Оглавление

[1. Понятие и признаки проекта. 2](#_Toc195699720)

[2. Понятие управления программными проектами. История развития. 3](#_Toc195699721)

[3. Классификация и типы проектов. 3](#_Toc195699722)

[4. Структура проекта. Основные компоненты. 5](#_Toc195699723)

[5. Жизненный цикл проекта. 7](#_Toc195699724)

[6. Виды организационной структуры компании. 10](#_Toc195699725)

[7. Фаза инициации проекта. Управление приоритетами проекта. 12](#_Toc195699726)

[8. Концепция проекта. 13](#_Toc195699727)

[9. Анализ экономической эффективности проекта. Показатели эффективности. 13](#_Toc195699728)

[10. Анализ альтернативных проектов. 15](#_Toc195699729)

[11. Общие принципы анализа трудоёмкости проектов. 19](#_Toc195699730)

[12. Анализ трудоёмкости проекта по методике PERT. 23](#_Toc195699731)

[13. Анализ трудоёмкости проекта по методу функциональных точек. 23](#_Toc195699732)

[14. Анализ трудоёмкости проекта по методу COCOMO II. 23](#_Toc195699733)

[15. Планирование проекта. Составление иерархической структуры работ. 24](#_Toc195699734)

[16. Планирование организационной структуры проекта. 24](#_Toc195699735)

[17. Матрица ответственности проекта. 24](#_Toc195699736)

[18. Управление конфигурациями и качеством проекта. 24](#_Toc195699737)

[19. Составление базового расписания проекта. Диаграмма Ганта. 25](#_Toc195699738)

[20. Использование методов сетевого планирования в управлении проектами. Сетевой график. 26](#_Toc195699739)

[21. Оценка параметров сетевого графика. 28](#_Toc195699740)

[22. Оценка вероятности сокращения сроков выполнения проекта. 28](#_Toc195699741)

[23. Управление рисками проекта. Понятие риска. Идентификация рисков. 28](#_Toc195699742)

[24. Планирование управления рисками. 28](#_Toc195699743)

[25. Качественный анализ рисков. 28](#_Toc195699744)

[26. Количественный анализ рисков. 28](#_Toc195699745)

[27. Планирование реагирования на риски. Управление направленное на снижение рисков. 28](#_Toc195699746)

[28. Основные риски программных проектов и способы их снижения. 28](#_Toc195699747)

[29. Формирование команды проекта. Роль лидера. Мотивация. 28](#_Toc195699748)

[30. Формирование команды проекта. Организация эффективного взаимодействия между членами команды. 28](#_Toc195699749)

[31. Рабочее планирование. 28](#_Toc195699750)

[32. Принципы количественного управления проектами. 28](#_Toc195699751)

[33. Гибкие методики управления проектами (Agile) общие принципы. 28](#_Toc195699752)

[34. Методика Scrum. Основные принципы. 28](#_Toc195699753)

[35. Методика KANBAN. Основные принципы. 28](#_Toc195699754)

[36. Методика LEAN. Основные принципы. 28](#_Toc195699755)

1. Понятие и признаки проекта.

В литературе приводятся различные определения проекта:

1) проект — временное предприятие, предназначенное для

создания уникальных продуктов, услуг или результатов [1, 2];

2) проект — комплекс уникальных действий, не опирающийся на организационную структуру, имеющий определенные

дату начала и окончания, расписание, стоимость и технические

задачи [3];

3) проект — комплекс взаимосвязанных действий, предпринимаемых с целью получения уникальных конкретных результатов при заданных ограничениях по времени, денежным средствам, ресурсам и качеству конечных результатов проекта [4];

4) проект — произвольный ряд действий или задач, имеющих определенную цель, которая будет достигнута в рамках выполнения некоторых заданий, характеризующихся определенными датами начала и окончания, пределами финансирования и

ресурсами [5];

5) проект — временное усилие, применяемое для того, чтобы

создать уникальный продукт или услугу с определенной датой

начала и окончания действия, отличающегося от продолжающихся, повторных действий и требующего прогрессивного совершенствования характеристик [5]

Анализ представленных определений позволяет выделить специфические характеристики проекта:

 направленность проекта на достижение конкретного конечного результата, определяемого в терминах требуемых ресурсов, качества и времени реализации;

 уникальность проекта как разового (неповторяющегося)

мероприятия, требующего специфической организации управления;

 ограниченность проекта по времени и ресурсам (финансовым, трудовым, материальным) и, как следствие, необходимость

нахождения постоянного компромисса между объемом работ,

ресурсами, временем, качеством и рисками и их перераспределения в ходе выполнения проекта;

 структурная сложность проекта как комплекса тесно взаимосвязанных мероприятий и его высокая неопределенность,

обусловленная возможными изменениями условий реализации,

потребности в тех или иных видах ресурсов.

Обобщая вышеизложенное, определение программного проекта можно привести в следующей формулировке: это комплекс

взаимосвязанных работ, выполняемых командой проекта с целью

получения уникального программного продукта или услуги в

течение заданного периода при установленном бюджете и потребляемых в ходе реализации проекта ресурсах в условиях повышенного риска, требующих специфического управления.

Это из управления программными проектами

1. Понятие управления программными проектами. История развития.

Управление программным проектом будем рассматривать как деятельность по управлению всеми этапами ЖЦ программного продукта путем планирования, реализации, мониторинга и контроля проектных работ, в ходе которых достигаются цели проекта при нахождении компромисса между объемом работ, ресурсами, временем, качеством и рисками.

Управление проектами — методология (говорят также — искусство) организации, планирования, руководства, координации трудовых, финансовых и материально-технических ресурсов на протяжении проектного цикла, направленная на эффективное достижение его целей путем применения современных методов, техники и технологии управления для достижения определенных в проекте результатов по составу и объему работ, стоимости, времени, качеству и удовлетворению участников проекта.

Необходимость в самостоятельной дисциплине «Управление проектами» (Project Management) была осознана в развитых странах Запада с рыночной экономикой в 50-х гг. XX в. Это было вызвано массовым ростом масштабов проектов и тем, что понятие успешности проекта стало измеряться, в первую очередь, соответствием его окончательной стоимости объему выделенных ассигнований, величиной экономии и размерами прибыли.

В числе первых методов управления проектами в конце 50-х гг. были разработаны методы сетевого планирования и управления (методы СРМ и PERT). Впервые они были использованы для управления ракетной программой «Атлас» и при строительстве крупного завода синтетического волокна. В середине 60-х гг. эти методы стали активно изучаться и, в меньшей степени, внедряться в практику капитального строительства СССР.

Здесь следует отметить, что еще в конце 30-х гг. советскими учеными были разработаны теоретические основы и практические методы календарного планирования и поточного строительства с использованием диаграмм Ганта и т. н. циклограмм, что во многом можно считать фундаментом созданного позднее аппарата управления проектами. В 70-е гг. большое число крупных компаний на Западе в ответ на растущие масштабы и сложность их деятельности в условиях жесткой конкуренции стали развивать и использовать методы управления проектами. Ускорению этого процесса способствовало широкое внедрение компьютерных систем обработки информации

1. Классификация и типы проектов.

Проекты укрупнено можно классифицировать по многим при-знакам. Так, если делить проекты по затрагиваемому ими уровню управления народным хозяйством, то будет получена следующая классификация:

мегапроекты — осуществление различного рода реформ и це-левых программ на высшем уровне управления экономикой страны;

мультипроекты — переплетение в одном проекте различных субпроектов: экономических, социальных, организационных, техни-ческих, а также наличие многоканальной системы снабжения и сбы-та продукции;

монопроекты — решение преимущественно одной задачи на уровне функционирования предприятия. По масштабности решаемых задач проекты классифицируются следующим образом:

глобальные (международные), реализация которых сущест-венно влияет на экономическую, социальную или экологическую ситуацию на планете;

крупномасштабные (национальные) — масштаба одной страны;

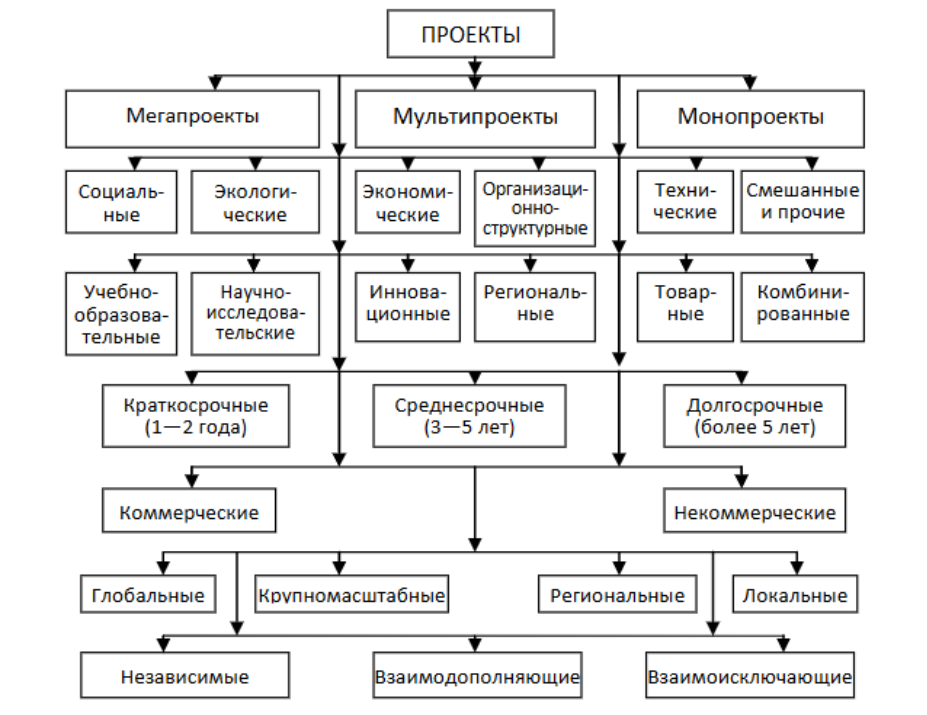
региональные (отраслевые) — городского масштаба;

локальные — масштаба предприятия. С точки зрения возможности наилучшего альтернативного выбора проекты могут быть:

независимые;

взаимодополняющие;

взаимоисключающие.



<https://www.iprbookshop.ru/epd-reader?publicationId=20044> стр 10

# Структура проекта. Основные компоненты.

В этом вопросе немножечко нихуя не понятно о какой структуре идет речь, конкретно такого определения с четким описанием ни в одной книжке я не нашел. Во всех говорится об **организационной** структуре, но это к вопросу №6, а тут пожалуй имелось в виду перечислить из чего состоит проект, ну поэтому ответ чатика вставлю хз

**Структура проекта** — это система организации его компонентов (целей, задач, ресурсов, процессов, ролей), которая обеспечивает четкое планирование, контроль и достижение результатов. Она служит «каркасом», определяющим:

* Что нужно сделать?
* Кто будет делать?
* Как и когда это будет выполнено?

**2. Основные компоненты структуры проекта**

**2.1. Цели и задачи проекта**

* **Цели** — глобальные результаты, которые должен достичь проект (например, «разработать мобильное приложение для онлайн-банкинга»).
* **Задачи** — конкретные шаги для достижения целей («реализовать авторизацию пользователей», «интегрировать платежный шлюз»).
* **Связь с требованиями заказчика**: Цели формулируются на основе потребностей стейкхолдеров. Критерии успеха (например, «приложение должно поддерживать 10 тыс. пользователей одновременно») фиксируются в техническом задании (ТЗ).

**2.2. Жизненный цикл проекта**

Этапы:

1. **Инициация**: Определение целей, анализ осуществимости, утверждение устава проекта.
2. **Планирование**: Создание планов по срокам, бюджету, рискам, качеству. Формирование WBS.
3. **Исполнение**: Реализация задач, координация команды.
4. **Мониторинг и контроль**: Сравнение плановых и фактических показателей, управление изменениями.
5. **Завершение**: Сдача продукта, оценка результатов, анализ ошибок.

**2.3. Организационная структура**

* **Роли**:
  + **Менеджер проекта**: Управляет сроками, бюджетом, коммуникациями.
  + **Разработчики/тестировщики**: Реализуют и проверяют функционал.
  + **Стейкхолдеры**: Заказчики, спонсоры, пользователи — их требования влияют на приоритеты.

**2.4. Ресурсы**

* **Человеческие**: Команда проекта (программисты, дизайнеры).
* **Технические**: Инструменты разработки, серверы, лицензии.
* **Финансовые**: Бюджет на зарплаты, закупки, маркетинг.  
  **Распределение ресурсов** отражается в календарном плане (например, с помощью диаграммы Ганта).

**2.5. Рабочая структура проекта (WBS)**

* **WBS** — иерархическая декомпозиция проекта на подзадачи. Например:
  + Уровень 1: «Разработка приложения».
  + Уровень 2: «Backend», «Frontend», «Тестирование».
  + Уровень 3: «Реализация API», «Создание интерфейса входа».
* **Роль WBS**: Упрощает оценку трудозатрат, распределение ответственности и контроль прогресса.

**2.6. Управление рисками, качеством и изменениями**

* **Риски**: Выявляются на этапе планирования (например, «нехватка разработчиков»), для каждого определяется стратегия (избежание, снижение, принятие).
* **Качество**: Обеспечивается через тестирование, код-ревью, соответствие стандартам (например, ISO).
* **Изменения**: Управляются через процесс Change Request (запрос → анализ → утверждение/отклонение).

**2.7. Документация**

* **Устав проекта**: Фиксирует цели, стейкхолдеров, бюджет.
* **Техническое задание (ТЗ)**: Детализирует требования к продукту.
* **Планы управления**: По рискам, качеству, коммуникациям.

# 5. Жизненный цикл проекта.

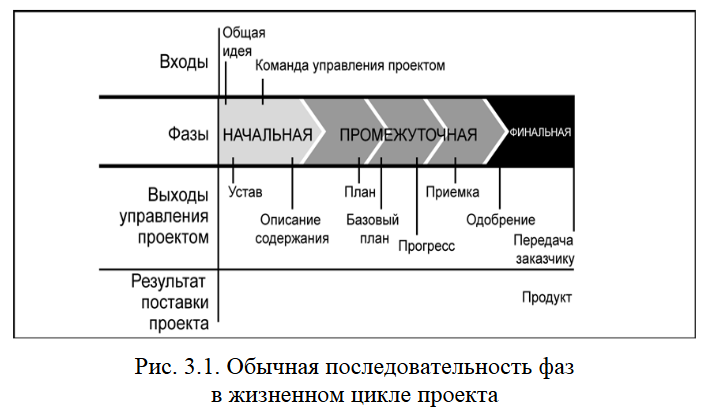
Жизненный цикл проекта (ProjectLife Cycle) — полный набор по-следовательных фаз проекта, название и число которых определяет-ся исходя из технологии производства работ и потребностей контро-ля со стороны организации или организаций, вовлеченных в проект.

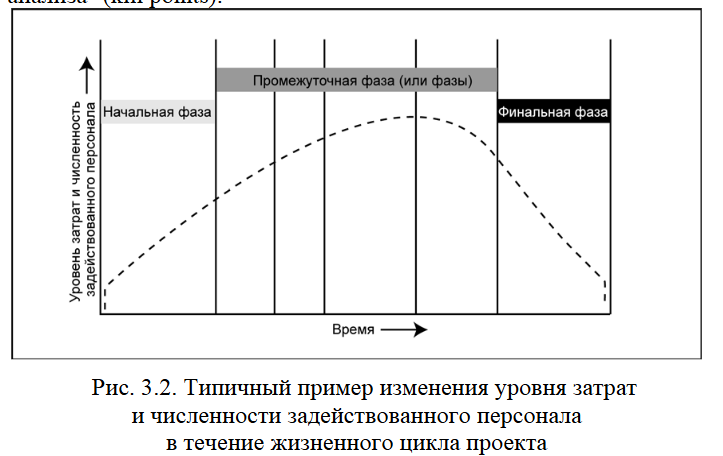
Жизненный цикл продукта(Product Life Cycle) — набор упоря-доченных идей, решений и действий с момента зарождения идеи продукта до снятия его с производства

<https://www.iprbookshop.ru/epd-reader?publicationId=20044> стр 18

Изображение выглядит как текст, снимок экрана, Шрифт, документ

Контент, сгенерированный ИИ, может содержать ошибки.





<https://www.iprbookshop.ru/epd-reader?publicationId=11440> стр 40



Выше конкретный пример из Labs/Управление%20проектами/6024-upravlenie\_programm\_proekt.pdf стр 28

**1. Инициация**

На этапе инициации определяются цели проекта, его обоснование и ключевые участники. Это начальная фаза, где принимается решение о запуске проекта. Основные задачи включают:

* Определение потребностей и целей проекта.
* Оценка целесообразности и рисков.
* Назначение руководителя проекта и ключевых участников команды.
* Составление устава проекта.

**2. Планирование**

Этап планирования включает разработку детального плана действий для достижения целей проекта. Здесь создаются:

* План управления проектом, который охватывает такие аспекты, как сроки, бюджет, ресурсы, риски и качество.
* График работ (например, диаграмма Ганта).
* Смета затрат.
* Матрица ответственности.
* Планы коммуникаций и управления рисками.

**3. Исполнение**

На стадии исполнения план проекта реализуется на практике. Это самая активная фаза, когда выполняются работы, необходимые для достижения поставленных целей. Включает следующие задачи:

* Организация и координация работы команды.
* Выполнение запланированных задач.
* Управление изменениями, возникающими в процессе реализации.
* Контроль качества выполняемых работ.

**4. Мониторинг и контроль**

Этот этап обеспечивает постоянный мониторинг хода выполнения проекта и сравнение фактического состояния дел с плановыми показателями. Основные функции мониторинга и контроля:

* Сбор и анализ данных о ходе проекта.
* Внесение корректировок в случае отклонений от плана.
* Отчетность перед заинтересованными сторонами.
* Регулярный пересмотр планов и графиков.

**5. Завершение**

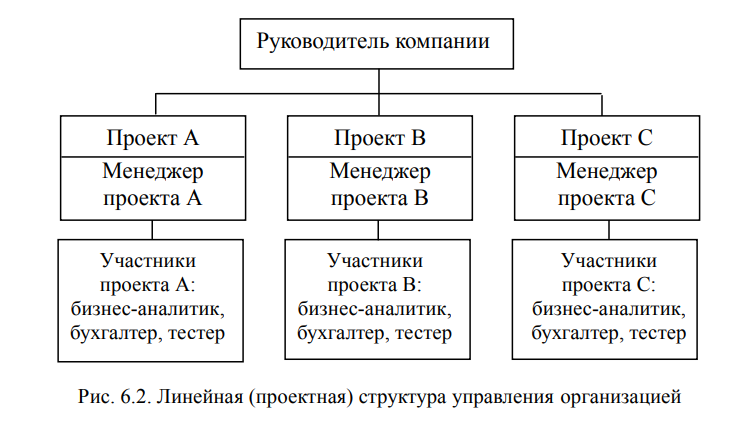
Заключительный этап, на котором происходит завершение всех работ и формальное закрытие проекта. Этот процесс включает:

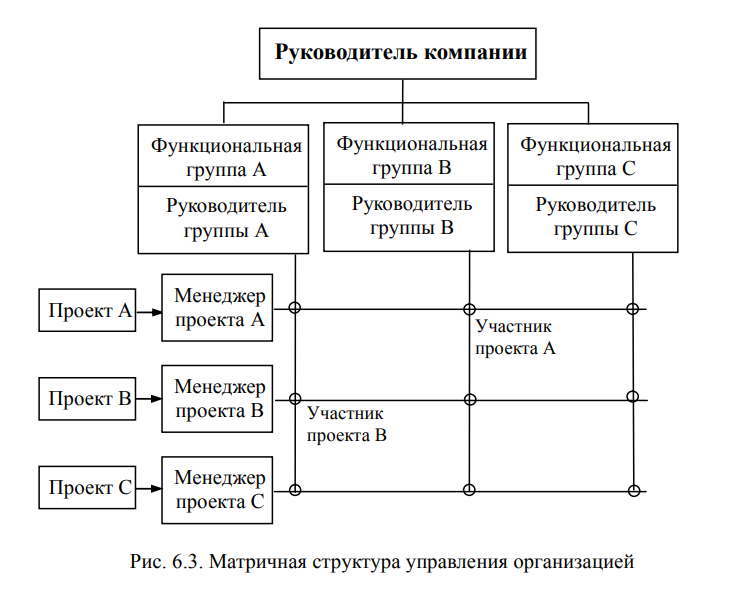
* Проверку результатов проекта на соответствие целям и требованиям.
* Подведение итогов и документирование опыта.
* Передача результатов заказчику или конечным пользователям.
* Архивирование проектной документации.
* Формальная ликвидация проектной команды.

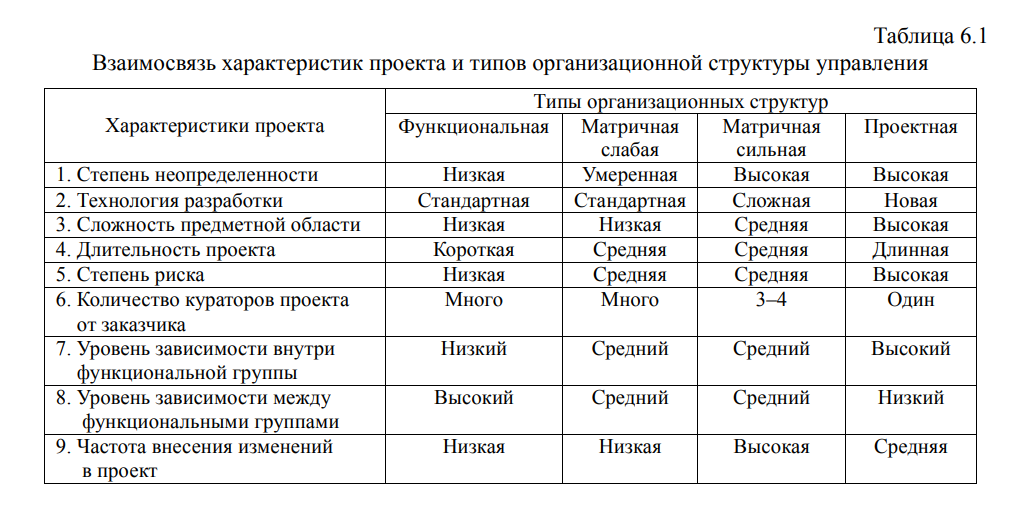
# 6. Виды организационной структуры компании.

Labs/Управление%20проектами/6024-upravlenie\_programm\_proekt.pdf стр 143









# 7. Фаза инициации проекта. Управление приоритетами проекта.

Про приоритеты тут все красиво

<https://www.iemag.ru/master-class/detail.php?ID=15700>

Инициация какбэ включает в себя

* Создание творческого ядра в команде разрабов продукта
* Генерацию и отбор идей
* На основе оных идей разработка концепций
* Отбор наиболее перспективной концепции

Глава 4 вся посвящена этой инициации, а 4.1 конкретно генерации и отбору идей, а концепция проекта это уже некст вопрос, надо ли о ней тут рассказывать я хуй его знает

Labs/Управление%20проектами/6024-upravlenie\_programm\_proekt.pdf стр 84

# 8. Концепция проекта.

Книга из пред. Вопроса, глава 4.2 стр 89

Концепция программного проекта содержит следующие разделы:

1) бизнес-обоснование потребности или необходимости в

разработке ПП;

2) цели, ограничения и содержание программного проекта;

3) основные сегменты рынка и потенциальные пользователи;

4) экономика программного проекта;

5) потенциал исполнителей;

6) ожидаемые риски программного проекта.

# 9. Анализ экономической эффективности проекта. Показатели эффективности.

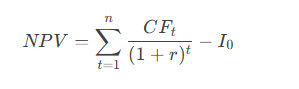
В принципе достаточно нищий вопрос для того, чтобы банально сгенерить ответ чатом, искать эту инфу в книжках – только ебать себе мозги.

**Анализ экономической эффективности проекта: основные показатели**

Оценка экономической эффективности проекта позволяет определить его финансовую целесообразность, сопоставляя затраты и будущие доходы. Ниже приведены ключевые показатели, используемые для анализа:

**1. Чистая приведенная стоимость (NPV, Net Present Value)**

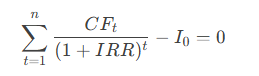
* **Определение**: NPV отражает разницу между текущей стоимостью денежных потоков и первоначальными инвестициями.
* **Формула**:

где:

* + *CFt*​ — денежный поток в период t*t*,
  + *r* — ставка дисконтирования,
  + *I*0​ — начальные инвестиции.
* **Интерпретация**:
  + NPV>0: проект прибылен.
  + NPV=0: проект окупает затраты.
  + NPV<0: проект убыточен.
* **Преимущества**: Учитывает временную стоимость денег и все денежные потоки.
* **Ограничения**: Требует точного выбора ставки дисконтирования.

**2. Внутренняя норма доходности (IRR, Internal Rate of Return)**

* **Определение**: Ставка дисконтирования, при которой NPV = 0.
* **Формула**: Решается уравнение:



* **Интерпретация**:
  + Если IRR >*r*, проект считается эффективным.
* **Преимущества**: Показывает доходность в процентах.
* **Ограничения**: Может давать множественные решения при нестандартных денежных потоках. Не учитывает масштаб проекта.

**3. Срок окупаемости (Payback Period, PP)**

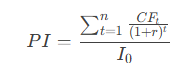
* **Определение**: Время, за которое инвестиции окупаются за счет чистых денежных потоков.
* **Типы**:
  + **Простой PP**: Без учета временной стоимости денег.
  + **Дисконтированный PP**: С учетом дисконтирования потоков.
* **Формула**:

****

* **Интерпретация**: Чем меньше срок, тем ниже риски.
* **Преимущества**: Простота расчета.
* **Ограничения**: Игнорирует денежные потоки после срока окупаемости.

**4. Индекс рентабельности (PI, Profitability Index)**

* **Определение**: Отношение приведенной стоимости денежных потоков к начальным инвестициям.
* **Формула**:

​​​

* **Интерпретация**:
  + PI>1: проект эффективен.
* **Преимущества**: Учитывает масштаб инвестиций.
* **Ограничения**: Зависит от ставки дисконтирования.

**Итог**: Экономическая эффективность проекта оценивается через комбинацию показателей, учитывающих доходность, риски и временную стоимость денег. Оптимальное решение требует анализа всех критериев и учета специфики проекта.

# 10. Анализ альтернативных проектов.

Тоже хуй проссышь о чем речь, чатик кормит

Анализ альтернативных проектов в управлении программными проектами — это процесс сравнения нескольких вариантов реализации инициатив с целью выбора оптимального решения, которое максимизирует ценность для организации. Этот подход опирается на методы экономического анализа, управление рисками, стратегическое выравнивание и учет ограничений ресурсов. Рассмотрим ключевые аспекты, опираясь на классические и современные источники по управлению проектами.

**1. Стратегическое выравнивание**

Любой проект должен соответствовать стратегическим целям организации. Как отмечает **PMBOK® Guide (7th Edition)**, проекты должны оцениваться через призму их вклада в бизнес-ценность. Например:

* **Пример**: Выбор между разработкой нового продукта и модернизацией существующей системы зависит от того, как это согласуется с долгосрочной стратегией компании (увеличение доли рынка vs. оптимизация затрат).

**Книги**:

* **"Strategic Project Management"** (Paul Roberts) подчеркивает, что проекты без четкой связи со стратегией часто становятся «зомби-проектами», поглощающими ресурсы без отдачи.

**2. Экономическая оценка альтернатив**

Для анализа используются финансовые показатели, такие как **NPV, IRR, PI** (см. предыдущий ответ), но в контексте программных проектов важно учитывать специфические факторы:

* **Стоимость владения (TCO)**: Включает не только разработку, но и поддержку, обновления, интеграцию с другими системами.
* **Гибкость архитектуры**: Проект с более высокой начальной стоимостью, но модульной архитектурой, может быть выгоднее в долгосрочной перспективе (см. **"Software Engineering Economics"**, Barry Boehm).

**Пример**: Выбор между:

* **Вариант A**: Разработка кастомного решения (высокие начальные затраты, но полный контроль).
* **Вариант B**: Использование SaaS-платформы (низкие затраты на старте, но зависимость от вендора).

**3. Управление рисками**

Анализ альтернатив требует оценки рисков каждого варианта. **"Waltzing with Bears"** (Tom DeMarco, Timothy Lister) предлагает:

* **Количественная оценка рисков**: Например, вероятность задержки из-за сложности интеграции.
* **Сценарный анализ**: Моделирование «наихудшего», «наилучшего» и «ожидаемого» сценариев.

**Пример**: Проект с использованием новой технологии (высокий риск провала, но потенциальное конкурентное преимущество) vs. проверенный стек (низкий риск, но меньшая инновационность).

**4. Ограничения ресурсов**

По **"The Mythical Man-Month"** (Fred Brooks), добавление людей в проект не всегда ускоряет его. При выборе альтернатив нужно учитывать:

* **Наличие экспертизы команды**: Например, проект на Python может быть предпочтительнее, если команда не имеет опыта в Java.
* **Временные ограничения**: Agile-проекты с итеративной разработкой могут быть менее рискованными при сжатых сроках (см. **"Scrum: The Art of Doing Twice the Work in Half the Time"**, Jeff Sutherland).

**5. Методы анализа альтернатив**

**a) Модели scoring**

Используются критерии с весами (например, стратегическая значимость, ROI, техническая сложность).

* **Пример из практики**: В **"Project Portfolio Management"** (Harvey Levine) описывается матрица, где проекты ранжируются по баллам от 1 до 5.

**b) Cost-Benefit Analysis (CBA)**

Сравнение затрат и выгод, включая нематериальные факторы (улучшение репутации, удовлетворенность клиентов). По **"Software Estimation"** (Steve McConnell), для ПО важно оценивать не только прямые затраты, но и «стоимость ошибки» (например, ущерб от багов).

**c) Real Options**

Метод, предложенный в **"Real Options"** (Tom Copeland), подходит для ИТ-проектов с высокой неопределенностью. Например, инвестиции в MVP (Minimum Viable Product) позволяют отложить решение о полном запуске до получения обратной связи.

**6. Case Study: Agile vs. Waterfall**

* **Waterfall**: Четкий план, но низкая гибкость. Подходит для проектов с фиксированными требованиями (например, разработка ПО для медицинского оборудования).
* **Agile**: Итеративность и адаптивность. Как отмечает **"User Story Mapping"** (Jeff Patton), Agile позволяет быстро тестировать гипотезы и перенаправлять ресурсы между альтернативными фичами.

**7. Рекомендации из учебных пособий**

* **"Making Things Happen"** (Scott Berkun): Приоритезируйте проекты, которые создают «цепную реакцию ценности» (например, автоматизация процесса освобождает ресурсы для других задач).
* **"Lean Software Development"** (Mary Poppendieck): Устраняйте альтернативы, которые не добавляют ценности (бюрократия, избыточная функциональность).

**Итог:**

Анализ альтернативных проектов в управлении программными проектами требует **комплексного подхода**:

1. **Стратегия**: Соответствие целям организации.
2. **Экономика**: NPV, TCO, ROI.
3. **Риски**: Оценка по DeMarco и Lister.
4. **Ресурсы**: Учет закона Брукса и экспертизы команды.
5. **Методы**: Scoring, CBA, Real Options.

**Ключевая мысль**: Как пишет **Robert Cooper в "Winning at New Products"**, успешные компании тратят до 50% времени на анализ альтернатив на ранних стадиях, чтобы избежать дорогостоящих ошибок на этапе реализации. В программных проектах это особенно критично из-за высокой скорости изменений и зависимости от внешних факторов (рынок, технологии).

# 11. Общие принципы анализа трудоёмкости проектов.

Ну хорошо, вы меня раскусили. Мне просто в падлу читать эти злоебучие книги чтобы искать ответы на эти злоебучие вопросы. Книг этих – жопой жуй, и в каждой эти «общие единые незыблемые» принципы нихуя не общие не единые и не незыблемые, везде сука разные. Короче из ответа чата можно че то по верхам пройтись почитать, а конкретнее по каждому методу оценки трудоемкости в следующих вопросах.

**Общие принципы анализа трудоёмкости проектов: ключевые аспекты и методы**

Анализ трудоёмкости проектов — это оценка времени и ресурсов, необходимых для выполнения задач, с учётом специфики команды, технологий и требований. В управлении программными проектами это критически важно для планирования сроков, бюджета и распределения ресурсов. Рассмотрим основные принципы, опираясь на классические и современные источники.

**1. Базовые понятия и цели**

**Трудоёмкость** — это количество человеко-часов, дней или месяцев, требуемых для реализации проекта или его этапа.  
**Цели анализа**:

* Предотвращение перерасхода бюджета и срывов сроков.
* Оптимизация загрузки команды.
* Формирование реалистичных ожиданий у стейкхолдеров.

**Пример из практики**:  
В книге **«Software Estimation: Demystifying the Black Art»** (Steve McConnell) подчёркивается, что ошибки в оценке трудоёмкости — основная причина провала 60% ИТ-проектов.

**2. Методы оценки трудоёмкости**

**a) Экспертные оценки**

* **Суть**: Опытные участники команды (архитекторы, тимлиды) прогнозируют затраты на основе аналогичных проектов.
* **Плюсы**: Учитывает нюансы конкретной команды и технологии.
* **Минусы**: Субъективность.
* **Пример**: В **«Peopleware»** (Tom DeMarco, Timothy Lister) описано, как «мозговой штурм» экспертов снижает погрешность оценок.

**b) Параметрические модели**

* **COCOMO (Constructive Cost Model)**: Алгоритм, разработанный Barry Boehm (**«Software Engineering Economics»**), который связывает трудоёмкость с количеством строк кода (KLOC) и коэффициентами сложности.

Effort=a×(KLOC)b×EAF*Effort*=*a*×(*KLOC*)*b*×*EAF*

где a,b*a*,*b* — константы проекта, EAF*EAF* — факторы окружения (например, опыт команды).

* **Function Points**: Оценка через функциональные модули системы (ввод данных, отчёты и т.д.).

**c) Оценка по аналогии**

* Сравнение с завершёнными проектами. Например, если предыдущий модуль занял 200 часов, аналогичный оценивается в ±20%.

**d) Agile-подходы**

* **Story Points**: В Scrum задачи оцениваются в относительных единицах сложности (1, 2, 3, 5, 8 и т.д.).
* **Velocity**: На основе прошлых спринтов прогнозируется, сколько story points команда выполнит за итерацию.
* **Источник**: **«Scrum: The Art of Doing Twice the Work in Half the Time»** (Jeff Sutherland).

**3. Факторы, влияющие на трудоёмкость**

* **Сложность задач**:  
  Технические риски, интеграция с legacy-системами, требования к безопасности.
* **Опыт команды**:  
  По **«The Mythical Man-Month»** (Fred Brooks), добавление новых людей в проект может увеличить трудоёмкость из-за затрат на коммуникацию.
* **Изменчивость требований**:  
  Частые правки в ТЗ (типично для Agile) требуют гибкого пересчёта оценок.
* **Качество документации**:  
  Чёткие спецификации сокращают время на уточнения (см. **«Clean Agile»** Robert C. Martin).

**4. Инструменты и технологии**

* **Диаграммы Ганта**: Визуализация сроков и зависимостей задач (Microsoft Project, Jira).
* **Метод PERT**: Оценка времени по формуле:

t=O+4M+P6*t*=6*O*+4*M*+*P*​

где O*O* — оптимистичный срок, M*M* — реалистичный, P*P* — пессимистичный.

* **Машинное обучение**: Современные системы (например, **Google’s AutoML**) анализируют исторические данные для прогнозирования трудоёмкости.

**5. Рекомендации из литературы**

* **PMBOK® Guide**:
  + Разбивайте проект на мелкие задачи (Work Breakdown Structure).
  + Учитывайте риски через резервы времени («буферы»).
* **«Lean Software Development»** (Mary Poppendieck):  
  Устраняйте «мусорные» активности (например, избыточные согласования), которые увеличивают трудоёмкость.
* **«Agile Estimating and Planning»** (Mike Cohn):  
  Регулярно пересматривайте оценки по мере поступления новых данных (принцип «probe-adjust»).

**6. Кейс: Waterfall vs. Agile**

* **Waterfall**:  
  Трудоёмкость оценивается на этапе планирования. При изменении требований оценки становятся неактуальными.
* **Agile**:  
  Постепенная оценка в каждом спринте. Пример из **«Succeeding with Agile»** (Mike Cohn): Команда снизила погрешность оценок с 50% до 15% за счёт ретроспектив и анализа velocity.

**Итог:**

Анализ трудоёмкости — это **итеративный процесс**, требующий:

1. **Комбинации методов**: Экспертные оценки + параметрические модели.
2. **Учёта человеческого фактора**: Опыт команды, мотивация, коммуникация.
3. **Адаптивности**: Регулярный пересмотр оценок в Agile-проектах.

**Ключевая мысль**: Как отмечает **Barry Boehm**, «ошибка в оценке трудоёмкости на 50% в начале проекта приводит к отклонению в сроках на 35%». Поэтому инвестиции в качественный анализ на старте окупаются снижением рисков и повышением предсказуемости результатов.

# 12. Анализ трудоёмкости проекта по методике PERT.

<https://asana.com/ru/resources/pert-chart>  
ну типа лол время выполнения любой задачи эта хуйня оценивает как

(пессимистичная оценка + 4 \* реалистичная оценка + оптимистичная оценка) / 6

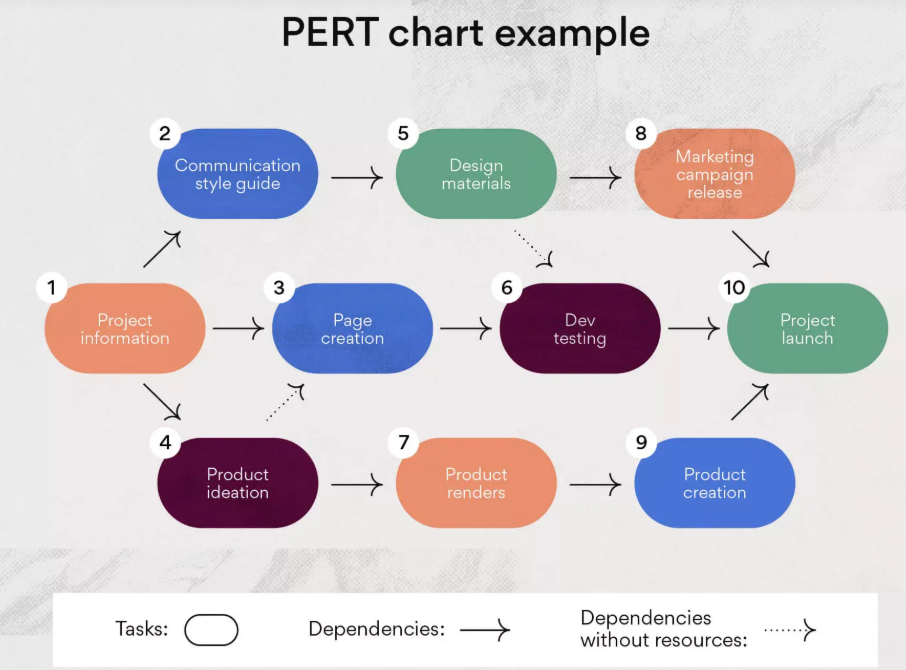
Мда технологии я ахуеваю. Таким образом в большинстве случаев (кто бы блять мог подумать) ожидаемое время выполнения задачи будет близко к его реалистичной оценке…

Еще есть PERT диаграмма, типа строятся события в проекте а стрелочками между ними зависимости, и типа можно увидеть последовательность задач и какие из них могут выполняться параллельно.

Внимание цитата: Цель состоит в том, чтобы найти наиболее долгий путь, на который уйдёт больше всего времени, чтобы оценить наиболее наименьшую общую длительность проекта

Ну это же просто пиздец, как такое можно высрать.

Рисунок ебану по приколу

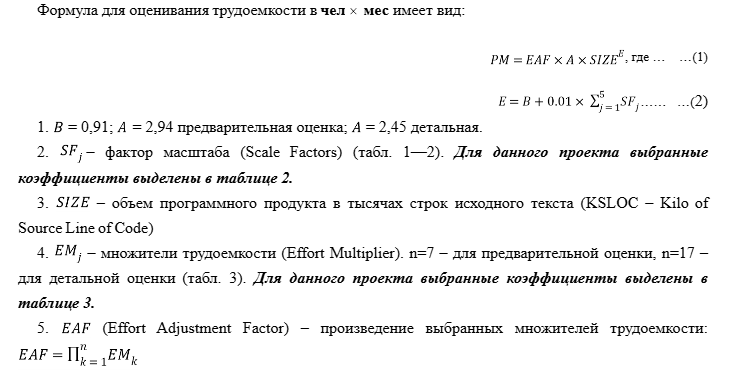


# 13. Анализ трудоёмкости проекта по методу функциональных точек.

<https://citforum.ru/SE/project/arkhipenkov_lectures/12.shtml>

ну тут по ссылочке все красиво, лучше и не скажешь. Все прям знать я думаю не обязательно, но представление иметь надо.

# 14. Анализ трудоёмкости проекта по методу COCOMO II.



<http://economyandbusiness.ru/primenenie-modeli-cocomo-ii-dlya-otsenki-razrabotki-programmnogo-obespecheniya-v-windows-proektah>

гамнишко

# 15. Планирование проекта. Составление иерархической структуры работ.

<https://www.youtube.com/watch?v=PmDGdmB0zi4>

ну типа хз, вот как звучит так оно и есть на самом деле, иерархию работ построить надо.

# 16. Планирование организационной структуры проекта.

Пока нихуя не понял, у фроловой надо уточнить в чем разница с вопросом №6

# 17. Матрица ответственности проекта.

<https://skillbox.ru/media/management/chto-takoe-matritsa-raci-i-kak-ona-pomogaet-vypolnit-proekt-v-srok-ne-rasteryav-zadachi/>

вопрос достаточно простой, по ссылке все красиво

# 18. Управление конфигурациями и качеством проекта.

<https://aspro.cloud/blog/project-quality-control/>

по ссылке про качество, кофигурация ниже

Управление конфигурацией в IT-проектах — это процесс, который помогает следить за всеми элементами системы, их состоянием и изменениями. Это нужно для того, чтобы гарантировать совместимость и функциональность системы в течение всего её жизненного цикла. 📊

Приведем простой пример:

1. **Программное обеспечение**: Представьте, что вы создаете приложение. Каждая версия приложения — это конфигурация. Когда вы добавляете новую функцию или исправляете ошибку, вы создаете новую конфигурацию. Управление конфигурацией помогает следить за тем, какая версия приложения используется в какой среде (например, тестирование, подготовка к выпуску, рабочая).
2. **Аппаратное обеспечение**: Если у вас есть серверы с разным оборудованием, управление конфигурацией поможет отслеживать, какое оборудование установлено на каждом сервере. Это важно, чтобы знать, какие обновления или замену оборудования нужно проводить.
3. **Документация**: Кроме программ и оборудования, важно управлять и документацией — например, руководствами пользователя или техническими требованиями. Если ваше приложение изменяется, нужно обновить и соответствующую документацию.

Управление конфигурацией включает в себя следующие шаги:

* **Идентификация**: Определение всех элементов конфигурации.
* **Контроль**: Мониторинг изменений, чтобы избегать нежелательных изменений и ошибок.
* **Аудит**: Проверка, чтобы убедиться, что фактическое состояние соответствует запланированному.

Это позволяет команде лучше координировать усилия, минимизировать ошибки и обеспечивать более стабильную работу проекта! 🚀

# 19. Составление базового расписания проекта. Диаграмма Ганта.

**Базовое расписание проекта** — это утвержденный график, который включает в себя:

* Список задач
* Даты начала и окончания каждой задачи
* Зависимости между задачами
* Продолжительность

Основная цель базового расписания — **определить, когда и как долго будут выполняться задачи**.

Составление **базового расписания** — это структурированный процесс, который включает:

1. Определение задач
2. Оценку продолжительности
3. Установку зависимостей
4. Распределение задач по времени
5. Визуализацию расписания с использованием диаграмм Ганта или других инструментов.

**Диаграмма Ганта** — это график в виде полос (баров), который показывает:

* Список задач по вертикали
* Календарные сроки по горизонтали
* Для каждой задачи — когда она начинается, заканчивается и сколько длится

📌 Она **визуализирует базовое расписание** проекта.

# 20. Использование методов сетевого планирования в управлении проектами. Сетевой график.

Меня окончательно заебали книжки, чат – в атаку

🔹 Что такое сетевое планирование?

**Сетевое планирование** — это метод управления проектами, который используется для планирования, координации и контроля выполнения задач (работ), входящих в проект. Он основывается на построении **сетевого графика** — визуального представления логической последовательности выполнения задач.

Методы сетевого планирования помогают:

* выявить зависимость между задачами,
* определить критический путь,
* рассчитать общую продолжительность проекта,
* оценить временные резервы.

🔹 Основные термины

* **Работа (операция)** — это отдельная задача, которая занимает время и требует ресурсов.
* **Событие (веха)** — это факт завершения одной или нескольких работ и/или начало других.
* **Сетевой график** — граф, в котором вершины — события, а дуги — работы.
* **Критический путь** — самая длинная последовательность зависимых работ, определяющая минимальное время завершения проекта.
* **Временные резервы** — время, на которое выполнение задачи можно отложить без влияния на общий срок проекта.

🔹 Виды методов сетевого планирования

1. **Метод критического пути (Critical Path Method, CPM)**
   * Используется для определения наиболее длительного пути в проекте.
   * Позволяет найти критические задачи, от которых зависит срок завершения проекта.
   * Основное внимание уделяется срокам.
2. **Метод оценки и анализа программ (Program Evaluation and Review Technique, PERT)**
   * Используется при высокой степени неопределённости.
   * Позволяет учитывать три вида оценки времени:
     + оптимистичную (O),
     + наиболее вероятную (M),
     + пессимистичную (P),
   * Формула расчёта ожидаемого времени:  
     Te = (O + 4M + P) / 6

🔹 Как строится сетевой график?

1. Определить все задачи проекта и их длительность.
2. Установить зависимости между задачами (что за чем следует).
3. Нарисовать граф: задачи в виде стрелок, события — в виде узлов.
4. Рассчитать:
   * ранние и поздние сроки начала и окончания каждой работы,
   * временные резервы,
   * критический путь.

🔹 Применение в управлении проектами

* **Планирование сроков** — можно оценить реалистичные сроки проекта.
* **Выявление узких мест** — критические задачи требуют особого контроля.
* **Распределение ресурсов** — за счёт временных резервов можно оптимизировать загрузку команды.
* **Контроль выполнения** — сравнение фактических сроков с планом по сетевому графику.

🔹 Пример (упрощённый)

Задачи:

* A (3 дня)
* B (5 дней, после A)
* C (2 дня, после A)
* D (4 дня, после B и C)

Сетевой график будет содержать 4 работы и 5 событий. Критический путь: A → B → D = 3 + 5 + 4 = **12 дней**

🔹 Заключение

Методы сетевого планирования — это мощный инструмент в управлении проектами. Они позволяют эффективно управлять сроками, ресурсами и контролировать ход выполнения проекта. Особенно полезны в сложных ИТ-проектах с множеством взаимосвязанных задач.

# 21. Оценка параметров сетевого графика.

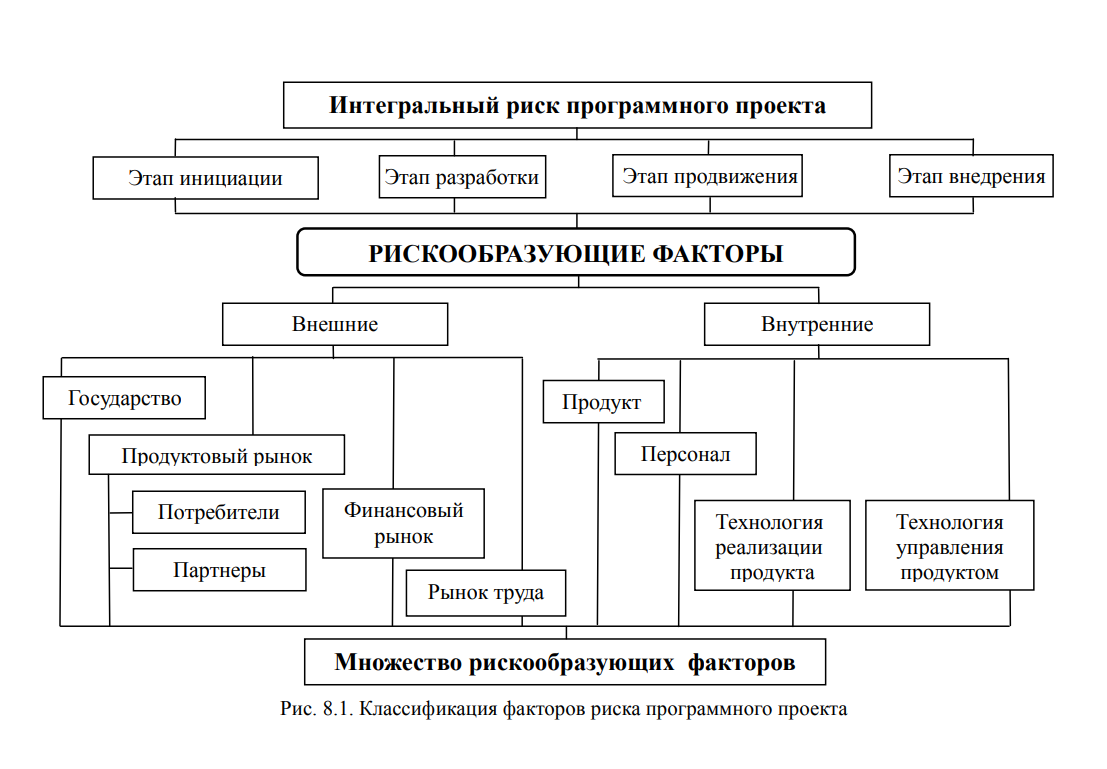
<https://projmanagsiu.ucoz.ru/publ/vvedenie/7_5_raschet_parametrov_setevogo_grafika/2-1-0-22>

все красиво

# 22. Оценка вероятности сокращения сроков выполнения проекта.

# 23. Управление рисками проекта. Понятие риска. Идентификация рисков.

В литературе под риском проекта понимается событие или условие, которое может произойти либо не произойти в будущем при реализации программного проекта и негативно повлиять на степень достижения одной или нескольких характеристик целей проекта.



Идентификация — этап, позволяющий выявить и коллективно обсудить возможность проявления риска и рискообразующих факторов, способных повлиять на цели проекта, документально описать результаты в виде логически увязанных их характеристик.

# 24. Планирование управления рисками.

Labs/Управление%20проектами/6024-upravlenie\_programm\_proekt.pdf

В этой книжке с 200 страницы так и поперло про эти риски хуиски, 25 вопрос туда же

# 25. Качественный анализ рисков.

# 26. Количественный анализ рисков.

PERT

# 27. Планирование реагирования на риски. Управление направленное на снижение рисков.

# 28. Основные риски программных проектов и способы их снижения.

## **1. Неправильное понимание требований**

**Риск:** Заказчик и разработчики по-разному интерпретируют требования → продукт не соответствует ожиданиям.  
**Как снизить:**  
✅ Чёткий **анализ требований** (SRS, User Stories, Use Cases).  
✅ **Прототипирование** (Figma, интерактивные макеты).  
✅ **Регулярные демо** заказчику (Agile, Scrum).  
✅ Использование **BDD/TDD** (тесты на основе требований).

## **2. Недооценка сроков и бюджета**

**Риск:** Оптимистичные оценки → дедлайны сорваны, бюджет исчерпан.  
**Как снизить:**  
✅ **Разбиение на этапы** (MVP → постепенное развитие).  
✅ **Методы оценки** (PERT, Planning Poker, Monte Carlo).  
✅ **Буфер времени** (+20–30% к оценкам).  
✅ **Гибкие методологии** (Agile, Kanban вместо Waterfall).

## **3. Проблемы с качеством кода**

**Риск:** Техдолг, баги, нечитаемый код → замедление разработки.  
**Как снизить:**  
✅ **Code Review** (GitHub/GitLab MR).  
✅ **Статический анализ** (SonarQube, ESLint, Pylint).  
✅ **Автоматизированное тестирование** (Unit, Integration, E2E).  
✅ **Стандарты кодирования** (Google Style Guides, PEP 8).

## **4. Уход ключевых разработчиков**

**Риск:** Потеря знаний → задержки и рост стоимости.  
**Как снизить:**  
✅ **Документация** (Confluence, Notion, Swagger).  
✅ **Парное программирование** (Knowledge Sharing).  
✅ **Мотивация команды** (карьерный рост, бонусы).  
✅ **Перекрёстное обучение** (разработчики знают разные части системы).

## **5. Проблемы с безопасностью**

**Риск:** Уязвимости → утечки данных, атаки.  
**Как снизить:**  
✅ **Security Review** (OWASP Top 10, Pentest).  
✅ **Шифрование данных** (TLS, AES).  
✅ **Принцип минимальных привилегий** (RBAC).  
✅ **Автоматическое сканирование** (Dependabot, Snyk).

## **6. Несовместимость и интеграционные проблемы**

**Риск:** API/библиотеки не работают вместе → переделки.  
**Как снизить:**  
✅ **Раннее тестирование API** (Postman, Swagger).  
✅ **Стандартизация** (REST/GraphQL, версионирование).  
✅ **Моки и заглушки** (для тестирования интеграций).

## **7. Масштабируемость и производительность**

**Риск:** Система тормозит при росте нагрузки.  
**Как снизить:**  
✅ **Нагрузочное тестирование** (JMeter, k6).  
✅ **Кеширование** (Redis, CDN).  
✅ **Горизонтальное масштабирование** (Kubernetes, микросервисы).

## **8. Юридические и compliance-риски**

**Риск:** Нарушение GDPR, лицензий ПО → штрафы.  
**Как снизить:**  
✅ **Аудит лицензий** (FOSS, коммерческие библиотеки).  
✅ **Юридическая проверка** (GDPR, HIPAA, CCPA).

# 29. Формирование команды проекта. Роль лидера. Мотивация.

# 30. Формирование команды проекта. Организация эффективного взаимодействия между членами команды.

# 31. Рабочее планирование.

# 32. Принципы количественного управления проектами.

# 33. Гибкие методики управления проектами (Agile) общие принципы.

# 34. Методика Scrum. Основные принципы.

# 35. Методика KANBAN. Основные принципы.

# 36. Методика LEAN. Основные принципы.