Унифицированный процесс разработки ПО

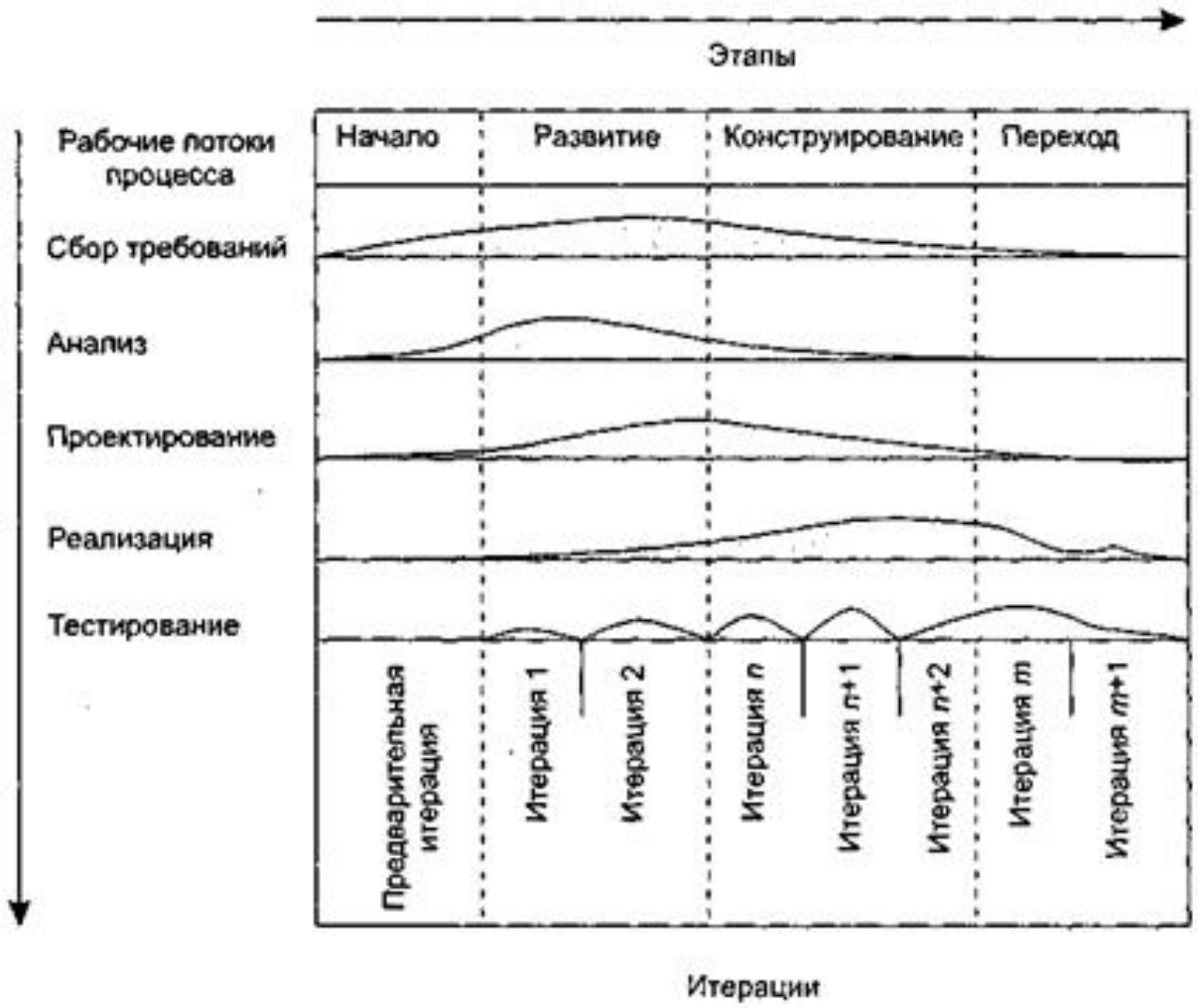
Технологии разработки программного обеспечения

Виноградова М.В.
МГТУ им. Н.Э. Баумана
Кафедра СОИУ (ИУ5)

RUP - Rational Unified Process

- Использует UML (визуальное проектирование);
- Поддерживается Rational / IBM;
- Управляется прецедентами
- Ориентирован на архитектуру
- Итеративность и инкрементность

График RUP


Этапы
Рабочие потоки процесса
Начало
Развитие
Конструирование
Переход
Сбор требований
Анализ
Проектирование
Реализация
Тестирование
Предварительная итерация
Итерация 1
Итерация 2
Итерация n
Итерация n+1
Итерация n+2
Итерация m
Итерация m+1
Итерации

Этапы RUP

- **Начало** - спецификация представления продукта;
- **Развитие** - планирование действий и требуемых решений;
- **Конструирование** - построение ПО в виде серии инкрементных итераций;
- **Переход** - внедрение ПО в среду пользователя (промышленное производство, доставка и применение)

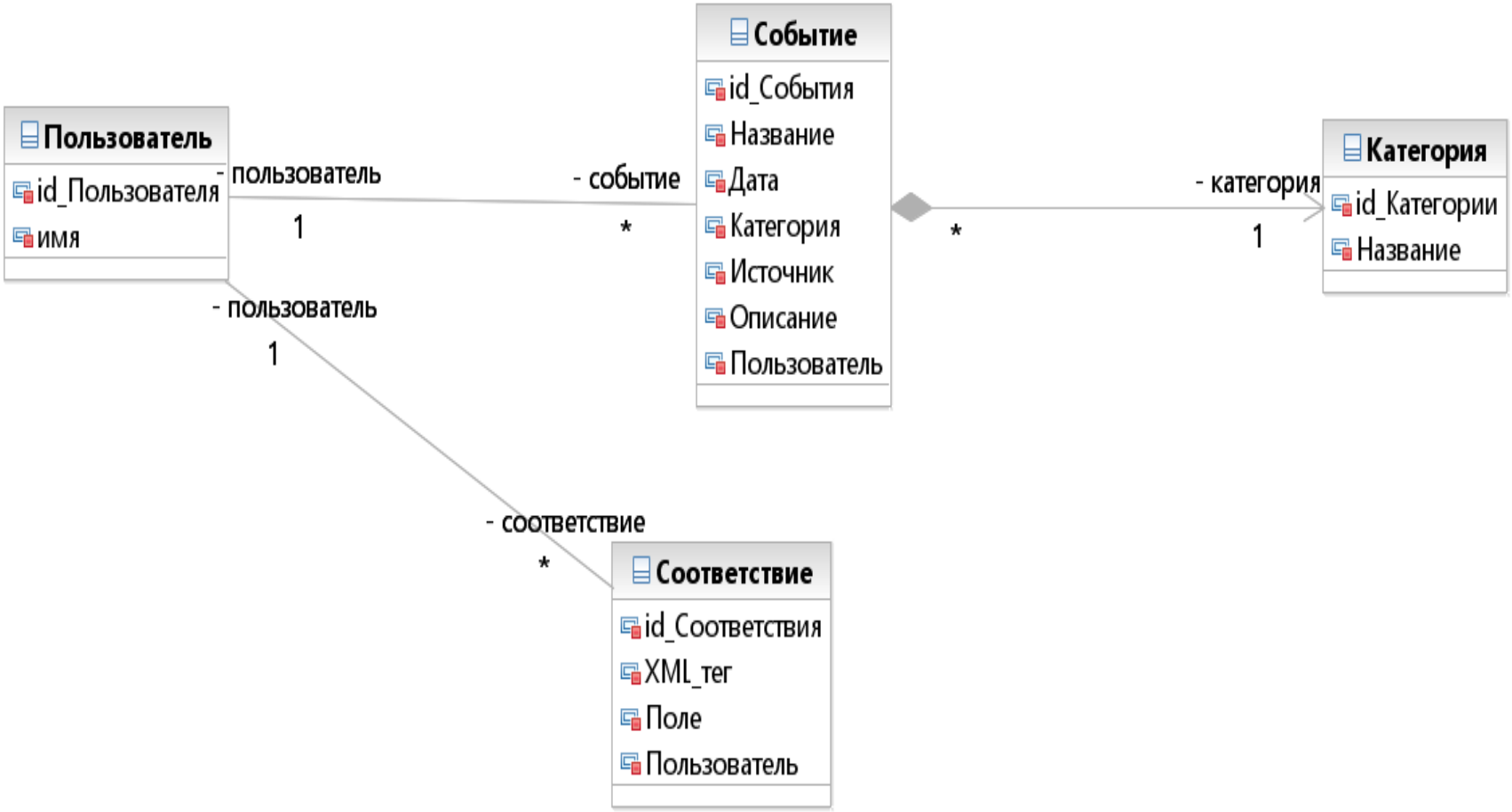
Рабочие процессы RUP

- Сбор требований - что делает система;
- Анализ - преобразование требований в классы и объекты предметной области;
- Проектирование - создание статического и динамического представления системы для выполнения требований;
- Реализация - производство программного кода;
- Тестирование - проверка системы в целом.

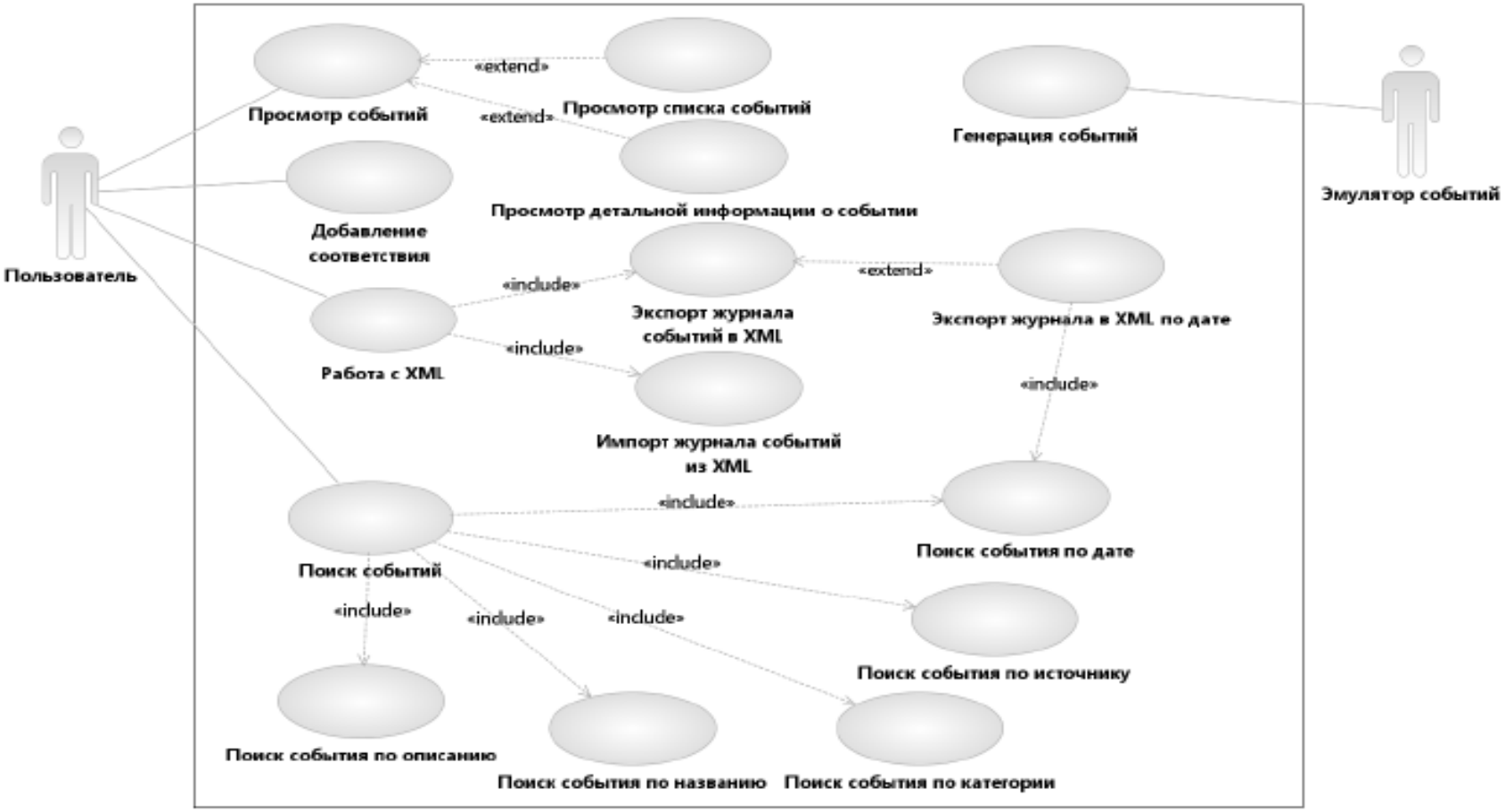
Рабочий процесс Определение требований

- Перечисление кандидатов в требования
- Осознание контекста системы
- Определение функциональных требований (в виде прецедентов)
- Определение нефункциональных требований

Модель предметной области – пример ЖСС



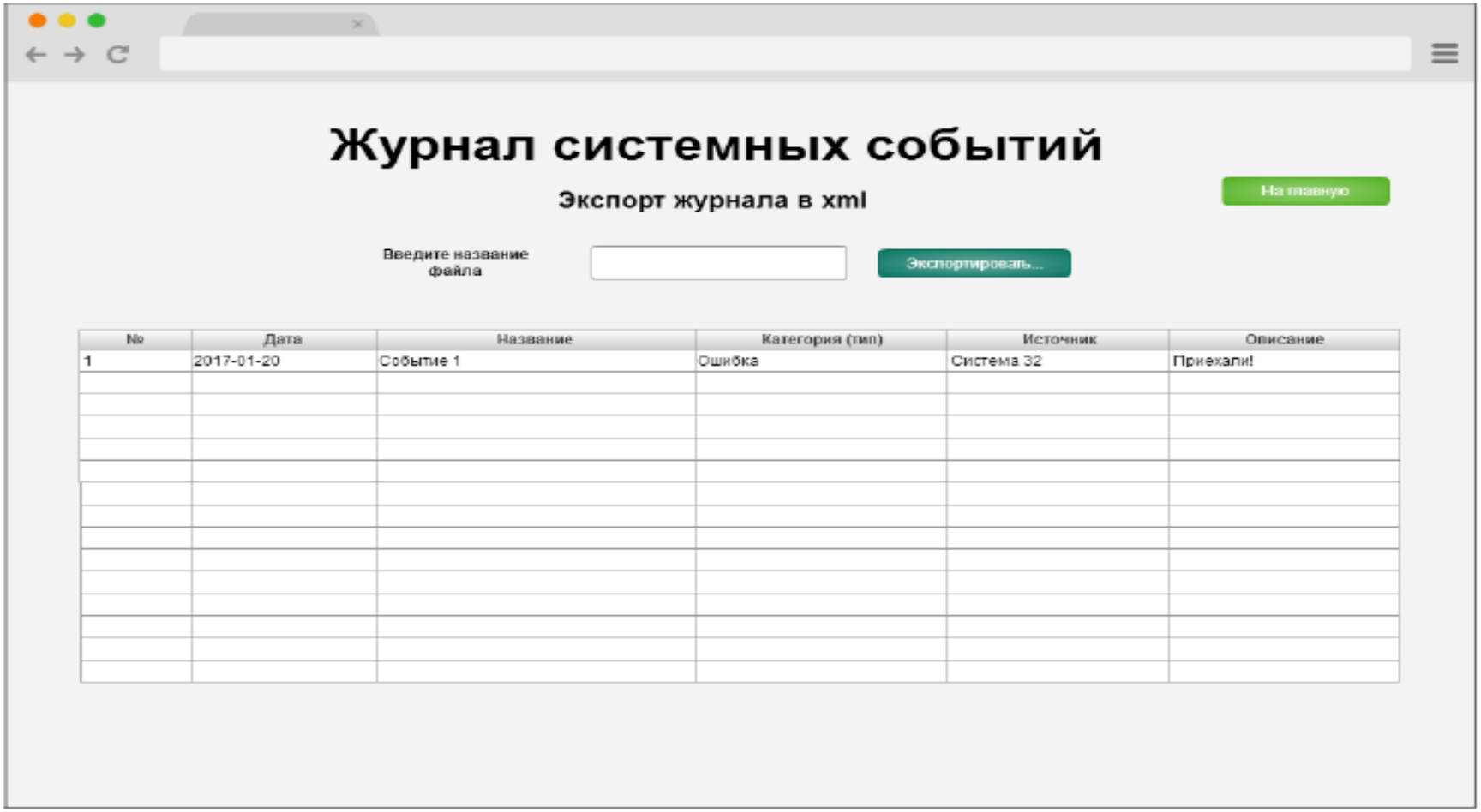
Модель прецедентов – пример ЖСС



Спецификация прецедента – пример ЖСС

Краткое описание: Предназначен для экспорта журнала событий в документ с расширением XML
Главные актеры: Пользователь
Второстепенные актеры: Нет
Предусловие: 1. Выполнен прецедент Работа с XML
Основной поток: 1. Прецедент начинается после выбора пользователем 2. Система проверяет введенные параметры пользователем 3. Система формирует пользователю документ в формате XML с информацией о событиях из БД 4. Система загружает документ в формате XML пользователю 5. Система отображает документ в формате XML пользователю
Постусловие: Нет
Альтернативные потоки: 1. Прецедент начинается после выбора пользователем 2. Система проверяет введенные параметры пользователем 3. Система выдает ошибку пользователю
Прецедент: Импорт журнала событий из XML

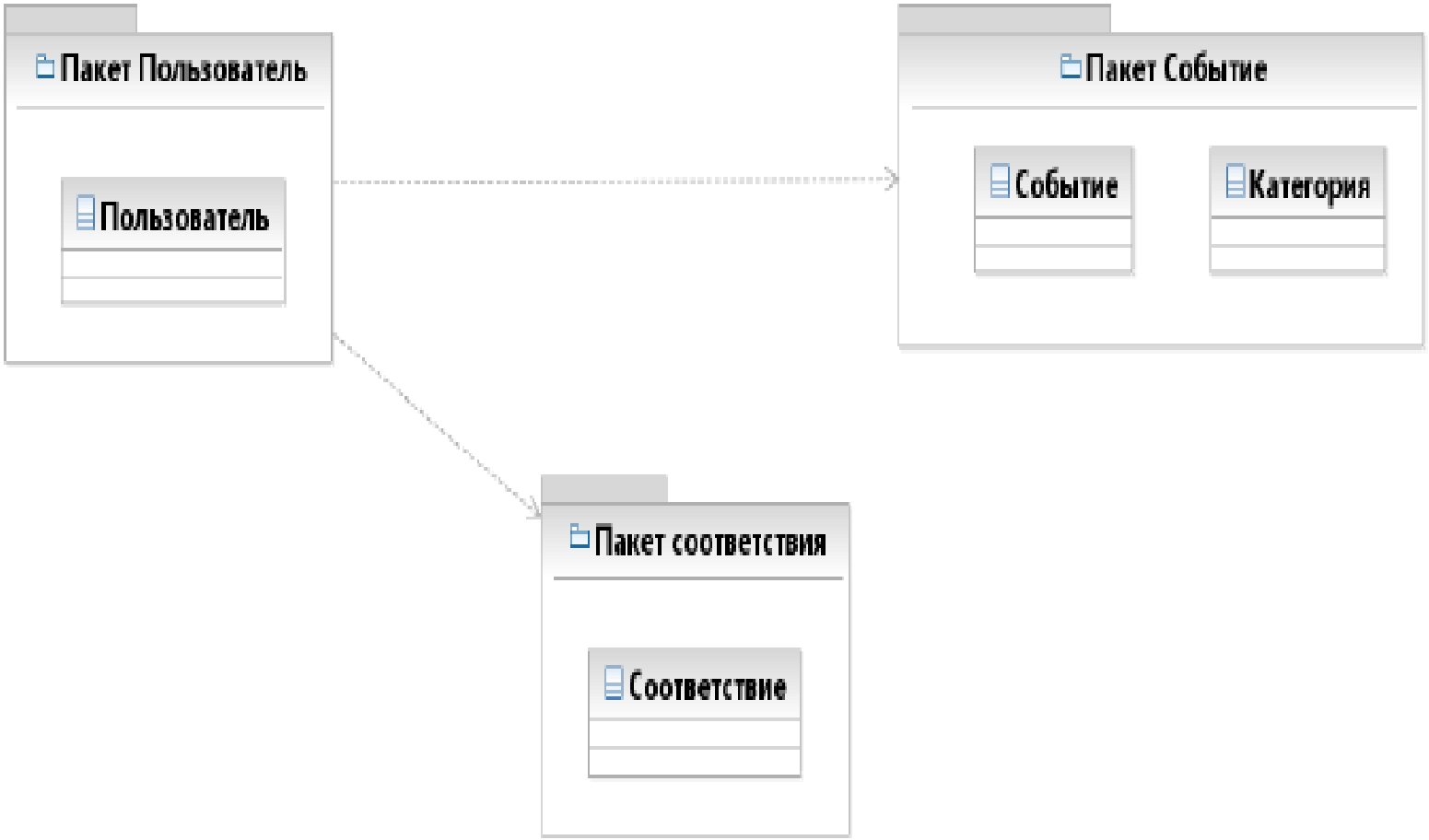
Прототип ПИ – пример ЖСС



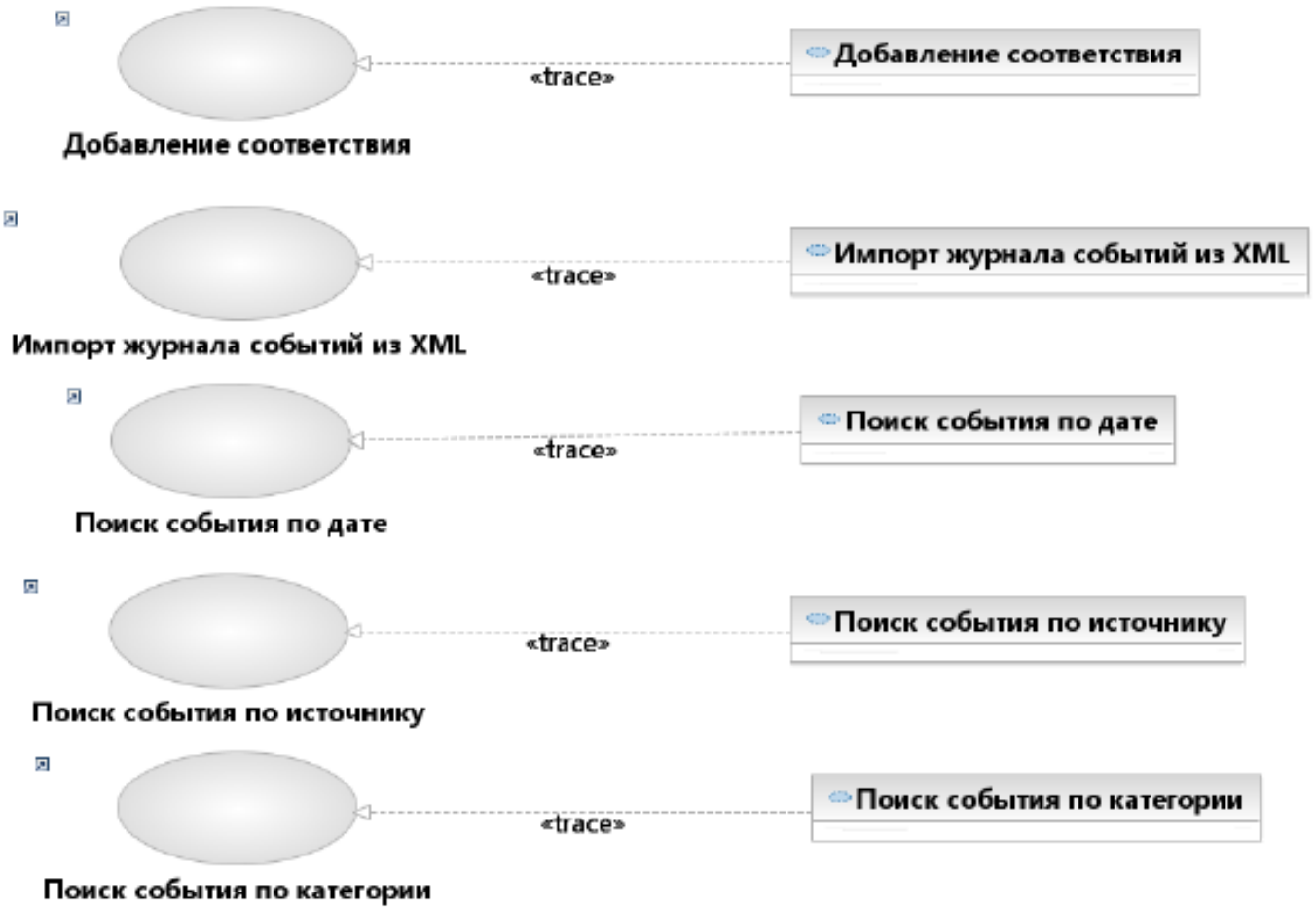
Рабочий процесс Анализ (требований)

- Анализ архитектуры
- Анализ кооперацій
- Анализ классов
- Анализ пакетов

Пакеты анализа – пример ЖСС

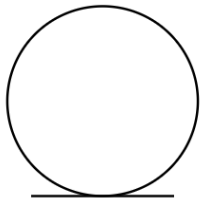


Трассировка кооперацій от прецедентов – пример ЖСС

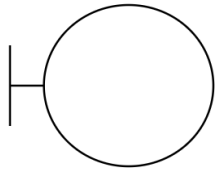


Классы анализа

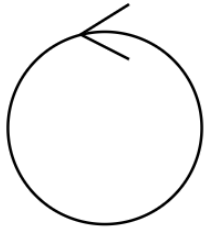
- Класс сущности - для моделирования долгоживущей системы, часто сохраняемой информацию о человеке / объекте / событии реального мира. Может иметь сложное поведение. Показывает логическую структуру данных.



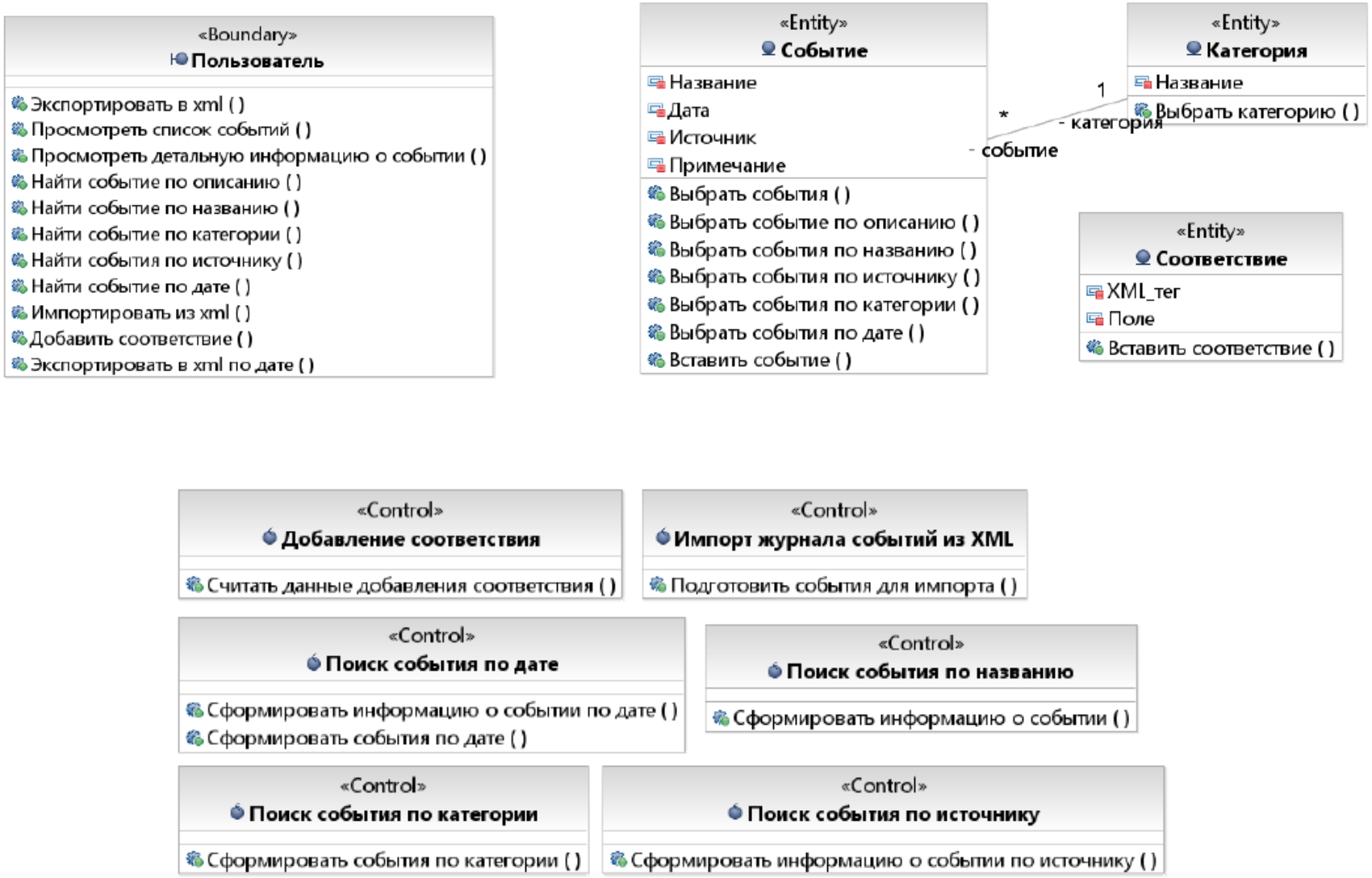
- Граничный класс - для моделирования взаимодействия между системами и актерами:получение / передача информации / запросов. Соответствуют пользовательскому интерфейсу / устройству/ коммуникации / API на внешнем уровне без реализации.



- Управляющий класс - координация / последовательность /взаимодействие / управление другими объектами для прецедента. Обрабатывают и координируют действия и потоки управления; реализуют бизнес-логику.

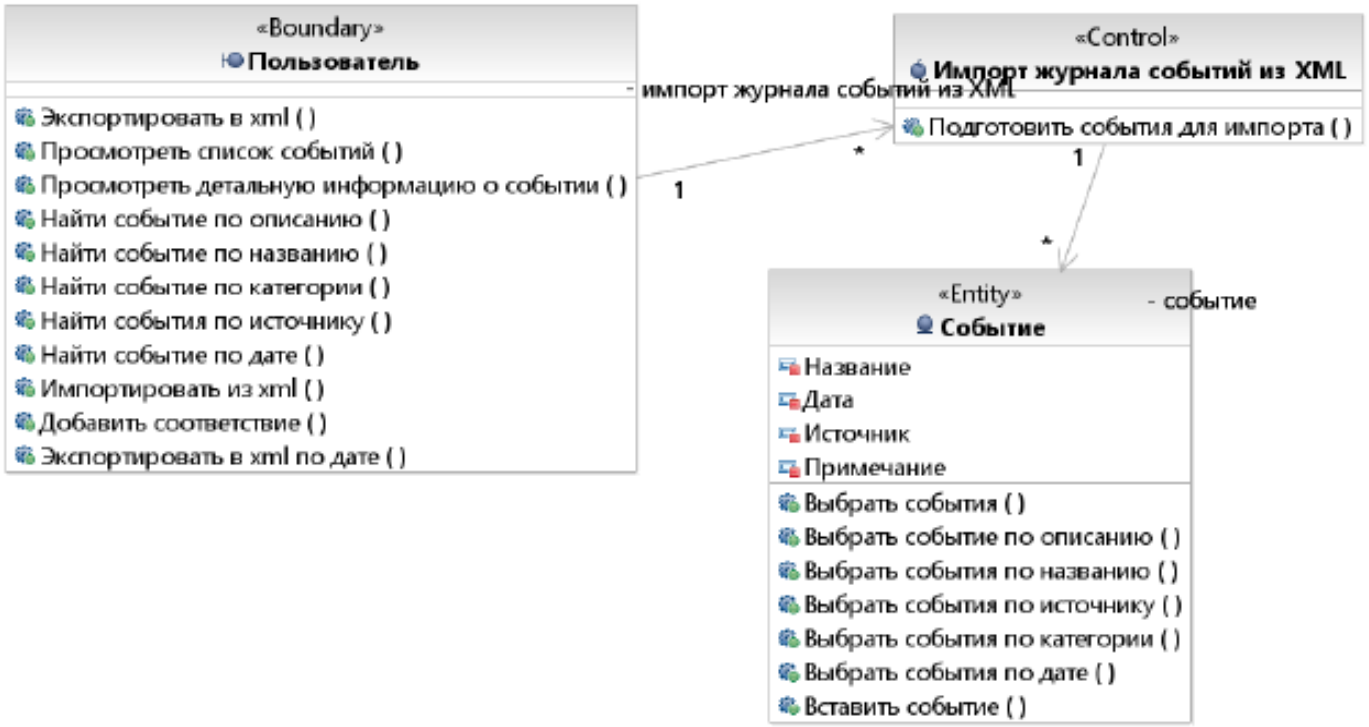


Классы анализа – пример ЖСС



Кооперация – пример ЖСС

Импорт журнала событий из XML



Анализ классов – пример ЖСС

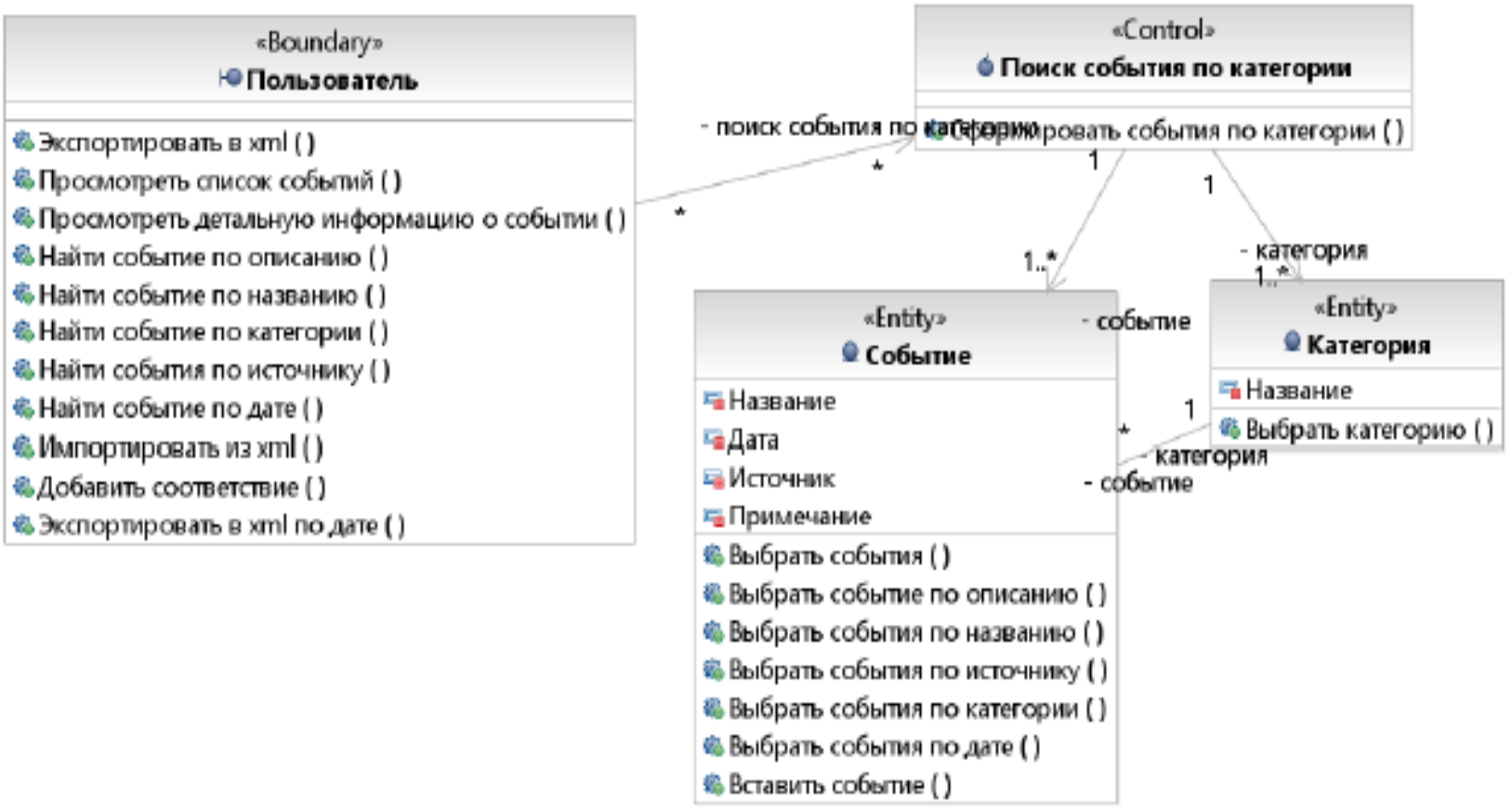
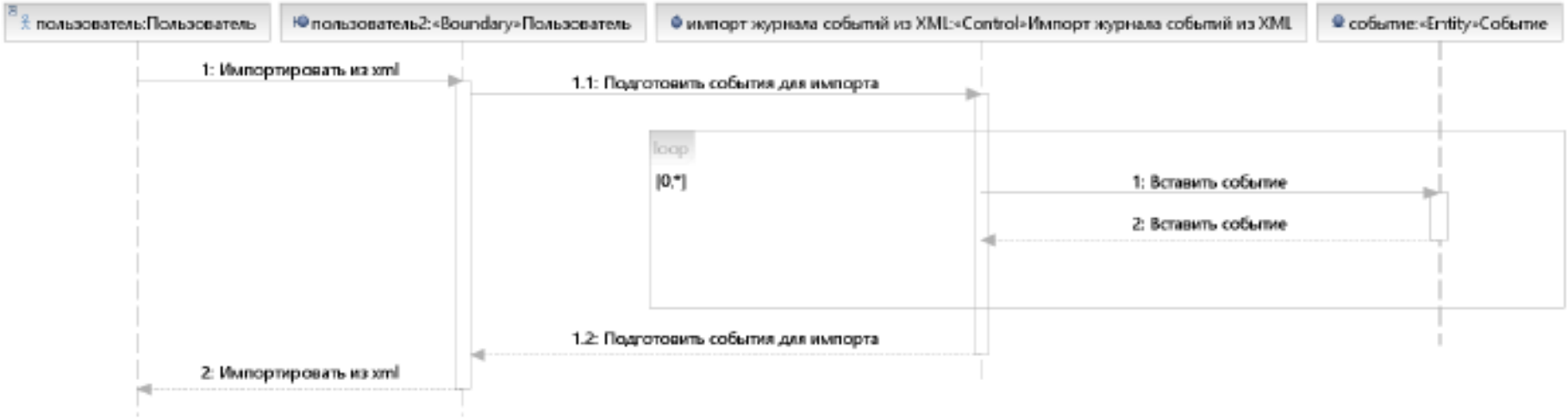
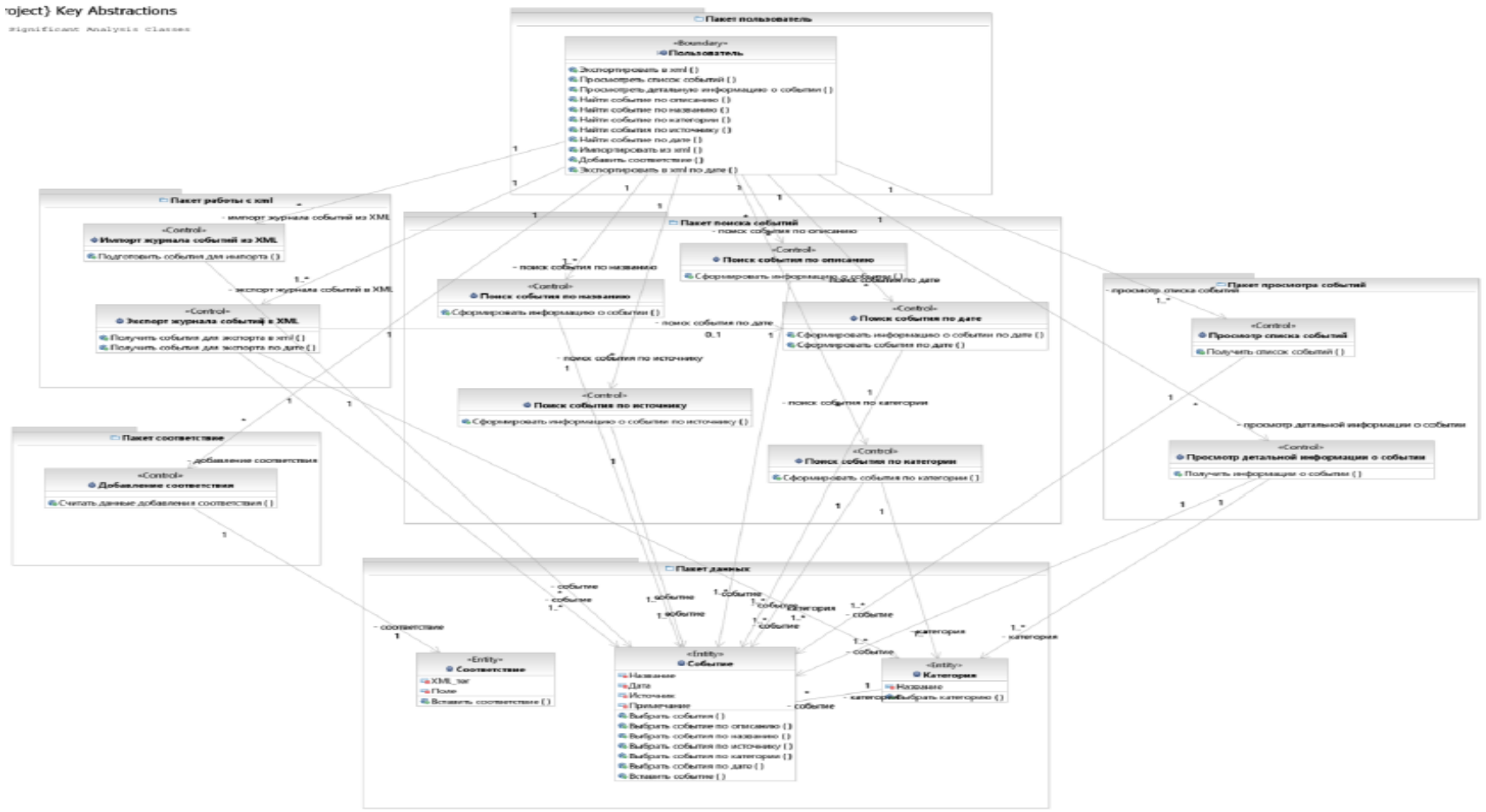


Диаграмма последовательностей кооперации – пример ЖСС



Анализ пакетов – пример ЖСС



Рабочий процесс – проектирование

- Проектирование архитектуры
- Проектирование прецедентов
- Проектирование классов
- Проектирование подсистем

Проектирование архитектуры

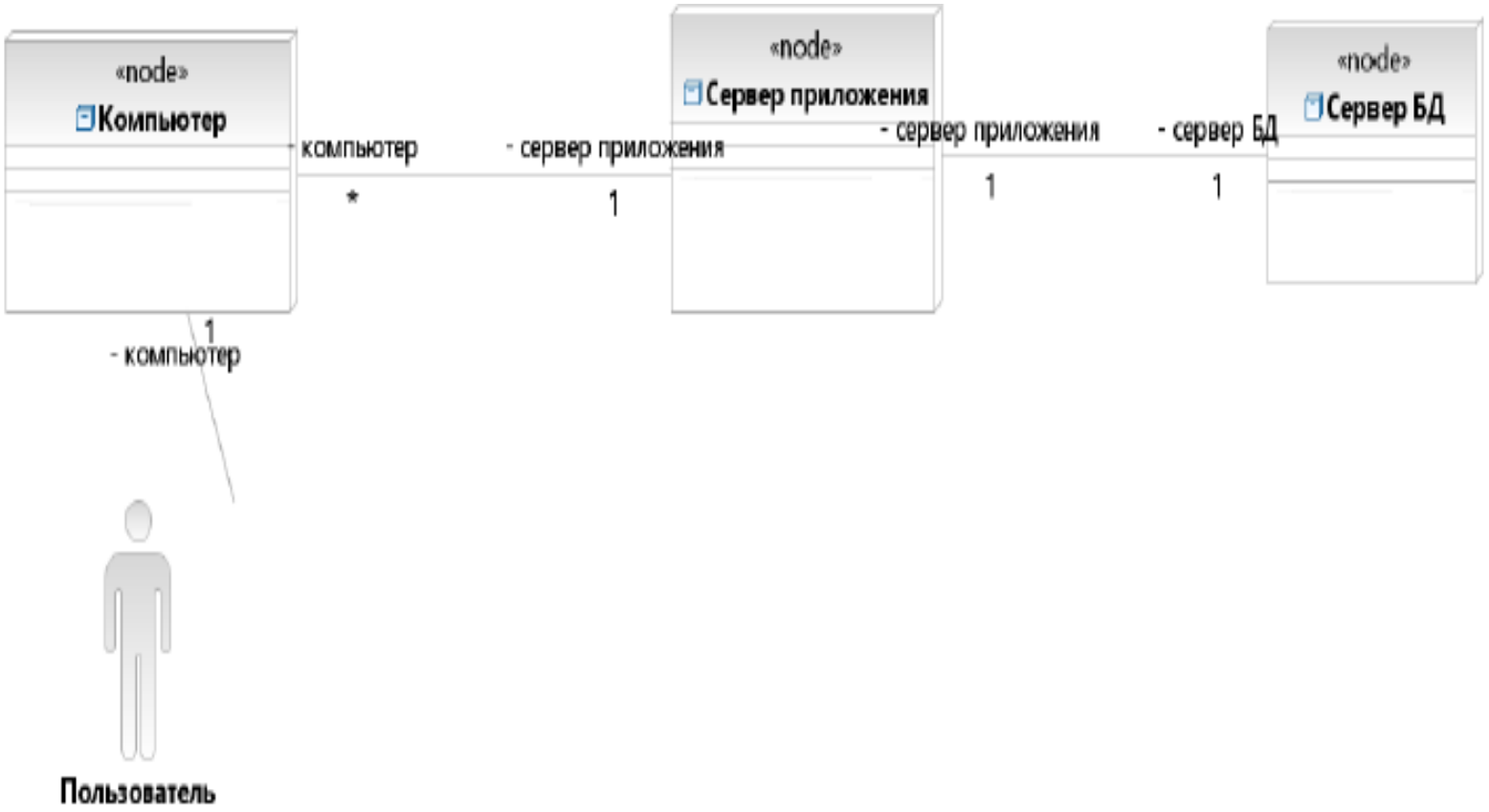
- Определение узлов и сетевых конфигураций
- Определение подсистем и их интерфейсов
  - Определение прикладных подсистем
  - Определение сервисных подсистем
- Определение интерфейсов подсистем
- Определение архитектурно значимых классов проектирования
- Определение обобщенных механизмов проектирование

Определение узлов и сетевых конфигураций

- перечень узлов и их конфигураций
  - конфигурация сети (линии связи между узлами; протоколы; характеристики передачи; скорость; качество;...)
- дополнительные требования
- конфигурация для тестирования / моделирования ПО

=> диаграмма развертывания

Диаграмма развертывания – пример ЖСС

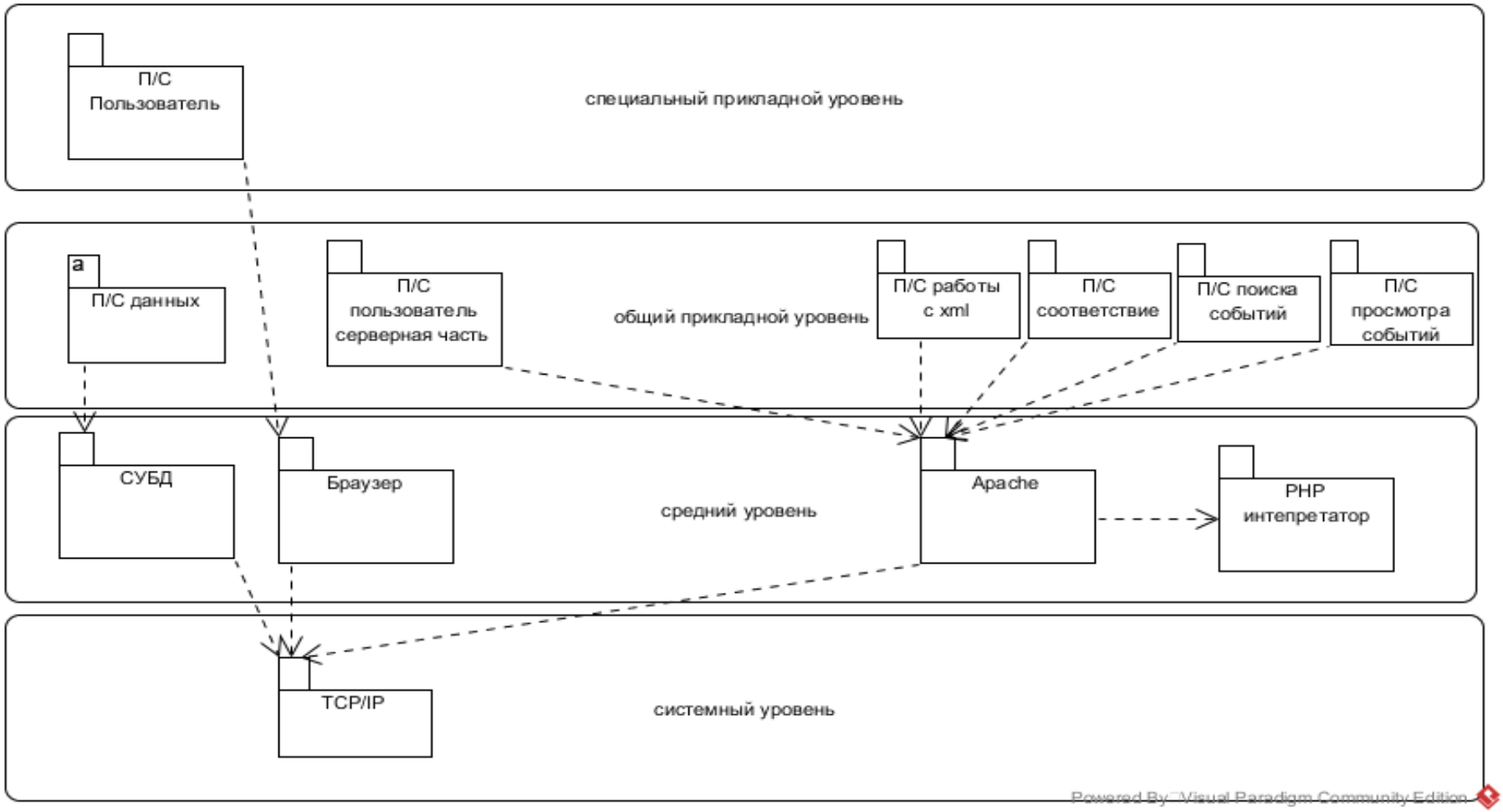


Определение подсистем и их интерфейсов

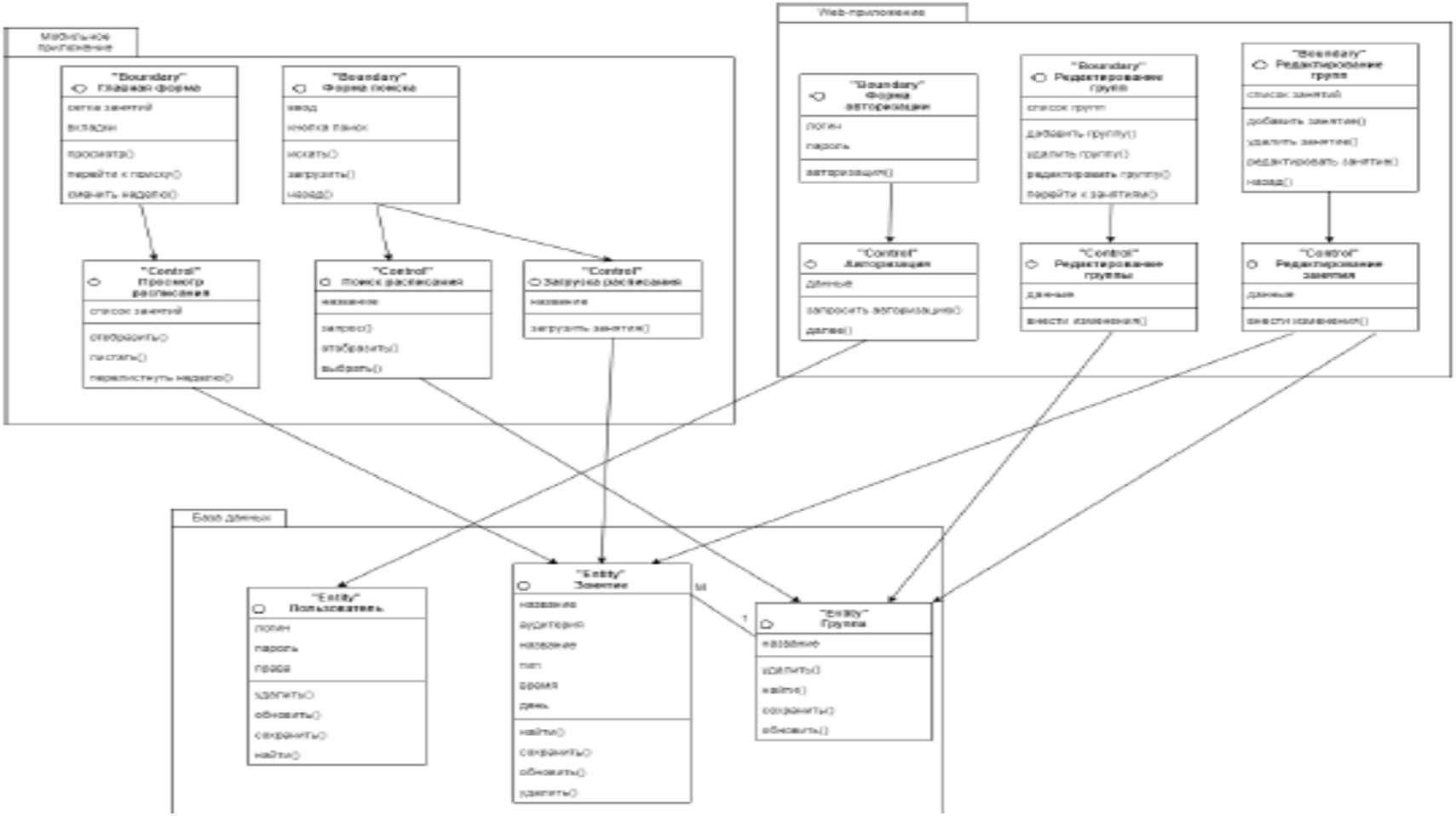
- Определение прикладных подсистем
  - пакет анализа трассируется в подсистему проектирования
  - декомпозиции при совместном использованиеи
  - разнесение по узлам
- Определение подсистем среднего уровня и уровня системного ПО
  - ОС, СУБД, коммуникация, распределение, транзакции
  - Интеграция покупаемых / создаваемых компонентов
  - Оценка эффективности и пригодности
- Определение зависимостей между подсистемами

- => диаграмма уровней подсистем

Диаграмма уровней подсистем – пример ЖСС



Пакеты анализа - пример



Трассировка - пример

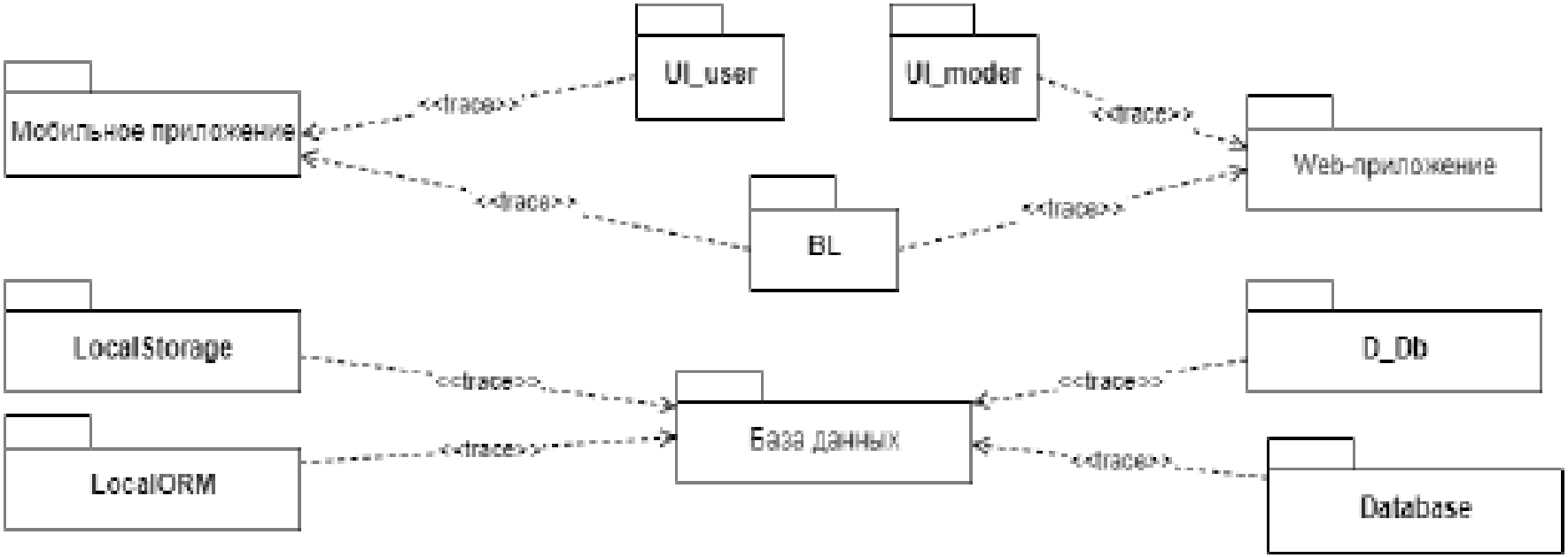


Диаграмма уровней подсистем – пример, фрагмент

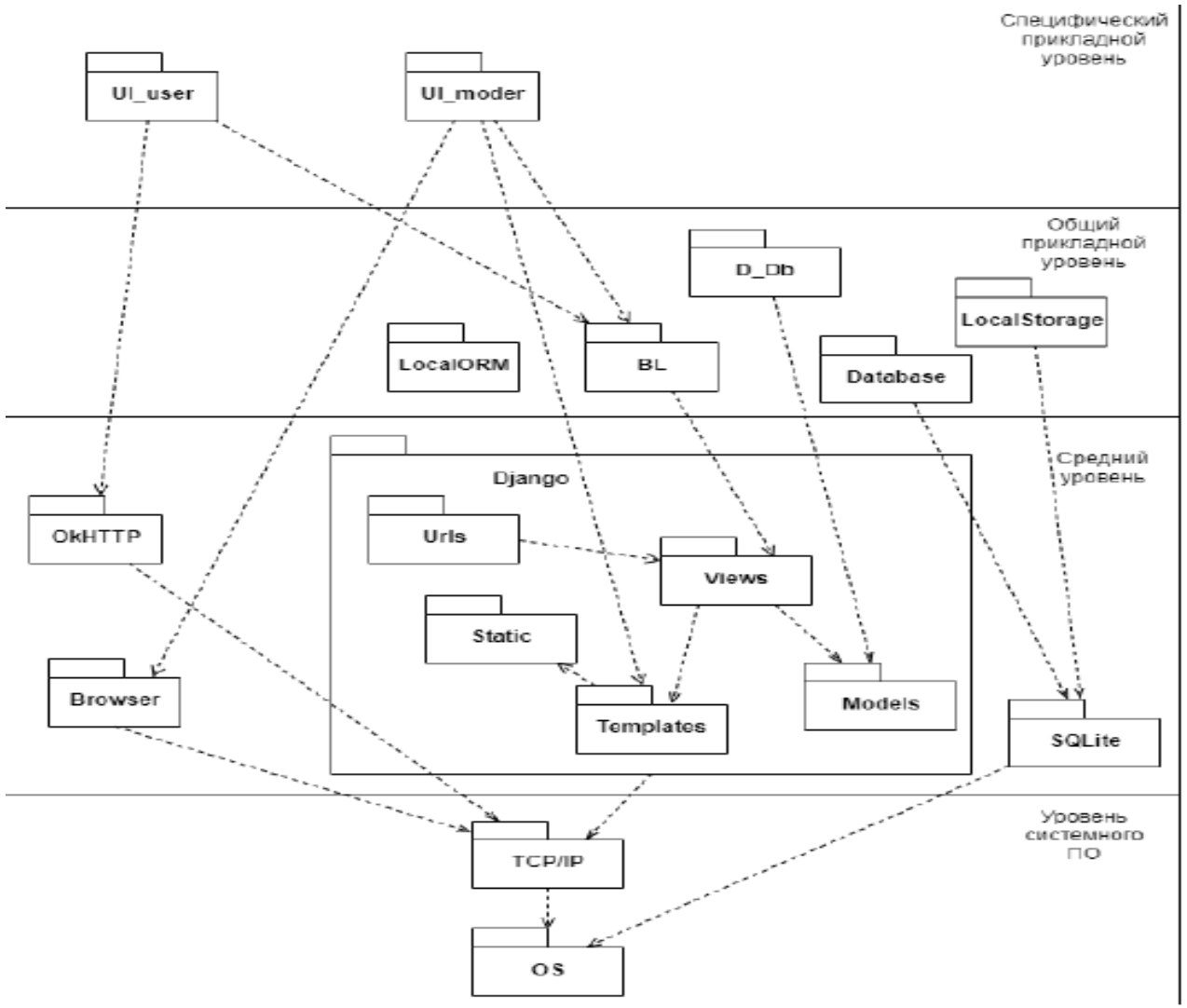
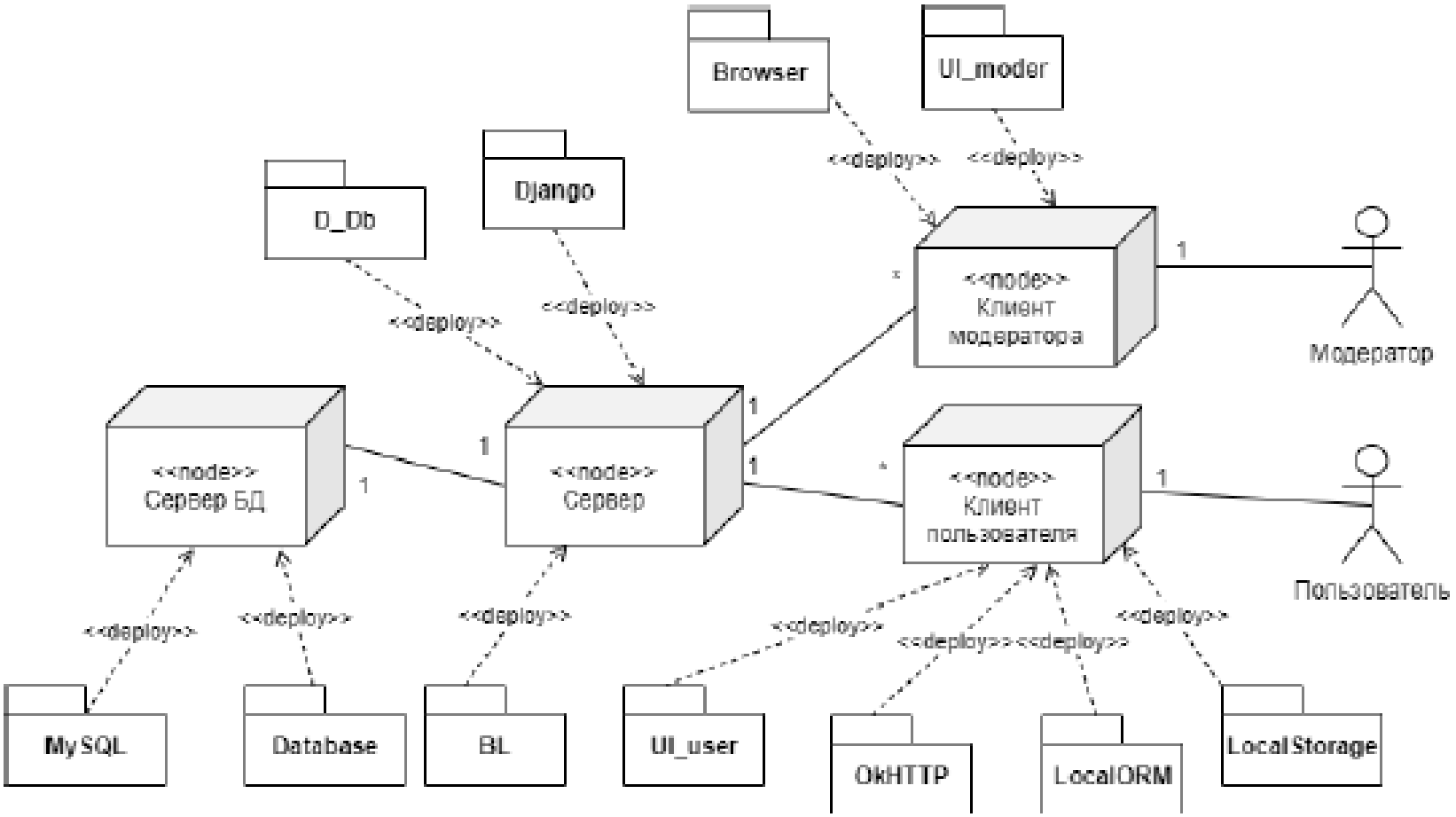


Диаграмма распределения подсистем по узлам - пример



Определение интерфейсов подсистем

- перечень операций, доступных извне;
- на основе классов в пакетах анализа (к которым обращаются извне);
- на основе зависимостей подсистем;
- начать с подсистем верхнего уровня;

- Для среднего и нижнего уровня могут быть существующие заранее (существующее ПО)

Определение архитектурно-значимых классов проектирования

- Определение классов проектирования на основе сущностных классов анализа.
- Определение активных классов (своя нить управления, процесс)
  - один или более на узел;
  - один на обмен между узлами;
  - для повышения производительности (ответ без задержек);
  - для запуска / остановки / конфигурации / снятие блокировки / восстановление и т.д.

- Дальнейшая трассировка:
  - активный класс -> исполняемый файл;
  - подсистема -> компонент

Определение обобщенных механизмов проектирования

- Реализуют требования:
  - длительное хранение;
  - распределение объектов;
  - средства безопасности;
  - контроль и исправление ошибок;
  - управление транзакциями.
- Выбор существующих технологий проектирования и реализации
- Обобщенные механизмы проектирования:
  - для хранения -> реляционная база данных, файл, объектно-ориентированная база данных,...
  - распределенная обоботка -> java.rmi, corba, com,...
- Использование обобщённых кооперацій - образцов, паттернов
- Обычно:
  - Поставляемое ПО -> средний уровень и уровень СПО
  - Создаемое ПО -> прикладной уровень

Проектирование прецедентов (кооперацій)

- Определение участвующих в кооперации классов проектирования
- Описание взаимодействия объектов проектирования
- Определение участвующих подсистем и их интерфейсов
- Описание взаимодействия подсистем
- Определение специальных требований к кооперациям

Трассировка классов анализа в классы проектирования

- **Граничные** – в классы форм и их компонентов (производные от стандартных)
- **Сущности** – в таблицы БД, файлы, объекты в ОП
- **Управляющие** – в классы реализации бизнес-логики

- если класса проектирования нет, то его надо определить

Трассировка граничного класса - пример

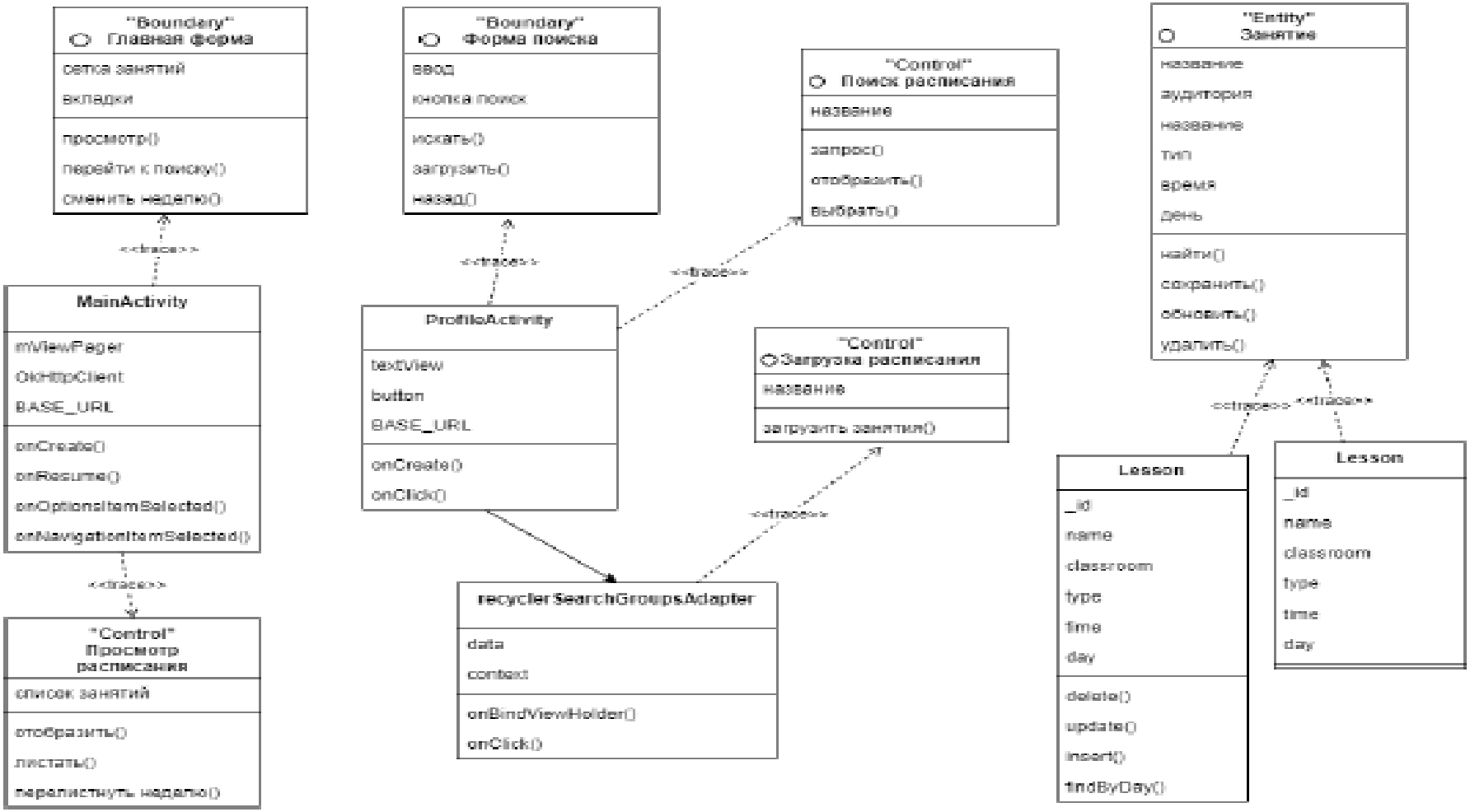


Диаграмма последовательности объектов - пример

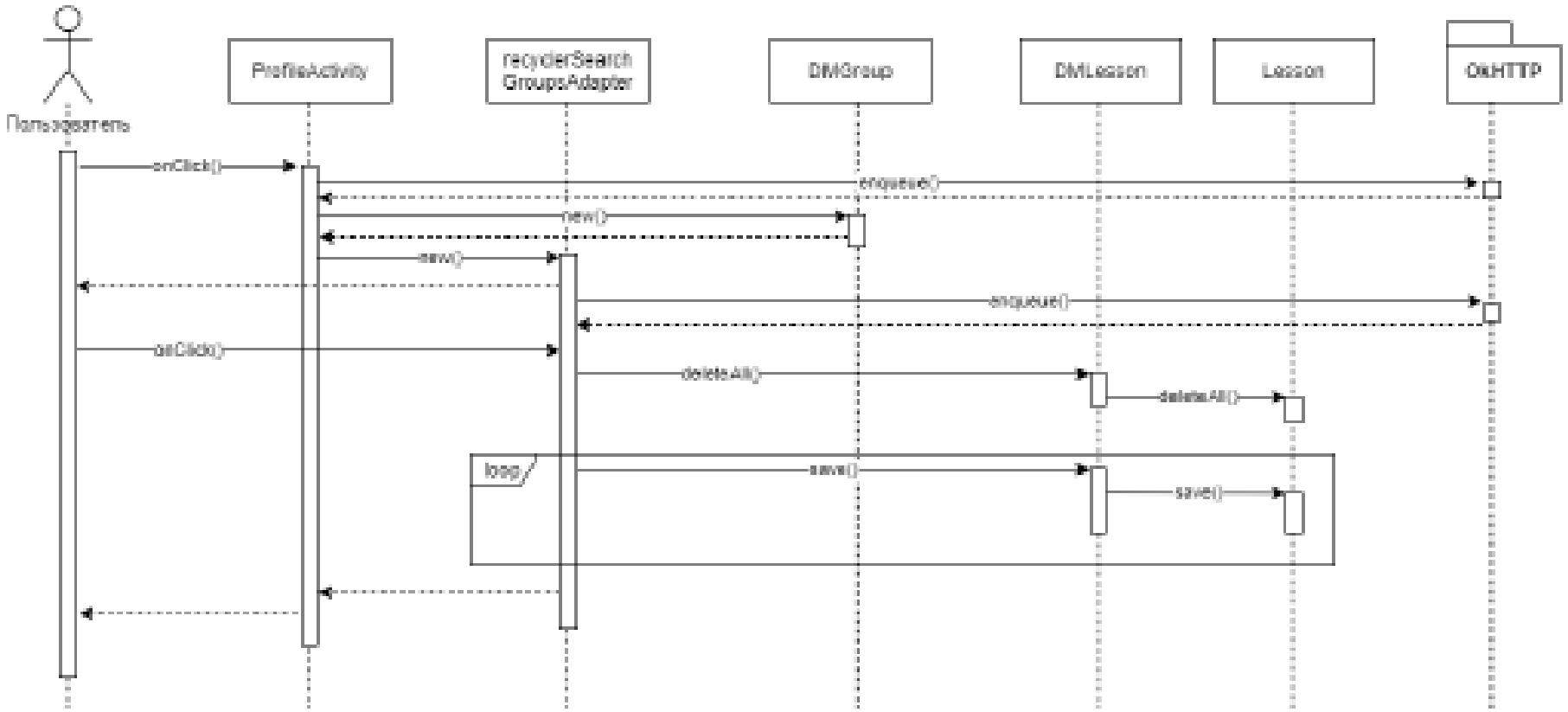
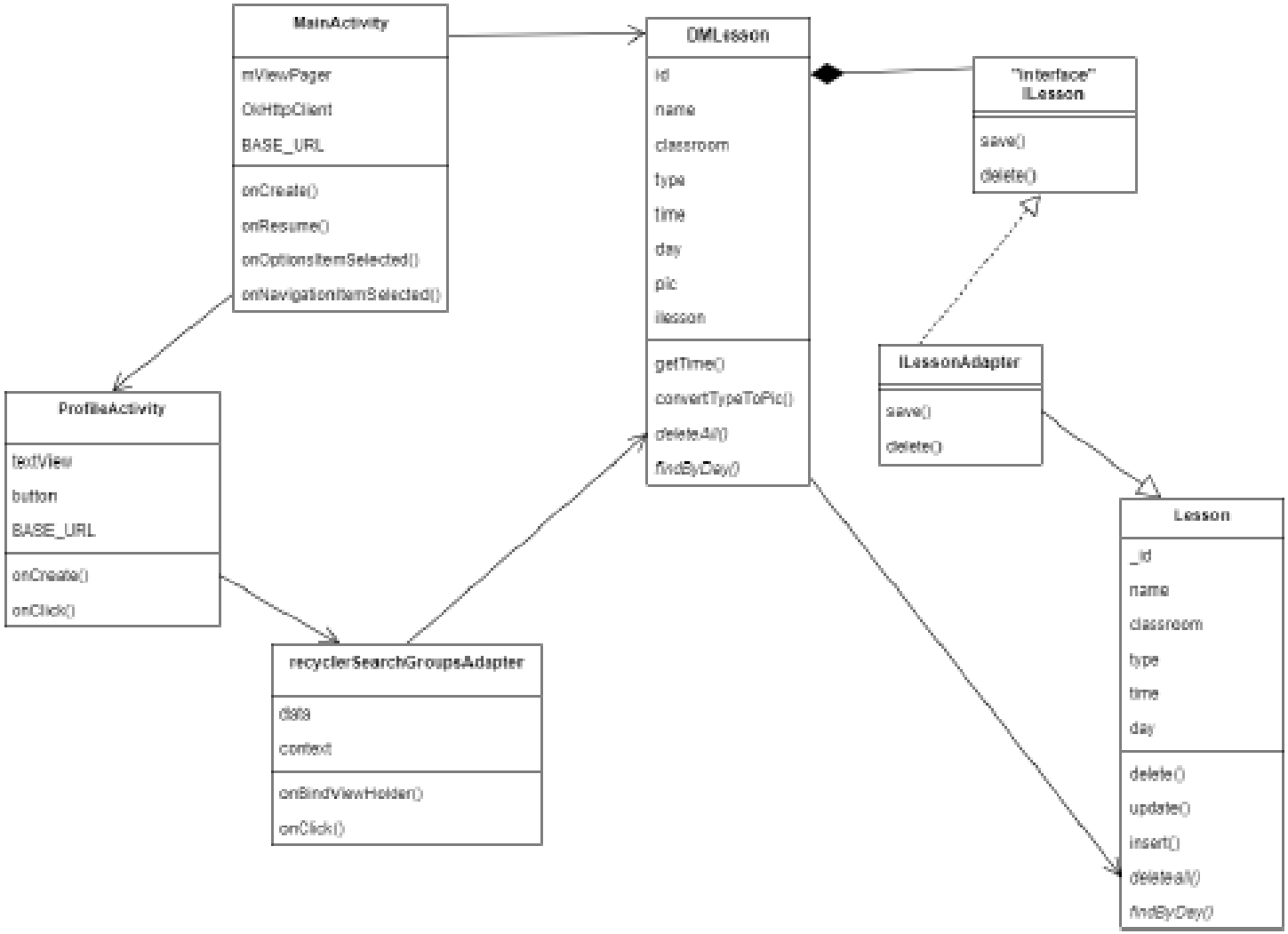


Диаграмма классов проектирования - пример



Распределение классов по подсистемам

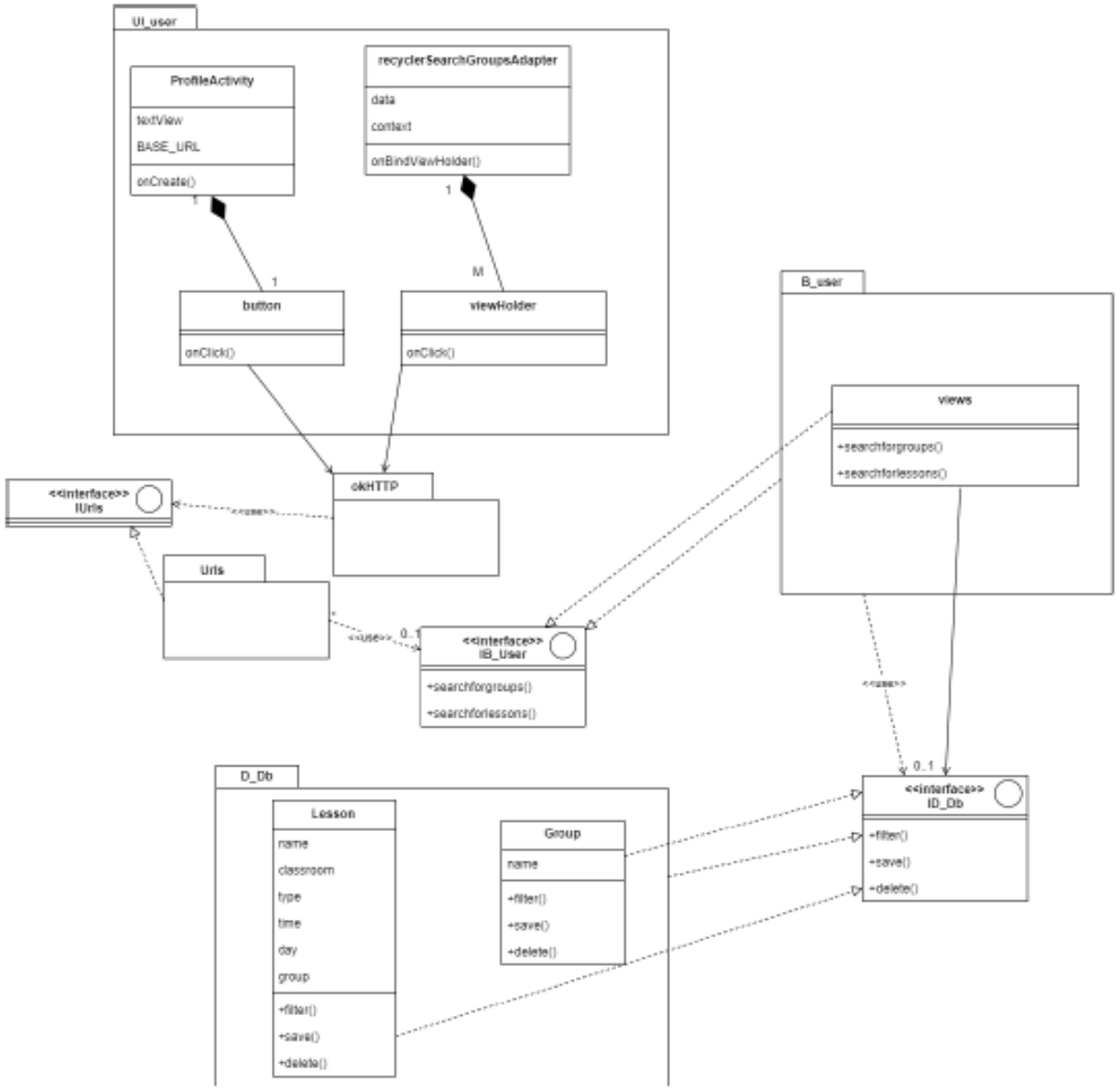
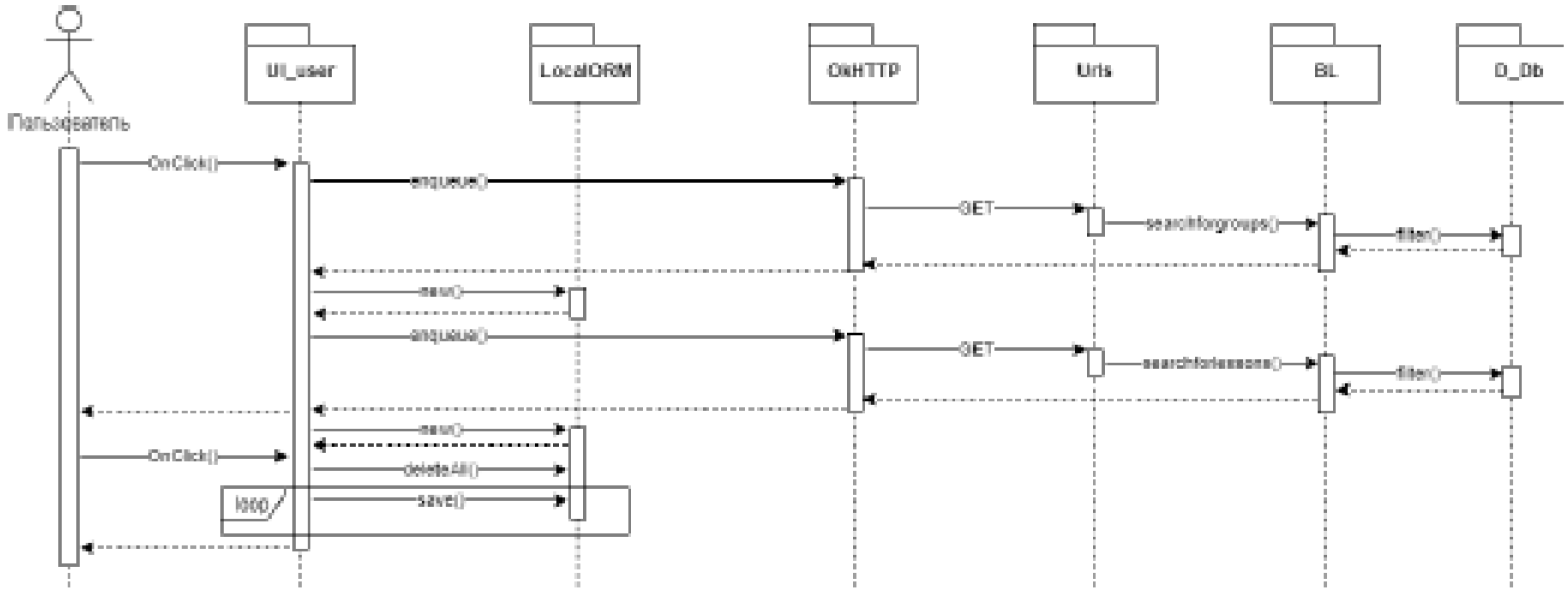


Диаграмма последовательности подсистем - пример



Проектирование классов

- - операции (сигратуры в терминах языка);
  - атрибуты (стандартные типы);
  - отношения (ассоциации, агрегации, обобщение);
  - методы реализации (алгоритмы)
  - состояния
- - зависимость от обобщенных механизмов проектирования
  - требования к реализации
  - правильность реализации всех интерфейсов

Проектирование подсистем

- сильная связность и слабое сцепление (перемещение классов)
- выполнение задач
- правильный интерфейс.

- Сохранение зависимостей между подсистемами
  - Связь классов подсистемы => связь подсистем
    Желательнее связь между интерфейсами, чем подсистемами
- Сохранение предоставляемых подсистемой интерфейсов
  - Операции интерфейса подсистем должны соответствовать всем ролям подсистем.
- Сохранение содержимого подсистемы
  - Для любого интерфейса подсистемы должен быть класс его реализации;
  - Для любого интерфейса / любого прецедента должна быть кооперация в терминах классов подсистемы