OSTIS-2013
(Open Semantic Technologies for Intelligent Systems)

УДК 004.822:514

УНИФИЦИРОВАННЫЕ СЕМАНТИЧЕСКИЕ МОДЕЛИ ПОЛЬЗОВАТЕЛЬСКИХ ИНТЕРФЕЙСОВ ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ И ТЕХНОЛОГИЯ ИХ КОМПОНЕНТНОГО ПРОЕКТИРОВАНИЯ

Корончик Д. Н.

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

denis.koronchik@gmail.com

В работе описывается пример использования и реализации модели пользовательского интерфейса интеллектуальной системы. Поэтапно описывается процесс получения запросов от пользователя, и вывод ответов на них. Все описанные в статье принципы и положения использованы при разработке сайта ims.ostis.net.

Ключевые слова: модель пользовательского интерфейса, машина обработки знаний, технология проектирования пользовательских интерфейсов, пользовательский интерфейс интеллектуальной системы.

Введение

В настоящее время существует много подходов, библиотек и инструментов для создания пользовательских интерфейсов. Среди них можно выделить следующие: Qt [Qt, 2012], Adobe Flex [Flex, 2012], Microsoft Silverlight [Silverlight, 2012]. Многие из них ориентируются на компонентное проектирование, но, к сожалению, стыковка компонентов между собой требует высокого уровня профессионализма от разработчика.

В большинстве своем пользовательские интерфейсы современных систем являются сложными, что обусловлено сложностью самих систем. Основной проблемой в них является то, что пользователю, имеющему низкий уровень квалификации, сложно разобраться. Это в свою очередь уменьшает количество пользователей и снижает эффективность их эксплуатации. Одним из важных факторов в этом является то, что все такие пользовательские интерфейсы имеют различные принципы организации. При переходе от одной системы к другой пользователю необходимо затратить некоторое время, чтобы освоить новую систему. Кроме того для их освоения необходимо читать большое количество документации.

В настоящее время разработка пользовательских интерфейсов отнимает большую часть времени затрачиваемого на разработку всей системы. Проектирование пользовательских интерфейсов интеллектуальных систем является более сложной задачей по ряду причин, что делает разработку технологии для их проектирования более актуальной. Сложность разработки таких интерфейсов обусловлена требованиями, которые предъявляются к ним:

- должны отображать различные виды знаний (при прочих равных условиях, чем больше различных видов знаний имеется в базе знаний системы, тем она интеллектуальней);
- должны обеспечивать возможность пользователю ставить перед системой существенно большее количество задач (в том числе и свободно конструируемых);
- должны как можно более четко отражать семантику предметной области в рамках которой происходит общение с пользователем.

Пользовательский интерфейс является единственным способом взаимодействия пользователя с программной системой. Поэтому они должны быть достаточно простыми и легкими в освоении [Поспелов, 1989]. Предлагаемая в данной работе технология направлена на решение описанных выше проблем.

1. Общие положения

В рамках проекта OSTIS ведется разработка семантической технологии проектирования пользовательских интерфейсов интеллектуальных систем. Более подробно с её описанием можно ознакомиться по ссылке [Корончик, 2012]. В данной статье мы рассмотрим важнейшую часть

технологии – унифицированную семантическую модель пользовательского интерфейса. Для этого приведем основные принципы, положенные в эту модель:

- пользовательский интерфейс рассматривается как специализированная интеллектуальная система, которая направлена на получение сообщений от пользователя и вывода ему ответов системы. Другими словами, основной задачей пользовательского интерфейса является перевод сообщения от пользователя, полученного на некотором внешнем языке, на язык понятный системе, а также вывод ответа системы на некотором внешнем языке, понятном пользователю;
- в основе графических интерфейсов лежит SCg-код (Semantic Code graphical – который является одним из возможных способов визуального представления текстов SC-кода) [Голенков и др, 2001]. Объекты, отображаемые на экране с помощью SCg-кода, будем называть sc.g-элементами. Основным принципом, положенным в его основу, является то, что все изображенные на экране объекты (sc.g-элементы), в том числе и элементы управления, являются изображением узлов семантической сети. Другими словами каждому изображенному на экране объекту соответствует узел в семантической сети (базе знаний);
- выделение семантики в пользовательских действиях, с последующим анализом, а также их унификация и четкая типология

Так как пользовательский интерфейс представляет собой интеллектуальную систему, построенную с помощью семантической технологии компонентного проектирования интеллектуальных систем, то он точно так же как и любая другая система строится с использованием компонентов.

Выделены следующие типы компонентов:

- трансляторы с текстов различных внешних языков в тексты SC-кода;
- трансляторы с текстов SC-кода в тексты различных внешних языков;
- компоненты вывода информационных конструкций пользователю;
- компоненты ввода информационных конструкций;

Каждый компонент пользовательского интерфейса состоит из некоторой базы знаний необходимой для его работы и набора sc-агентов [Голенков, 2012].

2. Применение на практике

Опишем применение описываемых моделей на примере разработки сайта **ims.ostis.net**. Основное назначение сайта в текущий момент – это навигация по некоторому объему хранимых знаний. Взаимодействие пользователя с сайтом осуществляется через обмен сообщениями. Общение происходит с помощью двух внешних языков: SCg, SCn [Голенков, 2012]. Эти языки могут быть расширены, так как сам сайт использует описываемый в модели компонентный подход.

Чтобы лучше понять каким образом происходит обмен сообщениями между пользователем и системой, рассмотрим всю цепочку событий, которые происходят от начала формирования запроса пользователя до вывода ответа на экран.

Вопрос пользователя системе формируется следующим образом:

1. пользователь указывает аргументы вопроса. Это делается с помощью левой клавиши мыши с одновременно зажатой клавишей Ctrl (без неё, ссылки происходит инициирование вывода полной семантической окрестности указанного объекта). Пользователь может формировать аргументы в любом порядке, важно, что количество и очередность аргументов зависят от команды, которую пользователь хочет инициировать;
2. пользователь выбирает команду из главного меню, и левым щелчком мыши инициирует её. Если же пользователь удерживает Ctrl при клике по команде в меню, то команда не будет инициирована и попадет в список аргументов. Другими словами пользователь может задавать вопросы и к самим элементам управления.

Предположим, что пользователь указал один аргумент “*объектX*” и инициировал команду “*Поиск всех выходящих дуг из указанного объекта*”. После инициирования команды в базе знаний генерируется конструкция, представленная на рисунке 1. Это еще не сам запрос поиска выходящих дуг, а лишь некоторая первичная информация о том, что пользователь указал некоторые аргументы и инициировал команду. На это событие реагирует sc-агент, который формирует уже сам запрос.

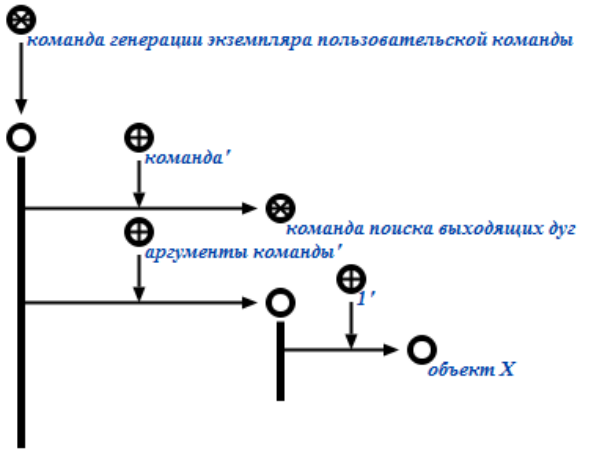

Рисунок 1 – Конструкция, генерируемая пользовательским интерфейсом при инициировании команды

Важно отметить, что команды главного меню – это классы команд, а не конкретные их экземпляры. Они делятся на два типа: атомарные и неатомарные команды. Неатомарные команды – это команды, инициирование которых аналогично запросу декомпозиции самих себя (раскрывающиеся пункты

меню). Атомарные команды в базе знаний описывают некоторую обобщенную формулировку, по которой потом генерируется некоторый запрос. Примером такой команды может быть Команда поиска выходящих дуг, её описание показано на рисунке 2.

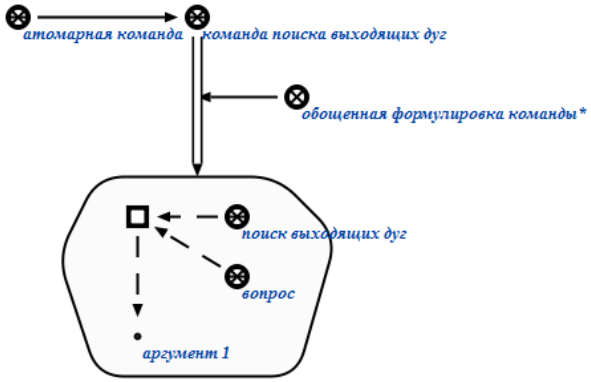

Рисунок 2 – Пример описания атомарной команды

Конкретный запрос пользователя генерируется на основании обобщенной формулировки команды. За этот процесс отвечает sc-агент, условием инициирования которого является появление конструкции представленной на рисунке 1. В результате работы этого sc-агента в базе знаний формируется конструкция, которая представлена на рисунке 3. Важно то, что при переходе на другую платформу или реализацию пользовательского интерфейса, к примеру, не только в web-ориентированном но и в desktop варианте, нам необходимо реализовать лишь генерацию первичной команды (рисунок 1). Так как второй этап уже не работает с внешней средой.

Рисунок 3 – Пример инициированного вопроса

На появление инициированного вопроса в базе знаний (рисунок 3) реагируют агенты машины обработки знаний, которые берутся за поиск ответа на данный вопрос. Параллельно с этим на появление результата генерации экземпляра команды реагирует sc-агент пользовательского интерфейса, который указывает автора вопроса, и внешние языки, на которые должен быть выведен ответ на данный вопрос. В дальнейшем эта информация понадобиться, чтобы принять решение о необходимости трансляции ответа на внешний язык и последующий его вывод пользователю. В результате его работы в базе знаний появляется конструкция, представленная на рисунке 4.

Рисунок 4 – Результат работы sc-агента, который указывает дополнительную информацию о вопросе

Сейчас все что остается пользовательскому интерфейсу, так это дождаться когда ответ вопрос будет обработан. Когда появляется информация о том, что обработка вопроса завершена (вопрос удален из множества инициированных вопросов и помещен во множество завершенных вопросов), снова запускается sc-агент пользовательского интерфейса, который находит автора вопроса. Если автором вопроса является пользователь, то запускается процесс вывода ответа. Сначала в базе знаний генерируется команда, которая инициирует трансляцию ответа на указанные внешние языки (рисунок 5).

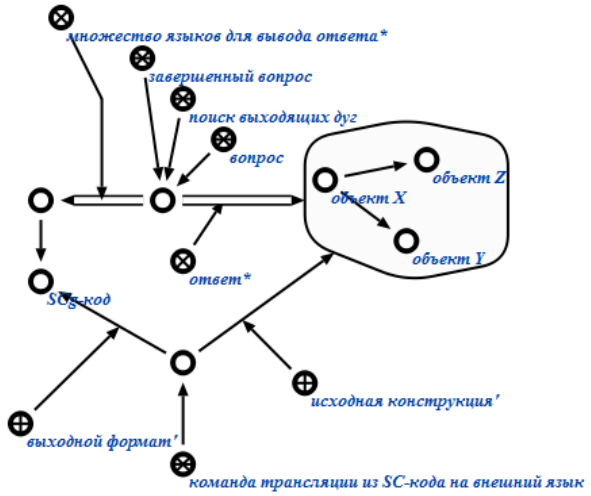

Рисунок 5 – Пример команды трансляции ответа на внешний язык

На появления такой команды реагирует sc-агент компонента трансляции из SC-кода на внешний язык (в нашем случае это компонент трансляции в SCg-код). После завершения его работы в базе знаний появляется конструкция, которая связывает ответ с некоторым файлом, в котором этот ответ записан на внешнем языке (рисунок 6), а команда трансляции помещается во множество завершенных команд. На данном этапе система готова

представить пользователю ответ на его вопрос в том виде в котором того просил пользователь.

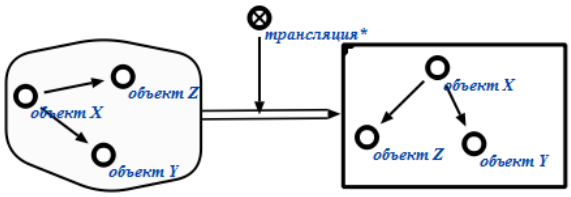

Рисунок 6 – Результат выполнения трансляции

После завершения трансляции запускается “эффекторный” sc-агент. Он выводит содержимое файла, в котором содержится ответ на вопрос, пользователю, который этот вопрос задавал.

В ответе системы, который

Заключение

Разработанный прототип работает в среде интернет. Взаимодействие с пользователем осуществляется по описанным выше принципам. Данный подход показал свою жизнеспособность и продемонстрировал некоторые преимущества.

К примеру, из-за слабой зависимости между подсистемами (зависимость лишь по топологии конструкций в базе знаний), они могут разрабатываться параллельно. Это дает существенное сокращение сроков разработки.

Интерфейс легко расширяем путем добавления новых компонентов, причем все компоненты могут быть добавлены динамически во время работы системы.

Возможность задавать вопросы к элементам управления позволяет существенно сократить время на освоение системы, за счет того, что пользователь может узнавать у системы как пользоваться теми или иными командами.

В процессе разработки системы, разработчики могут использовать уже имеющиеся в библиотеке компоненты, что сокращает сроки разработки, также пополнять библиотеку компонентов новыми компонентами, что позволяет технологии развиваться.

Библиографический список

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

[Голенков, 2012] Графодинамические модели параллельной обработки знаний / В. В. Голенков, Н. А. Гулякина // Материалы международной научно-технической конференции «Открытые семантические технологии проектирования интеллектуальных систем» - Минск, 2012

[Корончик, 2012] 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

[Поспелов, 1989] Поспелов Д.А. Интеллектуальные интерфейсы для ЭВМ новых поколений// Электронная вычислительная техника. Сборник статей. Вып.3. - М.: Радио и связь, 1989. - С.4-20.

[Flex, 2012] Free, open-source framework | Adobe Flex [Электронный ресурс]. – 2012. – Режим доступа: http://www.adobe.com/products/flex.html. - Дата доступа: 05.10.2012

[OSTIS, 2012] Открытая семантическая технология проектирования интеллектуальных систем [Электронный ресурс]. – 2012. - Режим доступа: http://ostis.net. – Дата доступа: 11.10.2012

[Qt, 2012] Qt project [Электронный ресурс]. – 2012 – Режим доступа: http://qt-project.org/. – Дата доступа: 02.10.2012

[Silverlight, 2012] Microsoft Silverlight [Электронный ресурс]. -2012 – Режим доступа: http://www.microsoft.com/silverlight/. – Дата доступа: 03.10.2012

UNIFIED SEMANTIC MODELS OF USER INTERFACE FOR INTELLIGENT SYSTEMS AND TECHNOLOGY FOR THEIR DEVELOP

Koronchik D. N.

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

denis.koronchik@gmail.com

This article describes usage of semantic user interface model in develop process.

Introduction

The development of software engineering environments has had a long and close relationship with the development of advanced user interface technologies. There are many technologies for user interface development. The only way to interact with machine is user interface. It would be impossible to interact with systems without it.

Main Part

The main purposes of semantic user interface technology are: to decrease time of user interface development by using components; lower level of developer qualification; to decrease requirements to user’s qualification; make user interface integration with intelligent system much simpler. The offered technology based on a four principles: unified semantic model lies in a basis of all designed user interfaces; user interfaces designed from components; develop process supports by intelligent tools; technology includes help system, that helps developers in design process.

The offered technology includes: semantic model of user interfaces, components library, design tools, design technique, technique to study design process, help system for technology

Conclusion

Principles and techniques, described in this paper, was used to create site **ims.ostis.net**.