

УДК 004.921

DOI 10.24412/2413-7383-2025-3-38-158-171

А. А. Левашов, Д. А. Гаркуша

Федеральное государственное бюджетное научное учреждение
«Институт проблем искусственного интеллекта», г. Донецк, Россия
283048, г. Донецк, ул. Артема, дом 118б

ВИЗУАЛИЗАЦИЯ ПРОИЗВОДСТВЕННО- ТЕХНОЛОГИЧЕСКИХ ПРОЦЕССОВ МОРСКОГО ПОРТА *

А. А. Levashov, D. A. Garkusha

FSBSI "Institute of problems of artificial intelligence"
283048, Donetsk, Artema St., Building 118b

VISUALIZATION OF PRODUCTION AND TECHNOLOGICAL PROCESSES OF THE SEA PORT

В статье предложены требования, ориентированные на задачи оптимизации ресурсов и ускорения принятия решений на оперативном, тактическом и стратегическом уровнях управления морским портом. Сформулированы требования к отображению статистической и динамической информации, интерактивности, производительности, интеграции и пользовательского опыта (UX). Проведен сравнительный анализ веб-технологий для интерактивной 2D-графики. Обоснован выбор стека технологий React, Redux и Konva.js для реализации установленных требований. Представлены принципы реализации ключевых функций, включая drag-and-drop для ручного планирования размещения объектов. Результаты работы подтверждают эффективность предложенного метода и выбранного инструментария для создания масштабируемых, производительных и интерактивных систем веб-визуализации, для повышения управляемости и логистической эффективности морских портов.

Ключевые слова: веб-визуализация, морской порт, производственно-технологические процессы, интерактивность, drag-and-drop, React.js, Redux, Konva.js, оптимизация логистики.

The article proposes requirements focused on the tasks of optimizing resources and speeding up decision-making at the operational, tactical and strategic levels of seaport management. The requirements for displaying statistical and dynamic information, interactivity, performance, integration, and user experience (UX) are formulated. A comparative analysis of web technologies for interactive 2D graphics is carried out. The choice of the React, Redux and Konva technology stack is justified.js to implement the established requirements. The principles of implementing key functions, including drag-and-drop for manual object placement planning, are presented. The results of the work confirm the effectiveness of the proposed method and the selected tools for creating scalable, productive and interactive web visualization systems to improve the manageability and logistical efficiency of seaports.

Key words: web visualization, seaport, production and technological processes, interactivity, drag-and-drop, React.js, Redux, Konva.js, logistics optimization.

* **Поддержка исследований.** Работа выполнена при финансовой поддержке Министерства науки и высшего образования РФ в рамках НИР №Г/Р 123092600030-4.

Введение

Современные морские порты представляют собой сложные технологические комплексы с высокой интенсивностью грузопотоков и множеством взаимодействующих объектов [1]. Эффективное оперативное управление такими системами требует непрерывного мониторинга состояния производственно-технологического процесса [2], [3]. Традиционные SCADA/HMI-системы, хотя и надежны, часто обладают ограниченной гибкостью, масштабируемостью и доступностью [4]. Веб-технологии предлагают альтернативу этим системам [5] и позволяют создавать кроссплатформенные, легко обновляемые и интегрируемые приложения, предназначенные для повышения операционной эффективности, безопасности и управляемости производственно-технологических процессов морского порта [6].

Процесс визуализации представляет собой непрерывное преобразование сырых данных (позиции судов/техники, статусы оборудования, грузопотоки, метеоусловия) в интуитивно понятные графические образы (2D/3D-карты, мнемосхемы, анимированные модели) в режиме реального времени для последующего анализа. Визуализация является ключевым инструментом преобразования информации в знания [7] для эффективного управления на оперативном, тактическом и стратегическом уровнях принятия решений [8].

Оперативный уровень – определяет текущее состояние всех процессов, а также позволяет мгновенно выявить аномалии (авария, простой). Например, крановщик видит на схеме точное положение контейнера и путь погрузчика; диспетчер наблюдает общую картину загрузки причалов.

Тактический уровень – представляет возможность определить причинно-следственные связи и дать оценку эффективности тактических решений. Например, диспетчер анализирует тепловые карты простоев техники для оптимизации распределения ресурсов.

Стратегический уровень – процесс, способствующий формированию стратегических заключений, необходимых для развития инфраструктуры и бизнес-модели. Например, использование данных цифрового двойника для моделирования последствий расширения причала.

Целью данной работы является определение набора эффективных инструментов и технологических решений для разработки веб-интерфейса визуализации данных в системах управления логистикой морского порта, обеспечивающего интеграцию с компонентами искусственного интеллекта, экспертных систем и Smart Systems, на основе оценки их функциональности, производительности, удобства разработки и эксплуатации, а также соответствия специфическим требованиям портовой логистики.

Для достижения цели в работе поставлены и решены следующие задачи:

1. Проанализировать объект исследования – морской порт, выделив ключевые сущности, процессы и информационные потоки, подлежащие визуализации.
2. Сформулировать комплекс функциональных требований к системе веб-визуализации, ориентированных на задачи оперативного, тактического и стратегического управления.
3. Провести сравнительный анализ современных веб-технологий для создания интерактивной 2D-графики и выявить наиболее подходящие для отображения множества динамических объектов в реальном времени.
4. Обосновать выбор стека технологий (React, Redux, Konva.js) для реализации системы, доказав его соответствие сформулированным требованиям.
5. Разработать и описать принципы реализации ключевого интерактивного функционала (на примере операции drag-and-drop) для манипулирования объектами в контексте задач портовой логистики.

1. Характеристика объекта исследования: морской порт

Морской порт — высокودинамичный комплекс, где непрерывно взаимодействуют суда (прибытие, швартовка, погрузка/разгрузка), перегрузочная техника (краны, погрузчики), склады, наземный транспорт (ж/д, авто) и системы управления [9].

Морским портам присваивается статус класса объект по значимости: международные, межрегиональные и локальные. Также все порты специализируются на обработке определенных типов грузов, основными из которых являются: сыпучие, жидкие и контейнерные. В тип сыпучие входит сырье, которое разделяют на пищевое и непищевое, вследствие чего возникает необходимость разделять помещения под хранение данного вида сырья. То же относится к остальным типам груза. Контейнерные грузы бывают разных размеров, а также могут требовать разной обработки и определенных условий хранения, например климатические, что вынуждает эти же грузы хранить в специальных складах (рефрижераторы). Из вышесказанного следует, что морской порт — это сложный объект, который решает целый ряд комплексных задач по обеспечению доставки груза от пункта А до пункта Б с выполнением всех необходимых промежуточных этапов.

Данной теме посвящается множество работ, рассматривающих как повышение эффективности работы морского порта, так и предложения о разработке и внедрения информационных систем для решения ряда задач, относящихся к работам грузового фронта. Для более ясного понимания углубимся в специфику, на примере функционирования контейнерного терминала, описанного в одной из работ [10].

Контейнерный терминал в морском порту — это специализированный комплекс, выполняющий функции ключевого узла в глобальной цепи контейнерных перевозок и предназначенный для перевалки грузов между морскими судами и другими видами транспорта (автомобильным, железнодорожным) [11].

Контейнерный терминал состоит из множества зон. Их можно рассмотреть в качестве следующих объектов: площадки груженых, поврежденных и порожних контейнеров, площадки автомобильно-железнодорожного погрузочного фронта, сортировочного фронта, площадки комплектации контейнеров. В мировой практике наиболее распространены складские площадки открытого хранения контейнеров (кроме складов комплектации), обслуживаемых контейнеровозами или полуприцепами с тягачами. Рассмотрим схему внутрипортовых контейнерных грузопотоков в соответствии с рис. 1.

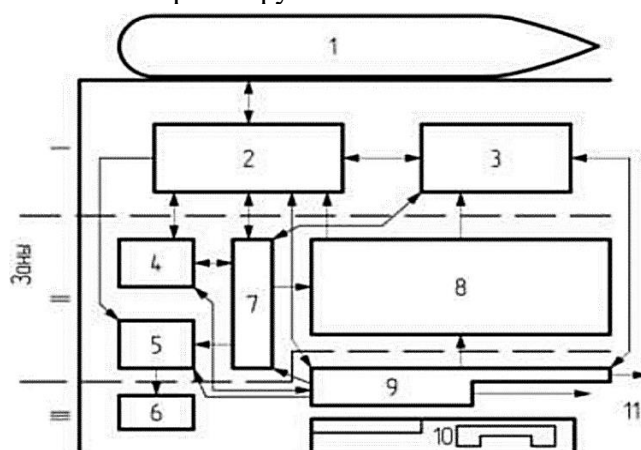


Рисунок 1 – Схема внутрипортовых контейнерных грузопотоков [9] где:

1 – судно-контейнеровоз; 2,3 – сортировочные площадки; 4,5 – склады порожних и поврежденных контейнеров; 6 – ремонтные мастерские; 7 – склад комплектации; 8 – складская площадка груженых контейнеров; 9 – оперативная площадка автомобильно-железнодорожного грузового фронта; 10 – административно-управленческие здания; 11 – железнодорожный и автомобильный везды

Схема внутрипортовых контейнерных грузопотоков предусматривает взаимодействие перечисленных объектов, демонстрируя пример перемещения и обработку контейнеров. Управление таким сложным комплексом требует непрерывного мониторинга большого количества параметров и оперативного принятия решений для минимизации простоев, оптимизации ресурсов и обеспечения безопасности. Для решения этой задачи необходимо обеспечить Лицо Принимающее Решения (ЛПР) актуальной и целостной информацией о производственно-технологических процессах морского порта в доступном для восприятия виде, применив информационные технологии для их визуализации.

2. Анализ существующих систем визуализации производственно-технологических процессов морского порта

Внедрение систем визуализации для мониторинга и анализа деятельности морских портов является актуальной задачей в условиях роста грузопотоков и требований к эффективности логистики. Существующие решения можно условно разделить на несколько категорий:

- SCADA/HMI-системы (Например: Ignition, WinCC, Citect): - традиционные системы диспетчерского управления, обеспечивающие визуализацию в реальном времени состояния оборудования (краны, конвейеры, шлюзы), телеметрии и базовых технологических параметров. Предназначены в первую очередь для оперативного персонала в диспетчерских [12].
- Системы имитационного моделирования с функциями визуализации (Например: AnyLogic, FlexSim, Arena) - мощные инструменты для создания детальных цифровых двойников технологических процессов, позволяющие не только визуализировать, но и глубоко анализировать, оптимизировать и прогнозировать работу объекта [13].

В отличие от вышеперечисленных технологий, существует способ визуализации с помощью веб технологий [14]. Они имеют ряд преимуществ в сравнении с классическими системами:

1. Кроссплатформенность и доступ – можно получить доступ через любой браузер на любом устройстве без установки специального программного обеспечения, в отличие от SCADA/HMI систем, которые требуют установки специализированных клиентских приложений (часто только под Windows) на конкретные рабочие места или систем имитационного моделирования, которые требуют специальное ПО для создания моделей [15].

2. Масштабируемость и гибкость – обладают гибкой архитектурой позволяя развивать отдельные модули приложения независимо и по необходимости масштабировать серверную часть под растущее число пользователей, в отличие от SCADA/HMI систем, в которых архитектура часто монолитная и жесткая, а масштабирование может быть сложным и дорогим, требующим обновления лицензий и инфраструктуры или систем имитационного моделирования, в которых также масштабирование является трудоёмким процессом.

3. Интеграция с современным стеком технологий – легко интегрируются с облачными сервисами (хранение данных, аналитика), BI-инструментами (Power BI, Tableau через API), системами планирования (ERP), мобильными приложениями в отличие от SCADA/HMI и систем имитационного моделирования, в которых интеграция возможна (через API), но часто сложнее, требует специалистов и промежуточного программного обеспечения [16].

4. Современный пользовательский интерфейс – в наличии широкий спектр популярных фреймворков, которые позволяют создавать интуитивные, интерактивные, адаптивные и визуально привлекательные интерфейсы, в отличие от SCADA/HMI в которых интерфейсы часто функциональны, но могут быть устаревшими визуально и менее интерактивными, ориентированными в первую очередь на оператора в диспетчерской, а не на аналитиков или руководство [17].

Главным недостатком самостоятельной разработки веб-приложения с использованием современных фреймворков выступает значительная трудоемкость и сложность проектирования и реализации полнофункционального интерфейса, сравнимого по возможностям визуализации сложных технологических процессов в реальном времени со специализированными SCADA/HMI-системами. В отличие от готовых промышленных решений, веб-разработка требует глубокой проработки архитектуры, реализации механизмов обработки и отображения большого объема динамических данных, что увеличивает сроки и стоимость создания системы.

Все вышеперечисленные преимущества и недостатки склоняют к выбору самостоятельной разработки веб-приложения с использованием современных веб-технологий (JavaScript-фреймворки), которые предоставляет полную свободу в проектировании интерфейса, интеграции, анализе данных и дают возможность контролировать производственно-технологический процесс в морском порту.

3. Основные требования к веб визуализации

Для решения задач оптимизации логистических решений и необходимых ресурсов для ускорения принятия решений определены следующие ключевые требования:

1. Отображение статистического контента – включает точное и масштабируемое представление схемы портовой акватории, причалов, складов и грузового фронта. Веб-приложение должно поддерживать многослойную визуализацию, включая фон, сетку, инфраструктуру, объекты и временные элементы, а также предоставлять возможность загрузки подложек, к примеру таких как спутниковые снимки.

2. Визуализация динамических объектов и состояний – охватывает отображение судов с детализацией (контур, тип, статус, название, осадка, уникальный идентификатор), их позиции и ориентир у причала или на рейде. Веб-приложение должно визуализировать портовую технику (краны, погрузчики, тягачи), отображая тип, состояние (работает, авария, простой), текущую позицию, направление движения и загрузку, а также обеспечивать трекинг контейнеров и грузов с указанием идентификации, местоположения (на складе, транспортном средстве, под краном) и статуса обработки. Транспортные средства (автотранспорт, ж/д вагоны) отображаются с информацией о статусе и грузе. Веб-приложение также должно динамически изменять состояние объектов, используя цветовую индикацию статусов и анимацию перемещения, а также визуализировать запрещенные участки и маршруты.

3. Интерактивность – реализуется через функции масштабирования схемы порта, выделения объектов при наведении курсора или клике, отображения детальной информации об объектах во всплывающих окнах или сайдбарах, фильтрации объектов по типу, статусу или причалу, а также поддержку операции drag-and-drop для ручного планирования или корректировки размещения.

4. Требования к производительности и масштабируемости – предполагает обеспечение плавной анимации и перемещений объектов даже при их значительном количестве, обновление состояний объектов при получении данных в реальном времени, а также возможность эффективно функционировать на электронно-вычислительной машине исключая требования к высокой производительности для клиентской части.

5. Интеграция и работа с данными – включают поддержку получения данных в реальном времени, возможность загрузки статистических данных для анализа, интеграцию с внешними системами через API и обязательное логирование действий пользователя.

6. Пользовательский опыт (UX) – фокусируется на создании интуитивно понятного интерфейса, предоставлении настраиваемых видов (с возможностью показа/скрытия слоев и сохранения положения), а также реализации адаптивного дизайна для корректного отображения на разных разрешениях экранов.

4. Анализ веб-технологий для реализации интерактивной 2D-графики

Разработка системы веб-визуализации сложных производственно-технологических процессов морского порта предъявляет специфические требования из раздела 3, обусловленные особенностями транспортно-логистической деятельности. Следовательно, программный комплекс, решающий множество задач и обеспечивающий контроль управления процессами (включая все этапы грузовых операций), должен обладать следующими ключевыми характеристиками: адаптивность, масштабируемость, производительность, интегрированность, интерактивность и надежность.

Существует несколько способов работы с веб-графикой и каждый из них обладает своими достоинствами и недостатками:

1. Чистый HTML/CSS/JS (DOM) – прост в освоении и обеспечивает хорошую доступность и событийную модель, но обладает крайне низкой производительностью при большом числе объектов, ограниченными анимационными возможностями и сложностями с масштабированием, что делает его непригодным для реализации 2D-графики [18].

2. SVG предлагает векторное масштабирование без потерь, хорошую поддержку событий и доступность, однако его производительность резко падает при сложных сценах и частых обновлениях, а управление иерархией объектов громоздко. Он подходит лишь для статических или умеренно динамических схем средней сложности, но не для интенсивной визуализации порта в реальном времени [19].

3. Canvas API (низкоуровневый) обеспечивает высочайшую производительность и полный контроль над отрисовкой, идеален для анимации множества объектов. Однако его низкоуровневый API сложен, отсутствует встроенная объектная модель и поддержка событий на объектах (требуется ручная обработка), что затрудняет создание сложных интерактивных сцен и управление структурой производственно-технологического процесса. Хотя технология мощная, она требует значительных усилий для реализации требований [20].

4. WebGL дает максимальную производительность для сложной 2D/3D графики и эффектов, но имеет очень высокий порог вхождения (требует знаний шейдеров, математики), а разработка 2D-интерфейсов поверх него сложна. Он избыточен и неоправданно сложен для большинства задач 2D-визуализации портовых процессов, будучи оптимальным лишь для 3D или 2D с экстремальной нагрузкой [21].

5. Библиотеки поверх Canvas (Konva.js, Fabric.js и т.п.) предоставляют высокоуровневый, объектно-ориентированный API, решая главные проблемы чистого Canvas. Данный способ наиболее перспективный поскольку данные библиотеки способны инкапсулировать логику отрисовки в объекты (фигуры, группы), реализуют встроенную поддержку событий на уровне объектов, а также механизмы для трансформаций, анимации, иерархии и слоев, значительно ускоряя разработку и делая их

оптимальным решением для задач интерактивной визуализации промышленных объектов, подобных морскому порту [22-23].

Среди популярных высокоуровневых библиотек, работающих поверх `<canvas>`, лидируют `Konva.js` и `Fabric.js`. Оба экземпляра предоставляют необходимый уровень абстракции, объектную модель и интерактивность, критически важные для отображения и управления такими сложными сущностями, как суда, краны, контейнерные площадки и технологические линии в режиме реального времени

Выбор `Konva.js` для реализации системы визуализации производственно-технологических процессов морского порта был обусловлен его преимуществами в контексте специфических требований проекта.

Архитектура `Konva.js`, изначально построенная вокруг иерархии Stage - Layer - Group - Shape, идеально отражает логику производственно-технологических процессов морского порта. В этой структуре общая карта (Stage) состоит из слоев (Layer), представляющих ключевые инфраструктурные элементы порта, такие как причалы, железная дорога, автодороги и грузовые зоны. Эти слои, в свою очередь, содержат группы объектов (Group), например, кран вместе с его зоной работы или судно с контейнерами на нем, а также отдельные графические объекты (Shape), такие как контейнер, вагон или автомобиль.

Кроме того, `Konva.js` отличается от `Fabric.js` своей основной направленностью. В то время как `Fabric.js` изначально развивалась с акцентом на редактирование изображений и текста, а также интерактивный дизайн (включая поддержку панелей инструментов и более сложную работу с импортом/экспортом SVG), фокус `Konva.js` лежит именно на области производительной, интерактивной 2D-анимации и визуализации, что критически важно для данного проекта.

Для более точного понимания вышеизложенный анализ сведен в таблицу 1.

Таблица 1 – Сравнительный анализ технологий веб-визуализации для систем управления производственными процессами

Критерий	HTML/CSS/JS (DOM)	SVG	Canvas API	WebGL	Библиотеки Konva/Fabric.js
Производительность	Низкая (при большом числе объектов)	Средняя (падает при сложных сценах)	Высокая	Максимальная	Высокая (с оптимизацией)
Масштабируемость	Ограниченная	Хорошая	Высокая	Максимальная	Высокая
Интерактивность	Высокая (встроенная событийная модель)	Хорошая	Низкая (требует ручной реализации)	Низкая	Высокая (встроенные события)
Сложность разработки	Низкая	Средняя	Высокая	Очень высокая	Средняя (объектно-ориентированный API)
Поддержка анимации	Ограниченная	Средняя	Высокая	Максимальная	Высокая (встроенные механизмы)
Применимость для задач порта	Непригоден	Для статических схем	Для высокопроизводительной визуализации	Для 3D и экстремальных нагрузок	Оптимально для интерактивной визуализации
Объектная модель	Встроенная	Встроенная	Отсутствует	Отсутствует	Встроенная (Stage-Layer-Group-Shape)
Соответствие требованиям	Не соответствует	Частично соответствует	Частично соответствует	Избыточен	Полностью соответствует

5. Обоснование выбора инструментов для реализации веб-визуализации производственно-технологических процессов морского порта

На основе сформулированных ранее критериев (производительность, гибкость, интеграция), с фокусом на интерактивную 2D-визуализацию и реализацию метода drag-and-drop в контексте задач манипулирования объектами морского порта, выбран ряд инструментов.

Первым инструментом для решения поставленной задачи выступает библиотека React [24]. В данной библиотеке реализован компонентный подход на рисунке 2, который позволяет разбить сложную визуализацию на компоненты (причал, кран, судно, контейнер, слой), что соответствует требованию отображения статистического контента из раздела 3. Альтернативой библиотеке React выступают библиотеки Vue.js и Angular. Vue.js предлагает аналогичный компонентный подход и может выступать альтернативой [25], однако, экосистема React, в частности наличие библиотеки react-konva для работы с высокопроизводительной графикой, на данный момент представляется более развитой для задач подобного масштаба, что и стало решающим фактором в пользу выбора React. Angular предоставляет полноценную, строгую архитектуру «из коробки», но его высокая сложность и размер были избыточны для данной задачи, сфокусированной на высокопроизводительной графике [26]. React был выбран за оптимальное сочетание минимализма, производительности Virtual DOM и огромной экосистемы.

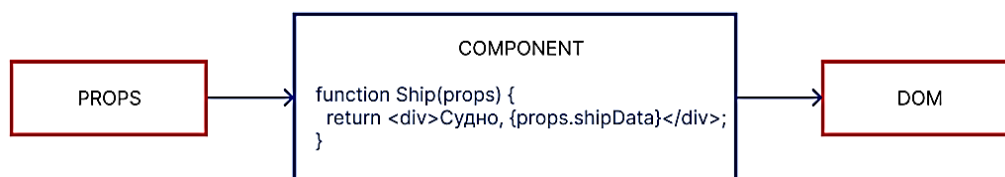


Рисунок 2 – Компонентный подход React

Для управления состоянием в React-приложениях используется библиотека Redux [27], которая не входит в состав React, но часто используется как дополнение и подходит для управления сложным, часто обновляемым состоянием множества объектов порта, что соответствует требованию отображения статистического и динамического контента из раздела 3. Изменение состояния автоматически приводит к перерисовке соответствующих компонентов.

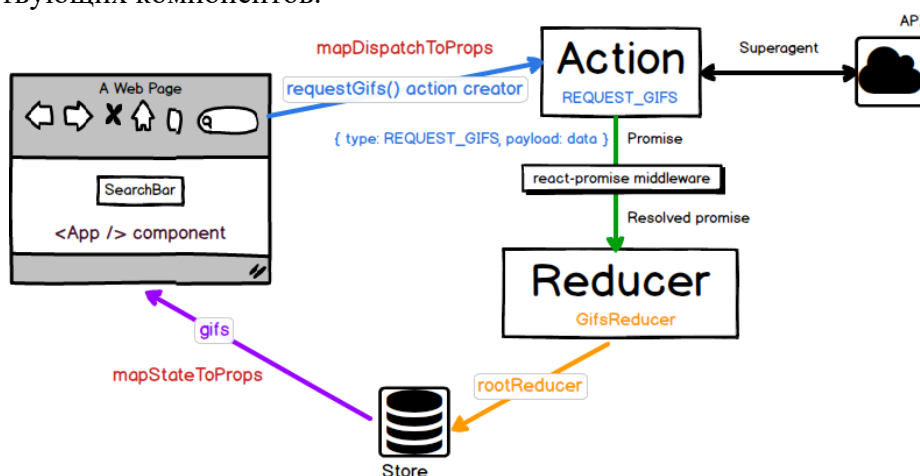


Рисунок 3 – Архитектура модульных React + Redux приложений [28]

Стоит отметить, что экосистема React предлагает и другие решения для управления состоянием. Встроенный Context API в сочетании с хуком `useReducer` подходит для приложений средней сложности, однако может оказаться менее производительным и предсказуемым при частых обновлениях большого количества взаимосвязанных данных, характерных для модели порта. Современные библиотеки, такие как Zustand, предлагают более простой и лаконичный API с меньшим количеством шаблонного кода (boilerplate), а MobX реализует реактивную парадигму, которая может быть интуитивно понятнее в некоторых сценариях [29-30].

Выбор Redux был обусловлен его масштабируемостью - строгая архитектура упрощает командную разработку и долгосрочную поддержку кода. Кроме того, его экосистема включает проверенные решения (middleware, например, Redux Thunk/Saga) для работы с асинхронными операциями и side-эффектами, что напрямую отвечает требованию интеграции с серверными API.

Вторым инструментом для решения поставленной задачи выступает библиотека Konva.js. Данная библиотека предоставляет объектно-ориентированную абстракцию над Canvas API, значительно упрощая работу со сложной графикой, такой как фигуры, группы и трансформации.

Требование к производительности и масштабируемости в разделе 3, предписывающее обеспечить плавную анимацию перемещений множества объектов (судов, кранов, контейнеров), напрямую удовлетворяется архитектурой Konva.js. Кэширование слоев (`layer.cache()`) и обновление только измененных областей (`layer.batchDraw()`), позволяют эффективно отрисовывать множество динамических объектов на схеме порта, сохраняя частоту кадров, необходимую для оперативного мониторинга.

Требования интерактивности в разделе 3 – выделение объектов, операции drag-and-drop – реализуются средствами Konva.js из «коробки». Встроенная система событий (`on('click')`, `on('mouseover')`, `on('dragmove')`) на уровне отдельных фигур (Rect, Circle, Group) прямо соответствует необходимости взаимодействия пользователя с конкретными судами, кранами или контейнерами. Например, обработчик `on('dragend')` на группе, представляющей кран, позволяет фиксировать его новое положение на причале после перемещения пользователем, реализуя функционал ручного планирования.

Konva.js обладает совместимостью с React через официальную библиотеку `react-konva`, которая создает мощную синергию для управления сложным состоянием производственно-технологических процессов морского порта и его визуализации. React-компонент `<Ship>`, управляемый состоянием Redux, инкапсулирует соответствующую Konva-группу (`<Group>`), содержащую фигуры корпуса (`<Rect>`), надстройки (`<Path>`) и меток (`<Text>`). При получении новых данных о позиции судна, обновление состояния Redux автоматически приводит к перерисовке компонента `<Ship>` с новыми пропсами `x` и `y`, что вызывает перемещение всей Konva-группы на схеме порта. Обратное, событие `onClick` на Konva-группе судна внутри компонента `<Ship>` может вызвать действие Redux `selectShip(id)`, обновляющее глобальное состояние и, например, открывающее сайдбар с детальной информацией – непосредственно реализуя требование интерактивного выделения и детализации объектов из раздела 3.

6. Принципы реализации метода drag-and-drop в контексте задач манипулирования объектами морского порта

Операция drag-and-drop (DnD) критически важна для ручного планирования и оперативной корректировки размещения объектов (кранов, контейнеров, техники) в

системе визуализации производственно-технологических процессов морского порта. Ее реализация на базе React и Konva.js основана на следующих принципах [31]. Перетаскивание активируется установкой `draggable={true}` для соответствующих Konva-узлов (объектов порта), при этом доступность DnD динамически контролируется через состояние приложения (например, запрет перемещения статичной инфраструктуры). Обработка жизненного цикла DnD использует встроенные события Konva.js: `dragstart` визуально выделяет объект и фиксирует начальную позицию; `dragmove` обеспечивает плавное перемещение объекта на холсте в реальном времени; `dragend` фиксирует конечную позицию. Ключевой шаг: новые координаты синхронизируются с глобальным состоянием (Redux Store), что инициирует перерисовку компонента и обновление данных в бэкенде (API). Одновременно проводится валидация позиции (границы зон, коллизии с препятствиями/объектами через методы Konva, например `shape.intersects()`), при невалидности объект возвращается или выводится предупреждение. Действие логируется. Для перемещения групп объектов (контейнер на погрузчике) используется Konva Group с `draggable={true}`. Управление состоянием строго следует принципу: единственный источник истины – глобальное состояние (Redux). Оптимизация обеспечивается обновлением только перемещаемого объекта во время `dragmove` (с помощью `layer.batchDraw()`) и кэшированием статичных слоев (`layer.cache()`). Интеграция через `react-konva` позволяет инкапсулировать логику DnD в компонентах: обработчики событий Konva вызывают обновление состояния Redux (`dispatch`), что приводит к автоматической перерисовке компонента с новыми координатами. Это обеспечивает удобную реализацию сложной интерактивности, необходимой для управления объектами морского порта.

Выводы

Разработка систем веб-визуализации сложных производственно-технологических процессов морского порта требует четко описанных требований и тщательного выбора технологий, обеспечивающих высокую производительность и интерактивность при работе с множеством динамических объектов.

Результат работы заключается в предложении основных требований к веб-визуализации производственно-технологических процессов морского порта. Проведен комплексный сравнительный анализ веб-технологий 2D-графики и обоснование выбора стека React, Redux и Konva.js как оптимального для задач визуализации производственно-технологических процессов, а также предложена реализация графического механизма `drag-and-drop` для интерактивного манипулирования объектами морского порта (краны, контейнеры), включая интеграцию событий Konva.js с глобальным состоянием приложения (Redux), валидацию позиций и логирование действий.

Данный метод работы планируется развивать как основу для создания интерактивного веб-приложения для повышения эффективности управления морским портом в рамках развития работ посвященным по направлению цифровым двойникам и цифровой индустрии промышленности.

Список литературы

1. Масюк Н.Н., Блюдик А.Р. Современные тенденции цифровой трансформации в морской области. *Естественно-гуманитарные исследования*. 2022. № 44(6). С. 203-207.
2. Вольнский И.А. Механизмы и инструменты управления развитием логистической инфраструктуры морских портов. *Вестник АГТУ*. 2022. №1. С. 78-83.

3. Румянцев В.В. О роли информационных технологий в развитии цивилизации. *Проблемы искусственного интеллекта*. 2021. №4(23). С.59-64.
4. Киселев А.О. Проблемы интеграции SCADA-систем с AS/RS на производствах. *Международный научный журнал «Вестник науки»*. 2025. №4(85). С. 788–893.
5. Ехлаков Ю.П., Жуковский О.И., Сенченко П.В., Гриценко Ю.Б., Милихин М.М. Базовые принципы разработки веб-ориентированных информационных систем управления инфраструктурой социально-экономических и технических процессов. *Доклады ТУСУРа*. 2017. №2. С. 63-67.
6. Анцыферов С.С., Фазилова К.Н., Русанов К.Е. Интеллектуальные системы управления технологическими процессами. *Проблемы искусственного интеллекта*. 2024. №2(33). С.37-44.
7. Романова И.К. Современные методы визуализации многомерных данных: анализ, классификация, реализация, приложения в технических системах. Наука и образование. МГТУ им. Н.Э. Баумана. 2016. №3. С. 133-167.
8. Калмакова Н.А. Уровни управления сбалансированным развитием промышленных предприятий на принципах самоорганизации. *Региональная экономика: теория и практика*. 2015. №27. С. 24-35.
9. Волынский И.А. Понятие, виды и особенности формирования потоковых процессов морского порта. *Вестник АГТУ*. 2020. №3. С. 71-78.
10. Ганин М.А., Лобанов Е.М. *Организация и технология перегрузочных процессов в морских портах*. СПб: Морсар, 2020.
11. Гаркуша Д.А. Метод наполнения и обработки базы знаний в задаче построения цифрового двойника. *ДМКС* 2025.
12. Луков Д.К. Автоматизированные системы управления технологическим процессом (АСУ ТП). *European science* №2(44). С. 19-21.
13. Шевченко А.М., Дыда А.А. Анализ реализации агентного подхода и особенности имитационного моделирования морского порта в среде AnyLogic. *Computational Nanotechnology*. 2024. №4. С. 135-145.
14. Прокопенко Е.В., Сарапулова Т.В. Разработка Web-приложений для поддержки стратегического управления. *Вестник Кузбасского государственного технологического университета*. 2011. С. 114-116.
15. Пчелкин А.Ю., Гусарова Н.Ф. Кроссплатформенная разработка на базе веб-технологий для поддержки решений в проблемно-ориентированных системах управления. *Экономика, право, инновации*. 2022. №1. С. 41-47.
16. Кондакова А.В., Золотухина Е.Б. *Обмен данными между облачным решением SAP ARIBA SOURSING и учетными системами с помощью технологии веб-сервисов*. [Электронный ресурс]. URL: <https://cyberleninka.ru/article/n/obmen-dannymi-mezhdu-oblachnym-resheniem-sap-ariba-sourcing-i-uchetnymi-sistemami-s-pomoschu-tehnologii-veb-servisov/viewer>.
17. Бурцев В.А. Проблемы и перспективы пользовательского опыта в цифровых платформах: теоретические подходы и значение для бизнес-приложений. *Инновации и инвестиции*. 2025. №4. С. 330-334.
18. Никсон Р. *Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML5*. 5-е изд. СПб.: БХВ-Петербург, 2022. 848 с.
19. Eisenberg J. D. *SVG Essentials: Producing Scalable Vector Graphics with XML*. 2nd ed. O'Reilly Media, 2014. 354 p.
20. *MDN Web Docs: Canvas API* [Электронный ресурс]. URL: https://developer.mozilla.org/ru/docs/Web/API/Canvas_API
21. Мацуда, К. *WebGL: программирование трехмерной графики*. Санкт-Петербург: Питер. 2016. 464 с.
22. *Konva.js: Официальная документация* [Электронный ресурс]. URL: <https://konvajs.org/>
23. *Fabric.js: Официальная документация* [Электронный ресурс]. URL: <https://fabricjs.com/>
24. *React.js: Официальная документация* [Электронный ресурс]. <https://react.dev>
25. *Vue.js: Официальная документация* [Электронный ресурс]. URL: <https://vuejs.org/guide>
26. *Angular: Официальная документация* [Электронный ресурс]. URL: <https://angular-doc.ru/docs>
27. *Redux: Официальная документация* [Электронный ресурс]. URL: <https://redux.js.org>
28. *Архитектура модульных React + Redux приложений* [Электронный ресурс]. URL: <https://habr.com/ru/articles/326484/>
29. *Zustand: Официальная документация* [Электронный ресурс]. URL: <https://zustand.docs.pmnd.rs/getting-started/introduction>
30. *Mobx: Официальная документация* [Электронный ресурс]. URL: <https://mobx.js.org/getting-started>
31. *React-DnD: Официальная документация* [Электронный ресурс]. URL: <https://react-dnd.github.io/react-dnd/docs/overview>

References

1. Masyuk, N.N. & Blyudik, A.R. (2022). Modern Trends of Digital Transformation in the Maritime Field. *Natural and Human Studies*, 44(6), 203–207.
2. Volynsky, I.A. (2022). Mechanisms and Tools for Managing the Development of Logistic Infrastructure of Seaports. *ASTU Herald*, (1), 78–83.
3. Rumyantsev V.V. On the role of information technologies in the development of civilization // *Problems of Artificial Intelligence*. 2021. №4(23). С. 59-64.
4. Kiselev, A.O. (2025). Problems of SCADA Systems Integration with AS/RS in Manufacturing. *International Scientific Journal "Herald of Science"*, 4(85), 788–893.
5. Yekhlakov, Yu.P., Zhukovsky, O.I., Senchenko, P.V., Gritsenko, Yu.B. & Milikhin, M.M. (2017). Basic Principles of Developing Web-Oriented Information Management Systems for Socio-Economic and Technical Processes Infrastructure. *TUSUR Reports*, (2), 63–67.
6. Antsyferov S.S., Fazilova K.N., Rusanov K.E. Intelligent systems process control // *Problems of Artificial Intelligence*. 2024. №2(33). С. 37-44.
7. Romanova, I.K. (2016). Modern Methods of Multidimensional Data Visualization: Analysis, Classification, Implementation, Applications in Technical Systems. *Science and Education*. Bauman Moscow State Technical University, (3), 133–167.
8. Kalmakova, N.A. (2015). Management Levels for Balanced Development of Industrial Enterprises Based on Self-Organization Principles. *Regional Economics: Theory and Practice*, (27), 24–35.
9. Volynsky, I.A. (2020). Concept, Types and Features of Flow Process Formation in a Seaport. *ASTU Herald*, (3), 71–78.
10. Ganin, M.A. & Lobanov, E.M. (2020). Organization and Technology of Handling Processes in Seaports. St. Petersburg: Morsar.
11. Garkusha, D.A. (2025). A Method for Filling and Processing a Knowledge Base in the Task of Building a Digital Twin. *DMKC 2025*.
12. Lukov, D.K. Automated Process Control Systems (APCS). *European Science*, 2(44), 19–21.
13. Shevchenko, A.M. & Dyda, A.A. (2024). Analysis of the Agent-Based Approach Implementation and Features of Seaport Simulation Modeling in the AnyLogic Environment. *Computational Nanotechnology*, (4), 135–145.
14. Prokopenko, E.V. & Sarapulova, T.V. (2011). Development of Web Applications to Support Strategic Management. *Herald of Kuzbass State Technical University*, 114–116.
15. Pchelkin, A.Yu. & Gusarova, N.F. (2022). Cross-Platform Development Based on Web Technologies to Support Decision-Making in Problem-Oriented Management Systems. *Economics, Law, Innovation*, (1), 41–47.
16. Kondakova, A.V. & Zolotukhina, E.B. Data Exchange Between the Cloud Solution SAP Ariba Sourcing and Accounting Systems Using Web Services Technology. [Electronic resource]. Retrieved from <https://cyberleninka.ru/article/n/obmen-dannymi-mezhdu-oblachnym-resheniem-sap-ariba-sourcing-i-uchetnymi-sistemami-s-pomoschyu-tehnologii-veb-servisov/viewer>
17. Burtsev, V.A. (2025). Problems and Prospects of User Experience in Digital Platforms: Theoretical Approaches and Significance for Business Applications. *Innovation and Investment*, (4), 330–334.
18. Nixon, R. (2022). *Learning PHP, MySQL, JavaScript, CSS & HTML5: Building Dynamic Websites*. 5th ed. St. Petersburg: BHV-Petersburg. 848 p.
19. Eisenberg, J.D. (2014). *SVG Essentials: Producing Scalable Vector Graphics with XML*. 2nd ed. O'Reilly Media. 354 p.
20. MDN Web Docs: Canvas API. [Electronic resource]. Retrieved from https://developer.mozilla.org/ru/docs/Web/API/Canvas_API
21. Matsuda, K. & Lea, R. (2016). *WebGL Programming Guide: Interactive 3D Graphics Programming with WebGL*. St. Petersburg: Piter. 464 p.
22. Konva.js: Official Documentation. [Electronic resource]. Retrieved from <https://konvajs.org/>
23. Fabric.js: Official Documentation. [Electronic resource]. Retrieved from <https://fabricjs.com/>
24. React.js: Official Documentation. [Electronic resource]. Retrieved from <https://react.dev>
25. Vue.js: Official Documentation. [Electronic resource]. Retrieved from <https://vuejs.org/guide>
26. Angular: Official Documentation. [Electronic resource]. Retrieved from <https://angular-doc.ru/docs>
27. Redux: Official Documentation. [Electronic resource]. Retrieved from <https://redux.js.org>
28. Architecture of Modular React + Redux Applications. [Electronic resource]. Retrieved from <https://habr.com/ru/articles/326484/>
29. Zustand: Official Documentation. [Electronic resource]. Retrieved from <https://zustand.docs.pmnd.rs/getting-started/introduction>
30. Mobx: Official Documentation. [Electronic resource]. Retrieved from <https://mobx.js.org/getting-started>
31. React-DnD: Official Documentation. [Electronic resource]. Retrieved from <https://react-dnd.github.io/react-dnd/docs/overview>

RESUME

A. A. Levashov, D. A. Garkusha

Visualization of production and technological processes of the sea port

The article is devoted to the development and substantiation of web visualization methods for managing the production and technological processes of a seaport. The paper formulates a set of system requirements focused on the tasks of operational, tactical and strategic management, and provides a comparative analysis of web technologies for interactive 2D graphics. Based on the analysis, the choice of the React, Redux and technology stack is justified Konva.js. The key problems and features of the implementation are considered. Based on this, solutions are proposed.:

To ensure high performance when rendering a variety of dynamic objects (ships, machinery, containers), a high-level framework was chosen instead of the low-level Canvas API or low-performance SVG and DOM Konva.js, which provides an object model, built-in event support, and optimization mechanisms (layer caching, batchDraw).

Instead of the built-in React tools (Context API) or other libraries (MobX), the Redux library is used to manage the complex state of many interconnected port objects, which is frequently and predictably updated. Its strict architecture ensures scalability, state predictability, and convenient integration with middleware for asynchronous operations.

To implement a drag-and-drop operation that is critical for planning, the user interacts with an object on the canvas (events Konva.js) is directly synchronized with the global application state (Redux). This ensures that the only source of truth about the object's position remains the state, which is necessary for validating positions, logging, and sending data to the server.

Built-in methods are used to ensure the validation of movement operations (checking zone boundaries, collisions) directly during the dragging process. Konva.js such as `shape.intersects()`, which allows you to provide instant feedback to the user.

As a result, the following conclusions can be drawn.

A hybrid approach is used to visualize complex objects, such as a seaport: the Konva graphics library.js provides high—performance rendering and interactivity, while React and Redux provide declarativeness and state management. This allows you to create scalable applications with the ability to directly manipulate objects in real time.

A promising direction is integration with digital twin and AI systems for logistics optimization.

РЕЗЮМЕ

А. А. Левашов, Д. А. Гаркуша

Визуализация производственно-технологических процессов морского порта

Статья посвящена обзору методов веб-визуализации для управления производственно-технологическими процессами морского порта. В работе сформулирован комплекс требований к системе, ориентированный на задачи оперативного, тактического и стратегического управления, и проведён сравнительный анализ веб-технологий для интерактивной 2D-графики. На основе анализа обоснован выбор стека технологий React, Redux и Konva.js. Рассмотрены ключевые проблемы и особенности реализации. На основе этого предложены решения:

Для обеспечения высокой производительности при отрисовке множества динамических объектов (суда, техника, контейнеры) вместо низкоуровневого Canvas API или малопроизводительных SVG и DOM выбран высокоуровневый фреймворк Konva.js, который предоставляет объектную модель, встроенную поддержку событий и механизмы оптимизации (кэширование слоев, batchDraw).

Для управления сложным состоянием множества взаимосвязанных объектов порта, которое часто и предсказуемо обновляется, вместо встроенных средств React (Context API) или иных библиотек (MobX) применена библиотека Redux. Её строгая архитектура обеспечивает масштабируемость, предсказуемость состояния и удобную интеграцию с middleware для асинхронных операций.

Для реализации критически важной для планирования операции drag-and-drop взаимодействие пользователя с объектом на холсте (события Konva.js) напрямую синхронизируется с глобальным состоянием приложения (Redux). Это гарантирует, что единственным источником истины о положении объекта остается состояние, что необходимо для валидации позиций, логирования и отправки данных на сервер.

Для обеспечения валидации операций перемещения (проверка границ зон, коллизий) непосредственно в процессе перетаскивания используются встроенные методы Konva.js, такие как `shape.intersects()`, что позволяет предоставлять пользователю мгновенную обратную связь.

В результате можно сделать следующие выводы.

Для визуализации сложных объектов, таких как морской порт, применяется гибридный подход: графическая библиотека Konva.js обеспечивает производительный рендеринг и интерактивность, а React и Redux — декларативность и управление состоянием. Это позволяет создавать масштабируемые приложения с возможностью прямого манипулирования объектами в реальном времени.

Перспективное направление - интеграция с системами цифрового двойника и ИИ для оптимизации логистики.

Левашов А. А. – младший научный сотрудник, ФГБНУ “Институт проблем искусственного интеллекта”, 283048, Донецк, ул. Артема, 118б, тел +7(949) 595-8245, artur.levashoff@gmail.com. *Область научных интересов:* визуализация данных, нейронные сети.

Гаркуша Д. А. – младший научный сотрудник, ФГБНУ “Институт проблем искусственного интеллекта”, 283048, Донецк, ул. Артема, 118б, тел +7(977) 016-9581, dmitrii.garkusha.97@mail.ru. *Область научных интересов:* базы знаний, цифровые двойники, нейронные сети.

Статья поступила в редакцию 21.04.2025.