Проектирование интерфейса и модель пользователя

Иванов С.Ю., доц., к.т.н.
МИЭМ НИУ ВШЭ, каф. ИТАС

В настоящее время создание новых интерфейсов является основным источником доходов для разработчиков программного обеспечения. Новые интерфейсы и обновления выходят постоянно и пользователя не спрашивают, нужны ли ему эти изменения. Интерфейсы обычно имеют жесткую конструкцию и работают в режиме «меню». Новые функции, часто не используемые, новое расположение кнопок, изменения в логике реализаций операций часто снижает популярность программного продукта. В первую очередь это относится к программным продуктам в интернете. Так популярность портала «Одноклассники» значительно снизилась за счет постоянных изменений в интерфейсе, введении множества мало востребованных функций.

Алан Купер в своей книге, посвященной разработке интерфейсов четко показывает, что в массе своей программы создаются для программистов: на основе их догадок о потребностях пользователя. «Неудобные, сложные продукты окружают нас, и этот круг становится все более тесным …» [1] Часто пользователям не нужно такое количество функций для решения своих задач. Более того усложнение интерфейсов ведет к повышению вероятности возникновения ошибок, которые потом списывают на «человеческий фактор».

У пользователя формируется набор представлений и правил, которые, по его мнению, должны обеспечивать взаимодействие с программной системой. Эту совокупность можно рассматривать как модель пользователя. Если пользователь уже работал с подобной программой, он будет думать, что и эта программа будет работать точно также.

У программной системы тоже есть своя модель. Эта модель является продуктом тех людей, которые разработали программный продукт, и людей, разработавших саму вычислительную систему. То есть интерфейс разрабатывается в соответствии с требованиями пользователя, но по правилам и стандартам разработчиков программного продукта, которые подчиняются правилам и стандартам разработчиков системного программного обеспечения. Часто интересы разработчика расходятся с интересами пользователя. Пользователь хочет дешевый, удобный и простой продукт, программист хочет реализовать все функции, указанные в техническом задании, а запросы пользователя для него остаются на втором плане.

В итоге мы имеем конфликт между моделью и программной моделью, а источником проблем является пользовательский интерфейс.

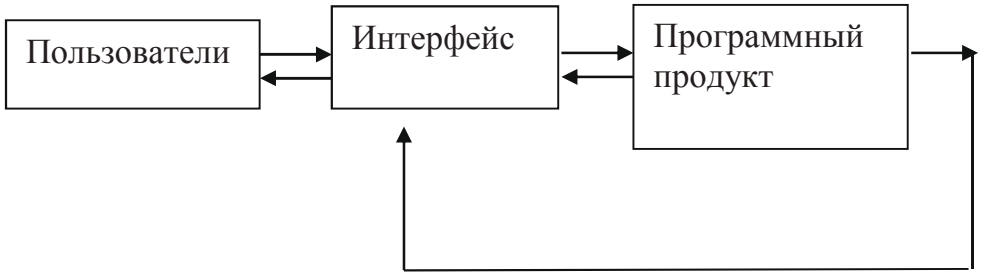

Рис.1 Традиционная схема организации диалога

При традиционном подходе основное внимание разработчиков уделяется реализации необходимых функций программного продукта и избавлению от ошибок при его работе. Интерфейсной части уделяется внимание в основном с точки зрения визуальной привлекательности. Руководители проектов не жалеют средств на дизайнеров и создателей различных визуальных эффектов. Обычно версия интерфейса программы рассматривается с точки зрения целесообразности. Если есть время (обычно его нет), то несколько человек выполнят на ней задания. В это время разработчик должен наблюдать за их действиями, цель - выяснить, чего они ожидают от программы[2]. Наиболее успешные разработчики программных продуктов формируют модели пользователя. С помощью таких моделей они адаптируют интерфейсы к определенным группам пользователей, пытаются понизить вероятность ошибок пользователя и исключить аварийные ситуации. При анализе действий пользователя необходимо учитывать психологические особенности различных групп пользователей, даже менталитет пользователей из различных стран и регионов. Авторы исследований, проведенных в рамках российско-белорусского проекта АИСТ, формирование комплексной модели пользователя для систем дистанционного обучения, отмечают, что существует 2 основных способа построения модели пользователя в интеллектуальных системах. Формирование модели в процессе наблюдений за взаимодействием пользователя с системой, а также построение модели в процессе тестирования пользователя.[3] Здесь же приводятся примеры использования ряда психологических тестов. Модели пользователя становятся сложными и многофакторными. За обилием рассматриваемой информации теряется основная цель – дать пользователю удобный интерфейс, а программисту удобную и понятную модель пользователя.

В данной статье предлагается использовать некоторую простую модель пользователя и модель программы, основанные на принципах систем имитационного моделирования. Предлагается использовать простую систему имитационного моделирования, ориентированную на события. В качестве такой системы берется собственная реализация системы моделирования, принципы работы которой похожи на принципы работы системы SIMPL. Наиболее полезными функциями системы являются функции планирования и выполнения событий, а так же сбор статистики. Планирование и выполнение событий соответствуют действиям пользователя при работе интерфейса, а средства ведения статистики позволяют запротоколировать все действия пользователя для дальнейшего анализа. Такой подход может служить основой для создания автоматизированных средств по формированию модели пользователя и имитатору работы программного продукта. Основной проблемой здесь является синхронизация имитационной модели, которая работает в модельном времени, и реального объекта, который работает в реальном времени. Такую синхронизацию можно реализовать за счет использования принципов, изложенных в работе [4].

Рассмотрим пример реализации интерфейса для реализации достаточно простого взаимодействия пользователя с программной системой, обеспечивающей биржевые операции. Существуют различные реализации электронной биржевой торговли. Они достаточно сложны и основное внимание в них уделяется наиболее важной проблеме – защите информации. Предлагается максимально простая система для пользователя, которая в автономном режиме позволяет реализовывать важные для него операции и получать только нужную ему информацию и необходимую статистику. Программа

использует сразу два режима диалога: заполнение бланка и ввода сообщения на ограниченном естественном языке. Предполагается, что упрощение системы приведет к уменьшению ошибок пользователя и формированию им обоснованных решений. Разделение основной системы электронной биржевой торговли дает пользователю дополнительное время и средства для оценки своих предполагаемых действий на бирже. Соревноваться в скорости проведения биржевых операций с профессиональными брокерами, а тем более с программами роботами не имеет смысла.

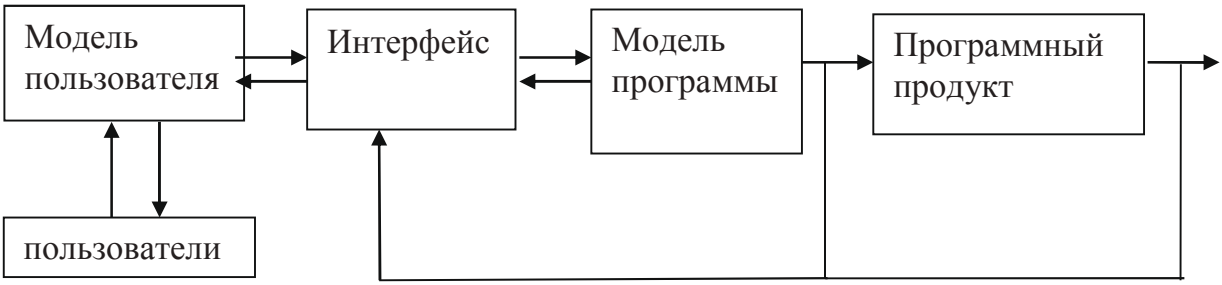

Рис.2 Схема организации диалога с использованием модели пользователя и модели программы

Пример взаимодействия пользователя с программой, ориентированной на анализ котировок и проведение операций на бирже драгоценных металлов. Пользователь задает ряд команд, связанных с покупкой и продажей металлов. Важным критерием для диалога в программе, который реализуется в режиме «бланк», является надежность и осмысленность при проведении операций. Существует возможность потерять деньги при необдуманном поведении пользователя, при ошибочном использовании интерфейса. Так как основное внимание пользователя привлечено к операциям на бирже, то интерфейс должен быть максимально простым в использовании. На самом деле основных операций всего две, продать или купить металл. Однако вариантов входного предложения на языке пользователя больше. Например, пользователь хочет купить 100 грамм золота. Возможные фразы: «купить 100 грамм золота», «100 грамм золота купить», «купить золота 100 г», «очень хочу купить золота 100 г» и т.п. Программа должна однозначно и безусловно понять приказ пользователя. Для этого в программе была использована фреймовая структура в виде сети. Причем анализируются во входной фразе только те слоты, которые соответствуют значащим для проведения операции словам. Именно они используются для анализа заданной фреймовой структуры. Остальные слова игнорируются программой, а пользователю дается возможность в удобном для него виде задавать команду. Для данного примера ядром фреймовой структуры будет сеть следующего вида (см. рис. 3).

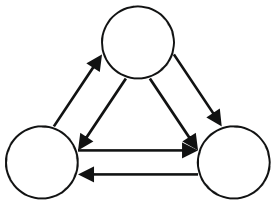

Рис.3 Ядро фреймовой структуры

Данный граф отображается матрицей «вершина - потомки» и массивом, где вершинам соответствуют определенные словари. Числовое значение не включается в структуру и обрабатывается отдельно. Словари могут быть вынесены из программы и модифицироваться без изменения программного кода.

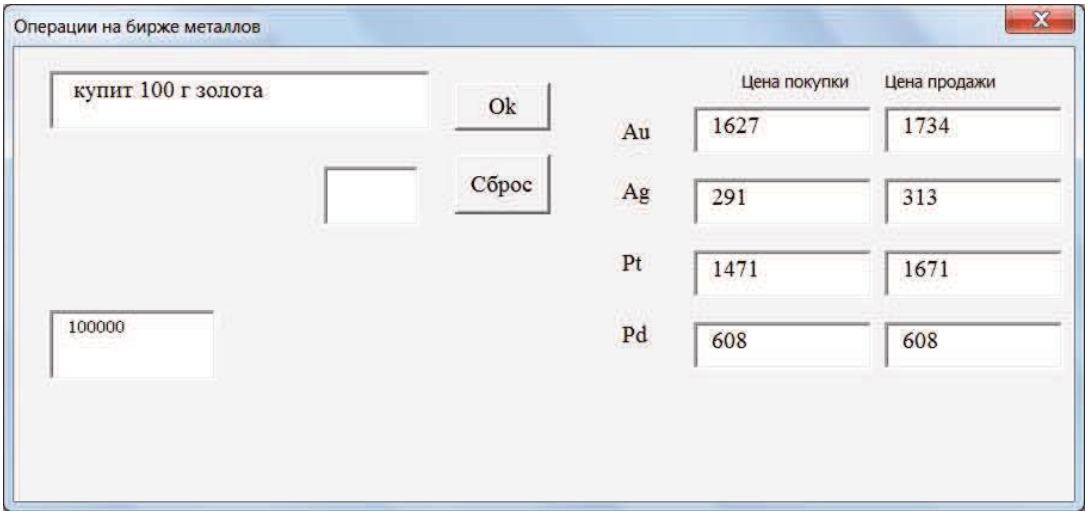

Рис.4 Пример интерфейса

При таком представлении фразы анализируются только значащие 3 слова. Причем фразу можно дополнять другими словами. Слова можно использовать в любом порядке, переставлять. Допускается возможная модификация слова. Например, вместо слова «грамм» может использоваться слово «грамм» или сокращение «г». Пользователь очень любит сокращения и не любит ограничения. Такую простую программу, с небольшим количеством реакций системы обычно реализуют с помощью жесткого аппарата меню, который заставляет его работать по жестким правилам. Однако сеть фреймов дает большие возможности для расширения системы. Особенно хорошо этот подход будет работать при распознавании голосовых команд. Но самое главное, уменьшается количество ограничений и безусловных правил для пользователя.

Список литературы

1. Купер А. // Психбольница в руках пациентов. Символ-Плюс. ISBN 978-5-93286-168-4, 0-672-31649-8; 2009 г.

2. Joel Spolsky. User Interface Design for Programmers, Apress, 2001. ISBN 1-893115-94-1

3. Gavrilova T., Vasilyeva E. One Approach to Individualised Interface Design // Proceedings of X-th Conference «Knowledge-Dialogue-Solution», Varna, Bulgaria, 2003. P. 221–226. (in English).

4. Иванов С.Ю. Синхронизация программных подсистем и реальных объектов при натурном и полунатурном моделировании. //Научные материалы Первой международной научно-технической конференции «Аэрокосмические технологии», МГТУ им. Р.Э. Баумана, Реутов, НПО«Машиностроение», 2004г.,с.186-188