова, 2006] обеспечивает основные табличные функции работы с онтологией (добавление, изменение, удаление записей; автоматическая коррекция записей; ведение нескольких онтологий, в том числе смешанных, т.е. с общими списками типов отношений, классов, текстовых эквивалентов и др.; импорт онтологий различных форматов данных; фильтрация онтологии; ведение автоматической статистики по объектам онтологии; поиск цепочек отношений и др.). Функции модуля визуализации поддерживают различные графические режимы системы, в том числе графический режим проектирования онтологий.

Подсистема "Анализатор текста" включает лингвистический инструментарий, предназначенный для решения задач морфологического анализа, онтологической разметки, разрешения многозначности, сегментации и прикладного лингвистического моделирования.

Подсистема ведения внешних лингвистических ресурсов поддерживает ведение основных лингвистических ресурсов, в составе которых выделяются размеченный грамматический словарь и ряд специализированных лингвистических баз данных.

Подсистема «Интегратор» обеспечивает интеграционную основу решения прикладной лингвистической задачи и управление построением решения прикладной задачи.

Процесс решения прикладной лингвистической задачи в системе "OntoIntegrator" реализуется под управлением системы онтологических моделей. Система онтологических моделей включает различные типы онтологий: прикладные онтологии, онтологию моделей и онтологию задач. Структура системы онтологических моделей представлена на рис. 1.

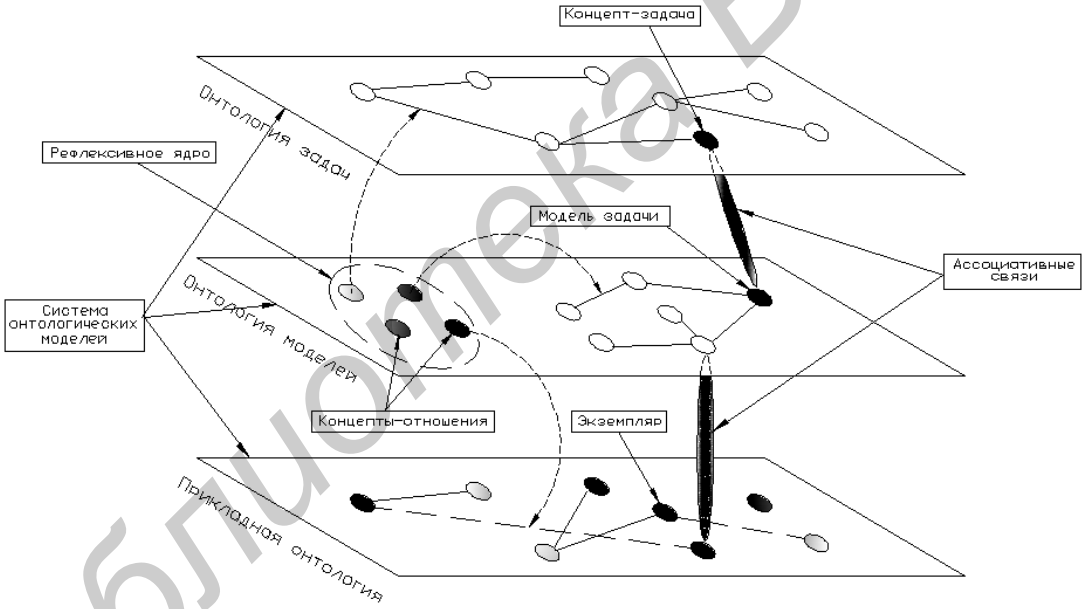

Рисунок 1 - Структура системы онтологических моделей

С точки зрения структурной организации система онтологических моделей представляет собой трехкомпонентную ассоциативную систему. Компонентами системы являются семантические сети (онтологические подсистемы): онтология задач, онтология моделей и прикладная онтология. В системе допускается интерпретация прикладной онтологии как совокупности онтологий разных проблемных областей (и, соответственно, разных семантических интерпретаций): внешних, подключаемых пользователем, и внутренних, встроенных в систему "OntoIntegrator" (с возможностью пополнения, редактирования и вычислительной поддержкой) с целью решения широкого круга прикладных задач. Примерами встроенных онтологий являются онтология метрик, онтология маркеров для разметки выходного текста по результатам решения прикладной задачи и т.д.

Формально, трехкомпонентная ассоциативная система есть структура вида

$S = (\Psi, \Sigma, \Omega)$, *где*

Ψ – *онтология задач,* Σ – *онтолия моделей,* Ω – *прикладная онтология*

Онтология задач $\Psi = (\Psi_K, \Psi_R, Z_\Psi)$ есть семантическая сеть, в которой Ψ_K – множество концептов; Ψ_R - множество отношений, в общем случае n-местных $\Psi_R \subseteq \Psi_K^n$; Z_Ψ - множество функций интерпретации.

Онтология моделей $\Sigma = (\Sigma_K, \Sigma_R, H_R, Z_\Sigma)$ есть семантическая сеть**,** в которой Σ_K - множество концептов; Σ_R - множество отношений, в общем случае n-местных $\Sigma_R \subseteq \Sigma_K^n$; Z_Σ - множество функций интерпретации, H_R - рефлексивное ядро онтологии моделей.

Рефлексивное ядро $H_R = (\Psi_R, \Sigma_R, \Omega_R)$, $H_R \subset \Sigma$ как выделенная составляющая онтологии моделей играет важнейшую, системообразующую роль в построении онтологической системы. Рефлексивное ядро содержит ссылки на все типы отношений в онтологической системе и эта информация представлена как внутренняя модель в онтологии моделей. Другими словами, в онтологической системе представлено знание о структуре системы (множестве всех отношений), организованное как внутреннее знание системы. Рефлексивное ядро может настраиваться (переопределяться) пользователем системы "OntoIntegrator" на основе собственной интерпретации концептов онтологии моделей.

Прикладная онтология $\Omega = (\Omega_K, \Omega_R, Z_\Omega)$ есть семантическая сеть, в которой Ω_K - множество концептов; Ω_R - множество отношений, в общем случае n-местных $\Omega_R \subseteq \Omega_K^n$; Z_Ω - множество функций интерпретации.

Между компонентами (семантическими сетями) онтологической системы могут быть установлены ассоциативные связи (ассоциативные отношения) *R*, $R \subseteq \Psi \times \Sigma$, $R \subseteq \Sigma \times \Omega$**.** Ассоциативные отношения устанавливаются между подсистемами соответствующих онтологий.

Далее рассмотрим структуру компонент онтологической системы.

Онтология задач, как описано выше, формально определяется как структура вида $\Psi = (\Psi_K, \Psi_R, Z_\Psi)$**,** где Ψ_K - множеств концептов-задач, Ψ_R - множество отношений, Z_Ψ - множество функций интерпретации. Концепт-задача $\psi(A_1, A_2, A_3) \in \Psi_K$ определяется набором атрибутов A_1, A_2, A_3. Атрибут A_1 определяет имя задачи, атрибут A_2 - функциональный тип задачи, атрибут A_3 - дескриптор задачи.

Правила объявления атрибутов задают имена атрибутов, функциональные типы и дескрипторы, а также устанавливаемые по умолчанию значения атрибутов. Имена атрибутов являются символьными данными. Дескриптор задачи вводит текстовое описание задачи (краткая аннотация задачи). Функциональный тип присваивает значение функционального класса задачи из открытого множества допустимых функциональных классов: *Операции (TOperations), Источники (TSources), Приемники (TSinks),* ψ *-Реализации (T*ψ *-Realizations).*

Ниже приведем пример построения экземпляра класса ψ*-Реализации* в виде последовательности концептов задач.

Концепт-задача «Функциональная омонимия текста» включает последовательность концептов-задач K1, K2,.., K9:

K1: TSinks // загрузка исходного текста
K2: TSinks //просмотр исходного текста
K3: TSinks // загрузка файла моделей текста
K4: TSources // лексический анализ текста
K5: : TSinks //просмотр файла моделей текста
K6: TSources // получение статистики по типам омонимии
K7: TSources // получение статистики по частотности омонимов

K8 TSources // получение статистики по частотности словоформ текста
K9: TSinks // просмотр файла моделей текста

Класс ψ-*Реализации* определяет структуру решения задачи в виде последовательности базовых операций и ψ-реализаций. Экземпляр класса ψ-*Реализации* представляет собой исполняемый модуль для решения определенной прикладной задачи. Класс *Операции* содержит расширяемый набор базовых операций, для которых фиксируется конкретная семантика. Классы *Источники* и *Приемники* определяют средства преобразования данных, различающиеся функциональностью. Источники обеспечивают доступ к содержимому и его преобразование. Приемники обеспечивают загрузку содержимого в память (во внутренние структуры данных). Формирование экземпляра класса ψ-*Реализации* происходит на основе специального механизма назначения последовательности операций, реализуемого на основе отношения включения из множества Ψ_R отношений онтологии задач. Отношение включения реализуется с атрибутом "номер следования", метрика отношения используется для передачи параметров между операциями.

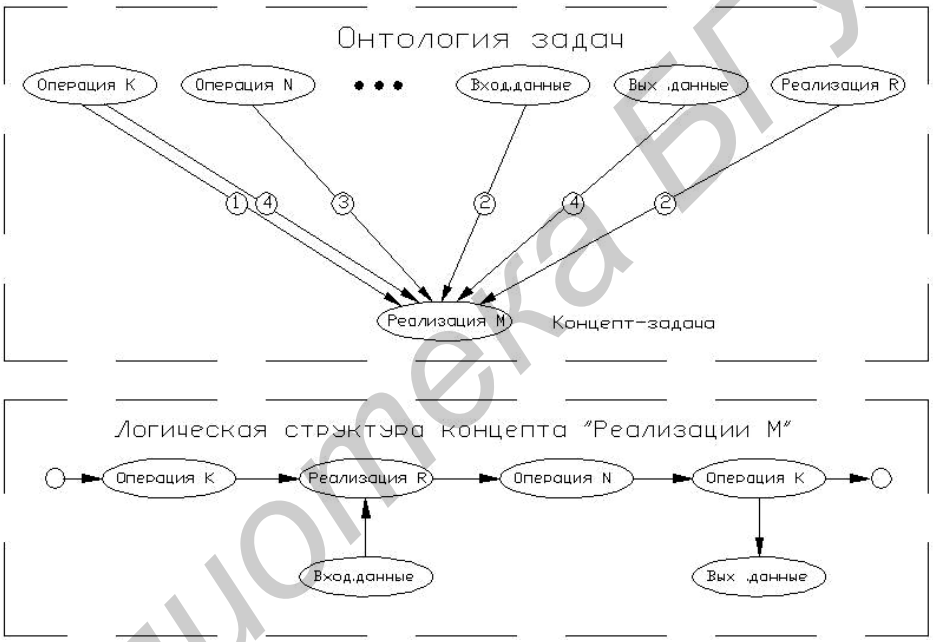

Рисунок 2 - Конструирование концепта-задачи

На рис.2. описан процесс назначения последовательности применяемых операций и реализаций при конструировании новой ψ-*Реализации* и логическая структура созданной ψ-*Реализации*. Конструирование нового концепта-задачи происходит на основе выбранных *операций, реализаций и источников данных*. Формирование логической структуры концепта "*Реализация N"* базируется на механизме приписывания выбранным *операциям* и *реализациям* номеров следования в общей логической последовательности. Так, в рассматриваемом примере "*операции K*" присвоены номера 1 и 4, "*операции N*" - номер 3, "*Реализации R*" и "*Источники*" - номера 2, "*Приемники*" – номер 4.

Планирование логической структуры реализации осуществляется в окне модуля "Конструктор задач" с последующей автоматической генерацией кода новой реализации.

Онтология моделей формально описывается как структура вида $M = \{M_1, M_2, M_3\}$, где M_1 - множество классов моделей, M_2 - множество отношений, M_3 - множество функций

интерпретации. Множество M_1 включает следующие классы: свойство (унарный предикат), отношение (n-местный предикат), референция (ссылочный тип) и m-реализация (библиотечный модуль).

Для каждого класса в онтологии моделей может быть выстроена собственная иерархия (например, иерархия свойств, иерархия отношений, иерархия реализаций). Разработка и использование иерархий зависит от типа лингвистической задачи и является динамическим процессом. Фактически, онтология моделей представляет собой совокупности различных иерархий – иерархий представлений (свойств, отношений) и вычислительных иерархий (класс реализаций). При этом некоторым элементам иерархий представлений могут быть сопоставлены методы в вычислительных иерархиях.

Структурные элементы решения лингвистической задачи, представленной концептом онтологии задач, отображаются на множество структур онтологии моделей. Содержательно, модели служат для реализации решения концепта-задачи и позволяют назначить (интерпретировать) компоненты решения как задачу назначения свойства, либо задачу установления отношения или задачу с известным алгоритмом вычисления. Множество моделей является открытым и пополняется динамически.

Для описания концепта-модели вводятся два атрибута (a_1 – имя класса и a_2 – функциональный тип модели). Конструирование концепта-модели происходит в два этапа: вначале специфицируется тип модели через отдельный механизм спецификации типа, затем создается экземпляр конкретной модели заданного типа. Концепт-модель типа m-реализация может порождаться за счет механизмов агрегации концептов-экземпляров, т.е. экземпляры концептов-моделей связываются по отношению агрегации из множества отношений M_2. Метрика отношения агрегации используется также для передачи значений параметров свойств, агрегированных в ту или иную концепт-модель.

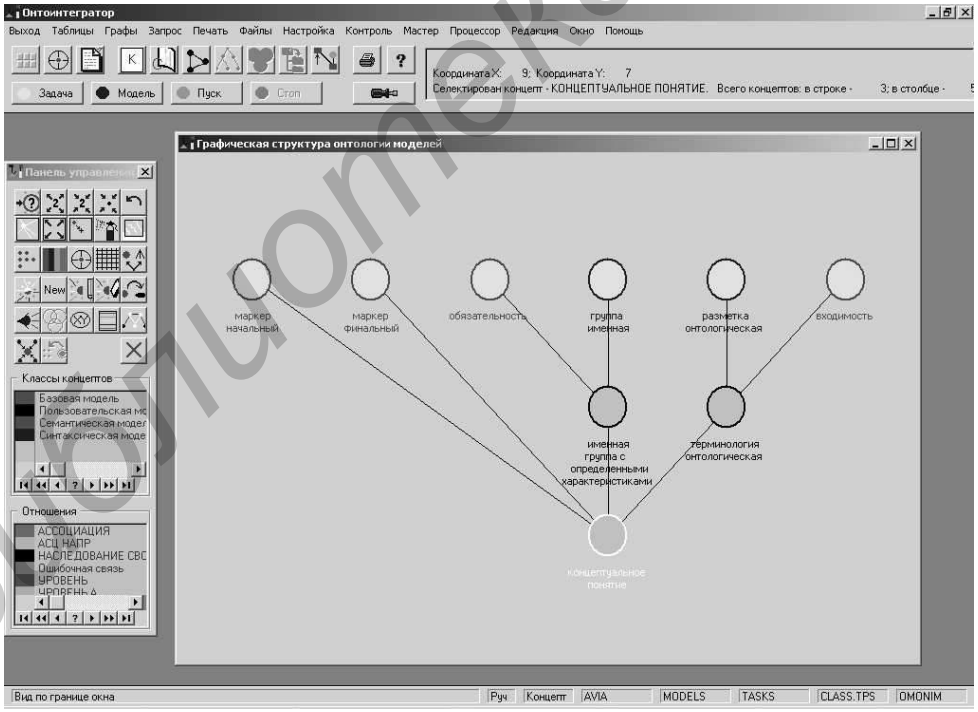

Рисунок 3 - Конструирование концепта-модели "Концептуальное понятие"

Для удобства интерпретации все концепты-модели делятся на условные группы, поддерживаемые развитыми механизмами визуализации:

- базовые модели, обеспечивающие минимальную функциональность системы онтологических моделей;

- синтаксические модели, отображающие Т-модели (модели текста) в онтологии моделей;
- семантические модели, создающие как адекватную прикладной задаче интерпретацию результатов решения, так и реализующие связь между структурой концепта-модели типа m-реализация (порожденной отношением агрегации) и принципиально последовательными комплексами Т-моделей в обрабатываемом тексте;
- пользовательские модели – это концепты-модели, динамически создаваемые пользователем в процессе работы с системой "OntoIntegrator".
- На рис. 3 представлен пример сконструированной структуры концепта-модели "Концептуальное понятие", где концепты "группа именная" и "разметка онтологическая" являются синтаксическими моделями; "маркер начальный", "маркер финальный", "обязательность", "входимость" – семантическими моделями, а все остальные – пользовательскими моделями.

2. Онтологическая модель прикладной задачи «Поиск определений в математических текстах»

Рассмотрим реализацию процесса решения прикладной задачи по поиску определений в математических текстах с финальной их разметкой как результата решения. На вход системы поступает математический документ в текстовом формате, на выходе формируется документ с разметкой математических объектов-определений.

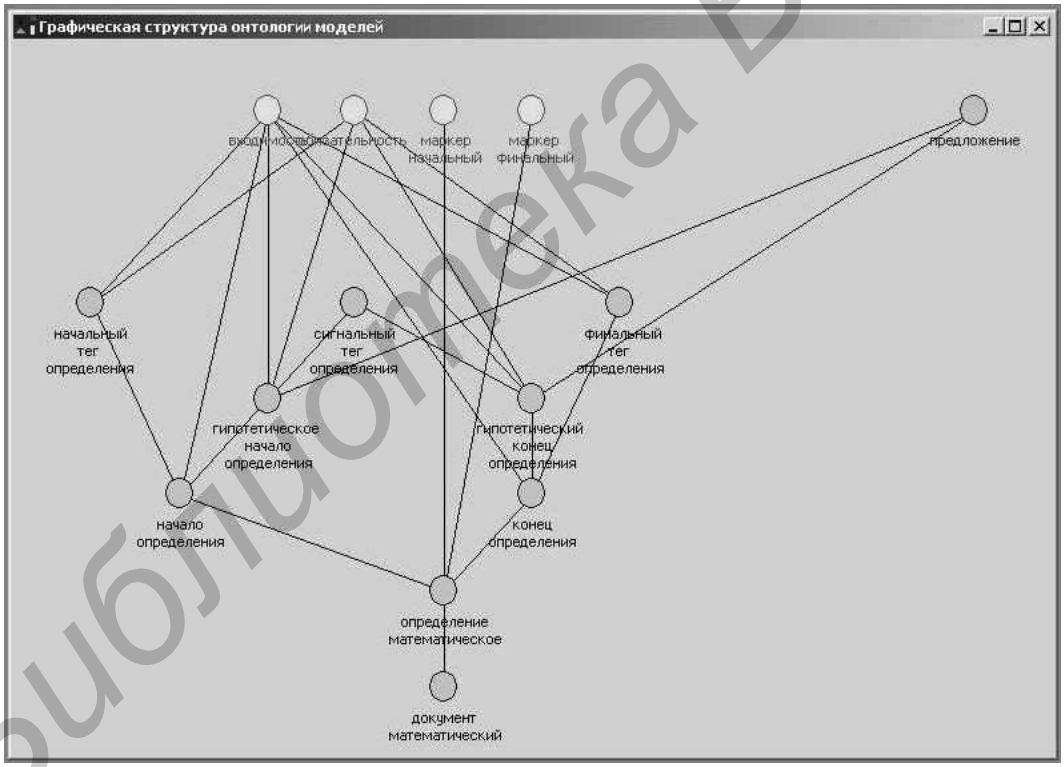

Рисунок 4 - Структура концепта-модели «Определение математическое»

В этом случае работа с системой "OntoIntegrator" включает следующие этапы:
- конструирование концепта-модели "Определение математическое";
- конструирование концепта-задачи "Разметка текста на основе модели задачи";
- построение прикладной онтологии, содержащей требуемые экземпляры классов, входящих в концепт-модель "Определение математическое" и их классификация (установление ассоциативных связей между онтологией моделей и прикладной онтологией) по рис. 2;

- добавление во встроенную онтологию маркеров требуемых маркеров обработанного текста;
- выбор задачи для Процессора системы (в нашем случае – выбор концепта-задачи "Разметка текста на основе модели задачи");
- выбор модели задачи (в нашем случае – выбор концепта-модели "Определение математическое"): установление ассоциативной связи между онтологией моделей и онтологией задач по рис. 2;
- запуск процесса решения задачи.

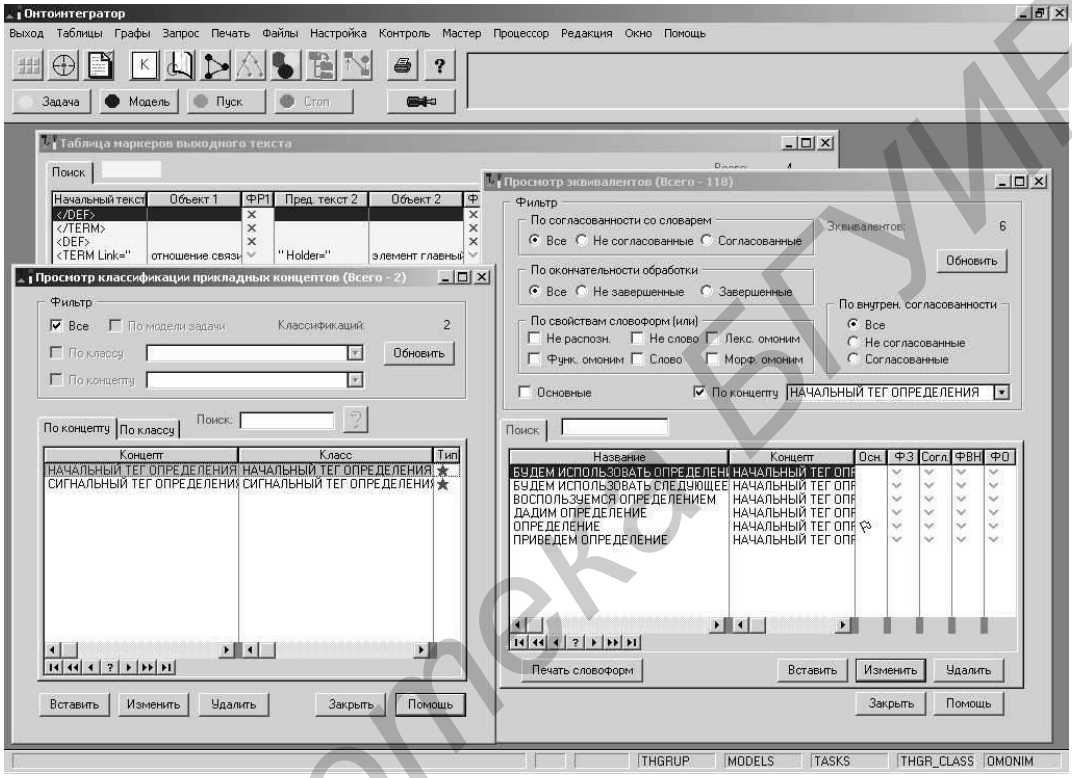

Рисунок 5 - Настройка параметров онтологической модели

На рис. 4 приведен пример структуры концепта-модели "Определение математическое", а на рис. 5 – подготовка онтологии маркеров и прикладной онтологии, а также классификация ее экземпляров.

Тогда после решения поставленной задачи на заданном входном тексте результат работы Процессора может быть проанализирован в окне представлений Т-моделей обработанного текста (см. рис.6) и практически реализован как выходной текст с соответствующей разметкой.

В сводной таблице, представленной на рис. 6 отображены результаты обработки входного текста различными процедурами. Для отображения информации предусмотрен развитый механизм фильтрации результатов. В окне реализованы различные режимы просмотра результатов, в том на числе в отдельных окнах. Просмотр управляется параметрами навигации.

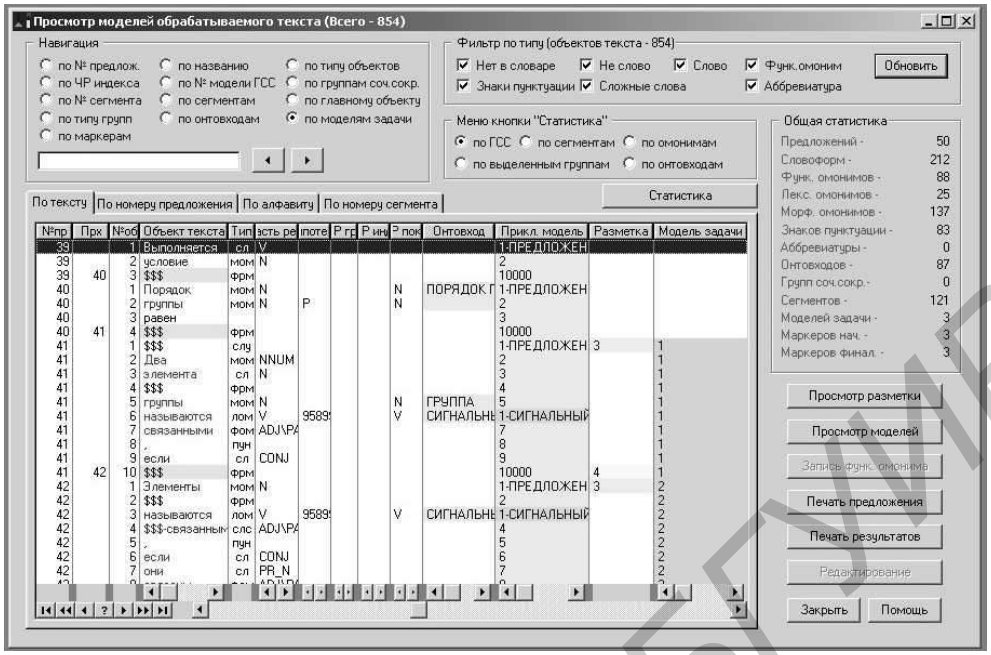

Рисунок 6 - Окно представлений Т-моделей входного текста

Заключение

В статье предложена концепция многоуровневой онтологической системы, реализованная в онтолингвистической научно-исследовательской инструментальной среде «OntoIntegrator». Система «OntoIntegrator» может быть использована для проектирования широкого класса прикладных лингвистических задач. В системе разработана унифицированная технология проектирования прикладных задач в структурах многоуровневой онтологической системы. В настоящее время данная технология применяется для решения различных прикладных задач онтологической разметки входных текстов концептами прикладных онтологий.

Библиографический список

[Невзорова, 2007] Невзорова, О.А. Онтолингвистические системы: технологии взаимодействия с прикладной онтологией /О.А. Невзорова // Ученые записки Казанского государственного университета. Серия физико-математические науки. – Том 149. – Кн. 2. –2007. – С. 105-115.

[Невзорова, 2006] Невзорова, О.А. Инструментальная система визуального проектирования онтологий "OntoEditor+" в лингвистических приложениях /О.А. Невзорова // Вестник КГТУ им. А.Н.Туполева. – № 3. – 2006. – С. 56-60.