Алексашин Сергей Васильевич

Магистрант

Направление: Информатика и вычислительная техника

Магистерская программа: Информационные системы

Разработка и моделирование системы поддержки принятия решений при проектировании локальных вычислительных сетей

Аннотация. В статье описана разработка прототипов пользовательского интерфейса с использованием технологии JavaFX, описаны алгоритмы поддержки принятия решений при проектировании ЛВС.

Ключевые слова: локальная сеть, информация, топология, пропорциональный доступ, передача данных, программные средства, коммутатор.

Глобальная информатизация и стремительное развитие компьютерных технологий позиционируют использование значительного объема данных. С одной стороны, информация дает возможность оперативно и точно выполнить необходимые расчеты и проанализировать материал, а с другой – превращает поиск нужного решения в сложную задачу [5, с. 54].

За последние 15 лет, как в России, так и в зарубежных странах появилось значительное число работ, в которых рассматривались системы поддержки принятия решений. Изученные работы К.А. Аксенова, А.Ю. Вишняковой, Д.О. Хромова помогли в достижении цели, а именно разработан программный комплекс, который позволяет облегчить решение многокритериальных задач при проектировании ЛВС.

Первоначальным этапов в разработке интерфейса начинается с схематического макета, который реализуется с помощью выбранных компонентов Swing. По нашему мнению, главное окно программы «меню» должно остаться неизменным. Дальнейшие изменения происходят в основном

окре программы – это будет завесить от шагов, на которых будет производиться расчет [3, с. 82].

В первоначальном окне программы представлено количество заявленных критериев и альтернатив. Пользователь самостоятельно вводит нужное количество, которые в последующем будут сравниваться. В окно программы входят следующие элементы: JPanel; JTextLabel; JTextField; JButton. Окно выбора критериев JButton позволяет пользователю выбрать критерии, которые отображаются в виде списка. Окно программы состоит из следующих элементов: JPanel; JLabel; JComboBox; JButton.

После того, как выбрано определенное количество критерием, нажимается активная кнопка и отображается окно ввода альтернатив. Данное окно служит для ввода сравнения определенного количества альтернатив. Оно состоит из таких элементов как JPanel; JLabel; JTextField: JButton;

В момент ввода определенных альтернатив начинает отображаться окно ввода матриц. На данном шаге пользователь заполняет матрицу попарного сравнения в соответствие со справкой. Данная справка отображается в правой части окна. Данное окно состоит из таких элементов, как: JPanel; JLabel; JTextField: JButton; JScrollPane; JTextArea. В последующих действиях интерфейс не меняется, происходит дублирование столько раз, сколько это потребуется пользователем. Может поменяться сама справка и название матрицы [3, с. 91].

Резюмируя проведенную работу можно сделать вывод, что интерфейс содержит минимум критериев для расчетных действий. Результат работы с технологией Swing показал, что необходимо потратить достаточно много времени для полноценной реализации всех возможностей программы. Код разметки программы практически идентичен для всех компонентов, только лишь в некоторых случаях отличается определенными параметрами.

Нами было принято решение в поиске возможности реализации интерфейса программы с использованием как можно меньшего объема ненужного и повторяющегося кода. Это позволит ускорить разработку, а

внешний вид разрабатываемого приложения будет улучшен. Принято решение переписывать большие объемы программного кода, что позволит модернизировать разрабатываемое приложение.

В ходе работы была использована еще одна технология JavaFX. Ее изучение показало, что с ее помощью достигаются все нужные для нас условия и реализации поставленных целей и задач. Она позволяет разработать удобный и понятный интерфейс, реализовать полезные функции, которые помогают понимать процессы решения многокритериальных задач со стороны простого пользователя, так как не все люди имеют хорошие знания в области программирования и использования ПО [1, с. 215].

При использовании Swing необходимо описывать расположение компонента в том же самом файле, где находится реализация действий, совершаемых данным компонентом. Ввиду данной особенности программист вынужден писать очень много кода там, где это не нужно и часто возникает путаница, что именно ты редактируешь, расположение компонента или же его функции. При переходе к использованию JavaFX данная проблема частично решается. Программист так же может писать код размещения компонентов вместе с логикой приложения, однако с использованием языка декларативной разметки FXML, созданного на основе XML, а также некоторых дополнительных программ, можно отделить код интерфейса и код взаимодействия компонентов с программой, что и было сделано в результате перехода на JavaFX.

Для разработки макета интерфейса была использована программа «SceneBuilder». Данная программа позволяет путем простого и наглядного расположения элементов интерфейса генерировать готовый FXML код, в виду чего у программиста отсутствует необходимость изучать FXMLи писать код самому. Данное приложение разделено на несколько секций, каждая из которых отвечает за определенные параметры будущего интерфейса. Слева находятся панели с доступными компонентами, разделенными на тематические группы такие как: Containers, Controls, Menu и т.д. Ниже находится вкладка Hierarchy,

которая показывает все дерево компонентов интерфейса, связанных между собой. Вкладки справа отвечают за свойства компонента (Properties), его расположение на экране (Layout), а также за связь элемента и логики приложения (Code). Редактирование FXMLкода приложения было минимальным, что является неоспоримым достоинством «SceneBuilder» с точки зрения начинающего и неопытного разработчика [2, с. 101].

Разработанным приложением сможет пользоваться любой человек, не обладающий специальными знаниями в теории принятия решений. Все подсказки даны в достаточном для понимания процесса работы программы объеме, что позволяет пользователю начать расчеты без изучения дополнительной литературы. Одно из окон состоит из следующих компонентов интерфейса: MenuBar, двух компонентов типа Label и кнопки типа Button. Компоненты обернуты в AnchorPane, а также VBox и HBox, что позволяет сохранять необходимые расстояния между компонентами, обеспечивая так называемый «резиновый макет». В дальнейшем все компоненты во всех созданных окнах обернуты похожим образом, что облегчает создание интерфейса.

Компоненты типа Labelсодержат в себе название программы, а также справочную информацию, рассказывающую о назначении приложения и правилах дальнейшей работы с ним. Компонент Button представляет собой кнопку по нажатию на которую открывается следующее окно, запускающее процесс расчета. Окно состоит из следующих компонентов: «четырех элементов типа Label, двух элементов TextField, элементов MenuBarи Button». Текст внутри окна выбора критериев выполнен с помощью эдементов Label. Выпадающий список реализован с помощью элемента ComboBox. Элемент позволяет отслеживать изменения в списке, позволяя перемещать критерии в список, выполненный с помощью элемента ListView. Окно «Ввод альтернатив» состоит из элементов типа Label, TextFieldи Button. В текстовое поле пользователь вводит название альтернативы и по нажатию кнопки «Добавить» она оказывается в списке введенных альтернатив [4, с. 168].

Сама матрица реализована с помощью двух компонентов: GridPane и TextField. GridPane представляет собой панель, располагающую элементы в виде таблицы, что позволяет легко строить матрицу на основе TextField. Использование технологии JavaFX позволило разработать удобный и понятный пользователю интерфейс взаимодействия с программой, упростило разработку системы и позволило разделить логику программы и написание интерфейса. Такой подход позволил структурировать необходимые работы и облегчил поиск и исправление ошибок, возникших при разработке приложения.

Интерфейс в полном объеме выполняет необходимые задачи, обладает возможностью автоматической растяжки компонентов в случае невозможности отображения всех необходимых элементов в окне приложения. Благодаря этому пользователь сможет комфортно работать с приложением вне зависимости от размеров экрана компьютера, что позволяет применять приложение как на современных компьютерах, так и на компьютерах, возможности которых в какой-либо степени ограничены [5, с. 51].

Разработанное приложение одинаково выглядит и работает на всех современных системах, так как имеет определенный принцип работы. Внедрение данного языка, в дальнейшем позволит перенести программу на платформу Android. Данная особенность позволит применить программу на любом мобильном устройстве без привязки к основному компьютеру.

Литература

1. Акопов А.С. Имитационное моделирование: Учебник и практикум для академического бакалавриата. М.: Юрайт, 2016.

2. Аксенов К.А., Гончарова Н.В. Моделирование и принятие решений в организационно-технических системах: Учебное пособие. В 2 ч. Ч.1. Екатеринбург: Изд-во Уральского ун-та, 2015.

3. Никонов О.И, Кругликов С.В., Медведева М.А. Математическое моделирование и методы принятия решений: Учеб. пособие. Екатеринбург: Изд-во Уральского ун-та, 2015.

4. Прокопенко Н.Ю. Системы поддержки принятия решений: Учеб. пособие. – Н. Новгород: ННГАСУ, 2017.

5. Хромов Д.О. Системы поддержки приятия решений для мониторинга работы ЛВС // Молодой ученый. 2014. № 16 (75).