Проектирование программного обеспечения

Лекция 1: Об архитектуре

Юрий Литвинов
y.litvinov@spbu.ru

13.02.2023

Организационное

- Лекционно-практический курс, две пары в неделю
- В конце устный экзамен и аттестация по домашним работам
- Работа на практиках за небольшие баллы
- Материалы по курсу и условия домашних работ: bit.ly/sd-2023
- Балльная система
  - 40% за домашки, 60% за экзамен
- Коммуникации — в чате курса в Telegram, https://t.me/+Bo1bNQ1ZTq42ZWIy
  - Также пишите в Telegram в личку, https://t.me/yurii_litvinov

Что будет в курсе

- Объектно-ориентированное проектирование
- Моделирование, язык UML и, немного, другие визуальные языки
- Шаблоны проектирования и антипаттерны
- Архитектурные стили
- Предметно-ориентированное проектирование
- Проектирование распределённых приложений
- Развёртывание и DevOps
- Примеры архитектур

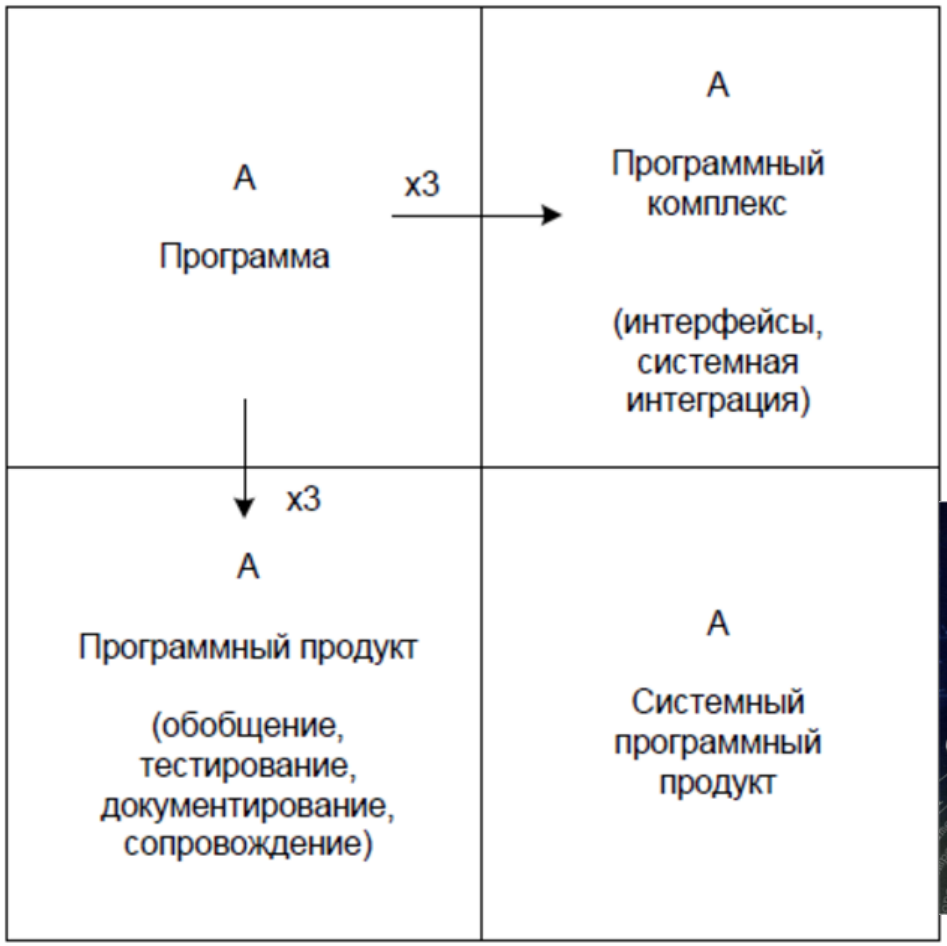
Критерии оценивания

- Меньше 60% заданий – 0
- От 60% до 100% — линейная шкала от 2 до 10 баллов
  - то есть, ровно 60% заданий — 2 балла
  - 80% заданий — 6 баллов
  - 100% заданий — 10 баллов
- Задачи оцениваются по 10 баллов
- Оценки за работу на паре с небольшим весом (ближайшая такая пара через две недели)
  - Оценка входит в процент выполненных заданий!
- В итоговой оценке практика учитывается с весом 0.4, экзамен — 0.6
- Округление арифметическое
- “Принцип мажорирующей двойки”

Комментарии по д/з

- Сдавать пуллреквестом в свой репозиторий и ссылку на HwProj
- Будет два больших командных задания и немного мелких
- Требуется архитектурная документация
  - Комментарии к каждому классу, интерфейсу и public-методу
  - Краткое описание деталей реализации в README
- Требуется CI и юнит-тесты
- Язык программирования — любой
- Овердизайн и активное использование знаний с лекций приветствуются
- Некоторые требования могут показаться ненужными — это нормально
  - Мы учимся не написанию кода, а инструментам и техникам проектирования

Программа и программный продукт



© Ф. Брукс, «Мифический человеко-месяц»

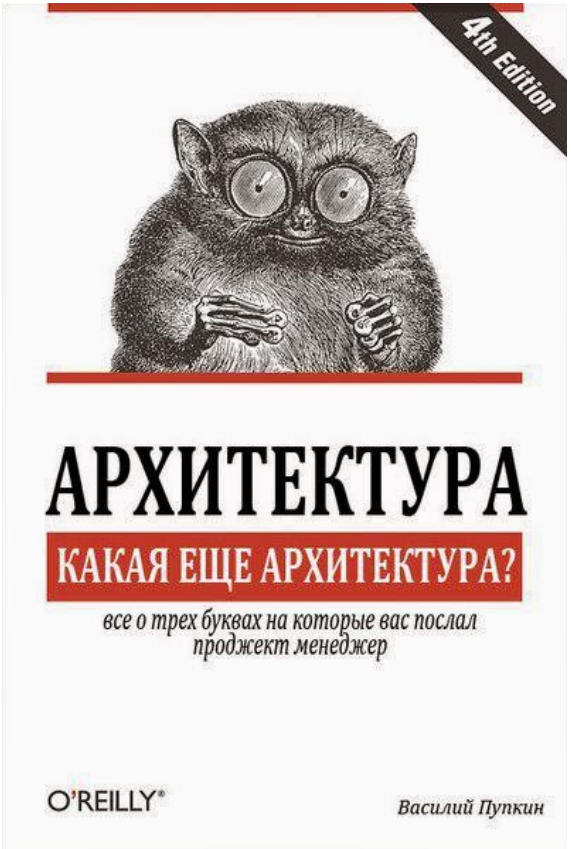
Размер типичного ПО

Простая игра для iOS	10000 LOC
Unix v1.0 (1971)	10000 LOC
Quake 3 engine	310000 LOC
Windows 3.1 (1992)	2.5M LOC
Linux kernel 2.6.0 (2003)	5.2M LOC
MySQL	12.5M LOC
Microsoft Office (2001)	25M LOC
Microsoft Office (2013)	45M LOC
Microsoft Visual Studio 2012	50M LOC
Windows Vista (2007)	50M LOC
Mac OS X 10.4	86M LOC

http://www.informationisbeautiful.net/visualizations/million-lines-of-code/

Архитектура

- Совокупность важнейших решений об организации программной системы
  - Эволюционирующий свод знаний
  - Разные точки зрения
  - Разный уровень детализации
- Для чего
  - База для реализации, «фундамент» системы
  - Инструмент для оценки трудоёмкости и отслеживания прогресса
  - Средство обеспечения переиспользования
  - Средство анализа системы ещё до того, как она реализована



© Интернеты

Профессия «Архитектор»

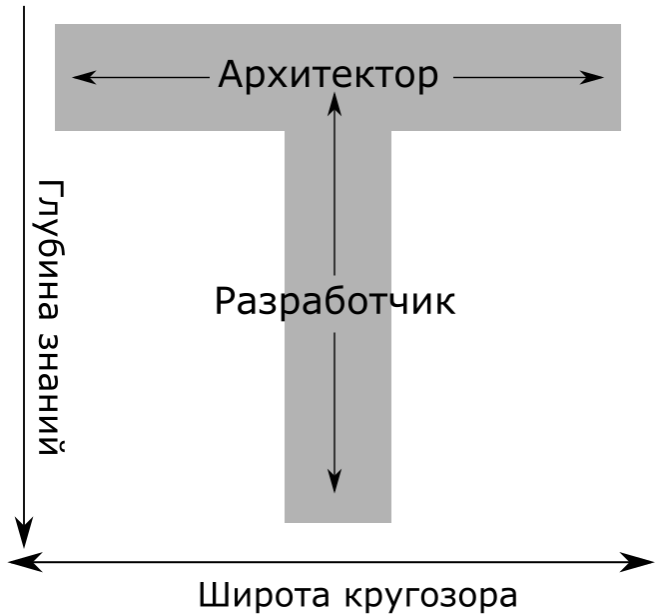
- Архитектор — специально выделенный человек (или группа людей), отвечающий за:
  - разработку и описание архитектуры системы
  - доведение её до всех заинтересованных лиц
  - контроль реализации архитектуры
  - поддержание её актуального состояния по ходу разработки и сопровождения

Трудовые функции архитектора

По профстандарту 06.003

- Выявление и согласование требований к программной системе с точки зрения архитектуры
- Выбор и моделирование архитектурного решения для реализации программной системы
- Разработка разделов по архитектуре проектных и эксплуатационных документов программной системы
- Контроль реализации и испытаний программной системы с точки зрения архитектуры
- Сопровождение эксплуатации программной системы с точки зрения архитектуры

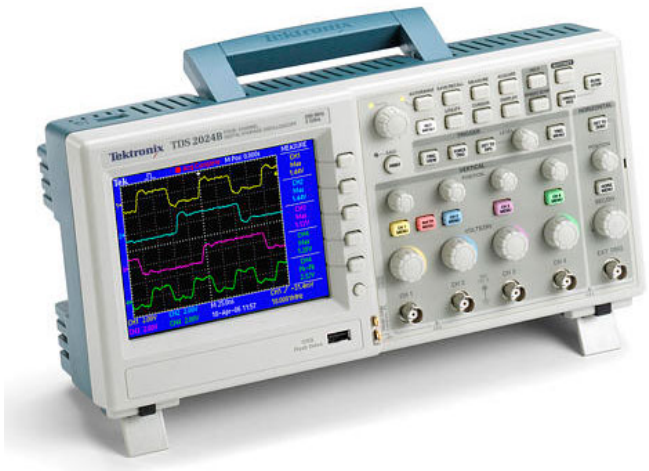
Архитектор vs разработчик



- Широта знаний
- Коммуникационные навыки
- Часто архитектор играет роль разработчика и наоборот
  - Архитектор в «башне из слоновой кости» — это плохо

Пример: ПО для осциллографа

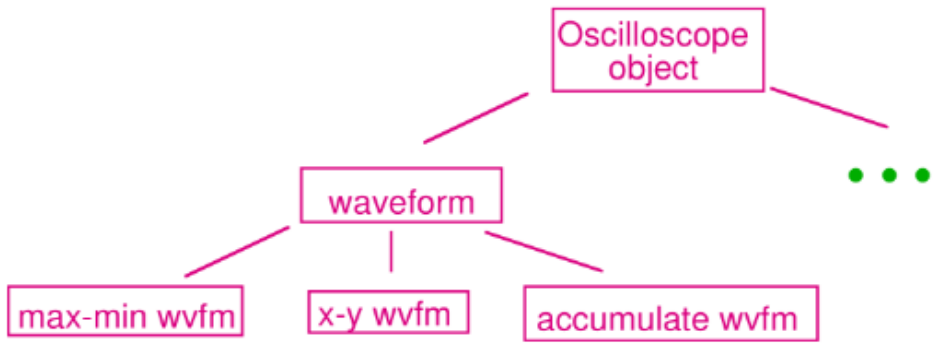
- Считывать параметры сигнала
- Оцифровывать и сохранять их
- Выполнять разные фильтрации и преобразования
- Отображать результаты на экране
  - С тач-скрином и встроенным хелпом
- Возможность настройки под конкретные задачи



© Hantek

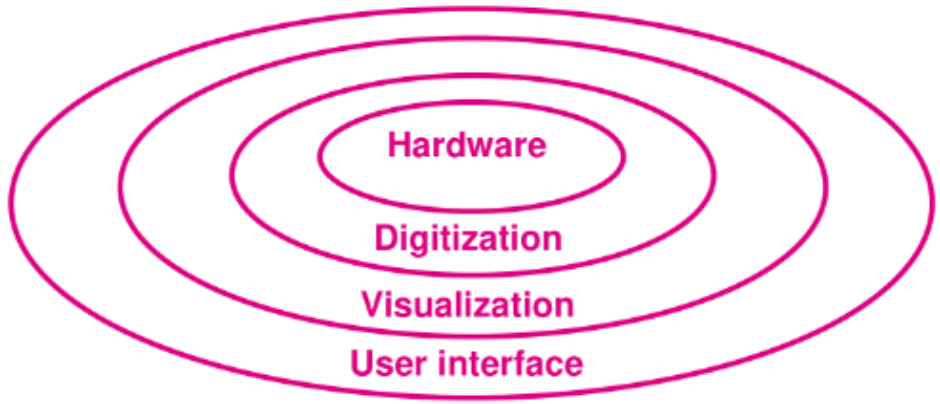
По статье Garlan D., Shaw M. An introduction to software architecture

Вариант 1: объектная модель



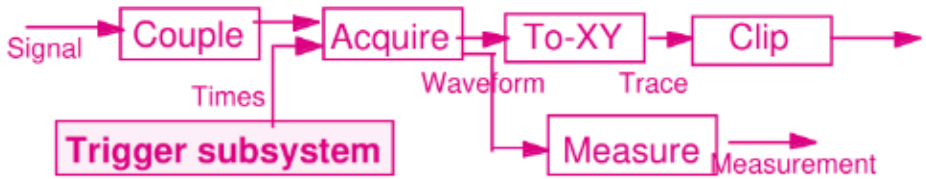
© Garlan D., Shaw M. An introduction to software architecture

Вариант 2: слоистая архитектура



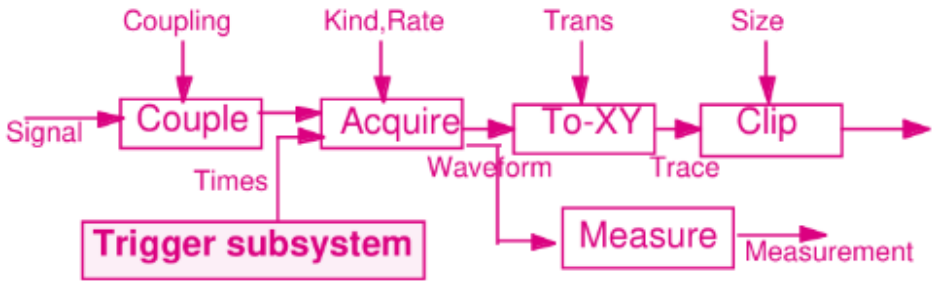
© Garlan D., Shaw M. An introduction to software architecture

Вариант 3: каналы и фильтры



© Garlan D., Shaw M. An introduction to software architecture

Вариант 4: модифицированные каналы и фильтры



© Garlan D., Shaw M. An introduction to software architecture

Выводы

- Можем делать утверждения о свойствах системы, базируясь на её структурных свойствах
  - Не написав ни строчки кода и даже не выбрав язык реализации
- Рассуждения очень субъективны
  - Многое зависит от интуиции и вкуса архитектора, однако ошибки очень дороги
- Можно выделить архитектурные стили — «архитектуры архитектур»
- Можно выделить архитектурные точки зрения и архитектурные виды
- Разный уровень детализации

Архитектурные виды

Стандарт IEEE 1016-2009

- Контекст — фиксирует окружение системы
- Композиция — описывает крупные части системы и их предназначение
- Логическая структура — структура системы в терминах классов, интерфейсов и отношений между ними
- Зависимости — определяет связи по данным между элементами
- Информационная структура — определяет персистентные данные в системе
- Использование шаблонов — документирование использования локальных паттернов проектирования

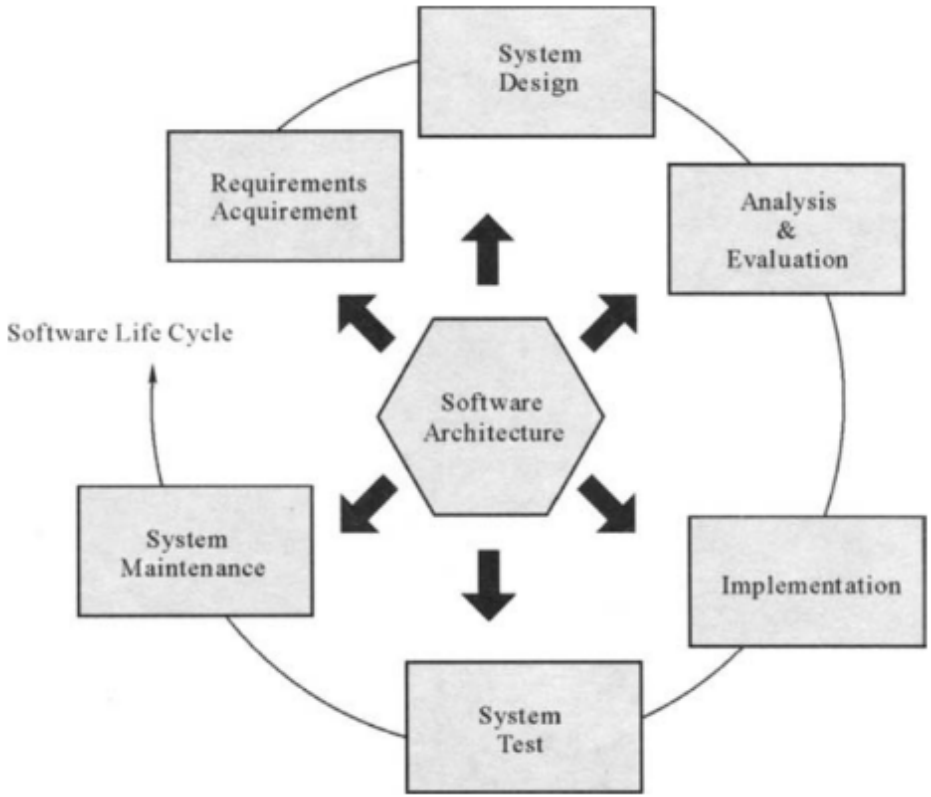
Архитектурные виды (2)

- Интерфейсы — специфицирует информацию о внешних и внутренних интерфейсах
- Структура системы — рекурсивное описание внутренней структуры компонентов системы
- Взаимодействия — описывает взаимодействие между сущностями
- Динамика состояний — описание состояний и правил переходов между состояниями
- Алгоритмы — описывает в деталях поведение каждой сущности
- Ресурсы — описывает использование внешних ресурсов

Ещё про архитектурные виды

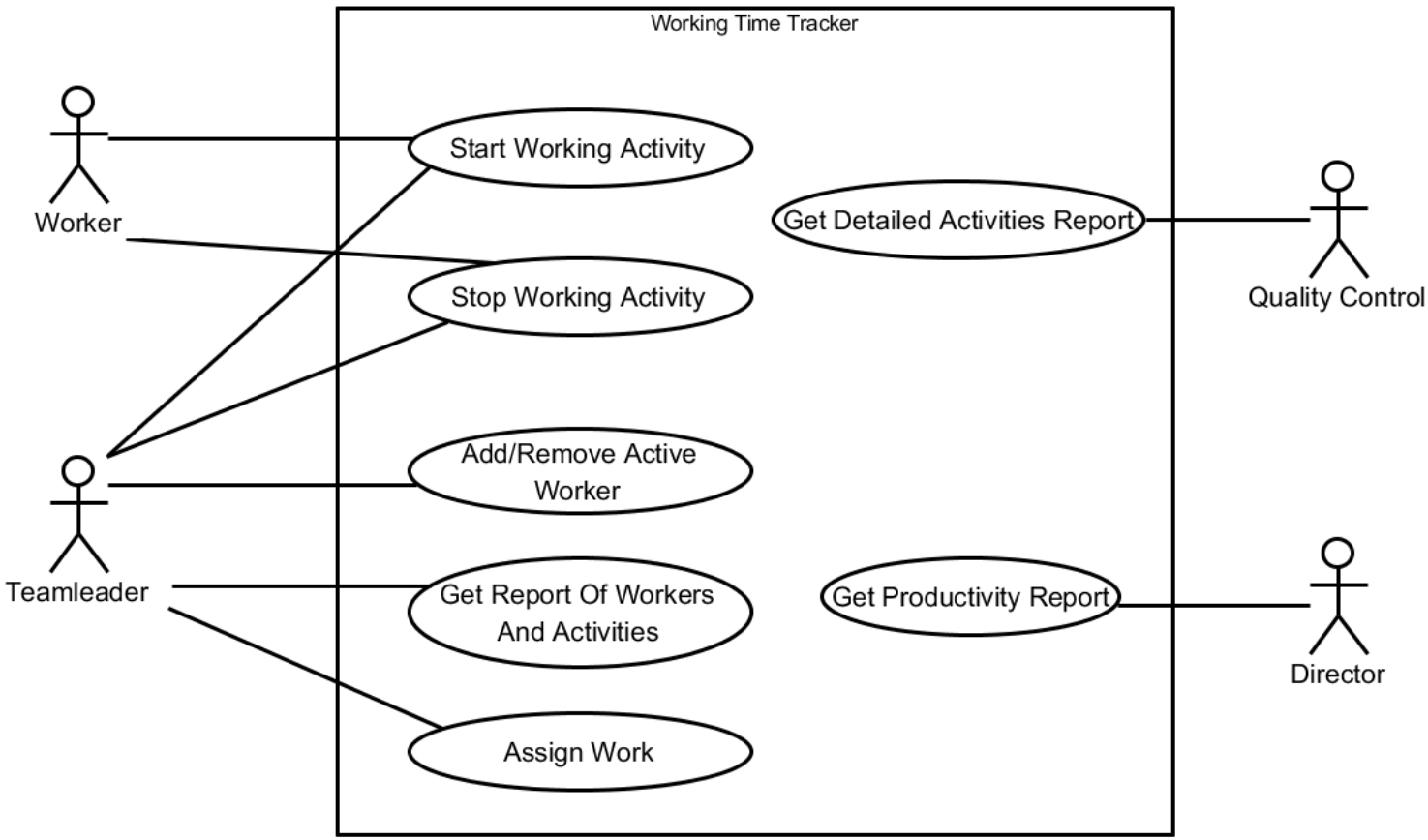
- Пример — http://robotics.ee.uwa.edu.au/courses/design/examples/example_design.pdf
- Ни один вид не обязателен
- Активно используются визуальные языки
  - В основном как дополнение к тексту
- Моделирование принципиально важно для архитектуры
  - Нельзя абстрагироваться от сложности, но можно декомпозировать сложность

Роль архитектуры в жизненном цикле



© Z. Quin, “Software Architecture”

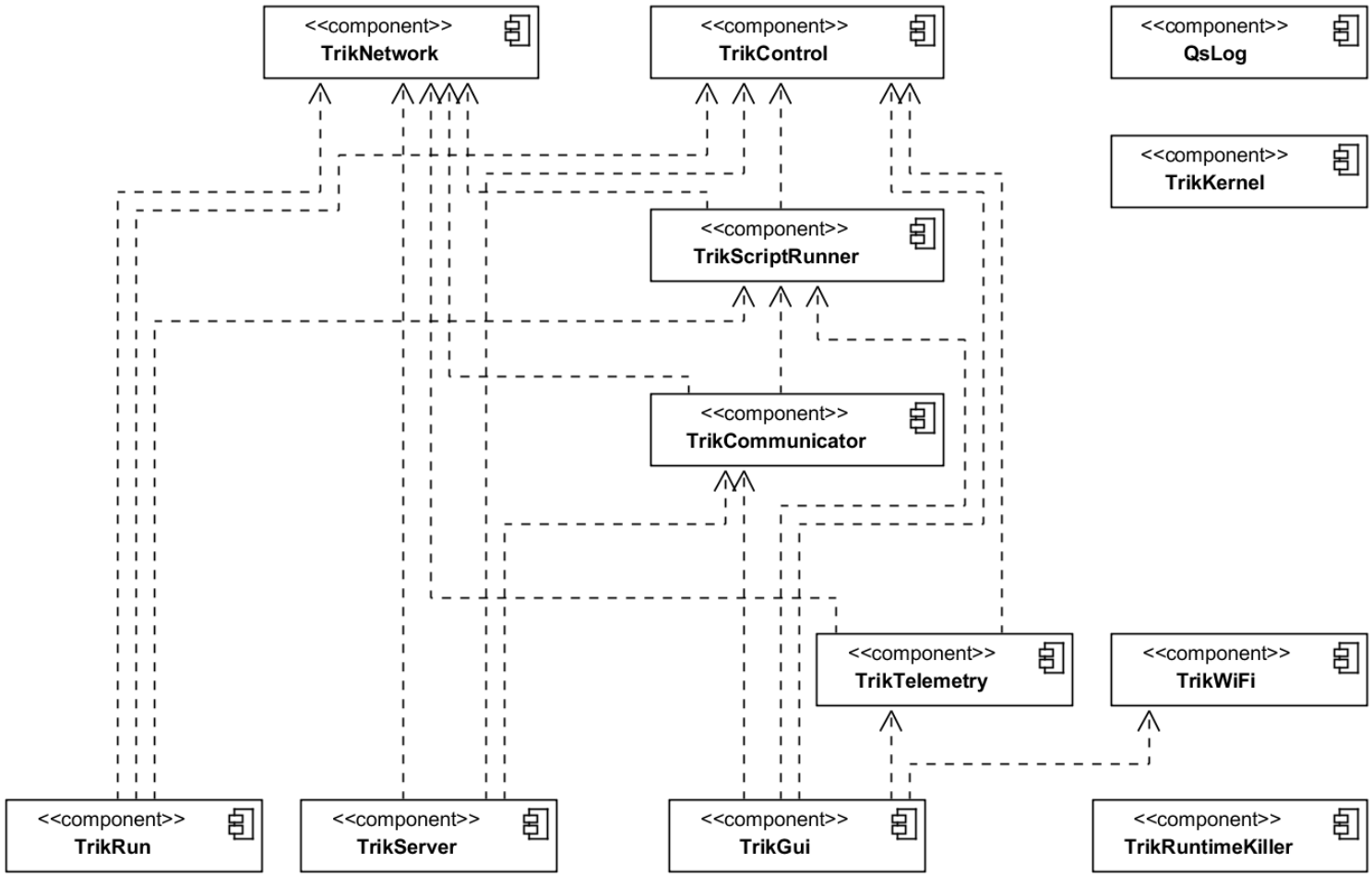
Архитектура и требования



Требования

- Функциональные — то, что система должна делать
- Нефункциональные — то, как система должна это делать
  - Эффективность
  - Масштабируемость
  - Удобство использования
  - Надёжность
  - Безопасность
  - Сопровождаемость и расширяемость
  - ...
- Ограничения
  - Технические
  - Бизнес-ограничения

Архитектура и проектирование



Архитектура и проектирование — задачи

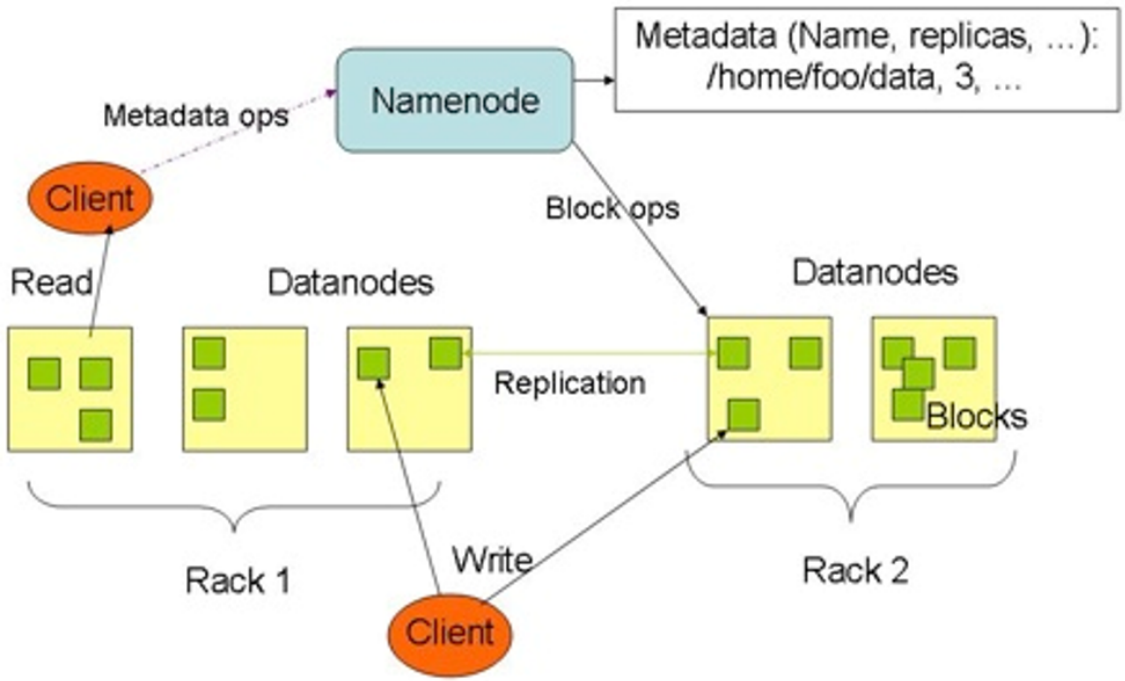
- Декомпозиция задачи
- Определение границ компонентов
- Определение интерфейсов между компонентами
- Общие для всей системы вопросы
  - Стратегия обработки ошибок
  - Стратегия логирования
  - Стратегия обновлений
  - Стратегия разделения доступа
  - Вопросы локализации
  - ...
- Анализ и верификация архитектуры

Архитектура и разработка

- prescriptive architecture — архитектура, как её определил архитектор
- descriptive architecture — архитектура, как она есть в системе
  - Архитектура у ПО есть всегда, как вес у камня
- architectural drift — «сползание» фактической архитектуры
  - появление в ней важных решений, которых нет в описательной архитектуре
- architectural erosion — «размывание» архитектуры
  - отклонения от описательной архитектуры, нарушения ограничений
- Антипаттерн «Big ball of mud» — результат эрозии

Пример: Hadoop

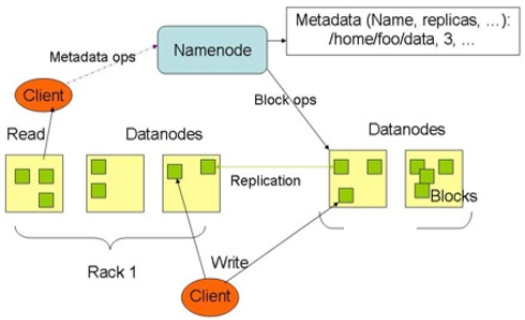
As-designed

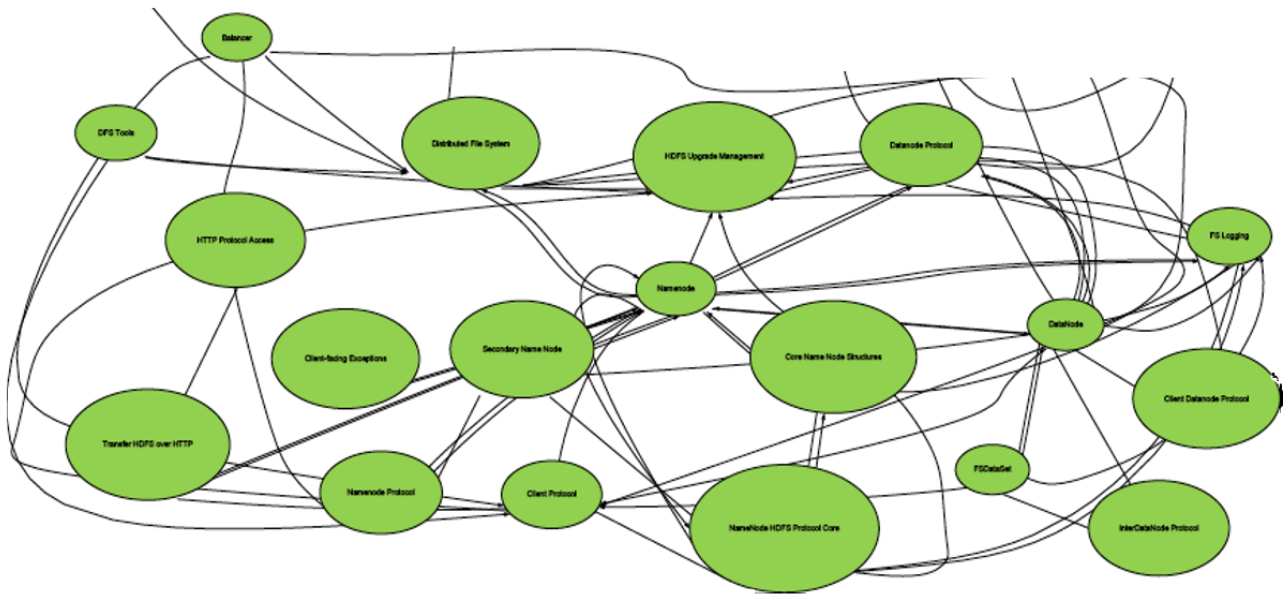


Special thanks to prof. Nenad Medvidovic (USC) for kind permission for using his slides

Hadoop as-built

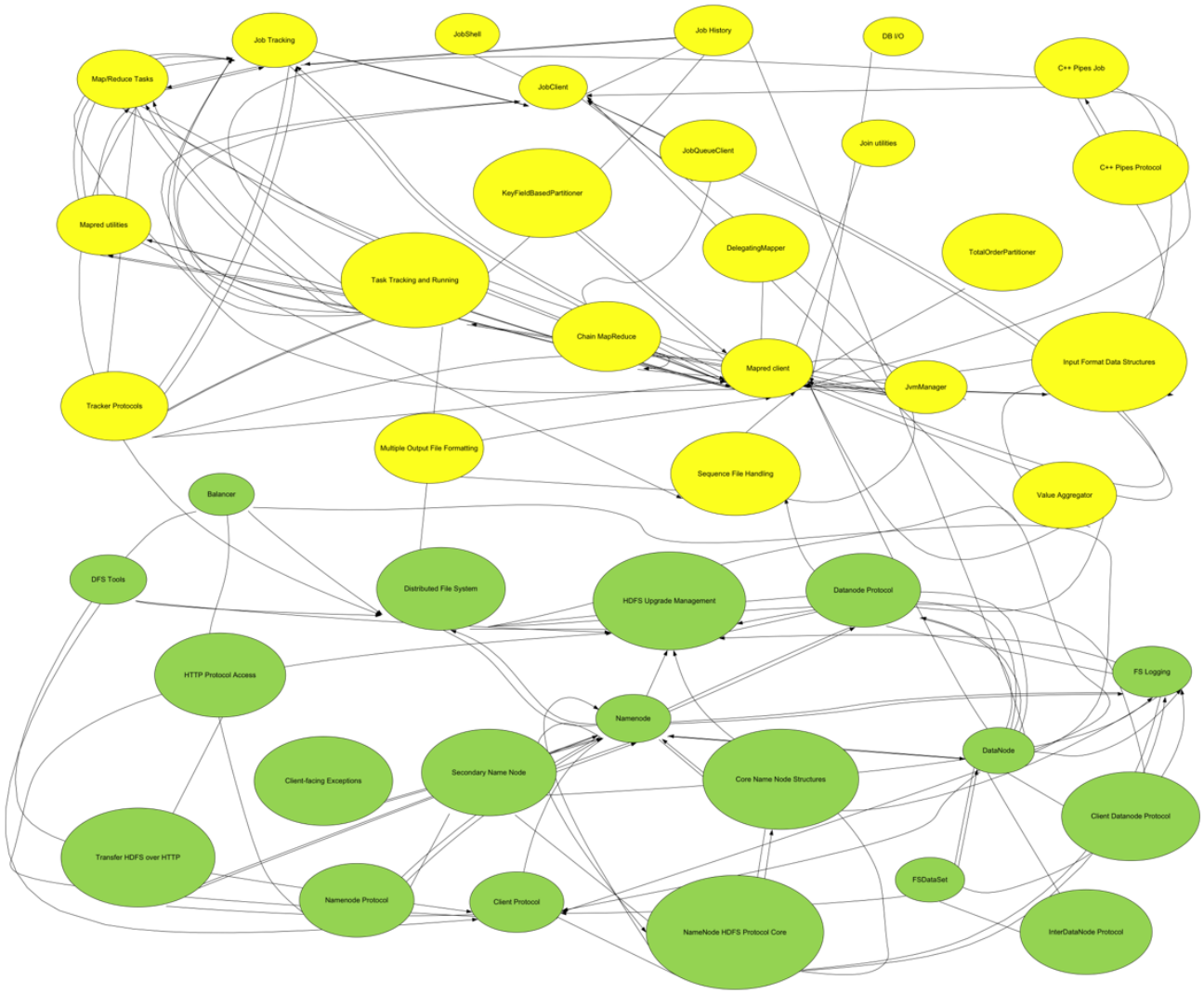
HDFS





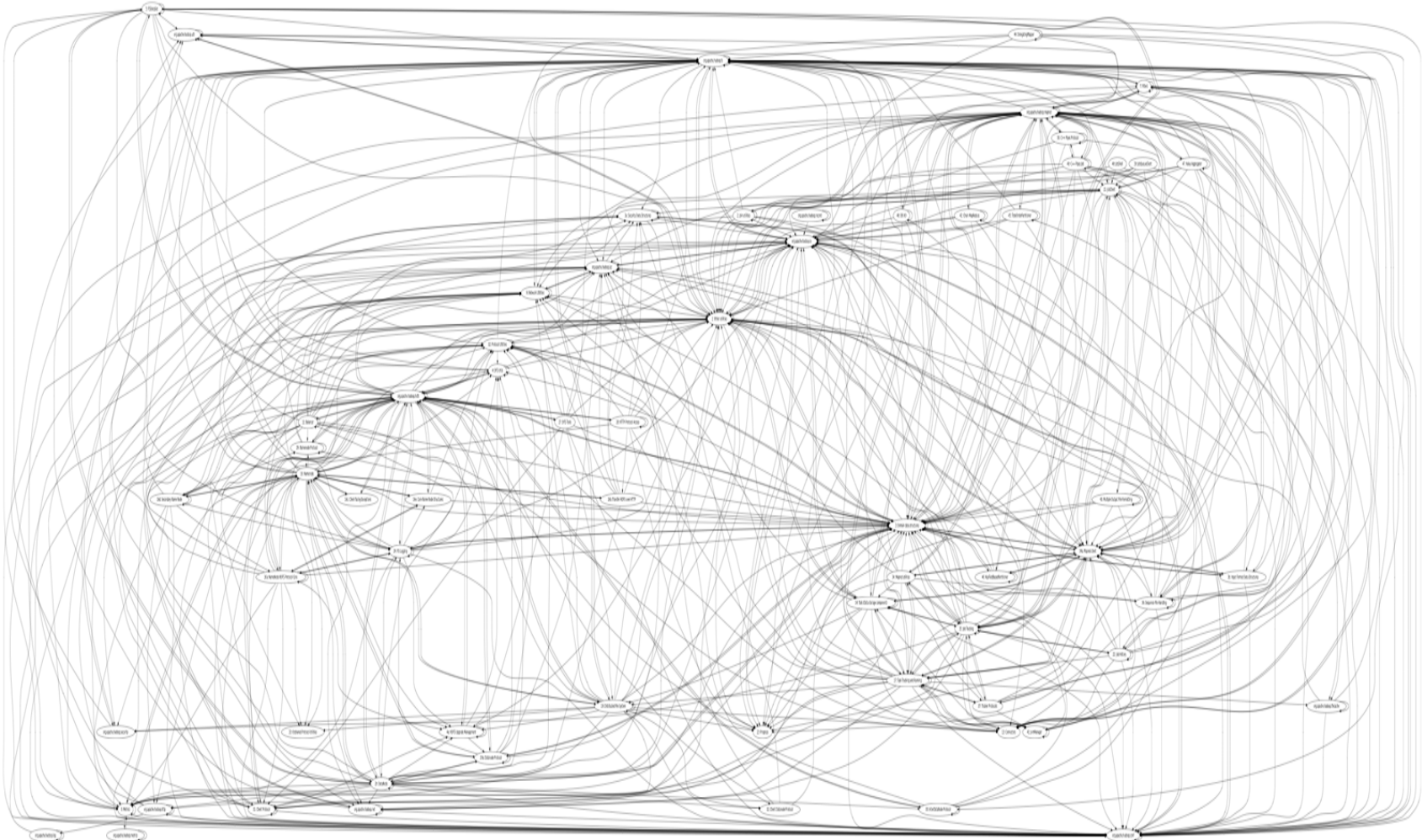
Hadoop as-built

HDFS + MapReduce



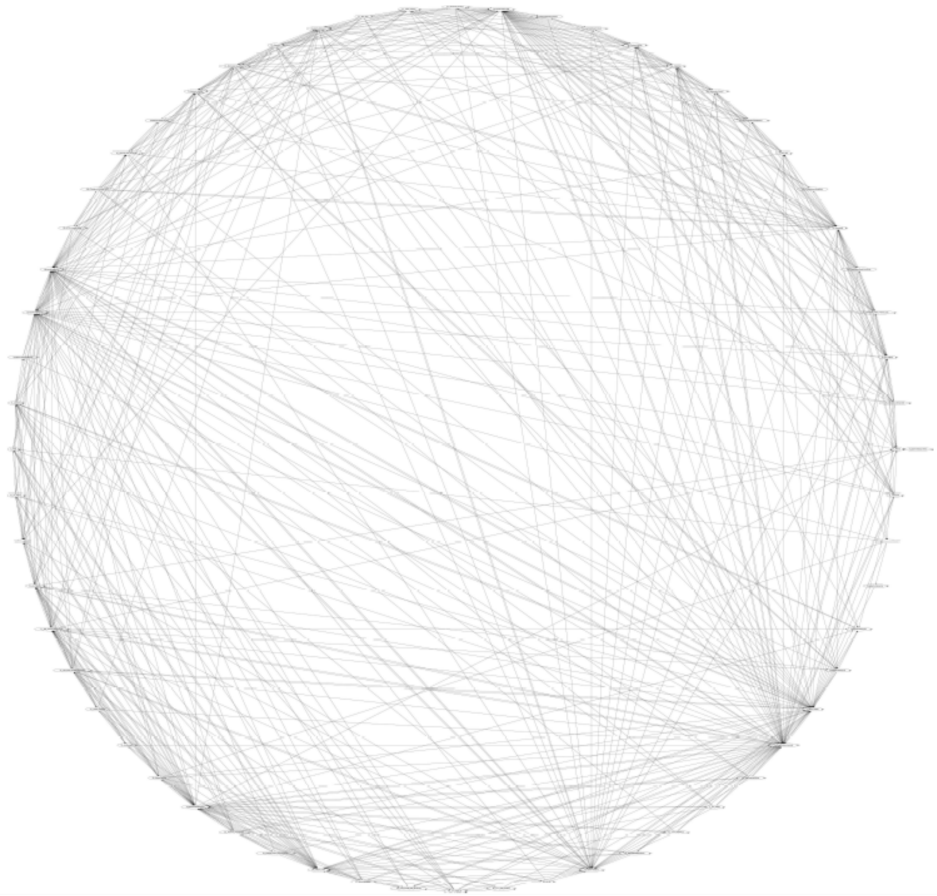
Hadoop as-built

Полная архитектура



Hadoop as-built

Граф зависимостей



© Nenad Medvidovic