Технологии проектирования ИС и ИТ

ФИО преподавателя: Смирнов Михаил Вячеславович

e-mail: smirnov.mirea@gmail.com

Лекция 5

Паттерны архитектуры ПО. Инструментарий управления проектами ИС и ИТ

Содержание

- Паттерны архитектуры ПО ИС и ИТ
- Типовой состав группы разработки ПО
- Инструмент управления проектами “Диаграмма Gantt”
- Инструмент управления проектами “Таблица RACI”
- SCRUM и Kanban
- Иерархическая структура проекта WBS

Архитектурные шаблоны (паттерны) приложений

- Многоуровневый
- Клиент-серверный
- Ведущий-ведомый
- Каналы и фильтры
- Шаблон посредника
- Одноранговый
- Шина событий
- Модель-представление-контроллер
- Доска
- Интерпретатор

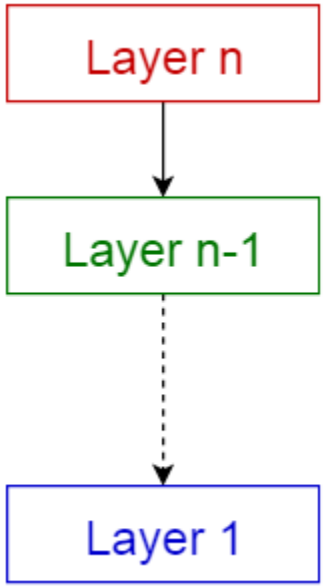
Многоуровневый шаблон архитектуры

Используется для структурирования программ, которые можно разложить на группы неких подзадач, находящихся на определенных уровнях абстракции. Каждый слой предоставляет службы для следующего, более высокого слоя.

- Слой представления (также известен как слой пользовательского интерфейса)
- Слой приложения (также известен как слой сервиса)
- Слой бизнес-логики (также известен как уровень предметной области)
- Слой доступа к данным (также известен как уровень хранения данных)

Многоуровневый шаблон архитектуры

Примеры применения: десктопные приложения, e-commerce



Клиент-серверный шаблон архитектуры

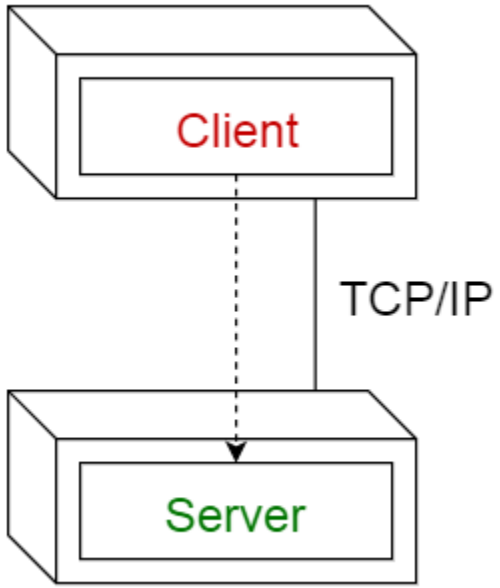
Состоит из двух частей: сервера и множества клиентов.

Серверный компонент предоставляет службы клиентским компонентам.

Клиенты запрашивают услуги у сервера, а он, в свою очередь, оказывает эти самые услуги клиентам.

Используется, как правило, в онлайн-приложениях (электронная почта, сервисы банковских услуг и т.д.)

Клиент-серверный шаблон архитектуры



Шаблон архитектуры Ведущий-ведомый

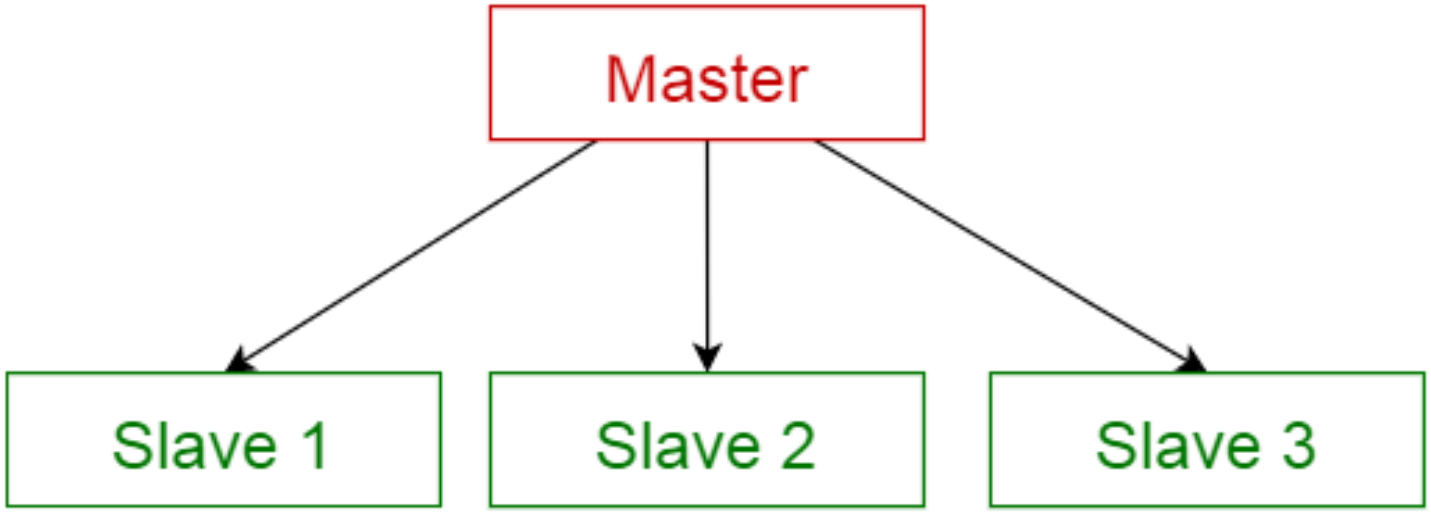
Задействованы два участника — ведущий и ведомые. Ведущий компонент распределяет задачи среди идентичных ведомых компонентов и вычисляет итоговый результат на основании результатов, полученных от своих «подчиненных».

Примеры использования:

Репликация баз данных. Там главная БД считается авторитетным источником, а подчиненные базы с ней синхронизируются.

Периферийные устройства, подключенные к шине в компьютере (ведущие и ведомые устройства).

Шаблон архитектуры Ведущий-ведомый



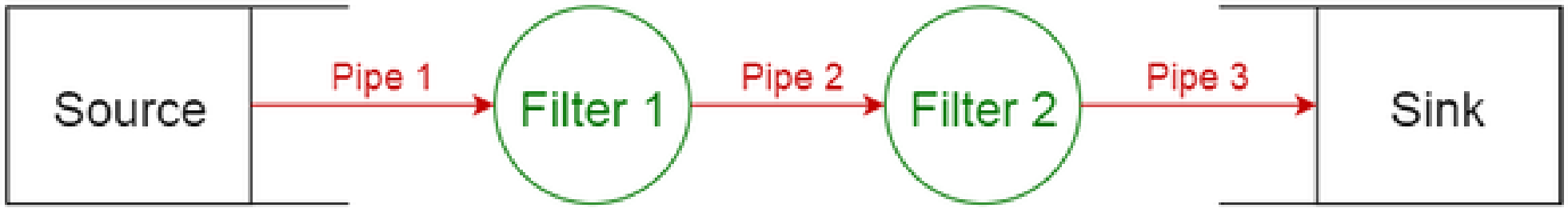
Шаблон архитектуры Каналы и фильтры

Этот шаблон подходит для систем, которые производят и обрабатывают потоки данных. Каждый этап обработки происходит внутри некоего компонента фильтра. Данные для обработки передаются через каналы. Эти каналы можно использовать для буферизации или синхронизации данных.

Архитектура каналов и фильтров применяется в самых разных приложениях, особенно при решении задач, обеспечивающих простую одностороннюю обработку — например, инструменты EDI (электронный обмен данными), ETL (извлечение, преобразование и загрузка).

Пример - компиляторы: последовательно расположенные фильтры выполняют лексический, синтаксический, семантический анализ и создание кода.

Шаблон архитектуры Каналы и фильтры



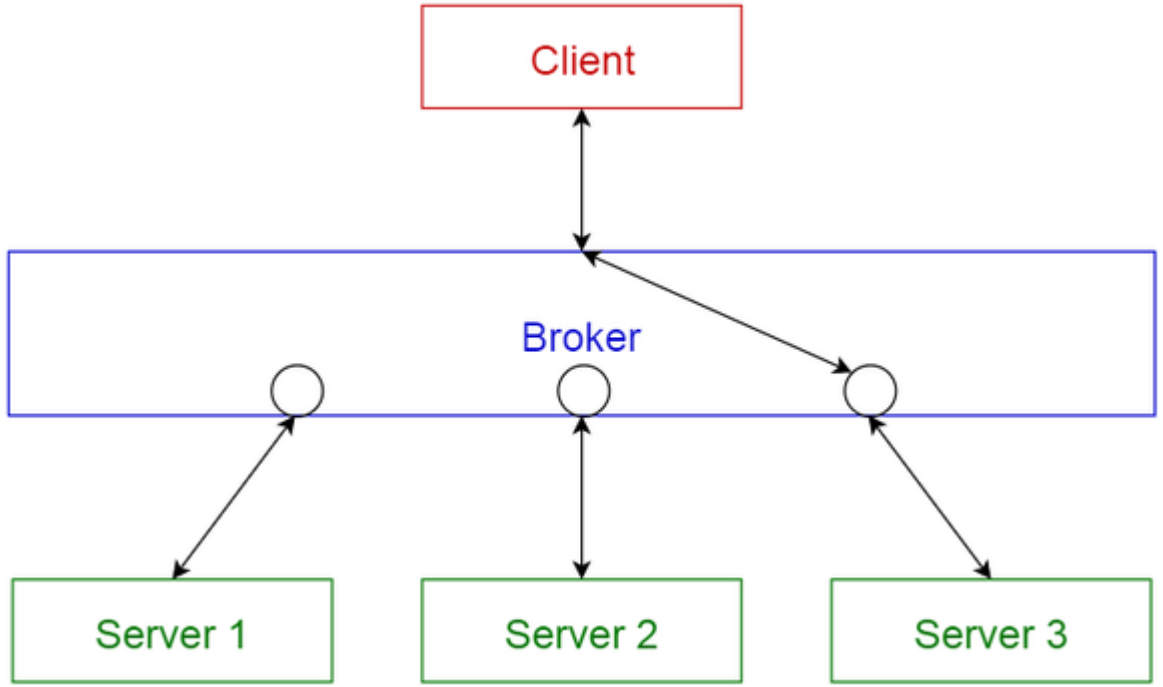
Шаблон архитектуры Посредник

Используется для структуризации распределенных систем с несвязными компонентами. Эти компоненты могут взаимодействовать друг с другом через удаленный вызов службы. Компонент посредник отвечает за координацию взаимодействия компонентов.

Сервер размещает свои возможности (службы и характеристики) у посредника (брокера). Клиент запрашивает услугу у брокера. Затем брокер перенаправляет клиента к подходящей службе из своего реестра.

Шаблон архитектуры Посредник

Используется брокерами сообщений: Apache ActiveMQ, Apache Kafka, RabbitMQ и т.д.

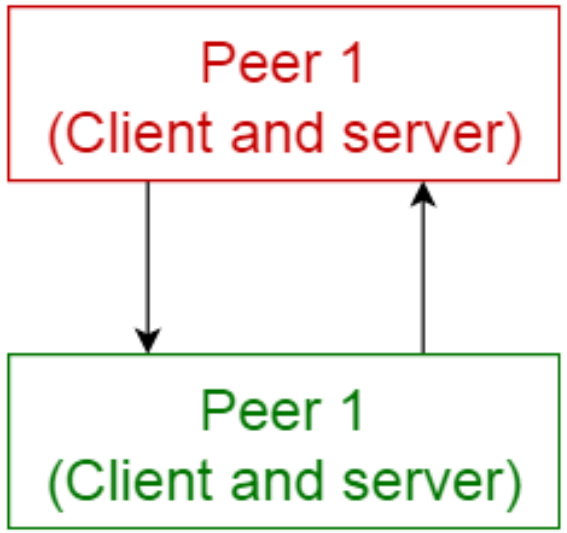


Одноранговый шаблон архитектуры (пиринг)

Существуют отдельные компоненты, так называемые ***пиры***. Пиры могут выступать в роли как клиента, запрашивающего услуги от других равноправных участников (пиров), так и сервера, предоставляющего услуги другим пирам. Пир может быть клиентом или сервером, или всем сразу, а также способен со временем динамически изменять свою роль.

Использование: мультимедийные протоколы (P2PTV и PDTP), проприетарные мультимедийные приложения (Spotify).

Одноранговый шаблон архитектуры (пиринг)



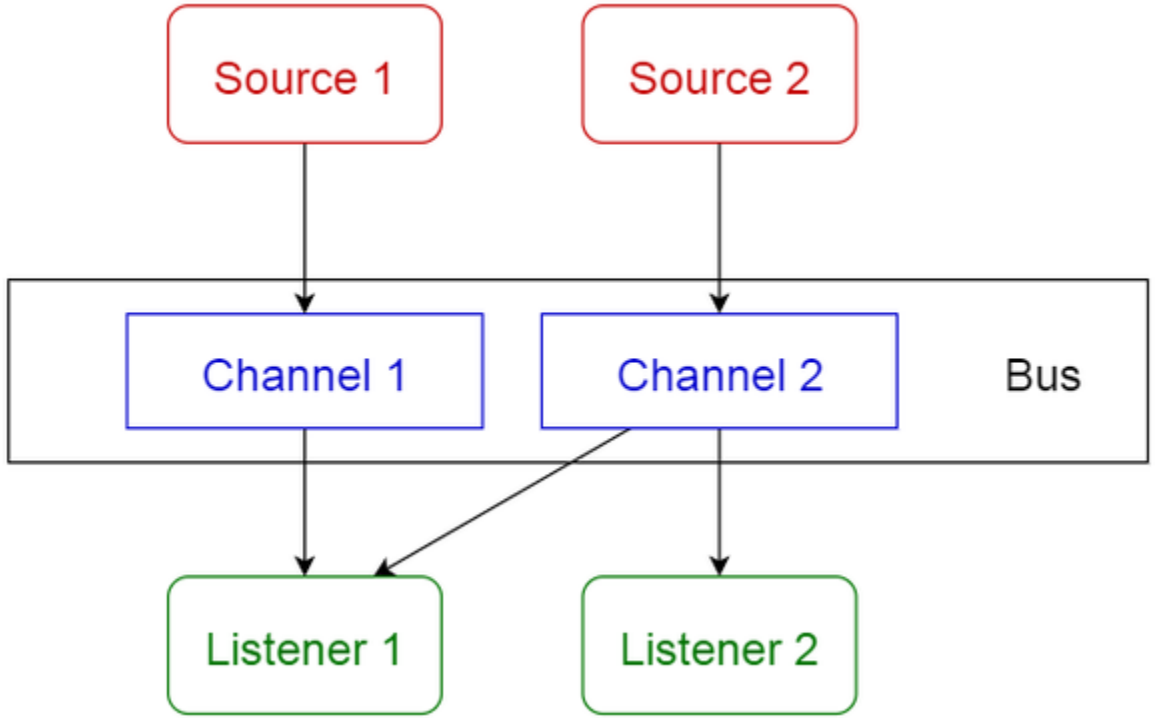
Шаблон архитектуры Шина событий

Взаимодействует с событиями и состоит из 4 главных компонентов: ***источник события***, ***прослушиватель события***, ***канал*** и ***шина событий***.

Источники размещают сообщения для определенных каналов на шине событий. Прослушиватели подписываются на определенные каналы. Прослушиватели получают уведомления о появлении сообщений, размещенных на каналах из их подписки.

Применяется, например, в сервисах уведомлений.

Шаблон архитектуры Шина событий



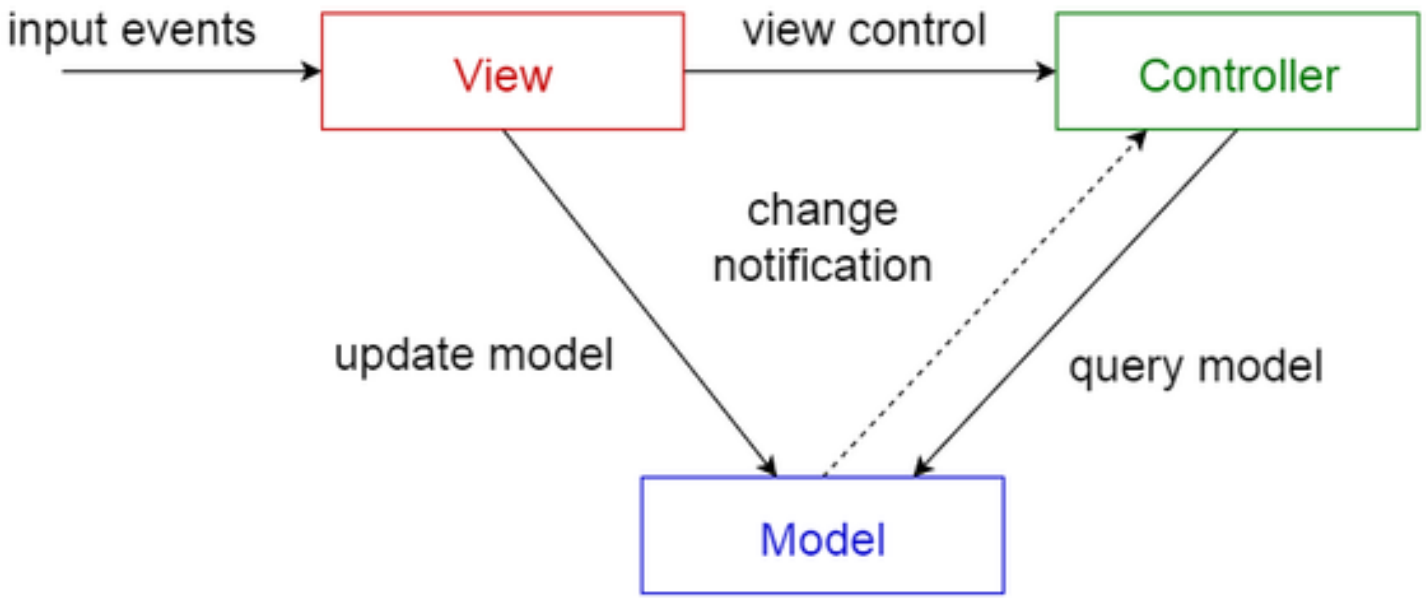
Шаблон архитектуры Модель-представление-контроллер (MVC-шаблон)

Разделяет интерактивные прикладные программы на 3 части:

1. модель - содержит ключевые данные и функционал;
2. представление - показывает информацию пользователю (можно задавать более одного представления);
3. контроллер - занимается обработкой данных от пользователя.

Использование: веб-приложения, веб-фреймворки (Django, Rails..)

Шаблон архитектуры Модель-представление-контроллер (MVC-шаблон)



Шаблон архитектуры Доска

Используется для решения проблем, для которых отсутствуют четкие детерминированные решения. Шаблон «Доска» состоит из 3 главных компонентов:

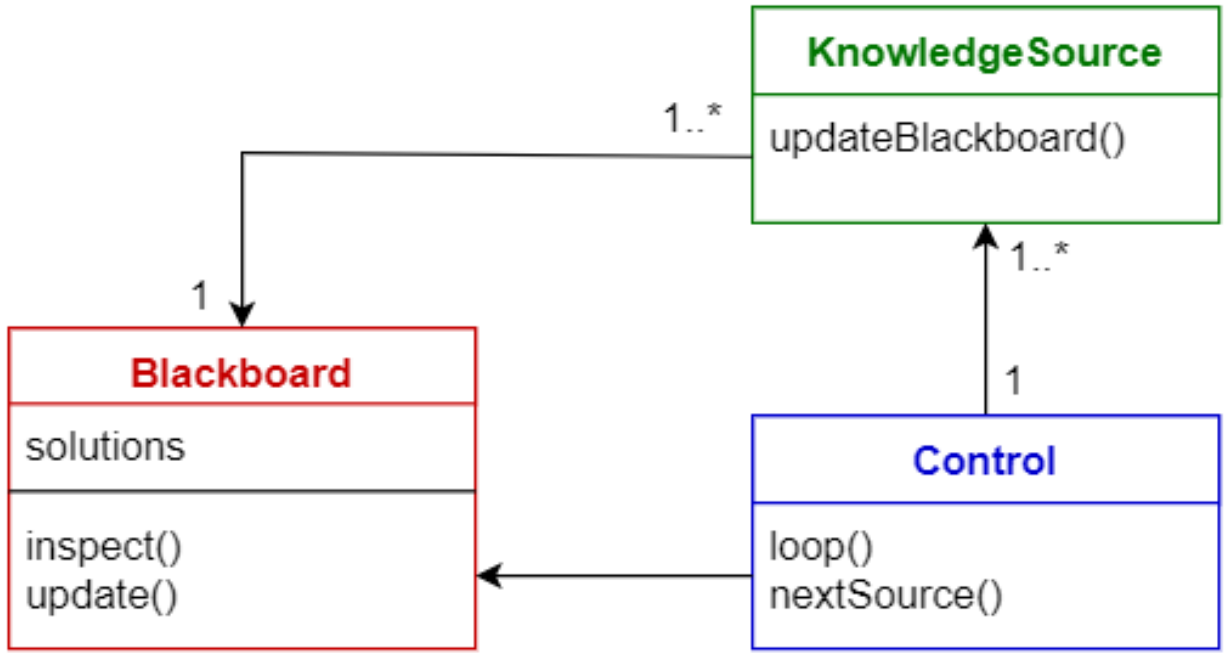
1. ***Доска (Blackboard)*** - это структурированная глобальная память, содержащая объекты из пространства возможных решений;

2. ***Источник знания (Knowledge Source)*** - специализированные модули со своим собственным представлением;

3. ***Компоненты управления (Control)*** - выбирает, настраивает и исполняет модули.

Используется при распознавании образов, речи и т.д.

Шаблон архитектуры Доска



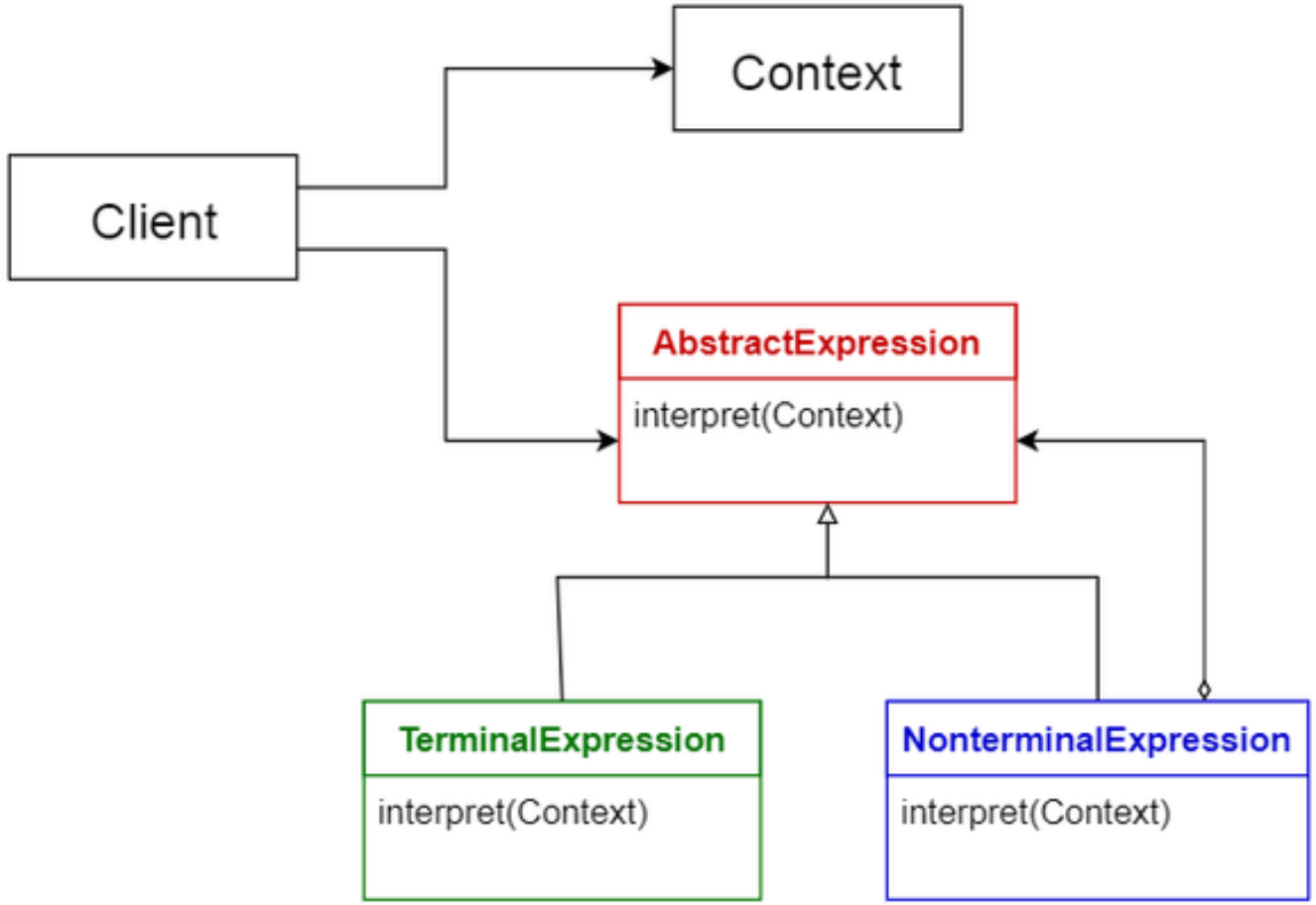
Шаблон Интерпретатор

Применяется для разработки компонента (утилиты), который должен интерпретировать программы, написанные на специальном языке программирования.

В основном, в функционале такого приложения изложено, как вычислять строки, написанные на каком-то определенном языке программирования.

Применение: языки запросов к БД, описание протоколов передачи данных

Шаблон Интерпретатор



Типовой состав группы проекта разработки ИС

- Аналитик – развитие и интерпретация требований заказчика
- Архитектор – проектирование и развитие архитектуры продукта
- Конструктор компонентов – создатель компонентов программного средства
- Специалист по применению – программирование компонентов программного средства
- Специалист по повторному использованию – внедрение уже готовых компонентов из библиотек
- Специалист по интеграции – сборка совместимых версий компонентов и тестирование их совместимости
- Специалист по документации – документация реализованных решений
- Системный программист – создание утилит, облегчающих процесс разработки
- Системный администратор – управление компьютерными ресурсами в проекта

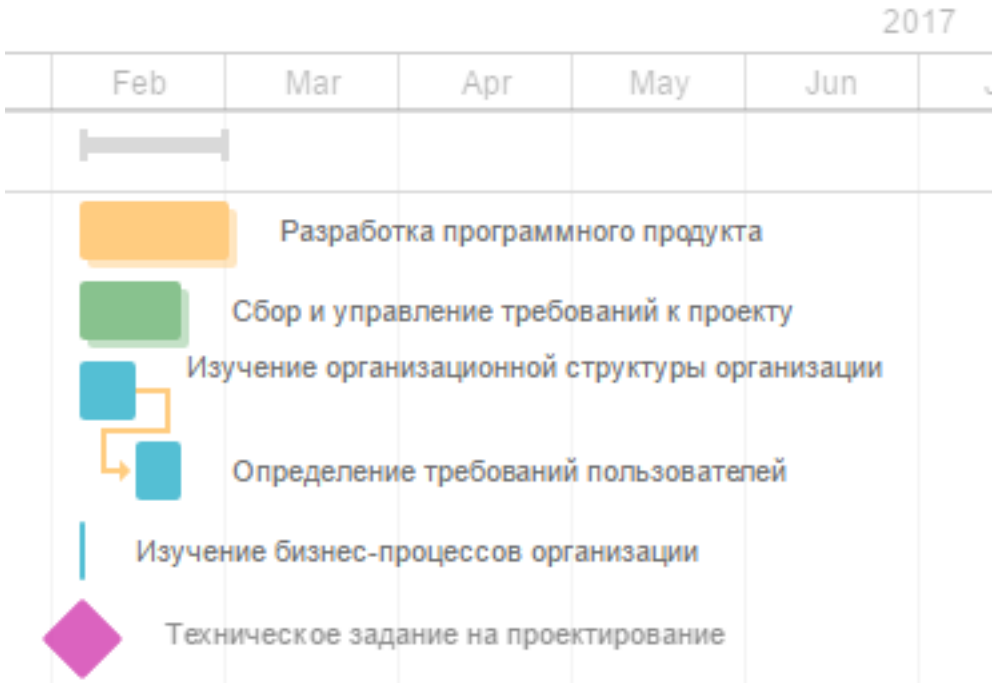
Инструмент управления проектами “Диаграмма Gantt”

- Популярный тип столбчатых диаграмм (гистограмм), который используется для иллюстрации плана, графика работ по какому-либо проекту. Является одним из методов планирования проектов. Используется в приложениях по управлению проектами.

- работа с параметром ее длительности

- веха, как результат работы или группы работ

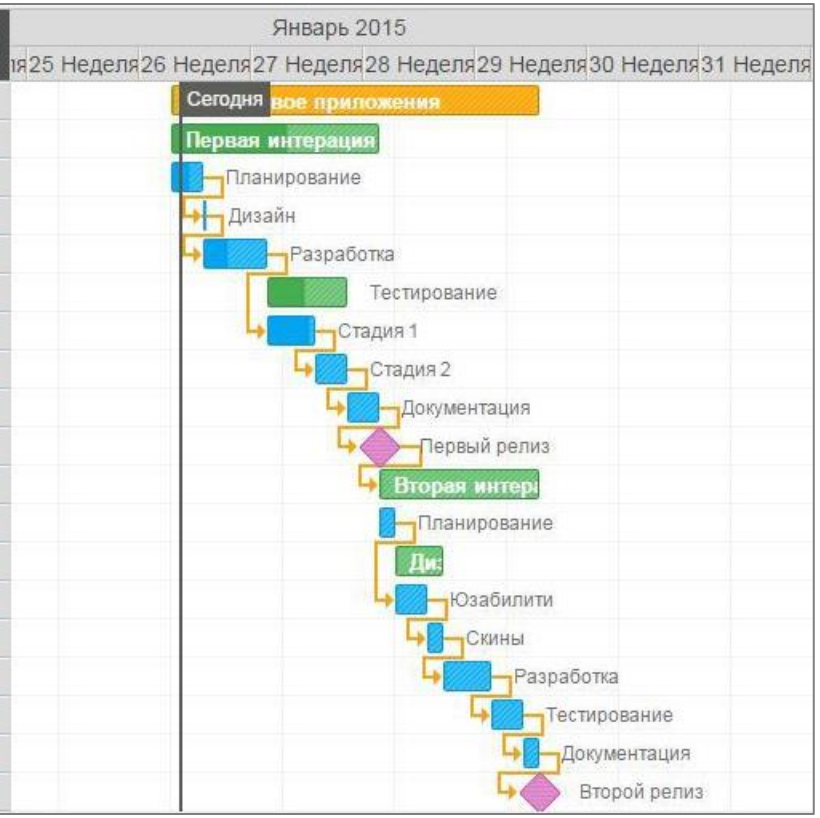
Инструмент управления проектами “Диаграмма Gantt”. Последовательные работы, пример.



Фрагмент, оформлен в веб-приложении Gantt-PRO: https://app.ganttpro.com*

Инструмент управления проектами “Диаграмма Gantt”. Итеративные работы, пример.

Задача	Начало	Продол. дн.	Назначе...	Действия
Мое первое приложения	2015-06-24	23.04		
Первая интерация	2015-06-24	13.04		
Планирование	2015-06-24	2.00	Менеджеры	
Дизайн	2015-06-26	0.04	Дизайнеры	
Разработка	2015-06-26	4.00	Разработчики	
Тестирование	2015-06-30	5.00		
Стадия 1	2015-06-30	3.00	QA	
Стадия 2	2015-07-03	2.00	QA	
Документация	2015-07-05	2.00	Тех. писател	
Первый релиз	2015-07-07	0.00		
Вторая интерация	2015-07-07	10.00		
Планирование	2015-07-07	1.00	Менеджеры	
Дизайн	2015-07-08	3.00		
Юзабилити	2015-07-08	2.00	UI-гуру	
Скины	2015-07-10	1.00	Дизайнеры	
Разработка	2015-07-11	3.00	Разработчики	
Тестирование	2015-07-14	2.00	QA	
Документация	2015-07-16	1.00	Тех. писател	
Второй релиз	2015-07-17	0.00		



Методика CPM

Метод критического пути - инструмент планирования расписания и управления сроками проекта.

В основе метода лежит определение наиболее длительной последовательности задач от начала проекта до его окончания с учетом их взаимосвязи.

Задачи, лежащие на критическом пути (критические задачи), имеют нулевой резерв времени выполнения, и, в случае изменения их длительности, изменяются сроки всего проекта. В связи с этим, при выполнении проекта критические задачи требуют более тщательного контроля, в частности, своевременного выявления проблем и рисков, влияющих на сроки их выполнения и, следовательно, на сроки выполнения проекта в целом. В процессе выполнения проекта критический путь проекта может меняться, так как при изменении длительности задач некоторые из них могут оказаться на критическом пути.

Ключевая задача метода – определить критический путь, и установить на нем максимальный контроль выполнения задач

Этапы построения диаграммы Гантта с CPM

Выделить индивидуальные задачи проекта. Следует начать со списка всех задач проекта. Он может быть использован как базис для создания последовательностей и оценки времени выполнения проекта на следующих этапах.

Определите последовательность выполнения выделенных задач. Выполнение одних задач напрямую связано с окончанием других.

Изобразить диаграмму в виде графической сети взаимосвязанных задач.

Оценить время каждой конкретной активности. Время, требуемое для завершения каждой задачи, может быть оценено на основе прошлого опыта или оценок экспертов.

Обозначить критический путь (наиболее длинная цепочка действий в рамках сети). Важность заключается в том, что задачи, располагающиеся на этом пути, не могут быть отложены либо просрочены без переноса дедлайнов проекта.

Мониторинг обновления диаграммы по ходу реализации проекта

Параметры критического пути CPM

Ближайшее время старта (БС) — момент, когда все предыдущие задачи завершены.

Ближайшее время окончания (БО) — Ближайшее время старта плюс время, необходимое для завершения задачи.

Последнее время окончания (ПО) — Финальный момент, когда все активности завершены без переносов сроков задач.

Последнее время начала (ПН) — Последнее время окончания минус время, требуемое для завершения задачи.

Резервное время — это время между ближайшим и последним временем старта, или между ближайшим и последним временем окончания проекта. Резервное время — это то время, на которое может быть отложены ближайшее время старта и финиша без движения дедлайнов всего проекта.

Критический путь — это цепочка через всю сеть задач проекта, в которой ни одна из задач не имеет резервного времени, когда БС=ПН и БО=ПО для всех задач на пути.

Отличие PERT от CPM

Бывают случаи, когда точно определить длительность задачи сложно или невозможно (инновационный проект или проект с рисками).

Для этих случаев лучше использовать PERT метод:

- он вероятностный (4 измерения времени)
- он учитывает риски

Этапы построения диаграммы Гантта с CPM

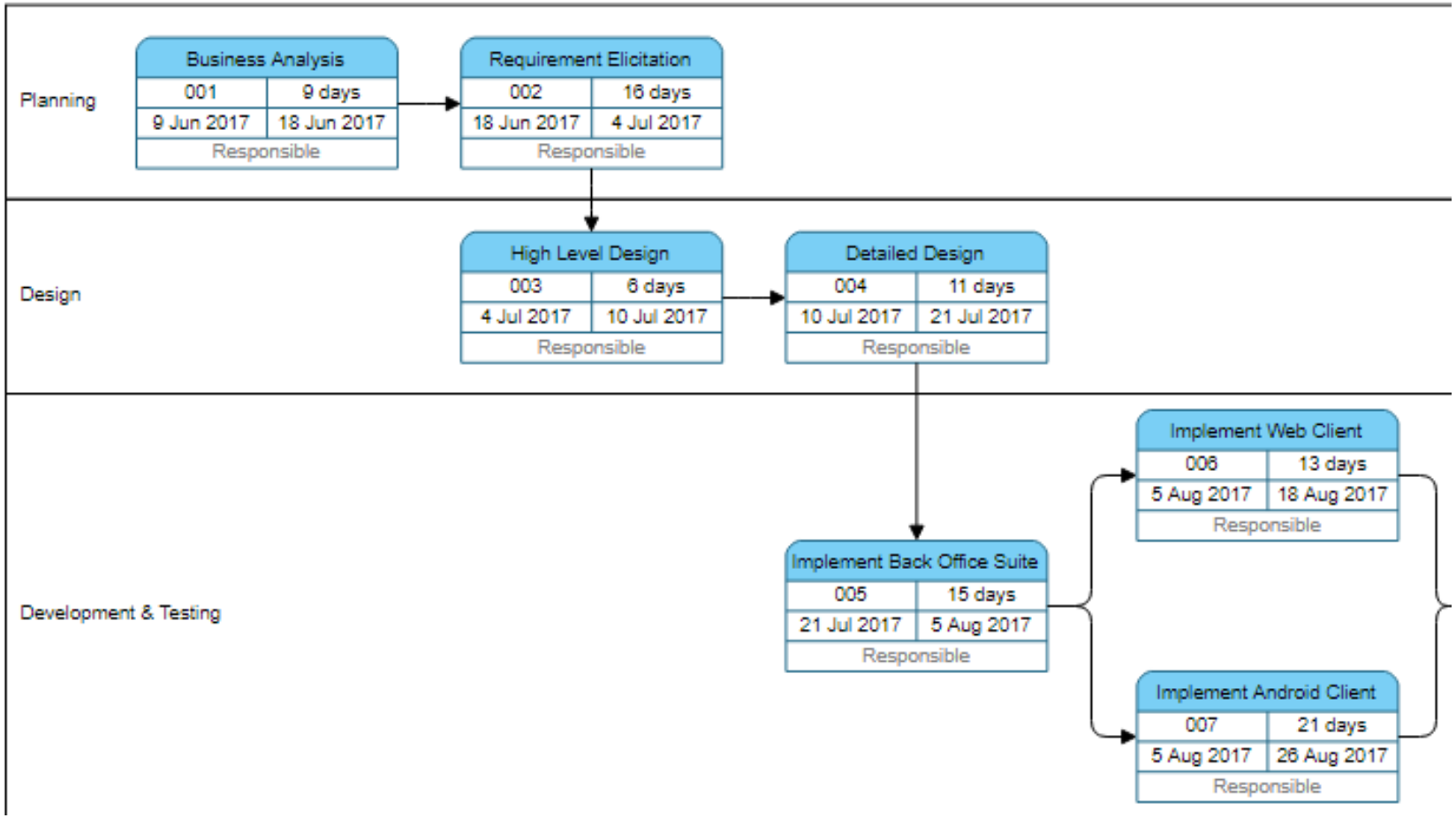
Оптимистическое время (optimistic time) (O): минимальное возможное время для выполнения задачи, в предположении что все происходит лучше чем ожидается.

Пессимистическое время (pessimistic time) (P): максимально возможное время требующееся для выполнения задачи, в предположении что все происходит неправильно (исключая крупные катастрофы).

Наиболее вероятное время (most likely time) (M): оценка времени, требующегося для выполнения задачи, в предположении, что все происходит как обычно.

Ожидаемое время (expected time) (TE): лучшая оценка времени, требуемого для выполнения задачи, учитывая, что вещи не всегда происходят как обычно. (Ожидаемое среднее время выполнения задачи, если она будет повторяться многократно).

Образец графика PERT для разработки ПО



Методика RACI

RACI matrix (RAM Responsibility Assignment Matrix) – график линейной ответственности. Показывает степень участия разных ролей в разных активностях процесса или проекта.

Выделяют множество альтернативных вариаций, из которых наиболее применимы RACI-VS и RASCI.

Матрица RACI

Матрица представляет собой таблицу, столбцами которой являются роли, а строками – деятельности процесса или проекта.

На пересечении столбца и строки устанавливается одна из следующих вариативных опций:

R – Responsible (ответственный исполнитель). На исполнителе лежит обязанность реализации поставленной задачи;

A – Accountable (утверждающий). Перед ним производится отчет в полученном результате, имеются полномочия, как принимать, так и отвергать предложения, накладывать на них вето;

C – Consult before doing (консультант). До реализации задачи консультирует участников проекта. Данная роль подразумевает двустороннюю связь между задействованными в проекте подразделениями;

I – Inform after doing (информируемый). Данное лицо оповещается после реализации проекта или отдельных его задач.

Вариации RACI

В вариации **RACI-VS** добавляются еще две функции: ***Verifies*** (**V**) – один сотрудник или группа, проверяющие соответствие результата выполнения задачи согласованным заранее допустимым критериям, ***Signs off*** (**S**) – утверждает сдачу продукта заказчику (выполнения задачи). Данную роль можно совместить с подотчетной ролью.

В вариации **RASCI** добавляется роль ***Supportive*** (**S**) – предоставляет дополнительные ресурсы для выполнения работы, такой как поддержка внедрения продукта, к примеру.

Пример RACI диаграммы (raci-chart.org)

	Facilities	Plant Mgr	HR	Security	Project Mgr
Identify a minimum of three asphalt contractors from Angie's List	C	-	-	-	R
Arrange for contractor visits and quotes	I	-	-	-	R
Review quotes and references, make contractor selection	A	I	I	-	R
Review and finalize contract, lock in plant shutdown week	I	I	-	-	R
Communicate project to shutdown maintenance crew, make sure all vehicles are removed from the lot	I	I	R	I	I
Provide security gate access codes for asphalt crew by June 15	I	-	A	R	I
Oversee the project during the plant shutdown week, ensure it is completed on time	A	I	I	-	R

R = Responsible, A = Accountable, C = Consulted, I = Informed

Основные правила матрицы RACI

Accountable – должен быть только один. Если это не так, то нужно четко ограничить рамки, в которых, либо в данный момент по данной деятельности, либо в данных условиях ответственный только один, но в других условиях по той же деятельности возможно ответственность несет другой.

Responsible – должен быть в наличии по каждой деятельности, их может быть несколько, причем возможны совмещения.

Каждая деятельность обязательно должна иметь Accountable и Responsible.

Инструмент управления проектами Agile-доска

Гибкая методология создания проекта программного средства (например, ИС или ИТ).

Привязана к основным положениям Agile методов разработки.

Представлена в двух видах: Kanban и Scrum.

Инструмент управления проектами Agile-доска

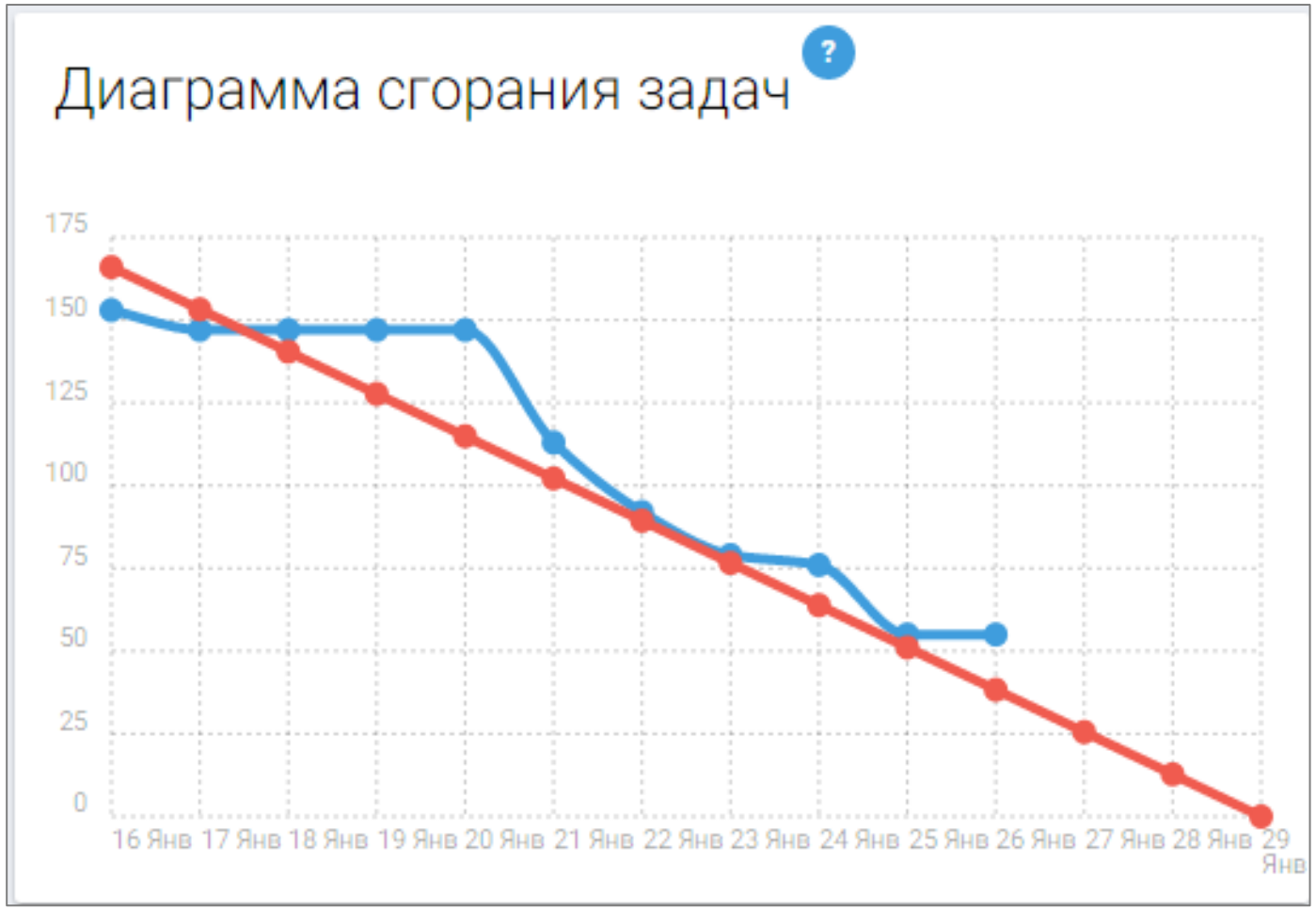
Бэклог - журнал оставшейся работы, которую необходимо выполнить команде.

Журнал проекта (Project backlog) - это список требований к функциональности, упорядоченный по их степени важности, подлежащих реализации. Элементы этого списка называются *пользовательскими историями* (user story). Журнал пожеланий проекта открыт для редактирования для всех участников scrum-процесса.

Спринт - итерация в SCRUM, в ходе которой создаётся функциональный рост программного обеспечения. Жёстко фиксирован по времени. Длительность одного спринта от 2 до 4 недель.

Журнал спринта (Sprint backlog) - содержит функциональность, выбранную владельцем проекта из журнала проекта. Все функции разбиты по задачам, каждая из которых оценивается scrum-командой. Каждый день команда оценивает объём работы, который нужно проделать для завершения спринта.

Диаграмма сгорания задач

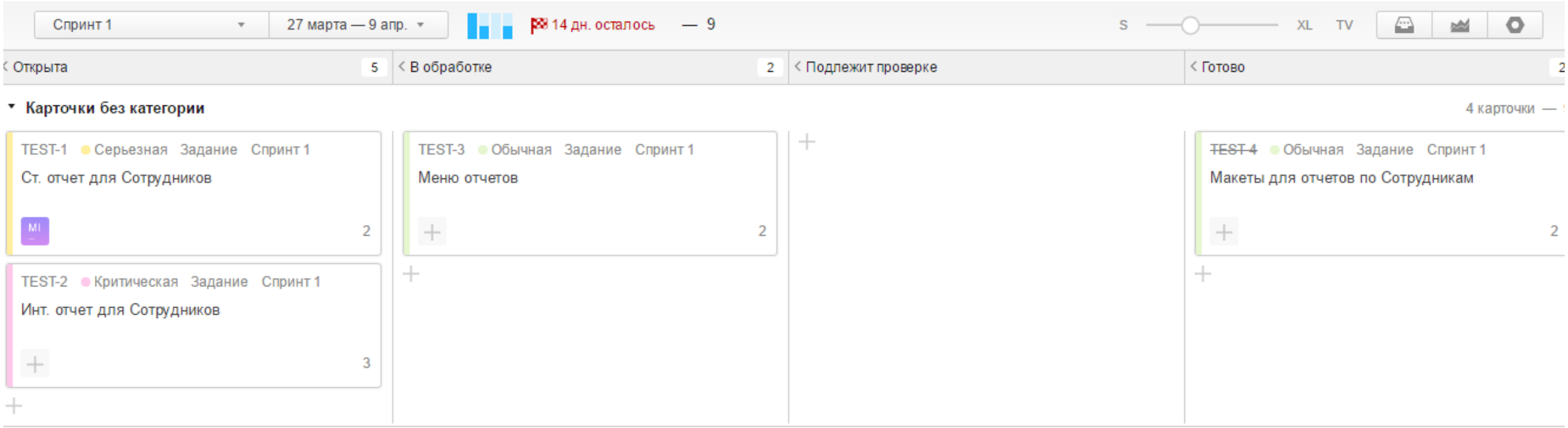


Основные определения SCRUM-элементов

Очки за пользовательскую историю (Story Points). Абстрактная метрика оценки сложности истории, которая не учитывает затраты в человеко-часах. Обычно используют одну из следующих шкал: ряд Фибоначчи (1,2,3,5,8,13,21,34,55); линейную шкалу (1,2,3,4 … n); степень двойки (1,2,4,8 … 2n); размеры одежды (XS, S, M, L, XL).

Скорость команды (Velocity). Общее количество очков, набранных командой за предыдущий спринт. Данная метрика помогает команде понять, сколько историй она может сделать за один спринт.

Доска SCRUM (образец)



Макет с примером доски SCRUM создан при помощи ПО YouTrack:
https://msuniversity.myjetbrains.com/youtrack/dashboard?id=6dd18ae5-545c-43c0-b3fe-d945347b9fa9

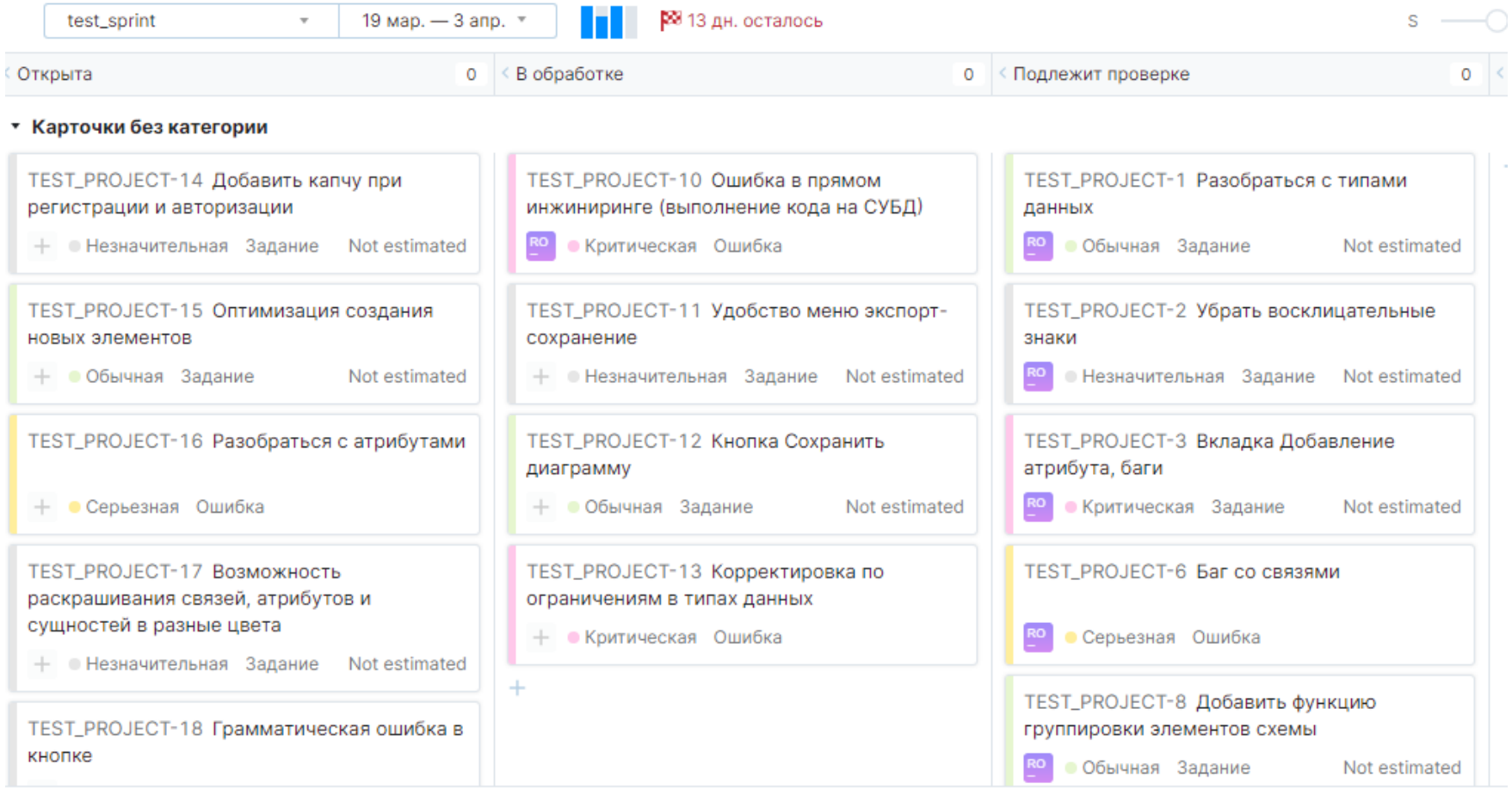
Доска SCRUM. Dashboard.

My Dashboard

#issues sort by: updated [19]

- O TEST_PROJECT-19 Настройки шрифтов и цвета
- H TEST_PROJECT-18 Грамматическая ошибка в кнопке
- H TEST_PROJECT-17 Возможность раскрашивания связей, атрибутов и сущностей в разные цвета
- C TEST_PROJECT-16 Разобраться с атрибутами
- O TEST_PROJECT-15 Оптимизация создания новых элементов
- H TEST_PROJECT-14 Добавить капчу при регистрации и авторизации
- K TEST_PROJECT-13 Корректировка по ограничениям в типах данных
- O TEST_PROJECT-12 Кнопка Сохранить диаграмму
- H TEST_PROJECT-11 Удобство меню экспорт-сохранение
- K TEST_PROJECT-10 Ошибка в прямом инжиниринге (выполнение кода на СУБД)
- K TEST_PROJECT-9 Баги сущностей
- O TEST_PROJECT-8 Добавить функцию группировки элементов схемы
- C TEST_PROJECT-6 Баг со связями
- K TEST_PROJECT-3 Вкладка Добавление атрибута, баги

Доска SCRUM. Sprint. User Story.



Доска Kanban. Специфика применения.

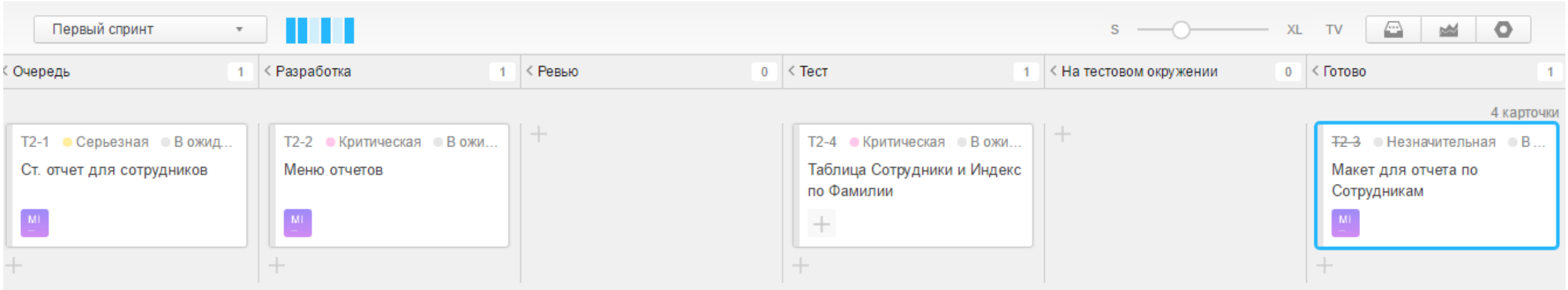
группы поддержки программного обеспечения, где не важен «план», но важна скорость реагирования на изменения;

группы тестирования, работающие отдельно от групп разработки;

службы поддержки;

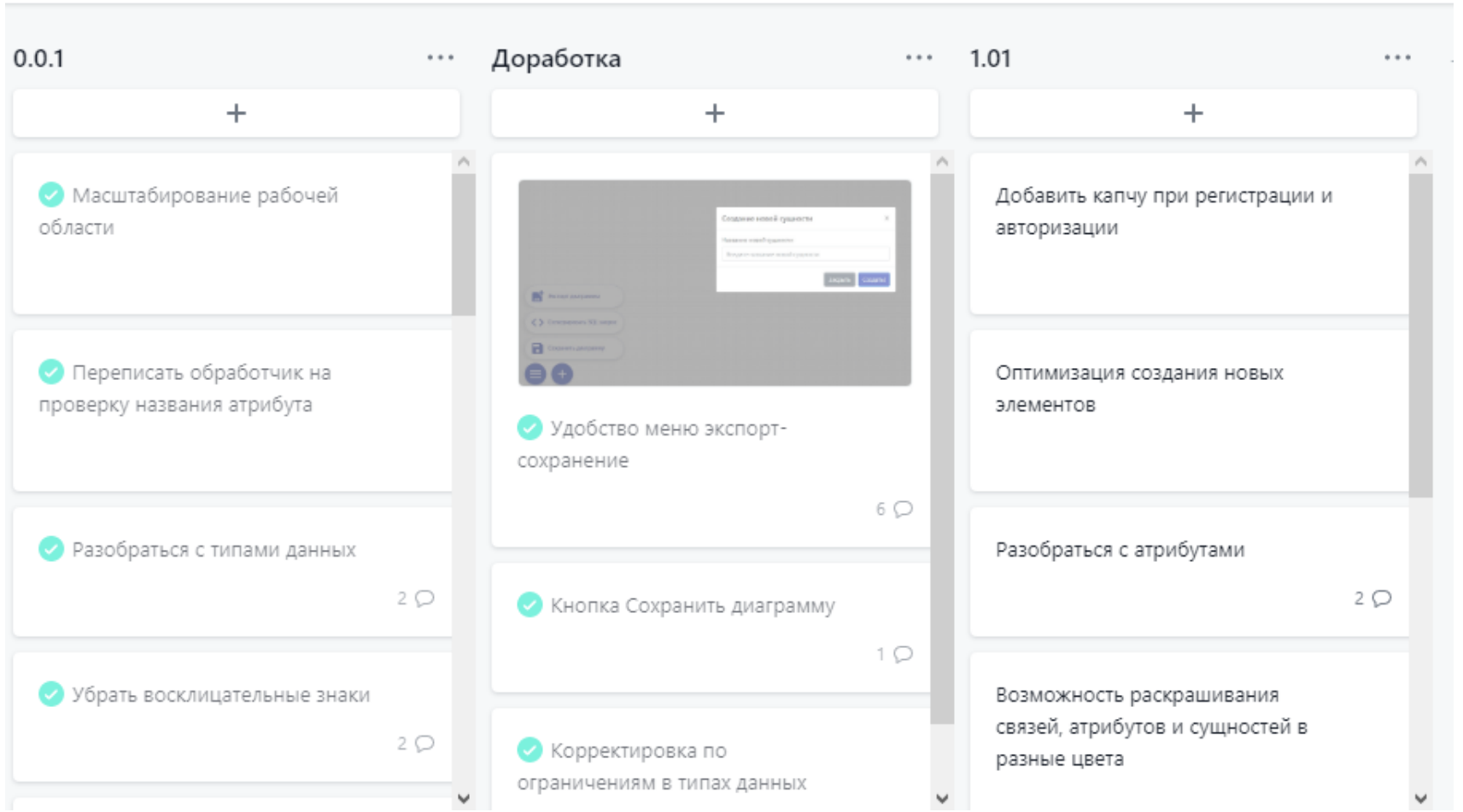
стартапы без четкого плана результата проекта.

Доска Kanban (образец)



Макет с примером доски kanban создан при помощи ПО YouTrack:
https://msuniversity.myjetbrains.com/youtrack/dashboard?id=6dd18ae5-545c-43c0-b3fe-d945347b9fa9

Доска Kanban

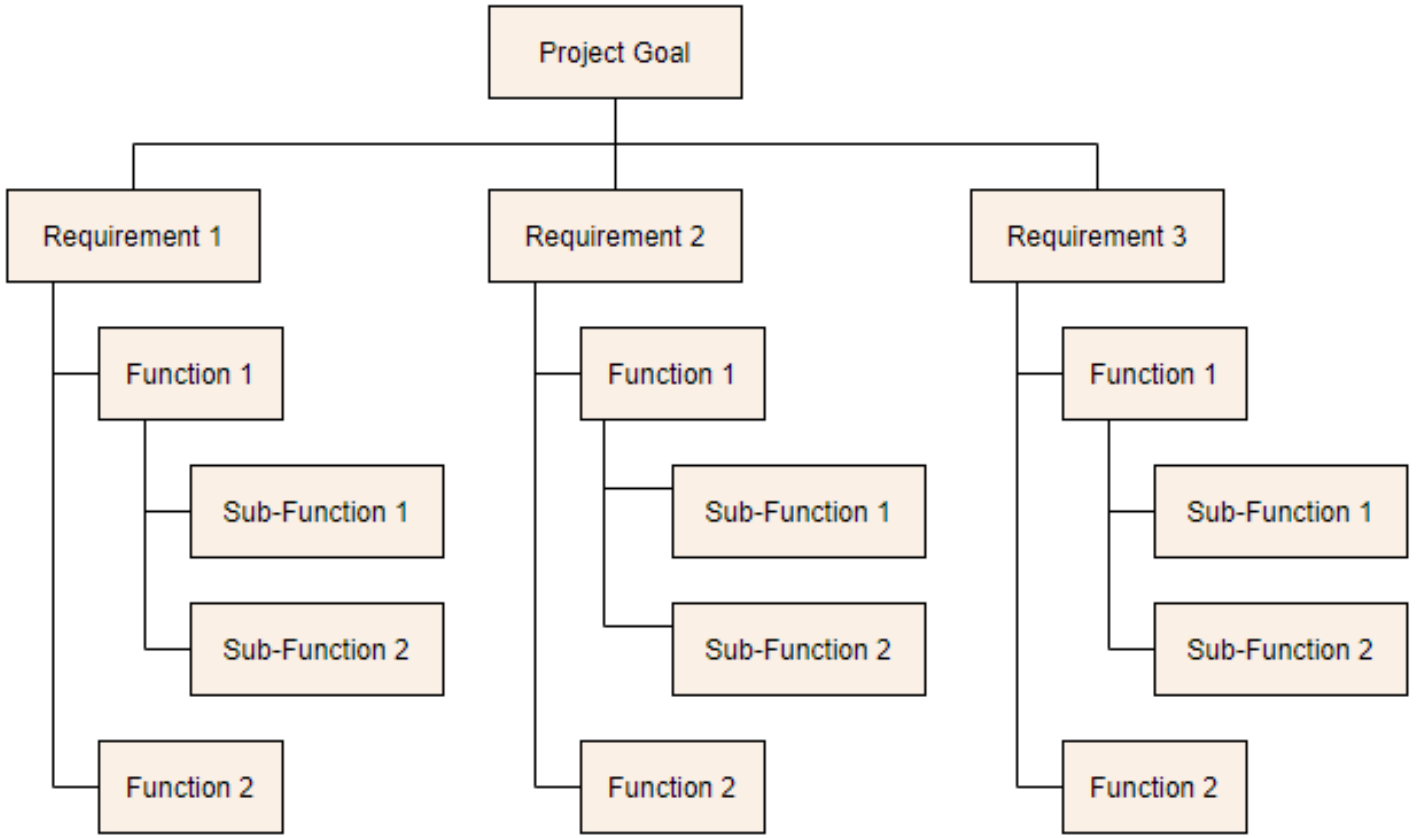


Иерархическая структура проекта WBS

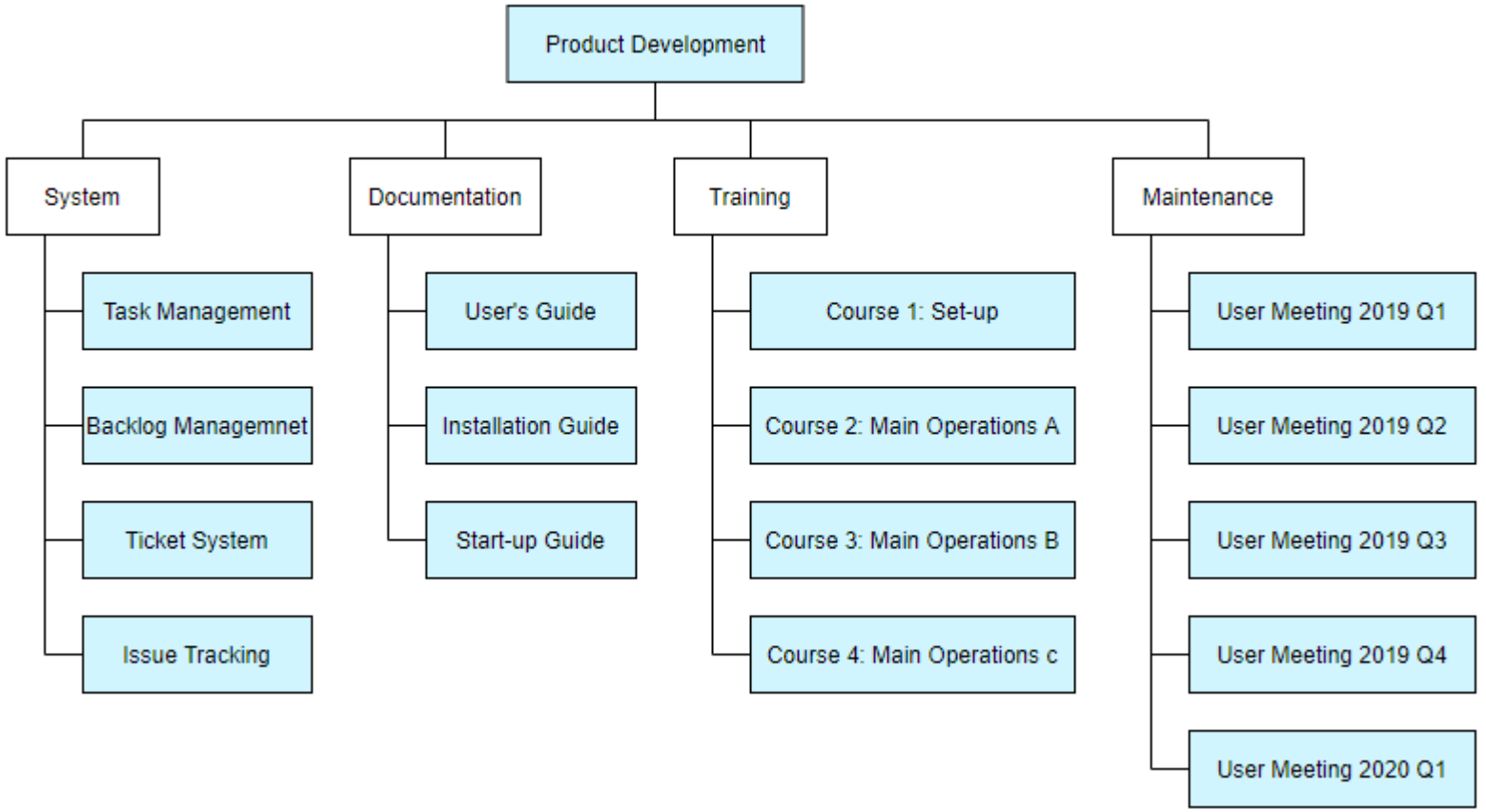
WBS – Work Breakdown Structure (иерархическая структура проекта), является частью PMBOK.

WBS является средством для разделения всех работ по проекту на ***управляемые***, ***определяемые пакеты*** работ, позволяющие достичь уровень детализации предоставляемой информации, соответствующий потребностям руководства проекта в контроле.

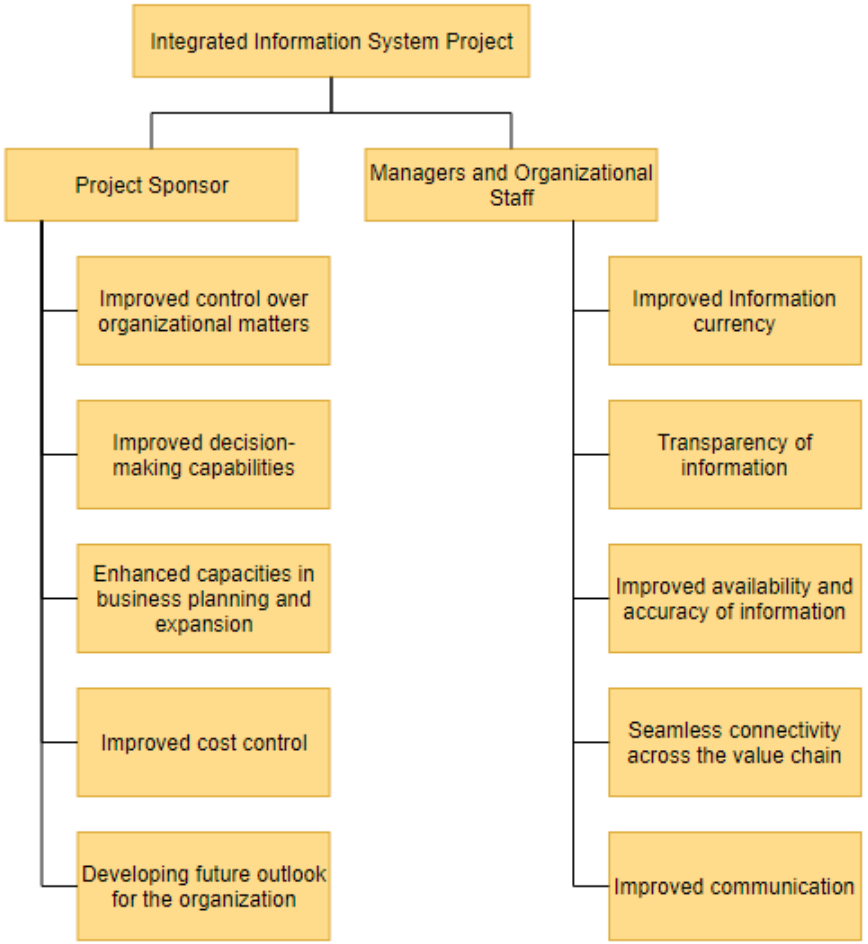
WBS - требования



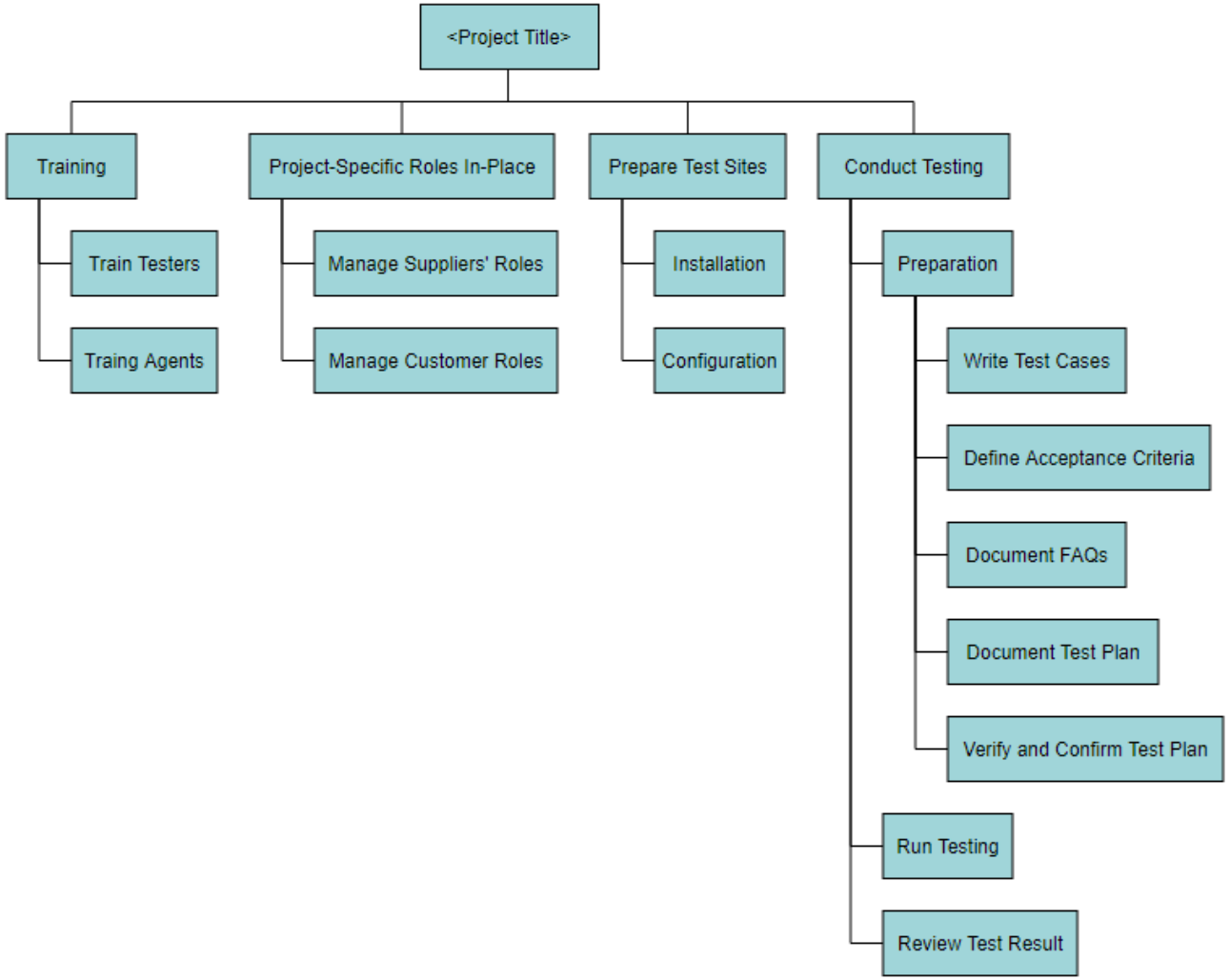
WBS – сопровождение продукта



WBS – взаимодействие со стейкхолдерами



WBS – UA тестирование



Материал для самостоятельного изучения

- Материалы лекции 4 на сайте http://msuniversity.ru
- Книга Основы инженерии ПО: Глава 8, Управление программной инженерией, стр. 613-673.
- Статья “Краткий обзор 10 популярных архитектурных шаблонов приложений”, https://medium.com/nuances-of-programming/краткий-обзор-10-популярных-архитектурных-шаблонов-приложений-81647be5c46f
- Online.visual-paradigm. PERT Chart templates. https://online.visual-paradigm.com/drive/#diagramlist:proj=0&new=PERTChart
- racichart. RACI template. https://racichart.org/raci-templates/
- Habr. 4 инструмента по полочкам. Управление проектами с WBS, Диаграммой Ганта, CPM и Time-Cost. https://habr.com/ru/post/282766/

Спасибо за внимание!