Назначение UI движка

Данная система является компонентом системы диспетчеризации.

Посредством данного компонента осуществляются запросы на получение данных сущностей, выполнение функций, получение/генерирование событий системы диспетчеризации.

Цель

- Дать возможность разработчикам модулей инструмент для более быстрого и удобного создания интерфейсов работы с модулем.
- Обеспечить единый вид и работу интерфейсам системы в условиях разработки модулей разными командами и постепенного добавления функционала.
- Дать возможность конечным пользователям частично настраивать внешний вид интерфейсов

Требования к системе управления графическими интерфейсами

Интерфейс должен автоматически адаптироваться под мобильные клиенты.

Графические элементы должны иметь возможность автоматически настраиваться в соответствии с сущностями модуля. Каждый модуль при регистрации в системе передает ядру описания своих сущностей. И при добавлении на страницу, например, таблицы с данными, разработчик указывает, что эта таблица будет показывать данные определенной сущности, и все колонки и фильтры будут добавлены автоматически в описание таблицы.

Интерфейс должен реагировать на изменения данных и на события в системе (список событий предоставляется с каждой формой)

Должны поддерживаться такие графические элементы как:

- Заголовок панели/окна
- Функциональные кнопки (икнока, текст)
- Дерево элементов (иконки, текст)
- Таблица данных
- Фильтры данных
- Графики
- Формы для ввода данных
- Элементы формы данных: текстовое поле, поле ввода даты, чекбокс, ввод чисел
- Модальные окна
- Окно сообщений о событиях в системе
- Панели с настраиваемым расположением
- List View (элементы списка в виде иконок)
- Вкладки
- Изображения

Элементы могут располагаться как в отдельных панелях, так и свободно в любом месте страницы

Должны поддерживаться скрипты (javascript) для задания логики работы интерфейса.

Если существующего функционала не хватает, то разработчикам модулей должна быть предоставлена возможность добавления кастомных элементов интерфейса с возможностью использования дополнительных библиотек. При этом должна быть обеспечена безопасность выполнения внешних скриптов.

Каждый пользователь имеет возможность настроить стили и размеры отображения элементов под свои нужды (цвет текста, фона, размер шрифта).

Должна быть возможность подключать отображение внешних интерфейсов (панели из grafana, отдельные интерфейсы модулей) через, например iframe элементы.

Конструктор интерфейсов

Пользователи конструктора: разработчики модулей, администраторы системы

Конструктор использует модель WYSIWYG для визуального создания и редактирования интерфейсов.

Используя данные о сущностях модуля, позволяет быстро добавлять необходимые элементы на экране.

Пример построения:

1. Пользователь создает новый интерфейс - создается пустой шаблон из двух панелей.
2. Пользователь добавляет элемент "таблица" в свободную панель.
3. Пользователь выбирает из списка сущностей модуля сущность "наличие на складах" и привязывает ее к таблице.
4. В таблице автоматически создаются колонки (№ единицы, марка, плавка, геометрия, вес, позиция на складе).
5. Автоматически создаются фильтры для данной таблицы.
6. Автоматически создается привязка data-source таблицы к нужной функции модуля.

Пользователь может привязывать javascript обработчики к функциональным кнопкам.

Если разработчику не хватает функционала элементов конструктора, он может добавить кастомный элемент в конструкторе.

Кастомные элементы

Конструктор предоставляет возможность добавления и описания кастомных элементов.

Формат описания кастомных элементов зависит от выбранного front-end движка. Это может быть как чистый HTML с CSS правилами, так и шаблоны React, Vue, Angular.

Внешние интерфейсы

Для сложных систем, где необходимы дополнительные возможности (мнемосхемы, сложные графики мониторинга) необходимо предусмотреть возможность подключения внешних (по отношению к ядру) систем построения интерфейсов.

Grafana

Одним из внешних интерфейсов может быть grafana как отдельный сервис. Её дашборды или панели могут быть включены в интерфейс ядра через iframe элементы.

Интерфейсы функциональных модулей

Модуль может включать в себя готовый движок для отображения интерфейсов, который может быть так же доступен через iframe элемент в общем интерфейса ядра.

При постепенном развитии библиотеки компонентов стандартного интерфейса ядра, будет отпадать необходимость во внешних интерфейсах.

Пользователи компонента

Разработчики модуля

Используют визуальный конструктор в процессе разработки модуля для создания интерфейса. В результате должно получиться описание интерфейса со всеми скриптами и настройками в виде одного файла, который будет передан ядру.

Разработчик должен иметь возможность подключать другие javascript библиотеки и создавать HTML разметку, если ему не хватает готовых элементов. В этом случае нужно по возможности обеспечить стандартизированный вид элементов и возможность настройки стиля обычными пользователями.

Администраторы системы

Имеют возможность использовать визуальный конструктор для изменения существующих в системе форм.

Настраивают права доступа пользователей к определенным формам и функциональным кнопкам. Не все формы одного модуля могут быть доступны всем пользователям. Часть функционала может быть доступна, например, только администраторам.

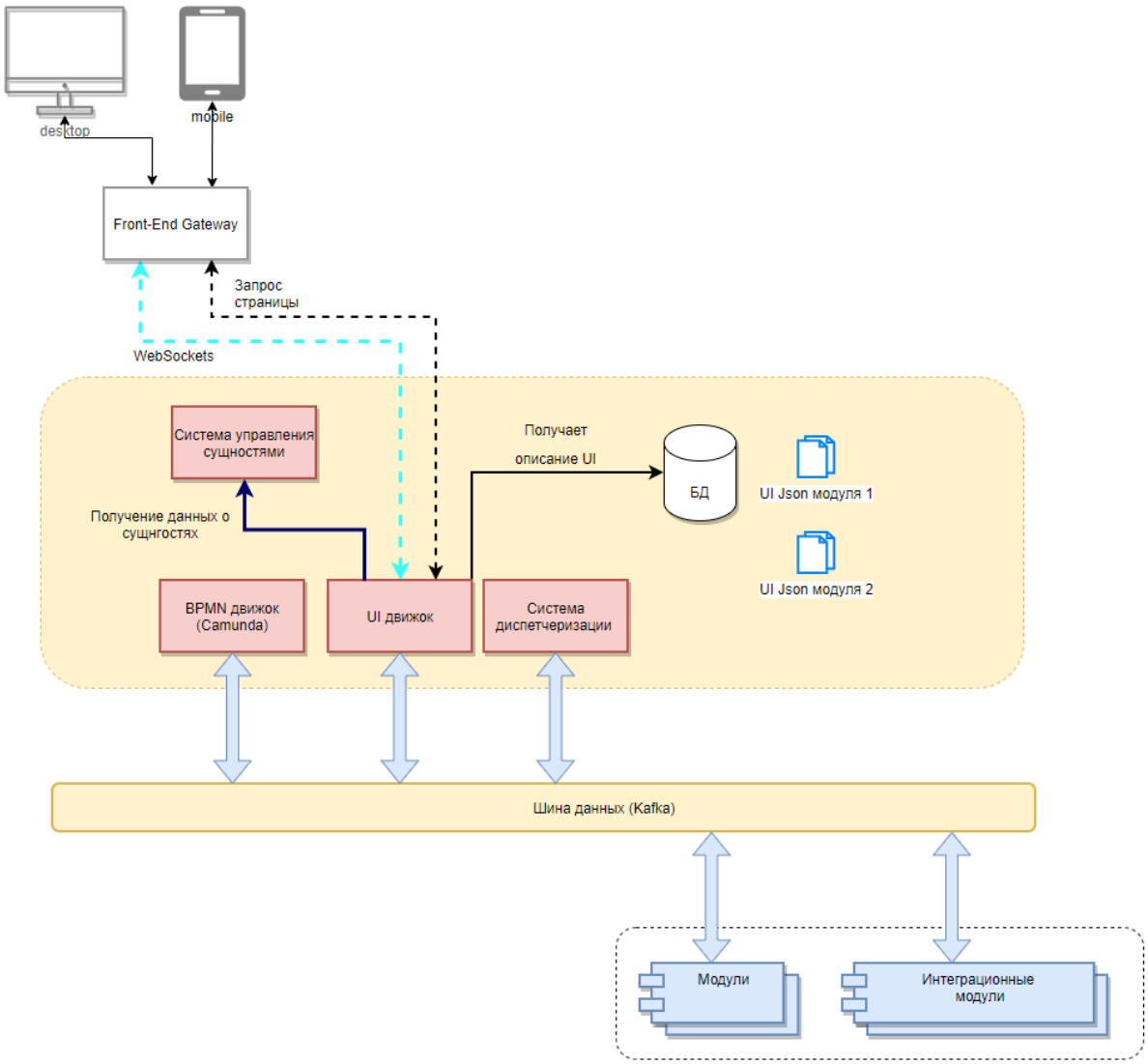
Имеют возможность создавать интерфейсы с нуля, используя функционал подключенных модулей

Обычные пользователи

Имеют возможность настраивать вид интерфейса под свои предпочтения:

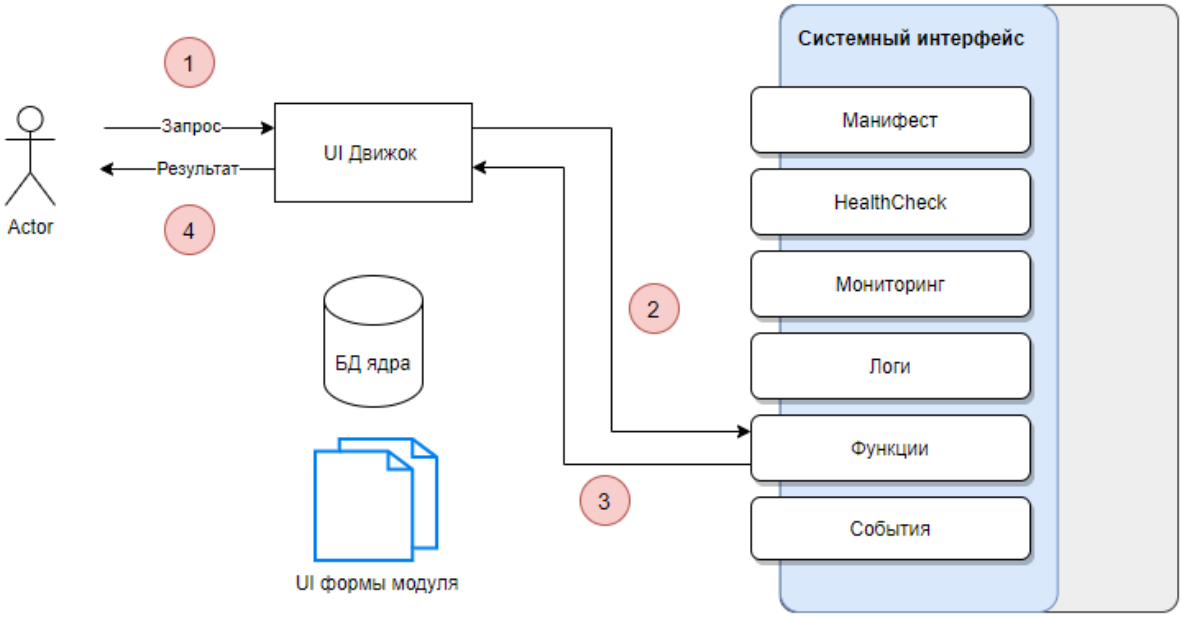
- Изменить основные цвета (цвет background, цвет текста)
- Изменить состав, размер, расположение колонок в таблицах
- Изменить расположение и размер модальных окон
- Изменить размер шрифтов

Архитектура



Процесс работы интерфейса

1. Пользователь отправляет запрос (запрашивает форму или наживает на функциональную кнопку).
2. Движок посылает запрос на выполнение функции модуля через систему коммуникации.
3. Модуль выполняет функцию и отправляет результат.
4. Движок получает результат и формирует ответ в браузер клиента.

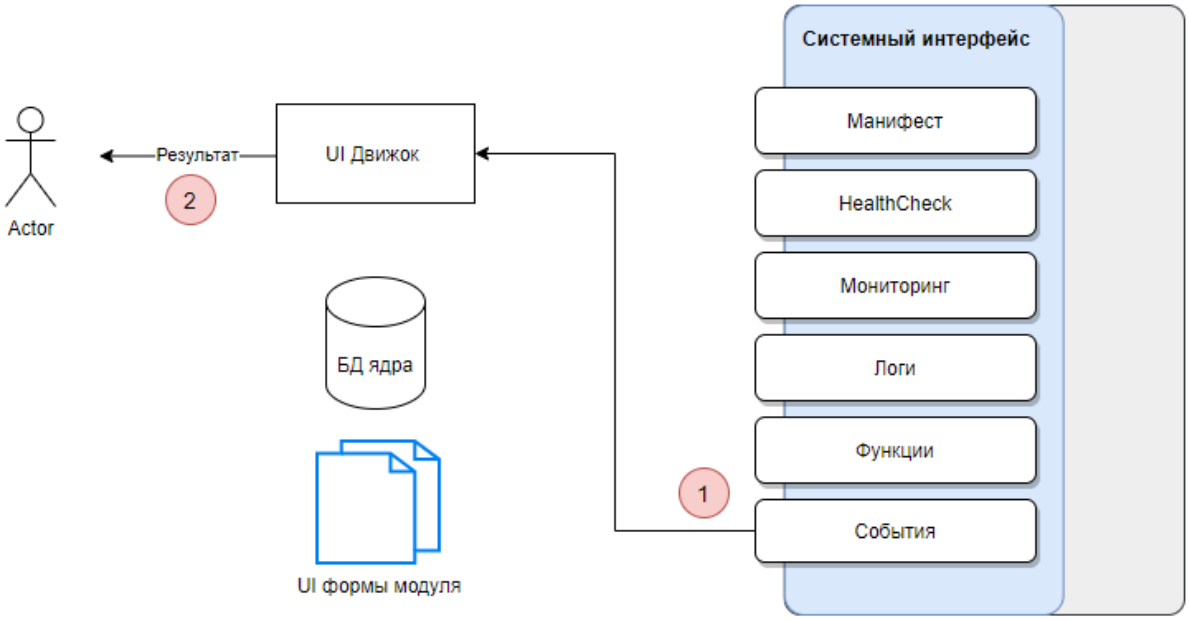


Процесс выполнения реакции на события

Для обеспечения реакции интерфейса на различные события в системе, движок должен уметь подписываться на события и соответственно обновлять интерфейс.

События, на которые должен реагировать интерфейс, описаны в соответствующих формах.

1. Модуль генерирует событие и отправляет его в шину данных
2. Движок, будучи подписанным на определенные события, ловит это событие и отправляет результат в браузер.


Системный интерфейс
Манифест
HealthCheck
Мониторинг
Логи
Функции
События
1
UI Движок
Результат
2
Actor
БД ядра
UI формы модуля