



Курс: «МАТЕМАТИЧЕСКИЕ МОДЕЛИ КОМПЛЕКСОВ ПРОГРАММ»

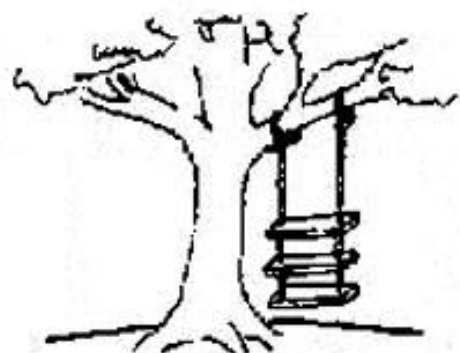
Модуль: **«АРХИТЕКТУРА ПРОГРАММНЫХ СИСТЕМ»**

Лекция 1 (Князьков К.В.)

ОБЗОР модуля

- Темы
 - Базовые понятия проектирования ПО: архитектура, атрибуты качества ПО.
 - Модели жизненного цикла ПО. Гибкие методологии разработки.
 - Моделирование ПО. Документирование архитектуры ПО.
 - Образцы проектирования архитектур. Паттерны ООП.
- Практика
 - Индивидуальные задания
 - Работа в командах

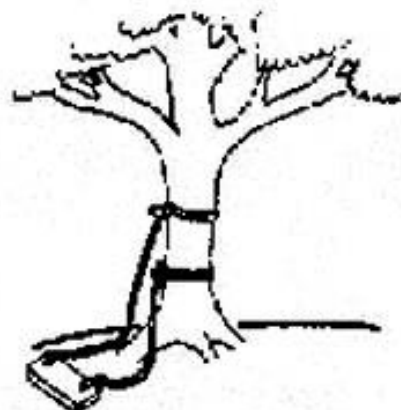
Как обычно пишутся программы



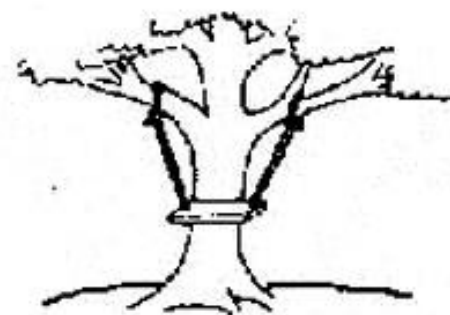
так было поставлено
техзадание



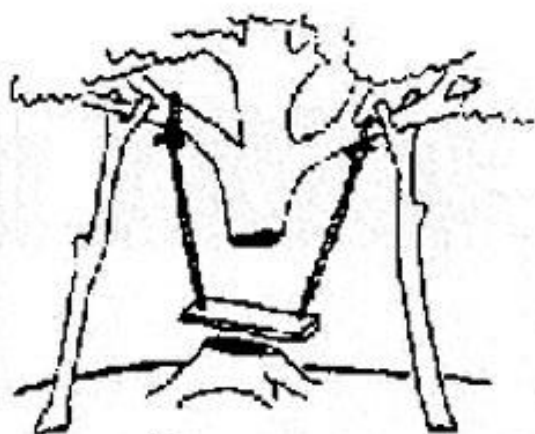
так его поняли
разработчики



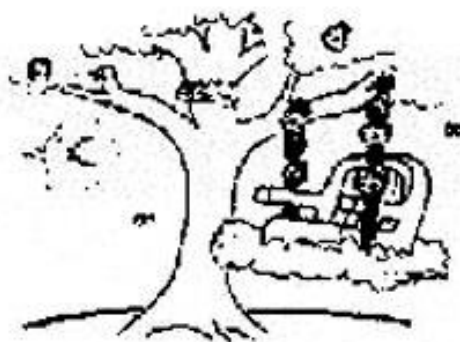
так эту задачу
решали раньше



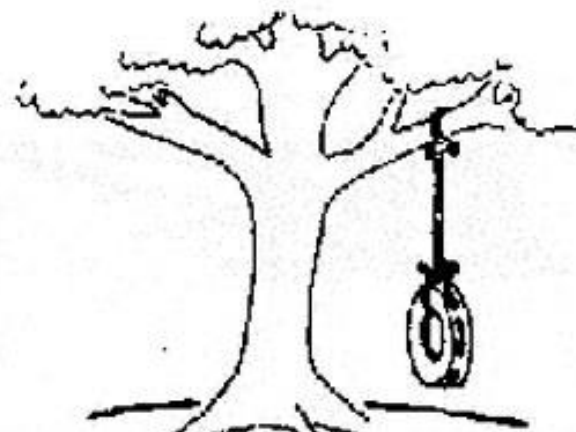
так ее решили теперь



такой программа стала
после отладки



так ее описали
в отделе рекламы



А, собственно, так ее
представлял себе
заказчик

ПРОБЛЕМЫ РАЗРАБОТКИ

1. Сложность
2. Изменчивость требований

ЧТО НАМ ПОМОЖЕТ?

- **Абстракция**
- Декомпозиция
- Модульность
(каждый модуль решает свою четко поставленную задачу)
- Повторное использование

64-БИТНЫЙ ПРОЦЕССОР БЕЗУСТАННО РАБОТАЕТ, ВЫПОЛНЯЯ НЕСКОЛЬКО МИЛЛИАРДОВ ОПЕРАЦИЙ В СЕКУНДУ, ЧТОБЫ ЗАПУСТИТЬ ЯДРО XNU, КОТОРОЕ ЧЕРЕЗ УРОВЕНЬ POSIX-СОВМЕСТИМОЙ АБСТРАКЦИИ ПОДНИМАЕТ СИСТЕМУ DARWIN, ЛЕЖАЩУЮ В ОСНОВЕ OS X, КОТОРАЯ, В СВОЮ ОЧЕРЕДЬ, НАПРЯГАЕТСЯ, ЧТОБЫ ЗАПУСТИТЬ FIREFOX И ЕГО ДВИЖОК GECKO, КОТОРЫЙ СОЗДАЕТ FLASH-ОБЪЕКТ, КОТОРЫЙ ОТРИСОВЫВАЕТ НЕСКОЛЬКО ДЕСЯТКОВ КАДРОВ ВИДЕО В СЕКУНДУ.

И ВСЁ ИЗ-ЗА ТОГО, ЧТО Я ХОЧУ ПОСМОТРЕТЬ НА ТО, КАК КОШКА ЗАПРЫГИВАЕТ В КОРОБКУ И СПОТЫКАЕТСЯ.

я — БОГ.



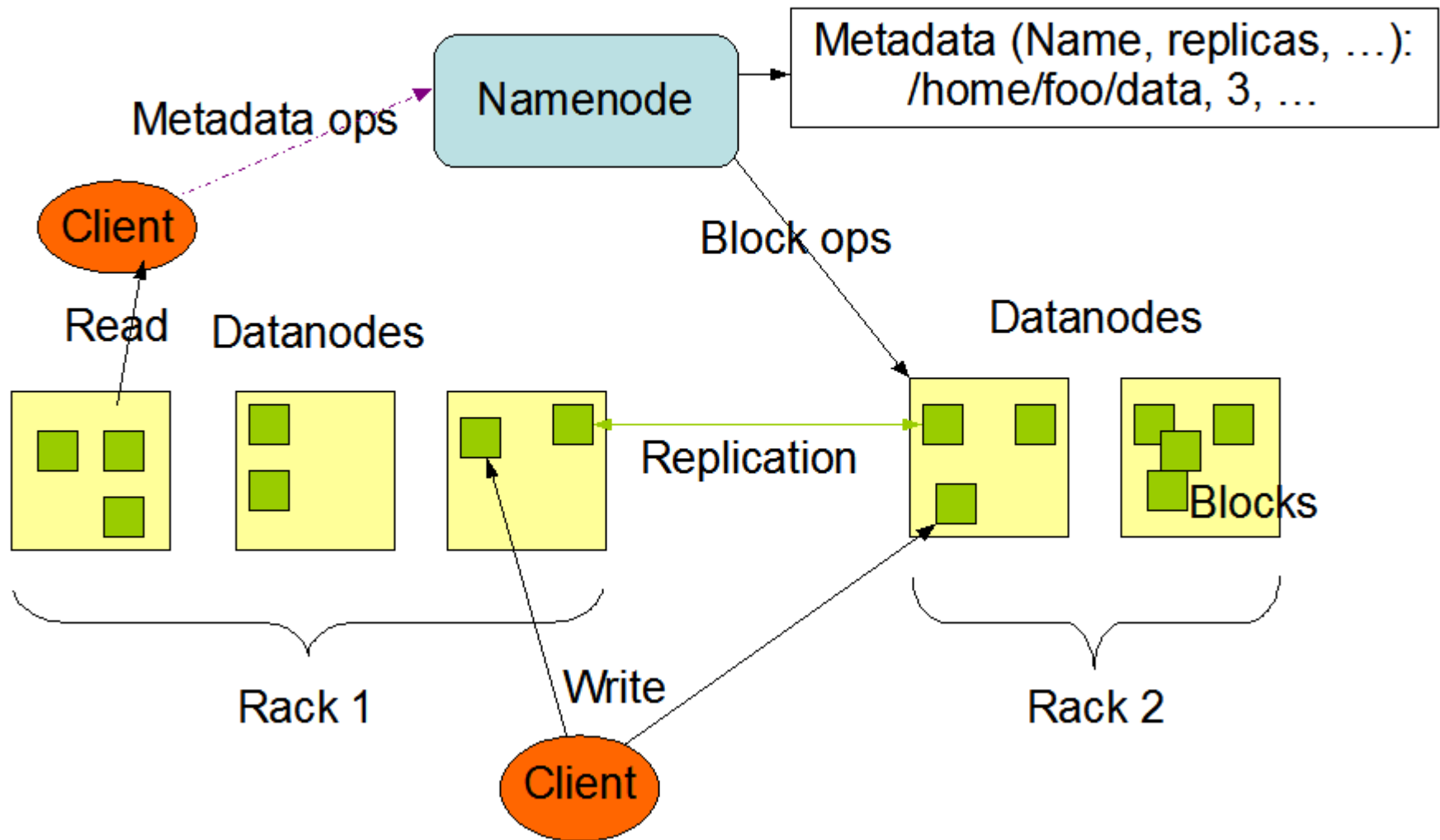
ЧТО ТАКОЕ АРХИТЕКТУРА?

- Архитектура программного обеспечения — это структура программы или вычислительной системы, которая включает программные компоненты, видимые снаружи свойства этих компонентов, а также отношения между ними. [Wikipedia]
- Архитектура - это базовая **организация системы**, воплощенная в ее **компонентах**, их **отношениях** между собой и с **окружением**, а также принципы, определяющие проектирование и развитие системы. [IEEE 1471]
- Архитектура - это **набор значимых решений** по поводу организации системы программного обеспечения, набор **структурных элементов** и их интерфейсов, при помощи которых конструируется система, вместе с их **поведением**, определяемым во взаимодействии между этими элементами, **компоновка** элементов в постепенно укрупняющиеся подсистемы, а также **стиль архитектуры** который направляет эту организацию -- элементы и их интерфейсы, взаимодействия и компоновку. [Крачтен (Kruchten)]
- Больше определений можно найти здесь:
<http://www.sei.cmu.edu/architecture/start/glossary/community.cfm>

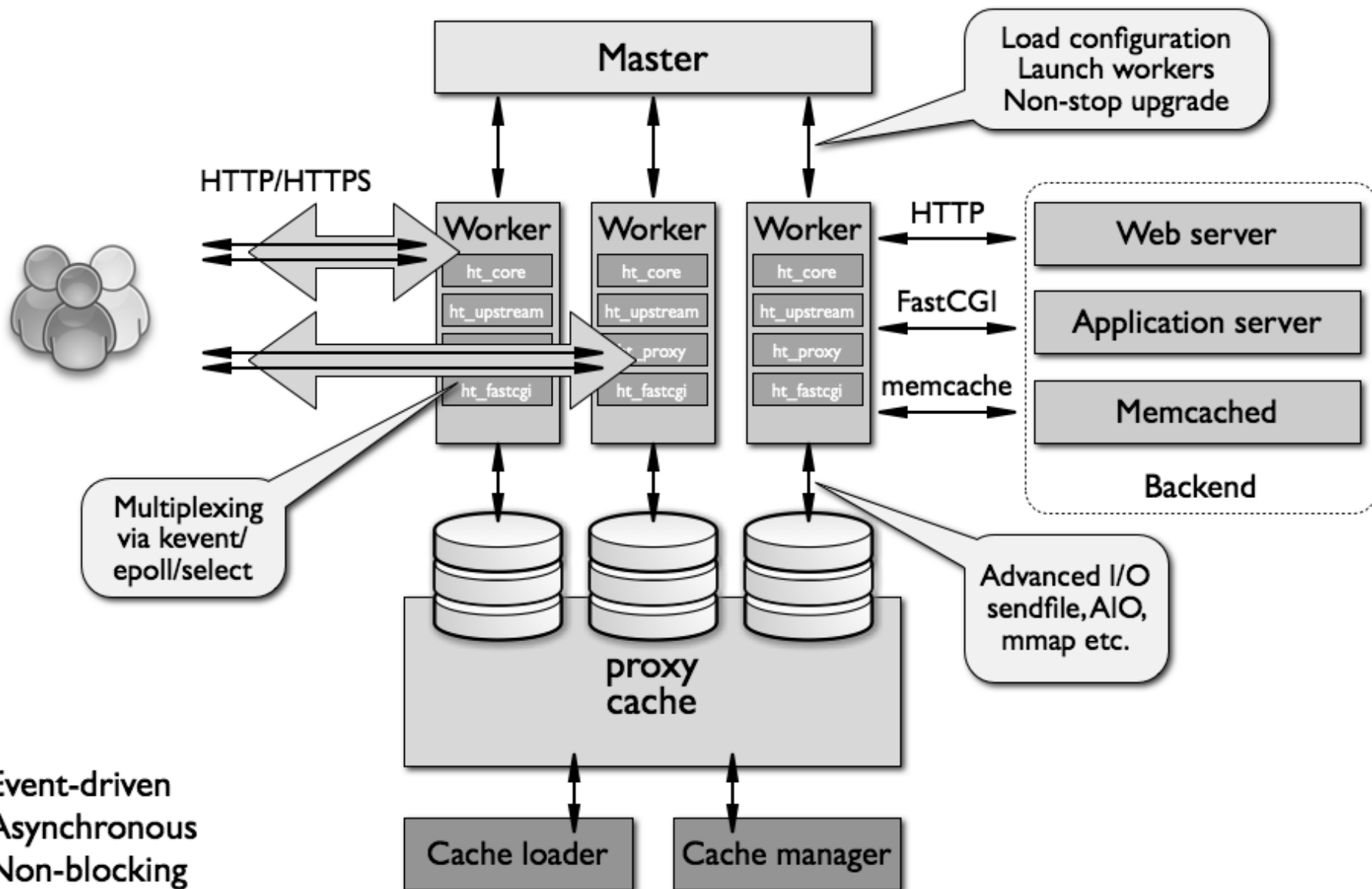
Примеры архитектур

HDFS

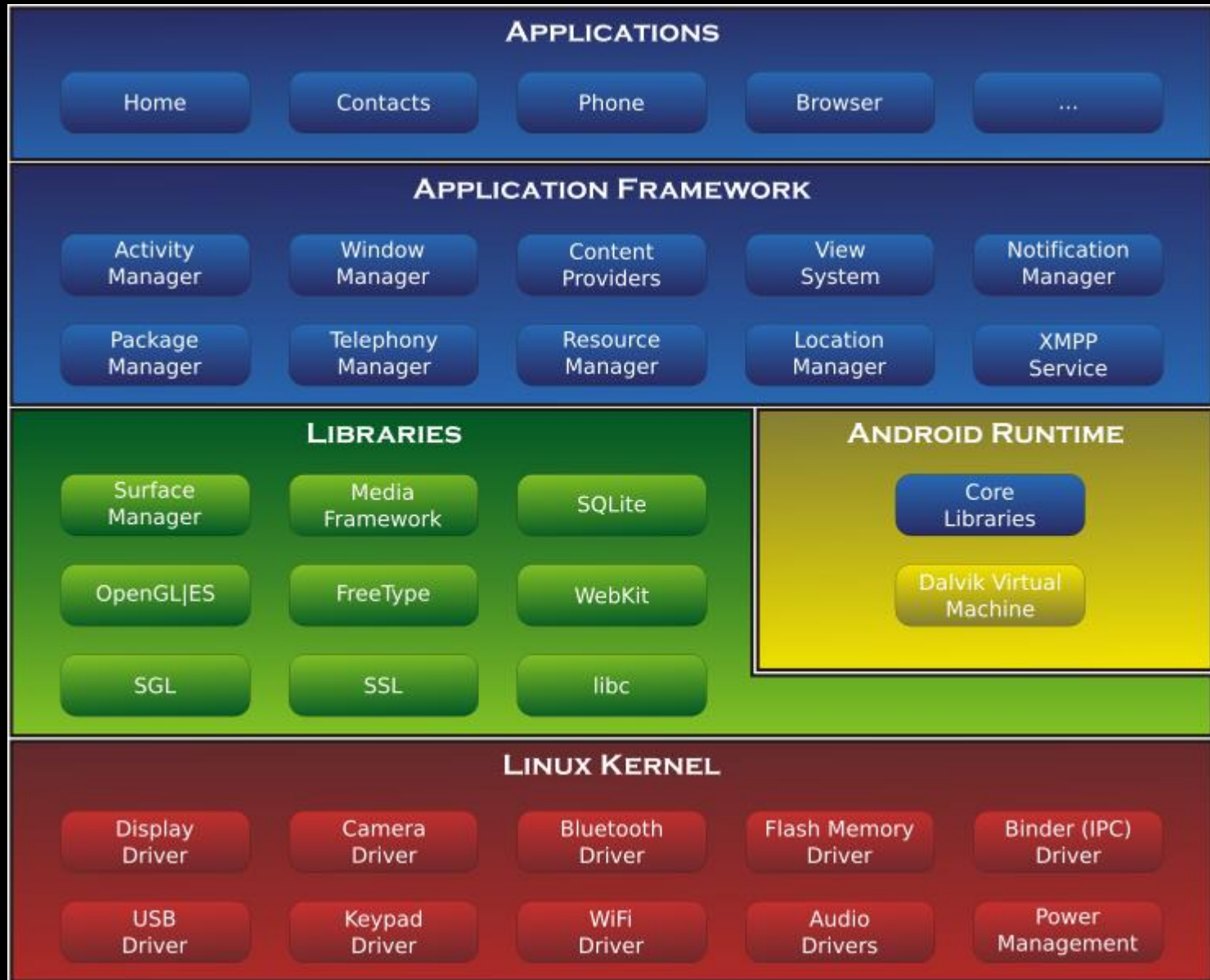
HDFS Architecture



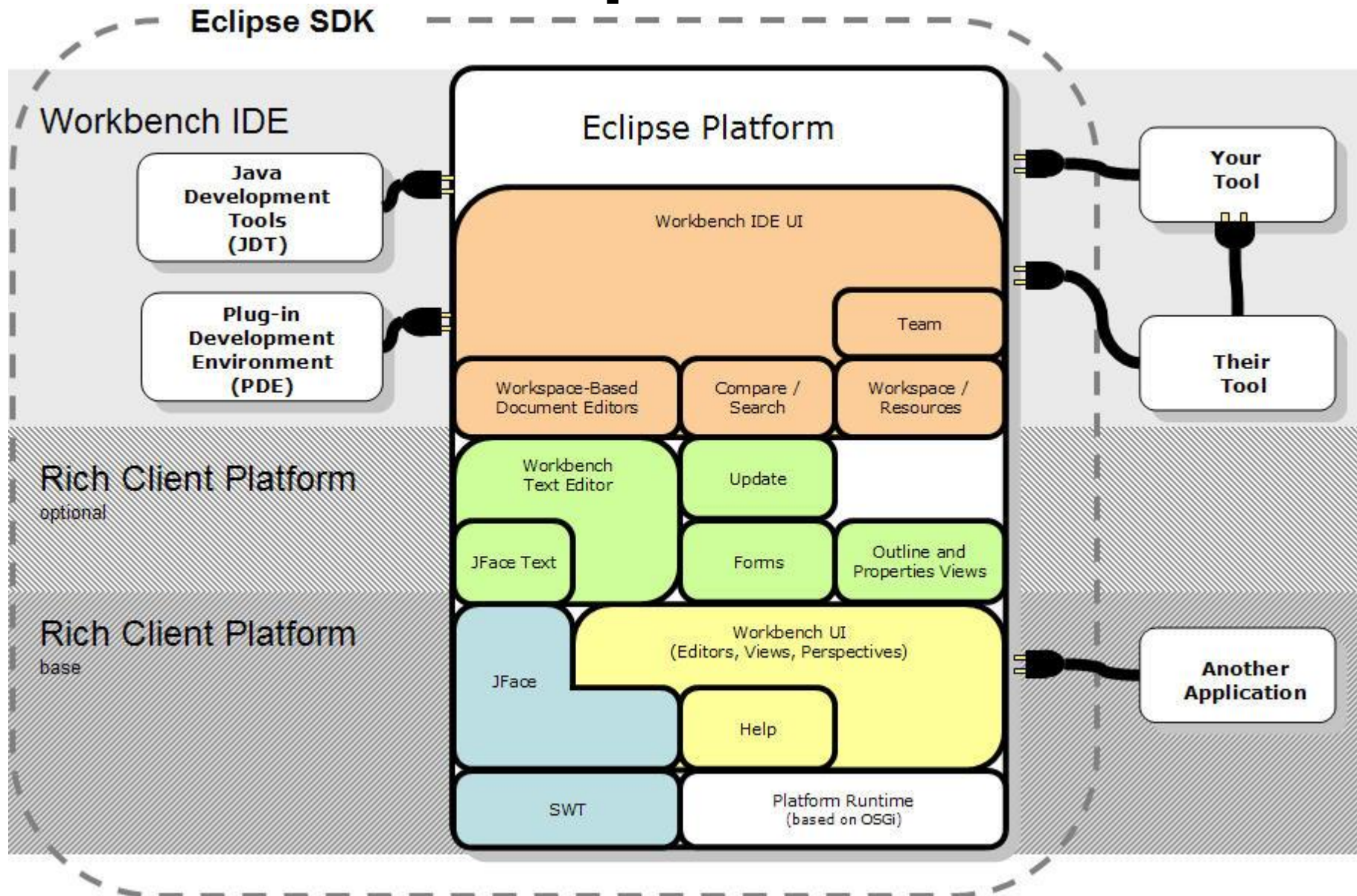
Nginx



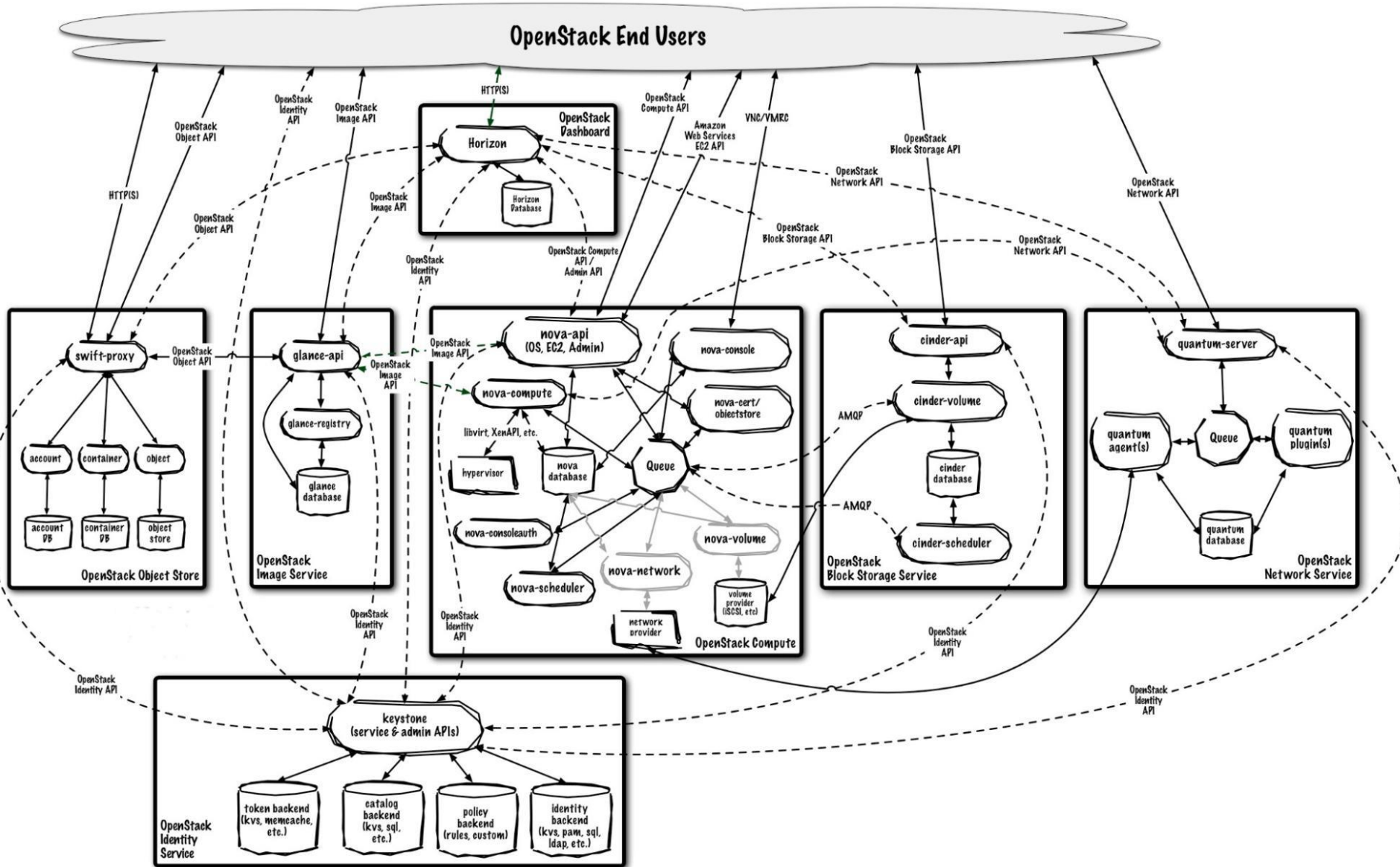
Android



Eclipse IDE



OpenStack



Документирование архитектуры

- DFD
- UML
- SysML
- В следующих лекциях...

ПОЧЕМУ АРХИТЕКТУРА ВАЖНА?

- Абстракция системы. Доступная для восприятия человека модель системы.
- Средство взаимодействия между заинтересованными лицами
- Способность системы реализовывать те или иные атрибуты качества обуславливается архитектурой
- Определяет организационную структуру
- Облегчает анализ изменений и их организацию
- Позволяет более точно рассчитывать стоимости и сроки

Заинтересованные лица

- **Конечный пользователь** заинтересован в интуитивно понятном и корректном поведении, производительности, надежности, удобстве использования, доступности и безопасности;
- **Системный администратор** заинтересован в интуитивно понятном поведении, управлении и инструментах мониторинга;
- **Специалист по маркетингу** заинтересован в конкурентноспособных функциях, времени до выхода программы, позиционировании среди других продуктов и в стоимости;
- **Клиент** заинтересован в цене, стабильности и возможности планировать;
- **Разработчик** заинтересован в понятных требованиях и простом и непротиворечивом принципе проектирования;
- **Руководитель проекта** заинтересован в предсказуемости хода проектирования, планировании, продуктивном использовании ресурсов и бюджета;
- **Специалист по сопровождению** заинтересован в понятном, непротиворечивом и документируемом принципе проекта, а также в легкости, с которой можно вносить изменения.

НЕФУНКЦИОНАЛЬНЫЕ АТТРИБУТЫ КАЧЕСТВА ПРОГРАММНОЙ СИСТЕМЫ

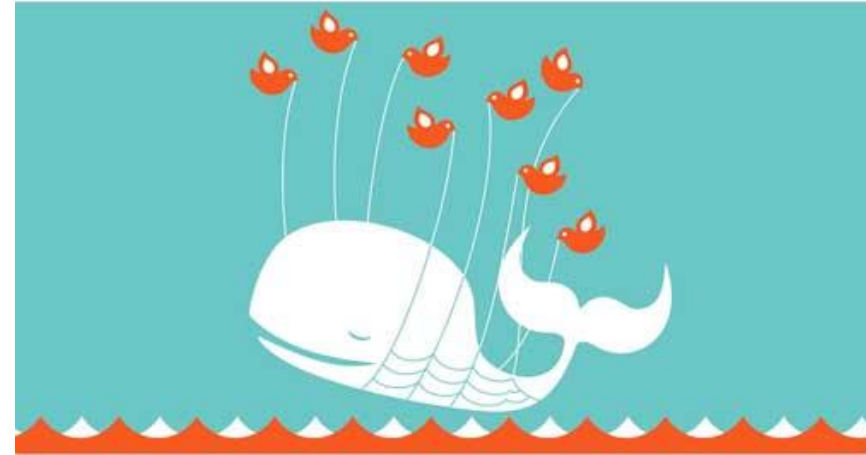
- Готовность (availability)
- Модифицируемость ()
- Производительность (performance)
- Безопасность (security)
- Контролепригодность
- Практичность
- Масштабируемость (scalability)
- Гибкость (flexibility)
- Расширяемость
- (extensibility)
- Сопровождаемость (maintainability)
- Интероперабельность или способность к взаимодействию (Interoperability)
- Надежность
- Модульность
- ...

больше примеров здесь https://en.wikipedia.org/wiki/List_of_system_quality_attributes

ГОТОВНОСТЬ, ДОСТУПНОСТЬ (AVAILABILITY)

- Отказ – положение, при котором система теряет способность предоставления услуг
- Готовность системы – вероятность функционирования, когда в этом есть необходимость

Twitter is over capacity.
Too many tweets! Please wait a moment and try again.



*Отказ (failure) Vs
неисправность (fault)*

$$A = \frac{Uptime}{Uptime + Downtime}$$

Uptime – среднее время до отказа,

Downtime – средняя продолжительность восстановления

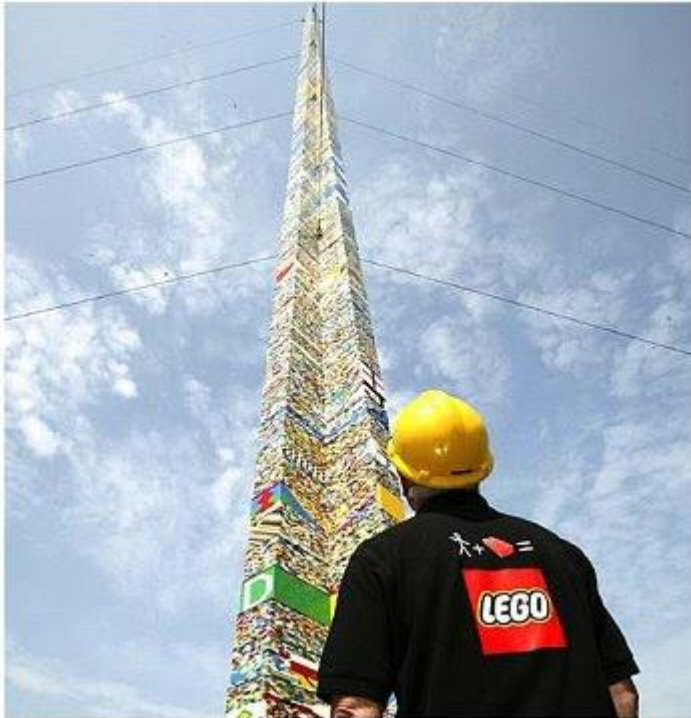


Модифицируемость

Отражает стоимость внесения изменений

Производительность

Модульность



- Модульность — свойство системы, которая может подвергаться декомпозиции на ряд внутренне связанных и слабо зависящих друг от друга модулей.

Масштабируемость

Расширяемость

ЖИЗНЕННЫЙ ЦИКЛ ПО

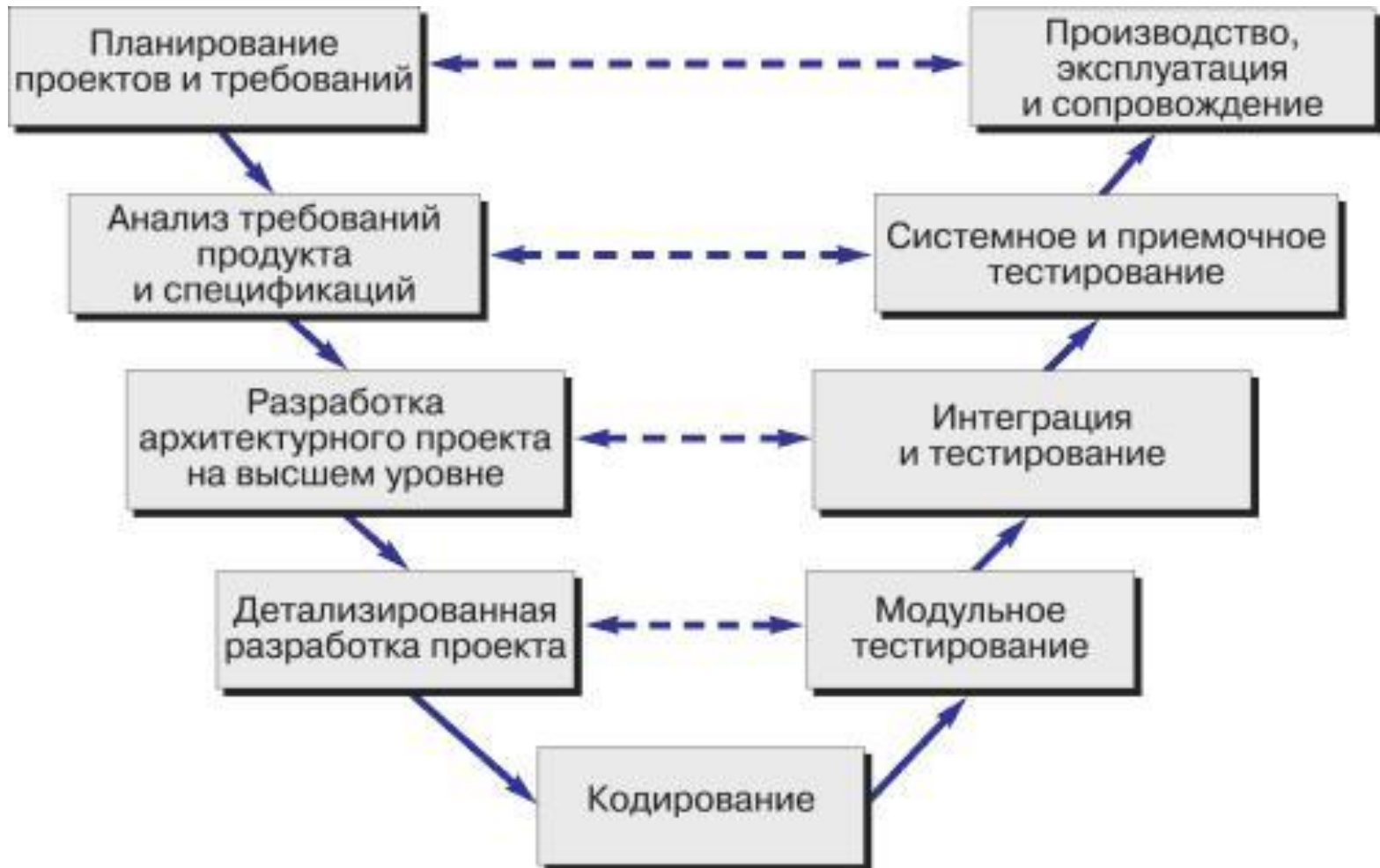
- **Жизненный цикл программного обеспечения (ПО)** — период времени, который начинается с момента принятия решения о необходимости создания программного продукта и заканчивается в момент его полного изъятия из эксплуатации.
- **Модель жизненного цикла ПО** — структура, определяющая последовательность выполнения и взаимосвязи процессов, действий и задач на протяжении жизненного цикла. Модель жизненного цикла зависит от специфики, масштаба и сложности проекта и специфики условий, в которых система создается и функционирует.

Водопадная модель (каскадная модель)

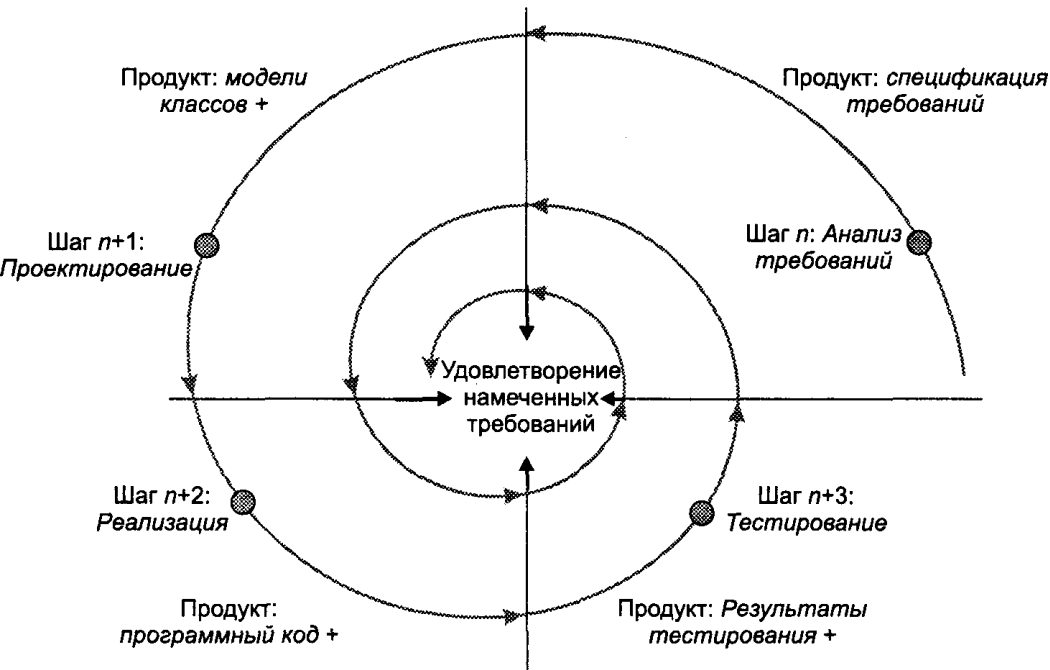


предложена в 1970 г. Уинстоном Ройсом

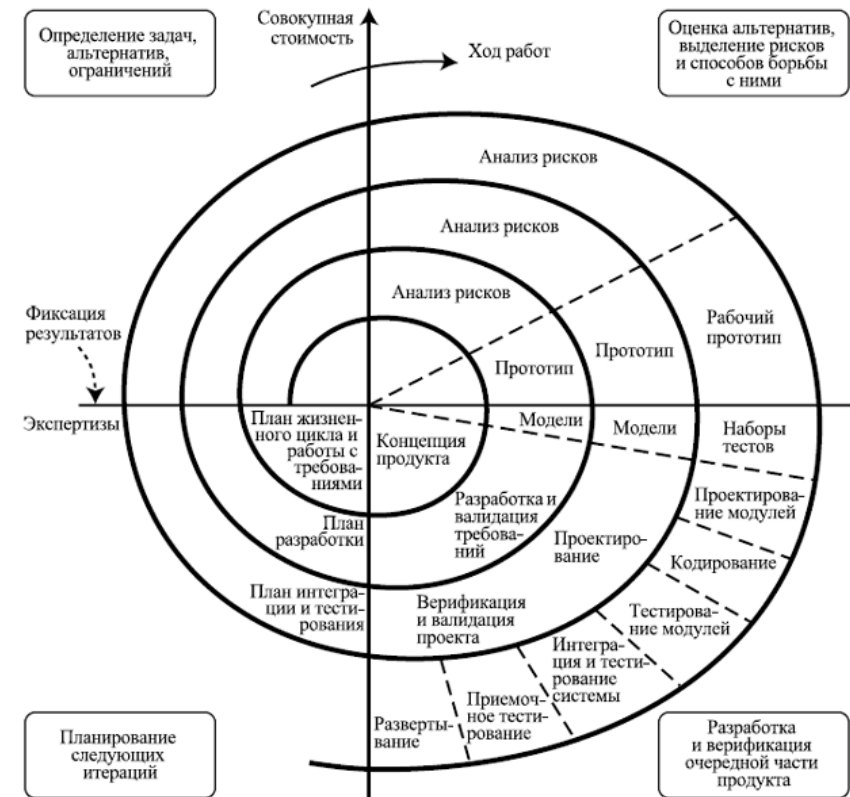
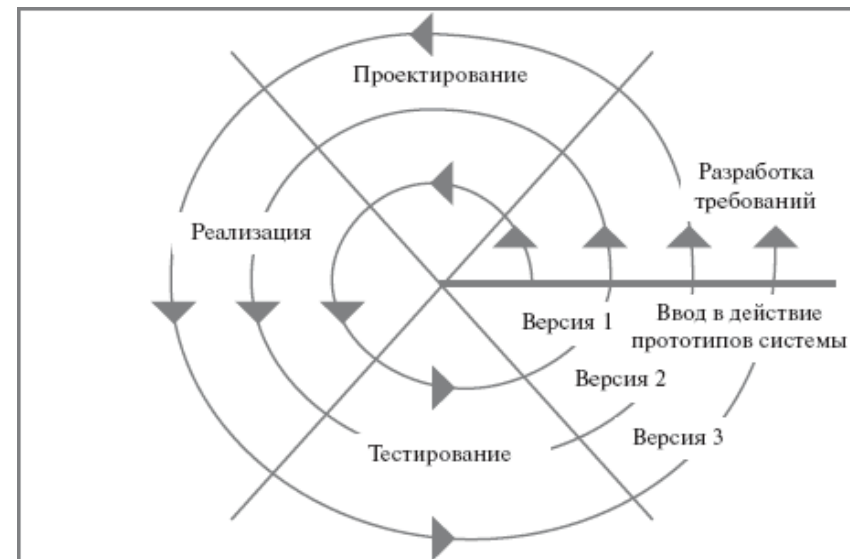
V-модель













СПИРАЛЬНАЯ МОДЕЛЬ



- Разработана в середине 1980-х годов Барри Боэмом
- Основана на цикле Деминга **PDCA** ([англ.](#) «**Plan-Do-Check-Act**» - планирование-действие-проверка-корректировка)



Инкрементальная

Номер итерации	1	2	3		867	868
Тестирование единого целого				Обновление плана управления программным проектом		
Интеграция				Обновление документации по тестированию		
Компонентное тестирование				Обновление исходного программного кода		
Реализация				Обновление проектной документации программного обеспечения		
Проектирование						
Анализ требований				Обновление спецификаций требований к программному обеспечению		

ИЗДЕРЖКИ ПРОЦЕССОВ РАЗРАБОТКИ

Фактор	Чистый водопад	Итеративные процессы	
		Спираль	Инкрементальный
Легкость контроля документации	Легче	Тяжелее	Тяжелее
Возможность взаимодействия с заказчиком	Тяжелее	Легче	Легче
Поддержание хорошего проектирования	Средне	Легче	Тяжелее
Сбор метрических данных, собранных в ходе проекта	Тяжелее	Средне	Средне

Гибкие методологии разработки

Гибкая методология разработки (англ. *Agile software development, agile-методы*) – серия подходов к разработке программного обеспечения, ориентированных на использование итеративной разработки, динамическое формирование требований и обеспечение их реализации в результате постоянного взаимодействия внутри самоорганизующихся рабочих групп, состоящих из специалистов различного профиля

Источники

- **Книги**

- Э. Дж. Брауде. Технология разработки программного обеспечения. Питер, 2004.
- Э. Гамма, Р. Хелм, Р. Джонсон, Дж. Влиссидес. Приемы объектно-ориентированного проектирования. Паттерны проектирования. Питер-ДМК, 2001.
- Басс Л., Клементс П., Кацман Р. Архитектура программного обеспечения на практике. – М. и др. : Питер, 2006.

- **Полезные курсы**

- Курс Дениса Иванова и Федора Новикова «UML для разработчиков»
http://uml3.ru/library/uml_for_developers/uml_for_developers.html
- Курс Андрея Бреслава «Software Design»
<https://sites.google.com/site/abreslav2/softwaredesign2>
- Курс «Software Architecture for Developers»
<http://www.codingthearchitecture.com/presentations/software-architecture-for-developers>

- **Примеры архитектур**

- Серия книг «The Architecture of Open Source Applications»
<http://aosabook.org/en/index.html>