# ИНФОРМАЦИОННАЯ СИСТЕМА МОДЕЛИРОВАНИЯ ТЕХНОГЕННЫХ КАТАСТРОФ

Поздняков С. В., Давыдов М. В.
Кафедра информационных технологий автоматизированных систем, кафедра теоретических основ электротехники, Белорусский государственный университет информатики и радиоэлектороники
Минск, Республика Беларусь
E-mail: sergey.posdnyakov@gmail.com, davydov-mv@bsuir.by

*В работе используются методы системного анализа, асинхронного программирования, автоматизированного проектирования информационных систем, технологии хранилищ данных. Разработанные модели сценариев позволяют определять состав необходимых ликвидационных мероприятий, а также оценивать их эффективность. Кроме того, методы формирования и коррекции планов ликвидации, реализованные в составе информационной системы моделирования, позволят повысить оперативность и эффективность планирования мероприятий при ликвидации крупных техногенных катастроф.*

## ВВЕДЕНИЕ

Актуальность разработки и моделирования информационной системы по предотвращению и ликвидации последствий чрезвычайных ситуаций обусловлена возрастанием рисков природных и техногенных катастроф. Потребность в разнообразной, своевременной, точной и адекватной информации о состоянии природно-промышленной системы, связанной с предотвращением возможных последствий различных чрезвычайных ситуаций, делает необходимым использование информационных систем, которые отслеживают все возможные состояния природно-промышленной системы, различные влияния на нее, её модели поведения. Рассмотрение вопроса о моделировании такой информационной системы является особо актуальным.

## I. АНАЛИЗ СУЩЕСТВУЮЩИХ МАТЕМАТИЧЕСКИХ МЕТОДОВ

Большой раздел современной теории безопасности сложных технических систем посвящен анализу и оценке риска, в которых риск используется в качестве одного из важных критериев безопасности. Риск определяется через функционал $F_R$ вероятности наступления катастрофы (природного или техногенного характера) и величины ущерба:

$$R = F_R(Q, C) \tag{1}$$

При анализе риска вероятность Q и ущерб C можно рассматривать как случайные величины, которые имеют свои распределения $f_Q(q)$ и $f_C(c)$ . Ущерб C есть случайная величина, а вероятность события Q также может считаться величиной случайной, потому что она оценивается с некоторой степенью достоверности по ограниченной выборке (например, по статистике аварий). Тогда мы имеем систему двух случайных величин с заданными распределениями и можем ставить вопрос о поиске совместного распределения $f_{QC}(q, c)$. При этом возможно два варианта предположений: случайные величины Q и C независимы (некоррелированы); случайные величины Q и C зависимы (коррелированы).

Таким образом, зная вид законов распределения и значения их параметров $f_Q(q)$ и $f_C(c)$, можно определить вид и значение параметров функций распределения $F_R(r)$ и плотности $f_R(r)$ техногенного риска.

## II. АНАЛИЗ И ВЫБОР ЭФФЕКТИВНЫХ АЛГОРИТМОВ И АРХИТЕКТУРЫ

Современные требования к пользовательским интерфейсам веб-ориентированных систем накладывают повышенные требования к интерактивности и удобству использования функций приложения. В связи с этим, язык программирования JavaScript активно используется для работы на стороне клиента, а вместе с ним и весь стек связанных с языком технологий. Из-за специфики веб-приложений, разработка для веб-приложений более трудоèмка, чем написание десктопных приложений. Для того, чтобы решить эту проблему, используют различные веб-фреймворки, основанные на архитектурах MVC и Flux. На данный момент известно несколько хорошо себя зарекомендовавших себя решений: Angular, Ember, Backbone, React. Полноценными фреймворками являются Angular и Ember, Backbone и React–библиотеки для построения пользовательских интерфейсов. Для больших веб-приложений остро встаèт проблема зависимости частей приложения между собой, что влечет значительное возрастание сложности архитектуры веб-системы и падения производительности ее работы.

Одним из путей решения данного вопроса может быть использование связки React.js и архитектуры Flux. Новая, активно развивающаяся и перспективная разработка компании Facebook - React в последнее время привлекает все большее внимание. В сочетании с применением архитектуры Flux библиотеку React.js можно считать полноценным фреймворком. Ос-

новной идеей React.js является компонентно-ориентированный подход, опирающийся на революционную идею сочетать JavaScript-код и разметку компонентов. Flux – это архитектура, базирующаяся на однонаправленном потоке данных и подходе управлением изменением состояния через события.

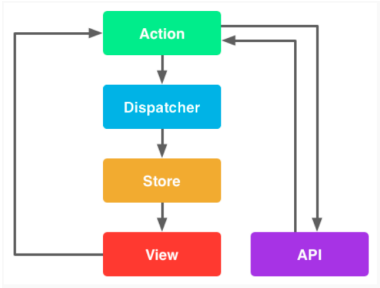


Рис. 1 – Структурная схема архитектуры

Основными элементами Flux-архитектуры являются:

- View – пользовательский интерфейс состоящий из react-компонентов;
- Dispatcher – система регистрации обратных вызовов и реагирования на события;
- Store – хранилище данных на клиенте, которое взаимодействует с View через Dispatcher;
- Action – прослойка между Dispatcher и View для регистрации обработчиков событий действий пользователя.

Типичным подходом в Flux является единственный диспетчер, в котором регистрируются все события приложения и многочисленные хранилища в соответствии с доменными областями приложения.

Таким образом, был рассмотрен подход к решению проблемы возрастания сложности и падения производительности в веб-приложениях, которая связана с особенностью архитектуры платформы, предполагающей единственный диспетчер событий. Подход разделения приложения на зоны ответственности нескольких диспетчеров обеспечивает упрощение кода приложения, что сокращает время на расширение и сопровождение кода, а также предотвращает возможную проблему производительности.

## III. Структура приложения

Разработанный пользовательский интерфейс, позволяет работать с основными климатическими характеристиками (температура, влажность, давление, облачность, географические координаты). Была разработана архитектура приложения, произведена сборка проекта (webpack), были созданы независимые друг от друга компоненты.

Главный компонент приложения включает в себя несколько дочерних компонентов, таких, как WeatherDisplay (компонент, включающий получение API из онлайн-сервиса погоды), NavItem (компонент, содержащий список белорусских городов), NavBar (компонент-хедер приложения). Получение API осуществляется в режиме реального времени с сайта https://openweathermap.org.

Обработка и получение данных в приложении осуществляется с помощью запроса на указанный выше URL и преобразования их в формат JSON. Ключом для получения данных является zip code (система почтовых индексов города).

## IV. Список литературы

1. Вишняков, А. Д. Общая теория риска / А. Д. Вишняков // Изд. центр «Академия», 2008. – 368 с.
2. Острейковский, В. А. Количественная оценка риска в теории техногенной безопасности сложных динамических систем / В. А. Острейковский // РАН, 2013. – с.12-31.
3. Документация по React.js [Электронный ресурс] – Режим доступа: https://facebook.github.io/react.
4. MVC в JavaScript [Электронный ресурс] – Режим доступа: http://designformasters.info/posts/mvc-javascript.
5. Flux официальный сайт [Электронный ресурс] – Режим доступа: https://facebook.github.io/flux/docs.