Из приведенных выше графиков можно видеть, что при увеличении числа правил улучшается качество регулирования.

Заключение

В работе была поставлена и решена задача построения нечеткого логического регулятора для стабилизации судна. Найденный регулятор обеспечивает устойчивость замкнутой системы.

ЛИТЕРАТУРА

1. *Неймарк Ю.И.* Динамические модели теории управления // Ю.И. Неймарк, Н.Я. Коган, В.П. Савельев. М.: Наука, 1985.
2. *Zimmermann H.J.* Fuzzy Set Theory And Its Applications// H. J. Zimmermann. Kluwer Academic Publishers , third edition, 1996.
3. *Vaneck T.W.* Fuzzy Guidance Controller for an Autonomous Boat/ T. W. Vaneck. IEEE Control Systems, april 1997.
4. *Гелиг А.Х.* Устойчивость нелинейных систем с неединственным состоянием равновесия// А.Х. Гелиг, Г.А. Леонов, В.А. Якубович. М.: Наука, 1978.

Статья представлена к публикации членом редколлегии М.М. Коганом.

УДК 004.5

© 2005 г. **А.В. Тарасов**
(Институт автоматики и процессов управления ДВО РАН, Владивосток)

МОДЕЛЬ ВЫРАЗИТЕЛЬНЫХ СРЕДСТВ ИНТЕРФЕЙСА ПРИ ЕГО РАЗРАБОТКЕ НА ОСНОВЕ ОНТОЛОГИЙ

В работе представлена модель выразительных средств пользовательского интерфейса, которая является компонентом модели интерфейса в онтологоориентированном подходе при его разработке. Описаны требования к модели, а также методы ее реализации.

Введение

Интерфейс пользователя является неотъемлемым компонентом большинства программных систем. Существующий рынок программных средств предлагает различные инструменты для его разработки.

Широко используемые в настоящее время Interface Builders [1] (по-

строители интерфейса) обладают очевидными недостатками – возможностью использования только программистами, громоздким процедурным кодом, отсутствием средств для проектирования всех составляющих интерфейса. Это стимулировало поиски новых методов разработки интерфейсов.

С середины 90-х гг. стал активно развиваться моделеориентированный подход к разработке интерфейса [2, 3]. Основной идеей подхода является автоматическая генерация интерфейса по декларативным, высокоуровневым моделям его составляющих. В результате значительно уменьшается число процедурных компонент, появляется возможность повторно использовать компоненты моделей. Однако моделеориентированный подход характеризуется следующими недостатками: элементы пользовательского интерфейса не стандартизированы, интерфейсы, разработанные с помощью различных моделеориентированных средств, не совместимы друг с другом, а значит, могут быть повторно использованы только в рамках этих средств.

Предложенный в [4] онтологоориентированный подход к разработке пользовательского интерфейса является продолжением моделеориентированного подхода и ставит своей целью предложить инструментарий, соответствующий требованиям современного этапа к разработке интерфейсов: обеспечение простой модифицируемости, повторной используемости компонентов модели интерфейса, автоматической генерации исполнимого кода по модели интерфейса. Согласно данному подходу модель пользовательского интерфейса состоит из следующих компонентов: модели предметной области, модели выразительных средств интерфейса, модели прикладной программы, модели связей «предметная область – выразительные средства интерфейса» и «предметная область – прикладная программа», а также модели сценария диалога. Данная работа посвящена описанию процесса проектирования и методам реализации одной из составляющих модели интерфейса – модели выразительных средств.

Подход к разработке пользовательского интерфейса на основе онтологий

Основными положениями концепции при разработке интерфейса на основе онтологий являются следующие [4]:

раздельное проектирование интерфейса и прикладной программы;

объединение однородной по содержанию информации в компоненты модели интерфейса, представление каждого компонента модели интерфейса в форме модели онтологии;

автоматическая генерация интерфейса по модели интерфейса;

возможность проектирования интерфейса как локального, так и сетевого приложения.

Однородная по содержанию информация объединяется в компоненты модели интерфейса. При этом модель пользовательского интерфейса должна содержать всю информацию о нем, а также допускать автоматическое построение пользовательского интерфейса по его модели. Модель интерфейса состоит из следующих составляющих: модели предметной области, модели выразительных средств интерфейса, модели прикладной программы, модели связей «предметная область – выразительные средства интерфейса» и «предметная область – прикладная программа», а также модели сценария диалога. Модель предметной области предназначена для обеспечения взаимодействия пользователя и прикладной программы в терминах той предметной области, задачи из которой решает прикладная программа. Модель выразительных средств описывает структуру и свойства интерфейсных элементов, используемых в конкретном интерфейсе. Модель прикладной программы описывает способы взаимодействия с прикладной программой. Модель сценария диалога предназначена для описания процесса взаимодействия пользователя с прикладной программой.

Модель выразительных средств пользовательского интерфейса

Графический интерфейс пользователя – это графическая среда организации взаимодействия пользователя с вычислительной системой для управления поведением вычислительной системы через визуальные элементы управления: окна, списки, кнопки и т.д. В настоящее время разработка пользовательских интерфейсов является самостоятельной областью программирования, в которой уже сформировалась довольно устойчивая система понятий. Эта система понятий отражена в онтологии графического пользовательского пнтерфейса (ОГПИ) [5] и не зависит от конкретной реализации. Если ОГПИ описывает все знания, связанные с интерфейсом, то в каждом конкретном интерфейсе используются определенные выразительные средства. Поэтому формирование *модели выразительных средств* (МВС) конкретного интерфейса сводится к определению значений терминов ОГПИ, т.е. к их конкретизации.

Основными требованиями к процессу формирования модели выразительных средств интерфейса являются: простота разработки модели выразительных средств интерфейса – наличие средств, которые обеспечивают построение модели выразительных средств пользовательского интерфейса без написания процедурного кода; нетрудоемкая модифицируемость – возможность легко модифицировать как отдельные фрагменты модели (без изменения ее в целом), так и любые комбинации таких фрагментов; повторное использование – возможность повторного использования любых фрагментов интерфейса и их композиций как в рамках построения одного пользовательского интерфейса, так и для построения любых других пользовательских интерфейсов; поддержка различных платформ – модель должна формироваться в терминах, общих для различных платформ. Для

реализации перечисленных требований предлагаются следующие подходы.

Для обеспечения простоты разработки предлагается формировать модель выразительных средств на основе ОГПИ, а также предлагается разработать редактор, цель которого – предоставить пользователю вербальные описания элементов ОГПИ и их графических представлений для конкретизации понятий ОГПИ с использованием техники непосредственного манипулирования.

Основными компонентами модели выразительных средств интерфейса являются окна. Как правило, окна содержат элементы управления. У всех интерфейсных элементов (и у окон, и у элементов управления) есть параметры. Параметры, значения которых задаются явно, будем называть определенными. Параметры, значения которых задаются опосредованно, будем называть неопределенными. Интерфейсный элемент, у которого есть неопределенные параметры, назовем интерфейсным прототипом (далее *прототип*). Интерфейсный элемент, у которого все параметры являются определенными, назовем интерфейсным объектом (далее *объект*). Часто различные части интерфейса содержат группы интерфейсных элементов, которые состоят из одних и тех же компонентов, а также обладают одинаковым поведением и семантикой. Такие группы будем называть композициями. Композиция может включать как интерфейсные прототипы, так и интерфейсные объекты. Для обеспечения нетрудоемкой модифицируемости предлагается формировать библиотеку компонентов, состоящую из четырех множеств:

множества прототипов;

множества объектов;

множества композиций;

множества окон.

Требование повторной используемости обеспечивается возможностью использования библиотеки компонентов, содержащей описания интерфейсных элементов, композиций и окон.

Для поддержки различных платформ предлагается описывать модель выразительных средств на основе ОГПИ, которая не зависит от конкретной платформы. Реализация МВС будет осуществляться путем генерации различного кода для различных платформ.

Процесс формирования модели выразительных средств графического пользовательского интерфейса

Модель выразительных средств интерфейса представляется как совокупность значений терминов ОГПИ. Процесс построения модели выразительных средств интерфейса сводится к выбору тех элементов графического пользовательского интерфейса, которые будут применяться в конкретном интерфейсе. Каждому элементу сопоставляется подмножество его свойств из множества возможных.

Модель выразительных средств создается на основе:

универсальной онтологии предметной области «графический пользовательский интерфейс» (ОГПИ);

графических представлений элементов ОГПИ;

способов визуального редактирования графических представлений элементов ОГПИ.

Онтология графического пользовательского интерфейса.

ОГПИ – это граф без циклов с размеченными вершинами и дугами, в котором может быть несколько начальных вершин.

ОГПИ представляет собой пару <C, L>, где C – множество вершин графа, представляющих термины онтологии; L – множество дуг графа, указывающих на отношения между ними.

Множество C вершин графа состоит из двух непересекающихся подмножеств, т.е. $C = M \cup D$, $M \cap D = \varnothing$. Подмножество M является множеством основных вершин графа, $M = \{m_1,\ldots,m_u\}$, D – подмножество вспомогательных вершин, $D = \{d_1,\ldots,d_v\}$.

Множество L дуг графа состоит из двух непересекающихся подмножеств: $L = G \cup A$. G – дуги, определяющие отношение «общее-частное», $G=\{g_1,\ldots,g_m\}$, A – дуги, определяющие отношение «целое-часть», $A=\{a_1,\ldots,a_k\}$.

Каждая основная вершина $m_i \in M$ характеризуется уникальным именем и типом. Различаются два типа основных вершин – интерфейсные элементы и составные свойства интерфейсных элементов. Каждая вспомогательная вершина $d_i \in D$ представляет собой простое свойство интерфейсного элемента и характеризуется своим типом – строковым или целочисленным. Все вершины множества D являются терминальными вершинами, вершины множества M могут быть как терминальными, так и нетерминальными.

Дуги подмножества G определяют отношение «общее-частное» и могут связывать только основные вершины (вершины множества M).

Отношение «общее-частное» является направленным и обладает свойством транзитивности. Дуги подмножества A определяют отношение «целое-часть». Каждая дуга множества A снабжена разметкой, которая состоит из уникального имени, характеризующего эту дугу, и порядка, характеризующего кардинальность отношения. Порядок задается либо в виде неотрицательного целого числа, либо в виде интервала на множестве неотрицательных целых чисел.

В настоящее время ОГПИ разработана и доступна в Internet [5]]. В примере 1 приведена трактовка формального представления ОГПИ на фрагменте реальной онтологии.

Пример 1. Фрагмент ОГПИ.

Множество основных вершин онтологии состоит из следующих эле-

ментов: M = {(Диалоговое окно, elem), (Окно-рамка, elem), (Элемент ГПИ, elem), (Оконный элемент, elem), (Кнопочный элемент управления, elem), (Кнопка управления, elem), (Список, elem)}; множество вспомогательных вершин онтологии D = {integer, string}.

Множество дуг онтологии L = G ∪ A. Пусть дуга множества G описывается следующим образом: (Начальная вершина, Конечная вершина), где Начальная вершина, Конечная вершина – это вершины из множества M. В этом случае множество дуг G, определяющее отношение «общее-частное», примет вид: G = {(Диалоговое окно, Окно-рамка), (Окно-рамка, Элемент ГПИ), (Кнопка управления, Кнопочный элемент управления), (Кнопочный элемент управления, Оконный элемент), (Список, Оконный элемент), (Оконный элемент, Элемент ГПИ)}.

Пусть дуга множества A описывается следующим образом: (Начальная вершина, Конечная вершина, Имя, Порядок), где Начальная вершина – это вершина из множества M, Конечная вершина – это вершина множества M или D, Имя – это идентификатор дуги, Порядок – кардинальность отношения.

В этом случае множество дуг A, определяющее отношение «целое-часть» примет вид: A = {(Элемент ГПИ, integer, высота, 1), (Элемент ГПИ, integer, ширина, 1), (Список, string, элементы, [0...n]), (Кнопочный элемент управления, string, текст, 1)}.

Данный пример описывает фрагмент ОГПИ, описывающий Диалоговое окно, два оконных элемента управления: Кнопку управления и Список, а также промежуточные классы, необходимые для описания общих частей. ОГПИ содержит декларативные описания следующих групп интерфейсных элементов: окна, панели управления, оконные меню, кнопочные элементы, интервальные элементы, текстовые элементы и составные элементы.

Каждый интерфейсный элемент характеризуется набором свойств, – например, кнопка характеризуется ее координатами на экране, размерами, текстом и т. д. Каждый элемент ОГПИ имеет свое графическое представление, которое может меняться в зависимости от его свойств.

Правила процесса формирования модели выразительных средств.

Процесс формирования МВС сводится к обходу графа ОГПИ и последовательному порождению множеств прототипов, объектов и окон. Процесс состоит из последовательности шагов. Каждый шаг – это последовательное применение следующих правил:

1) если между двумя вершинами имеется дуга $a_i \in A$ такая, что ее порядок задается в виде интервала $[n_1 \ldots n_r]$, то порядок дуги заменяется на значение из этого интервала;

2) если между двумя вершинами имеется дуга $a_i \in A$, такая, что порядок дуги задается в виде числа n, где n = 0, то:

2.1) дуга a_i удаляется из графа;

2.2) все вершины, которые стали недостижимыми из корневой после

удаления вершины a_i, удаляются из графа;

2.3. все дуги, у которых нет начальной или конечной вершины также удаляются из графа;

3) если между двумя вершинами имеется дуга $a_i \in A$, выходящая из некоторой вершины $m_j \in M$ и входящая в вершину $c_k \in M \cup D$, такая, что порядок дуги a_i задается в виде натурального числа p, где $p > 1$, то:

3.1) она заменяется на множество дуг $\{a_1,\ldots,a_p\}$, выходящих из m_j и входящих в c_k, где для каждой дуги из этого множества порядок равен 1;

3.2) вершина c_k заменяется на множество вершин $\{c_1,\ldots,c_p\}$ в которые входят, соответственно, дуги $\{a_1,\ldots,a_p\}$;

4) если в графе, сформированном по правилам 1 – 3, имеется вершина $d_i \in D$, то ей сопоставляется некоторое значение (число или строка в зависимости от типа вершины).

Интерфейсный прототип.

В процессе порождения МВС часто оказывается, что в одном пользовательском интерфейсе встречается множество однотипных интерфейсных элементов, имеющих различия только в одном или нескольких его свойствах. В этом случае удобно задать прототип, в котором определены общие для всех однотипных элементов интерфейса свойства, и затем на основе созданного прототипа, формировать конкретный элемент интерфейса, доопределив его уникальные свойства.

Интерфейсный прототип – это подграф ОГПИ, содержащий единственную корневую вершину типа «интерфейсный элемент», имя которого не равно «окно-рамка», а также множества вершин и дуг, полученных из графа ОГПИ путем применения правил 1 – 4, причем к этому подграфу может быть применено хотя бы одно из данных правил. Для одного интерфейсного элемента может быть создано множество различных его прототипов.

В примере 2 приведен интерфейсный прототип, построенный на основе фрагмента ОГПИ из примера 1.

Пример 2. Интерфейсный прототип.

M = {(Элемент ГПИ, elem), (Оконный элемент, elem), (Список, elem)}; D = {integer, string}.

G = {(Список, Оконный элемент), (Оконный элемент, Элемент ГПИ)}.

A = {(Элемент ГПИ, 50, высота, 1), (Элемент ГПИ, 400, ширина, 1), (Список, string, элементы, [0...n])}.

Данный пример описывает интерфейсный прототип Списка. У этого прототипа все параметры, кроме высоты и ширины, являются неопределенными.

Интерфейсный объект.

Интерфейсным объектом называется интерфейсный элемент, у кото-

рого определены все его свойства. Интерфейсные объектом будем называть либо:

подграф ОГПИ, полученный на основе интерфейсного прототипа путем применения правил 1-4 до тех пор, пока невозможно применение ни одного из этих правил (т.е. правила применяются до тех пор, пока ни одно из них не может быть применено);

подграф, у которого корневая вершина типа «интерфейсный элемент» с именем «окно рамка», а вершины и дуги графа формируются по правилам 1-4, за исключением вершины «оконный элемент». Вершина «оконный элемент» заменяется на множество вершин $m_1,\ldots,m_f$, где $1<=f<=k$, где k – количество интерфейсных объектов (подграфов с корневыми вершинами $m_1,\ldots,m_f$) и, соответственно, f дуг, ведущих из корневой вершины в вершины $m_1,\ldots,m_f$.

В примере 3 приведен интерфейсный объект, построенный на основе интерфейсного прототипа из примера 2.

Пример 3. Интерфейсный объект.

M = {(Элемент ГПИ, elem), (Оконный элемент, elem), (Список, elem)}; D = {50, 300, «Строка 1», «Строка 2», «Строка 3»}.

G = {(Список, Оконный элемент), (Оконный элемент, Элемент ГПИ)}.

A = {(Элемент ГПИ, 50, высота, 1), (Элемент ГПИ, 300, ширина, 1), (Список, string, элементы[1], «Строка 1»), (Список, string, элементы[2], «Строка 2»), (Список, string, элементы[3], «Строка 3»)}.

Данный пример описывает интерфейсный объект Списка. Этот объект содержит три строковых элемента: «Строка 1», «Строка 2», «Строка 3», его ширина равна 300 пикселей, а высота – 50 пикселей.

Окно.

Окна являются основными компонентами МВС. Порождение объектов и прототипов окон производится с помощью тех же процессов, которые описаны выше. Прототип окна – интерфейсный прототип, для которого начальной вершиной является «окно-рамка». Объектом окна является интерфейсный объект, созданный на основе интерфейсного прототипа окна.

В примере 4 приведен объект окна, построенный на основе фрагмента ОПГИ из примера 1.

Пример 4. Окно.

M = {(Диалоговое окно, elem), (Окно-рамка, elem), (Элемент ГПИ, elem), (Оконный элемент, elem), (Список, elem)}; D = {200, 400}.

G = {(Диалоговое окно, Окно-рамка), (Окно-рамка, Элемент ГПИ)}.

A = {(Элемент ГПИ, 200, высота, 1), (Элемент ГПИ, 400, ширина, 1)}.

Данный пример описывает объект Диалогового окна. Он не содержит интерфейсных элементов, его высота равна 200 пикселей, а ширина –

400 пикселей.

Композиция.

Композиция является совокупностью некоторого количества оконных интерфейсных элементов.

В примере 5 представлена композиция, состоящая из интерфейсного объекта и прототипа, приведенных в примерах 2 и 3.

Пример 5. Композиция.

{

(M = {(Элемент ГПИ, elem), (Оконный элемент, elem), (Список, elem)}; D = {integer, string}. G = {(Список, Оконный элемент), (Оконный элемент, Элемент ГПИ)}. A = {(Элемент ГПИ, integer, высота, 1), (Элемент ГПИ, integer, ширина, 1), (Список, string, элементы, [0...n])})

(M = {(Элемент ГПИ, elem), (Оконный элемент, elem), (Список, elem)}; D = {50, 300, «Строка 1», «Строка 2», «Строка 3»}. G = {(Список, Оконный элемент), (Оконный элемент, Элемент ГПИ)}. A = {(Элемент ГПИ, 50, высота, 1), (Элемент ГПИ, 300, ширина, 1), (Список, string, элементы[1], «Строка 1»), (Список, string, элементы[2], «Строка 2»), (Список, string, элементы[3], «Строка 3»)}}

}

Данный пример описывает композицию, состоящую из интерфейсного прототипа, описанного в примере 2, и интерфейсного объекта, описанного в примере 3.

Начальное состояние процесса порождения МВС интерфейса – ОГПИ и множество графических представлений элементов ОГПИ. Процесс порождения МВС состоит в последовательном порождении множеств прототипов, объектов, окон с помощью процессов, описанных выше. Процесс может быть завершен, если порожден хотя бы один элемент из множества окон.

Реализация модели выразительных средств

Для поддержки разработки МВС были разработаны следующие специализированные редакторы: редактор ОГПИ и редактор МВС.

Функцией редактора ОГПИ являются формирование и поддержка ОГПИ в актуальном виде, пользователями редактора являются разработчики ОГПИ. С помощью данного редактора в диалоговом режиме пользователь формирует ОГПИ. Онтология ГПИ содержит декларативные описания интерфейсных элементов и необходима для формирования МВС.

Функцией редактора МВС является формирование модели выразительных средств на основе ОГПИ и библиотеки повторно используемых компонентов. Пользователями редактора являются проектировщики интерфейсов. С помощью данного редактора пользователь создает МВС, используя технику непосредственного манипулирования и средства структурного редактирования. Каждый компонент МВС (интерфейсный прото-

тип, объект, окно или композиция) может быть помещен в библиотеку компонентов и повторно использован в дальнейшем при разработке новых МВС.

Заключение

В данной статье дано краткое описание подхода к разработке пользовательского интерфейса на основе онтологий и модели выразительных средств в рамках этого подхода, сформулированы требования к модели и методы их реализации, описан процесс формирования модели выразительных средств и методы реализации модели.

ЛИТЕРАТУРА

1. *Myers B.A.* User Interface Software Tools. ACM Transactions on Computer-Interaction, Vol.2, No.1, March 1995.
2. *Puerta A.* The Mecano Project: Comprehensive and Integrated Support for Model-Based Interface Development //1996. http://www.smi.stanford.edu/projects/mecano.
3. *Брауни Т., Дэйвила Д., Рюгэйбер С., Стайрволт К.* Использование декларативных описаний для моделирования пользовательского интерфейса с помощью системы Mastermind //1998. http://www.isi.edu/isd/Mastermind.
4. *Kleshchev A.A., Gribova V.V.* From an ontology-oriented approach conception to user interface development. International Journal "Information Theories & Applications. 2003. Vol. 10, Num.1. P. 87-94,
5. http://interface.es.dvo.ru.

Статья представлена к публикации членом редколлегии А.С. Клещевым.