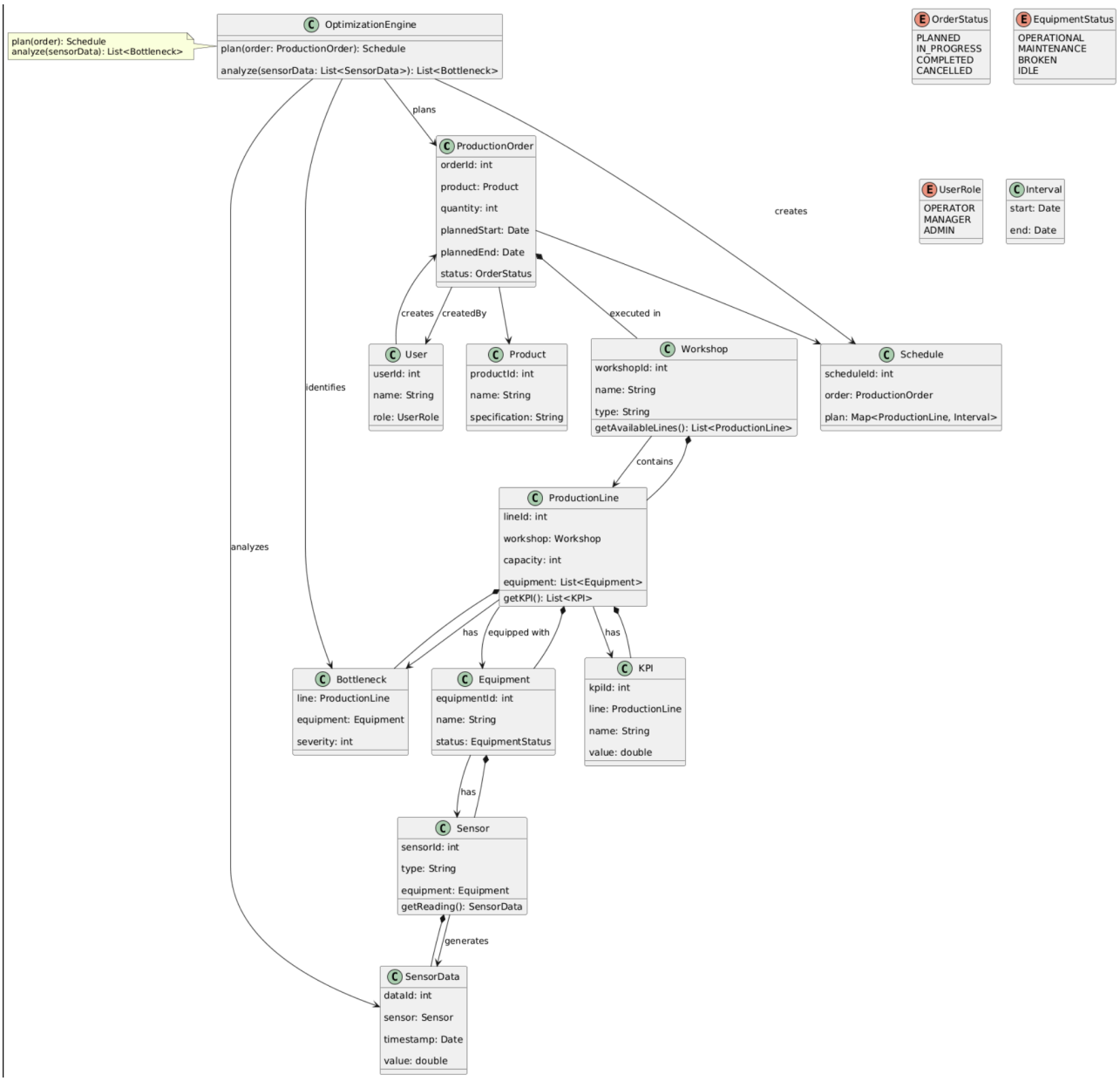
Интеллектуальная система оптимизации производственных процессов

A. Базовая диаграмма классов системы

1. Диаграмма классов модели



В качестве редактора для диаграмм классов был использован PlantUML, поэтому некоторые обозначения могут не совпадать со стандартными.

Пояснения:

Функционал расширения и "точки изменения" явно не отображены — базовая структура устойчива к доработкам (применится принцип открытости/закрытости позже).

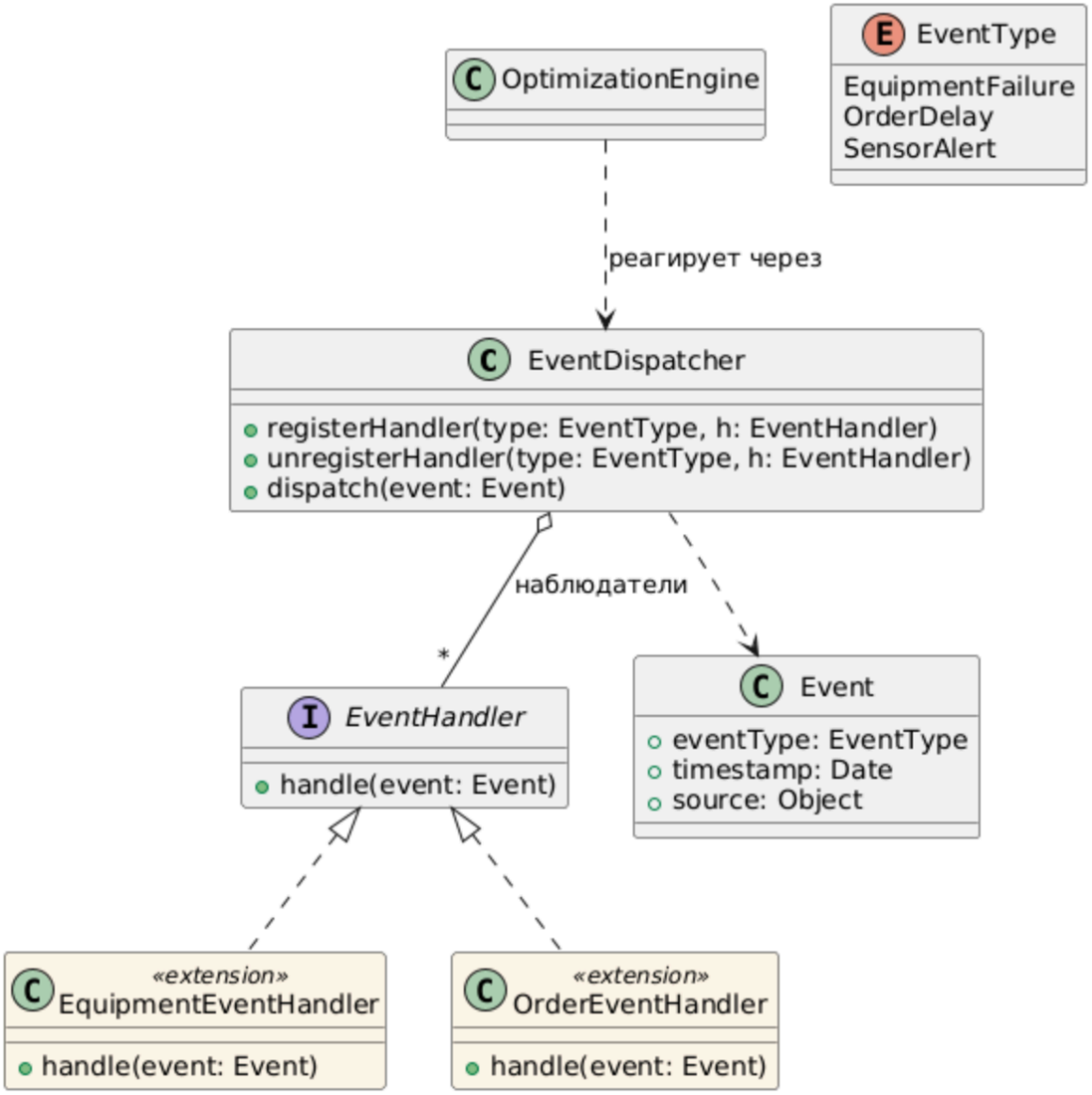
Все функции развития будут добавляться через наследование, композицию или внедрение новых интерфейсов.

B. Диаграммы классов для направлений развития

2.1. Первое направление: Реактивность и события в реальном времени

Задача: Система оперативно реагирует на события (аварии, сбои, изменения состояния оборудования и заказов).

- Open/Closed Principle: Новые реактивные обработчики добавляются через расширение (наследование/композиция), базовые классы не модифицируются.
- Классы-обработчики событий и диспетчер событий. Расширяемые методы для регистрации обработчиков. Вызов — через диспетчер событий, создает новые обработчики.

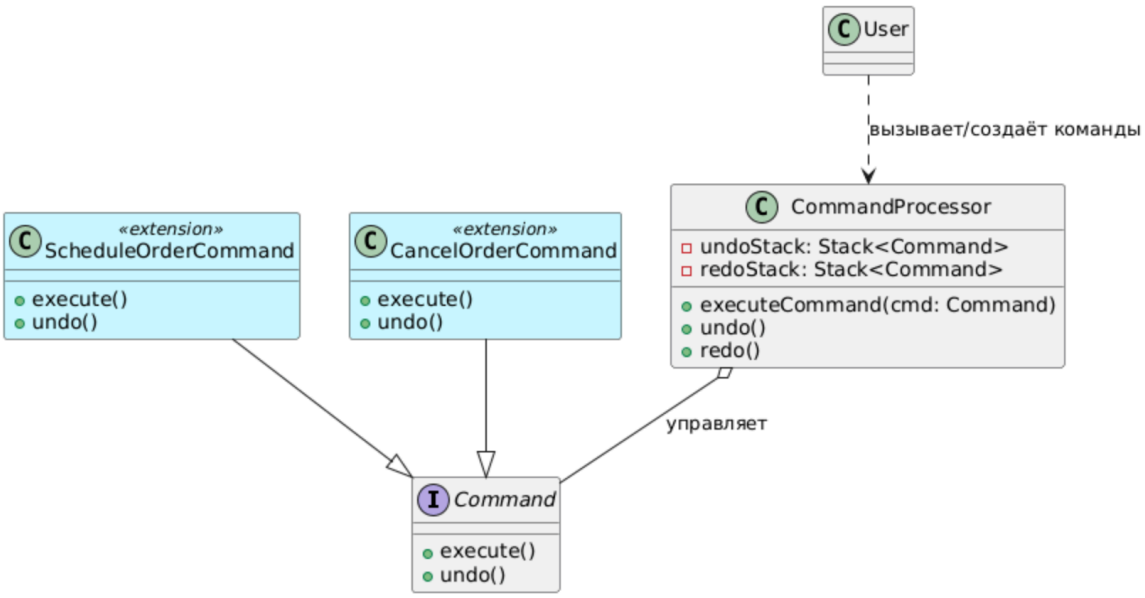


- Новые типы событий и обработчики реализуются как новые классы по интерфейсу EventHandler.
- Базовые классы системы не изменяются (SOLID OCP).
- Вызовы: диспетчер событий создает новые экземпляры обработчиков; методы вызываются из OptimizationEngine или подсистем мониторинга.

2.2. Второе направление: Управление действиями (undo/redo, очереди)

Задача: Система обрабатывает действия через шаблон Command, поддерживает отмену, повтор, очередь и аудит.

- Open/Closed Principle: Новые команды могут быть добавлены через наследование от команды, без изменения базового CommandProcessor.
- Пояснение изменений: Separate классы для команд, реестр команд и обработчик истории. Вызов — через CommandProcessor, создание — из контроллера/планировщика.

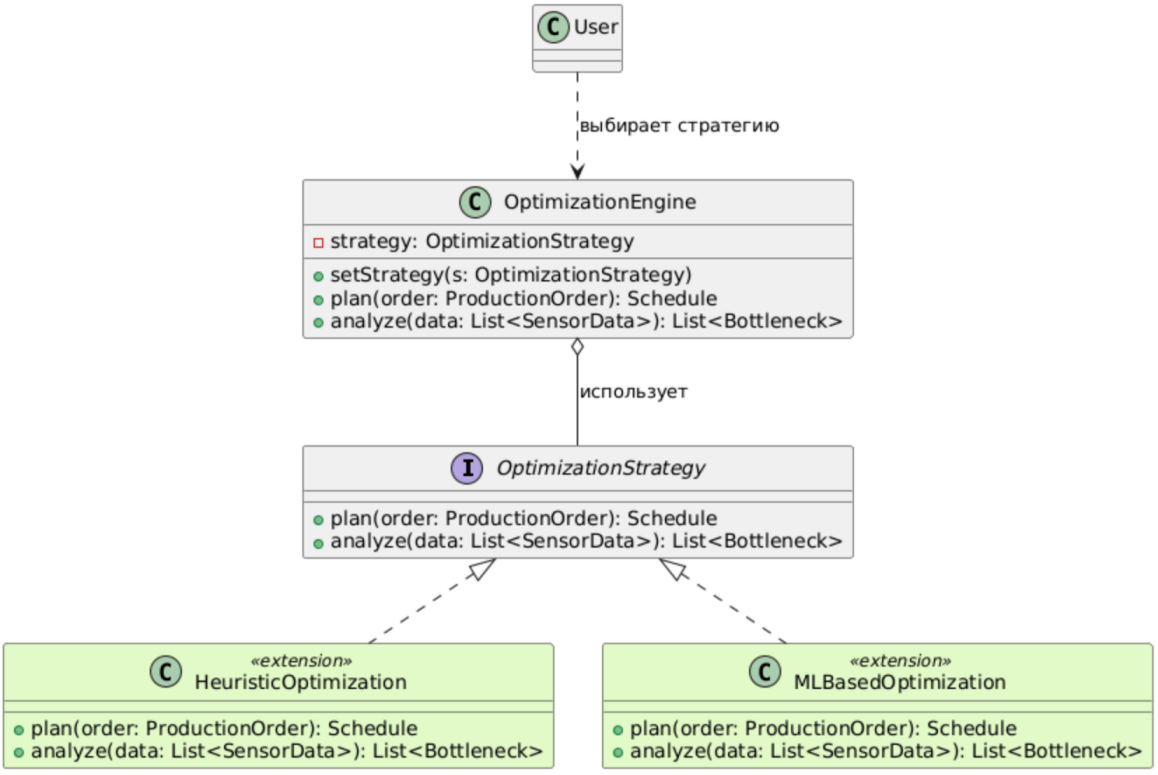


- Каждый новый тип действия (например, перенос заказа, блокировка линии) — отдельный класс-команда.
- CommandProcessor не меняется при добавлении новых действий (OCP).
- Пользователь (интерфейс, планировщик, автоматизация) создает и вызывает команды через CommandProcessor.

2.3. Третье направление: Плагины — расширяемые алгоритмы оптимизации

Задача: Позволяет пользователям/разработчикам выбирать, внедрять и сравнивать новые алгоритмы оптимизации для производственных процессов.

- Open/Closed Principle: Оптимизационные алгоритмы реализуют единый интерфейс, новые варианты — новые классы, существующий OptimizationEngine не изменяется.
- Пояснение изменений: Интерфейс OptimizationStrategy, различные потомки. Пользователь/инженер выбирает или внедряет новые алгоритмы. OptimizationEngine становится стратегическим контекстом.



- Каждый новый алгоритм оптимизации — отдельный класс по интерфейсу OptimizationStrategy.
- Пользователь или инженер может "подключить" новый класс, не меняя OptimizationEngine (строго OCP).

- OptimizationEngine делегирует работу выбранной стратегии (паттерн Стратегия), меняет их динамически, через setStrategy().