МОДЕЛЬ ВЗАИМОДЕЙСТВИЯ КОМПОНЕНТОВ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ

К. В. Русецкий, А. Ф. Хусаинов
Кафедра интеллектуальных информационных технологий, Белорусский государственный университет информатики и радиоэлектроники
НИИ «Прикладная семиотика» Академии наук Республики Татарстан
Минск, Республика Беларусь; Казань, Российская Федерация
E-mail: {rusetski.k, khusainov.aidar}@gmail.com

В данной работе рассматривается семантическая модель интеграции компонентов пользовательского интерфейса, основанная на их взаимодействии посредством общей базы знаний, хранимой в семантической памяти. В частности, приводится описание языка команд пользовательского интерфейса, обеспечивающего такое взаимодействие.

Введение

Пользовательский интерфейс (ПИ) интеллектуальных систем, разрабатываемых по технологии OSTIS [6] строится по тем же принципам [1], что и сама система. Иными словами, пользовательский интерфейс является специализированной интеллектуальной системой, которая решает задачу обеспечения диалога пользователя с системой.

При построении пользовательского интерфейса используется компонентный подход [4,5]. Использование компонентов при проектировании ПИ позволяет существенно сократить сроки их проектирования за счет многократного использования компонентов в различных системах.

При использовании компонентного подхода возникает проблема организации взаимодействия между ними. Прямое взаимодействие между компонентами, с точки зрения процесса разработки, имеет низкую эффективность. Оно повышает сложность ПИ, что существенно увеличивает сроки разработки. Для решения этой проблемы предлагается использовать модель взаимодействия компонентов пользовательского интерфейса через общую базу знаний.

I. Взаимодействие компонентов пользовательского интерфейса

В предлагаемой модели все компоненты взаимодействуют исключительно через базу знаний, которая хранится в семантической памяти [1,2]. Такой способ организации взаимодействия позволяет упростить интеграцию компонентов в систему за счет снижения количества их взаимосвязей.

Взаимодействие между компонентами в базе знаний строится на основании многоагентного подхода. Каждый компонент ПИ включает в себя одного или несколько агентов [3]. Под агентом будем понимать некоторую программу, которая инициируется при происхождении события в памяти системы или же во внешней среде. Выделено три типа агентов, которые используются при построении ПИ:

- **эффекторные** – агенты, которые реагируют на изменение внутри памяти системы и на основании их, производят некоторые изменения во внешней среде. Они обеспечивают вывод информации;
- **рецепторные** – агенты, которые реагируют на изменения во внешней среде и изменяют состояние памяти системы. Они обеспечивают ввод информации;
- **внутренние** – агенты, которые никак не взаимодействуют с внешней средой, а реагируют на изменения состояния памяти и изменяют состояние памяти системы. Они обеспечивают обработку информации.

Совокупность агентов, которые относятся к перечисленным выше типам, образуют машину обработки знаний (МОЗ) пользовательского интерфейса. Очевидно, что для корректного взаимодействия между агентами, необходимы некоторые правила поведения. Для этого разработан специализированный универсальный семантический язык, который позволяет записывать команды инициирующие агенты ПИ – язык команд пользовательского интерфейса.

II. Язык команд пользовательского интерфейса

Язык команд ПИ используется агентами для организации взаимодействия между ними. Он включает в себя следующие классы команд:

- команда трансляции информационных конструкций с SC-кода [2] на внешние языки и обратно;
- команда создания экземпляра пользовательской команды, на основании обобщенной формулировки;
- команда инициирования вывода информации пользователю;

Каждая команда данного языка - это фрагмент семантической сети, который является условием инициирования соответствующего

агента. Другими словами, это некоторая информационная конструкция, при появлении которой инициируется некоторый агент ПИ. Команда пользовательского интерфейса может принадлежать одному из следующих множеств, которые отражают состояние команды:

- множество инициированных команд. В этом множестве находятся все инициированные команды. Инициированная команда - это команда, которая должна быть выполнена некоторым агентом;
- множество выполняемых команд. В этом множестве находятся команды, которые в текущий момент выполняются некоторым агентом;
- множество прерванных команд. В этом множестве находятся все команды, выполнение которых было прервано по различным причинам;
- множество завершенных команд. В этом множестве находятся все команды, выполнение которых было завершено (не прервано).

Агент, который реализует ту или иную команду, обязан соблюдать правила её перехода в различные состояния. При этом в каждый момент времени команда не может входить в два разных множества состояний. Другими словами, при смене состояния, команда сначала должны быть исключена из предыдущего множества, а затем добавлена в новое. В противном случае поведение агентов ПИ не предсказуемо.

Параметры команды записываются с помощью ролевого отношения ***аргументы команды'***. Результат выполнения, команды связывается со знаком самой команды с помощью ролевого отношения ***результат выполнения команды'***. Результат выполнения команды может присутствовать лишь для завершенных команд.

Команда трансляции информационных конструкций с SC-кода на внешние языки и обратно инициирует агенты, которые транслируют исходную информационную конструкцию в некоторое представление в SC-коде или на внешнем языке. У данной команды два аргумента:

1. знак информационной конструкции, которую необходимо транслировать. Если трансляция производится с внешнего языка представления знаний в SC-код, то первым параметром является sc-ссылка на файл, в котором записана транслируемая информация. В случае с трансляцией из SC-кода, в качестве первого параметра выступает знак структуры, которая содержит все элементы для трансляции;
2. знак формата, с помощью которого должна быть представлена транслируемая информационная конструкция. Если трансляция производится в текст SC-кода, то в качестве этого параметра выступает знак, обозначающий SC-код. В случае трансляции на внешний язык, в качестве аргумента выступает знак, который обозначает некоторый формат (***формат scg, формат png*** и т. д.).

В результате выполнения команды генерируется некоторая информационная конструкция в указанном формате. Исходная конструкция и результат трансляции связываются бинарным отношением ***трансляция sc-текста****, которое всегда исходит от SC-текста.

Команда создания экземпляра пользовательской команды на основании обобщенной формулировки инициирует агентов, которые создают экземпляр пользовательской команды с заранее известными аргументами, её обобщенной формулировки. Данная команда имеет два аргумента:

1. знак обозначающий класс пользовательских команд. Данный класс связан отношением обобщенная формулировка команды' со структурой, которая её содержит;
2. множество аргументов пользовательской команды.

Результатом выполнения команды является экземпляр пользовательской команды с заранее определенными аргументами – создается копия структуры, в которой все переменные узлы заменены на константы, а ключевые узлы специального вида – на аргументы.

1. Голенков, В. В. Графодинамические модели параллельной обработки знаний: принципы построения, реализации и проектирования / В. В. Голенков, Н. А. Гулякина // Материалы междунар. научн.-техн. конф. «Открытые семантические технологии проектирования интеллектуальных систем» (OSTIS-2012). – Минск: БГУИР, 2012. – С. 23–52.
2. Голенков, В. В. Представление и обработка знаний в графодинамических ассоциативных машинах / В. В. Голенков [и др.]. – Мн.: БГУИР, 2001.
3. Шункевич, Д. В. Смысловая модель обработки знаний в интеллектуальных системах / Д. В. Шункевич, К. В. Русецкий // Электроника-инфо. – 2014. – №3. – С. 35-37
4. Корончик, Д. Н. Семантические модели мультимодальных пользовательских интерфейсов и семантическая технология их проектирования / Д. Н. Корончик // Материалы международной научно-технической конференции «Открытые семантические технологии проектирования интеллектуальных систем» (OSTIS-2012). – Минск : БГУИР, 2012. – С. 339-346.
5. Корончик, Д. Н. Унифицированные семантические модели пользовательских интерфейсов интеллектуальных систем и технология их проектирования / Д. Н. Корончик // Материалы международной научно-технической конференции «Открытые семантические технологии проектирования интеллектуальных систем» (OSTIS-2013). – Минск: БГУИР, 2013. – С. 403-406.
6. ims.ostis.net [Электронный ресурс]. – 2016. – Режим доступа: http://ims.ostis.net/. – Дата доступа: 15 сентября 2016 г.