Онтологические методы проектирования пользовательских интерфейсов

В. Л. Литвинов
Санкт-Петербургский государственный электротехнический университет
«ЛЭТИ» им. В. И. Ульянова (Ленина)
vlad.litvinov61@gmail.com

***Аннотация*. Развитие технологий и появление новых требований к разработке информационных систем приводит к усложнению интерфейсов, связанному не только с увеличением набора функций информационных систем, но и с различными изменяющимися условиями их эксплуатации. В данной работе рассмотрены концепции расширяемости инструментария для проектирования и реализации пользовательского интерфейса в рамках онтологического подхода. Представлен анализ онтологического подхода к автоматизации проектирования пользовательского интерфейса. Предложена методология генерации программного кода пользовательского интерфейса по его проекту.**

Ключевые слова: пользовательский интерфейс; онтология предметной области; RDF-модель; SPARQL

Разработка пользовательского интерфейса (ПИ) является трудоемкой задачей. По подсчетам различных специалистов, в среднем она занимает не менее половины времени разработки программного продукта. Для снижения трудоемкости разработки и сопровождения ПИ в настоящее время существуют различные средства автоматизации проектирования и реализации: построители WIMP-интерфейсов, моделеориентированные средства и средства, основанные на онтологическом подходе.

Целью настоящей работы является исследование концепций расширяемости инструментальных средств для проектирования и реализации пользовательского интерфейса в рамках онтологического подхода.

В рамках онтологического подхода проектирование интерфейса заключается в построении его модели, т.е. определения всех компонентов пользовательского интерфейса. Модель интерфейса – это некоторое описание, созданное с помощью структурных и графических редакторов, на базе систем понятий четырех классов, отражающих специфику каждой составляющей модели интерфейса. Так как классы понятий постоянно изменяются, расширяемость средств проектирования предполагает также значительное расширение этих систем понятий с соответствующей модификацией графических и структурных редакторов, управляемых данными системами понятий [1].

Для обеспечения такой расширяемости методов и средств проектирования необходимо:

- представить системы понятий в форме моделей онтологий;
- разработчику интерфейса предоставить графические и структурные редакторы, призванные интерпретировать модели онтологий;
- не фиксировать жестко состав моделей онтологий, а предоставить редакторы моделей онтологий предметной области специалистам по сопровождению инструментария.

Модель ПИ – это декларативное описание, по которому автоматически генерируется программный код пользовательского интерфейса. Она содержит только ту информацию, которая может измениться при изменении требований к интерфейсу или прикладной программе. С каждым компонентом модели интерфейса связывается своя система понятий, а сам компонент модели интерфейса – это информация, представленная в этой системе понятий (таким образом, различные компоненты модели интерфейса определяются различными системами понятий).

Выделяют четыре основных компонента модели интерфейса и, соответственно, четыре класса систем понятий [2].

1. Система понятий (словарь предметной области) пользователя, в терминах которой он осуществляет взаимодействие с прикладной программой. В этой системе понятий выражаются все входные и выходные данные прикладной программы, а также информация для интеллектуальной поддержки действий пользователя.
2. Система понятий, в терминах которой определяются различные типы диалога (система понятий представления информации). Данный класс содержит три типа систем понятий (далее количество таких систем понятий может увеличиться) – это система понятий графического пользовательского интерфейса (основанных на формах или WIMP-интерфейсов), система понятий статических графических сцен и система понятий для формирования текстов. Таким образом, каждая из систем понятий поддерживает проектирование своего типа диалога.

3. Система понятий для определения сценария диалога. Она определяет термины для описания обработчиков событий (действия, выполняемые при возникновении событий, источники событий, вид режимов переходов между окнами, способы выбора экземпляров окна и др.).
4. Система понятий, в терминах которой осуществляется связь между прикладной программой и пользовательским интерфейсом. Она определяет переменные, их типы значений, а также протоколы, с помощью которых осуществляется коммуникация.

Определение интерфейса в рамках онтологического подхода предполагает наличие только той информации, которая может измениться в жизненном цикле информационной системы. Разработчики приложений используют эти знания для создания подходящих интерфейсов для представления данных: разумный набор данных, группирование и последовательность элементов ввода, отображение/скрытие разделов или навигация между страницами. Это знание неявно используется разработчиком и основано на его опыте или других правилах, которые являются неявным знанием. Основная идея данного подхода – включить эти семантические знания в модель, ориентированную на данные, вместе с обработанными данными приложения.

Важным требованием к архитектуре современных инструментальных комплексов является открытость. Для инструментальных средств автоматической генерации программного кода это достигается явным представлением систем понятий в виде онтологий. Разработчику предоставляются сторонние структурные и графические редакторы для формирования компонент пользовательского интерфейса, управляемых онтологиями. Такой подход позволяет изменять компоненты без модификации кода, однако, если модифицирование инструментария требует изменение кода интерфейса, то разработчик может работать с моделью генерации кода, которая описывает соответствия между алгоритмами бизнес-логики и компонентами модели интерфейса. Методология генерации онтологии представлена на рис. 1. На рис. 2 представлена архитектура инструментального комплекса.

Разработка программного комплекса может быть проведена с помощью библиотеки EasyRdf [3]. EasyRdf – это PHP-библиотека, предназначенная для упрощения работой с RDF файлами. После разбора EasyRdf создает граф объектов PHP, который затем можно использовать, чтобы разместить данные на странице. Имеются методы отладки, позволяющие проверить, какие данные доступны во время разработки. Данные обычно загружаются в объект EasyRdf \ Graph из исходных RDF-документов, загружаемых из Интернета через HTTP. Класс EasyRdf \ GraphStore упрощает загрузку и сохранение данных, а также дальнейшую работу с использованием SPARQL. Пример показан на рис. 3.

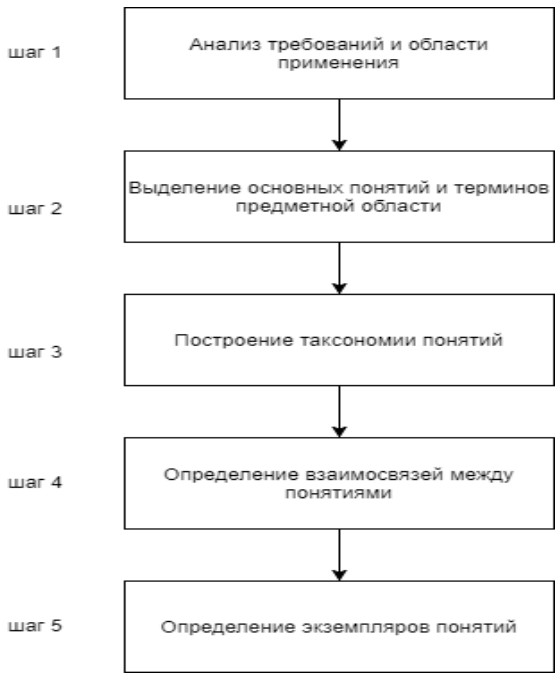

Рис. 1. Обобщенный алгоритм построения онтологий

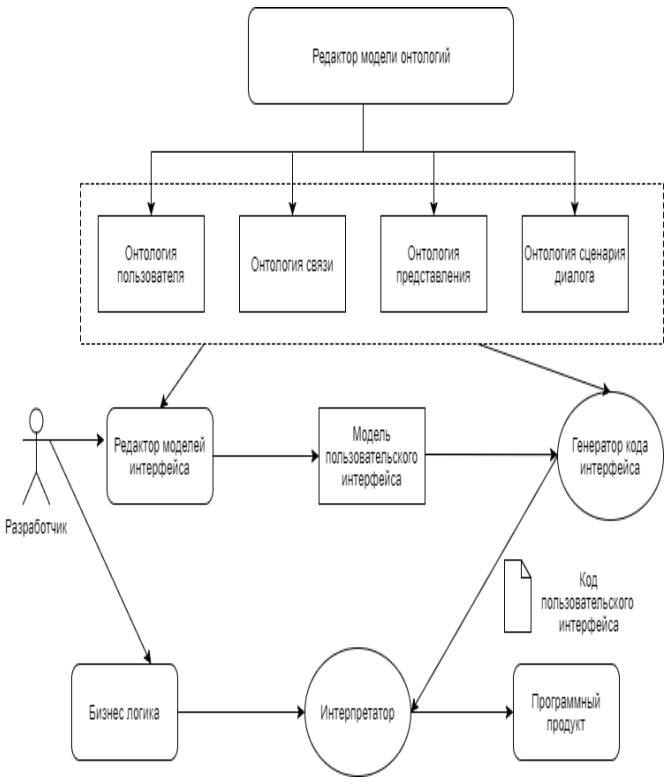

Рис. 2. Архитектура инструментального комплекса

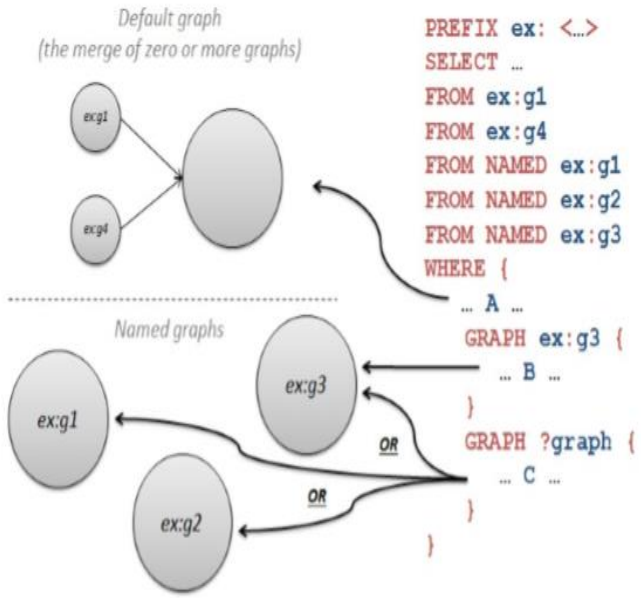

Рис. 3. Пример SPARQL запроса к дереву RDF

Для описания элементов данных (т.е. типов и ограничений вида диапазонов или допустимых значений) необходима типовая и структурная информация, а также значимая временная последовательность экранов.

Поведенческая информация необходима для моделирования динамических, связанных с данными аспектов пользовательского интерфейса для проверки во время выполнения (условия существования, активации элементов / групп, связанных с содержанием других элементов данных в модели, указание для комплексной проверки, операции, связанные с элементами данных и группами, вызванными изменениями входных данных (реакций) или вызванными действиями пользователя.

Эта совокупность информации была сочтена адекватной для получения различных аспектов пользовательского интерфейса.

На основе полученных данных была разработана метамодель, которая включает идентифицированную информацию. Эта модель послужила основой для разработки описаний данных для генерации пользовательского интерфейса (рис. 4).

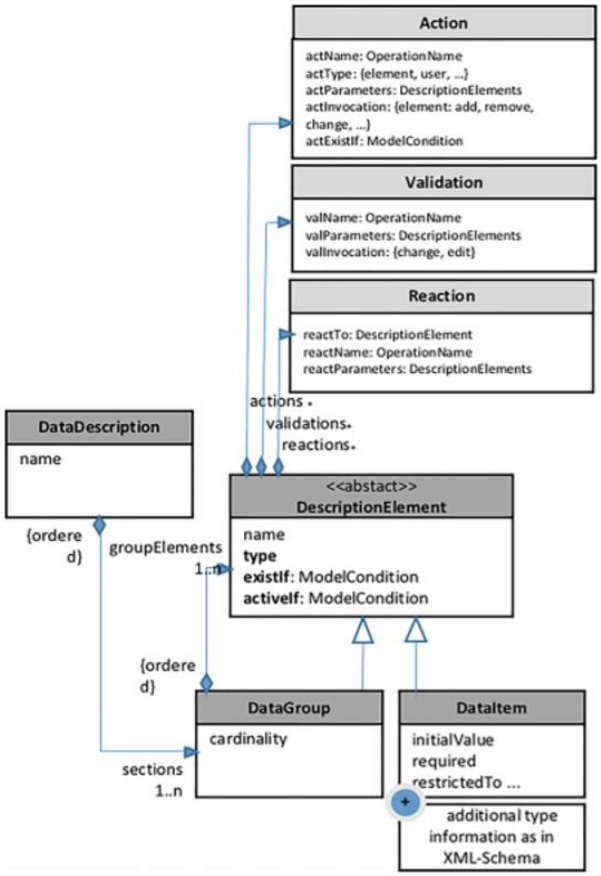

Рис. 4. Метамодель, описанная с помощью UML нотации

Таким образом, в статье описана методология автоматической генерации программного кода пользовательского интерфейса по его проекту, представлен анализ онтологического подхода к автоматизации проектирования пользовательского интерфейса.

Список литературы

[1] Грибова В.В., Клещев А.С. Управление проектированием и реализацией пользовательского интерфейса на основе онтологий // Проблемы управления. 2006, №2. С.58-62.

[2] Автоматизация разработки пользовательских интерфейсов с динамическими данными / Грибова В.В., Черкезишвили Н.Н // Открытые семантические технологии проектирования интеллектуальных систем = Open Semantic Technologies for Intelligent Systems (OSTIS-2011): материалы междунар. научн.-техн. конф., Минск, 10-12 февраля 2011 г. / редкол.: В.В. Голенков (отв. ред.) [и др.]. Минск: БГУИР, 2011. С. 287–293.

[3] EasyRdf Documentation [электронный ресурс]. Режим доступа: https://github.com/njh/easyrdf. – (дата обращения: 25.05.2018).