

Архитектура ПО

Лекция 1: Об архитектуре

Юрий Литвинов
yurii.litvinov@gmail.com

12.02.2020г

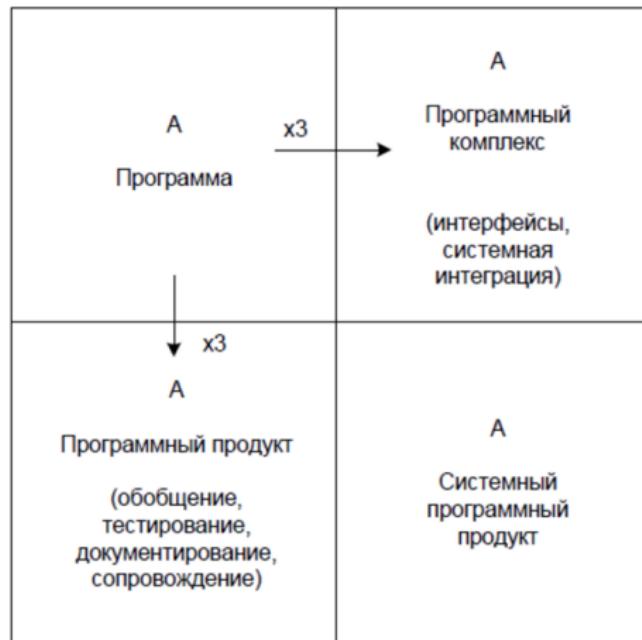
Организационное

- ▶ Курс с лекциями и практиками
- ▶ Теоретический экзамен в конце + зачёт по практикам
 - ▶ Много домашки
 - ▶ Экзамен бывало что не сдавали
- ▶ Лекции будет читать Тимофей Брыксин
(timofey.bryksin@gmail.com)

Что будет в курсе

- ▶ Про профессию архитектора, красивый ОО-код и т.п.
- ▶ Визуальное моделирование, UML
- ▶ Паттерны проектирования, архитектурные стили
- ▶ Предметно-ориентированное проектирование
- ▶ Проектирование распределённых приложений
- ▶ Примеры архитектур

Программа и программный продукт



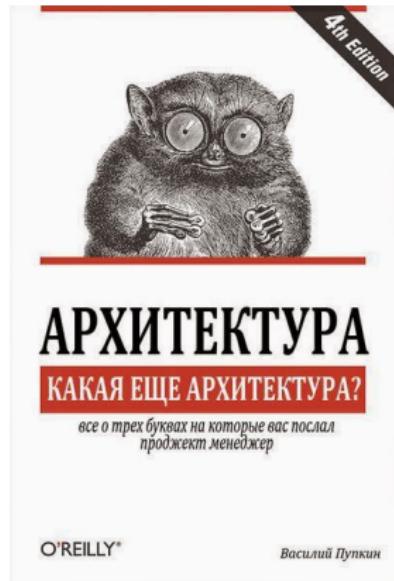
Размер типичного ПО

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
- ▶ Заказчику на неё почти всегда плевать



Разработка ПО vs Строительство

▶ Процесс

- ▶ Сбор требований
 - ▶ Высокоуровневая архитектура
 - ▶ Уточнение и формализация в моделях
 - ▶ Производство
 - ▶ Приёмка и сдача
 - ▶ Эксплуатация



Некоторые выводы из метафоры

- ▶ Процесс разработки учитывает пожелания и требования пользователей
- ▶ Возможно разделение труда
 - ▶ Архитекторы
 - ▶ Реализаторы
- ▶ Прогресс контролируется в ключевых точках

Архитектура как отдельная сущность

- ▶ Архитектура — есть!

- ▶ Проектируется, развивается, документируется, тестируется, сравнивается с другими, проверяется на воплощение в коде
- ▶ Даже если про неё никто специально не думал

Поэтому:

- ▶ Откуда берётся архитектура?
- ▶ Чем она характеризуется?
- ▶ Какими свойствами обладает?
- ▶ Что такое хорошая и плохая архитектура?

Архитектура и качества системы



Требования Заказчика



Согласовано с разработчиком



Техническое задание



Результат разработки



Сдано Заказчику



Осталось после опытной эксплуатации



Что было нужно на самом деле

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

Профессиональный стандарт «Архитектор»

Создание и сопровождение архитектуры программных средств, заключающейся

- ▶ в синтезе и документировании решений о структуре
- ▶ в компонентном устройстве
- ▶ в основных показателях назначения
- ▶ порядке и способах реализации программных средств в рамках системной архитектуры
- ▶ реализации требований к программным средствам
- ▶ контроле реализации и ревизии решений

Трудовые функции архитектора

По стандарту

- ▶ Создание вариантов архитектуры программного средства
- ▶ Документирование архитектуры программных средств
- ▶ Реализация программных средств (*в основном контроль и анализ*)
- ▶ Оценка требований к программному средству
- ▶ Оценка и выбор варианта архитектуры программного средства
- ▶ Контроль реализации программного средства
- ▶ Контроль сопровождения программных средств
- ▶ Оценка возможности создания архитектурного проекта
- ▶ Утверждение и контроль методов и способов взаимодействия программного средства со своим окружением
- ▶ Модернизация программного средства и его окружения

Архитектор vs разработчик



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

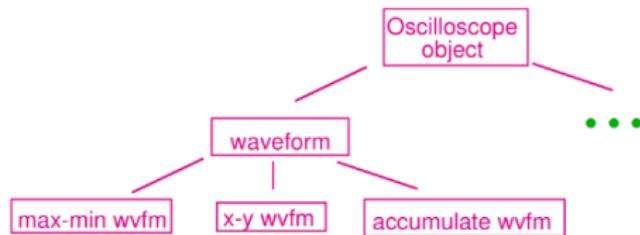
Пример: ПО для осциллографа

- ▶ 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

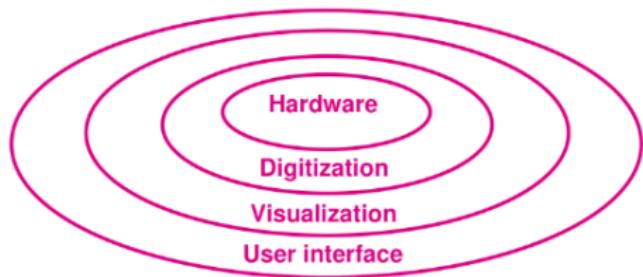


По статье Garlan D., Shaw M. An introduction to software architecture

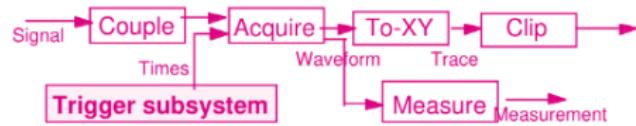
Вариант 1: объектная модель



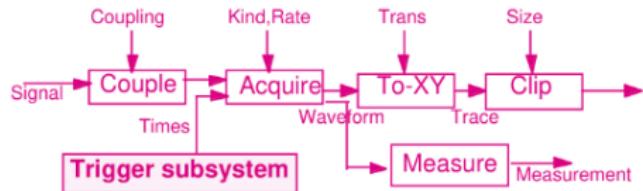
Вариант 2: слоистая архитектура



Вариант 3: каналы и фильтры



Вариант 4: модифицированные каналы и фильтры



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

Роль архитектуры в жизненном цикле

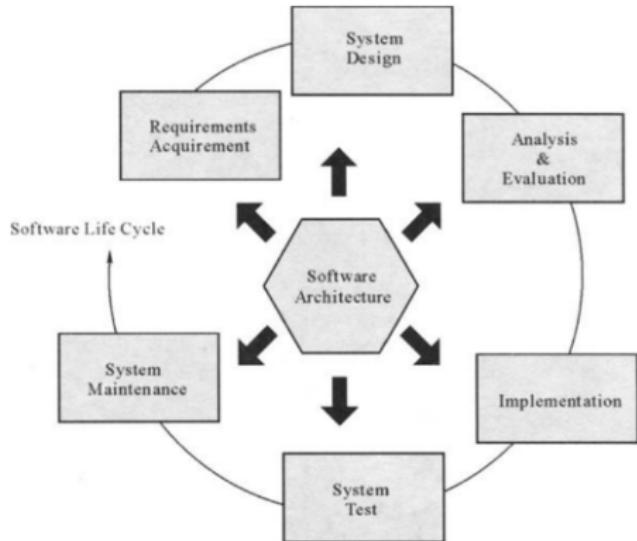
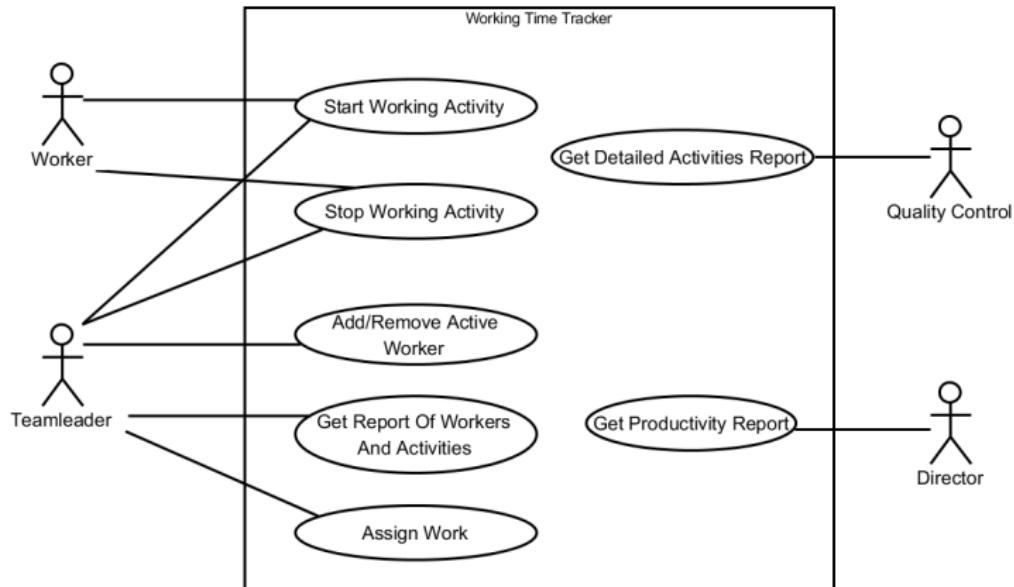


Рисунок из Z. Quin, "Software Architecture"

Архитектура и требования



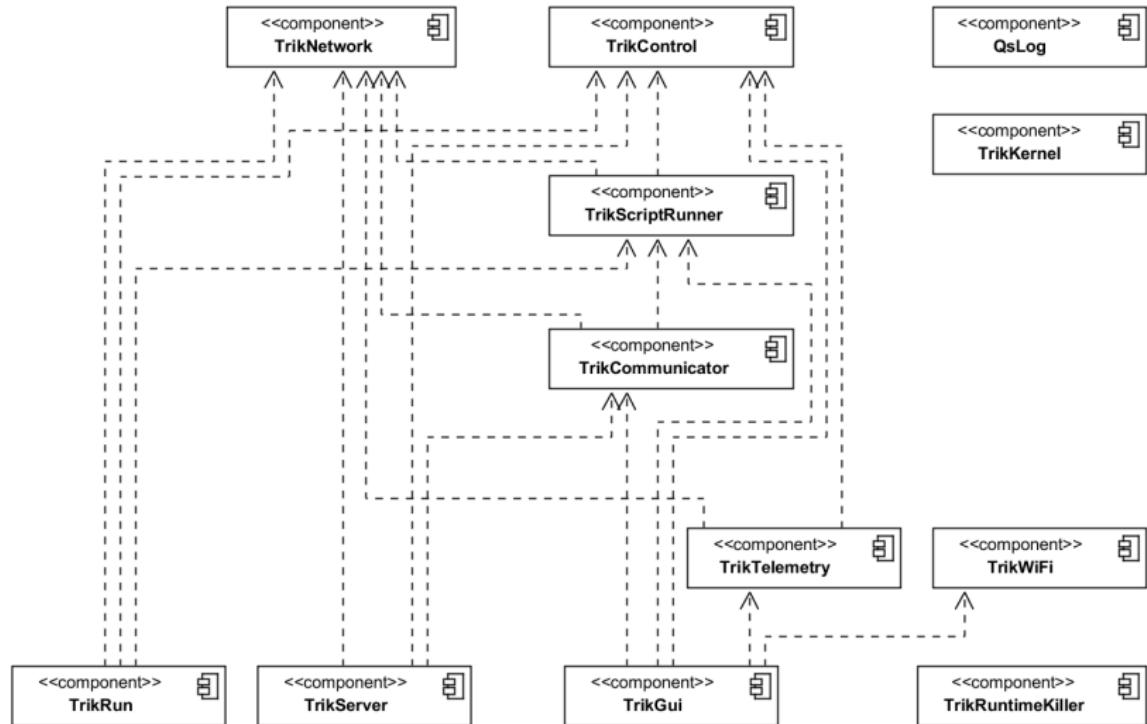
Требования

- ▶ 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

Требования и качество ПО



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 тестирование

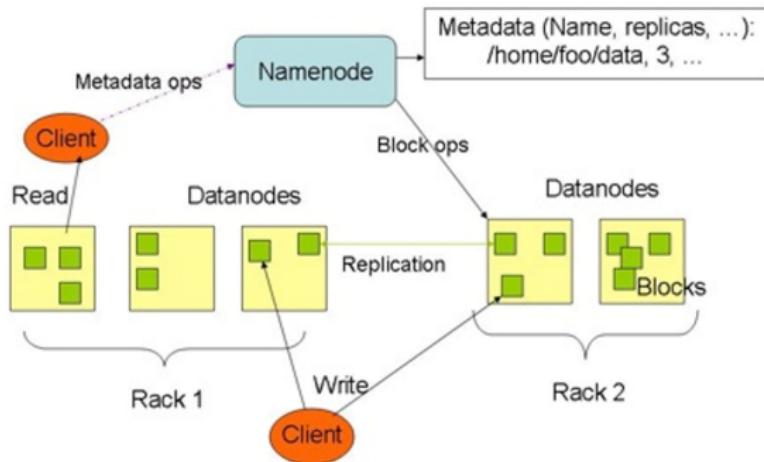
- ▶ Можно тестировать не только код, но и архитектуру
 - ▶ Целостность
 - ▶ Синтаксическая и семантическая корректность
 - ▶ Поиск антипаттернов
 - ▶ Control/data flow analysis
 - ▶ Анализ многопоточной работы
 - ▶ Поиск проблем с безопасностью
 - ▶ Оценка сложности и размера программы
 - ▶ Соответствие требованиям
- ▶ Архитектура как источник данных для тестов

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

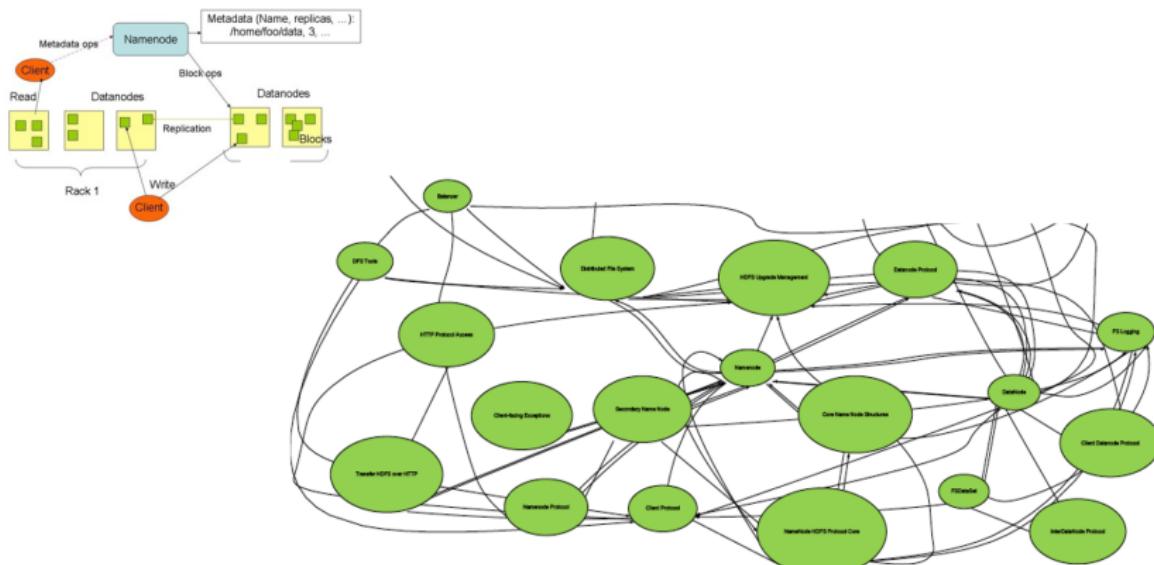
Пример: Hadoop

As-designed



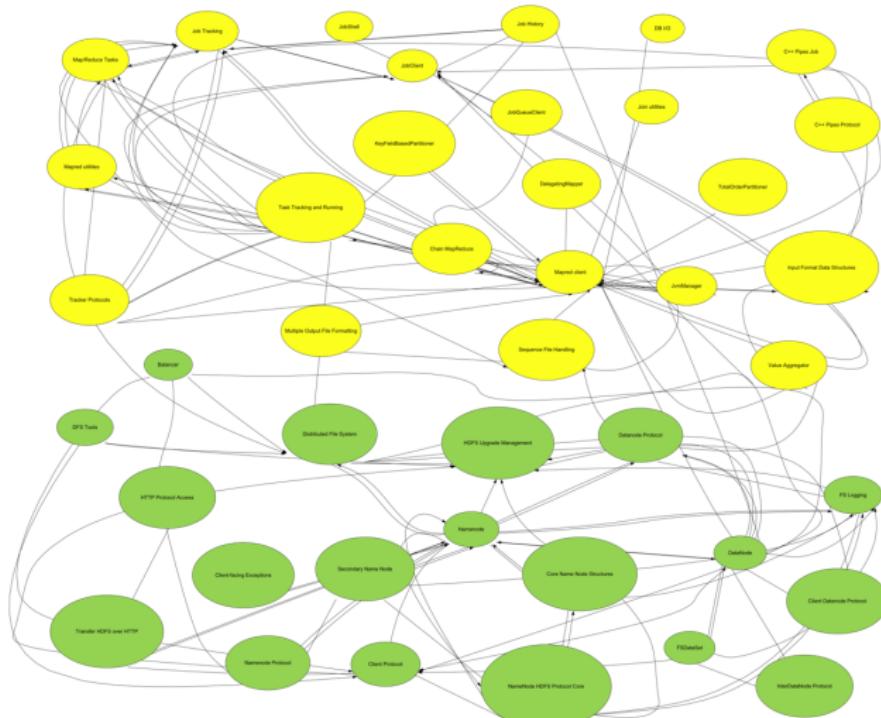
Special thanks to prof. Nenad Medvidovic (USC) for kind permission for using his slides

Hadoop as-built



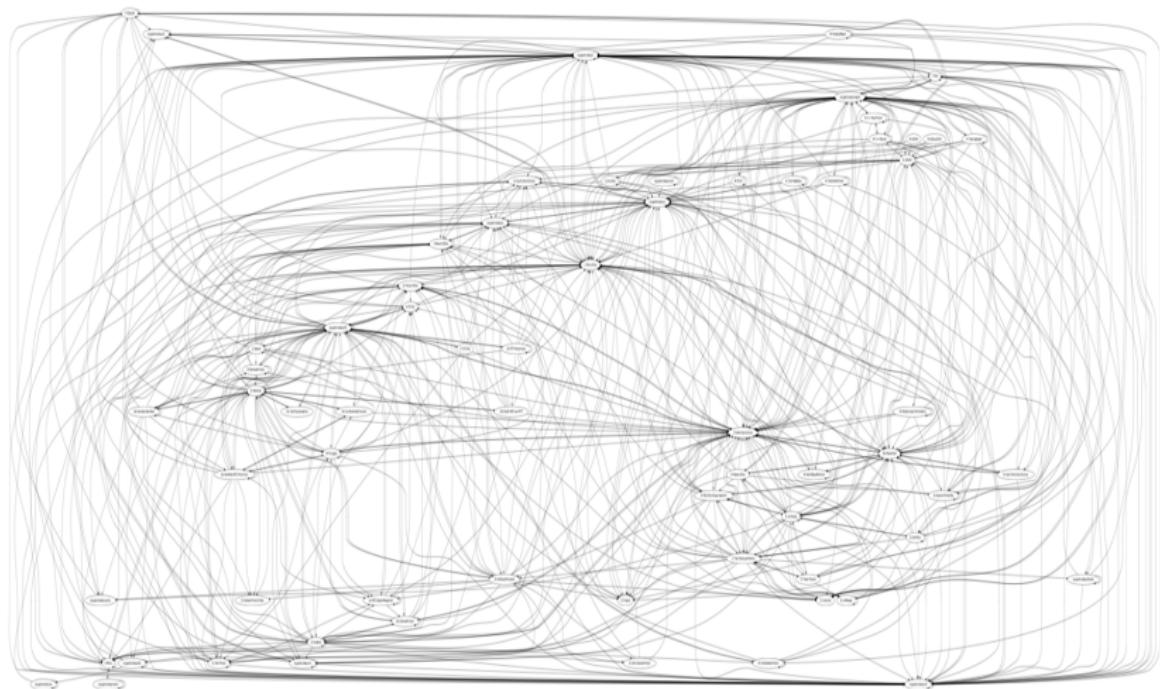
Hadoop as-built

HDFS + MapReduce



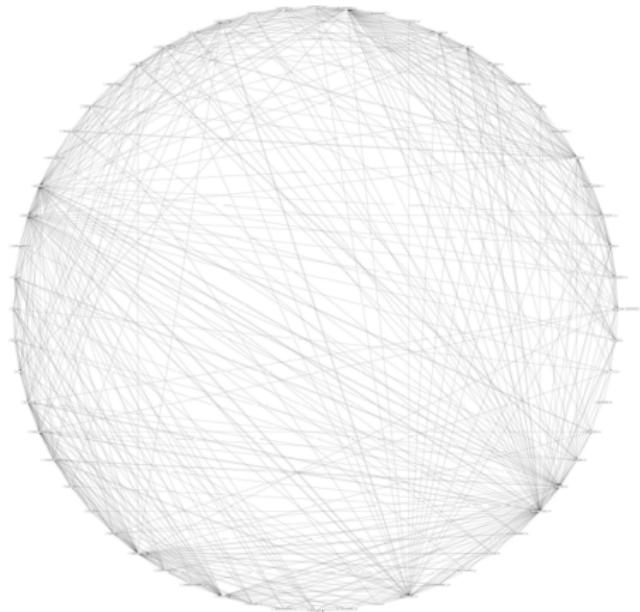
Hadoop as-built

Полная архитектура



Hadoop as-built

Граф зависимостей



Дальнейший план курса

- ▶ Принципы проектирования, декомпозиция
- ▶ Проектирование и ООП
- ▶ Моделирование при разработке архитектуры
- ▶ Стили/шаблоны проектирования
- ▶ Средства интеграции приложений
- ▶ Проектирование распределённых приложений
- ▶ Case studies