Унифицированный процесс разработки ПО: Реализация и тестирование

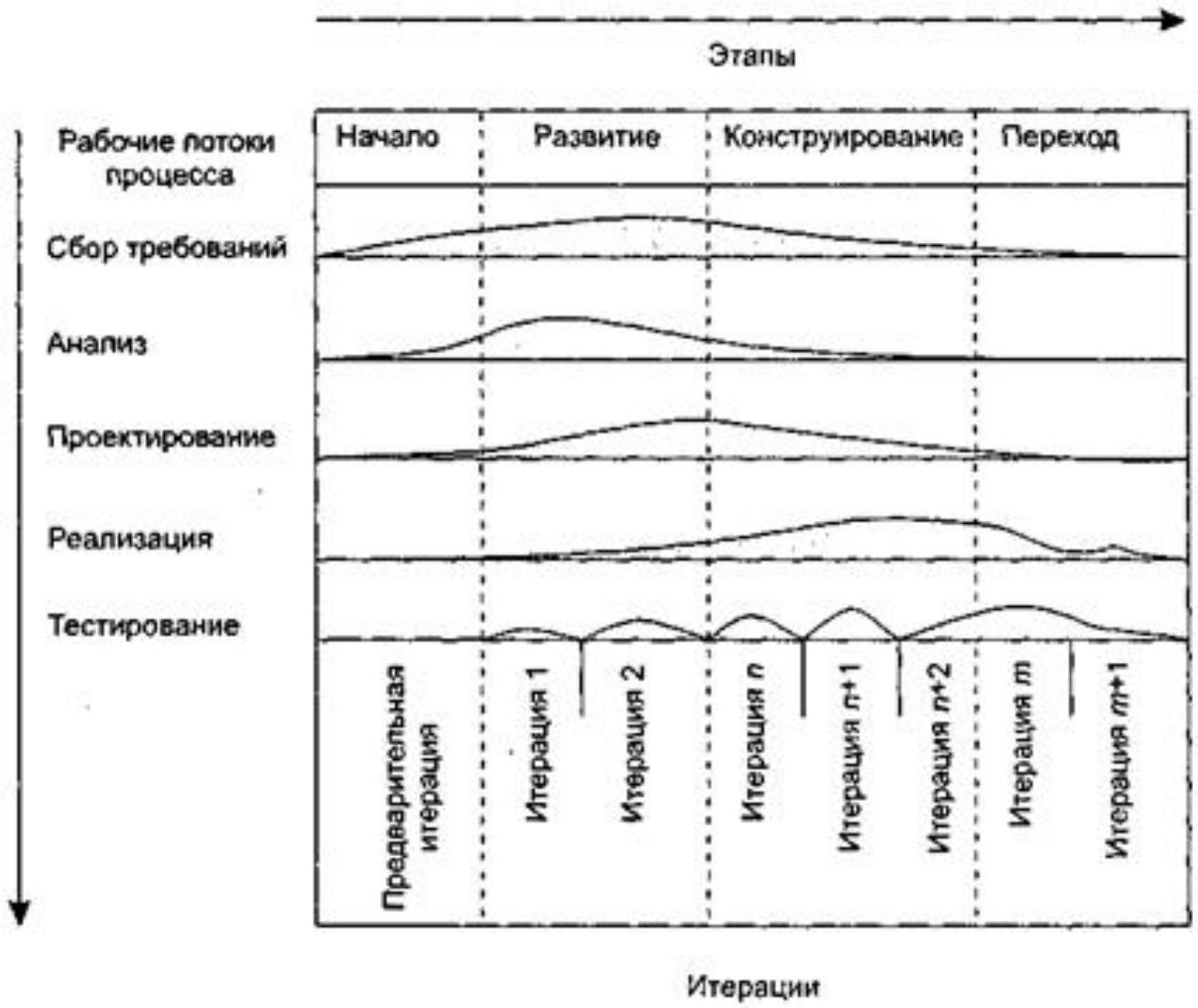
Технологии разработки программного обеспечения

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 продукта и оценка выполнимости проекта;
- **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.

Определение требований - артефакты

- Контекст системы:
  - Глоссарий понятий
  - Диаграма классов предметной области
  - Бизнес-модель
- Функциональные требования:
  - Диаграммы прецедентов
  - Спецификации прецедентов
  - Диаграммы активности (или иные)
  - Прототип пользовательского интерфейса
- Нефункциональные требования

Анализ требований - артефакты

- Анализ архитектуры:
  - Трассировка прецедентов в кооперации
  - Пакеты анализа и сервисные пакеты
  - Классы анализа: граничные, управляющие и сущности
- Анализ кооперации:
  - Диаграммы классов – участников кооперации
  - Диаграммы последовательностей для кооперации
- Анализ пакетов:
  - Распределение классов по пакетам
  - Зависимости пакетов анализа

Проектирование - артефакты

- Проектирование архитектуры:
  - Диаграмма развертывания
  - Трассировка пакетов в подсистемы
  - Диаграмма уровней подсистем
  - Общесистемные механизмы проектирования
- Проектирование кооперaций:
  - Трассировка классов анализа в классы проектирования
  - Диаграммы классов – участников кооперaций
  - Диаграммы последовательностей в терминах классов и в терминах подсистем для кооперaций
- Проектирование подсистем:
  - Распределение классов по подсисистемам
  - Зависимости и интерфейсы подсистем

Диаграмма развертывания

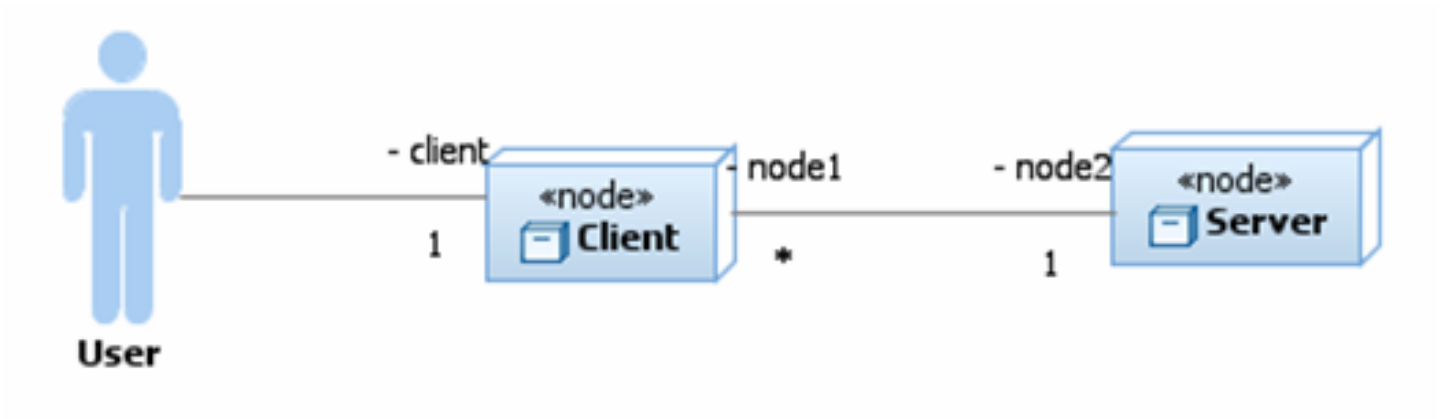
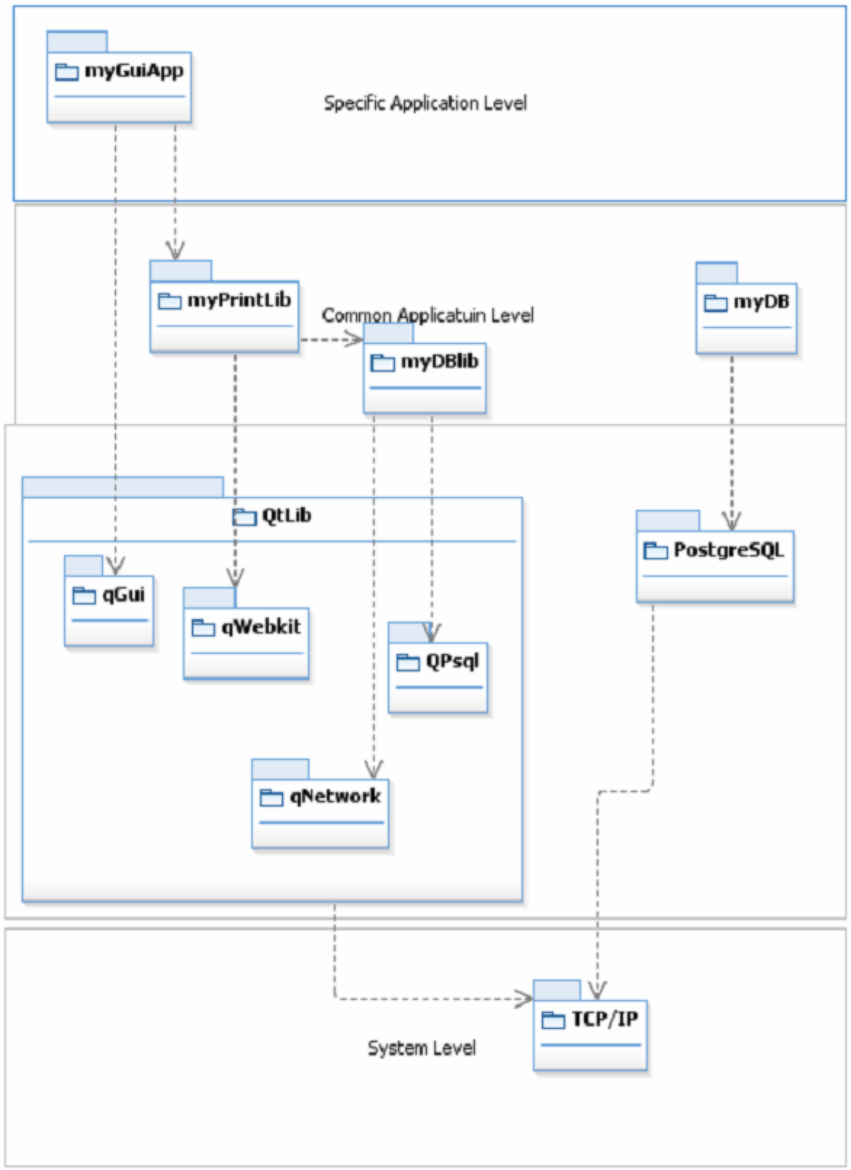
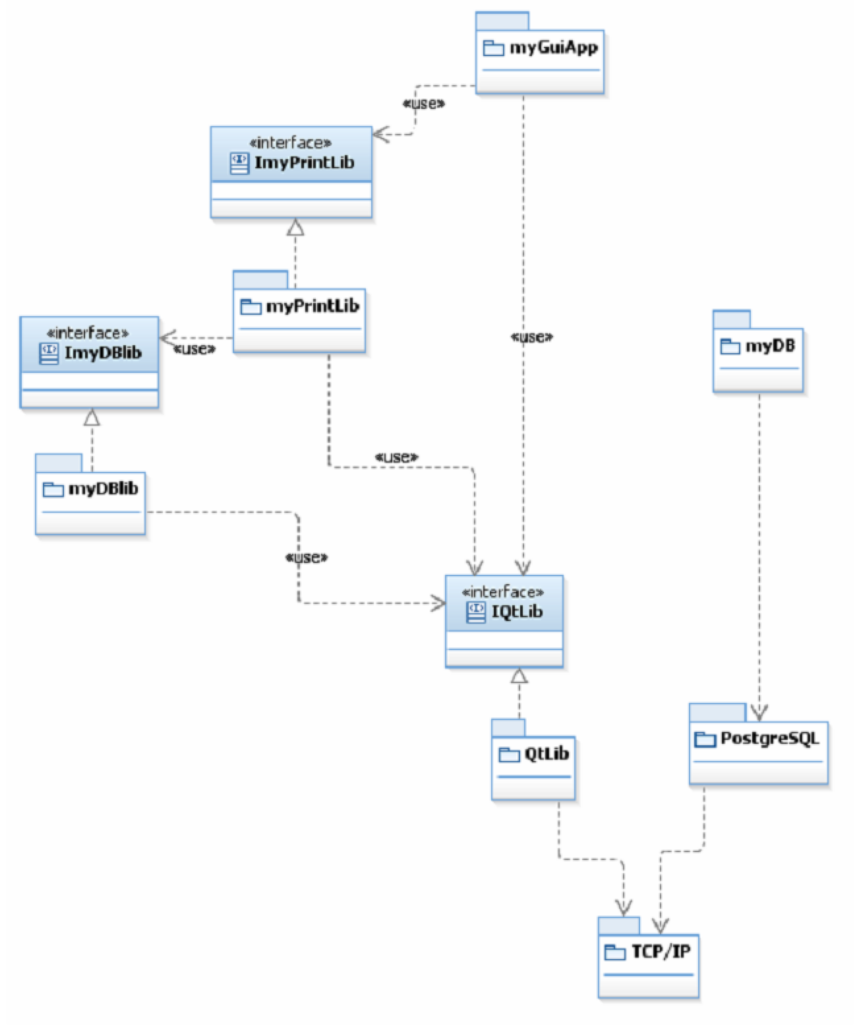
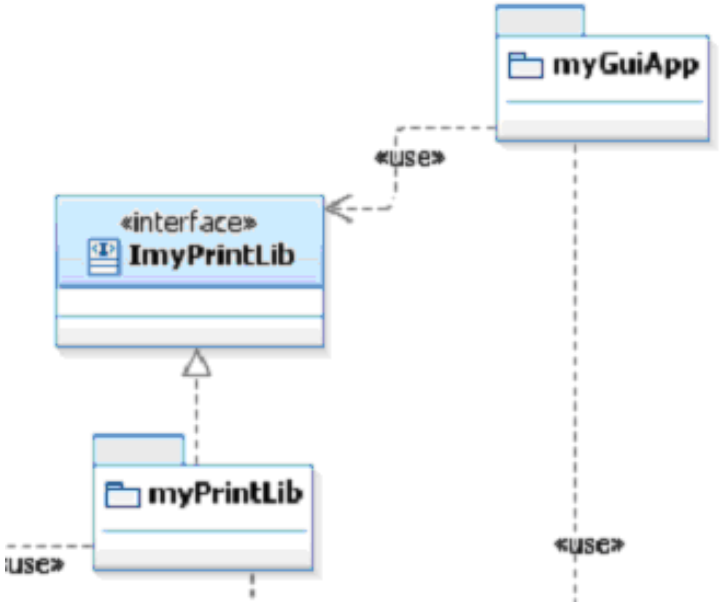


Диаграмма уровней подсистем

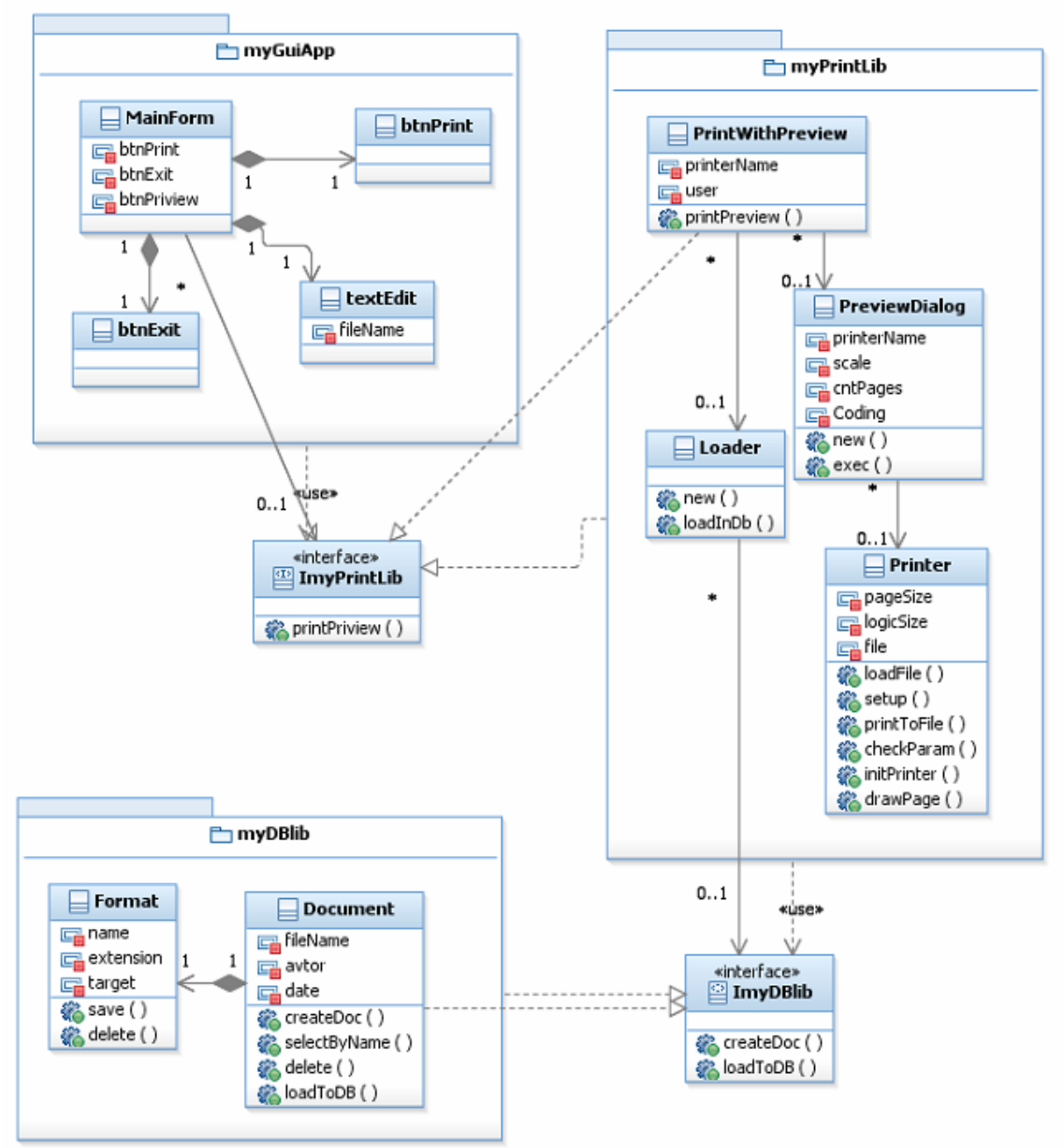


Зависимости и интерфейсы подсистем

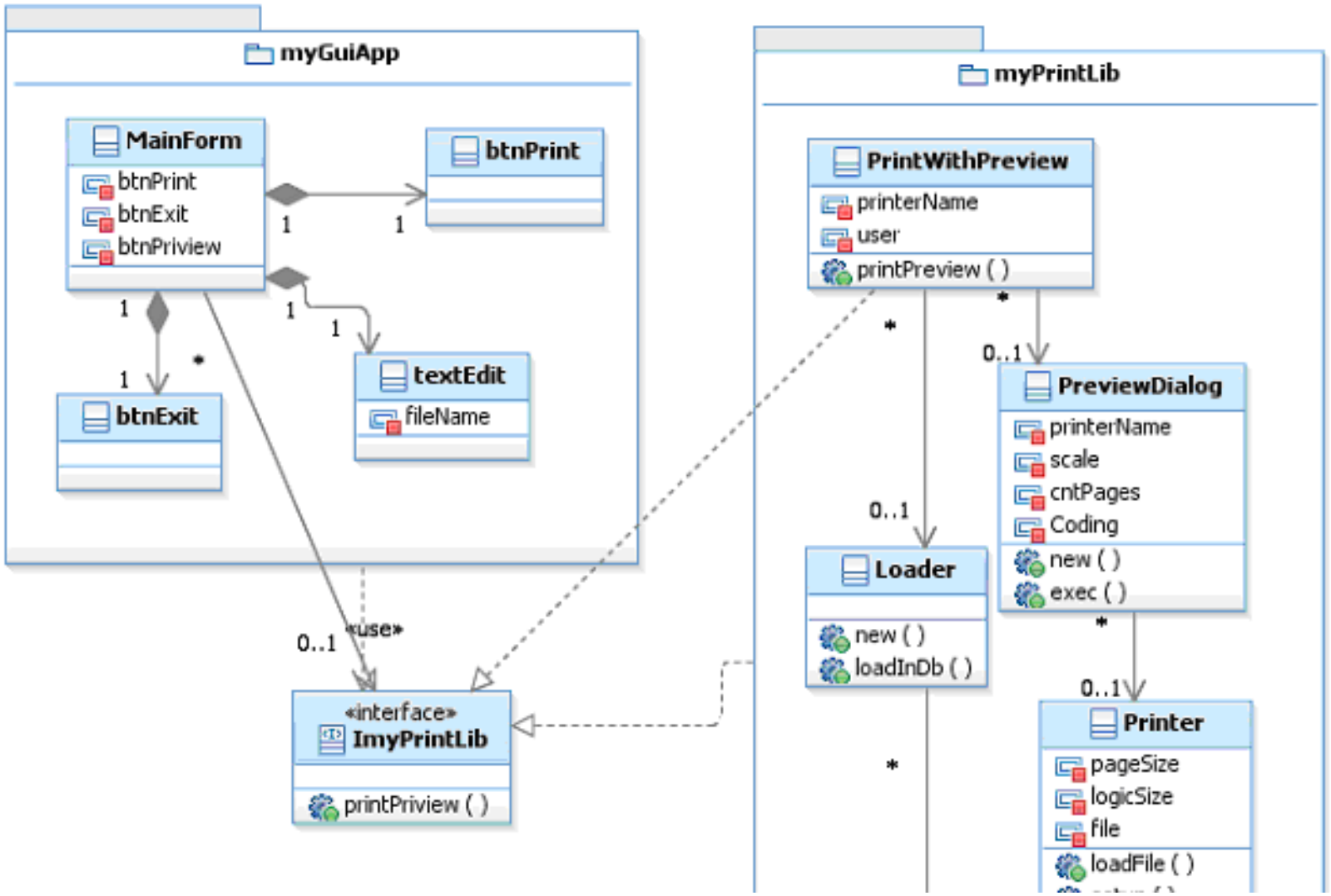




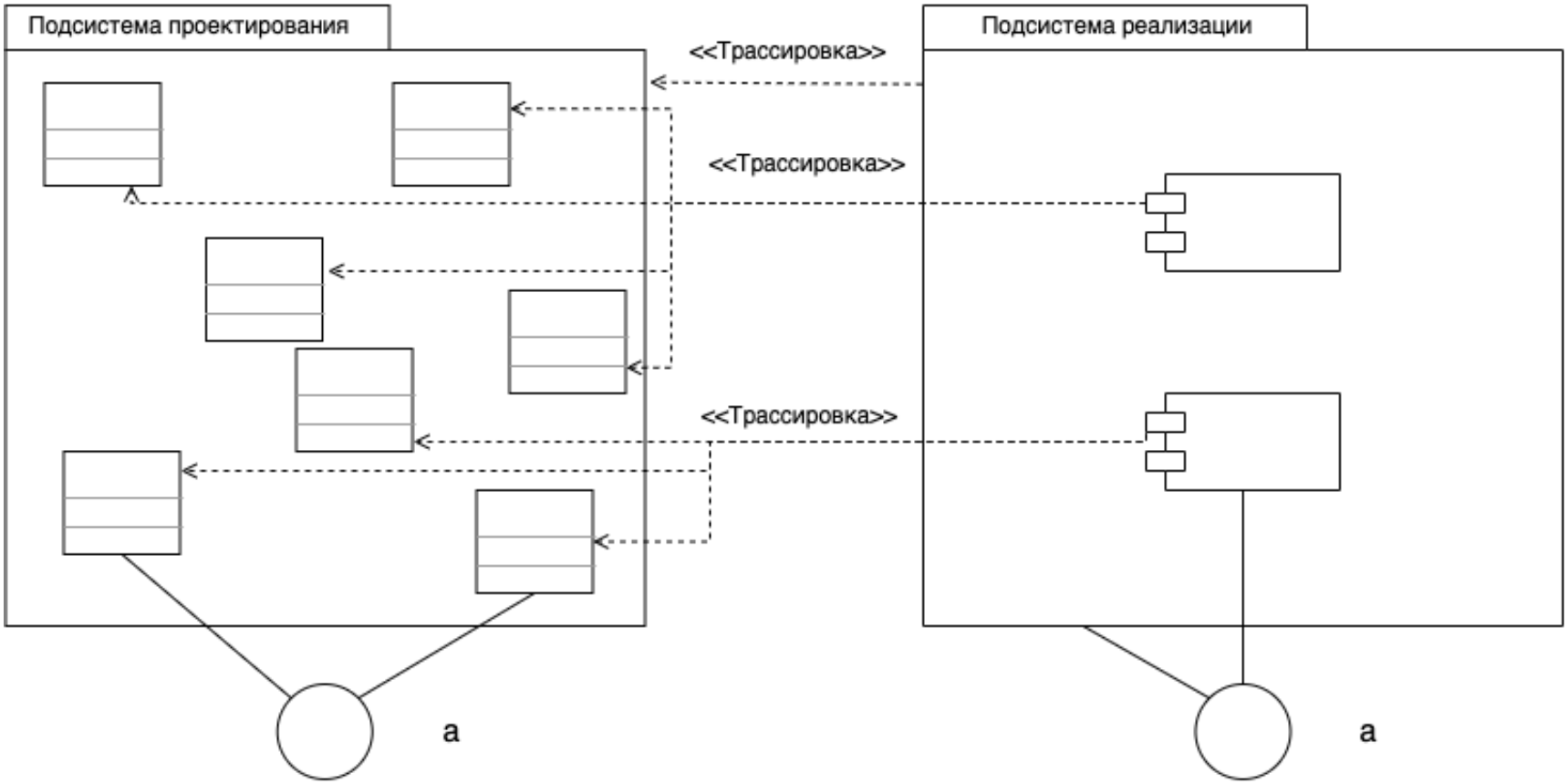
Распределение классов по подсистемам



Сохранение интерфейсов



Переход от проектирования к реализации



Трассировка (от проектирования к реализации)

- **Подсистема проектирования** трассируется в подсистему реализации.
- **Сервисная подсистема** - в сервисную подсистему.
- **Классы проектирования** - в файл(ы) компонентов (исходный текст программы; 1-* зависят от языка программирования).
- **Активный класс** - в исполняемый компонент.
- **Проект кооперации** - в сборку (билд).
- **Модель развертывания и конфигурации сети** - в распределенную систему развертывания исполняемых компонентов.

Рабочий процесс - Реализация

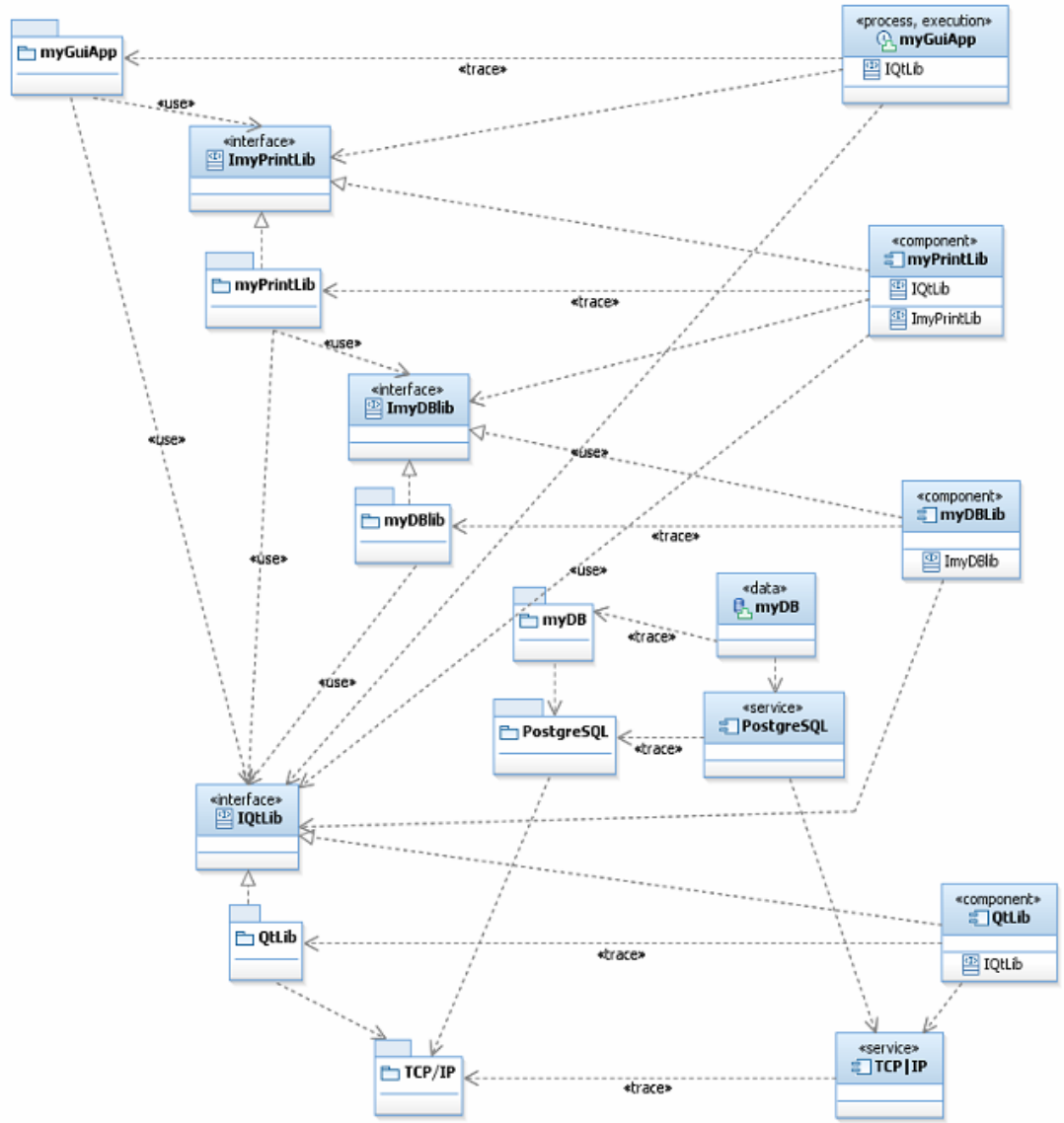
- Реализация архитектуры
- Сборка системы
- Реализация подсистем
- Реализация классов
- Тестирование модулей

Реализация архитектуры

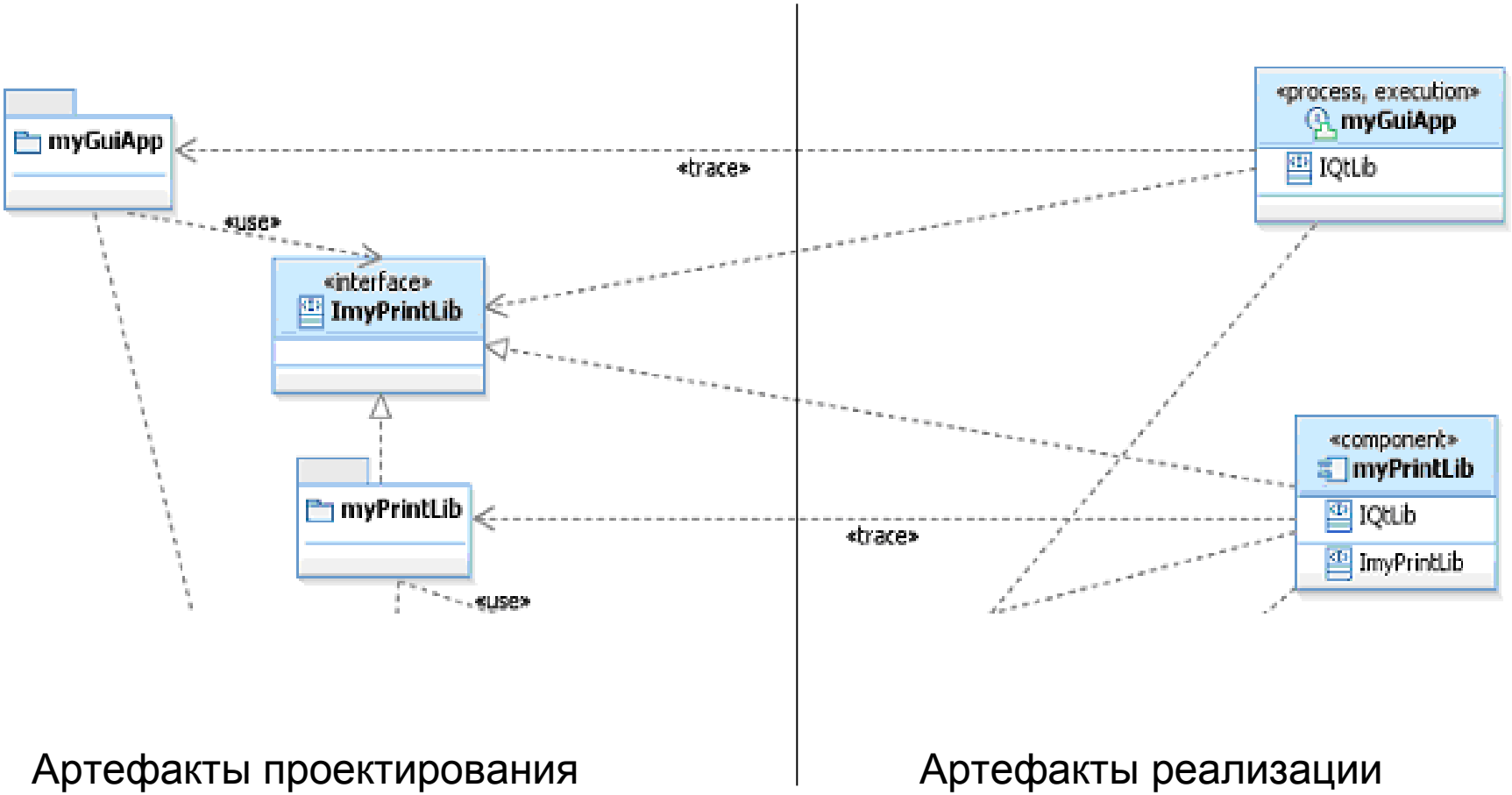
- Определение архитектурно значимых компонентов:
  - трассируются от архитектурно значимых элементов проектирования.
- Определение исполняемых компонентов и распределение их по узлам.

- Компонент - это физический контейнер для элементов (в том числе классов), зависит от языка программирования.
- Стереотипы компонентов:
  - <<исполняемый файл>> - программа;
  - <<файл>> - исходные данные / данные;
  - <<библиотека>> - статическая / динамическая;
  - <<таблица>>;
  - <<документ>>.

Трассировка подсистем в компоненты



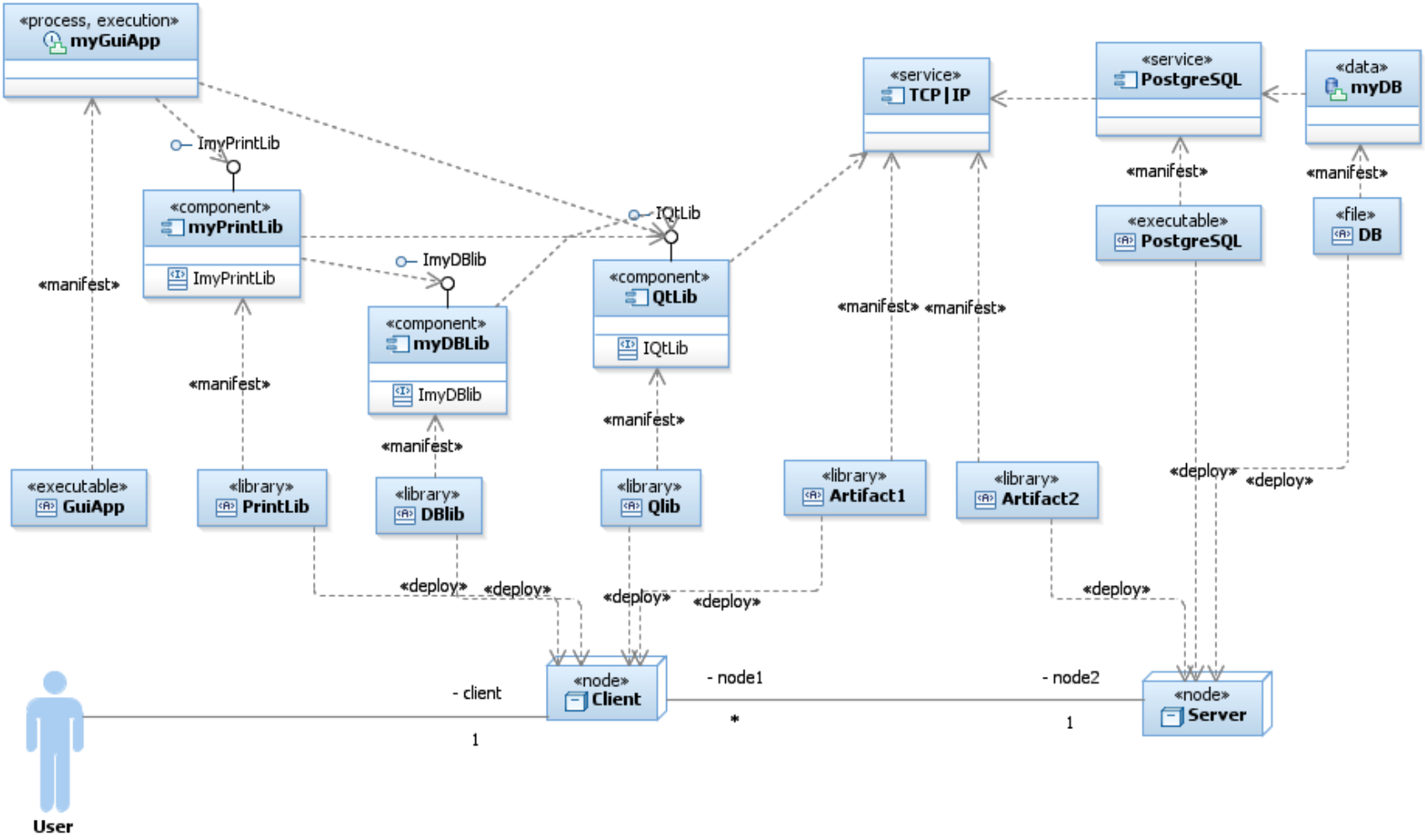
Сохранение интерфейсов при трассировке



Артефакты проектирования

Артефакты реализации

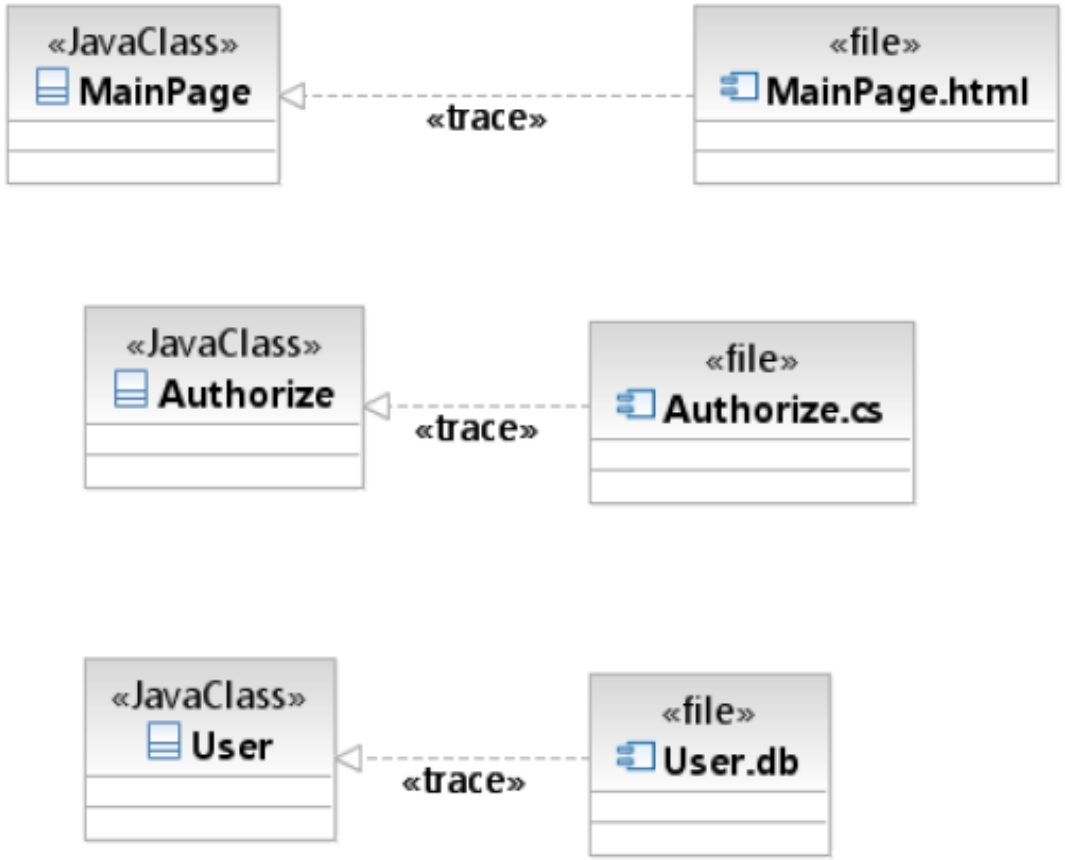
Манифестирование компонентов



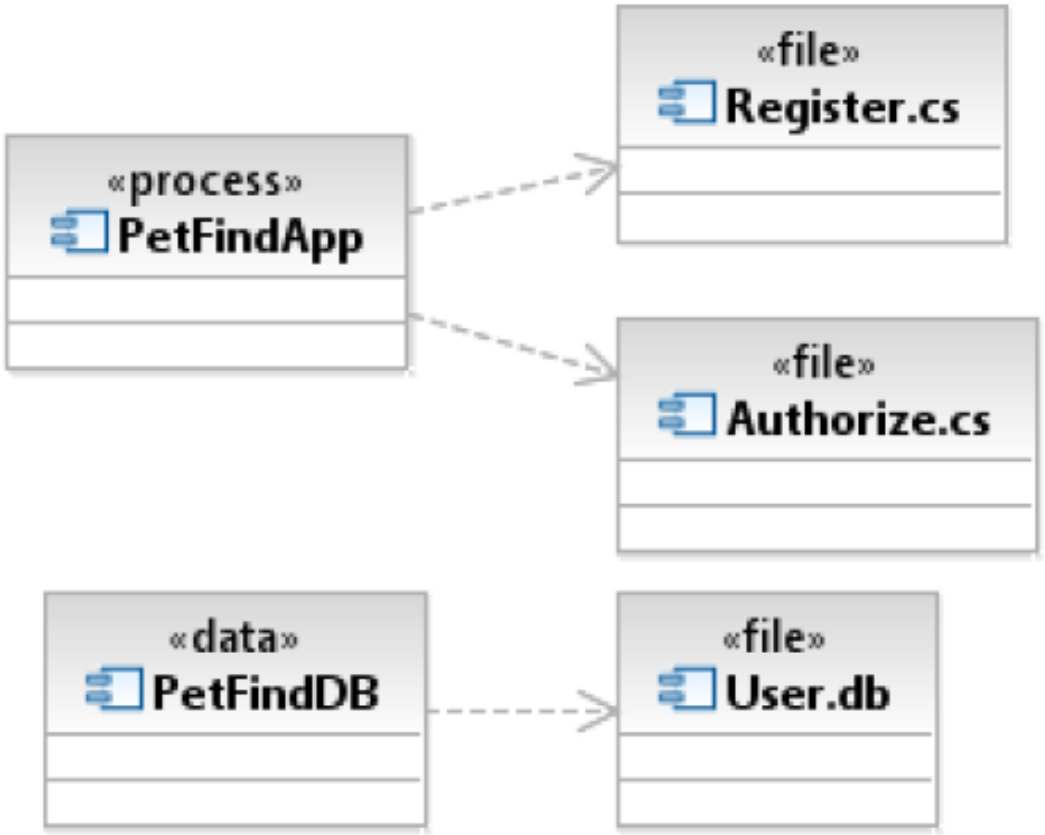
Сборка системы

- Планирование последующего билда
  - Билд = функциональность (>= 10) + специальные требования + список компонентов и подсистем реализации .
  - До итерации определяется набор билдов (1 итерация -> N билдов):
    - реализует набор прецедентов / сценариев;
    - инкремент функциональности;
    - не очень много новых компонентов (могут быть заглушки);
    - развивается вверх и в сторону по иерархии подсистем (начало с нижнего уровня);
  - И состав билда:
    - Определить проект прецедента;
    - Определить классы подсистем прецедента;
    - Определить подсистемы и компоненты ренализации;
    - Определить действия по реализации и проверить реализуемость и соответствие требованиям к билду;
    - если ОК -> реализация подсистем и компонентов -> тестирование -> интеграция.
- Сборка билда:
  - компиляция снизу вверх;
  - подбор правильных версий подсистем и компонентов.

Трасировка классов в исходный файлы компонентов



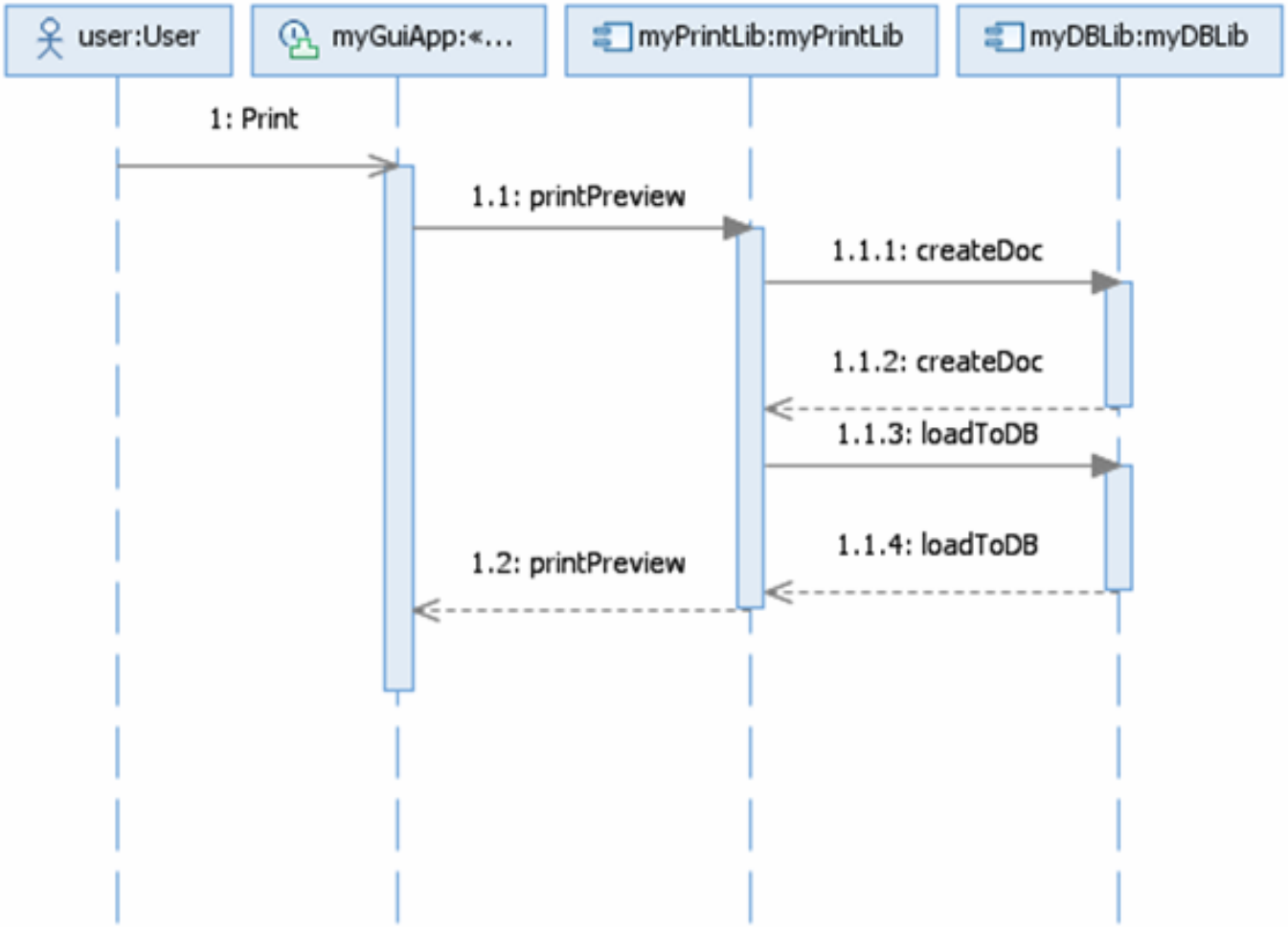
Зависимость компонентов от исходных файлов



Реализация подсистемы

- Работа с содержимым подсистемы:
  - каждый класс проектирования должен трассироваться в файл реализации в компоненте;
  - любой интерфейс должен быть представлен в реализации.
- Реализация подсистемы:
  - подсистема - это проект (c++), пакет (java) и так далее.

Диаграмма последовательностей в терминах компонентов



Реализация класса

- Описание файлов компонентов:
  - зависит от языка программирования.
- Генерация кода для класса проектирования:
  - автоматически генерируется сигнатура,
  - однонаправленная ассоциация = ссылки (коллекция ссылок), (имя атрибута = роль).
- Реализация операций:
  - вручную.
- Создание компонентов, представляющих правильный интерфейсе:
  - должно быть обеспечено.

Тестирование модулей

- Тестирование спецификации
  - черный ящик.
- Тестирование структуры
  - белый ящик.

Реализация - артефакты

- Реализация архитектуры:
  - Трассировка подсистем в компоненты с сохранением интерфейсов.
  - Диаграмма размещения компонентов.
- Реализация подсистем и классов:
  - Диаграммы последовательностей в терминах подсистем для кооперации.
  - Трассировка классов в исходные файлы.
  - Зависимости компонентов от исходных файлов.
- Сборки:
  - Перечень и состав билдов на следующую итерацию.
- Модульные тесты:
  - структурные и функциональные для билдов.

Рабочий процесс - Тестирование

- Планирование тестирования
  - стратегия;
  - оценка ресурсов тестирования;
  - график работы.
- Разработка тестов:
  - тесты целостности (функции системы);
  - системные тесты (ресурсы, параллельные запросы);
  - регрессионное тестирование (для билда после сборки);
  - процедуры тестирования.
- Реализация тестов
  - автоматическая (на основе процедуры тестирования).
- Тестирование целостности
  - для билда.
- Тестирование системы
  - системные тесты для итерации.
- Оценка результатов тестирования
  - полнота тестирования;
  - надежность.

Тестирование - артефакты

- план тестирования;
- тестовые примеры;
- процедуры тестирования;
- тестовые компоненты (автоматическое тестирование) ;
- дефекты/ошибки;
- оценка тестов.