СИСТЕМЫ АВТОМАТИЗАЦИИ ПРОЕКТИРОВАНИЯ

УДК 681.518

П.И. Соснин, А.Б. Шамшев

КОМПЛЕКС СРЕДСТВ КОНТРОЛЯ СЕМАНТИКИ ПРОЕКТНЫХ ЗАДАЧ И ПРОЕКТНЫХ РЕШЕНИЙ

Соснин Петр Иванович, *доктор технических наук, профессор, окончил радиотехнический факультет Ульяновского политехнического института. Заведующий кафедрой «Вычислительная техника» Ульяновского государственного технического университета. Имеет статьи и монографии в области автоматизации проектирования и искусственного интеллекта. [e-mail: sosnin@ulstu.ru].*

Шамшев Алексей Борисович, *аспирант, окончил Ульяновский государственный технический университет. Имеет статьи и монографии в области автоматизации проектирования и искусственного интеллекта. [e-mail: alshamshev@yandex.ru].*

Аннотация

Представлен комплекс средств, который обеспечивает контроль семантики, используемой и материализуемой в задачах проектирования, за счет оперативного сравнения с реальностью, стоящей за проектом автоматизированной системы (АС). Контроль осуществляется в версии, получившей название «предикатно-онтологический контроль постановок проектных задач и формулировок проектных решений». Потенциал метода предикатно-онтологического контроля способствует обнаружению семантических ошибок, управлению процессом проектирования, а также развитию онтологии проекта и нормативного естественно-профессионального языка, который следует использовать разработчикам АС.

Метод предикатно-онтологического контроля реализован программно (C#, .Net 3.5, .Net Remouting) в виде набора средств, встроенных в инструментальную среду моделирования WIQA [1] в составе комплекса расширений (плагинов), обслуживающего в проектировании АС решение логико-лингвистических задач.

Ключевые слова: автоматизированное проектирование, контроль семантики, проектные задачи, проектные решения, системная инженерия.

Abstract

The article presents a system of means ensuring the control of semantics used and materialized in design tasks thanks to operational comparison with reality whish underlines a project of computer-aided system. The control is performed in a version whish is called ‘predicate and ontological control of design-task statement and design-solutions definition’. The potential of the technique of predicate and ontological control contributes to detection of semantic errors, control of design process as well as development of project ontology and reference natural and professional language which should be used by developers of computer-aided systems.

The technique of predicate and ontological control is implemented by means of software (C#, .Net 3.5, .Net Remouting) in the form of a set of facilities embedded in tool environment of modeling WIQA [1] within a system of extensions (plug-in) which serves solution of logical and linguistic tasks in design of computer-aided systems.

Key words: computer-aided design, control of semantics, design tasks, design solutions, system engineering.

Введение

Профессионально зрелые разработки сложных автоматизированных систем осуществляют в рамках специальных технологий, используя связную совокупность различных инженерий. Нормы любой технологии овеществляются в виде процессов оперативного согласованного решения технологических задач и предметных задач, за которыми стоят онтологии технологии и предметной области АС, объединение которых в единое целое принято называть «онтологией проекта».

Следовательно, онтология (а значит и семантика) АС и онтология (а значит и семантика) технологии ее создания занимают очень важное место в процессах разработки АС, особенно на концептуальном (понятийном) этапе ее проектирования, когда принципиальные задачи АС, обеспечивающие ее погружение в корпоративную среду, решаются понятийно (концептуально). Как показывает практика разработок АС, цена проектных ошибок чрезвычайно велика и достаточно часто приводит исполняемую работу к провалу [2].

Следовательно, семантику, оперативно используемую и материализуемую в задачах проектирования и принимаемых проектных решениях, следует контролировать, сравнивая ее с реальностью, стоящей за проектом АС, и с отражением этой реальности в онтологии проекта.

В статье представляются проблемы такого контроля и его версия, разработанная авторами и получившая название «предикатно-онтологический контроль постановок проектных задач и формулировок проектных решений».

1 Подход к контролю семантики

Задачи – это специфический класс систем, к работе с которыми применим опыт системной инженерии, в частности стандарт **ГОСТ Р ИСО/МЭК 15288-2005 «Системная инженерия. Процессы жизненного цикла систем»**.

В общем случае «задача» может включать подчиненные задачи и так далее до определенной глубины подчиненности. А значит, к таким задачам, представляя и решая их, логично подходить как к иерархическим системам систем, элементами которых на каждом уровне иерархии являются задачи, каждая из которых материализована в корпоративной среде проектирования как специализированная АС.

В построениях АС, моделирующих задачи, следует помнить, что без использования понятийной деятельности (без рассуждений и действий над ними) многие **работы**, которые приходится проводить с любой задачей, невозможны, и только с помощью рассуждений можно:

- провести анализ задачной ситуации и сформулировать исходную постановку задачи, причем сформулировать так, чтобы ее решение было потенциально достижимо;
- представить задачу в виде структуры из (разнородных) подзадач, связанных в единое целое;
- адаптировать привлекаемый опыт к специфике сложившейся и исследуемой задачной ситуации;
- принять в сложившихся задачах выбора рациональные и, в первую очередь, творческие решения;
- создать руководства по реализации решения задачи, в том числе и в повторных его использованиях (**reuse**).

Важность отмеченных видов (понятийных, концептуальных) работ неоспорима, что указывает на необходимость их контроля с позиций таких принципиальных результатов рассуждений, как «постановка проектной задачи» и «формулировка проектного решения», зарегистрированных в текстовой форме на естественно-профессиональном языке проекта. Контроль семантики таких текстов (Т) переносит свои позитивы на контроль понятийных действий, которые не только приводят к адекватным задачам проектирования, но и предопределяют их возможные решения.

В практике разработок АС применяют различные подходы к контролю семантики задач и решений:

- представление проектных задач в виде их разложения по базису нормативных технологических задач, семантика которых нормативно определена [3]. В частности, представление технологических задач управления требованиями [4], каждое из которых является определенным проектным решением;
- представление проектных задач и их решений в базисе «block and line» схем, чаще всего в базисе UML-диаграмм, семантика которых дополнена семантикой исполняемого проекта [5];
- использование в работах с проектными задачами и проектными решениями специализированных формальных языков, например, языков **AMN** [6],**Community Z Tools** [7], **Estelle** [8] и **SDL** [9].

Характерной чертой существующих средств контроля семантики задач и решений является то, что для контроля текстовых единиц, в том числе и в проверках на их соответствие онтологии проекта в практике разработок АС применяют многократный визуальный просмотр текстов проектировщиками и другими лицами, заинтересованными в успешности проекта.

В контроле семантики текстов, представляющих задачи и решения, авторами предлагается:

- автоматизировать проверки семантики каждого текста на его соответствие онтологии проекта и реальности, осуществляя **машинный перевод** текста с языка проекта на прологоподобный язык семантики;
- возложить **функцию базовой единицы семантического контроля на простое предложение**, экземпляры которого извлекаются из текстов постановок проектных задач и проектных решений и представляются предикатными моделями.

Кроме того, так как онтология любого проекта создается по ходу разработки конкретной АС, то **контроль семантики текстовых единиц на рассогласования с онтологией проекта предлагается совмещать и согласовывать с развитием онтологии проекта**, используя наличие очередного рассогласования как причину развития или коррекции соответствующей составляющей онтологии.

Учитывая, что в предлагаемом подходе к контролю семантики его специфику определяют проверки предикатных моделей (используемых проектировщиками) на их соответствие онтологии, в название контроля введена спецификация «предикатно-онтологического контроля».

2 Детали предикатно-онтологического подхода

2.1 Модель простого предложения

Детализацию подхода начнем с базовой единицы семантического контроля, роль которой возложена на простое предложение (ПП), извлекаемое из проверяемого текста. Контроль такой единицы решено проводить исходя из ее системного представления (модели ПП), приведенного на рисунке 1.

В структуре модели простого предложения выделены:

- «план выражения» (сохраним для него обозначение ПП) как совокупность символьных конструкций на естественно-профессиональном языке проекта АС;

- «план содержания» (введем обозначение РР) для семантики «**Объект** имеет **Свойство**» (вторая типовая версия предложения предназначена для представления семантики «**Объект_1** связан **Отношением** с **Объектом_2**»).

В общем случае конкретное $ПП_i$ (с планом содержания $РР_i$) включено в определенный контекст (текст о реальности + его употребление в коллективной деятельности, которая также относится к реальности). Разумеется, предложение создано и употребляется с определенными целями и обладает необходимыми характеристиками, обеспечивающими выполнение предложением $ПП_i$ возложенных на него функций (в процессе проектирования в инструментальной корпоративной среде $КС_q^и$ конкретной $АС_j$, интенсивно использующей ПО в общем случае, также в корпоративной среде, предметной среде $КС_r^п$).

На рисунке 1 отражено и то, что предложение $ПП_i$ имеет отношения к ценностям, в частности к таким ценностям, как «Язык» и «Речь», и, в общем случае, может оказаться полезным и/или необходимым для развития системы ценностей.

Особо важной характеристикой предложения $ПП_i$ является «истинность», отражающая соответствие употребляемого «плана содержания» реальности. К числу принципиальных характеристик $ПП_i$, отражающих его отношение к реальности, относится и «время», особенно для разработок АС реального времени. На рисунке 1 показано, что в разных ситуациях употребления $ПП_i$ ему могут быть приписаны (и выражены специальными средствами) характеристики количества и качества, в частности необходимые версии модальности (например, «вероятность» или «нечеткость»).

Завершая пункт, представим отношения простого предложения с онтологией проекта, в обозначении которой $О^п(t)$ отражен факт ее развития по ходу проектирования $АС_j$. Возможны следующие варианты результатов проверки $ПП_j$ на соответствие онтологии $О^п(t)$:

- предложение $ПП_j$, включает только варианты употребления понятий, каждый из которых уже определен в состоянии онтологии $О^п(t)$ в текущий момент времени t;

- в сопоставлении $ПП_j$ с онтологией выявлены несоответствия используемых понятий текущему состоянию $О^п(t)$.

Первый вариант результата подтверждает корректность $ПП_j$ только с позиций его соответствия онтологии, оставляя открытым вопрос соответствия $ПП_j$ реальности, но факт соответствия онтологии является существенным аргументом истинностной оценки $ПП_j$. В этом плане контроль семантики обладает потенциалом убеждения.

Второй тип результата указывает либо на ошибки в $ПП_j$, либо на необходимость коррекции $О^п(t)$, возможно в формах развития онтологии до более «богатого» состояния. А значит, контроль семантики обладает управляющим потенциалом.

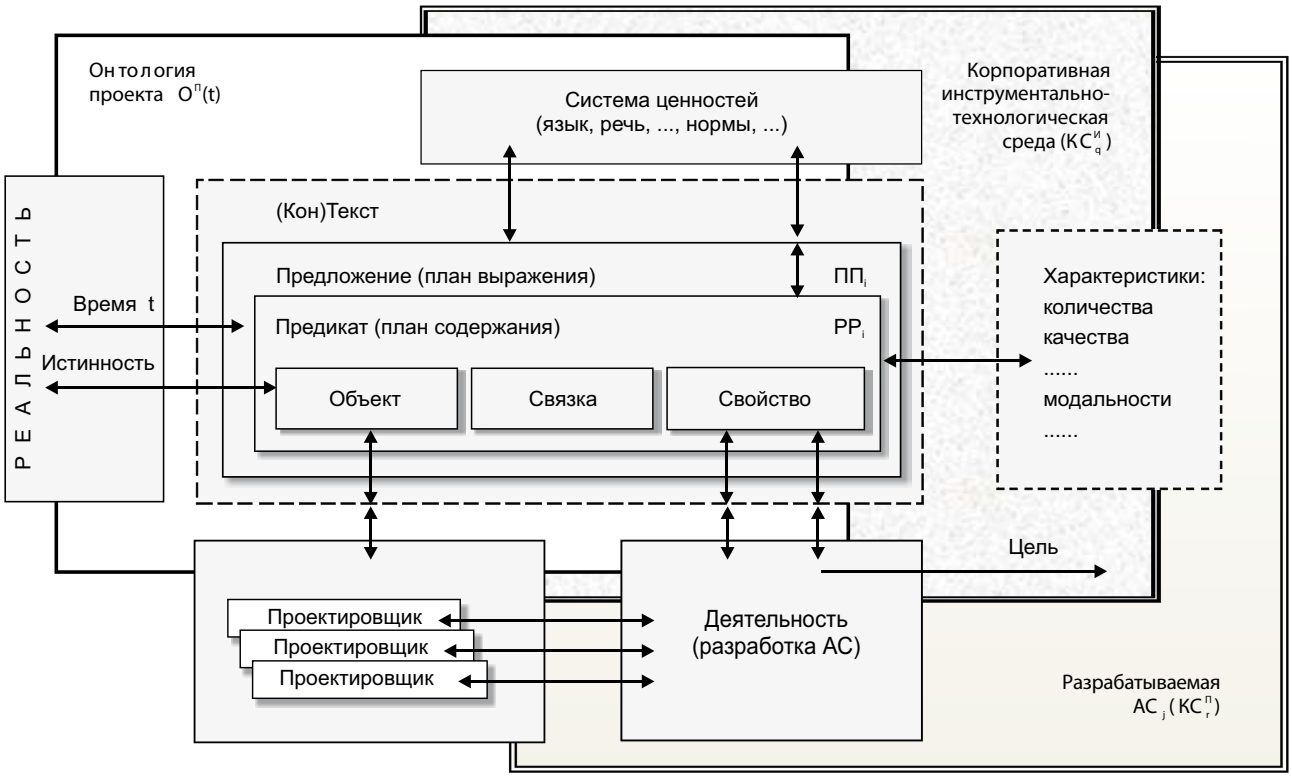

Рис. 1. Модель простого предложения как системы

2.2 Структура семантики

Совокупность позитивных эффектов от контроля онтологической семантики, включающая подтверждение корректности, воздействие на убежденность и управляющий потенциал, привели к решению дополнить контроль на соответствие онтологии проверками других семантических составляющих простого предложения. Принято решение связать дополнительные составляющие семантики с отношениями предложения $ПП_j$ с его окружением, включенным в модель $ПП_j$ как системы (рис. 2).

При таком подходе к анализу и контролю семантики, результат проверок **$S(ПП_{ijk})$** рационально представить выражением:

$$S(ПП_{ijk}) = S_0(ПП_{ijk}) + \sum \Delta S_m(ПП_{ijk}),$$

в котором знак «+» отражает не более чем аддитивный характер общего результата. Отметим, что в множество семантических составляющих, которое открыто для дополнений, включены:

- $S_0(ПП_{ijk})$ – предикатная модель $РР_{ijk}$, прошедшая проверку на соответствие онтологии;
- $\Delta S_1(ПП_{ijk})$ – синтаксемные характеристики [10] предложения $ПП_{ijk}$;
- $\Delta S_2(ПП_{ijk})$ – вероятностные характеристики $ПП_{ijk}$;
- $\Delta S_3(ПП_{ijk})$ – характеристики нечеткости свойства или отношения, названного в $ПП_{ijk}$;
- $\Delta S_4(ПП_{ijk})$ – причина управляющего воздействия на процесс проектирования;
- $\Delta S_5(ПП_{ijk})$ – вариант интерпретации проектировщиком содержания $ПП_{ijk}$.

Реальность семантических анализа и проверок такова, что «вычисления» семантических составляющих могут быть прерваны из-за различных объективных причин (употреблено новое понятие или вариант употребления понятия, непонимание, неполнота знаний, преждевременность предикации и т. д.). Часть из этих причин приводит к вопросам, на которые следует найти ответ или построить его (развитие онтологии, управление процессом).

3 Метод предикатно-онтологического контроля

3.1 Схема метода

Проведенная детализация подхода к контролю семантики достаточна для представления его реализации в виде «метода предикатно-онтологического контроля постановок проектных задач и формулировок проектных решений», элементы теории которого приведены выше.

Стратегически потоки работ метода нацелены на машинный перевод исследуемых текстов, записанных на естественно-профессиональном языке проекта $L^{П}$, на прологоподобный язык семантики, что обобщенно отражено на рисунке 3.

Предикатно-онтологический контроль осуществляется в той части схемы перевода, которая выделена штриховым овалом, и обеспечивает машинный перевод текста на язык простых предикатов. Основные модели, обслуживающие перевод, представлены выше.

Тактически потоки работ метода организованы следующим образом:

1. Действия метода начинаются с выбора текстовой единицы (в разработанном комплексе из дерева задач инструментально-технологической среды WIQA [11]), семантика которой должна контролироваться.

2. Выбранный текст T_i преобразуется в список предложений $Sp(\{П_{ij}\})$, каждому из которых приписывается второй индекс для последующей циклической обработки.

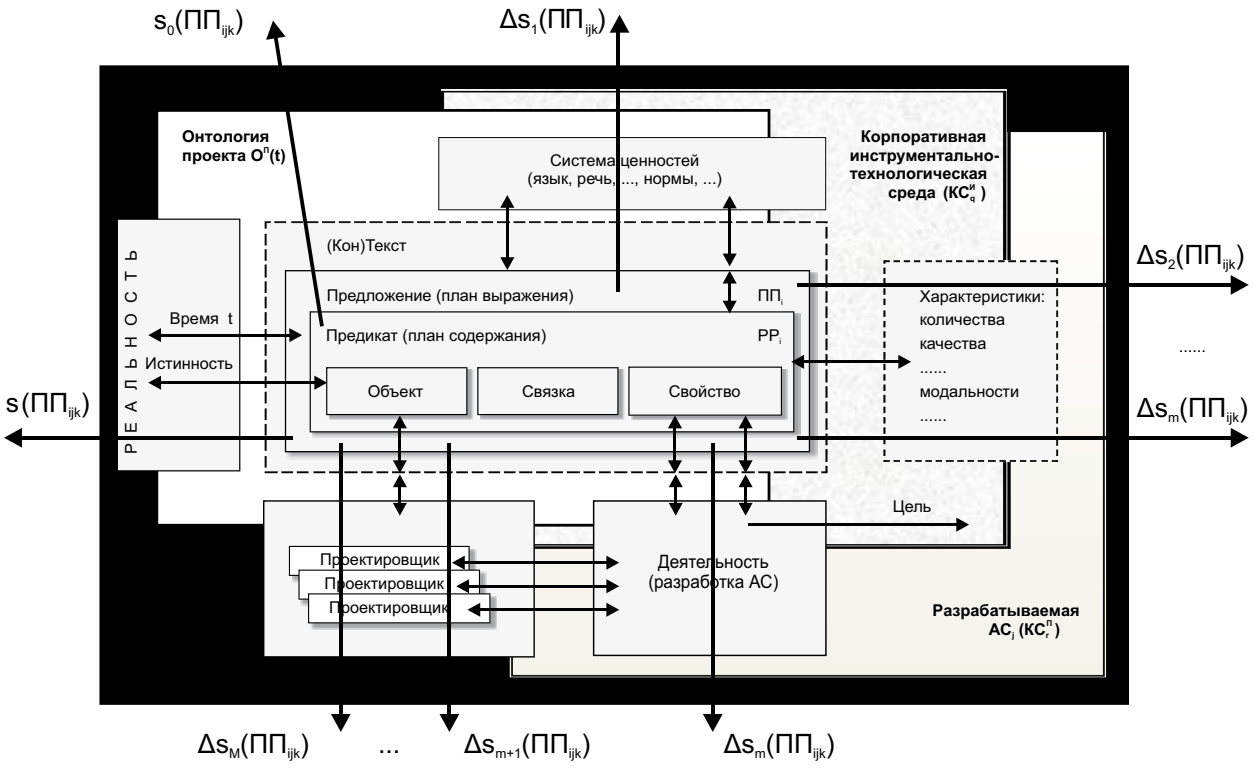

Рис. 2. Структура составляющих семантического контроля

3. Если список Sp({$П_{ij}$}) не пуст, **то** из него выбирается первое в очереди (и исключается из очереди) предложение $П_{ij}$ для его семантического контроля, иначе Перейти к п. 4.

3.1 Для предложения $П_{ij}$ осуществляется его псевдофизическое моделирование, в результате которого формируется список Sp({$П_{ijk}$}) простых предложений $П_{ijk}$.

3.2 **Если** список Sp({$П_{ijk}$}) не пуст, **то** из него выбирается первое в очереди предложение $П_{ijk}$ для его семантического контроля, **иначе** Перейти к п. 3.1.

3.2.1 Для предложения $П_{ijk}$ осуществляется его проверка на соответствие онтологии и формируется результат проверки R($П_{ijk}$).

3.2.2 **Если** R($П_{ijk}$) указывает на соответствие онтологии, **то** Перейти к п. 3.2.6.

3.2.3 **Если** R($П_{ijk}$) указывает на рассогласование с онтологией, но ошибки отсутствуют и рассогласование можно снять за счет коррекции онтологии, **то** Перейти к п. 3.2.10.

3.2.4 **Если** R($П_{ijk}$) указывает на ошибки, **то если** Ошибки можно исправить, **то** Исправить ошибки и Перейти к п. 3.2.1, **иначе** Сформулировать вопросы об ошибках и Включить их в список вопросов Sp({Q_p}), а предложение $П_{ijk}$ оставить в списке Sp({$П_{ijk}$}) и Перейти к п. 3.2

3.2.5 **Если** R($П_{ijk}$) указывает на другие причины (непонимание, неполнота знаний, преждевременная предикация), которые препятствуют квалифицировать R($П_{ijk}$), соответствующим онтологии, **то** Перейти к п. 3.2.9.

3.2.6 Провести для $П_{ijk}$ синтаксемный анализ и зарегистрировать его результат SSA($П_{ijk}$).

3.2.7 **Если** в $П_{ijk}$ присутствуют индикаторы вероятностной модальности, **то** провести анализ вероятностной модальности и зарегистрировать его результат PA($П_{ijk}$).

3.2.8 **Если** в $П_{ijk}$ присутствуют индикаторы нечеткости (например, качественный признак), **то** провести анализ нечеткости и зарегистрировать его результат FA($П_{ijk}$).

3.2.9 **Если** результаты анализов SSA($П_{ijk}$), PA($П_{ijk}$) и FA($П_{ijk}$) приводят к вопросам {Q_r}, **то** включить их в список вопросов Sp({Q_r}), а предложение $П_{ijk}$ оставить в списке Sp({$П_{ijk}$}) и Перейти к п. 3.2, **иначе** Предложение $П_{ijk}$ исключить из списка Sp({$П_{ijk}$}) и Включить R($П_{ijk}$) в «Рабочий словарь».

3.2.10 Включить R($П_{ijk}$) в список Sp^0({R($П_{ijk}$)}) заданий на коррекцию и развитие онтологии.

3.2.11 Предложение $П_{ijk}$ оставить в списке Sp({$П_{ijk}$}) и Перейти к п. 3.2.

4. Включить элементы списка вопросов Sp({Q_r}) в дерево задач проекта.

5. Конец.

В соответствии с обобщенными псевдокодами метода в результате обработки входного потока текстовых единиц, поступающих из среды коллективного проектирования WIQA, порождаются информационные потоки:

- список Sp({$П_{ijk}$}), содержащий «задания» на коррекцию и развитие онтологии проекта;
- список вопросов Sp({Q_r}), определяющий «направления» развития проекта;
- множество предложений {$П_{ijk}$}), подтвердивших свою корректность и направленных в статьи «Рабочего словаря» для построения формальных спецификаций проектных задач и проектных решений, материализация которых приведет к созданию АС.

3.2 Проверки на соответствие онтологии

С операционных позиций представим для метода только основные действия проверок простых предложений на соответствие онтологии. Напомним, что в результате псевдофизического моделирования для каждого $П_{ijk}$ формируется его предикатная модель PP_{ijk}, регистрирующая наличие у объекта определенного свойства или существование определенного отношения между объектами. Такая регистрация, использующая формы логики предикатов и регистрирующая наличие свойства, представлена следующим образом:

$$Сл_1,\ Сл_2,\ \ldots,\ Сл_l,\ N_p\ (Сл'_1,\ Сл'_2,\ \ldots,\ Сл'_J,\ N_q),\ Сл_1'',\ Сл_2'',\ \ldots,\ Сл_s'';$$

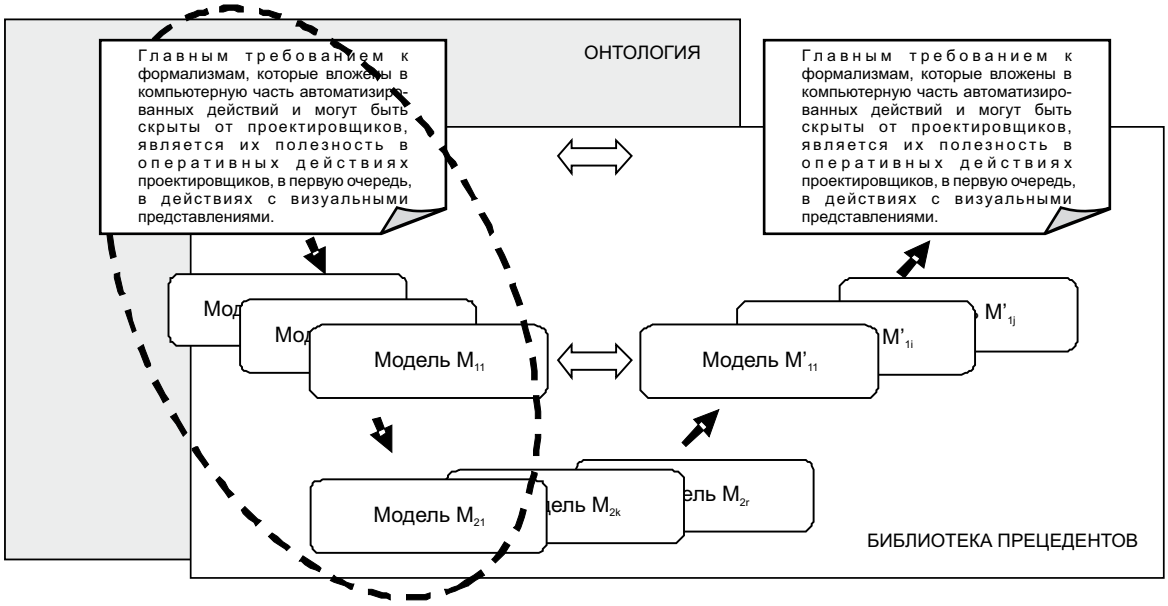

Рис. 3. Имитация машинного перевода

где N_p и N_q – имена понятий, а элемены типа $Сл_i$ обозначает либо одно слово w_i, либо группу слов, одно из которых является главным (обозначим w_i), а остальные зависимыми (обозначим $g_{i1}\ g_{i2}\ldots\ g_{in}$), то есть

$$Сл_i = W_i = g_{i1}\ g_{i2}\ \ldots\ g_{in}\ w_i.$$

В регистрации существования отношения на одну цепочку слов (представляющих дополнение) больше.

При сопоставлении предикатной модели с онтологией осуществляется сравнение варианта употребления понятия с подходящим нормативным употреблением этого понятия, зарегистрированным в онтологии, для представления которого также используется списковая структура (но не только). А значит, к сравнению следует сначала подготовиться, выделив из проверяемой предикатной модели используемый в ней «вариант употребления понятия».

Для приведенной версии предикатной модели в ее содержание введены два варианта употребления понятия:

$$Var^C(N_p) = Сл_1,\ Сл_2,\ \ldots,\ Сл_I,\ N_p =$$
$$g_{11}g_{12}\ldots\ g_{1m}w_1,\ g_{21}g_{22}\ \ldots\ g_{2m},\ w_2,\ \ldots\ ,\ g_{i1}\ g_{i2}\ \ldots\ g_{iM}\ w_K,\ N_p,$$
$$Var^C(N_q) = Сл'_1,\ Сл'_2,\ \ldots,\ Сл'_J,\ N_p = g_{11}\ g_{12}\ \ldots\ g_{1r}\ w_1,\ \ldots,\ g_{i1}\ g_{i2}\ \ldots\ g_{iR}w_L,\ Var(N_p),\ N_q$$

или короче:

$$Var^C(N_p) = Сл_1,\ Сл_2,\ \ldots,\ Сл_I,\ N_p =$$
$$W_1,'\ W_2,\ \ldots\ ,\ W_K,\ N_p$$
$$Var^C(N_q) = W_1,'\ W_2,\ \ldots,\ W_K,\ Var^C(N_p),\ N_q,$$

где первый из вариантов указывает на свойство $Var^C(N_p)$ объекта, определяемого ситуативным вариантом употребления $Var^C(N_q)$ понятия N_q. Каждый из этих вариантов является списком слов, нормализованных с помощью морфологического анализатора, и каждый из них должен пройти через сопоставление с нормативными списками, зарегистрированными в текущем состоянии онтологии проекта как нормативные употребления понятий $Var^H(N_p)$ и $Var^H(N_q)$.

Проверки на соответствие онтологии, представляющие собой сравнения списков, начинаются с понятия о свойстве N_p, после чего проверяется корректность употребления понятия N_q об объекте. Общий случай результата сопоставления приведен на рисунке 4, где надстрочный символ «н» указывает на нормативный список в онтологии.

Сопоставление не составляет труда, и при его выполнении, в общем случае, формируются три промежуточных результата δ_{p1}, σ_p и δ_{p2}, где: δ_{p1} – набор свойств,

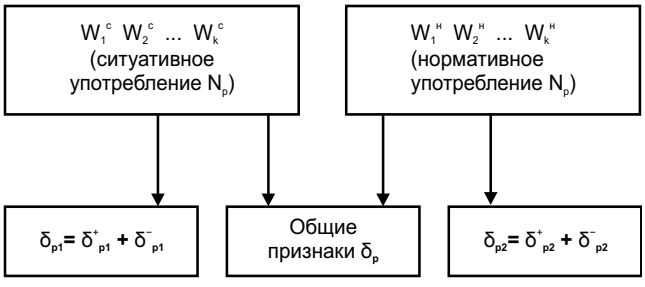

Рис. 4. Сопоставление с онтологией

названных только в ситуативном $Var^C(N_p)$; σ_p – общий для $Var^C{}_p(N_p)$ и $Var^H(N_p)$ набор свойств; δ_{p2} – набор свойств, присутствующий только в нормативе $Var^H(N_p)$. За положительную оценку соответствия онтологии отвечает состав (и содержание) списка общих признаков σ_p, который может быть скорректирован за счет интерактивного визуального анализа списков δ_{p1} и δ_{p2}.

Для того чтобы убедиться в действительных различиях между ситуативным и нормативным (эталонным), в анализе следует ответить на вопросы:

Действительно ли каждое из свойств, входящих в δ_{p1}, присутствует в ситуативном варианте $Var^C(N_p)$?

Действительно ли каждое из свойств, входящих в δ_{p2}, отсутствует в $Var^C(N_p)$?

Предположим, что на каждый из этих вопросов получены ответы, согласно которым рассогласования δ_{pi} разбиваются на части $\delta_{p1} = \delta^+_{p1} + \delta^-_{p1}$ и $\delta_{p2} = \delta^+_{p2} + \delta^-_{p2}$, где надстрочный индекс «+» означает положительный ответ о части свойств, а надстрочный индекс «–» означает отрицательный ответ.

Представим интерпретацию каждого из ответов и возможность их использования:

• рассогласование δ^+_{p1} указывает на отсутствие в текущем состоянии онтологии O(t) подходящего варианта $Var^H(N_p)$ употребления понятия N_p, что может соответствовать следующим ситуациям и реакциям на них:

если это рассогласование оценивается как несущественное, **то** им можно пренебречь, **иначе** следует либо модифицировать **$Var^H(N_p)$**, **если** построение **$Var^H(N_p)$** еще не завершено, **либо** создать новый нормативный вариант употребления понятия **N_p**

• рассогласование δ^-_{p1} позволяет исправить ситуативное употребление, исключив из него свойства, введенные в описание по ошибке;

• рассогласование δ^+_{p2} подобно рассогласованию δ^+_{p1}, причем если оно оценивается как несущественное, то им следует пренебречь, в противном случае, следует создавать новый нормативный вариант;

• рассогласование δ^-_{p2} подсказывает исполнителю предикации те свойства, которые по тем или иным причинам не были включены в исследуемое предложение (то есть δ^-_{p2} способно выполнять функцию подсказки).

Анализ частичных рассогласований, несмотря на свою простоту, демонстрирует принципиальные положительные эффекты, что подтверждает целесообразность его проведения.

3.3 Обнаружение, идентификация и кодирование вопросов

Очень важное место в предикатно-онтологическом контроле занимает формирование списка вопросов $Sp(\{Q_p\})$, обеспечивающего обратную связь процесса контроля семантики с процессом проектирования АС. Основным источником вопросов являются те акты проверок на соответствие онтологии, в которых:

• обнаруживаются ошибки, для устранения которых у проектировщика (по различным причинам) отсутствует необходимая информация;

• проектировщик не может оценить результат сопоставления из-за отсутствия у него необходимой информации;

• проектировщик не может оценить результат сопоставления из-за непонимания (по различным причинам) употребляемого понятия или понятий;

• проектировщик не может оценить результат сопоставления из-за отсутствия у него необходимых знаний.

Для всех названных случаев характерно то, что факт наличия любого из них обнаруживается ситуативно и регистрируется с помощью вопроса, что логично понимать как факт «обнаружения вопроса», требующий выполнения в проекте АС определенных действий. В этом плане «обнаружение вопросов» обладает управляющим потенциалом, а значит управляющим потенциалом обладает и предлагаемый метод контроля семантики.

Если вопрос (пусть Q_p) обнаружен, то он должен быть идентифицирован (определены его тип и другие необходимые свойства), а затем адекватным образом закодирован, то есть представлен определенной текстовой единицей T_j с помощью средств языка L^{Π}, в том числе и той его части, которая уже присутствует в текущем состоянии онтологии. Важными характеристиками идентификации вопроса являются индикаторы его типа, места в дереве задач и персонификации (авторства), а также момент времени включения вопроса в дерево задач.

4 Комплекс средств предикатно-онтологического контроля

Метод предикатно-онтологического контроля реализован программно (C#, .Net 3.5, .Net Remouting) в виде набора средств (рис. 5), встроенных в инструментальную среду WIQA в составе комплекса расширений (плагинов), обслуживающего в проектировании АС решение логико-лингвистических задач и получившего название LINA (Linguistic Into Nominative Activity).

В логико-лингвистических задачах выделены три класса:

• класс задач предикатно-онтологического контроля, основная нагрузка в работах с которыми возложена на лингвистический процессор (область, в которой решаются задачи этого класса интересов, выделена штриховой **bold** линией);

• класс задач построения формальных описаний постановок проектных задач и проектных решений (исследован и доведен до инструментального комплекса «Логический процессор», область выделена точечной линией);

• класс задач построения онтологий проекта и их оперативного использования в средах коллективной разработки АС (область выделена штрих-пунктирной линией).

Выделенные классы задач определены и решаются с использованием общего рабочего поля – рабочего словаря, обеспечивающего как разделение этих классов задач, так и их связность.

Интересы предикатно-онтологического контроля находятся за рамками задач класса 2 и затрагивают только работу с онтологией из-за необходимости проверок на соответствие с онтологией. Отметим, что проверенные предикатные модели регистрируются в словарных статьях рабочего словаря как исходный информационный материал для работы логического процессора, обслуживающего формализацию проектных задач и проектных решений.

В проверках на соответствие онтологии используется только одна словарная статья «Рабочего словаря», получившая название «виртуальной статьи» (нулевой статьи) из-за того, что до сопоставлений любого простого предиката, еще неизвестно, какой статье онтологии он соответствует и соответствует ли вообще. В то же время «Рабочий словарь» выполняет функции «мягкой версии онтологии», повторяя структуру словарных статей.

С позиций информационных потоков «виртуальная статья» выполняет функции транспортного узла, в котором результаты Sp({$П_{ijk}$}), Sp({Q_p}) и {$П_{ijk}$} лингвистической обработки входного потока заданий {T_j} «сортируются» и отправляются для последующего использования в онтологии, управлении процессом проектирования и формализации проектных конструктов (в логическом процессоре).

Заключение

В коллективной разработке АС особо опасным источником ошибок является понятийная деятельность проектировщиков, в результате которой с помощью естественно-профессионального языка создаются постановки проектных задач и формулируются проектные решения. Для снижения количества ошибок и их цены такую деятельность следует автоматизировать, а ее продукты, регистрируемые в форме текстов, необходимо контролировать, в первую очередь, с позиций их соответствия исполняемому проекту, то есть с позиций их семантики.

Представленный выше метод предикатно-онтологического контроля способствует автоматизированному обнаружению семантических ошибок, управлению процессом проектирования и развитию онтологии проекта, а значит и нормативного естественно-профессионального языка, который следует использовать разработчикам АС.

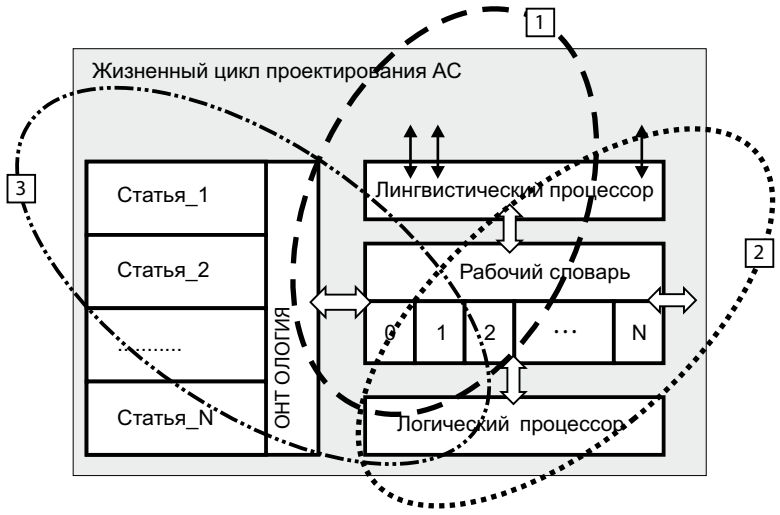

Рис. 5. Логико-лингвистический процессор LINA

Разработка метода включает его теоретическое представление и обоснование, методическое обеспечение на стратегическом, тактическом и оперативном уровнях и программную реализацию в составе комплекса средств LINA, встроенного в вопросно-ответную инструментальную среду. Метод прошел апробацию на текстах постановок проектных задач и проектных решений в ряде разработок АС, в частности при разработке «Системы экспертного мониторинга окружающей обстановки морского судна».

СПИСОК ЛИТЕРАТУРЫ

1. Соснин П. И. Вопросно-ответное моделирование в разработке автоматизированных систем / П. И. Соснин. – Ульяновск : УлГТУ , 2007. – 333 с.

2. Charette R. N. Why software falls. IEEE Spectrum, vol 42, #9, 2005. – pp. 36–43.

3. Rational Unified Process: Сайт разработчика. – Режим доступа: www.ibm.com/software/awdtools/rup.

4. Giorgini P. Requirements Engineering Introduction – Режим доступа: http://dit.unitn.it/tropos/re04/slides/2-RE-intro.pdf.

5. UML. – Режим доступа: http://en.wikipedia.org/wiki/Unified_Modeling_Language.

6. The B-Toolkit.: Сайт разработчика. – Режим доступа: http://www.b-core.com/ONLINEDOC/BToolkit.html.

7. CZT-проект: Официальный сайт. – Режим доступа: http://czt.sourceforge.net/.

8. Budkowski S., Dembinski P., Diaz M. ISO standardized description technique ESTELLE.: Статья. – Режим доступа: http://www.cs.uga.edu/~kochut/Teaching/8060/presentations/ papers/protocol.

9. Grant M. OBJ and Parameterised Programming.: Официальный сайт. – Режим доступа: http://portal.etsi.org/mbs/Languages/SDL/sdl.asp

10. Коммуникативная грамматика русского языка. / Г. А. Золотова [и др.]. – М. : Институт русского языка им. В. В. Виноградова РАН, 2004. – 544 с.

11. Sosnin P. Means of question-answer interaction for collaborative development activity / Hindawi Publishing Corporation, Advances in Human-Computer Interaction, Volume 2009, Article ID 619405, 2009. P. 18 , doi:10.1155/2009/619405.