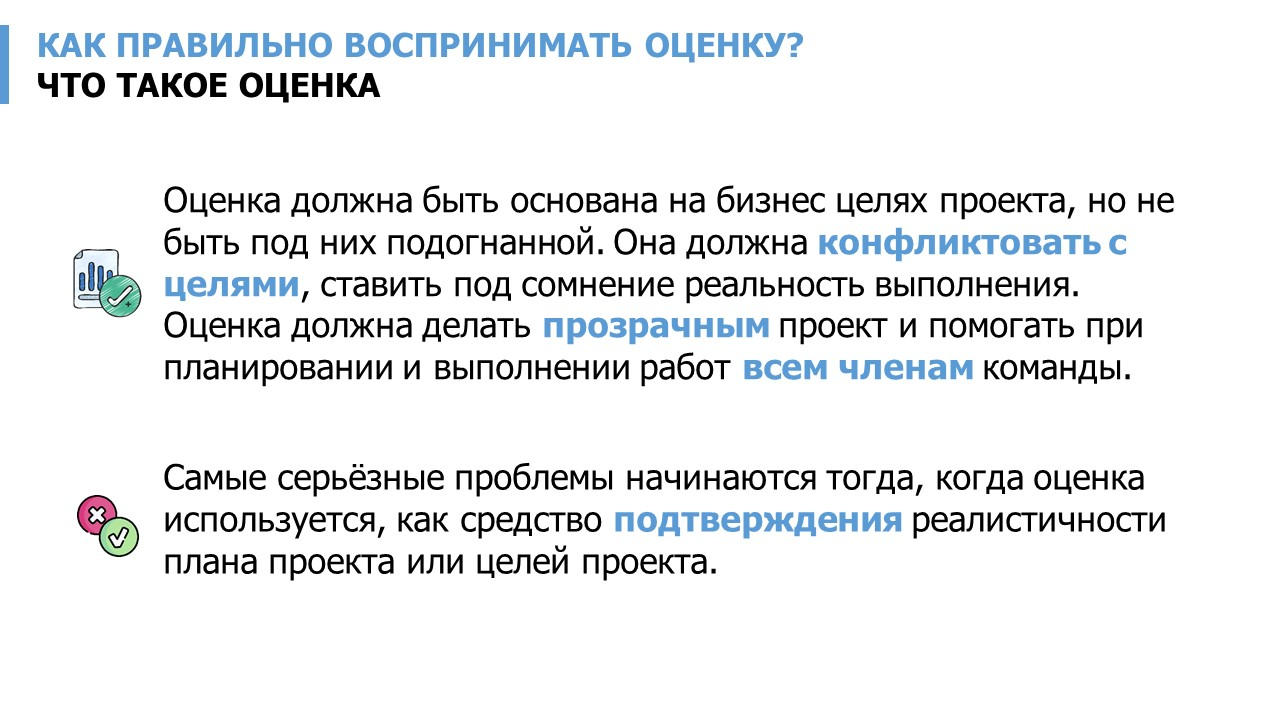
**Стоимостная** **оценка** - это процесс установления **стоимости** ресурсов **проекта**, основанный на определенных фактах и допущениях. Для определения **стоимостной** **оценки** прежде всего необходимо определить операции (пакет операций), длительность операций и требуемые ресурсы. Процесс **оценки** и его результат в значительной степени зависят от точности описания содержания, качества доступной информации, от стадии **проекта**.

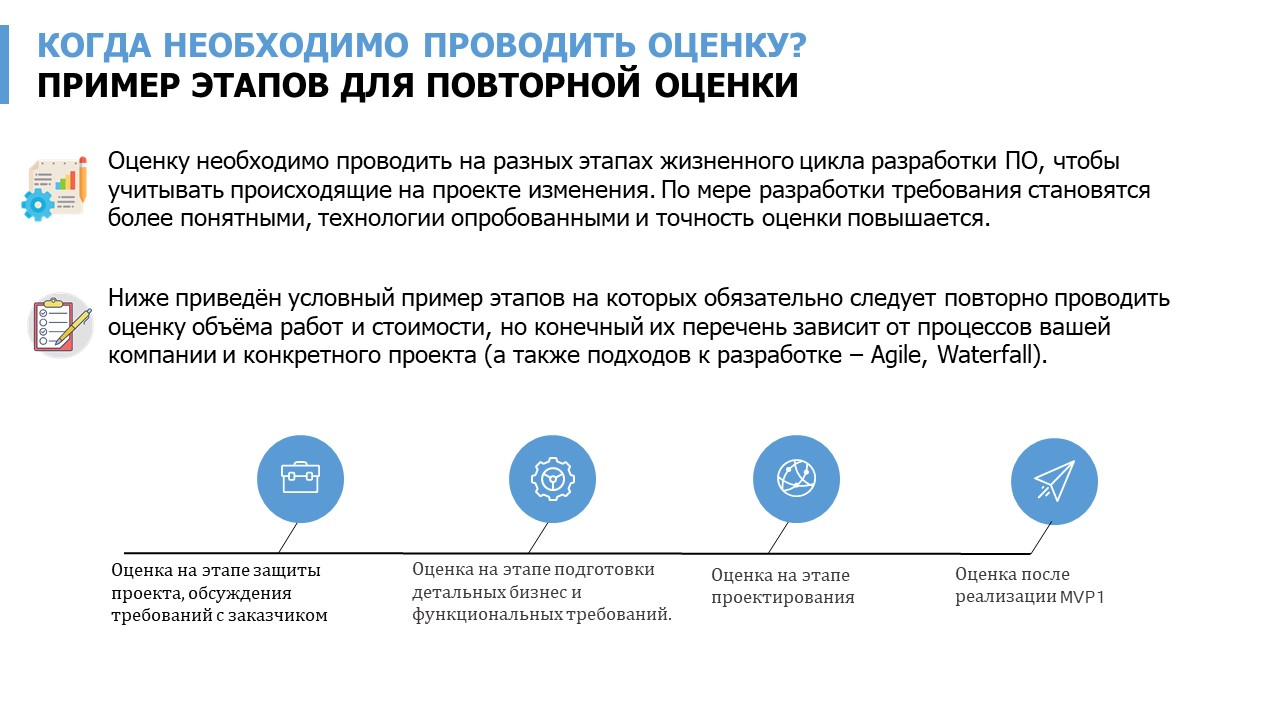
## 1. Базовая теория

Перед тем, как начать рассматривать методы оценки проектов по разработке систем, стоит ответить на вопрос, что из себя представляет оценка. Дело в том, что большинство специалистов заблуждаются. Путают оценку с планированием и подгоняют её под бизнес-цели.

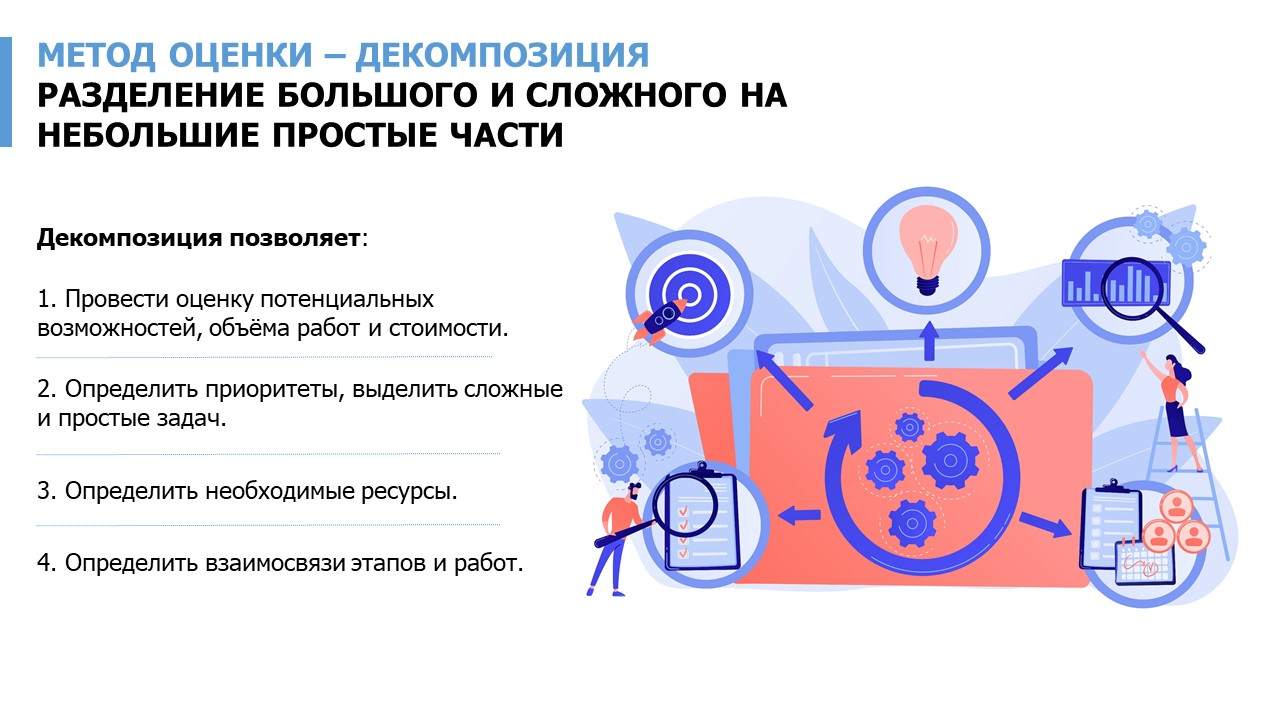
Данная путаница (заблуждение) не позволяет оценивать проекты разработки так, чтобы они были успешными и учитывали реальную ситуацию.



Очень важно пересматривать оценку по несколько раз. С каждым пройденным этапам жизненного цикла программного продукта (и этапов проекта) проводить новую итерацию оценки, при этом точность будет повышаться (понимаю, что это очевидное утверждение, но часто оценка проводится только на старте, а дальнейшее управление проектом происходит, как реагирование на возникаюющие изменения, а оценка, как инструмент уже не используется).



## 2. Метод оценки - Декомпозиция



Метод декомпозиции подразумевает, что составляется верхнеуровневый список задач по созданию ИТ-продукта, который детализируется настолько, насколько это возможно. Чем детальнее будет выполнена декомпозиция, тем более точной будет оценка. Есть много хорошего материала, как правильно делать декомпозицию, в этой статье я постараюсь передать именно суть метода при проведении оценок.

Конечно, на начальных этапах проекта данный метод будет менее точным, чем на последующих.

**Ниже приведён пример таблицы для декомпозиции задач:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Задача** | **Ресурс** | **Объём (ч/ч)** | **Стоимость за ед. ресурса (т.р.)** | **Способ расчёта стоимости за ед.** | **Вид затрат** |
| 1 | Подготовка требований | Аналитик | 120 | 1.2 | Средняя стоимость часа работы аналитика по компании или рынку | Основная и дополнительная заработная плата сотрудников |
| 1.2. | ... |  |  |  |  |  |

**В неё можно добавить дополнительные аналитики, например:**

* грейд сотрудника (junior, middle, senior - т.к. от грейда будет зависеть ставка).
* Локация сотрудника (Москва, СПБ, Екатеринбург и т.д.)
* Подразделение сотрудника
* Риски (описание возможных рисков)
* Сложность задачи
* Важность задачи
* и т.д.

Приведённая выше таблица не является исчерпывающей. Вы можете добавлять в неё любое количество аналитик, которое потребуется. Я специально не использую термин ИСР (Иерархическая структура работ) или WBS (Work Breakdown Structure) т.к. для большинства специалистов он ассоциируется с планированием работ, хотя на самом деле инструмент многоуровневой декомпозиции нужен именно для оценки. Он не включает в себя календарное планирование (календарные сроки выполнения работ). Для календарного планирования правильно использовать другой инструмент - план-график проекта. К тому же, если мы говорим про гибкие подходы к разработке, то детальных план-графиков может не быть, планирование происходит по другому (спринты, канбан и т.д., другие словами итеративное планирование и выполнение задач), при этом метод декомпозиции всегда будет актуален.

При декомпозиции задач я заранее стараюсь их классифицировать по видам затрат, в дальнейшем это поможет правильно учесть расходы в бюджете компании и посчитать себестоимость.

**Упрощённый список основных видов затрат для ИТ-проектов (для примера):**

|  |
| --- |
| Основная и дополнительная заработная плата сотрудников |
| Стоимость электроэнергии, потребляемой комплексом вычислительной техники |
| Стоимость вспомогательных материалов |
| Затраты, связанные с арендой помещения |
| Затраты на оборудование (ЦОД) |
| Затраты на текущий и профилактический ремонт |
| Затраты на сетевое управление |
| Затраты на управление системой |
| Затраты на оборудование сотрудников |
| Амортизационные отчисления |

Ключевой вопрос при декомпозиции - как сильно надо детализировать? т.е. на насколько мелкие задачи необходимо разбивать? На старте проекта важно учесть все задачи, которые влияют на scope (содержание) проекта, при этом понимая, что со временем требования будут детализироваться и оценку потребуется уточнять, сверять с бизнес-целями.

На картинке ниже продемонстрировано, как снижается неопределённость по мере детализации требований.

### 3. Метод оценки - Расчёт по метрикам и сравнение исторических данных

  
Суть данного метода в использовании всех имеющихся данных для расчёта стоимости разработки и проведения оценки. Оценка происходит на основании значений разных метрик, которые умеет считать компания.

**Метрика**- это сформулированная единица измерения, позволяющая получить численное значение определённого свойства ИТ-процесса, ИТ-проекта или продукта. Т.е. с помощью метрик мы измеряем процесс, проект или продукт в целом. На самом деле метрики можно использовать на группу связанных процессов, на портфель проектов или схожие продукты. Важнее всего, чтобы метрика была обоснованной и полезной, иначе её применимость может быть под вопросом.

**Примеры метрик, которые могут быть использованы для оценки объёма работ и стоимости разработки:**

|  |  |
| --- | --- |
| **Категория** | **Метрика** |
| Бизнес-требования | Среднее количество часов на подготовку бизнес-требований |
| Сценарии использования (Use cases) | Среднее количество часов на подготовку бизнес-требований |
| Истории пользователей (user story) | Среднее количество часов на подготовку истории пользователей |
| Проектирование (архитектура) | Среднее количество часов на подготовку архитектуры |
| Реализация | Среднее количество часов на реализацию одной задачи средней сложности (низкой, высокой) |
| Запросы на изменение | Среднее количество часов на реализацию запроса на изменение |
| Дефекты | Среднее количество часов на дефект |
| Процесс разработки | Sprint Goals Delivery – % достижения целей спринта (соответствие плана спринта/ выполненному факту) |
| Процесс разработки | Team interaction – вовлечённость команды разработки в продукт на ретроспективе, встречах, планировании спринтов, задач (опрос и оценка руководителя) |
| Эффективность | % соответствия реализации функционала ожиданиям заказчика, ожиданиям рынка |
| Эффективность | % соответствия реализации функционала бизнес-требованиям заказчика в рамках запроса на изменение |

Это явно не исчерпывающий перечень. Показателей крайне много и самое главное разумно их использовать. Конечно, собирать вручную данную информацию невозможно, для этого есть специальные ITSM системы, трекеры задач и системы управления разработкой, которые обладают специальный функционалом.

При использовании метода расчёта по метрикам очень важно понимать какие проекты сравнивать. Многое зависит от языка разработки и стека технологий. Например, надо очень осторожно сравнивать показатели из проекта на Платформе 1С и проектов на open source технологиях, даже если бизнес задача идентичная (что маловероятно, но для простоты примера). А вот если сравнивать проекты на базе 1С с другими проектами на 1С (при условии схожести задачи), тогда точность оценки на базе метрик будет высокой.

Мне кажется важным ещё раз подчеркнуть, что использовать метрики необходимо осторожно и не всегда на базе них можно сделать корректную оценку, важно учитывать особенности заказчика, особенности команды и т.д. Плюс в больших компаниях даже среднее время на реализацию одной задачи может считаться по разному в разных проектах.

Сравнивать показатели можно не только по проектам внутри компании, но и по отрасли в целом (т.е. с другими схожими компаниями). Есть известный термин - [бенчмаркинг](https://ru.wikipedia.org/wiki/%D0%91%D0%B5%D0%BD%D1%87%D0%BC%D0%B0%D1%80%D0%BA%D0%B8%D0%BD%D0%B3). По сути это процесс сравнения собственных показателей с их значениями у других компаний с целью выявления возможных зон роста и развития. Обычно, в бенчмаркинге сравнивают финансовые показатели компании (часто посредниками являются консалтинговые фирмы), но на разных ИТ конференциях тоже можно пообщаться с коллегами и обменяться опытом.

## 3.1. Определение размера программного продукта

Для проведения корректных оценок на основании исторических данных надо уметь определять размер программного продукта. Для этого существуют два виды показателей:

**Системные метрики, например:**

1. Модули системы (если они системно разделены)
2. Веб-страницы, пользовательские окна (интерфейс)
3. Классы
4. Компоненты системы
5. Таблицы базы данных
6. Строки кода (LOC)

**Функциональные метрики, например:**

1. Функции системы
2. Систематизированные требования к системе (если у требований есть чёткая классификация)
3. Сценарии использования (*Use Case*)
4. Пользовательские истории (user story)
5. Количество запросов от пользователей
6. Количество запросов по API

Функциональные метрики (функционально-ориентированные метрики) позволяют измерять программный продукт, процесс его разработки с помощью классификации и разделения функционала.

Системные метрики (размерно-ориентированные метрики) позволяют достаточно точно измерить программный продукт и процесс его разработки с помощью системных артефактов. Самый популярный и всем известный пример размерно-ориентированной метрики это - LOC (Lines Of Code) количество строк кода в программном продукте.

Для разных программных продуктов будет правильно использовать разные системные и функциональные метрики, например для относительно простых сайтов подойдут - веб страницы. Для систем, которые чётко разделены на разные функции - функции системы. При этом стоит использовать несколько метрик сразу (например, таблицы БД, строчки кода и функции).

Ниже приведён достаточно простой пример определения средней стоимости, качества и производительности на основании количества строк кода.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Проект | Затраты чел/мес | Стоимость (млн. руб.) | KLOC | Документация, страницы | Ошибки | Команда, чел. |
| № 1 | 4 | 12 млн. | 11,2 | 424 | 24 | 12 |
| № 2 | 7 | 20 млн. | 18,5 | 680 | 67 | 10 |
| № 3 | 10 | 22 млн. | 23,1 | 560 | 83 | 15 |
| № 4 | 2 | 8 млн. | 4,5 | 240 | 10 | 10 |
| № 5 | 12 | 28 млн. | 25,5 | 928 | 45 | 10 |

Таблица содержит данные о проектах за последние несколько лет и на основании неё можно определить размерно-ориентированные метрики производительности и качества (для каждого проекта) с помощью формул ниже. Это позволит использовать эти показатели для проведения оценок и сравнения с новым проектом.

Производительность = Длина тысяч LOC / затраты чел.мес

Качество = Кол-во ошибок / Длина тысяч LOC

Средняя стоимость = Стоимость (млн. руб.) / Длина тысяч LOC

Документированность = Кол-во страниц документации / Длина тысяч LOC

Системные метрики имеют свои недостатки. Они очень зависимы от языка и стека технологий. Сравнить совершенно разные программные продукты бессмысленно.

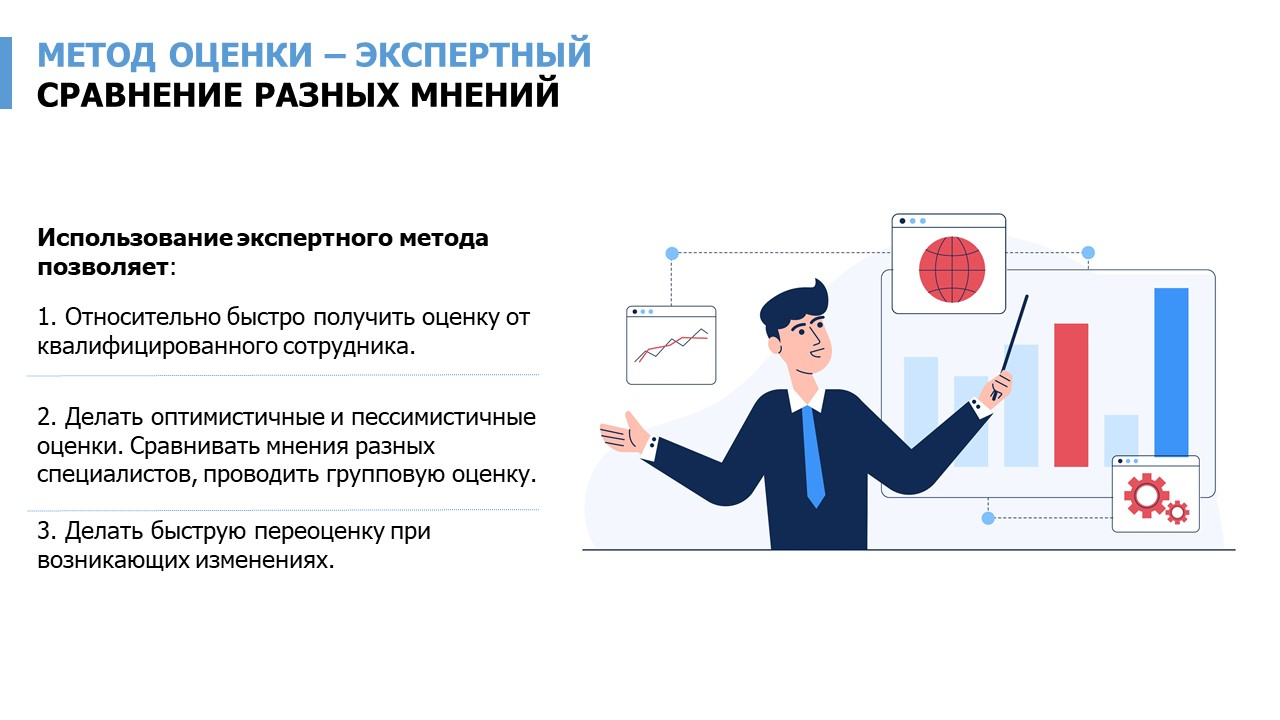
## 3.2. Использование поправочных коэффициентов

Поправочные коэффициенты следует использовать, чтобы учесть разные факторы влияющие на процесс разработки. Данный метод подойдёт на ранних этапах, когда высокий уровень неопределённости. Поправочный коэффициент можно использовать, как для стоимости, так и для оценки объёма работ (всё зависит от перечня факторов).

Ниже приведён пример списка возможных факторов и итоговый поправочный коэффициент (который посчитан, как среднее значение). Но метод расчёта коэффициентов может отличаться и завесить от степени влияния самого фактора, а также других условий. Для каждого отдельного проекта разработки следует рассчитывать собственные, уникальные поправочные коэффициенты.

|  |  |  |
| --- | --- | --- |
| **Описание** **фактора** | **Оценка фактора** | **Коэффициент** |
| Требуемая надежность | Критично | 1,3 |
| Требуемое качество продукта | Высокое | 1,2 |
| Сложность продукта | Крайне высокая | 1,3 |
| Документирование жизненного цикла | Не обязательно | 0,7 |
| Возможности аналитика | Низкие, используются сотрудники начального уровня | 1,2 |
| Опыт работы с приложением | Низкий | 1,1 |
| Опыт работы со стеком | Низкий | 1,2 |
| Непрерывность персонала | Низкий | 1,2 |
| **Среднее значение** |  | 1,15 |

## 4. Метод оценки - Экспертный



Не существует такого эксперта, который сможет корректно оценить проект в целом, поэтому метод экспертной оценки подразумевает оценку каждой отдельной задачи соответствующим квалифицированным специалистом, а в дальнейшем суммирование всех этих оценок для получения объёма работ, ч/ч и стоимости.

Есть несколько основных правил использования данного метода:

1. Важно сравнить мнения разных экспертов для одной и той же задачи, проводить групповые обсуждения.
2. Существенную пользу принесёт подготовка чек листов для всех участников. Это поможет не упустить важные моменты и повысить точность. Чек листы будут очень сильно отличаться в зависимости от типа задачи, поэтому следует попросить экспертов самостоятельно составить их.
3. Самой важной частью данного метода является сравнение оценок с фактическими результатами и поиск способ повышения точности, добавление нового пункта в чек лист или необходимость консультации дополнительного эксперта.

В экспертной оценке часто используют метод PERT (Program Evaluation Review Technique). Идея PERT заключается в определение ожидаемой оценки на основании оптимистичного, среднего и пессимистичного сроков выполнения задачи. Вот, как это выглядит:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Задача** | Оптимистично (ч/ч) | Наиболее вероятно (ч/ч) | Пессимистично (ч/ч) | Расчёт PERT (ожидаемый вариант) |
| Разработка экранной формы приёма платежей для ФЛ | 3 | 6 | 12 | 6.5 |

**Формула для расчёта:**

PERT = (Оптимистично + (4 \* Наиболее вероятно) + Пессимистично)/6

## 5. Другие модели и подходы к оценке

В данной статье я затронул те методы оценки, которые сам использую, но данный перечень не является исчерпывающим. Ниже представил более общую информацию о других методах оценки проектов разработки ПО, которая может вам приходится.

**5.1. Косвенные оценки**

Данный метод предполагает использование связанных показателей для оценки задачи. Например, у вас есть информация, что один тестовый сценарий на аналогичном проекте выполняется за 2 часа, но нет понимания, сколько этих тестовых сценариев будет. Самый простой способ, это посчитать их относительно количества сформулированных требований (например, user story) или количества запросов на изменение.

Объём дополнительного времени на разработку можно посчитать с учётом времени, которое было затрачено на исправление дефектов, выполнение изменений по уже реализованному функционалу.

**5.2. Специальные системы для проведения оценок**

На рынке есть специализированные системы для оценки проектов разработки, которые используют сложные математические формулы на основе данных, которые сможет предоставить компания.

**5.3.** [**Покер планирование**](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BA%D0%B5%D1%80_%D0%BF%D0%BB%D0%B0%D0%BD%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) **(или Scrum Poker)**

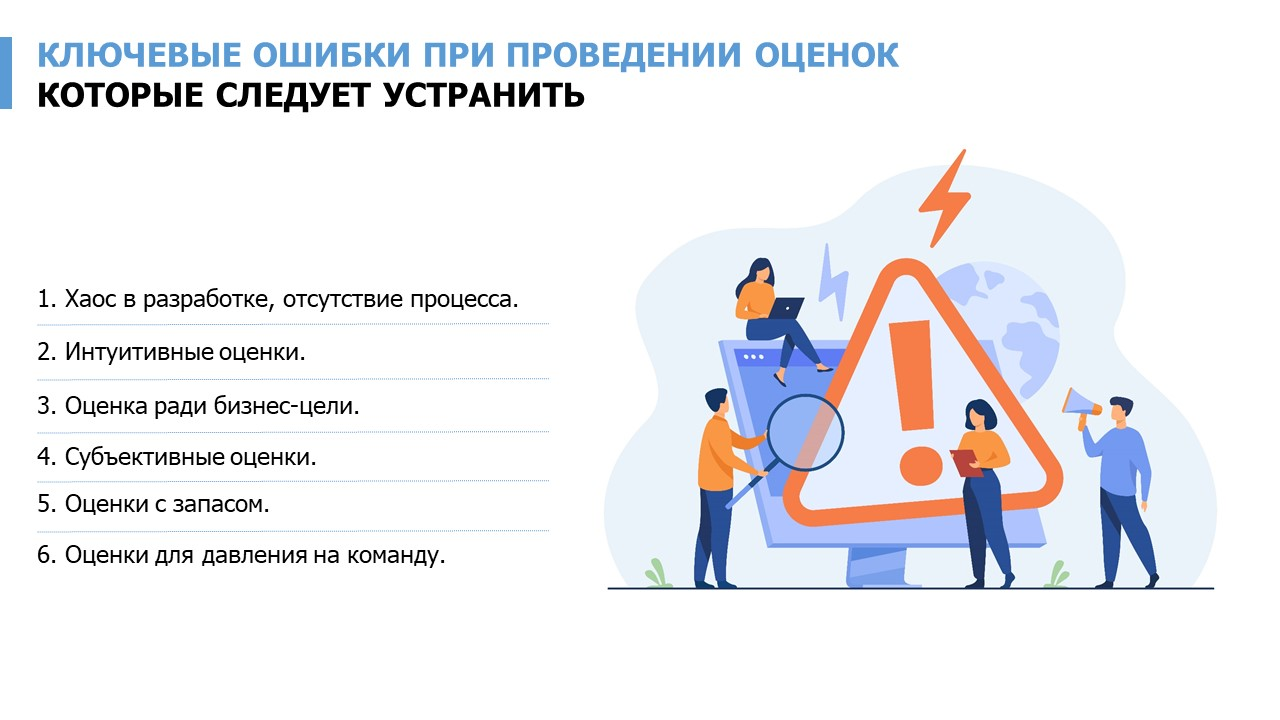
Это разновидность метода экспертной оценки, подходит для не очень больших проектов с маленькими командами. Является одной из самых популярных техник оценки в проектном менеджменте, в основана на достижении договорённости по объему работ и временных затрат между участниками команды.

**5.4. Модели оценки (стандарты)**

Есть несколько, если можно так выразится, стандартов по проведению оценки стоимости проектов по разработке ПО:

* Модель Oracle AIM для оценки трудоёмкости. Она основании на полном и точном знании процесса, а также крайне глубокой декомпозиции.
* Модель СОСОМО (и СОСОМО II) - сложная модель оценки разработанная ещё в 70-х года, которая устанавливает соответствие между размером системы в тысячах условных строк кода, типом проекта и трудоемкостью разработки системы.
* Немного более современные модели IFPUG FPA и FPA mkII, в них были введены понятия транзакции, а также репозитория кода.
* ГОСТы:
* ГОСТ Р 58302-2018 управление стоимостью жизненного цикла. Номенклатура показателей для оценивания стоимостижизненного цикла изделия.
* ГОСТ Р 56136-2014 управление жизненным циклом продукции военного назначения. Термины и определения
* ГОСТ Р ИСО/МЭК 25021-2014 Информационные технологии. Системная и программная инженерия. Требования и оценка качества систем и программного обеспечения(square).

## 6. Ключевые ошибки при проведении оценок

  
**Вредные оценки.** Использование оценок для давление на команду, а не как инструмент повышения точности планирования и реализации бизнес-целей.

**Оценки с запасом.** Часто сотрудники берут немного времени "про запас" т.е. сверху и дают завышенную оценку.

**Интуитивные оценки.** Лучше потратить 10 минут на анализ задачи, её детализацию и уточнение, а потом уже давать оценку. Интуитивные оценки чаще всего ведут к нарушению сроков и увеличению стоимости.

**Оценка ради бизнес-цели.** Когда оценку даёт человек, который заинтересован в выполнении бизнес-цели. Желание выполнить поставленную цель не является плохим явлением, но очень важно давать объективные оценки (особенно, если это касается заказчика или вывода продукта на рынок).

**Субъективные оценки.** Следует избегать исключительно субъективных оценок, ведь разработчики чаще дают оптимистические оценки, а менеджеры проектов наоборот, пессимистические (так сказать закладывают риски). Вместо этого лучше использовать работу над ведением рисков и доведение их до стейкхолдеров, делать планы реагирования.

**Косвенные затраты и задачи.** Не весь объём задач по разработке системы учитывается при оценке. Часто это бывает из за недостаточности проработанной бизнес модели, бизнес требований или отсутствие проектирования.

**Хаос в разработке, отсутствие процесса.** Ключевые проблемы (это не полный перечень, уверен, что его можно дополнить):

* Отказ от полноценного планирования из за давление со стороны заказчика или руководства. Под полноценным я понимаю тот уровень, который подойдёт для достижения целей и эффективной работы команды.
* Отсутствие коммуникаций внутри команды (отсутствие дейли или другие митингов для синхронизации команды, наличие психологических барьеров в общении участников команды).
* Отсутствие стандартизированного процесса разработки (когда все понимают, какой порядок работы в JIRA, Confluence и как задача переходит из аналитики в проектирование, а из проектирования в разработку и т.д.).
* Упрощённый и недостаточный анализ требований.
* Отсутствие вовлечённости бизнес пользователей и заказчика.
* Упрощённое проектирование и отсутствие подходов к разработке, качеству написания кода.
* Отсутствие подходов к тестированию или тестирования в целом.

## 7. Итог

Надеюсь, что данный материал оказался вам полезными и вы сможете его применить в своей работе. Методы оценки разделены, чтобы лучше понять их, но конечно использоваться могут все одновременно.

Цель статьи, это дать ключевую информацию по теме проведения оценок разработки программных продуктов, чтобы читатель смог сформировать общее представление.

Буду благодарен конструктивной критике и буду рад, если поделитесь своими методами проведения оценок, которые не учтены в статье. Спасибо.

**Теги:**

* [оценка стоимости проекта](https://habr.com/ru/search/?target_type=posts&order=relevance&q=%5B%D0%BE%D1%86%D0%B5%D0%BD%D0%BA%D0%B0%20%D1%81%D1%82%D0%BE%D0%B8%D0%BC%D0%BE%D1%81%D1%82%D0%B8%20%D0%BF%D1%80%D0%BE%D0%B5%D0%BA%D1%82%D0%B0%5D)
* [управление проектами](https://habr.com/ru/search/?target_type=posts&order=relevance&q=%5B%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5%20%D0%BF%D1%80%D0%BE%D0%B5%D0%BA%D1%82%D0%B0%D0%BC%D0%B8%5D)
* [управление разработкой](https://habr.com/ru/search/?target_type=posts&order=relevance&q=%5B%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5%20%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%BE%D0%B9%5D)
* [оценка сроков](https://habr.com/ru/search/?target_type=posts&order=relevance&q=%5B%D0%BE%D1%86%D0%B5%D0%BD%D0%BA%D0%B0%20%D1%81%D1%80%D0%BE%D0%BA%D0%BE%D0%B2%5D)
* [оценка трудозатрат](https://habr.com/ru/search/?target_type=posts&order=relevance&q=%5B%D0%BE%D1%86%D0%B5%D0%BD%D0%BA%D0%B0%20%D1%82%D1%80%D1%83%D0%B4%D0%BE%D0%B7%D0%B0%D1%82%D1%80%D0%B0%D1%82%5D)
* [оценка задачи](https://habr.com/ru/search/?target_type=posts&order=relevance&q=%5B%D0%BE%D1%86%D0%B5%D0%BD%D0%BA%D0%B0%20%D0%B7%D0%B0%D0%B4%D0%B0%D1%87%D0%B8%5D)
* [управление изменениями](https://habr.com/ru/search/?target_type=posts&order=relevance&q=%5B%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5%20%D0%B8%D0%B7%D0%BC%D0%B5%D0%BD%D0%B5%D0%BD%D0%B8%D1%8F%D0%BC%D0%B8%5D)
* [стоимость разработки](https://habr.com/ru/search/?target_type=posts&order=relevance&q=%5B%D1%81%D1%82%D0%BE%D0%B8%D0%BC%D0%BE%D1%81%D1%82%D1%8C%20%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8%5D)

**Хабы:**

* [Анализ и проектирование систем](https://habr.com/ru/hubs/analysis_design/)
* [Управление разработкой](https://habr.com/ru/hubs/dev_management/)
* [Управление проектами](https://habr.com/ru/hubs/pm/)
* [Управление продуктом](https://habr.com/ru/hubs/productpm/)