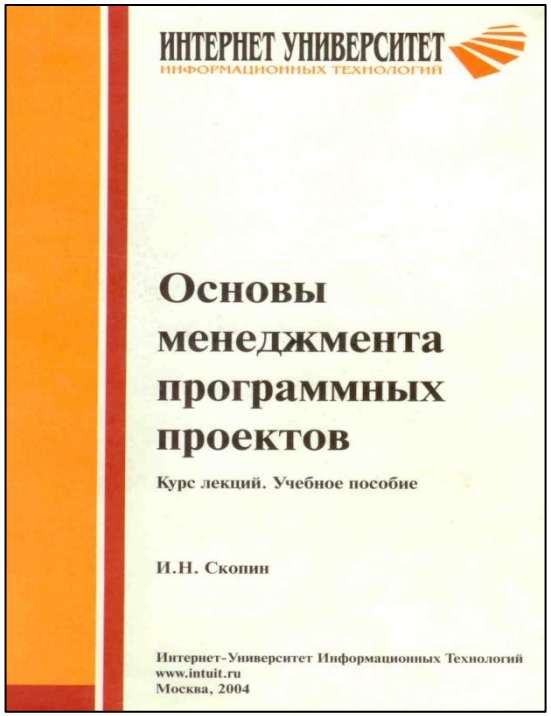
Менеджмент разработки программных изделий
(руководство командой и управление проектом)



И.Н. Скопин

Основы менеджмента программных проектов. Курс лекций.
Учебное пособие // М.: ИНТУИТ.РУ «Интернет-Университет Информационных технологий», — 2004. — 363 с.

http://www.intuit.ru

Содержание

1. Введение. Основные понятия. Функции и роли разработчиков программных проектов. Ключевые роли. Подбор кадров. Принцип осмысленности действий
2. Принципы построения системы деятельностей программного проекта
3. Жизненный цикл программного обеспечения и его модели
4. Производственные функции в моделировании жизненного цикла: модель фазы — функции
5. Моделирование жизненного цикла объектно-ориентированных программных проектов
6. Технологические аспекты развития программных систем в моделях жизненного цикла
7. Особенности первой итерации объектно-ориентированного программного проекта
8. Жизненный цикл в методологиях быстрого развития проектов
9. Проблемы оперирования требованиями
10. Методическая поддержка оперирования требованиями. Примеры: работа с резюме, прием на работу
11. Результативность программистской проектной деятельности
12. Управление рисками
13. Организация коллективной работы
14. Взаимодействия разработчиков проекта
15. Конфликты в проектном коллективе
16. Планирование и контроль развития проекта

1. Введение. Основные понятия. Функции и роли разработчиков программных проектов. Ключевые роли. Подбор кадров

Руководство и управление в проектной деятельности

- Руководить можно людьми
- Управлять можно проектом

Менеджмент должен сочетать и то и другое

Эта двойственность характерна для любого менеджмента, но для менеджмента программных проектов она играет решающую роль, поскольку

- В этой отрасли производятся нематериальные продукты — ***артефакты***
- Это ***творческая деятельность*** в большей степени, нежели технологический процесс

Три схемы организации менеджмента проекта

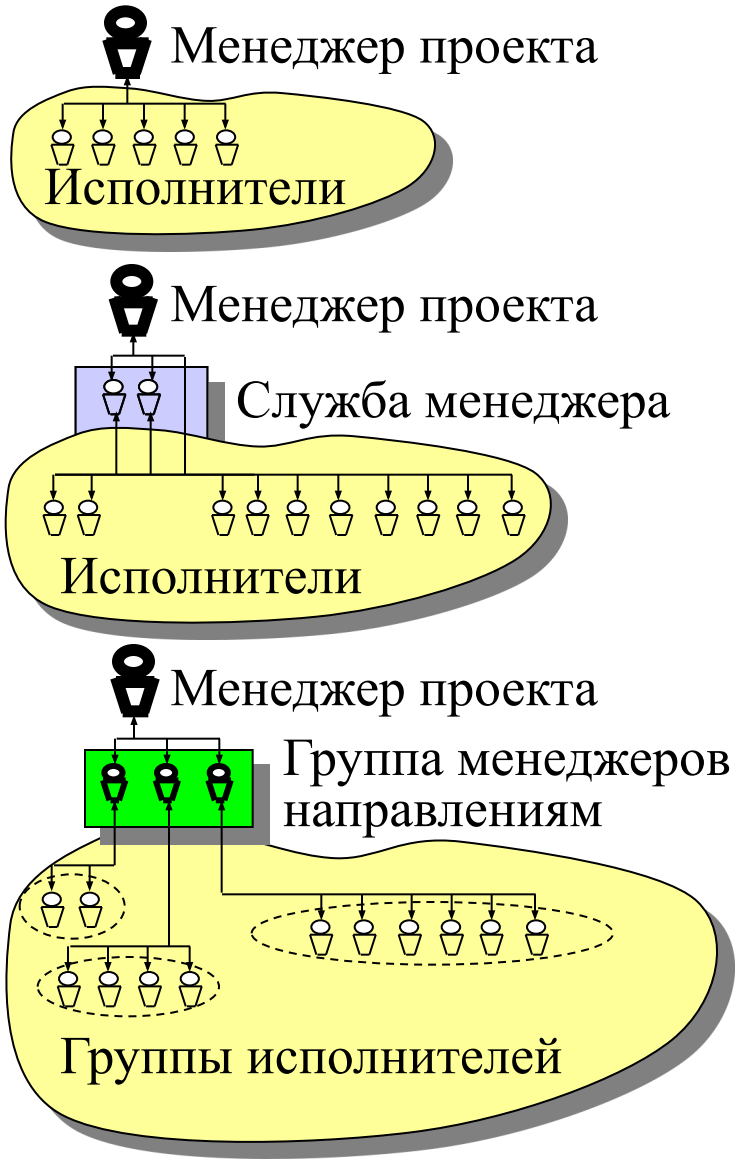


Схема с одним менеджером

Схема со службой менеджера

Схема с группой менеджеров по направлениям

Зависимость от масштаба проекта. Другие варианты схем

Разработка программного обеспечения —

коллективный труд специалистов, направленный на удовлетворение потребности пользователей в *автоматизации их деятельности* с помощью применения создаваемой программной системы.

Проектная деятельность и ее исполнители

- **Проект нацелен на удовлетворение потребности автоматизации некоторой пользовательской деятельности ⇨ *проектная деятельность* включена в систему деятельностей. В какую систему? Неполный ответ:**
  - **– Автоматизируемая пользовательская деятельность**
  - **+ *Изучение пользовательской потребности***
  - **– Поставка пользователю оборудования и программного продукта**
  - **– *Обучение и поддержка пользователя***
  - **+ *Разработка системы* (реализация, разработка проекта)**
  - **+ *Планирование и управление проектом***

Обязательные составляющие проектной деятельности отмечены

Вопрос к слушателям:

Вопрос к слушателям:

2. А как еще могут быть связаны деятельности?

Вопрос к слушателям:

3. Раскройте самостоятельно «многое другое»

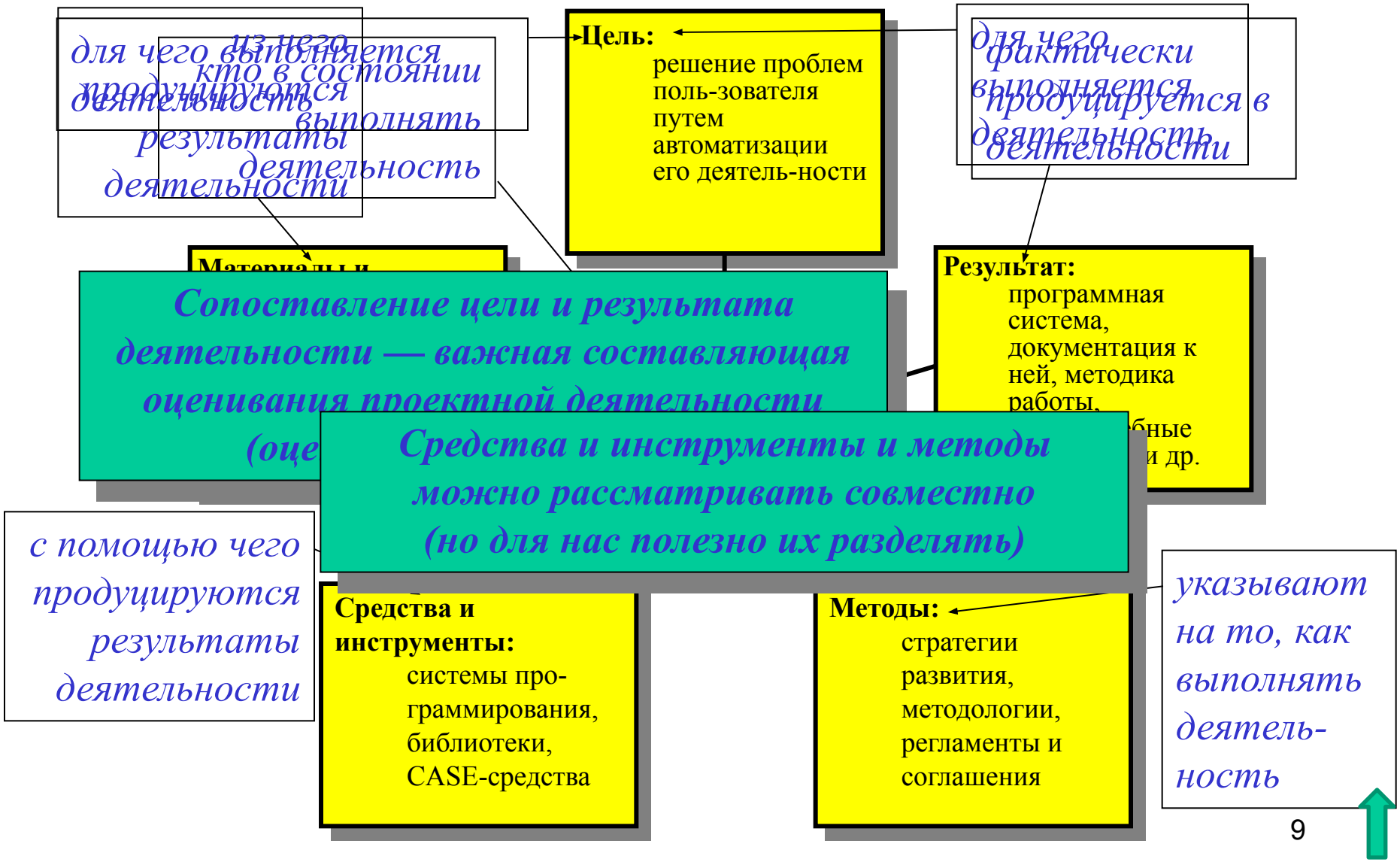
Вопрос к слушателям:

4. Являются ли производственными функциями
   - деятельность «Обед исполнителя»?
   - деятельность уборщицы?

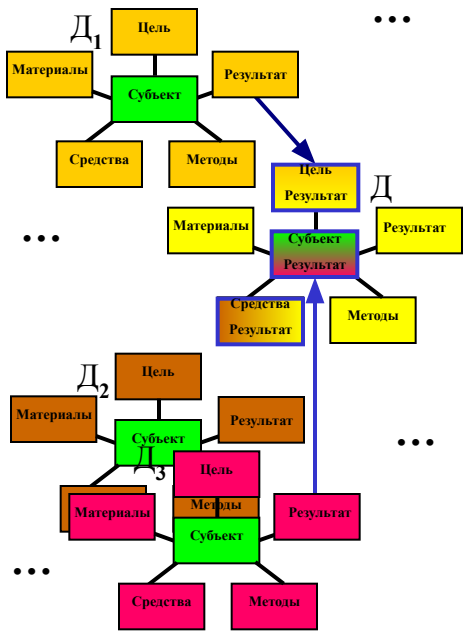
Декомпозиция проектной деятельности

- ***Проект — большая производственная функция***, **выполнение которой требует осуществления многих различных деятельностей.**
- **Деятельности проекта взаимосвязаны, нуждаются в планировании, обеспечении ресурсами и др. От существования многих из них приходится абстрагироваться (несущественные?)**
- ***Организованность*** **совокупности проектных и внешних (косвенно связанных с проектом) деятельностей должна быть достигнута!**
- **Для построения нужной организованности применяются методологии развития проектов. Среди прочего они решают задачу *декомпозиции*.**
- **Это одно из назначений методологии. А какие они бывают?**
- **Производственные функции и исполнители как декомпозируемые сущности**
  - **можно *структурировать выполнение функции*, разбивая ее на составляющие, определяя назначение каждой из составляющих и связи между ними так, чтобы результат совместного выполнения совпадал с требуемым результатом разбиваемой функции**
  - **можно *структурировать обобщенного исполнителя*, иными словами, конкретизировать исполнителей, отвечающих за разные аспекты выполнения функции**
- **Оба разбиения допускают продолжение в глубину:**
  - **для исполнителей — до конкретных индивидуумов**

Элементы деятельностей



Системы деятельностей



Любая деятельность есть часть некоторой ***общей системы деятельностей***, охватывающей группу субъектов-исполнителей. Деятельности, субъекты которых не попадают в выделенную группу, являются ***окружением данной системы***. Окружение связано с системой следующими способами:

- из окружения *поставляются элементы деятельностей системы*;
- деятельностям окружения *передаются результаты* деятельностей системы;
- система в целом и ее отдельные деятельности *являются элементами деятельностей* окружения

Вопросы к слушателям:

1. **А как это связано с исполнителями проекта?**
2. **В каких деятельностях они участвуют и в каком качестве?**

Соотнесите свои ответы с понятием роли

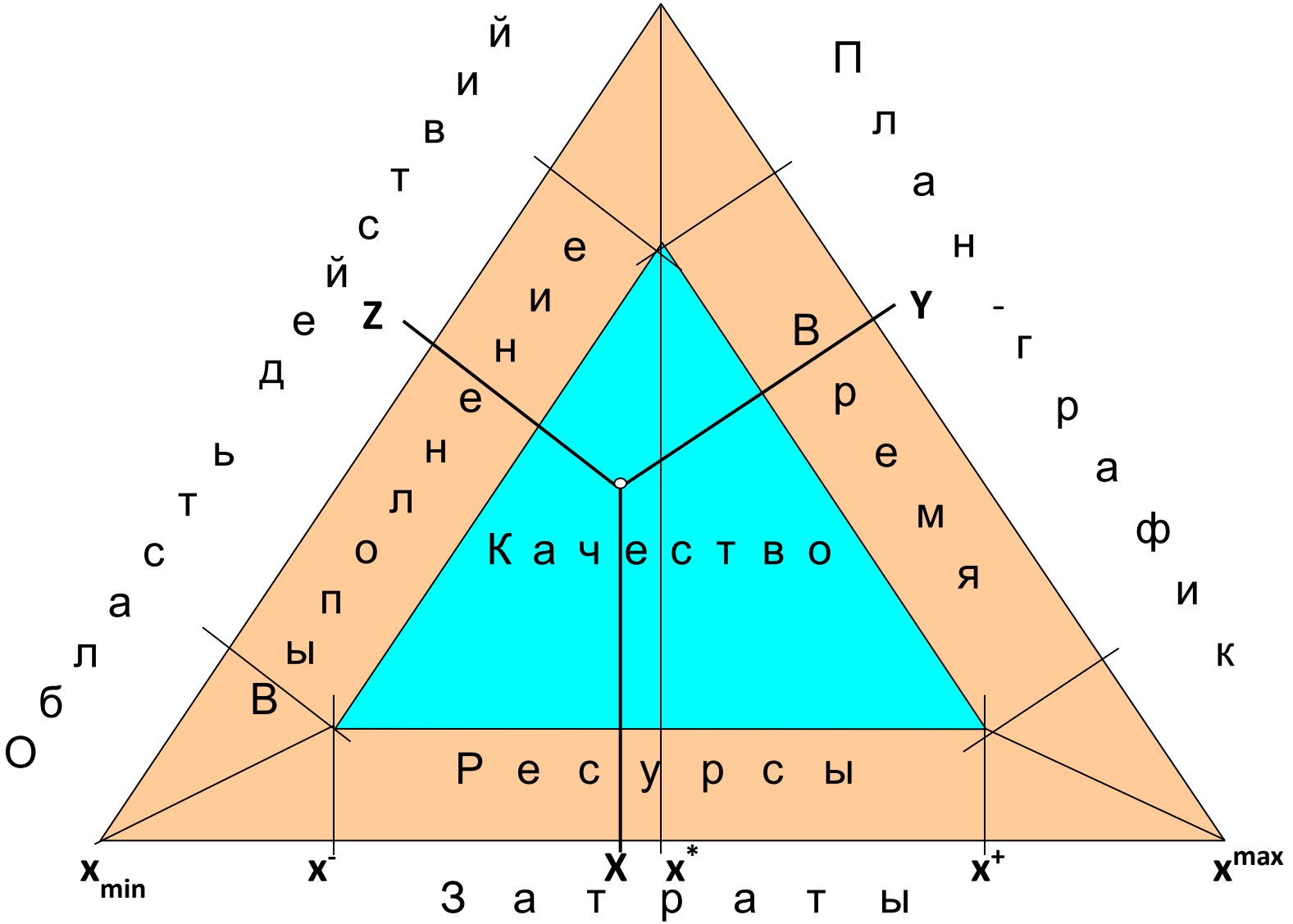
Деятельность менеджера и составляющие системы проектных деятельностей

- *Цель* — направление других проектных деятельностей так, чтобы они продвигали проект к выполнению (задаваемых вне системы) *работ* в условиях ограничений по *времени, финансам* и *качеству* = достижение целей деятельности, задающей проект в целом.
- *Согласование параметров проекта*: объем работ, сроки, запрашиваемые финансы (внешняя по отношению к работам проекта деятельность)
- Менеджмент проекта — обеспечение предоставления продукта для его использования, разработка которого
  - требует выполнения определенного *объема работ* — *область действия*,
  - использует *затраты* (в определенных пределах) — *ресурсы*,
  - старается укладываться в заданные рамки *времени* — *план-график работ*,

  и должна удовлетворять приемлемому уровню *качества*.
- Это хорошо известный *треугольник менеджмента «хорошо-быстро-дешево»*: из трех параметров выбери два — получишь третий!

Задание: 1. Конкретизировать элементы деятельности менеджера и их связи с другими деятельностями;
2. Сопоставить свой результат с полученным другими;
3. Объяснить различия.

Треугольник менеджмента проектов — иллюстративная модель



Другие подобные модели см. в книге

Несколько методических положений

- ***Делегирование полномочий*** — инструмент разделения труда (не только менеджера)
- ***Персонифицированная*** и ***деперсонифицированная*** ответственность
- ***Абстрактное действующее лицо*** и ***конкретный сотрудник***
- Виды ***деятельности***:
  - ***продукционная деятельность*** (производство результата, нужного для проекта)
  - ***управляющая деятельность*** (производство траектории развития)
  - ***наблюдательная деятельность*** (производство познавательного результата)
- Три варианта ***целей*** разработки программного
  - производство программ, прямо не связанное с получением дохода
  - производство рыночного продукта
  - производство программ под *заказ*

Роль заказчика, пусть даже лишь виртуального **очень значительна**!

- ***Главная и постоянная задача менеджмента проекта***: продвижение проекта к получению результатов, обозначенных в начале развития проекта как его цели

Функции, выполняемые разработчиками проекта

- *Типовые функции* (кодирование, анализ требований, тестирование, отладка и т.д.)
  - *Распределение функций* между разработчиками проекта → *роли* исполнителей (объединение родственных функций)
  - *Поручения* — разовые или систематические *задания*, из которых складываются *действия*, необходимые для выполнения функции
- *Технологичные функции* — такие, для которых
  - определен регламент выполнения (как последовательность заданий),
  - этот регламент не требует дополнительных разъяснений для исполнителя

(зависят от исполнителей, не путать с технологией!)

Функции подразделяются на

- *организационные* — создают условия для выполнения проектных заданий
- *производственные* — непосредственно связаны с выполнением этих заданий

Участники разработки и ***функциональные роли*** в коллективе разработчиков

Этапы развития проекта — жизненный цикл (программного изделия)

Задача менеджмента:

- **в части управления** – организационно-управленческая *деятельность*, поддерживающая процесс разработки программного изделия на всех этапах его жизненного цикла (возможность методов, методик, методологий и технологий)

Ролевые кластеры модели проектной группы MSF

Задача — повышение эффективности использования продукта.

Ключевая цель кластера — обеспечивать удовлетворение заказчика.

Для ее достижения кластер

- представление интересов заказчика,
- планирование продукта,
- маркетинг.

Функциональные роли в программном проекте

Вне проекта	• **Заказчик (Customer)**	— внешняя роль
	• **Планировщик ресурсов (Planner)**	— административная роль
Проектная группа (коллектив разработчиков)	• **Менеджер проекта (Project Manager)**	— руководитель проекта
	• **Руководитель команды (Team Leader)**	
	• **Архитектор (Architect)**	— проектировщики
	• **Проектировщик подсистемы (Designer)**	
	• **Разработчик (Developer)**	— разработчики
	• **Разработчик информационной поддержки (Information Developer)**	
	• **Специалист по пользовательскому интерфейсу (Human Factors Engineer)**	— эксперты
	• **Эксперт предметной области (Domain Expert)**	
	• **Тестировщик (Tester)**	— обслуживающий персонал
	• **Библиотекарь (Librarian)**	
	• ***Катализатор* (Catalyst)**	— *неявные функции*

Принципы, определяющие регламент совмещения ролей

- не следует допускать совмещение ролей, которые имеют конфликтные или противоречивые интересы;
- предоставление ролей с конфликтными интересами различным людям способствует равновесию противоречащих точек зрения;
- дополнительные роли для разработчиков всегда приводят к росту времени выполнения их основной работы;
- **если работнику поручается несколько ролей, то он всегда должен знать, какую из них он выполняет в данный момент**.

Совмещение ролей в программном проекте

Программист один разрабатывает проект для себя — предельный случай полного совмещения

Заказчик и планировщик с другими ролями — экзотика

- Менеджер и архитектор
- Менеджер и руководитель команды
- Руководитель команды и проектировщик п
- Менеджер и разработчик
- Различные разработчиков
- Разработчики документации (все работник
- Специалист по интерфейсу и менеджер
- Эксперт предметной области и менеджер
- Специалист по интерфейсу и эксперт предметной области
- Эксперт предметной области и разработчи
- Специалист по интерфейсу и разработчик
- Библиотекарь и один из разработчиков
- Тестировщики и другие члены команды

Игра

Давайте

- *угадаем*, что будет написано справа от указанных пар ролей;
- *посмотрим* и *сравним* с вашими ответами;
- *поспорим*. Соглашаться никто не обязан!

Сделаем выводы *самостоятельно*!

Лидер коллектива — один из работников ключевых ролей или сам менеджер

Ситуации, в которых действует менеджер при подборе кадров

• У менеджера есть коллектив потенциальных исполнителей, готовый приступить к работе над проектом;	Лидер есть
• Менеджерский коллектив потенциальных исполнителей недостаточен: среди его сотрудников не все обладают нужной квалификацией;	Лидер есть
• В поле зрения менеджера есть независимые потенциальные исполнители, из которых можно сформировать коллектив для работы над проектом;	Лидера нет
• Менеджеру приходится прибегать к найму рабочей силы со стороны.	Лидера нет

Задача поиска лидера

Решение задачи определения кадровых ресурсов проекта

Кадровые потребности проекта

Оценка распределения кадровых потребностей по времени

Возможности подбора кадров на проект

До официального начала выполнения проекта

График привлечения сотрудников к проекту

Критические ролевые позиции проекта

Утверждение кадровой политики проекта

Заполнение вакансий

По мере необходимости в ходе выполнения проекта

Задача определения кадровых ресурсов проекта никогда не может быть решена окончательно!

Тренинг: когда возможно, чтобы 1+1+1 > 3 или 1+1+1 < 3?

Ответ: Параллельно выполняемые работы

- Командная деятельность — лучший пример такой работы, При условии слаженности коллективных действий и эффективного их распараллеливания три исполнителя выполнят больше, чем они выполнили бы поодиночке, т.е. $1+1+1 > 3$,
  иначе $1+1+1 = 3$ или даже $1+1+1 < 3$
- Эффекту роста производительности способствуют кооперация и специализация (учет квалификации, склонностей и пр.)
- Для организации коллективной работы необходимы:
  - Определение функций, необходимых для выполнения работы
  - Определение ролей, назначаемых исполнителям
  - Правильное распределение ролей
  - Планирование деятельности
  - Четкое следование плану

Вычислительная машина на человеческой элементной базе

- *Задача* из истории: во Франции в связи с переходом на метрическую систему требовалось точно и как можно быстрее пересчитать по известным формулам артиллерийские поправки прицеливания на дальность, ветер и др.
  Условие: Много формул, типов оружия, результаты нужны как можно скорее
- Карно предложил использовать две роты солдат, каждому из которых поручено выполнять одну арифметическую операцию с аргументами, получаемыми от соседей, и передавать результат одному из них.
- В каждой роте было выделено по три группы солдат:
  1) для приема входных данных, 2) для счета, 3) для вывода
- Это *действительно* вычислительная машина (не арифмометр!)), у которой в «памяти» записана программа, каждый элемент «памяти» активен, когда появляются входные данные: распределенные вычисления, управление потоками данных, высокая степень параллелизма, защита от сбоев

Общая задача для каждой группы участников:

- Написать программу для проведения вычислительных расчетов (любую!)
- Выполнить два варианта расчетов с заданной точностью

Условия, правила и соглашения игры

Соревноваться будут N команд (~ 4 – 5 человек). N+1-ая команда – наблюдатели (следят за соблюдением правил, собирают материал для анализа, готовят итоговый отчет)

1. *5 минут*: Определить N лидеров. *Результат* – имена лидеров
2. *15 минут*: Лидер подбирает исполнителей. *Результат* – списки команд и их исполнителей
3. *10 минут*: Первичное определение ролей (любое!) и распределе-ние их между исполнителями в команде. *Результат* – N ролей
4. *5 минут*: **предъявление задачи**. *Результат* – понимание задания
5. *15 минут* + перерыв (*10 минут*): Коллективно написать программу решения предъявленной задачи. Откорректировать роли и их распределение. *Результат* – N откорректированных ролей и их распределения N программ
6. *< 25 минут*: **Соревнование на скорость**: проведение расчетов для двух комплектов входных данных. Неверные ответы – дисквалифи-кация команды. Наблюдатели анализируют процесс. *Результат*: N*2 комплектов полученных данных + список команд, упорядочен-ных по времени выполнения.
7. *20 минут*: Итоговый анализ результатов игры.

Задача для коллективного решения

- Любым способом решить систему уравнений с точность до двух знаков после запятой:

  $ax + by = e$

  $cx + dy = f$

  для трех комплектов входных данных (три последовательности, каждая из 6 чисел : $a_1\ b_1\ c_1\ d_1\ e_1\ f_1$, $a_2\ b_2\ c_2\ d_2\ e_2\ f_2$ и $a_3\ b_3\ c_3\ d_3\ e_3\ f_3$).
- Решение складывается из следующих этапов:
  1. ***Составление программы.*** Метод, алгоритм, язык, распределение вычислений по исполнителям, а также роли и распределение ролей выбираются коллективно и утверждаются лидером команды. *Время выполнения этапа ограничено*!
  2. ***Счет I.*** Получение данных ⇨ выполнение программы ⇨ передача результата наблюдателям. *Если результат неверен, команда выбывает из игры*
  3. ***Счет II.*** Получение данных ⇨ выполнение программы ⇨ передача результата наблюдателям. *Если результат неверен, команда выбывает из игры*
  4. ***Счет III.*** Получение данных ⇨ выполнение программы ⇨ передача результата наблюдателям. *Если результат неверен, команда выбывает из игры*
- При нарушении условий, правил и соглашений игры команда дисквалифицируется
- Оцениваются (критерий1 > критерий 2 > критерий 3):
  - суммарное время выполнения этапов 2 и 3 – **критерий 1** (основной);
  - степень загруженности исполнителей при выполнении этапов 2 и 3 – **критерий 2**;
  - качество определения ролей и их распределения (предварительного и окончательного) – **критерий 3**;

Задача для коллективного решения

- Любым способом решить систему уравнений с точность до двух знаков после запятой:

  $ax + by + cz = p \qquad dx + ey + fz = q \qquad gx + hy + iz = r$

  для входных данных, состоящих из трех последовательностей по 12 чисел:
  $a_1\ b_1\ c_1\ d_1\ e_1\ f_1\ g_1\ h_1\ i_1\ p_1\ q_1\ r_1$, $a_2\ b_2\ c_2\ d_2\ e_2\ f_2\ g_2\ h_2\ i_2\ p_2\ q_2\ r_2$, $a_3\ b_3\ c_3\ d_3\ e_3\ f_3\ g_3\ h_3\ i_3\ p_3\ q_3\ r_3$.
- Решение складывается из следующих этапов:
  1. ***Составление программы.*** Метод, алгоритм, язык, распределение вычислений по исполнителям, а также роли и распределение ролей выбираются коллективно и утверждаются лидером команды. *Время выполнения этапа ограничено*!
  2. ***Счет I.*** Получение данных ⇨ выполнение программы ⇨ передача результата наблюдателям. *Если результат неверен, команда выбывает из игры*
  3. ***Счет II.*** Получение данных ⇨ выполнение программы ⇨ передача результата наблюдателям. *Если результат неверен, команда выбывает из игры*
  4. ***Счет III.*** Получение данных ⇨ выполнение программы ⇨ передача результата наблюдателям. *Если результат неверен, команда выбывает из игры*
- При нарушении условий, правил и соглашений игры команда дисквалифицируется
- Оцениваются (критерий1 > критерий 2 > критерий 3):
  - суммарное время выполнения этапов 2 и 3 – **критерий 1** (основной);
  - степень загруженности исполнителей при выполнении этапов 2 и 3 – **критерий 2**;
  - качество определения ролей и их распределения (предварительного и окончательного) – **критерий 3**;

Принцип осмысленности действий

Принцип: всякий раз, когда работнику приходится выполнять какую-либо работу, он должен четко осознавать, зачем он это должен делать и делает

Почему он далеко не всегда выполняется?

Подмена целей

- Тот, от кого исходит предложение или приказ, скорее всего, осознает, зачем ему это нужно
- Но надо еще сопоставить свое «*зачем*» с системой целей и приоритетов работника, которая, очевидно, не совпадает с его целями.
- Предлагающий работу часто полагает, что это не так, и он может быть искренне убежден, что, раз уж работнику объяснили нечто, он это обязательно воспринял, и обязательно будет соблюдать предписанные правила.

Осмысленность и целенаправленность

- **Целенаправленность**: цель действий определена и известна работнику
- **Осмысленность**: соответствие целей задания (внешних) внутренним целям и установкам (индивидуальным). *Подмена целей — когда это не так.*

Мотивация к цели имеет косвенное отношение (может сформировать цель), но не обязательно соответствующую цели задания. Опасность вместо выполнения получить его имитацию.

Необходимы **приемы и методы** доведения заданий до разработчиков.

Нужно учитывать **варианты подмены целей**, включая как простое недопонимание, с одной стороны, а с другой — прямой, но невысказанный саботаж.

Требуются **особые подходы**. Это один из существенных аспектов квалификации.

Принцип осмысленности действий не сводится *только* к индивидуальным воздействиям:

- коллективное обсуждение,
- открытое распределение обязанностей и др.

} эффективно стимулируют осмысленность действий («на миру и смерть красна!»).

- **Но не в случае авторитарного и директивного руководства.**
- Осознанность – это, когда решения приняты на индивидуальном уровне, всеми участниками

Мы ратуем за консенсус в руководстве коллективом

2. Принципы построения системы деятельностей программного проекта

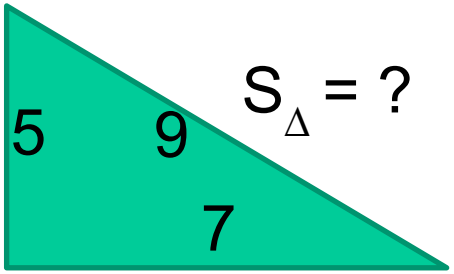
Эпиграф (тест)

- Требуется за одну минуту и предоставить решение следующей задачи
- Найти площадь прямоугольного треугольника со сторонами 5, 7 и 9 единиц длины
- Ответы: (1) 17.5

  (2) 17.41

  (3) 17.5, если гипотенуза = 8.6

  (4) Условие задачи противоречиво!



- Чтобы принимать решение осознано, нужно знать
  - Для чего *это* делается? *Цель,* ожидаемые *результаты*, критерии успеха
  - Из чего *это* делается? Какие *ресурсы* и *материалы* доступны
  - Какие *средства* и *инструменты* есть в распоряжении?
  - Как *это* делается? Какие *методы* можно или нужно применять для использования средств и инструментов

Составляющие элементы деятельности

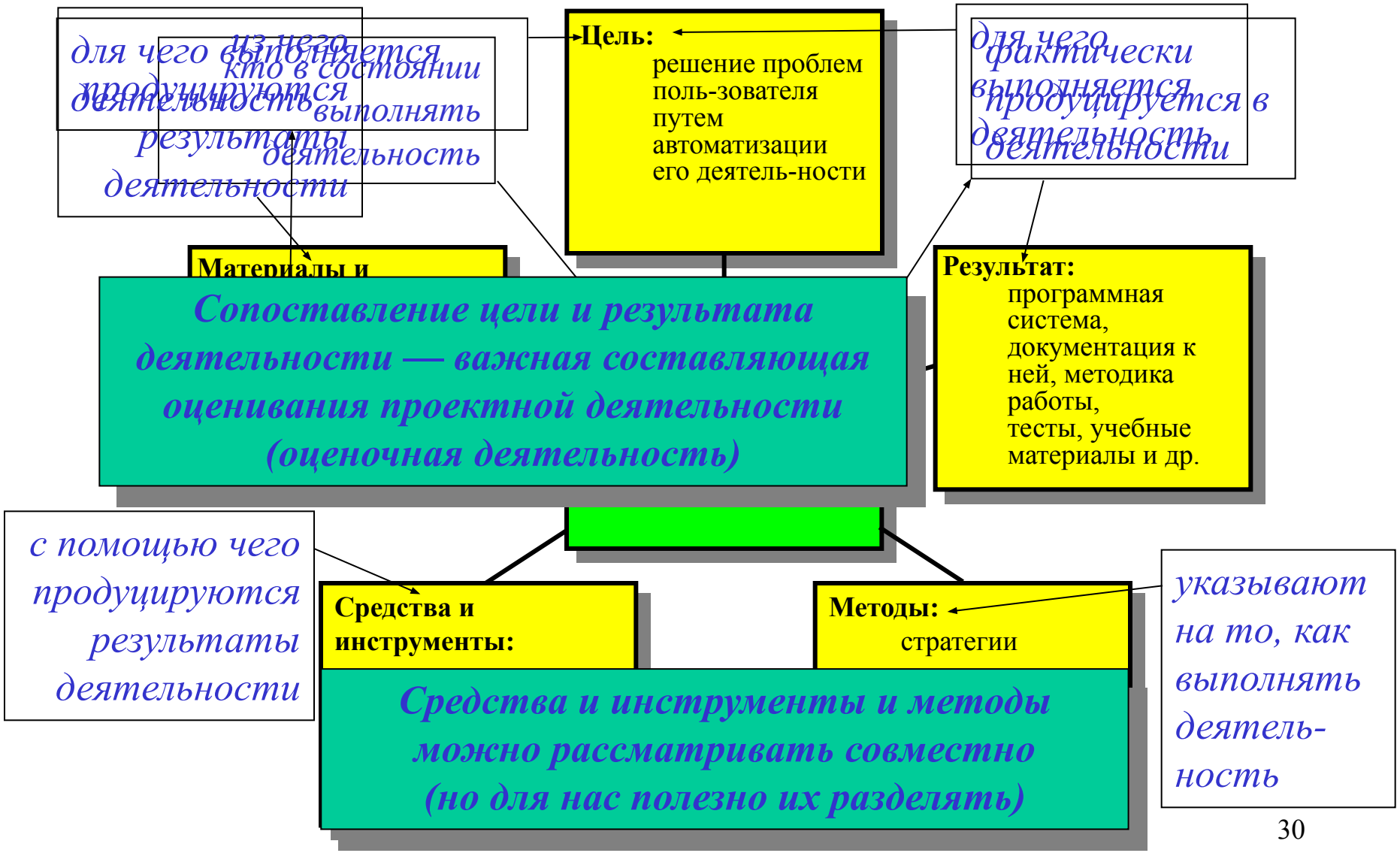
  + Для чего *это* делается? Как *используются* получаемые результаты?
  + Для кого *это* делается? Кто *использует* получаемые результаты?
  + В каком качестве предполагается использование?

Окружение деятельности

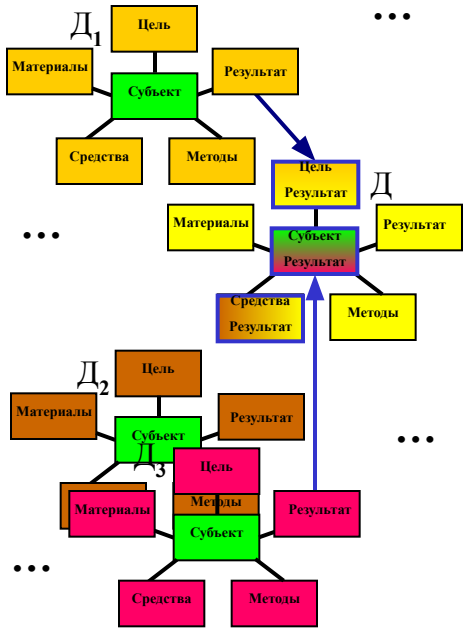
Декомпозиция проектной деятельности

- *Проект — большая производственная функция*, выполнение которой требует осуществления многих различных деятельностей.
- Деятельности проекта взаимосвязаны, нуждаются в планировании, обеспечении ресурсами и др. От существования многих из них приходится абстрагироваться (несущественные?)
- *Организованность* совокупности проектных и внешних (косвенно связанных с проектом) деятельностей должна быть достигнута!
- Для построения нужной организованности применяются методологии развития проектов. Среди прочего они решают задачу *декомпозиции*.
- Это одно из назначений методологии. А какие они бывают?
- Производственные функции и исполнители как декомпозируемые сущности
  - можно *структурировать выполнение функции*, разбивая ее на составляющие, определяя назначение каждой из составляющих и связи между ними так, чтобы результат совместного выполнения совпадал с требуемым результатом разбиваемой функции
  - можно *структурировать обобщенного исполнителя*, иными словами, конкретизировать исполнителей, отвечающих за разные аспекты выполнения функции
- Оба разбиения допускают продолжение в глубину:
  - для исполнителей — до конкретных индивидуумов
  - для функций — до неделимых единиц действий

Элементы деятельностей



Системы деятельностей



Любая деятельность есть часть некоторой ***общей системы деятельностей***, охватывающей группу субъектов-исполнителей. Деятельности, субъекты которых не попадают в выделенную группу, являются ***окружением* данной системы**. Окружение связано с системой следующими способами:

- из окружения поставляются элементы деятельностей системы;
- деятельностям окружения передаются результаты деятельностей системы;
- система в целом и ее отдельные деятельности являются элементами деятельностей окружения

- **Нужно всегда знать *место методики* (*методологии*) в системе проектных деятельностей**
- **Нужно всегда знать *место* всех деятельностей, на которые возможно и следует влиять, чтобы система проектных деятельностей (проект) развивалась успешно!**
- **Недостаточно говорить только о процессах (обычно этим и ограничиваются — см. слайды про PMBOK)!**

Автоматизированная и автоматическая деятельность. Цель программирования и цель методологии программирования

Автоматизированная — по сравнению с неавтоматизированной приводит к результатам с меньшими затратами (всего)

Автоматическая — неодушевленный субъект ≡ деятельность без субъекта ≡ одушевленный субъект осуществляет единственное воздействие-активизацию

Цель программирования — построить автоматические или автоматизированные деятельности для внешнего субъекта

Цель методологии (методики) программирования — построить автоматические или автоматизированные деятельности, заменяющие неавтоматизированные аналоги (по возможности — *всех*!) деятельностей, относящихся к выполнению проектного задания

Понятие требований
(к проекту, программной системе и др.)

- Замысел, основная идея проекта → желаемый результат.
- Что такое «*желаемый результат*»? Ответ в требованиях (к проекту, процессу разработки к квалификации исполнителей и др.)
- Что такое *требования*? Много разночтений.
  - *Пользовательские требования* — *что* должна делать система, какие функции она должна выполнять, какие возможности взаимодействия с ней должны быть предложены и др. + *дополнения* (например, языковая среда пользователей) *и ограничения* — часто об этом умалчивают
  - *Системные требования* — *как* система должна работать,
    (1) характеристики производительности при выполнении функций, эргономичности и др., (2) описание необходимого окружения: оборудование, программы и др. (3) совместимость, переиспользуемость и др. + *ограничения*

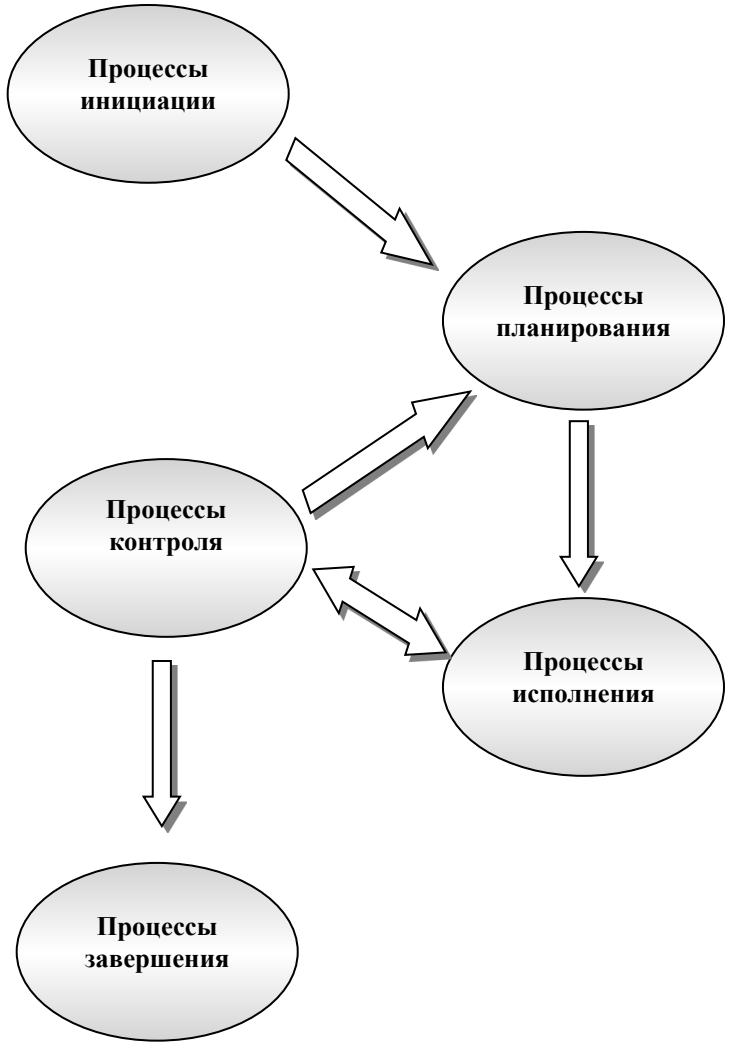
  Иногда дополняют это. Например так:
  - *Проектная спецификация* — «обобщенное описание структуры программной системы [?], которое будет основой для более детального проектирования системы и ее последующей реализации» И. Сомервилл
- Как требования возникают, как они задаются, как учитываются в разработке и др.?

Подробности — в дальнейшем.

Сопоставление со стандартом PMBOK

PMBOK — Project Management Body of Knowledge (институт менеджмента проектов PMI)

- «Процесс — это серия действий, приводящая к результату» (действие — ?, результат —?)
- «Процесс — любая деятельность, в которой используются ресурсы для преобразования входов в выходы» (ISO 9000)
- Деятельность *шире* процесса!
- Группы процессов PMBOK:
  - *процессы инициации*
  - *процессы планирования*
  - *процессы исполнения*
  - *процессы контроля*
  - *процессы завершения*



PMBOK-процессы и система деятельностей разработки программных проектов

Схема иллюстрирует сущность менеджмента проекта на самом абстрактном уровне как *деятельность, направленную на организацию и поддержку целенаправленного развития обозначенных на ней групп процессов*

Более конкретная интерпретация **системы деятельностей по разработке программных проектов** включает:

- …
- анализ предварительных требований,
- конструирование архитектуры,
- составление программного кода,
- проверка кода,
- …

Это именно деятельности, а не просто процессы! Нужно иметь ввиду (учитывать, планировать и т.п.) *все элементы каждой деятельности*!

Эти деятельности зависят между собой, поставляя свои результаты друг другу, они ***не могут выполняться в произвольном порядке***.

Дальнейшая конкретизация проектировочных видов деятельности должна следовать выбранной для проекта методической установке.

Отделение существенного от несущественного — важный аспект формирования системы как объекта управления.

IDEF0 – методология функционального моделирования
(1993 – федеральный стандарт в США, 2000 – стандарт РФ)

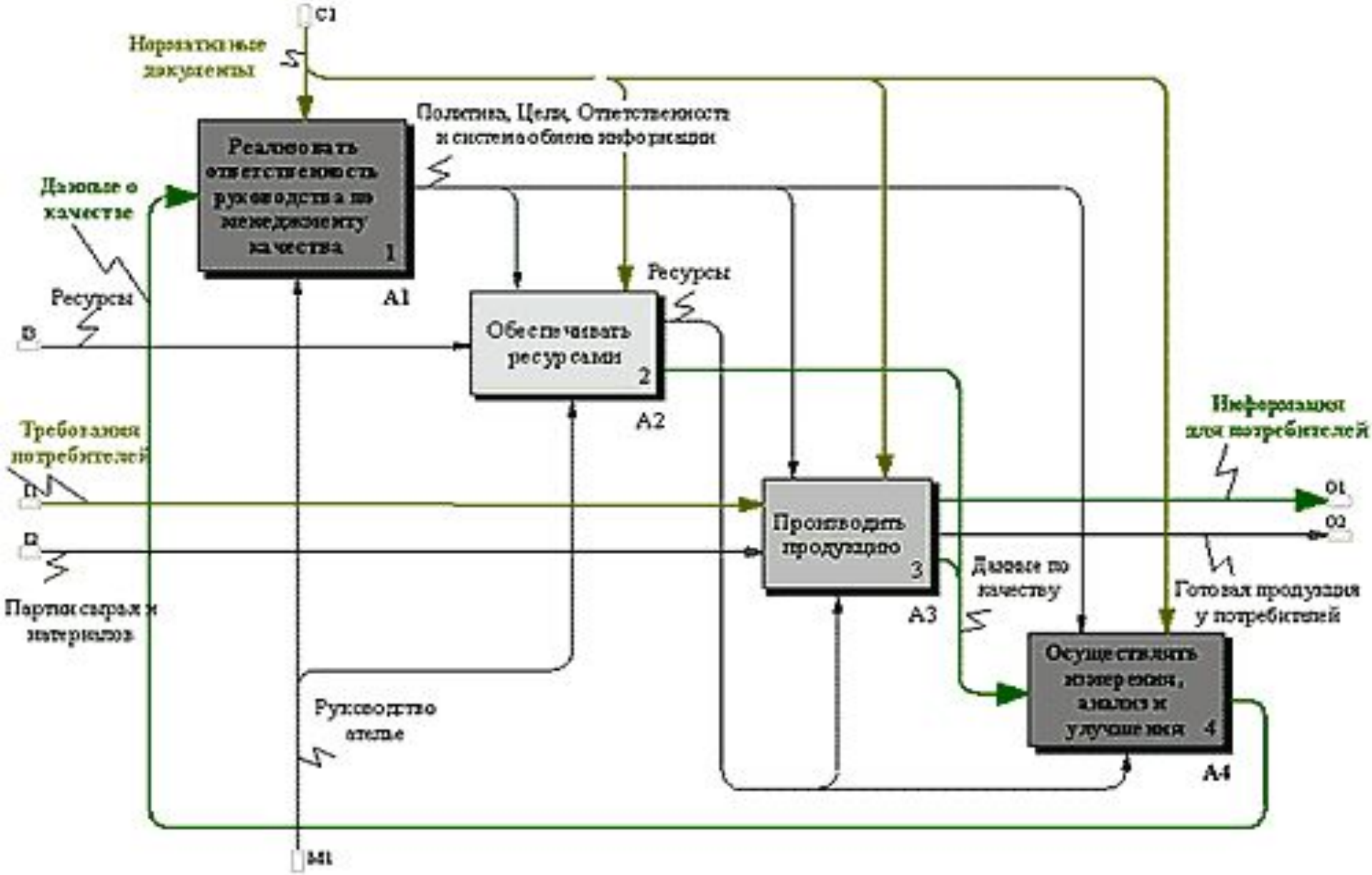
- Процессы вместе с взаимосвязями и взаимодействиями представляют сеть процессов организации.

Деловой процесс – это совокупность процессов (операций, действий) и взаимодействий между ними, результатом (выходом) которой является продукция и/или услуги, поставляемые потребителям, а входами – материальные, информационные и трудовые ресурсы, поставляемые внешними поставщиками.

Функциональная модель делового процесса охватывает процессы жизненного цикла, а также связанные с ними вспомогательные процессы и процессы менеджмента, входящие в состав деятельности организации. Это полностью согласуется с требованиями МС ИСО семейства 9000 версии 2000 года.

Деловой процесс в швейном ателье

Пример детализации IDEF0 диаграммы



Ограниченность процессного и деятельностного представлений поведения системы

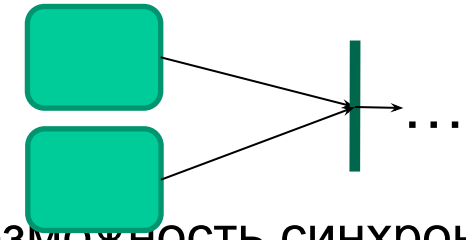
Процессы

- Последовательное выполнение

Зависимость:

Связи входов с выходами

- Параллельное выполнение



Возможность синхронизации

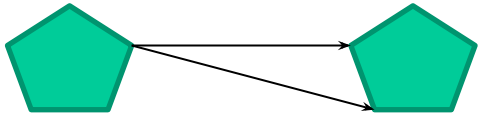
Барьеры

- Конвейерное выполнение

Нет средств отображения (дополнительный процесс не существует)!

Деятельности

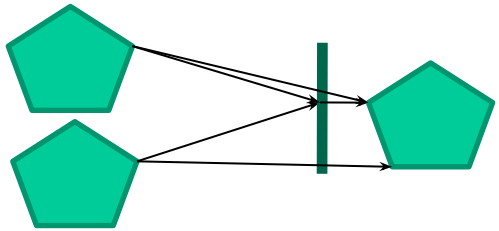
- Последовательное выполнение



Зависимость:

Поставка элементов (любых!)

- Параллельное выполнение



Возможность синхронизации

Барьеры

- Конвейерное выполнение

За счет разделения поставки и использования элемента (внутри звездочки!)

Процессы и система деятельностей разработки программных проектов

Процесс ассоциируется с понятием времени (хотя и без привязки к нему)

Звездочка деятельности – нет, но только потому, что время нельзя рассматривать изолированно от других деятельностей (как и процессы).

Виды деятельностей в системе (условно):

- Продукционная – производит какой-то продукт
- Проектировочная – в качестве продукта производит план выполнения некоторой продукционной деятельности
- Наблюдательная – следит (?) за ходом продукционной деятельности
- Управляющая – наблюдательная, которая имеет средства воздействия на продукционную с тем, чтобы достигалось соответствие с плану.

Событие – воспринимается в наблюдательной деятельности, она может продуцировать протокол продукционной деятельности (последовательность событий), т.е. локальное время.

Глобальное время не существует! Это условность, удобная для примитивной приблизительной синхронизации.

События – основа адекватного отслеживания времени в системы (контрольные точки, вехи – элементы синхронизации деятельностей проекта с точки зрения его менеджмента

Вопросы и задания

Для всех:

- Какими деятельностями мы занимаемся, изучая предмет нашего курса?
  - Указать субъектов с их целями и другими элементами деятельностей
  - Соотнести цели с получаемыми результатами
- Что такое «польза»?

Для тех, кто считает нужным сертифицироваться в PMI:

- Сопоставить процессы с системой проектных деятельностей
  - найти все элементы деятельностей PMBOK-процессов
  - найти все влияния их на окружения
- Характеризовать методики, которые предлагаются PMBOK для выполнения их процессов с точки зрения

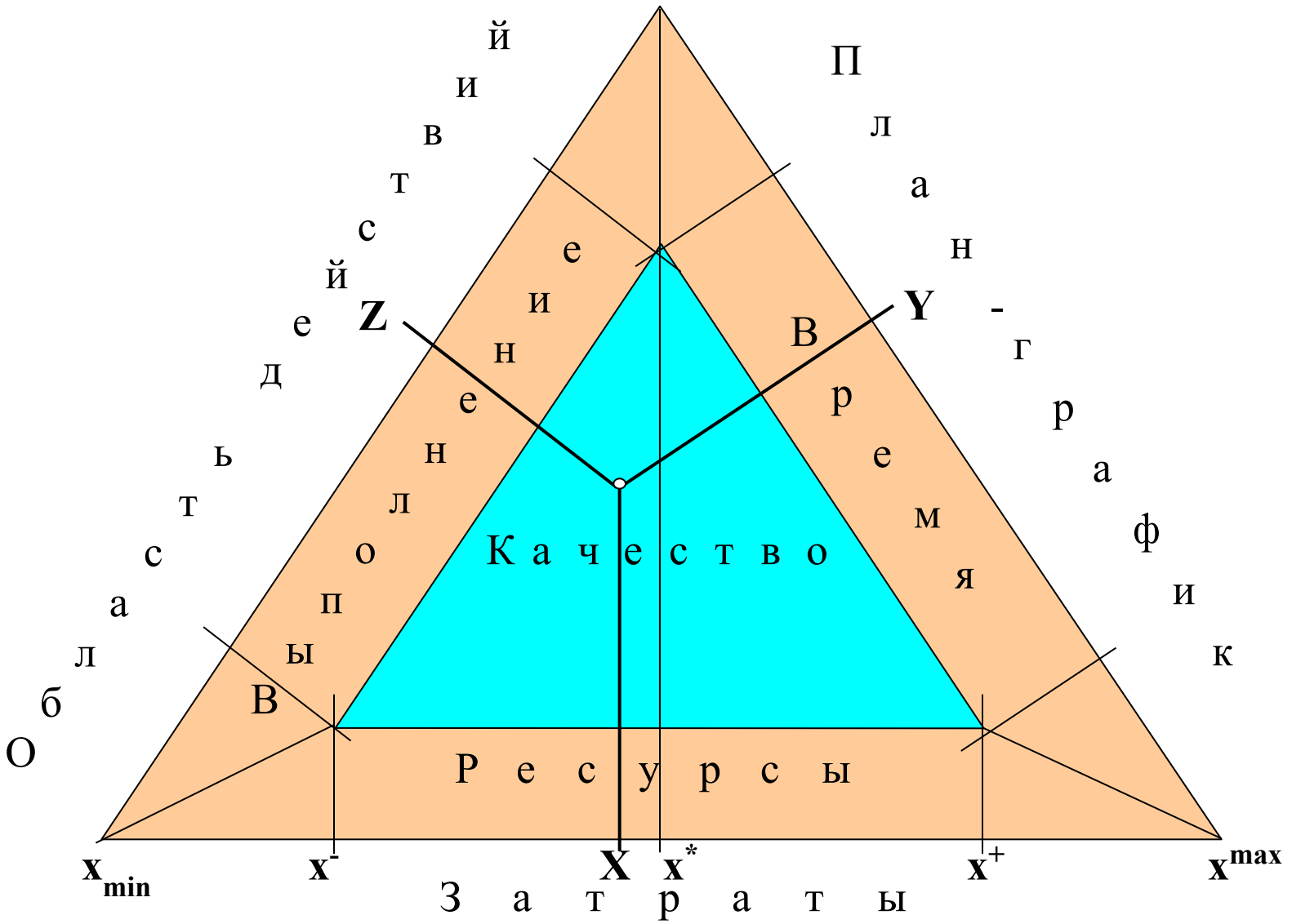
Деятельность менеджера и составляющие системы проектных деятельностей

- *Цель* — направление других проектных деятельностей так, чтобы они продвигали проект к выполнению (задаваемых вне системы) *работ* в условиях ограничений по *времени*, *финансам* и *качеству* = достижение целей деятельности, задающей проект в целом.
- *Согласование параметров проекта*: объем работ, сроки, запрашиваемые финансы (внешняя по отношению к работам проекта деятельность)
- Менеджмент проекта — обеспечение предоставления продукта для его использования, разработка которого
  - требует выполнения определенного объема работ — *область действия*,
  - использует *затраты* (в определенных пределах) — *ресурсы*,
  - старается укладываться в заданные рамки *времени* — *план-график работ*,

  и должна удовлетворять приемлемому уровню *качества*.
- Это хорошо известный *треугольник менеджмента «хорошо-быстро-дешево»*: из трех параметров выбери два — получишь третий!

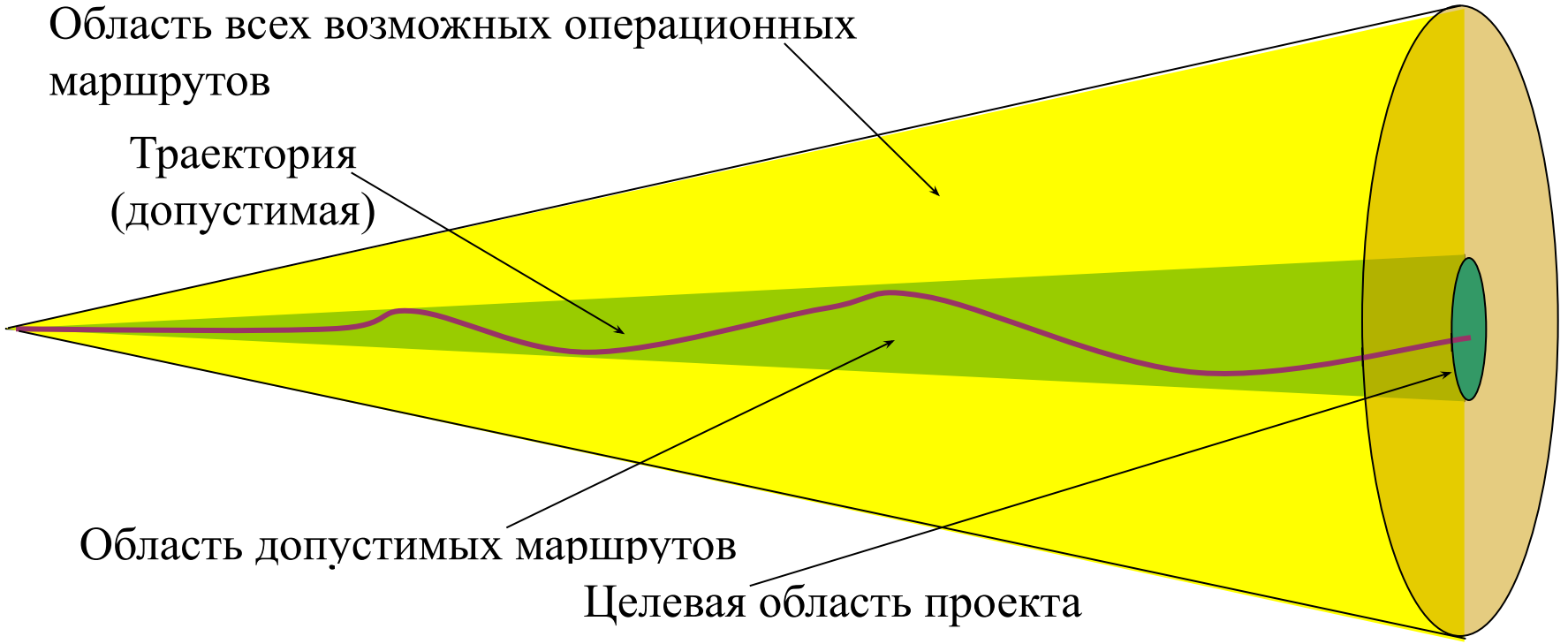
Задание: 1. Конкретизировать элементы деятельности менеджера и их связи с другими деятельностями;
2. Сопоставить свой результат с полученным другими;
3. Объяснить различия.

Треугольник менеджмента проектов — иллюстративная модель



Другие подобные модели см. в книге

Операционные маршруты и траектории деятельности



Операционный маршрут — возможные последовательности действий, в каждый момент выполнения деятельности.

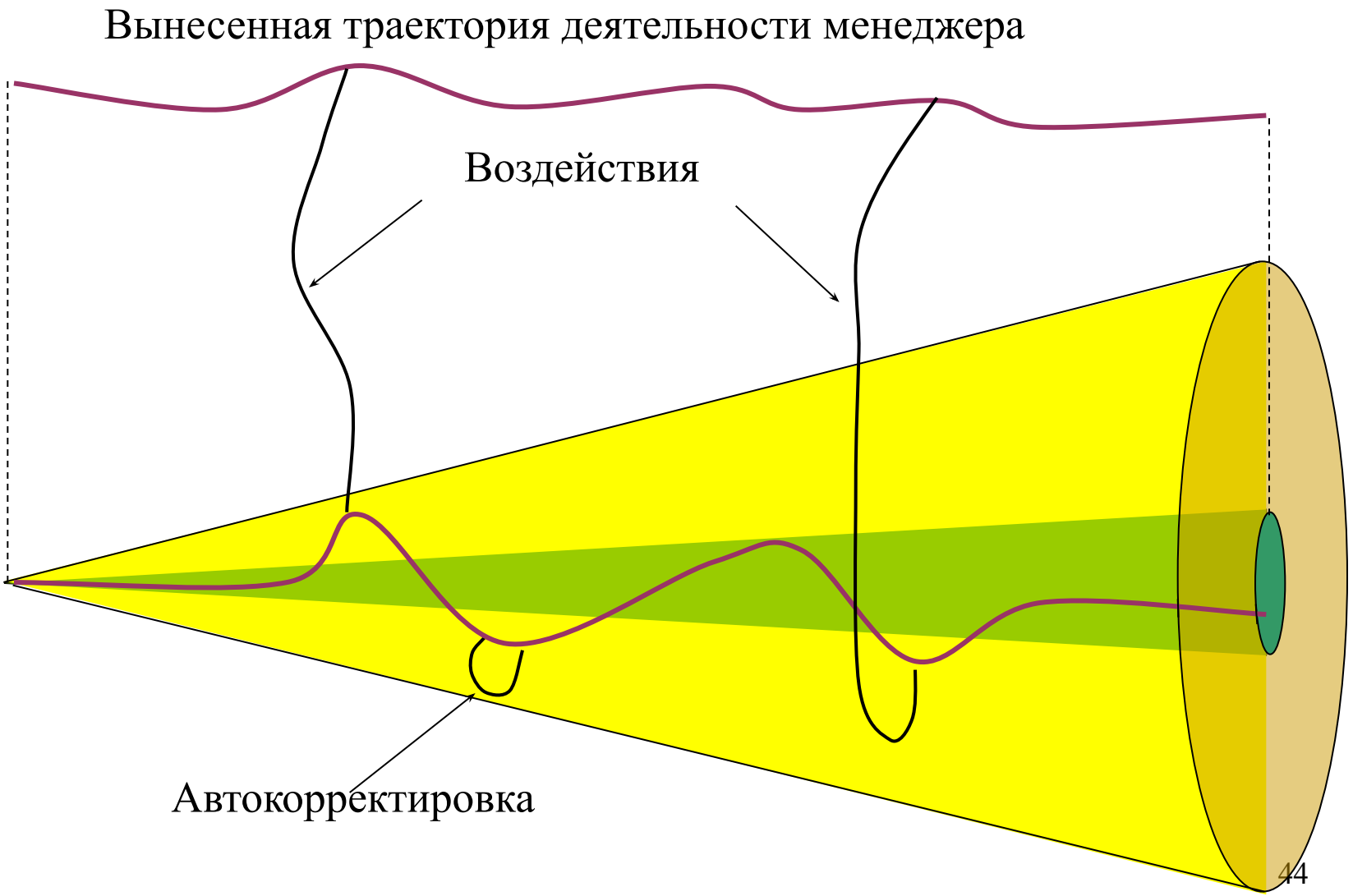
Траектория — реализуемый операционный маршрут

Это *атрибуты* — чего?

- роли;
- индивидуума;
- инструмента — осуществимость с его помощью определенных маршрутов (полезных для выполнения тех или иных производственных функций)

Обстановка операционного маршрута — все элементы деятельности, которые могут использоваться субъектом для выбора продолжения траектории

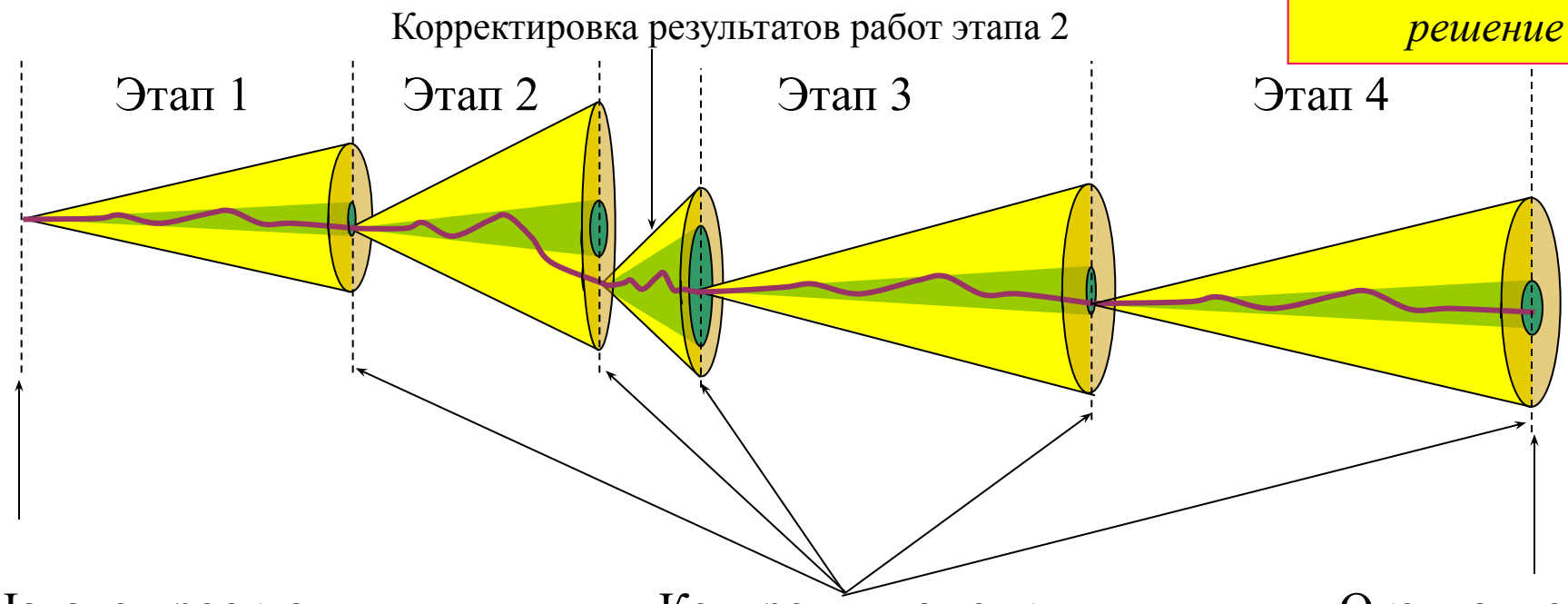
Выяснение отклонений и корректировка траектории



Определение этапов проекта:

последовательное развитие проекта

Это всего лишь иллюстративная стратегия, а не решение!



Постановка задачи каждого этапа характеризуется:

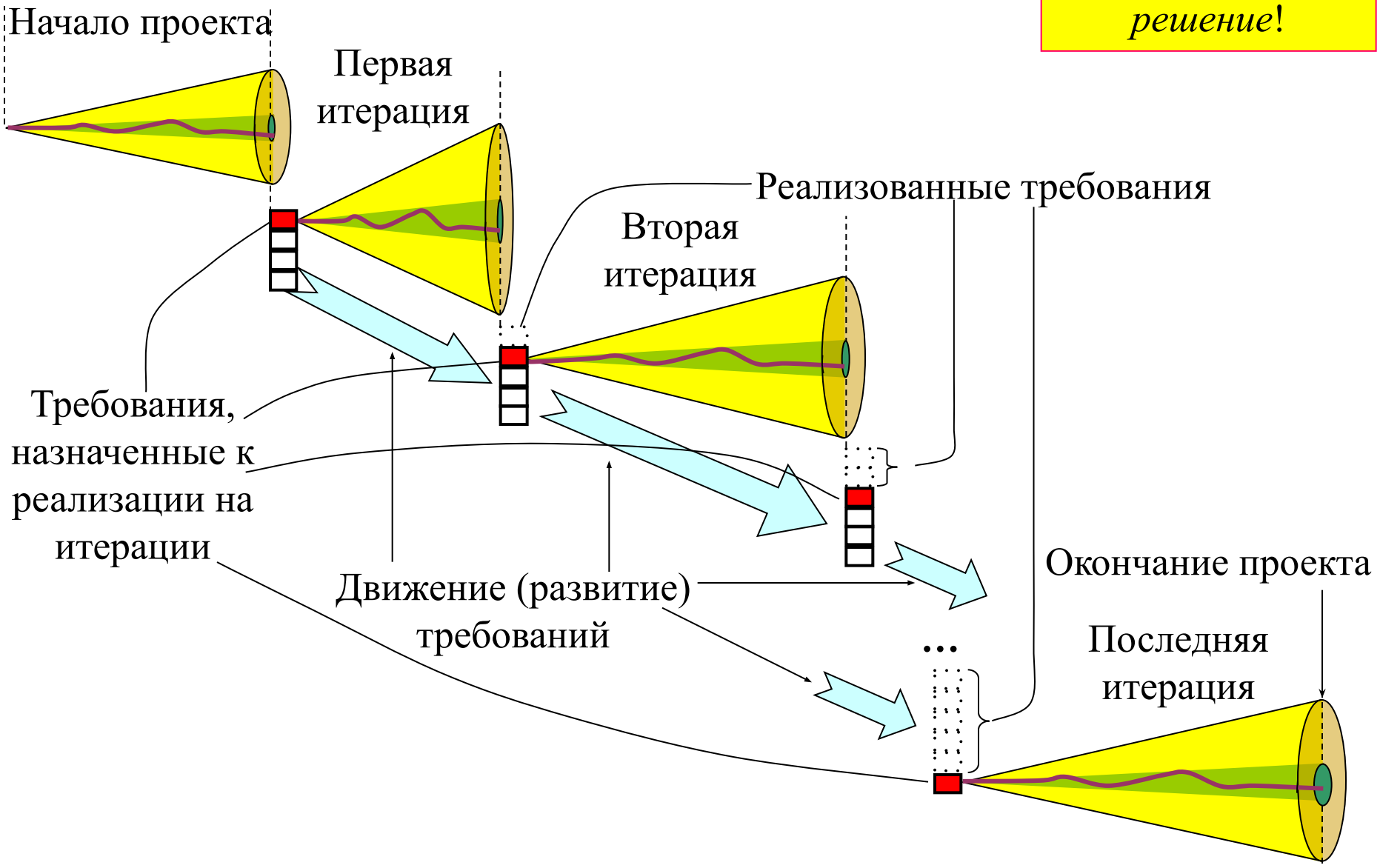
- субъектом-исполнителем;
- сроками выполнения;
- выделенными ресурсами;
- средствами, методами и инструментами;
- контрольными мероприятиями.

Разбиение этапа на локальные этапы, работы, задания.
Параллельное выполнение их.
Деятельность менеджеров по направлениям

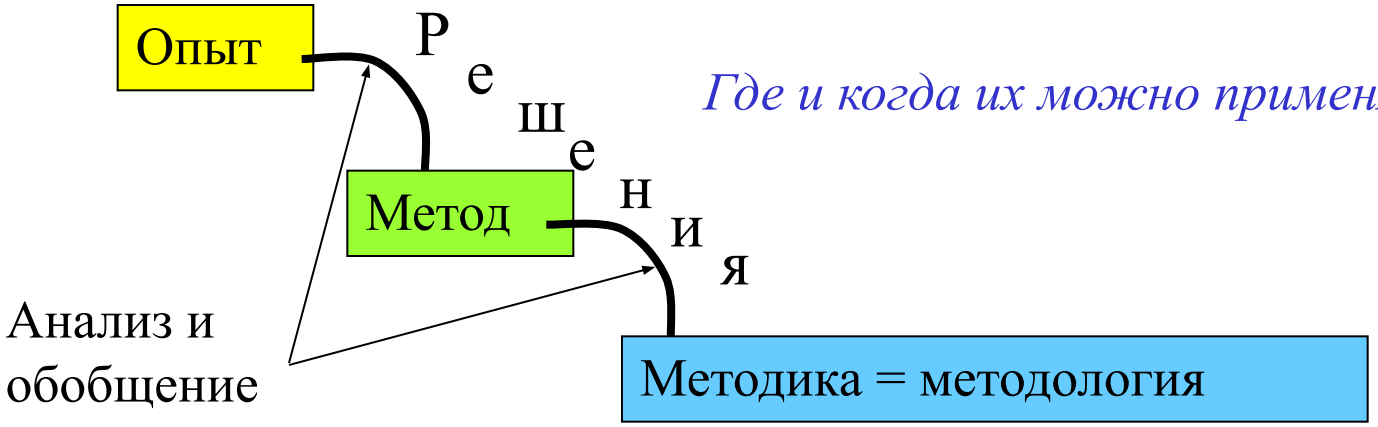
Сокращение объема конуса за счет использования более мелких конусов

Сужение текущей задачи проекта:
итеративное наращивание возможностей

Это всего лишь иллюстративная стратегия, а не решение!



Жесткие и гибкие стратегии в методологиях программирования



Где и когда их можно применять?

Жесткие / тенденция стандарта (CMM, МО США)

Жесткие ≡ тяжеловесные ≡ **монументальные**

Быстрые / agile development

Быстрые ≡ ***гибкие*** ≡ шустрые ≡ облегченные

Agile Manifesto

- **Индивидуумы и взаимодействия *важнее* процессов и инструментов;**
- **Работоспособное ПО *важнее* обширной документации;**
- **Сотрудничество с заказчиком *важнее* заключения контракта;**
- **Готовность к изменениям *важнее* следования плану.**

Императивные и креативные деятельности

Императивные деятельности — выполняются в известных ситуациях, в которых могут использоваться известные предписания, регламенты и методы, с тем, чтобы операционные маршруты не приводили к недопустимым траекториям.

Креативные деятельности — допускают ситуации, в которых известные предписания, регламенты, методы могут не срабатывать, а потому предполагают конструирование новых методов, которые обеспечивают допустимость траекторий операционных маршрутов. **Это более высокий уровень знаний и умений!**

Методологии регламентируют не одну, а комплекс деятельностей, потому говорить строго об императивных или креативных методологиях неправомерно, но …

Жесткие методологии	**Быстрые методологии**
Предсказуемость проекта	Непредсказуемость проекта
Детерминированный процесс	Недетерминированный процесс
Выбор стратегии развития проекта	
***Преимущественно* императивные**	***Преимущественно* креативные**

Сопоставление жесткой и быстрой стратегий в методологиях программирования

Жесткие стратегии	Быстрые стратегии
Ориентация на предсказуемые процессы с хорошо обозначенными целями	Осознание того, что процессы разработки в принципе непредсказуемы
Распознавание ситуаций и применение готовых методов	Распознавание ситуаций и конструирование методов для работы в них
Планирование, в котором определяются этапы	Соблюдение баланса между параметрами проекта
Заказчик — внешний по отношению к проекту субъект	Заказчик (его представитель) — член команды разработчиков
Ролевое разделение труда работников	Совместная деятельность работников
Дисциплина и подчинение	Самодисциплина и сотрудничество
Обезличенный процесс, исполнители которого определяются только по квалификационным требованиям	Процесс, ориентированный на максимально полный учет человеческих особенностей исполнителей

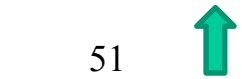
Технология и творчество

Технология какой-либо деятельности — это среда поддержки выполнения деятельности, обладающая средствами и инструментами, а также методами их применения, неукоснительное следование которым каким бы то ни было исполнителем с определенной квалификацией, гарантированно обеспечит производство, т.е. получение из предоставляемых ресурсов и материалов продукта-результата, соответствующего целям, в требуемом объеме за известное время и с приемлемым уровнем качества.

Жесткие методологии — стремление к абсолютной технологизации, но это заведомо **недостижимо для программных проектов!**

Вопросы и задания

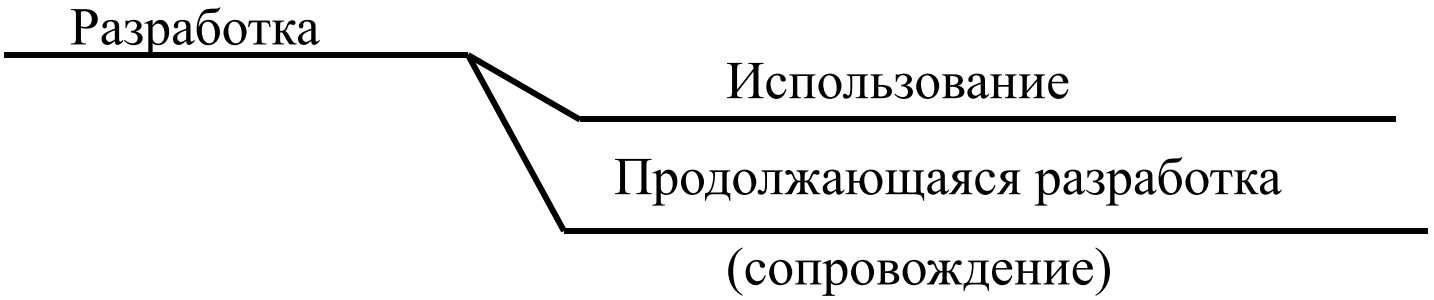
- Является ли разработка методологии креативным процессом?
- Можно ли жесткую методологию сделать гибкой? Ответ обосновать.
- Может ли гибкая методология стать жесткой? Ответ обосновать.
- Исследовать какую-либо традиционную жесткую методологию с точки зрения ответа на вопрос, какие методики предлагаются для поддержки креативных деятельностей в программных проектах.
- Выяснить, для какого типа проектов нерационально использовать какую-либо гибкую методологию (например, Extreme Programming).



3. Жизненный цикл программного обеспечения и его модели

Мотивация изучения жизненного цикла и его моделей

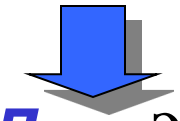
- Жизненный цикл — база методологий
- Жизненные циклы технических и программных разработок (конструкционные материалы и окружение ПО, старение, продолжающаяся разработка (сопровождение):



- Причины изучения моделирования жизненного цикла:
  - для непрофессионалов — понимание реальных возможностей;
  - основа адекватного применения инструментов и методов разработки;
  - общие знания — ориентиры для планирования, аргументированность решений;
  - правильное распределение обязанностей в коллективе
- Соглашения, предписания, регламенты разработки и цена их игнорирования

Жизненный цикл программного обеспечения: определение

- Цикл разработки,
- Издержки после завершения разработки



Жизненный цикл — это проекция пользовательского понятия «время жизни» на понятие разработчика «технологический цикл (цикл разработки)».

Происхождение термина *жизненный цикл*

Последовательное развитие и итеративное наращивание и модели жизненного цикла.

ООП — реальная основа итеративной схемы (не только оно возможно)

Жизненный цикл: последовательные и итеративные схемы

Сегодня это самый популярный вариант итеративной схемы

Традиционные схемы	Объектно-ориентированная схема
Полностью завершенные фазы проектирования и программирования	Итеративно наращиваемые возможности Традиционные фазы распределяются по итерациям
Продукт в конце периода разработки	Рабочие продукты на каждой итерации
Техническое описание как итог конструирования	Рабочее описание, дополняемое на каждой итерации
Последовательная разработка	Возвратно-поступательная разработка
Модули действий, операций	Иерархии классов объектов
Структурная, пошаговая детализация	Наследование, переопределение, полиморфизм

Задача методологии и жизненный цикл

- Методологии — это инструменты, с помощью которых создание программного продукта превращается в упорядоченный процесс, а работа программиста становится более прогнозируемой и эффективной ⇒ ***планирование процесса***
- В материальном производстве: ***замысел → чертежи → проектные решения →*** ***точное воспроизводство плана***

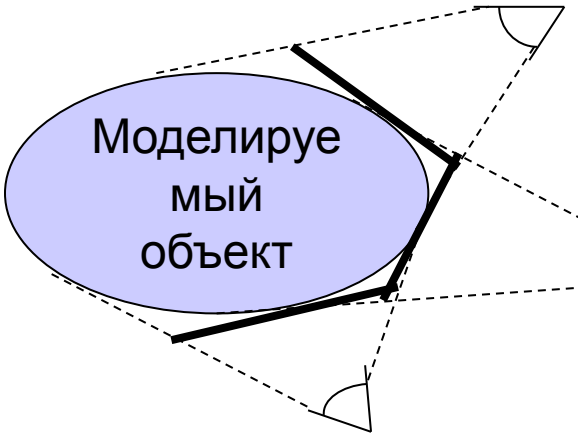
  креативность
  - ⇒ Расчет времени и стоимости, определение требуемой квалификации и др.
  - ⇒ В традиционном производстве: ***рост технологичности & снижение креативности***
- Перенос в программирование. Разграничение двух видов деятельности:
  - ***проектирование*** (креативность)
  - ***производство*** (технологичность)

  Полностью избежать креативности не получится!
- Задача методологии: ***минимизировать творческий элемент в рутинных случаях*** ⇒ стремление разграничить:
  - план и конструирование программы
  - спецификации пользовательской потребности и план,
  - выбор инструментов работы программиста и саму работу

  ⇒ ***Появление соглашений, регламентов и предписаний, следование которым уменьшает вероятность ошибочных решений***
- Форма представления жизненных циклов в разных методологиях различна, но в основе любых представлений разработки и сопровождения программных изделий лежат общие процессы, которые ведут проекты от замыслов к удовлетворению пользовательской потребности.

Любая методология предписывает организацию этих общих процессов

Модели процесса и продукта



Модель процесса разработки:

- **Целенаправленное развитие объекта** под воздействием разработчиков
- **Ключевые понятия**: развитие, система деятельностей, субъект ⇔ объект, этапы деятельностей, инструменты деятельности, цели, результаты и продукты

Модели продукта (различные)**:**

- **Как устроен** (построен) продукт? **Для чего нужен**?
- **Один из типов моделей продукта**: проекция модели процесса, в которой игнорируется все, связанное с субъектом
  возможна реконструкция модели процесса, которая необязательно совпадает с исходной моделью процесса!

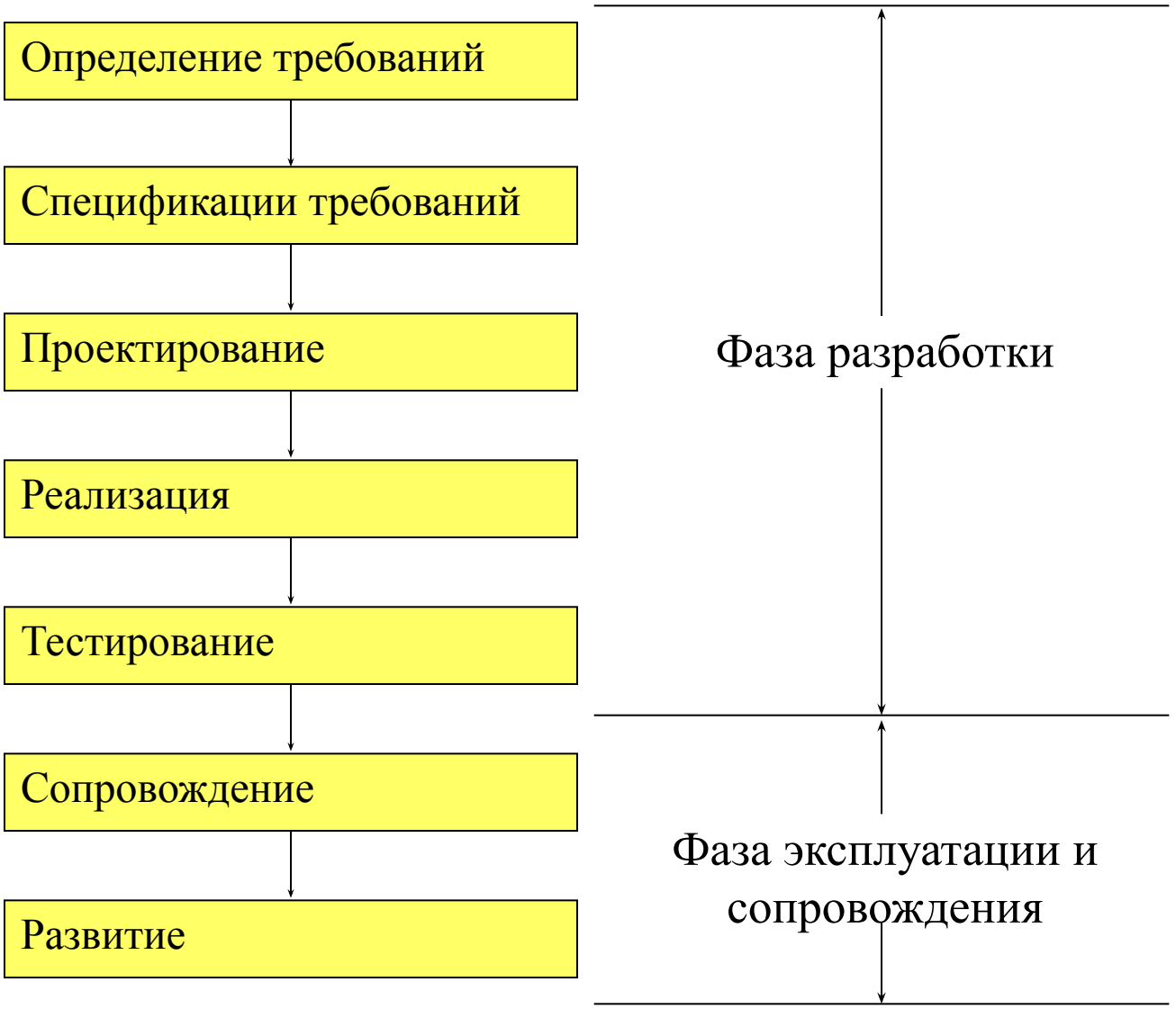
Иллюстративность модели: явное выделение некоторых аспектов для облегчения их понимания

Инструментальность модели: использование ее в качестве инструмента некоторой деятельности (т.е. способствует целенаправленному развитию).

Нельзя смешивать иллюстративные и инструментальные модели

Вопросы в связи с моделью: **Что будет, если…?** и **Соответствует ли…?**

Общепринятая модель жизненного цикла программного обеспечения

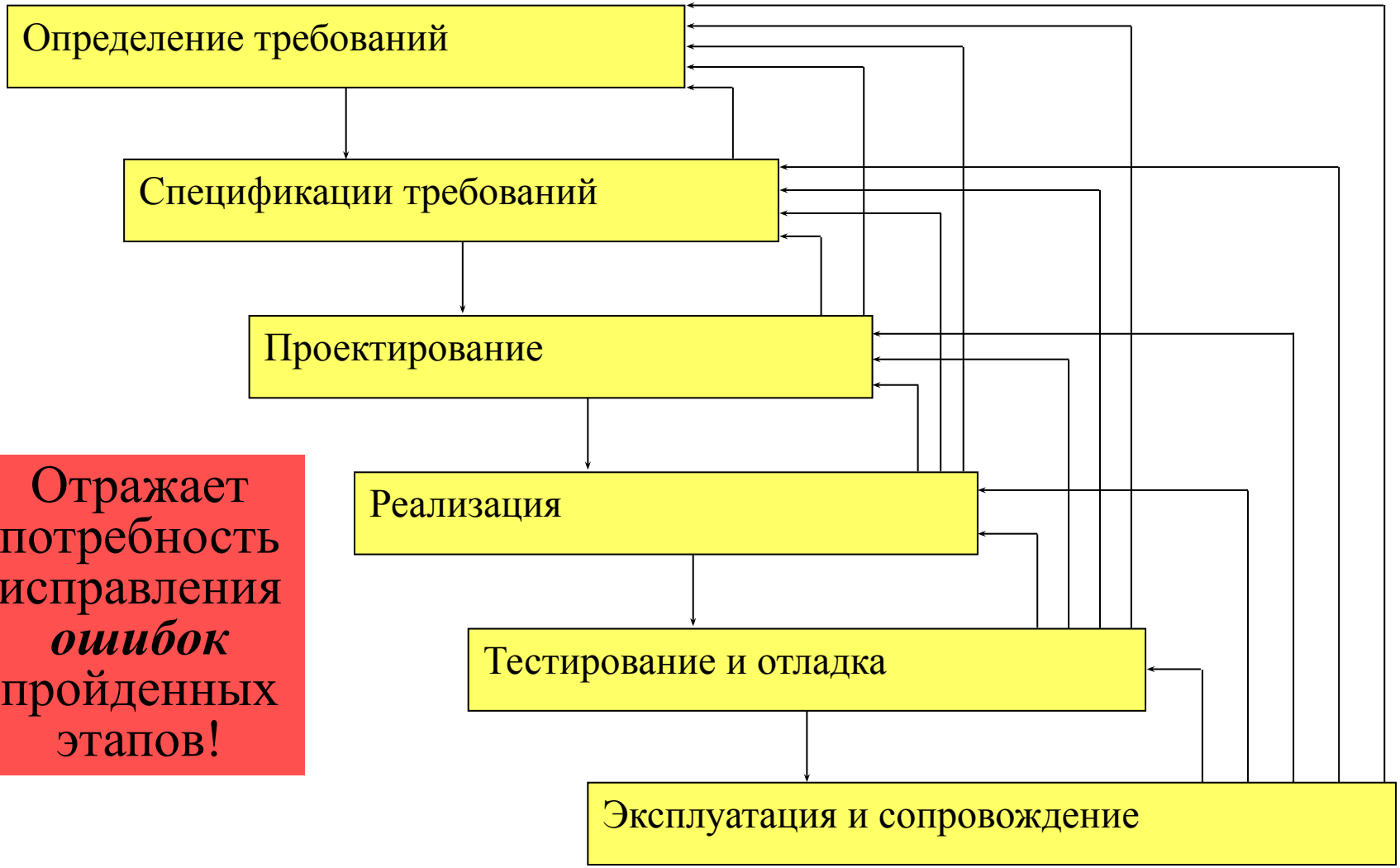


Общепринятая модель — источник базовых понятий

Начало разработки — *идентификация потребности*

- *Определение требований* — определяются: контекст задачи, ожидаемые функции ограничения. Проекта еще нет.
- *Спецификации системы в соответствии с требованиями* — Определяется поведение системы: ***что*** она должна делать. Фактическое начало работ
- *Проектирование (конструирование, дизайн)* — определяется декомпозиция системы /архитектура & результат ее построения/:
  как достигается соответствие спецификациям
- *Реализация (кодирование)* — разрабатываются модули (в модели нет этапа *интеграции*):
  что обеспечивает требуемый результат
- *Тестирование* — проверка соответствия спецификациям
- *Сопровождение* — поддержка использования продукта
- *Развитие* — поддержка эволюции системы (*новый проект*?)

Классическая итерационная модель



Отражает потребность исправления ***ошибок*** пройденных этапов!

Исправление ошибок или адаптивность проекта?

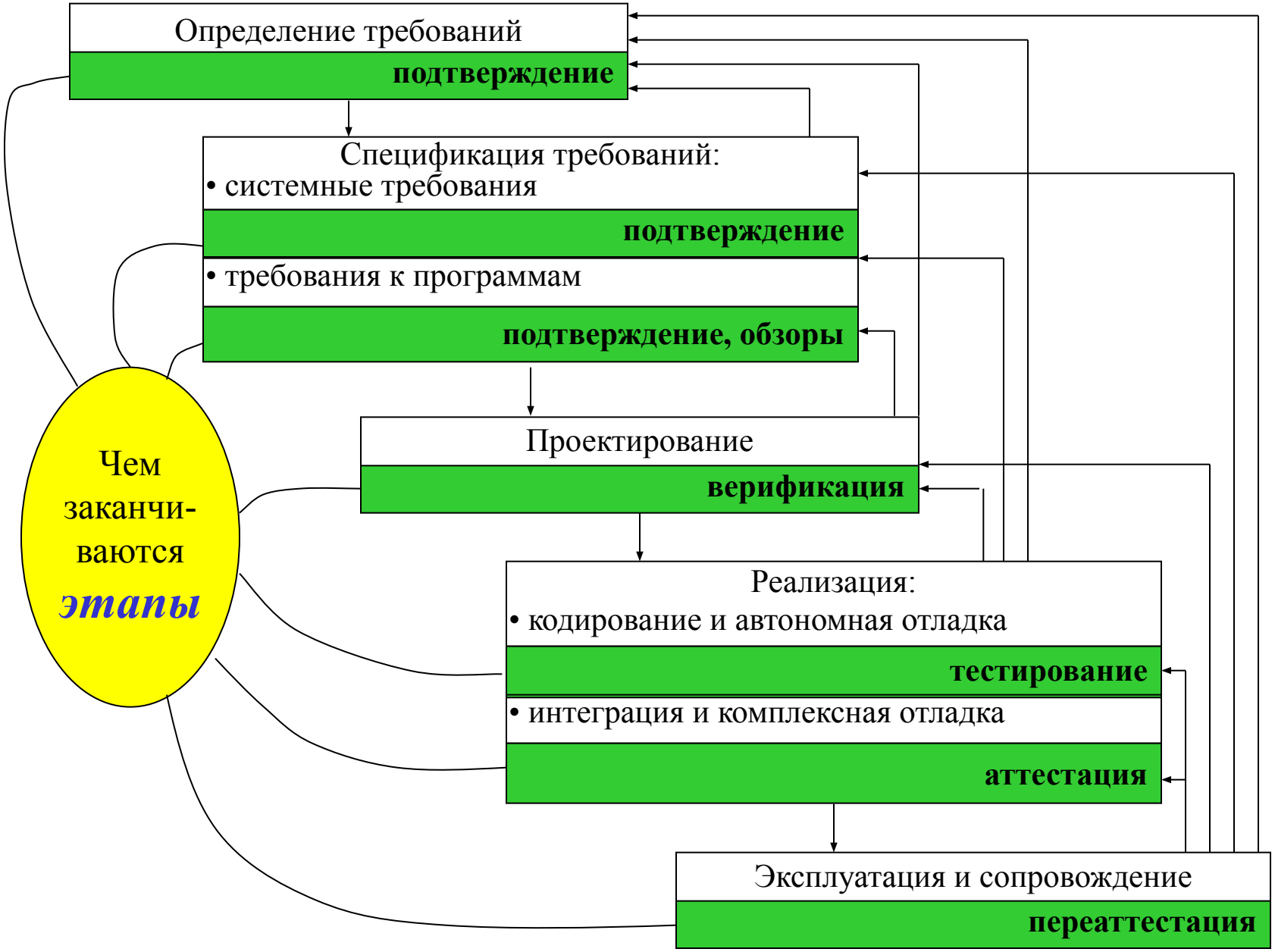
- Классическая итерационная модель — абсолютизация возможности возвратов для *исправления ошибок*, т.е.
- Идея завершенности пройденных этапов — предсказуемость
- Стратегия итеративного наращивания возможностей ослабляет требование завершенности
- ООП + CASE-системы — принципиальная возможность поддержки итеративного наращивания (не более!)
- Адаптивность проекта — альтернатива предсказуемости
- Адаптивность должна закладываться в проект на самых ранних этапах его развития

Задание: Приведите примеры, когда

a. адаптивность вредна для разработки,
b. поддержка адаптивности не помогает справиться со сложностью разработки.

Ответы обоснуйте!

Каскадная модель



Каскадная модель: новые понятия

Характерные черты каскадной модели:

- завершение этапов *проверкой* полученных результатов
- циклическое *повторение* пройденных этапов

Чем заканчиваются этапы?

- *Подтверждение* — разного рода документальные согласования
- *Обзор* — документ, предоставляемый для согласования
- Проверка полученных результатов на соответствие целям:
  - *Верификация* — отвечает на вопрос, правильно ли создана (декомпозиция, программная система и др.)
  - *Аттестация* — отвечает на вопрос, правильно ли работает, т.е. будет использоваться (в первую очередь — программная система)
  - *Переаттестация* — аттестация, которая устанавливает насколько хорошо система соответствует пользовательским запросам

Строгая каскадная модель

За счет чего это достигается?

Строгая каскадная модель: дополнительные требования к разработке проекта

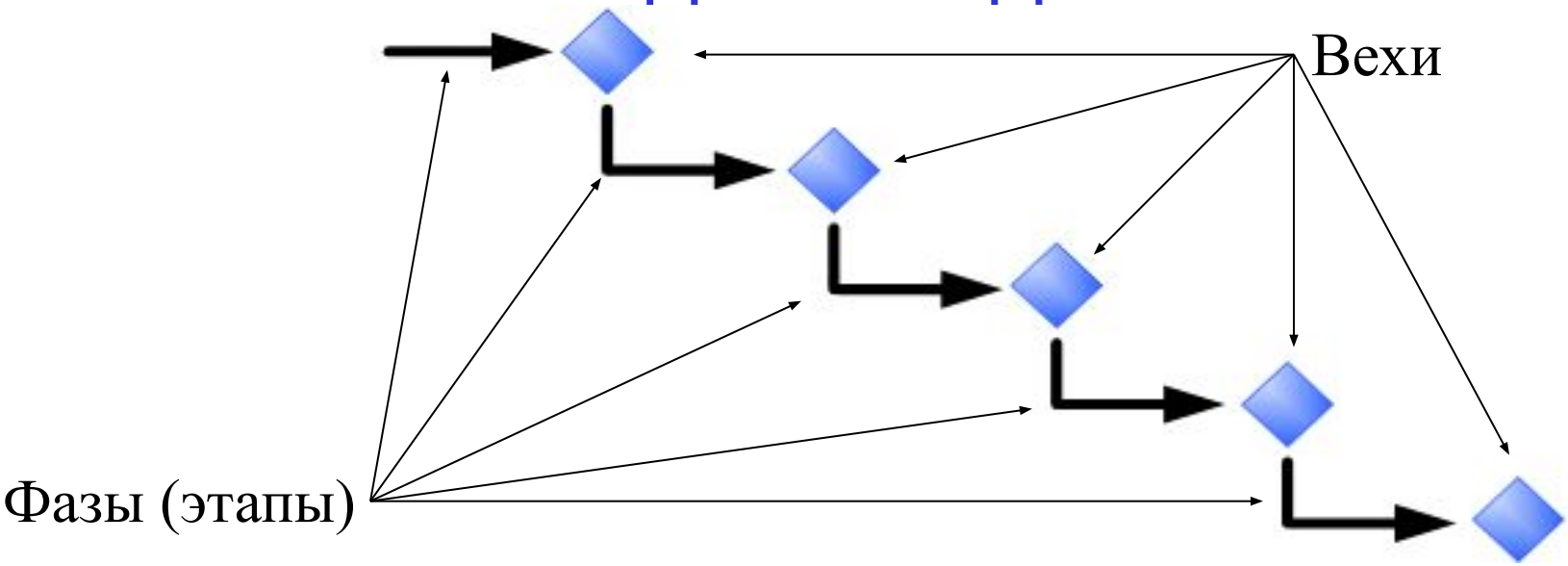
- Минимизация возвратов за счет ликвидации переходов через уровни
  - ужесточение проверок (*барьеров*)
  - переход вниз сопровождается передачей исходного материала для следующего этапа — задание на разработку
- Причины невыполнения задания:
  - i. оно *противоречиво*, т.е. содержит несовместные или невыполнимые требования;
  - ii. не выработаны *критерии* для выбора одного из возможных вариантов решения

⇒ (i и ii) — ошибка предыдущего этапа → он возобновляется:

  - a. ошибка ликвидируется → переход к следующему этапу (через барьер)
  - b. невозможность исправления — это ошибка более раннего этапа → он возобновляется
- Два существенных момента (отражающих реальности разработок проектов):
  - точное разделение работ, заданий и ответственности разработчиков и тех, кто инициирует переход к следующему этапу — *шаг к осознанию фактического разделения труда*
  - малые циклы между соседними этапами, в результате достигается компромиссное задание — *совместное выполнение соседних этапов, т.е. их **перекрытие***

Однако в каскадной модели оба момента отражаются лишь косвенно

Каскадная модель MSF



Вехи (контрольные точки) используются в качестве точек оценки и перехода от одной фазы к другой.

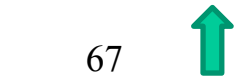
Все задачи одной фазы должны быть завершены до того, как начнется следующая фаза.

Каскадная модель работает, когда на начальном этапе проекта можно четко определить *неизменный набор требований* к разрабатываемому решению.

Оценка: Слабее рассмотренной ранее строгой каскадной модели, но применимость характеризуется верно

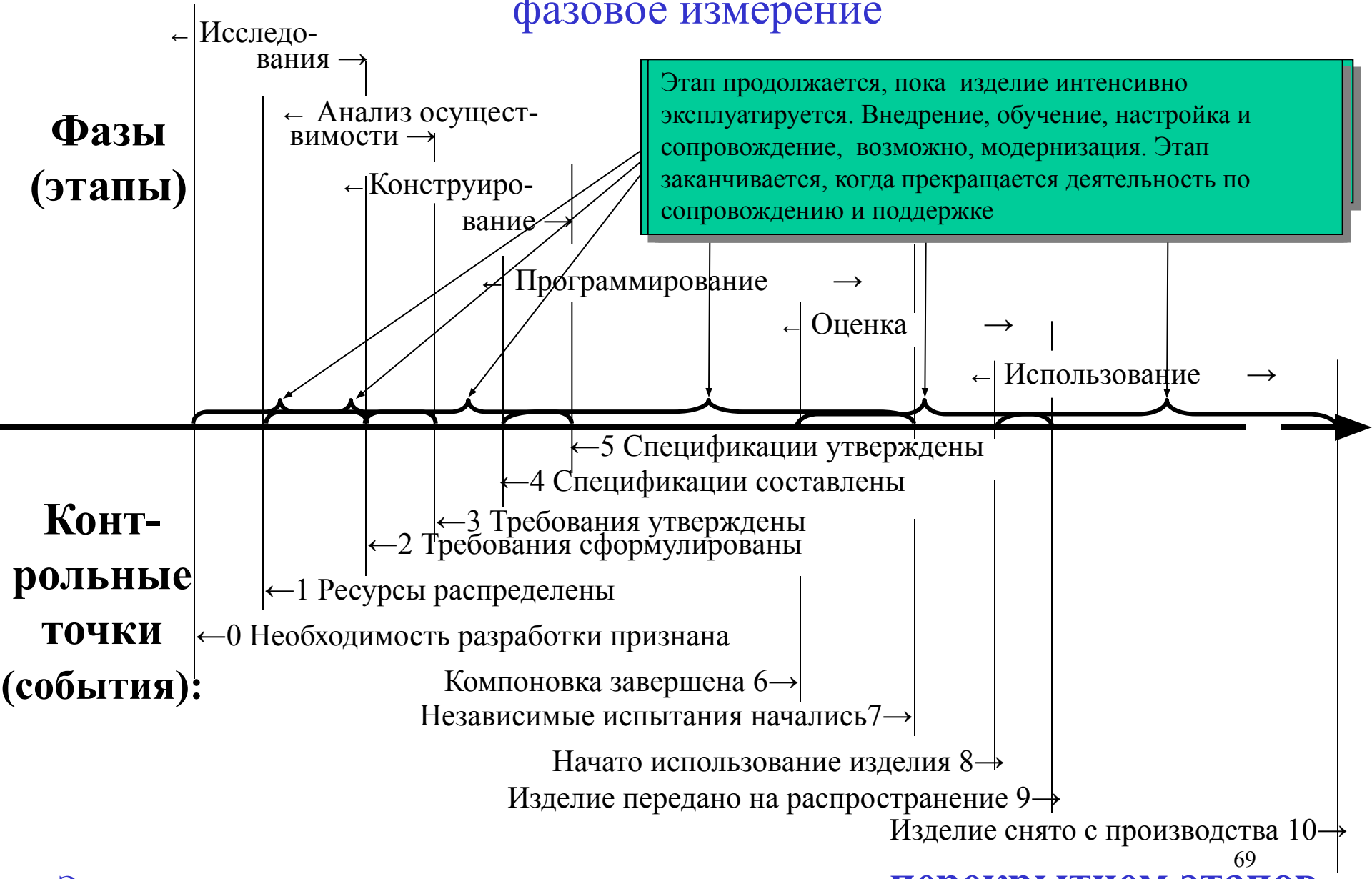
Вопросы и задания

- Какие из рассмотренных моделей можно сделать инструментальными, а какие не допускают этого? Ответ обосновать.
- С какими видами документов, используемых в качестве барьеров вы сталкивались? Ответ поясните.



4. Производственные функции в моделировании жизненного цикла: модель фазы — функции

Модель фазы—функции Гантера:
фазовое измерение



Это традиционное последовательное выполнение проекта с перекрытием этапов

Модель фазы—функции Гантера:
предпосылки функционального измерения (производственные функции — классы функций)

- Планирование
- Разработка
- Обслуживание
- Выпуск документации
- Испытания
- Поддержка
- Сопровождение

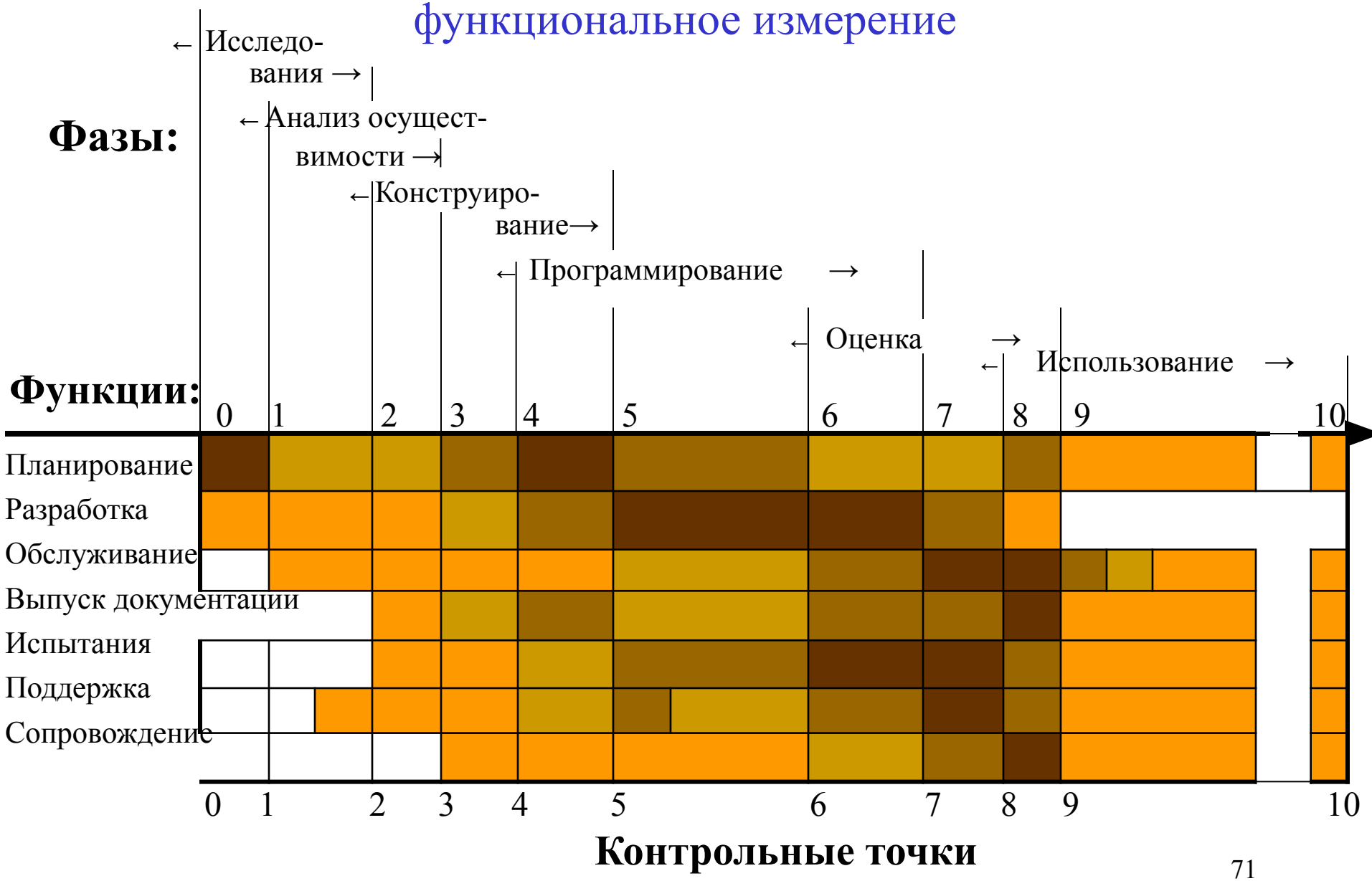
- Классы родственных функций можно считать выполняемыми в течение всего хода развития проекта;
- Содержание (цели) функции на различных этапах претерпевает изменение
- Интенсивность функции меняется как при переходе от этапа к этапу, так и в рамках работ одного этапа

Основной тезис: **На разных этапах функции**
имеют ***различное содержание***,
требуют ***различной интенсивности***,
при реализации проекта ***совмещаются***.

В конкретных проектах это понятие доопределяется (трактуется так, как полезно, например, как потребность или расходование ресурсов).

Модель фазы—функции Гантера:

функциональное измерение



Вариативность модели Гантера

- В зависимости от проекта функции можно трактовать свободно, дополнять другими классами функций, игнорировать некоторые из них и т.д.
- Можно рассматривать не только производственные функции, но и иное, полезное для управления (например, исполнителей проекта, трактуя интенсивность как занятость определенными заданиями)
- *Основной тезис*

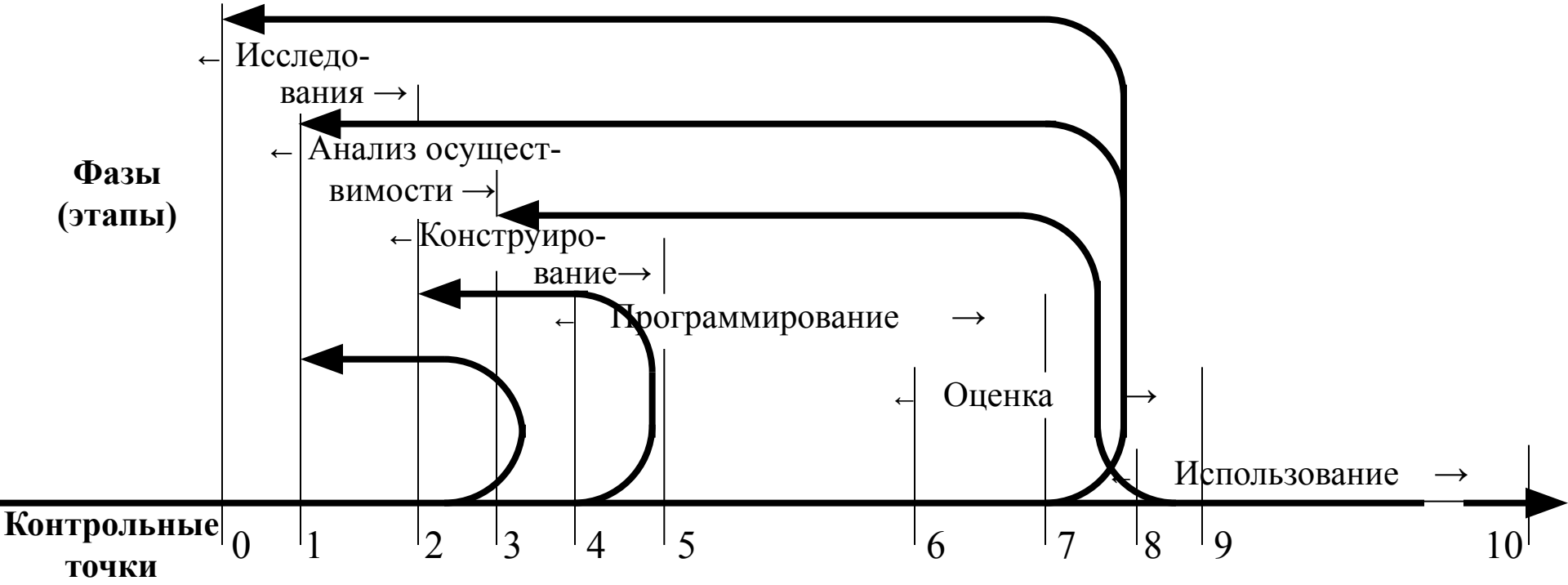
  — основа построен
  которое накладывает
  Матричная л

На разных этапах функции
имеют ***различное содержание***,
требуют ***различной интенсивности***,
при реализации проекта ***совмещаются***.

- Элементы модели можно развивать, сохраняя требуемые связи моделируемой системы

⇒ Возможность превращения модели в инструментальную

Учет итеративности в модели фазы—функции



Расщепление линии развития проекта (жизненного цикла):

1. ***Приостановка процесса*** (в любой момент, если обеспечена корректность слияния) — традиционная реакция на ошибку
2. ***Действительное расщепление*** — появляются два (и более?) процесса.
   Для корректности нужно оценивать ресурсы, планировать новые контрольные точки и определять содержание существующих контрольных точек

- ***Слияние расщепленных процессов*** в случае 2 — должно планироваться!

⇒ Действительное расщепление обязано быть регламентированным!

5. Моделирование жизненного цикла объектно-ориентированных программных проектов

Принципы объектно-ориентированного проектирования

1. **Итеративность развития —** возможность перейти от последовательного развития к стратегии итеративного наращивания возможностей
2. **Изменение функциональности —** пересмотр требований при развитии проекта
3. **Формирование системы понятий проекта** — развивающийся *глоссарий проекта*
4. **Наращивание функциональности в соответствии со сценариями** — реализация выделенных сценариев; последующие итерации реализуют другие сценарии
5. **Ничто не делается однократно** — отказ от завершенности работ классических этапов, повторное прохождение их на новых итерациях (с новым набором сценариев)
6. **Оперирование на размножающихся фазах подобно** — обычные этапы при выполнении любой итерации развития проекта:
   - *Определение требований, или планирование итерации;*
   - *Анализ;*
   - *Моделирование пользовательского интерфейса /новое/;*
   - *Конструирование;*
   - *Реализация (программирование);*
   - *Тестирование;*
   - *Оценка результатов (итерации)*

Вне итераций:

1. **Начальная фаза проекта**: требования, ближайшая и перспективные задачи, критерии оценки результатов;
2. **Фаза завершения проекта**: поставка и сопровождение + выделение переиспользуемых компонентов

Моделирование при объектно-ориентированном проектировании

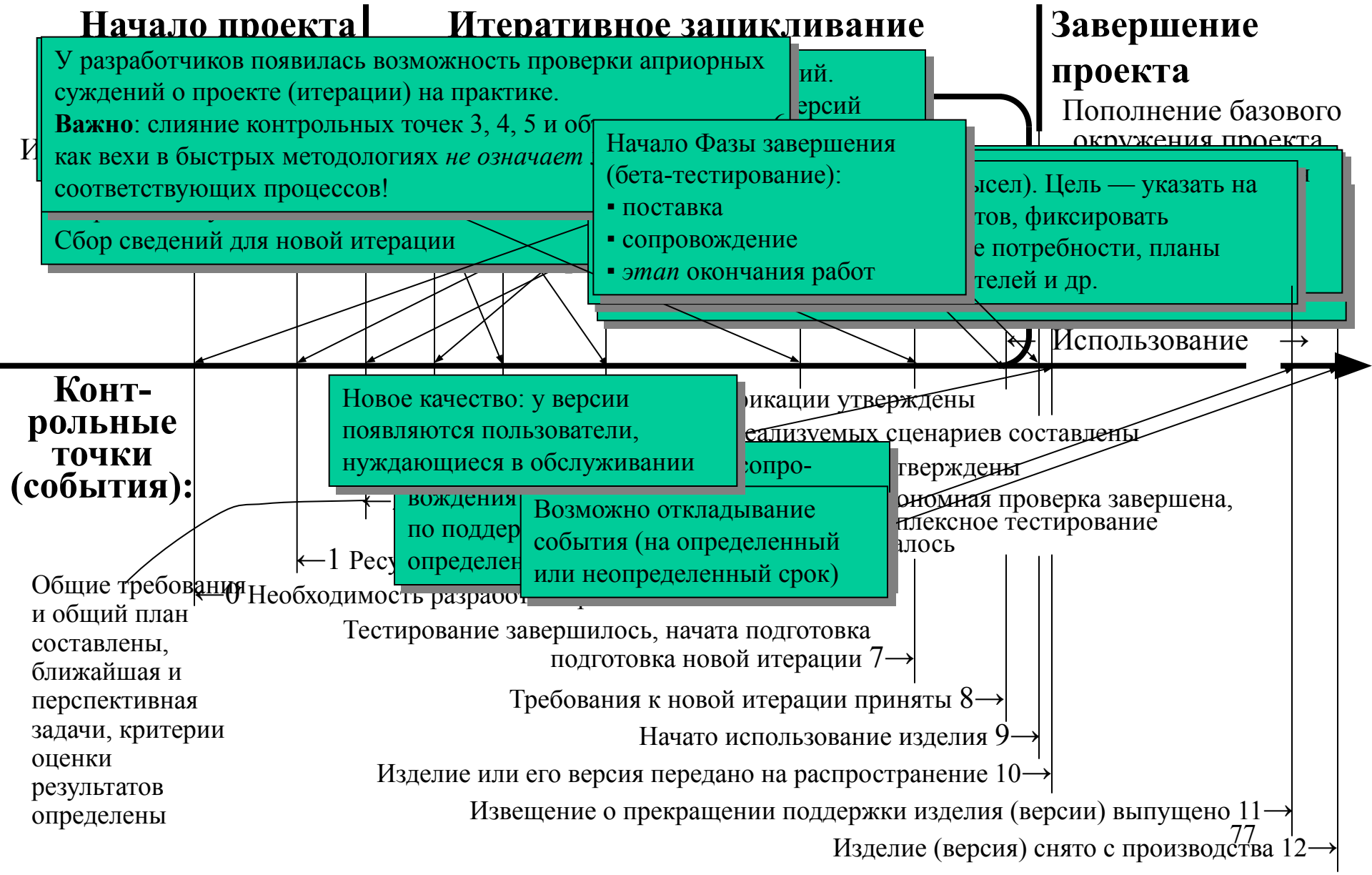
1. **Распределение реализуемых требований по итерациям**: Совокупность сценариев, реализуемых на очередной итерации + набор ранее реализованных сценариев образуют *законченную*, хотя и *неполную версию системы*, предлагаемую пользователям
   — *модели уровня анализа*
2. ***Особый стиль наращивания возможностей системы и ее развития***:
   Основа декомпозиции проекта при ООП подходе — набор связанных различными отношениями классов; новая итерация расширяет этот набор. Это расширение строится на базе построения
   — *моделей уровня конструирования*

Моделирование — *организационно-техническая* (*производственная*) *функция* всего процесса развития проекта, а не один из этапов!

Следствие:

Пополнение базового окружения проекта — дополнительный этап (вложенный в оценку), содержание которого — анализ возможного переиспользования накапливаемых компонентов ПО как для проекта, так и для будущего

Жизненный цикл при объектно-ориентированном развитии проекта (фазовое измерение)



Контрольные точки и вехи

- ***Контрольные точки*** (**check points**) — точки линии жизни жизненного цикла проекта, в которых возникают определенные события. Эти события рассматриваются как *существенные*, поскольку их необходимо отслеживать с целью управляемого развития проекта (такого, которое оставляет траекторию в рамках области допустимых операционных маршрутов)
- ***Вехи*** (**mail stones**) — это контрольные точки, прохождение которых сопровождается определенными планируемыми мероприятиями. Без успешного (результат соответствует цели) проведения таких мероприятий, прохождение *вехи* блокируется с целью выполнения активностей, направленных на исправление ситуации.
- ***Планирование*** получения ***результата*** и ***оценка полученного результата*** — основное содержание деятельности, связанной с вехами
- ***Конкретизация контрольных точек и вех*** — существенная задача, которую приходится решать в рамках выполнения функции планирования. Эта *конкретизация* делается на основе знания специфики выполняемого проекта и процесса его выполнения (т.е. приятой для проекта методологии). Специфика проекта и процесса определяет необходимость и количество вех.
- В *жестких методологиях* к прохождению *вехи* приурочивается *утверждение* соответствующих ей *рабочих продуктов*, в том числе и документов;
- В *быстрых методологиях вехи* служат лишь *ориентирами продвижения* в своем развитии (мероприятия не имеющие отношения к процедурам утверждения)

Общие требования, общий план, ближайшая и перспективные задачи

Для каждой итерации должны быть определены:

- *Общие требования* — что требуется от проекта в целом в данный момент
- *Общий план* — как предполагается достигать цели (стратегия)
- *Ближайшая задача* — набор конкретных реализуемых требований и сценариев ← критерии предпочтения того, что планируется реализовывать
- *Перспективные задачи* — те, которые рассматриваются (в данный момент) как основа для планирования дальнейших итераций (в проектах жесткой отчетности)
- ***Критерии предпочтения***:
  1) Актуальность для пользователя
  2) Полнота и функциональная замкнутость предлагаемых средств
     - Функциональная полнота
     - Реализационная полнота
     - Интерфейсная полнота
  3) Системная значимость (внутрипроектные предпочтения)
  4) Демонстрационная значимость
  5) Скорость реализации

Характеристики

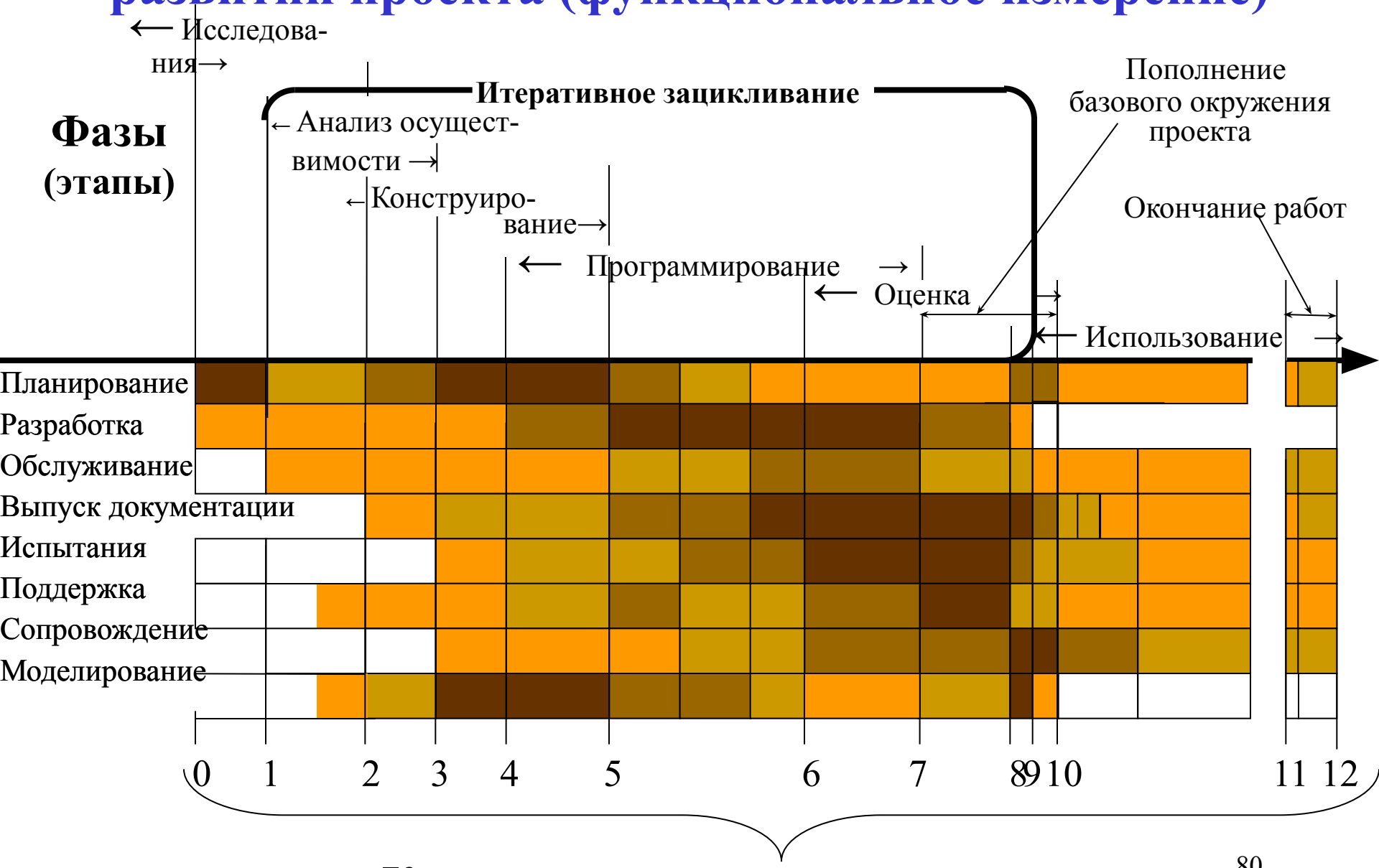
1) Актуальность для пользователя	Всегда лучше то, что актуально!
2) Полнота и функциональная замкнутость предлагаемых средств	Автоматизация деятельности в целом. Растет по мере увеличения объема уже выполненных работ
3) Системная значимость (внутрипроектные предпочтения)	Реализационные предпочтения. Конкурирует с (1). Более значим для начальных итераций
4) Демонстрационная значимость	Конкурирует с (1), (2) и (3) Максимально значимо для начальных итераций
5) Скорость реализации	Лучше то, что быстрее. Если время фиксировано, то для реализации определяется пул работ.

Возможные ограничения: время, объем работ, затраты (треугольник менеджмента

Иногда время не критерий, а ограничение

Минимизация реализуемого является критерием лишь для некоторых методологий!

Жизненный цикл при объектно-ориентированном развитии проекта (функциональное измерение)



Непрерывность поступления требований в моделях жизненного цикла

Трассировка требований (в модели Гантера)

- Варианты поступления требований:
  - требование или группа требований обрабатываются до начала итерации (при разработке ее сценариев)
  - требование или группа требований поступают, когда работы итерации начались
  - требование или группа требований поступают, когда релиз системы передан в эксплуатацию
- Возможные результаты анализа требований:
  - *требование отклоняется* — работа с требованием прекращается
  - *требование принимается к реализации на текущей итерации*
  - *реализация требования откладывается до следующих итераций*

Укладывается в представленную ранее схему модели фазы – функции

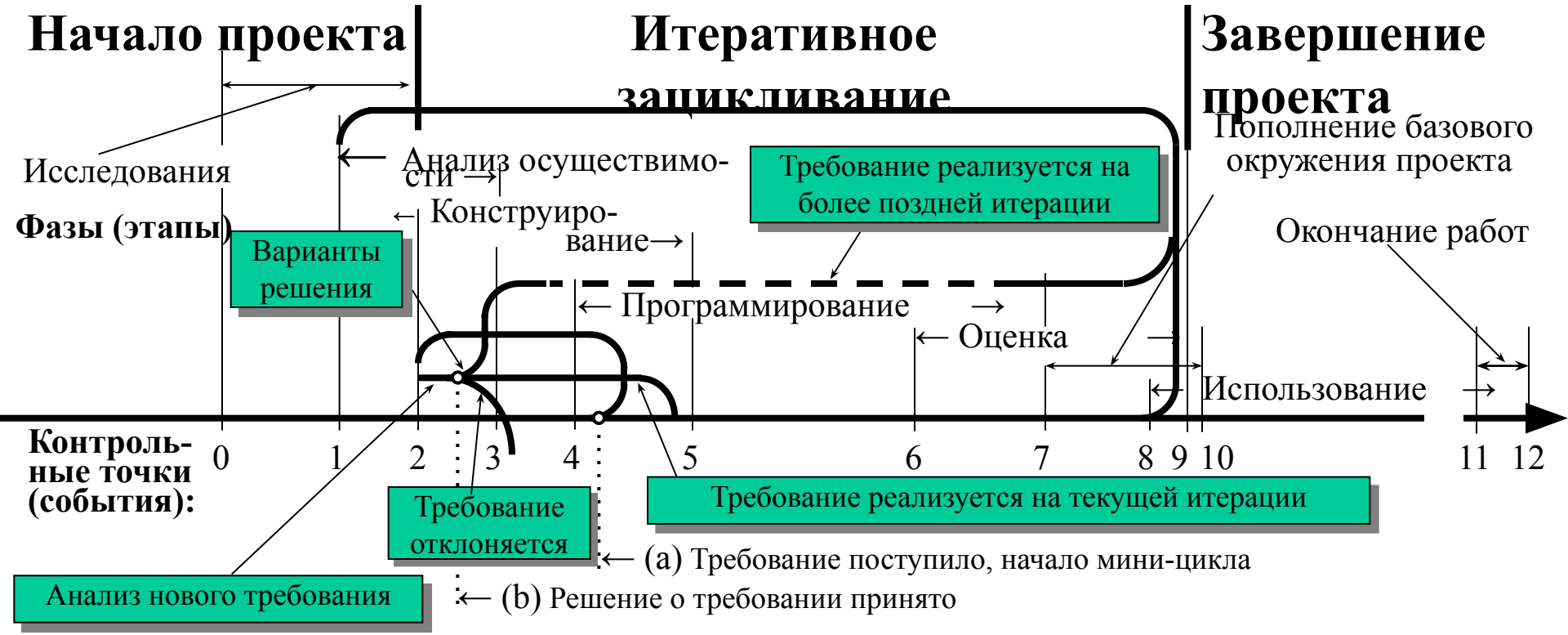
Схема дополняется мини-циклом обработки требований

До мини-цикла необходим предварительный анализ требований

Функциональное измерение меняется, но учесть это вне контекста конкретного проекта нереально

Почти все модели жизненного цикла слабо приспособлены к учету непрерывности поступления требований

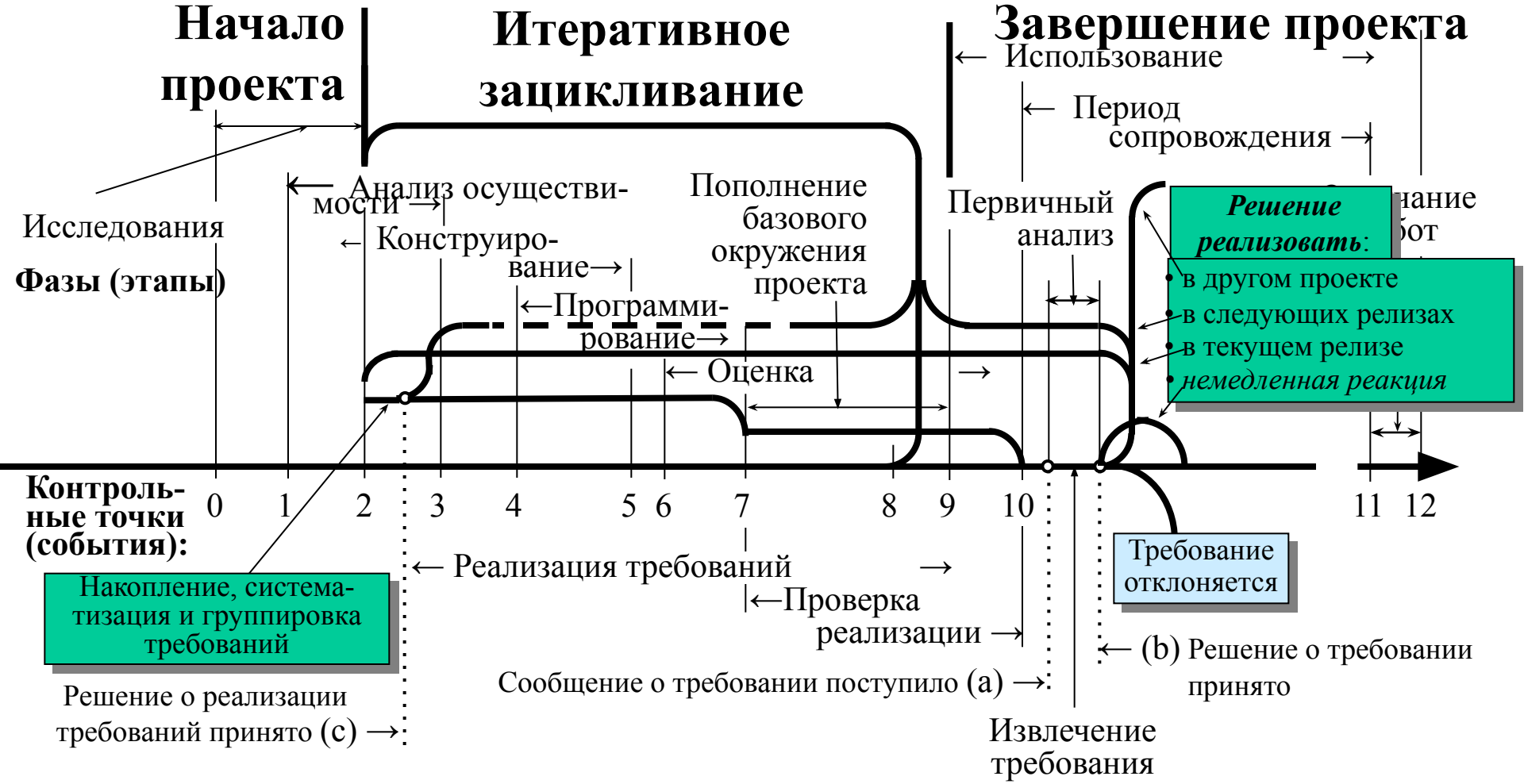
Жизненный цикл при объектно-ориентированном развитии проекта (фазовое измерение)



Шаги обработки требования или группы требований:

- *поступление* — в любой момент конструирования, программирования или оценки
- *расщепление*, переход к анализу
- *анализ*
- *принятие решения* /на общем участке этапов анализа и конструирования/
- *планирование* срока или будущей итерации реализации

Требования, поступающие в ходе эксплуатации



Шаги обработки требований

- ***Поступление сообщения***, содержащего требования — в любой момент периода сопровождения
- ***Первичный анализ*** — принятие первоначального решения о реализации:
  - ***немедленная реакция*** — быстрое устранение замечаний, пояснения для пользователей и др. Выполняется всегда, в том числе совместно с другими решениями
  - ***требование отклоняется*** — объяснение причин отклонения, путей преодоления трудностей
  - ***реализация требования в текущем релизе*** — если претензии обоснованы, а устранение замечаний, ошибок и т.п. возможно в обслуживаемом релизе, то организуется мини-цикл обработки требований в итерации
  - ***реализация требования в одном из следующих релизов*** — если устранение замечаний в рамках обслуживаемого релиза невозможно или нецелесообразно, то требование передается для исполнения на одной из следующих итераций проекта
  - ***реализация требования в другом проекте*** — если выясняется, что в данном проекте требование реализовать невозможно или нецелесообразно, то, быть может, оно станет одним из аргументов в пользу организации нового проекта
- ***Мини-цикл обработки требований*** начинается с *анализа* (возобновленный процесс), определяющего группу требований для реализации в текущем релизе в рамках обычных задач этапа
- ***Реализация*** отобранных ***требований*** на данной итерации осуществляется по обычной схеме: *конструирование, программирование, оценка*. Особая роль проверочных работ — дополнительный этап ***проверки реализации***. Обязательно повторение проверки того, что было отлажено ранее (пополнение тестовой базы)
- ***Распространение изменений*** — сделанные исправления должны стать доступными для всех пользователей. При массовом применении эта работа может потребовать значительных ресурсов

Действия, связанные с новыми проектными требованиями

- Требования, возникающие и изменяемые в течение этапов итерации, разделяются на ***принимаемые*** и ***отвергаемые***
- Для каждого *отвергаемого требования* составляется ***мотивированное заключение*** о том, почему оно не принимается:
  - невозможно удовлетворить,
  - нецелесообразно принимать (достижимо иным путем),
  - запланировано в качестве перспективы,
  - может быть принято при изменении финансовых и календарных планов и др.
- *Заключение* согласовывается с автором требования и с заказчиком
- Для каждого из *принимаемых требований* (их элементарных составляющих) определяется, ***когда оно может быть удовлетворено*** и ***когда его целесообразно удовлетворять***. Критериями служат:
  - приоритетность требования;
  - сложность рабочих продуктов и зависимостей рабочих продуктов от требования;
- *Простые требования* реализуются непосредственно ***в момент утверждения***;
- *Сложные требования* ***откладываются до завершения конструкторских работ итерации***, которое рассматривается как начало работ по учету комплекса предъявленных требований;
- *Требования, откладываемые до последующих итераций*, реализуются ***согласно общему плану проекта*** (корректируемому)
- Учитывается, что **ранее принятое требование может оказаться отвергнутым** вследствие принятия новых требований → необходимо подготовить и согласовать с автором требования и заказчиком ***заключение об отказе от реализации требований***,
- **Изменения планов и объемов работ, возникающие и/или планируемые в связи с изменениями требований, всегда согласуются с заказчиком**.

6. Технологические аспекты развития программных систем в моделях жизненного цикла

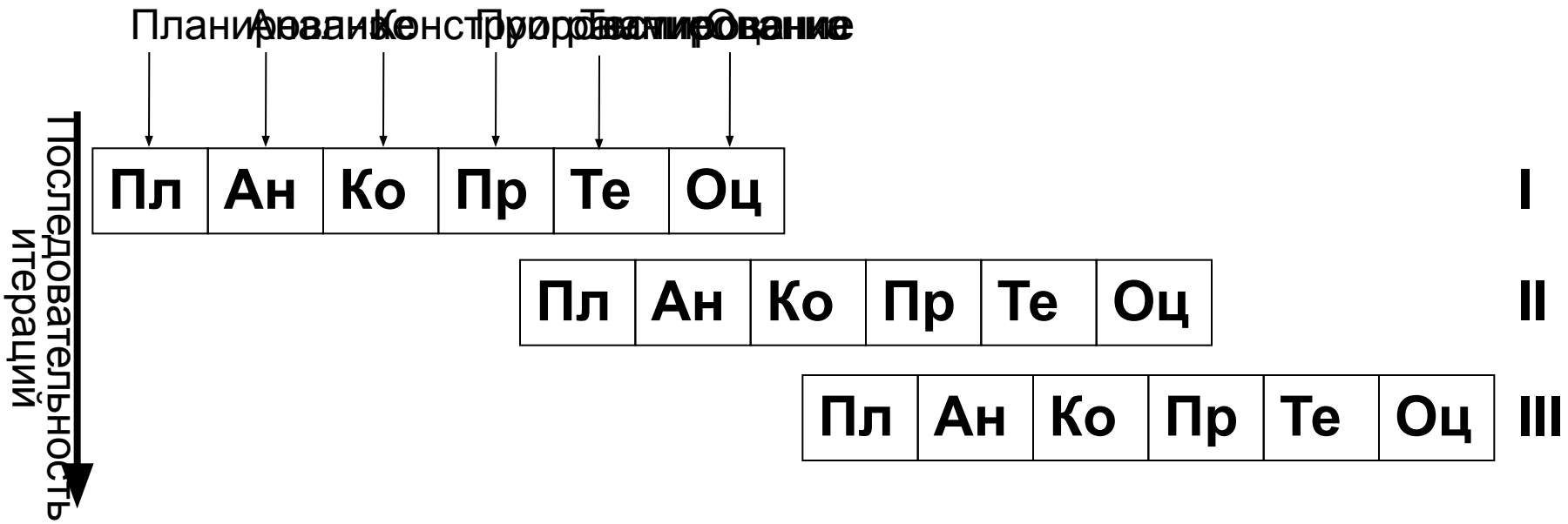
Модели жизненного цикла и задачи методологий разработки проектов

1. **Первая задача**: выяснить
   - способность моделей жизненного цикла отражать технологичные свойства процесса производства программного обеспечения, например, распараллеливание производственных операций и их распределение между исполнителями
   - возможность использования моделей жизненного цикла, согласованного с реальными процессами менеджмента и с другими инструментами, поддерживающими эти процессы
2. **Вторая задача**: показать возможности параллелизма выполнения проектных заданий

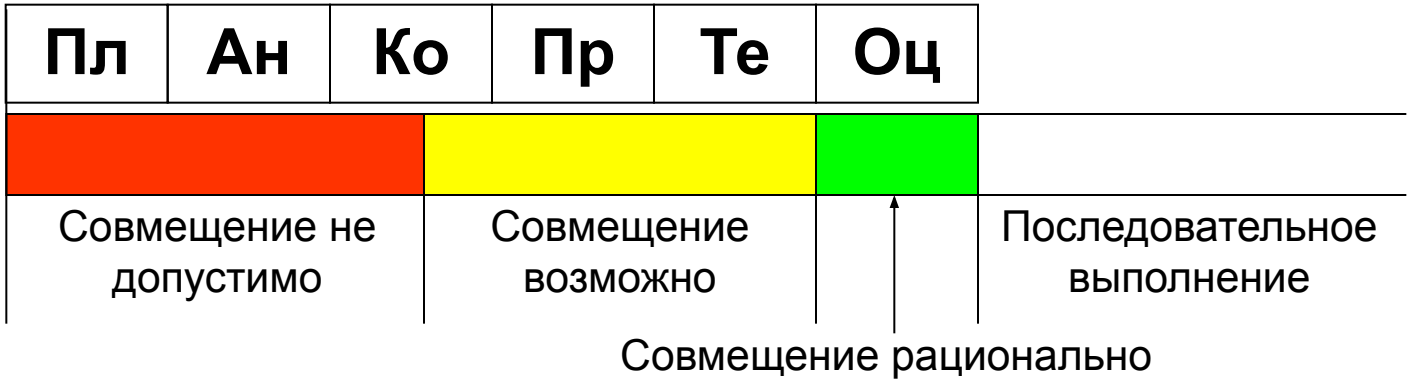
   Коллективный процесс разработки всегда подразумевает параллельное выполнение деятельностей:
   - совместная работа на смежных этапах (отмечалась ранее)
   - функциональное измерение (отражено в модели Фазы — функции)
   + параллельное выполнение деятельностей в рамках отдельных этапов (естественное требование к процессу разработки — обычно только про него и говорят)

   Итеративные схемы допускают еще один вид параллелизма:
   - одновременная разработка нескольких итераций (в известных пределах)

Параллельное выполнение итераций



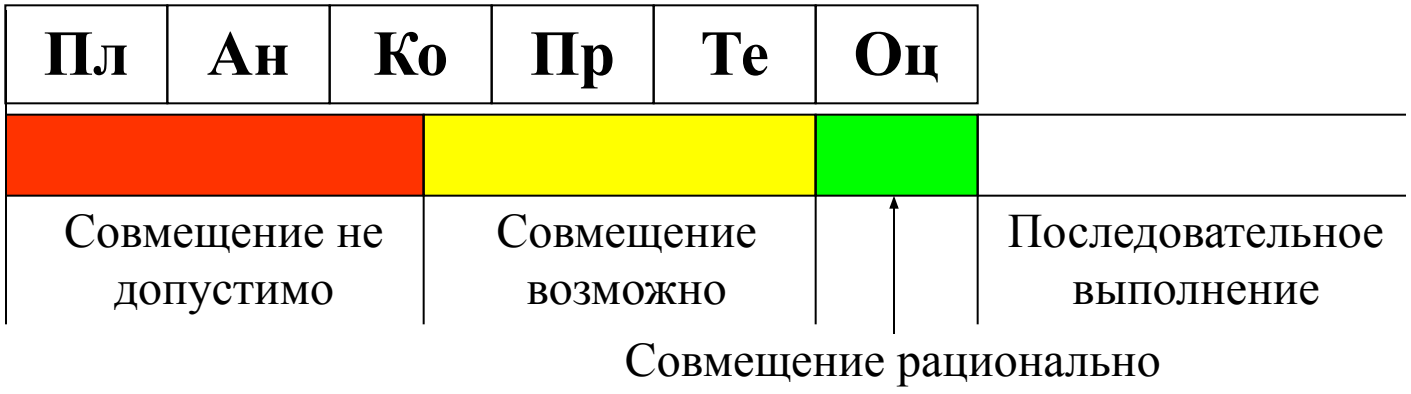
Пределы совмещения итераций в проекте



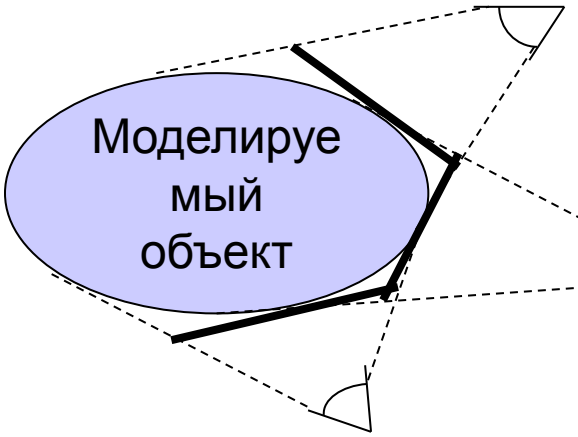
Пределы совмещения итераций в проекте

- ***Область недопустимого совмещения***: когда выполнение одной работы непосредственно зависит от результатов другой
- ***Область возможного совмещения***: когда зависимость ослаблена тем, что ожидаемые результаты предшествующей работы хорошо описаны (например, построены и проверены модели этапов конструирования, хотя программирование еще не выполнено)
- ***Область рационального совмещения***: когда зависимость работ фактически тем или иным способом экранирована (предшествующая работа выполнена, хотя, быть может, не до конца проверена, составлен и проверяется протокол взаимодействия работ и др.)

- ***Недопустимость совмещения*** означает, что для планирования очередной итерации нет достаточно полной информации, следовательно, оно не может быть выполнено эффективно.
- Когда такая информация появляется, появляется и ***возможность активизации работ над новой итерацией***
- ***Рациональное совмещение*** означает надежность параллельной работы**.** Разумнее всего начинать этап программирование новой итерации, когда рабочий продукт предыдущей итерации протестирован



Модели процесса и продукта (напоминание)



Модель процесса разработки:

- **Целенаправленное развитие объекта** под воздействием разработчиков
- **Ключевые понятия**: развитие, система деятельностей, субъект ⇔ объект, этапы деятельностей, инструменты деятельности, цели, результаты и продукты

Модели продукта (различные)**:**

- **Как устроен** (построен) продукт? **Для чего нужен**?
- **Один из типов моделей продукта**: проекция модели процесса, в которой игнорируется все, связанное с субъектом
  возможна реконструкция модели процесса, которая необязательно совпадает с исходной моделью процесса!

Иллюстративность модели: явное выделение некоторых аспектов для облегчения их понимания

Инструментальность модели: использование ее в качестве инструмента некоторой деятельности (т.е. способствует целенаправленному развитию).

Нельзя смешивать иллюстративные и инструментальные модели

Вопросы в связи с моделью: **Что будет, если…?** и **Соответствует ли…?**

Требования к инструментальной модели жизненного цикла

Модель должна:

- давать картину разработки и развития проекта (уровни организации планирования процесса для определения графика работ, для отслеживания их ресурсной обеспеченности и др.);
- давать средства декомпозиции процесса разработки, т.е. согласованного разбиения этапов на вложенные этапы и работы (поддержка планирования);
- обеспечивать переход от этапов к работам этапов и доступ к истории;
- позволять видеть текущее состояние проекта и варианты развития;
- **позволять *оперировать своими элементами*, а через это — влиять на ход моделируемого процесса выполнения проекта**

Однако это может приводить к потере наглядности модели

Способ сгладить противоречие — ориентация на определенные типы жизненного цикла (отказ от универсальности моделей)

Относительность качества инструментальности модели ⇒ параметры оценки, нормирование оценки. *Качественное* (не количественное) оценивание

Реальная и принципиальная возможность использования модели

Целесообразно рассматривать принципиальную возможность инструментального использования моделей

Параметры оценки инструментальности

Из ранее рассмотренных моделей только
строгая каскадная модель и
матрица Фазы — функции
могут претендовать на то, чтобы их можно было считать инструментальными:

1. ***Расширяемость*** (дополнительные блоки, вложенные этапы и др., расщепление линии жизни)
2. ***Атрибутивность*** матрица Фазы — функции выше, чем у каскадной модели (функциональное измерение)
3. ***Масштабируемость*** каскадной модели более развита, чем у матрицы Фазы — функции
4. ***Интегрированность*** — выше у каскадной модели, но принципиально возможна для обеих моделей

Сравнение инструментальности различных моделей

- Календарный план
- Диаграммы Гантта
- Каскадная модель
- Спираль развития Буча
- Инструментальная спиралевидная модель Боэма
- Модель RUP
- Модель процессов MSF
- Модифицированная модель Гантера (матрица фазы—функции)

Календарный план

Календарный план — документ, с помощью которого устанавливаются юридические отношения, касающиеся объема, сроков и (зачастую) ресурсных потребностей выполняемых работ, между всеми участниками разработки проекта, включая и заказчиков, и планировщиков.

Рубрикаци
прорабо
Возмож
проекта

Целесооб
самое
самый

Кто
Сро
(
Под

От каких поставок ресурсов зависит работа
Подписи ответственных «Ознакомлен»

Вн

Раз

урезанная до цикла разработки

Попытка охватить все аспекты, которые нужно учитывать при выделении работ и отслеживании сроков их выполнения

Наименование работ (тема, этап, работа, задача, задание)	Сроки выполнения начало/конец		Ответственный исполнитель и исполнители, роли	Требуемые ресурсы и сроки их предоставления план/факт	Примечания
	план	факт			
1	2	3	4	5	6

Календарный план: обсуждение

Удобен:

- Верхний уровень рубрикации должен совпадать с тем, что задается в техническом задании
- Дополнение новыми рубриками (в том числе в процессе выполнения) не вызывает трудностей
- Наглядность

Недостатки:

- Тенденция к разрастанию
- Неприспособленность к учету загруженности сотрудников, потребностей и к перераспределению ресурсов.
- Рубрикация противоречит распараллеливанию работ
- Трудно увидеть показатели на определенный момент времени
- Непригодность для отражения итеративности развития проекта

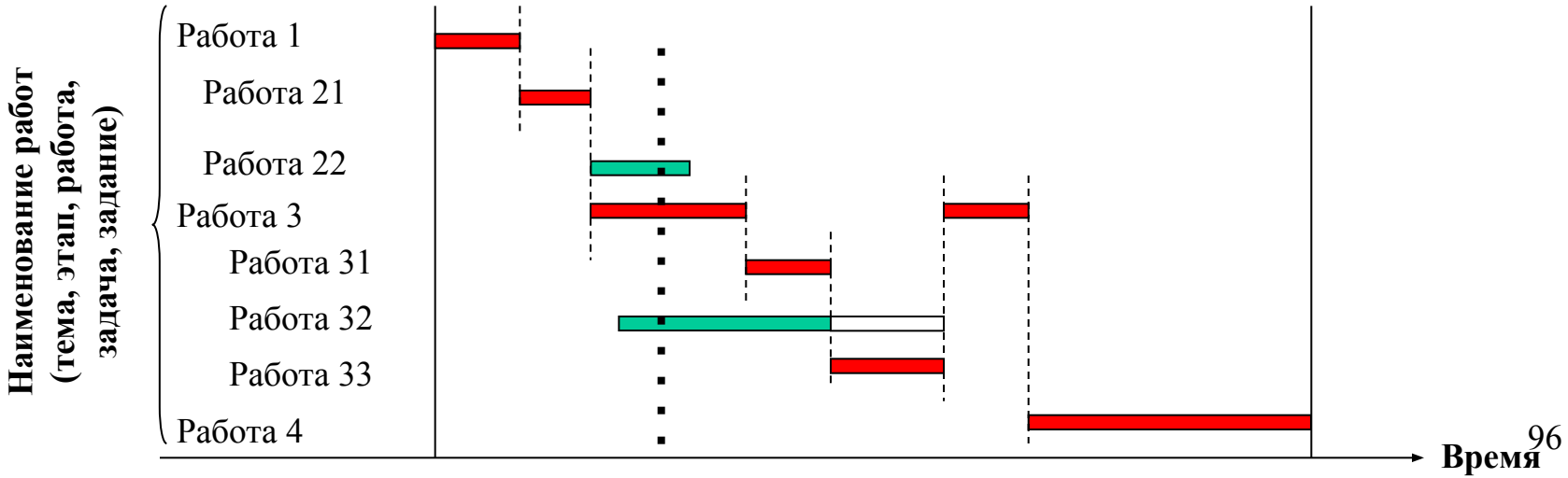
Оценка инструментальности:

- ***Атрибутивность*** относительна (рост числа отражаемых атрибутов ⇒ снижение наглядности)
- ***Расширяемость*** — одно из основных преимуществ
- ***Масштабируемость*** — слабое место
- ***Интегрированность*** с другими инструментами — возможна

Развитие /и ограничение!/ **календарного плана** — диаграммы Гантта как основа реальных инструментов поддержки управления проектами

Диаграммы Гантта

- **Критерии инструментальности 1-4 выполняются** (почти достаточно)
- Попытка отражать время, однако есть проблема расщепления времени, когда есть оперирование параллельными работами
- Для независимо действующих процессов нужно рассматривать *множественное время* + *связи времен* (синхронизация и др.) ⇒ *время как частично упорядоченное множество событий*
- MS Project — пример диаграмм Гантта. Это максимум того, что можно поддержать с помощью *универсального инструментария*
  // PERT-диаграммы (работы — дуги, события — вершины)
  используются реже (хуже отражают времена работ) //
- Скептическое отношение к моделям жизненного цикла вообще, исходящим из представления диаграмм Гантта (Брукс).

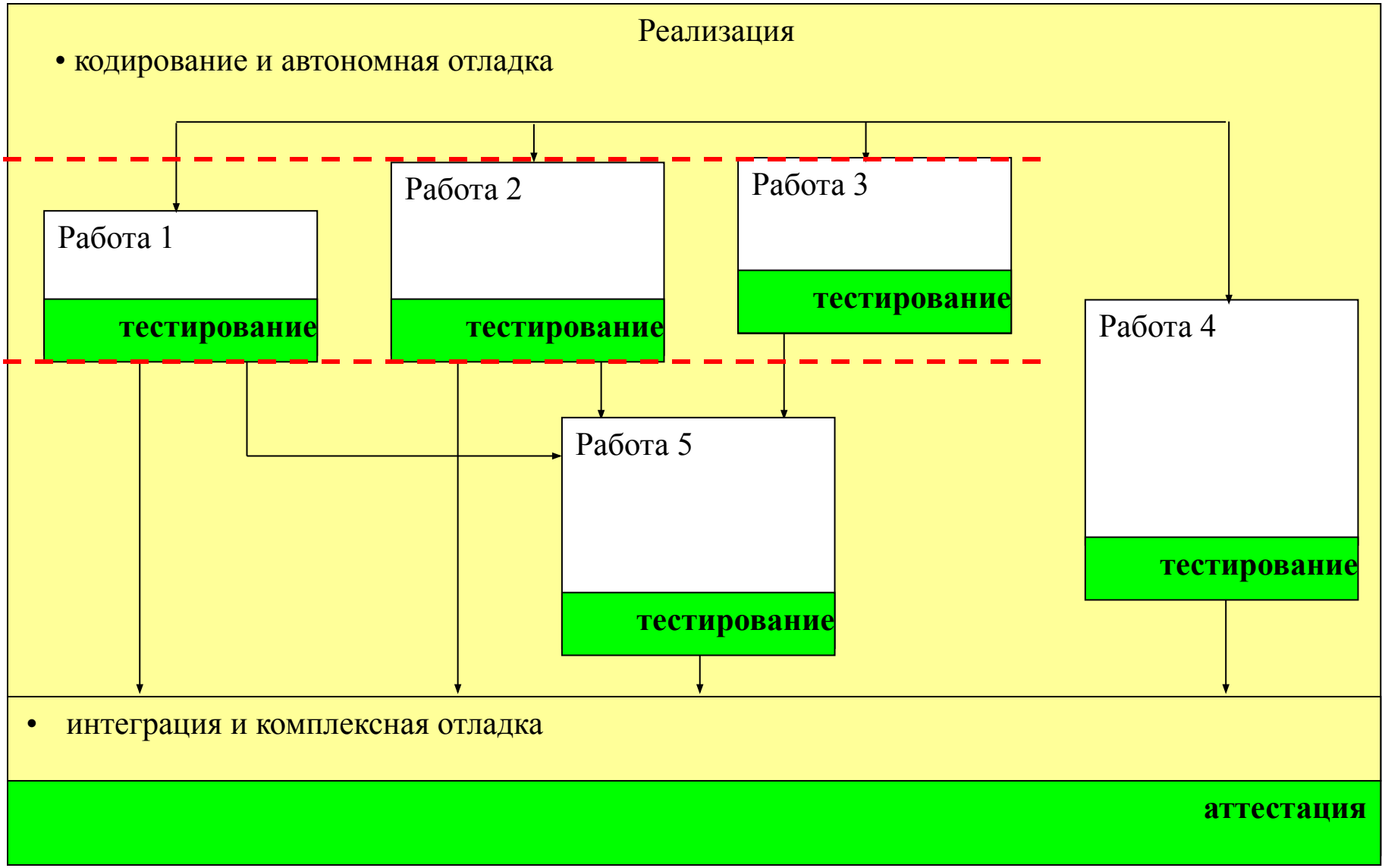


Каскадная модель

Каскадная модель: ограниченность

- Схема последовательного (а не итеративного) развития проекта
- Ограниченные возвраты (ошибки прошлых этапов)
- Ориентация на вполне определенную методологию развития проекта (контроль прохождения этапов — барьеры)
- **Блоки каскадной модели можно раскрывать внутрь: дробление процесса на задачи, работы и др. возможно.** При этом:
  - принципиально возможно вводить новые блоки и связи (*расширяемость*)
  - вертикальная ось времени
  - синхронизация

Каскадная модель: декомпозиция процесса на задачи, работы и др. (иллюстрация)



Каскадная модель: оценка инструментальности

Расширяемость достигается как элемент выбранной методологии

Атрибутивность относительна: показ дополнительных атрибутов может снижать наглядность

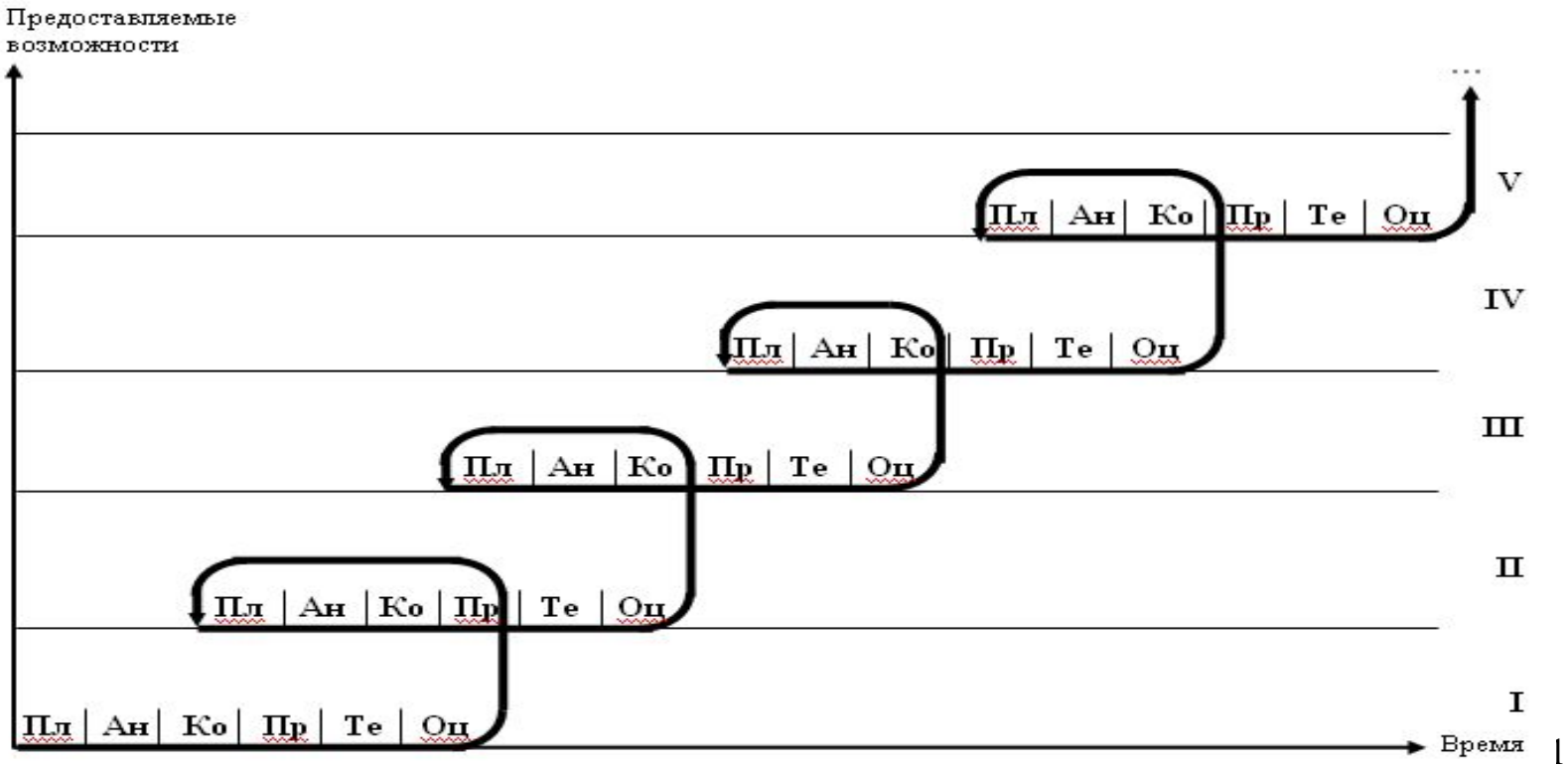
Масштабируемость слабая: переход на другой уровень (вверх и вниз), но для выбранной методологии этого достаточно

Интегрированность принципиально возможна (для выбранной методологии)

Инструменты поддержки каскадной модели представлены среди CASE-средств (чаще — фрагментарно, но приемлемо)

Спираль развития Буча

- Вариант Буча — чисто иллюстративная модель
- Модификации:
  - Изображение итеративного зацикливания
  - Система координат «время —предоставляемые возможности»
  - Изображение стандартных и, возможно, дополнительных этапов



Спираль развития Буча: обсуждение

- Как и у Буча, возможности, предоставляемые итерацией, никогда не отменяют ранее достигнутого уровня
- Можно показывать и отслеживать параллельное выполнение итераций
- Детализированное выделение этапов нарушает наглядность — *относительная расширяемость*
- Слабо приспособлена для отражения этапов внедрения
- Согласуется с детализацией верхнего уровня декомпозиции ⇒ для инструментальности нужна автоматизация перехода к другим уровням (к отдельной итерации)

Масштабируемость в принципе достижима

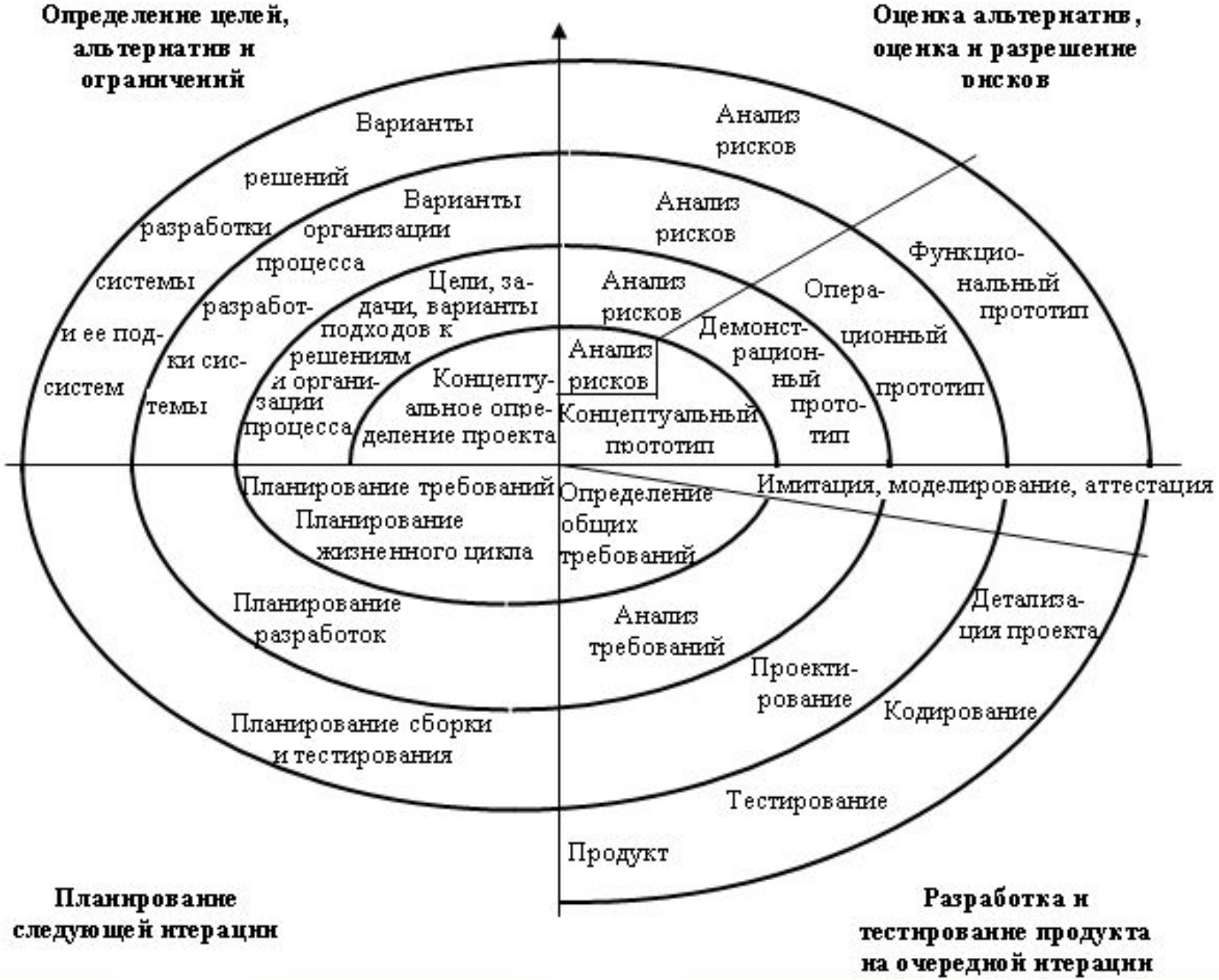
Атрибутивность — на уровнях отдельных итераций

Интегрированность в принципе достижима

Необходимая ***расширяемость*** достижима

Это уже другие модели

Инструментальная спиралевидная модель Боэма



Инструментальная спиралевидная модель Боэма: обсуждение

- Модель проработана с точки зрения процессов производства программ
- Возможна настройка модели на конкретные методологии.
- Модель явно указывает на действия, которые требуется выполнять при движении по спирали.
- ***Расширяемость*** — основное достоинство
- ***Масштабируемость*** — не очень существенна (взамен — движение по спирали)
- **Вполне определенная методика работы**
- Конкретное планирование действий витков — за рамками модели ⇒ ***атрибутивность*** вне модели
- ***Интегрированность*** в принципе достижима

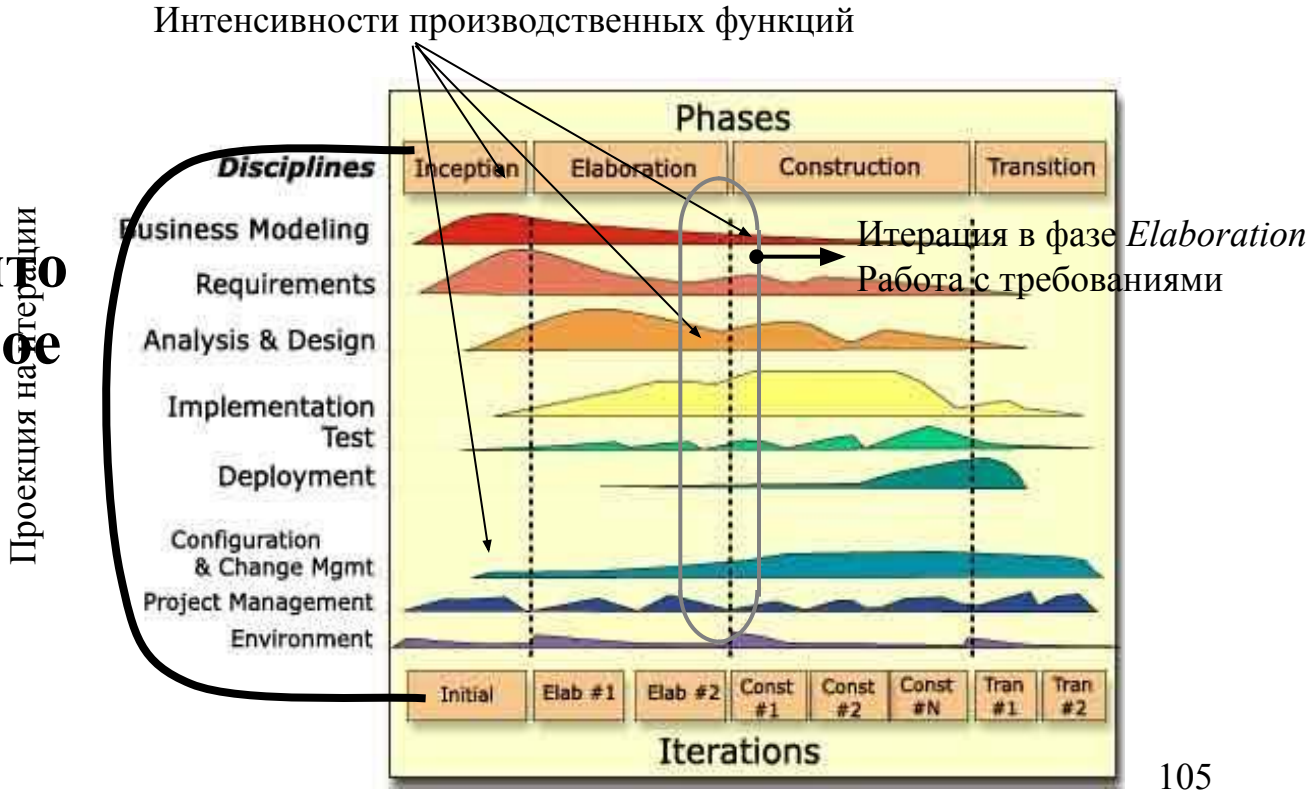
Недостатки:

- плохо отображаются временные соотношения между сроками выполнения работ на разных витках
- плохо приспособлена к учету и распределению ресурсов

Модель RUP

- 51% программных разработок применяют RUP
- RUP претендует на роль универсальной основы любых программных разработок
- Модель задается в виде матрицы интенсивностей функций, выполняемых на этапах (фазах), которые проецируются на итерации

Модель *выглядит как универсальная схема*:
она отражает то, что включается в любое производство программ



Модель RUP: обсуждение

- Не конкретизируются виды работ на этапах
- Время условно
- Возможность совместного выполнения некоторых производственных функций не отражается
- Включение дополнительных этапов и функций, отражение специфики конкретного процесса или коллектива затруднительно (**это нарушило бы фиксированную связь между жизненным циклом по RUP с моделями уровня проектирования**)

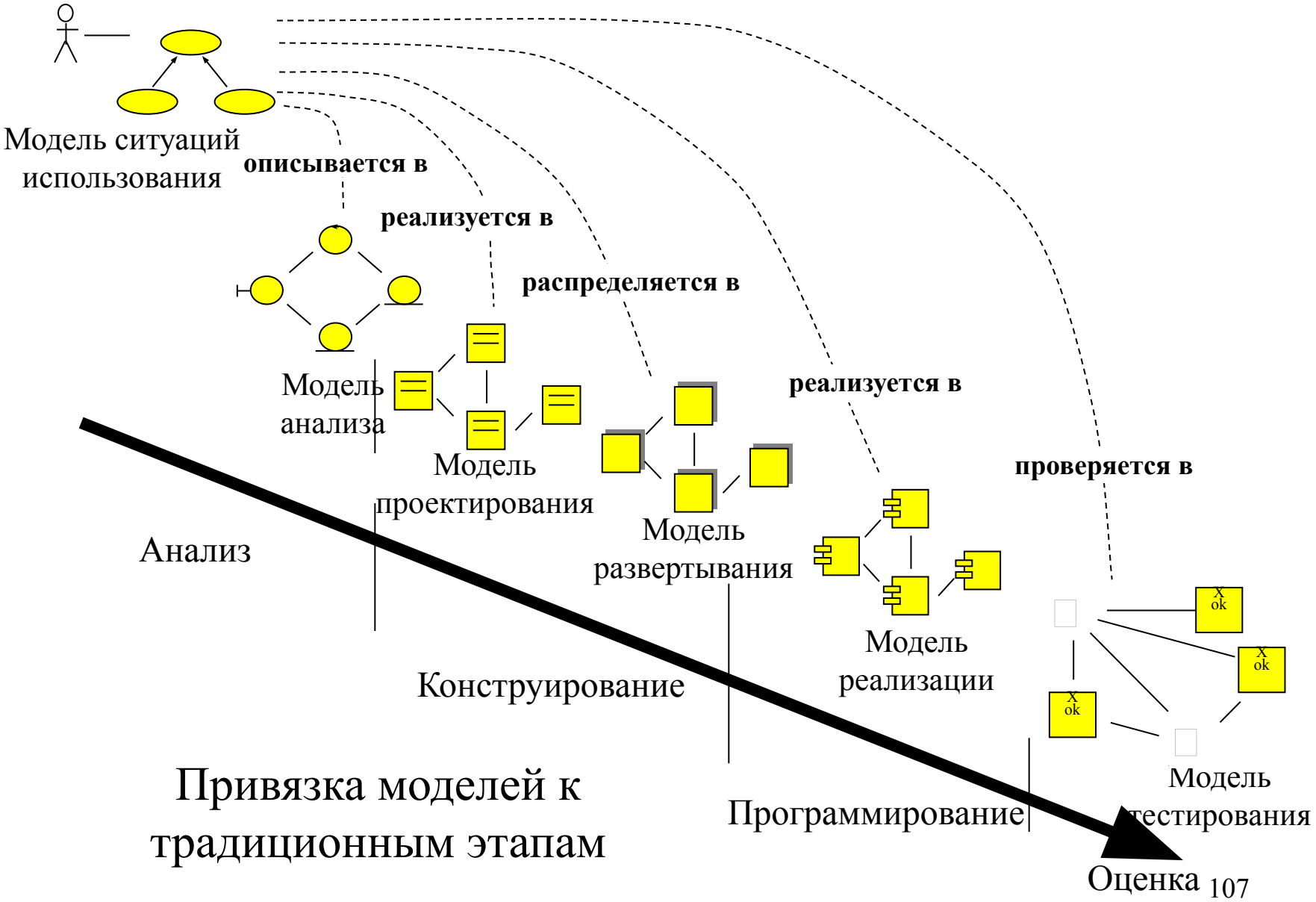
Конкретизация модели разрушает универсальность

Средства моделирования — элементы языка UML, а не инструменты фиксированной методологии

Предполагаемый выход: множество стандартных ситуаций, в которых можно воспользоваться предлагаемыми средствами

Стремление RUP к универсальности привело к иллюстративной модели жизненного цикла и к появлению инструментов и методов их применения (весьма полезных!), не связанных с моделью жизненного цикла

Система моделей RUP



Модель процессов MSF

Цитата:

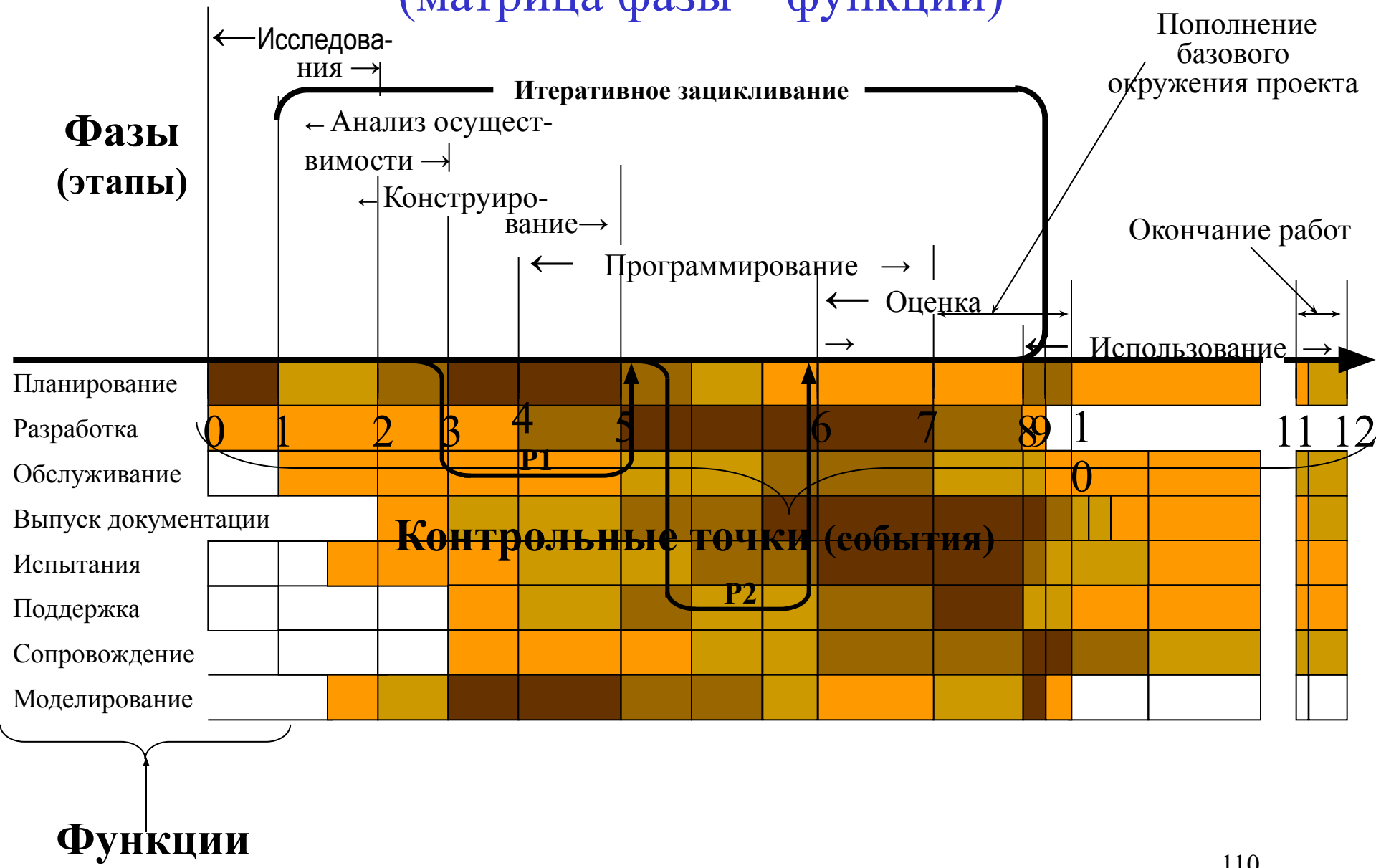
«Модель процессов объединяет в себе лучшие принципы каскадной и спиральной моделей. Она сохраняет преимущества упорядоченности каскадной модели, не теряя при этом гибкости и творческой ориентации модели спиральной, учитывает необходимость постоянного пересмотра, уточнения и оценки проектных требований, стимулирует активное взаимодействие между проектной группой и заказчиком, который оценивает ход и результаты работы на протяжении всего проекта»

Модель процессов MSF: обсуждение

- Стремление к универсальности приводит к огрублению ситуации в конкретных случаях и к необходимости **словесного дополнения схемы** (что и сделали авторы MSF)
- Невозможность отслеживания временных соотношений
- Трудности дополнения специфичных этапов
- Нет механизмов задания оперирования ресурсами и контроля их использования (слабая атрибутивность)

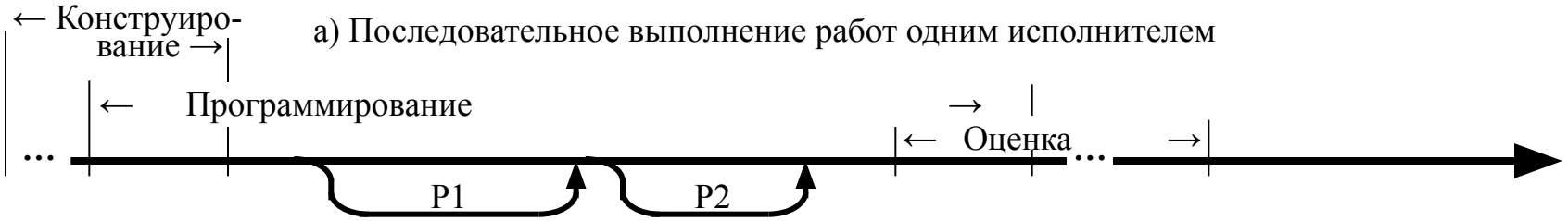
Модель является лишь иллюстративной

Модифицированная модель Гантера (матрица фазы—функции)

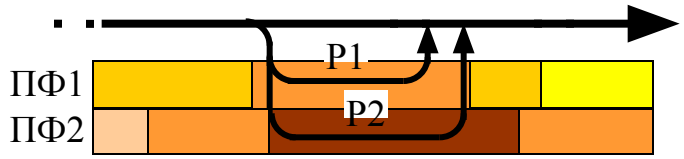


Распространение идеи расщепления на функциональное измерение

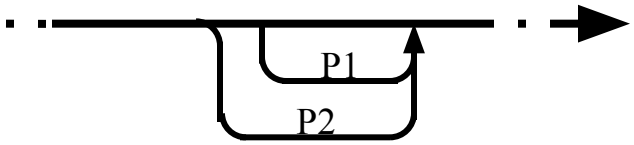
Модифицированная модель Гантера: «азбука» шаблонов



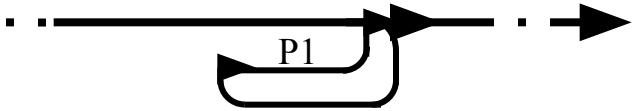
б) Одновременно начинающиеся работы



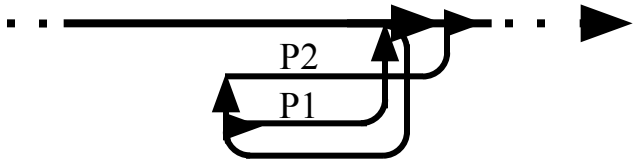
в) Одновременно завершающиеся работы



г) Зацикливание работы



д) Параллельные и зацикливаемые работы



е) Откладывание выполнения работы

ж) Раннее завершение выполнения работы

Дуги работ могут размещаться на функциональном измерении!
Т.е. относится к тем или иным производственным функциям

Модифицированная модель Гантера: оценка инструментальности

Расширяемость достигается за счет шаблонов

Атрибутивность очень высокая:

показ производственных функций, их интенсивностей,
возможность добавления новых функций +
перекрывающиеся этапы +
размещение работ на функциональном измерении + …

Масштабируемость слабое место: нуждается в дополнительной проработке способ показа уровней (итерации, работ и пр.)

Интегрированность **с разными инструментами** вполне возможна

Модель Гантера — **одна из возможных нотаций** (а не «универсальная» методология). Это **язык** схем жизненных циклов, допускающий адекватную инструментальную поддержку

Пример нестандартного применения: вместо функций можно задавать наименования рабочих групп (распределение работ по группам)

Вопрос: А нужно ли это для управления проектами?

Ответ: Возможно и другое, но не менее гибкое средство (язык!)

Итоги

- **Универсальность модели (т.е. пригодность для отражения всех жизненных циклов) противоречит инструментальности.**

  Надо ориентироваться на типовые жизненные циклы

- **Иллюстративные модели можно рассматривать как основу построения инструментальных моделей лишь в редких случаях (следствие предыдущего)**
- **Специальные средства часто поддержаны инструментально (ER-диаграммы, IDEF-диаграммы и диаграммы классов RUP),**

  но обычно это модели *продуктов*, а не *процессов*!

- **Надо различать**

  a) ***Информирующие*** **— получение сведений о ходе развития,**

  b) ***Направляющие*** **— получение и оценка вариантов развития,**

  c) ***Контролирующие*** **— автоматизация контрольных функций**

  виды модели со своими инструментами для каждого из вариантов типов жизненных циклов

- **Для каждого из типов жизненных циклов различна значимость (a, b, c)**
- **Что такое типы жизненных циклов?**
  - **Методология разработки проекта**
  - **Адаптация методологии к конкретным условиям (требования, персонал, концепции развития и т.д.)**
  - **Возможные операционные маршруты участников процесса (деятельность руководства проекта и разработчиков, а также ее регламенты)**

Выводы

- **Перспективность** инструментальных моделей развития инструментов поддержки зависит от методологии проекта, ее адаптации к конкретным условиям
- **Дает ли инструментальная модель возможность технологии?**
  — Нет! Это всего лишь средство поддержки
- **Какие преимущества появляются при использовании инструментальной модели?**
  — **Автоматизация деятельности** по управлению развития проектами данного типа.
  Не так уж мало!
- **Проблемы:**
  - Признание необходимости инструментальной поддержки регламентированной разработки проектов
  - Выбор адекватных нотаций (RUP — один из примеров)

Использованные источники

1. Боэм Б.У. Инженерное проектирование программного обеспечения. — М.: Радио и связь, 1985
2. Бркус Ф.П. Мифический человеко-месяц, или как проектируются программные системы. — СПб.: Символ-Плюс, 1999
3. Якобсон А., Буч Г., Рамбо Дж. Унифицированный процесс расзаботки программного обеспечения. — СПб.: Питер, 2002
4. Гантер Р. Методы управления проектированием программного изделия. — М.: Мир, 1981.
5. Скопин И.Н. Основы менеджмента программных проектов. — М.: ИНТУИТ.РУ «Интернет-Университет Информационных Технологий», 2004
6. Сомервилл И. Инженерия программного обеспечения. — М.: Вильямс, 2002
7. Шафер Д.Ф. Фатрелл Р.Т., Шафер Л.И. Управление программными проектами: достижение оптимального качества при минимуме затрат. — М.: Издательский дом «Вильямс», 2003
8. Boehm B. A Spiral Model of Software Development and Enhancement. — IEEE Computer, 21 (5), 1988. — pp. 61-72
9. Microsoft Solutions Framework. — http://www.microsoft.com/rus/msf

6. Особенности первой итерации объектно-ориентированного программного проекта

Мотивация особого подхода к выполнению первой итерации

В пределах одной итерации процесс развития проекта *остается последовательным*:

планирование,
определение требований,
анализ,
конструирование,
тирование,
рование и
ценка

и сохраняет недостатки традиционных технологий

+

велика неопределенность проекта

не хватает критериев предпочтения одних решений перед другими

+

не хватает опыта у разработчиков

Возрастает риск невыполнения проектного задания

Итеративное наращивание → серия коротких мини-циклов

Метод «Сначала в глубину»

разновидность нормального итеративного наращивания, приспособленного к задачам начального периода развития проекта

Противоположный подход — «***Сначала в ширину***»

Главные преимущества подхода «Сначала в глубину»:

- ***Процесс разработки*** продвигается вперед довольно ***быстро***
- ***Разработчики*** за короткое время ***начинают доверять методу***
- ***Критерии можно определить позже***
- ***Разработчики***, особенно новые, ***быстрее вникают в проект***

Рабочие продукты первой итерации как прототип будущей системы

- Первая реальная информация о фактических пользовательских потребностях ⇒ далее процесс конструирования будет более целенаправленным и экономным;
- Возможность уточнить априорное распределение ресурсов всех видов;
- Прототип это
  - первый контакт с пользователями, по которому будут судить о будущих перспективах
  - первая возможность выставить для заказчика счет за проделанную работу.

Первые достижения проекта, ревизия дальнейших планов. Целесообразно или нет дальнейшее развитие проекта. Чему научились:

- наличие конкурентных работ и сопоставление их с затратными и временными характеристиками проекта
- проектные требования понимаются правильнее по сравнению с первоначальным представлением о них
- правдоподобность сделанных априорно оценок (затратных, временных и др.)
- уровень сложности представляемого прототипа;
- фактическое потребление ресурсов в сравнении с их плановыми оценками

Может оказаться, что данный проект значительно труднее реализуем (дороже, требует повышенной квалификации кадров и др.), чем это казалось ранее.

Переход от предварительного анализа к первой итерации

Задачи менеджмента в контексте работ перехода к первой итерации.
Меняется точка зрения на итерацию и на проектируемое изделие:

- вместо проблемно-ориентированных объектов ⇒ реализационные объекты
- модели уровня анализа ⇒ модели для декомпозиции
- появляются специфичные для реализационной среды классы, (к примеру, специфицируются интерфейсные и классы обращения к базам данных)
- конкретизируются и уточняются стратегии и методики, намеченные ранее

Особенности первого для разработчиков перехода к первой итерации:

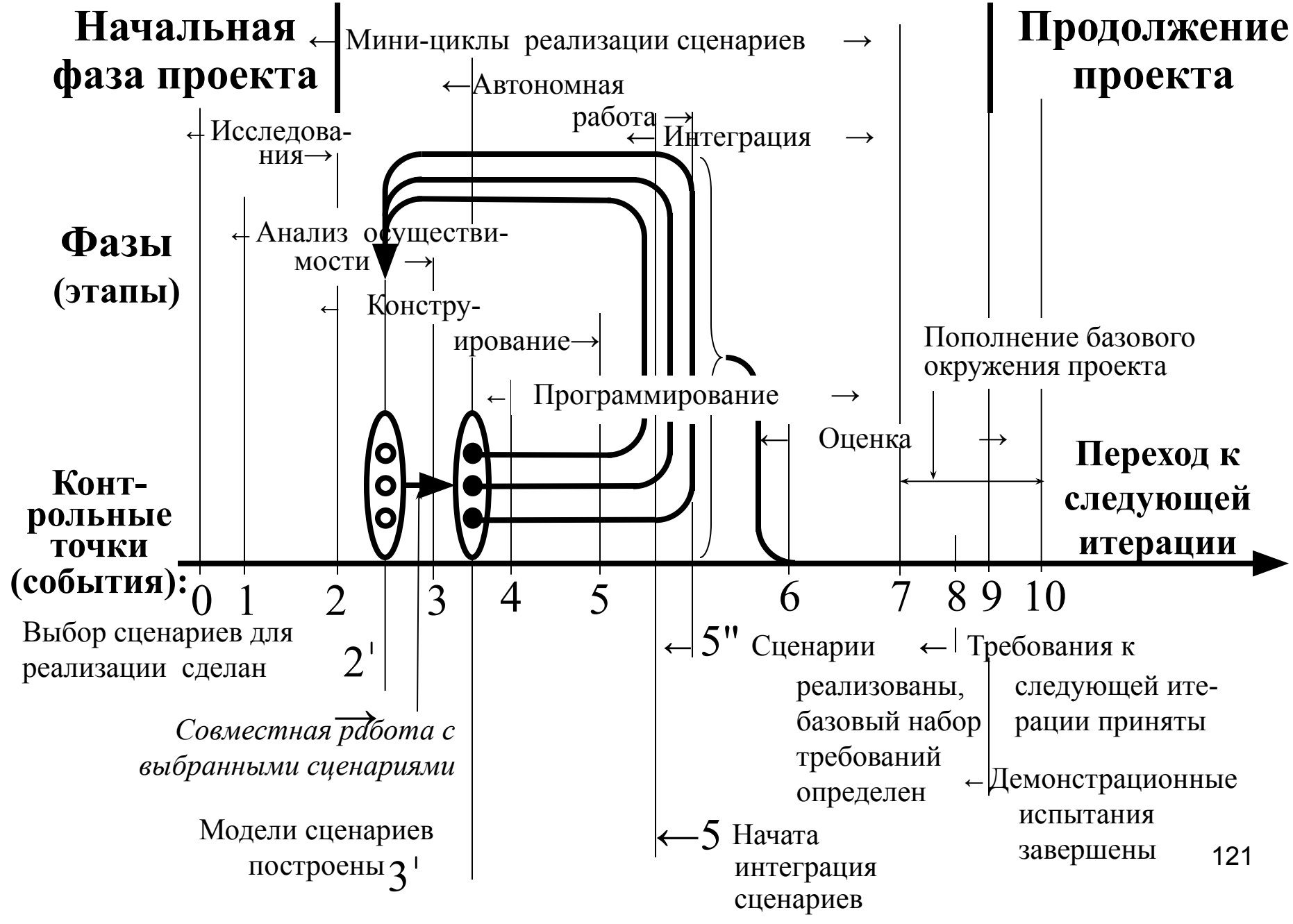
- Заканчивается *предварительный анализ* для всего проекта (общий план проекта и первые документы фазы анализа, *строится общая база всех итераций*);
- Начинается и заканчивается *текущий анализ* для итерации-*прототипа* (планирование)
- Работа над проектом перестает быть личным делом менеджера:
  - функции управления распределяются между исполнителями ключевых ролей,
  - функции контроля и согласования почти вытесняют все другие виды работ менеджера.

Методы предварительного анализа должны предусматривать возможность конструирования, когда нет объективных данных для принятия тех или иных реализационных решений о проекте

Схему ***Задание*** (выдает менеджер) ***— Выполнение*** (деятельность работника)

должна сменить ***организационная методология*** (методика) с более сложными связями в коллективе и более глубокими разделение труда и распределением ответственности

Модель метода «Сначала в глубину»



7. Жизненный цикл в методологиях быстрого развития проектов

Мотивация рассмотрения моделей жизненного цикла в методологиях быстрого развития

- Сторонники быстрого развития утверждают, что они не нуждаются в том, чтобы четко фиксировать этапы развития разработки программного проекта
- Отслеживание процесса не требует специальных документов о достигнутых результатах и проблемах.
- Деятельности менеджера в жестких методологиях противопоставляются самодисциплина и сотрудничество вместо дисциплины и подчинения;
- Особенности планирования, контрольных и других функций

⇒ Все это позволяет *менеджеру в большей мере сосредоточиться на руководстве командой, чем на управлении*.

Тем не менее, понятие жизненного цикла полезно для представления процесса разработки на концептуальном уровне

- *Модели жизненного цикла быстрого развития не претендуют на инструментальность*
- *Понятия контрольных точек и контрольных мероприятий, распределения ресурсов, оценки остаются, хотя их содержание становится менее формализованным, а выполнение — рассредоточенным*

Agile Manifesto

- Индивидуумы и взаимодействия
  важнее
  процессов и инструментов;
- Работоспособное ПО
  важнее
  обширной документации;
- Сотрудничество с заказчиком
  важнее
  заключения контракта;
- Готовность к изменениям
  важнее
  следования плану.

Общая модель жизненного цикла в методологиях быстрого развития

- ***Начальная фаза***. Она выделена, поскольку приходится выполнить работы, которые не являются характерными для основного процесса;
- ***Серия*** максимально коротких ***итераций***, состоящих из шагов:
  - ***выбор*** реализуемых ***требований*** (сценариев; в экстремальном программировании — ***пользовательских историй***),
  - ***реализация*** только отобранных требований,
  - передача результата для практического ***использования***;
  - короткий ***период оценки*** достигнутого (в зависимости от объема работ периода его можно назвать этапом или контрольным мероприятием);
- Фаза ***заключительной оценки*** разработки проекта

Реальные быстрые методологии конкретизируют эту схему, дополняют ее теми или иными методиками

Сегодня есть тенденция к стандартизации agile процессов и появились первые группы с международными сертификатами

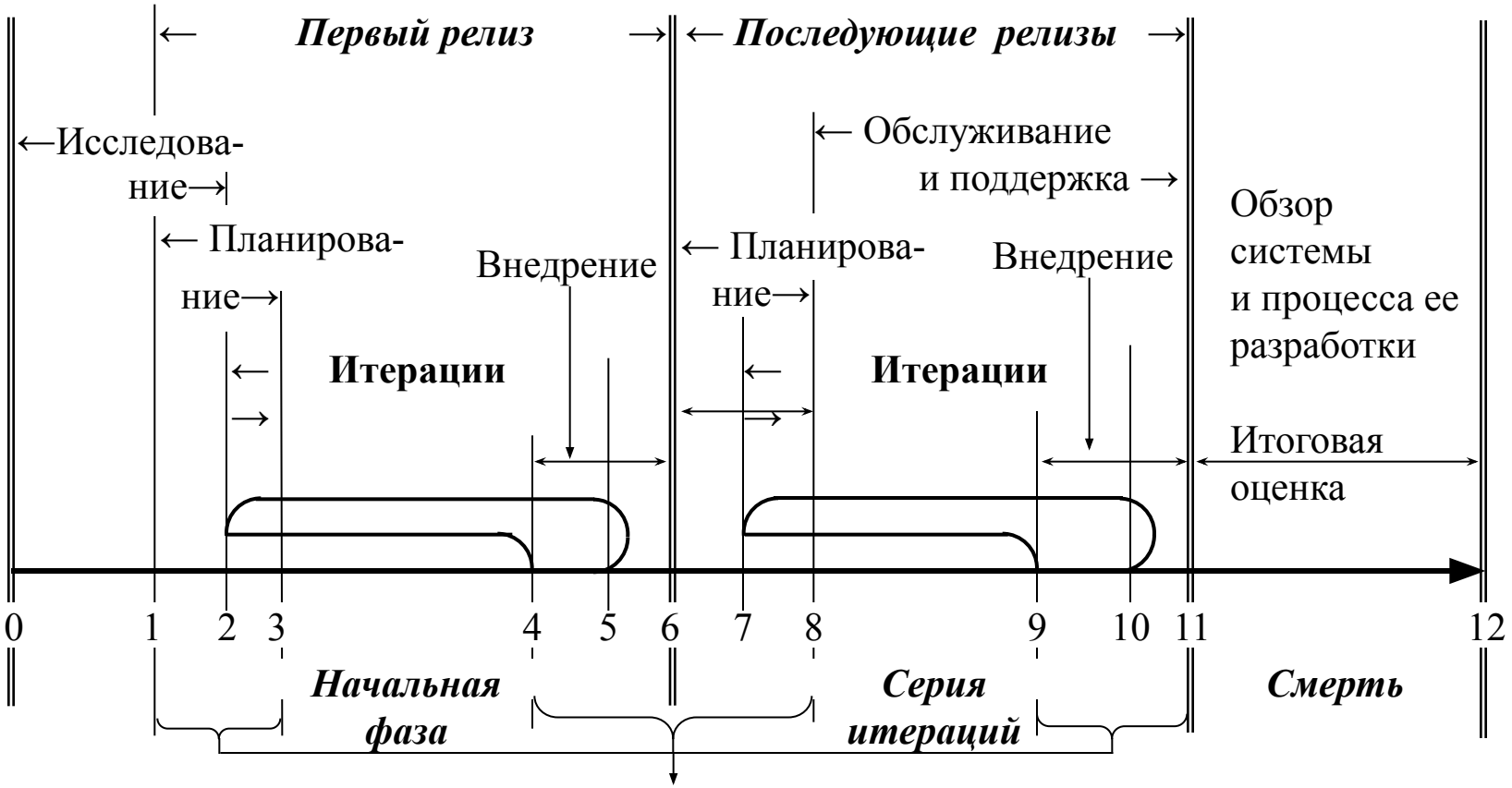
Не станут ли agile методологии жесткими?

Модель жизненного цикла экстремального программирования

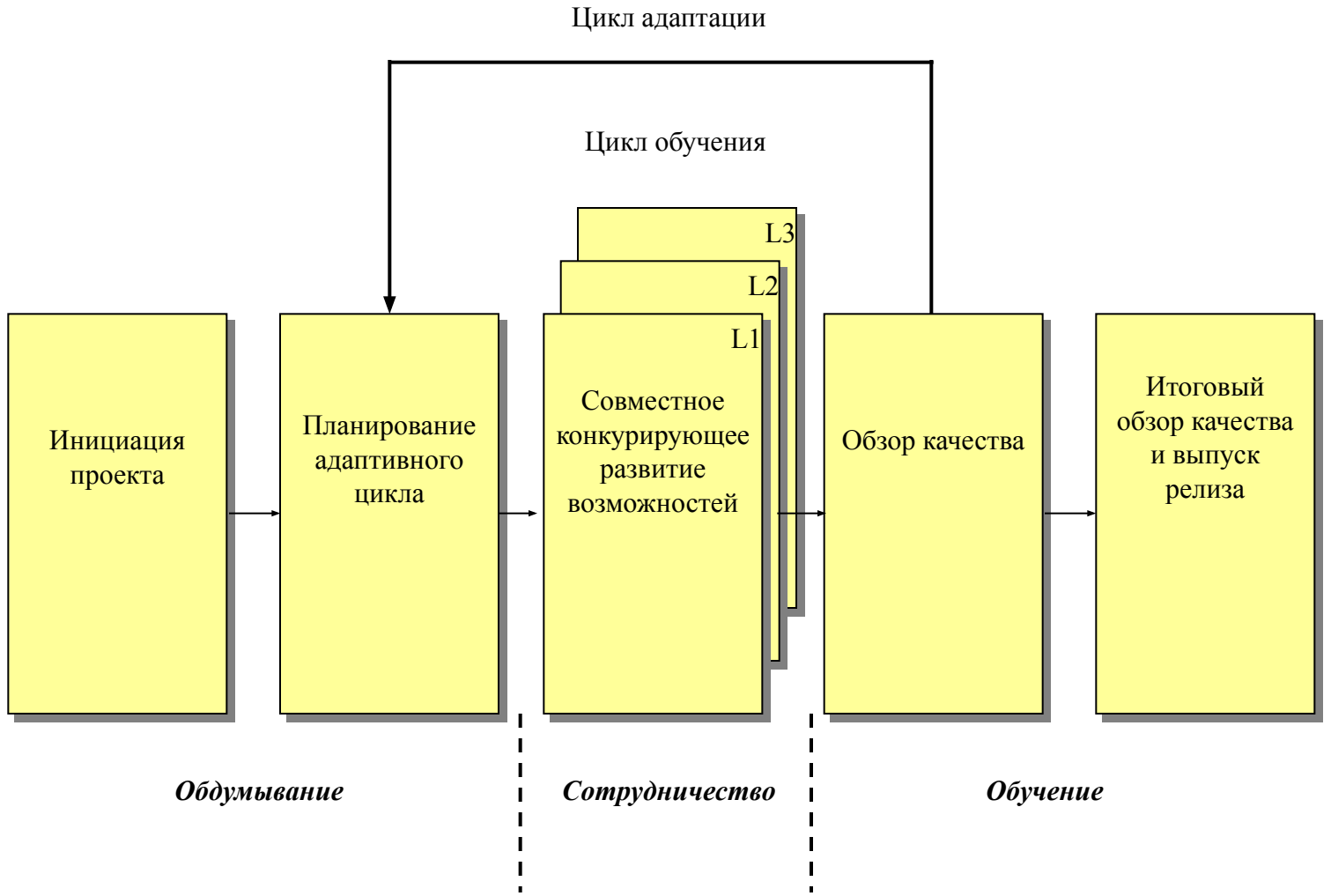
- ***12 методик, относящихся к управлению и руководству.***
  Бек подчеркивает, что все они должны быть внедрены
  1. **Упреждающее тестирование**
  2. **«Путешествие налегке»**
  3. **Общее владение кодом**
  4. **Частые интеграции**
  5. **Парное программирование**
  6. **Сбор пользовательских историй**
  7. **Заказчик как член команды**
  8. **Игра в планирование**
  9. **Менеджер — наставник**
  10. **«Стоячие» совещания**
  11. **12. Некоторые организационные пр**
- ***Утверждается, в частности, что нужен». Почему?***
- ***Как все это согласуется с общими Неявное* (деперсонифициранное, р рассредоточенное по проектным ра *что выполняется в любом проект***
  ⇒ ***слияние контрольных точек, облегченные подготовка к прохождению вех и само прохождение***

- Тесты составляются *до* программ
- Не стоит абсолютизировать
- Функция интеграции кода распределена
- Каждая новая реализуемая возможность,
- Программный модуль – сфера
- Основа разработки: то, что требуется
- Обязательное условие. Он отвечает за (6) и (8), является экспертом в предметной
- Функции менеджмента – помощь в
- Обсуждение стоя способствуют краткости совещаний → можно проводить
- Команда работает в одной комнате, индивидуальные рабочие места – визуально доступные отсеки
- Общая рабочая доска, на которой отмечаются текущие и будущие дела

Модель жизненного цикла экстремального программирования



Адаптивная разработка (ASD — Adaptive Software Development) по Хайсмиту



ASD — это не готовая методология, а базовая концепция для различных адаптивных разработок

Основные принципы адаптивного подхода

- Адаптивная природа всех быстрых методологий — следствие непредсказуемости процесса разработки ПО (Хайсмит использует идеи из области теорией хаоса)
- Основа ASD — три нелинейные перекрывающие друг друга фазы:
  обдумывание → *сотрудничество* → *обучение*
- В окружении, которое требует адаптивности, планирование — парадокс (непредсказуемость)
- Обычно отклонения от плана — *ошибки*, нуждающиеся в исправлении. В адаптивных разработках отклонения ведут к объективно обусловленным решениям ⇒ их следует считать *правильными*
- Неопределенность в непредсказуемой среде преодолевается за счет активного сотрудничества разработчиков
- Внимание менеджмента направлено на обеспечение коммуникации
  ⇒ Разработчики сами находят ответы на возникающие вопросы
- Повышенное внимание к обучению (в предсказуемых методологиях его роль часто занижается: все расписывается заранее, так что потом остается только следовать плану)

«Основное, наиболее действенное и первостепенное, достоинство жизненного цикла ASD заключается в том, что этот процесс заставляет отказаться от интеллектуальных построений, которые являются источником самообмана. Он вынуждает оценивать собственные способности более реалистично»

Семейство методологий Crystal:

- разным проектам нужны разные методологии
- градация проектов: по одной оси — количество людей в проекте, по другой — критичность ошибок
- каждая из методологий семейства предназначена для определенной ячейки получившейся сетки

«Проект, в котором занято 40 человек, и на котором можно позволить себе потерять некоторую сумму, будет работать по другой методологии, нежели проект для шести разработчиков, от которого зависит существование компании» **(Коуберн)**

8. Проблемы оперирования требованиями

Анализ требований

Что такое требования к программному изделию?

Два аспекта:

- Средства программного изделия, в которых *нуждается пользователь* для решения своих проблем или достижения определенных целей (***Что?***);
- Характеристики, которым должна обладать система в целом или ее компонент, чтобы удовлетворять соглашениям, спецификациям, стандартам или другой формально установленной документации (***Как?***).

Характеристики требований

Требования не всегда очевидны и имеют много источников

Требования не всегда легко выразить словами

Существует множество различных типов требований и различных уровней их детализации

Требования чаще всего взаимосвязаны и взаимозависимы, иногда противоречивы

Требования всегда уникальны (требование к фиксируемым требованиям)

Набор требований чаще всего является компромиссом

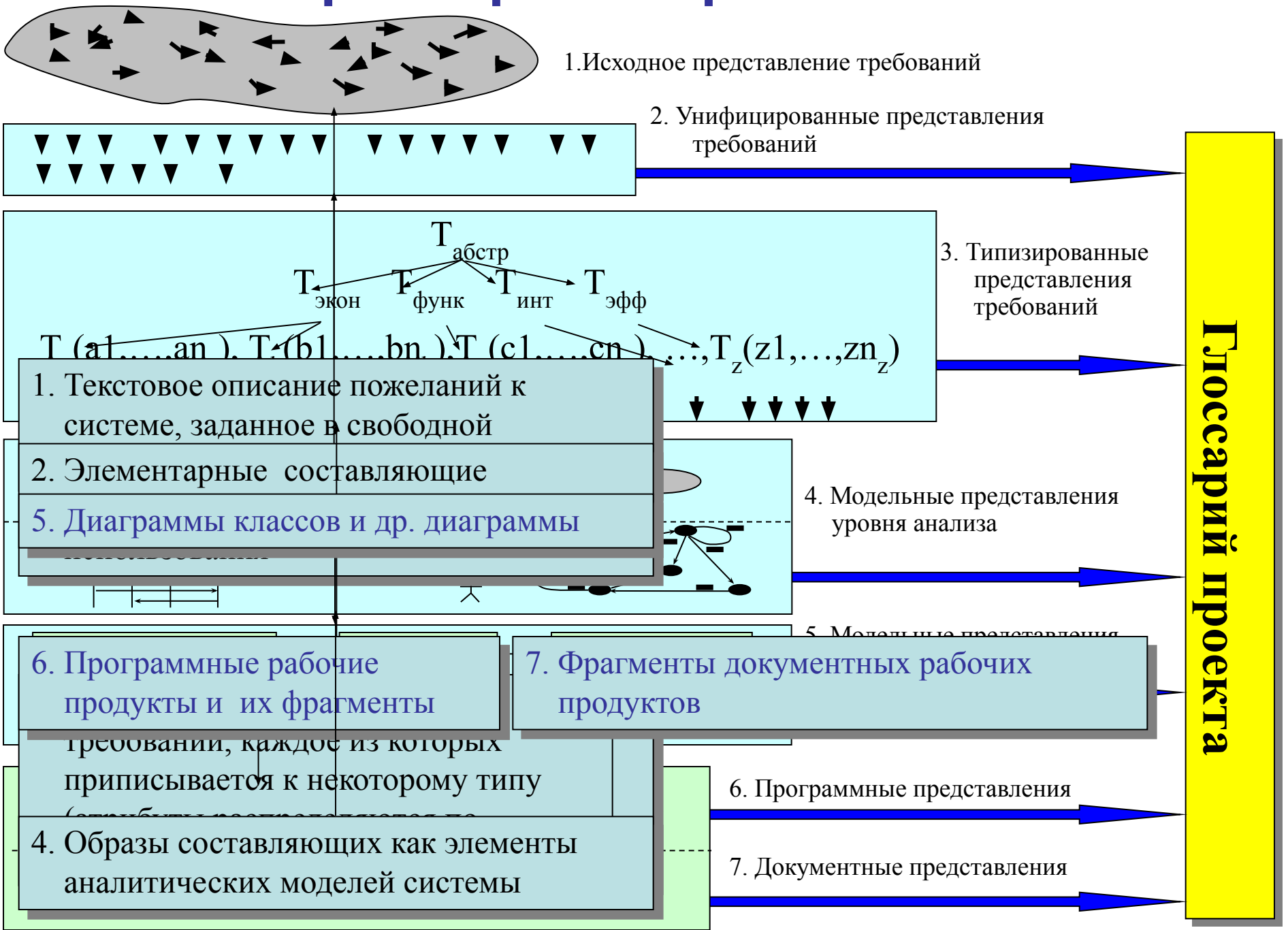
Требования изменяются

Требования зависят от времени

Непрерывность поступления требований

Нужны **специальные приемы** для работы с требованиями

Трассировка требований



Трансформация требований

1. ***Исходное представление*** — текстовое описание пожеланий к системе, заданное в свободной форме.
2. ***Унифицированные представления*** элементарные составляющие требований — для единообразного использования.
3. ***Типизированное представление*** — элементарные составляющие требований, каждое из которых приписывается к некоторому типу (атрибуты на уровнях иерархии типов).
4. ***Модельные представления уровня анализа*** — образы составляющих как элементы аналитических моделей системы.

---

5. ***Модельные представления уровня конструирования*** — диаграммы классов и др.

---

6. ***Программные представления*** — программные рабочие продукты и их фрагменты.
7. ***Документные представления*** — фрагменты документных рабочих продуктов.

Глоссарий проекта

Приемы работы с требованиями

1. Анализ проблем
2. Понимание пользовательских потребностей
3. Преодоление сложности многофункциональности
4. Оперирование с многомерными требованиями
5. Определение системы
6. Управление областью применимости системы
7. Детализированное определение системы
8. Использование метафор
9. Моделирование требований
10. Управление изменениями требований
11. Сохранение истории изменений требований
12. Организация работ с требованиями

1. Анализ проблем

Цель: выявить реальные нужды пользователей, оценка пожеланий с точки зрения их осуществимости в проекте.

Результат: **ранжированный по степени важности список потребностей пользователей с перечислением следствий того, что данная проблема будет решена**

Приемы работы с требованиями

1. Анализ проблем
2. Понимание пользовательских потребностей
3. Преодоление сложности многофункциональности
4. Оперирование с многомерными требованиями
5. Определение системы
6. Управление областью применимости системы
7. Детализированное определение системы
8. Использование метафор
9. Моделирование требований
10. Управление изменениями требований
11. Сохранение истории изменений требований
12. Организация работ с требованиями

2. Понимание пользовательских потребностей

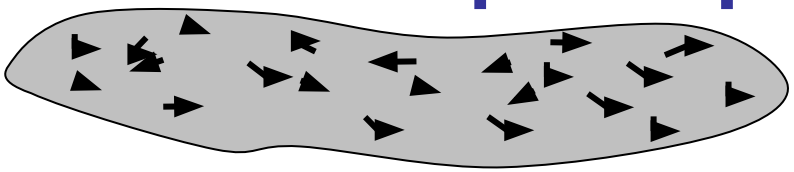
Требования исходят из многих источников, их количество бывает значительно.

Следовательно, необходима **типизация требований**.

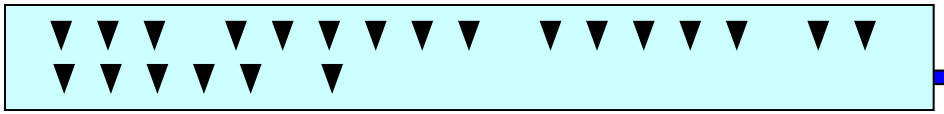
Уровни типов.

Результат: **система типов требований для данного проекта**.

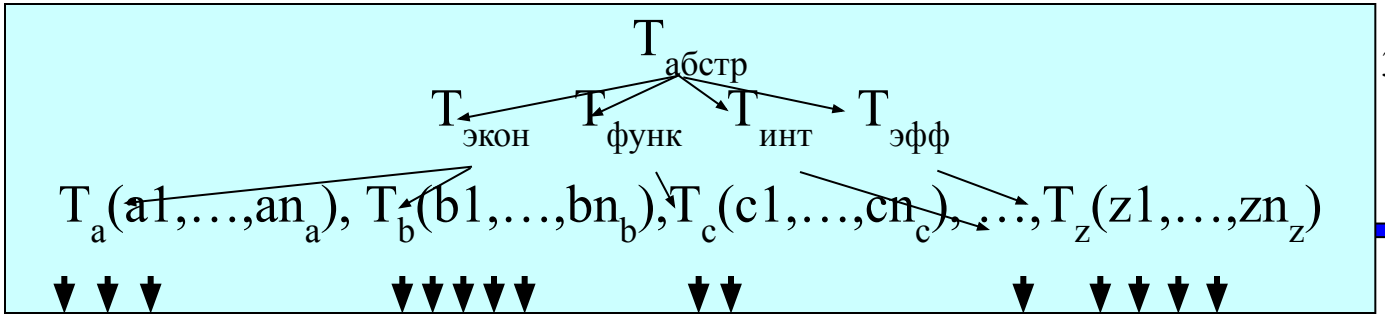
Трассировка требований



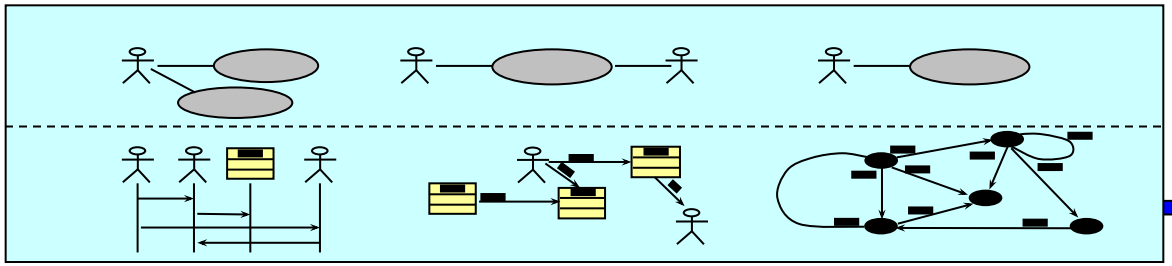
1.Исходное представление требований



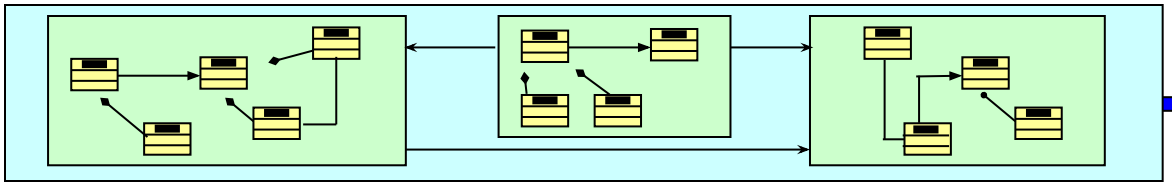
2. Унифицированные представления требований



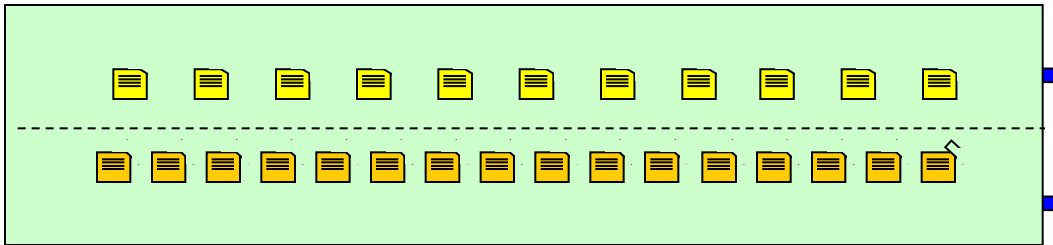
3. Типизированные представления требований



4. Модельные представления уровня анализа



5. Модельные представления уровня конструирования



6. Программные представления

7. Документные представления

Глоссарий проекта

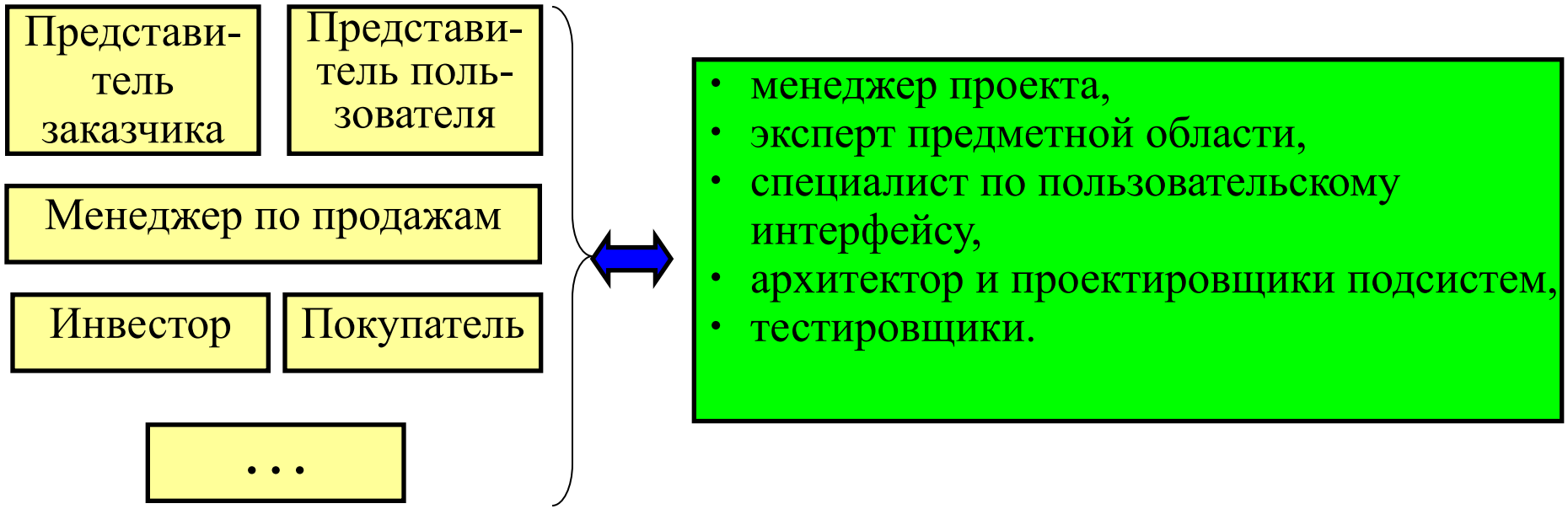
Приемы работы с требованиями

1. Анализ проблем
2. Понимание пользовательских потребностей
3. Преодоление сложности многофункциональности
4. Оперирование с многомерными требованиями
5. Определение системы
6. Управление областью применимости системы
7. Детализированное определение системы
8. Использование метафор
9. Моделирование требований
10. Управление изменениями требований
11. Сохранение истории изменений требований
12. Организация работ с требованиями

3. Преодоление сложности многофункциональности требований

Требования характеризуют изделие с разных сторон — ***многофункциональность***

Инициаторы работ (stakeholders) — те, кто определяют главные требования.



Методы преодоления сложности многофункциональности

Интервью и мозговые штурмы, опросы и анкетирование, изучение прототипов

Выполняется в течение всего проекта!

Результат: **перечень типизированных требований, описанных текстуально и/или графически, порядок которого соответствует приоритетности требований.**

Приемы работы с требованиями

1. Анализ проблем
2. Понимание пользовательских потребностей
3. Преодоление сложности многофункциональности
4. Оперирование с многомерными требованиями
5. Определение системы
6. Управление областью применимости системы
7. Детализированное определение системы
8. Использование метафор
9. Моделирование требований
10. Управление изменениями требований
11. Сохранение истории изменений требований
12. Организация работ с требованиями

4. Оперирование с многомерными требованиями

— одновременное оперирование с разными параметрами отбора требований для анализа

Цель: организация помощи при отборе требований

Параметры отбора: тип требования, приоритетность, …

Тип требования → набор атрибутов, атрибут → набор значений.

Организация хранения и предъявления требований

архитектор, проектировщики подсистем и менеджер проекта, разработчик информационной поддержки и библиотекарь

Оперирование с требованиями — один из основных методов аналитической работы

Результат: **группа требований, выделенная в процессе оперирования для тех или иных целей.**

Приемы работы с требованиями

1. Анализ проблем
2. Понимание пользовательских потребностей
3. Преодоление сложности многофункциональности
4. Оперирование с многомерными требованиями
5. Определение системы
6. Управление областью применимости системы
7. Детализированное определение системы
8. Использование метафор
9. Моделирование требований
10. Управление изменениями требований
11. Сохранение истории изменений требований
12. Организация работ с требованиями

5. Определение системы

Трансформация потребностей и перечня требований в описание для разработки.

Общие соглашения: как понимаются требования и их приоритетность, оценка затрат и ресурсных потребностей, рисковые ситуации и стратегия управления рисками. Границы применения будущей системы. Виды рабочих продуктов, правила их построения, проверки и т.д. Внешние рабочие продукты и способы их использования в проекте: применение результатов, использование как прототипов и др.

Форма представления определения системы: тексты, схемы, гипертекстовая структура и др.

Перед составлением формализованных модельных описаний следует сначала представить их на естественном языке!

Результат: **Точное и согласованное определение системы**.

Приемы работы с требованиями

1. Анализ проблем
2. Понимание пользовательских потребностей
3. Преодоление сложности многофункциональности
4. Оперирование с многомерными требованиями
5. Определение системы
6. Управление областью применимости системы
7. Детализированное определение системы
8. Использование метафор
9. Моделирование требований
10. Управление изменениями требований
11. Сохранение истории изменений требований
12. Организация работ с требованиями

6. Управление областью применимости системы

Цель: Выбор наиболее приоритетного для инициаторов работ направления развития проекта в условиях имеющихся в текущий момент ресурсов.

Методы: обсуждение атрибутов требований (**приоритетность задач**, потребность ресурсов, допустимый уровень риска) для *выявления реально осуществимых возможностей*.

Результат: **список требований, которые планируется удовлетворить (реализовать) на ближайших этапах, текущей и последующих итераций и для проекта в целом**

Приемы работы с требованиями

1. Анализ проблем
2. Понимание пользовательских потребностей
3. Преодоление сложности многофункциональности
4. Оперирование с многомерными требованиями
5. Определение системы
6. Управление областью применимости системы
7. Детализированное определение системы
8. Использование метафор
9. Моделирование требований
10. Управление изменениями требований
11. Сохранение истории изменений требований
12. Организация работ с требованиями

7. Детализированное определение системы

Цель: Нужно, прежде всего, чтобы инициаторы работ смогли *понять*, какое изделие им предлагается, и *решить*, соглашаться с этим предложением или нет.

Делается по отношению к функциональности, а также для выработки соглашений о порядке реализации требований, о практичности и надежности системы, о ее производительности, о поддержке и удобствах применения, о порядке тестирования.

Варианты детализированного определения для разных пользователей.

Результат: **определение системы в виде описаний ее возможностей, пригодных для предоставления пользователям очередного резиза**.

Приемы работы с требованиями

1. Анализ проблем
2. Понимание пользовательских потребностей
3. Преодоление сложности многофункциональности
4. Оперирование с многомерными требованиями
5. Определение системы
6. Управление областью применимости системы
7. Детализированное определение системы
8. Использование метафор
9. Моделирование требований
10. Управление изменениями требований
11. Сохранение истории изменений требований
12. Организация работ с требованиями

8. Использование метафор

Цель: метафора как база пользовательского представления системы

- Для каждого элемента автоматизируемой деятельности в рамках метафоры должны быть найдены образы среди средств системы (*функциональная* и *реализационная полнота*) и заданы адекватные формы (*интерфейсная полнота*).
- Принципы, способствующие, качеству системы метафор:
  - *Точность* — отражение в метафоре целей, ресурсов, средств и методов как элементов пользовательской деятельности
  - *Полнота* — отражение в всех элементов деятельности-прототипа
  - *Единый уровень абстракции в представлении метафоры*
  - *Структурность метафоры*
  - *Использование существующих метафор*
  - *Привычность и естественность метафоры.* Она всегда должна быть *узнаваема*
  - *Учет психологических и эргономических особенностей* (комплекс рекомендаций)

Результат: **дополнительные требования, связанные с реализацией метафор, которые предъявляются к архитектуре, интерфейсу и документации программной системы**.

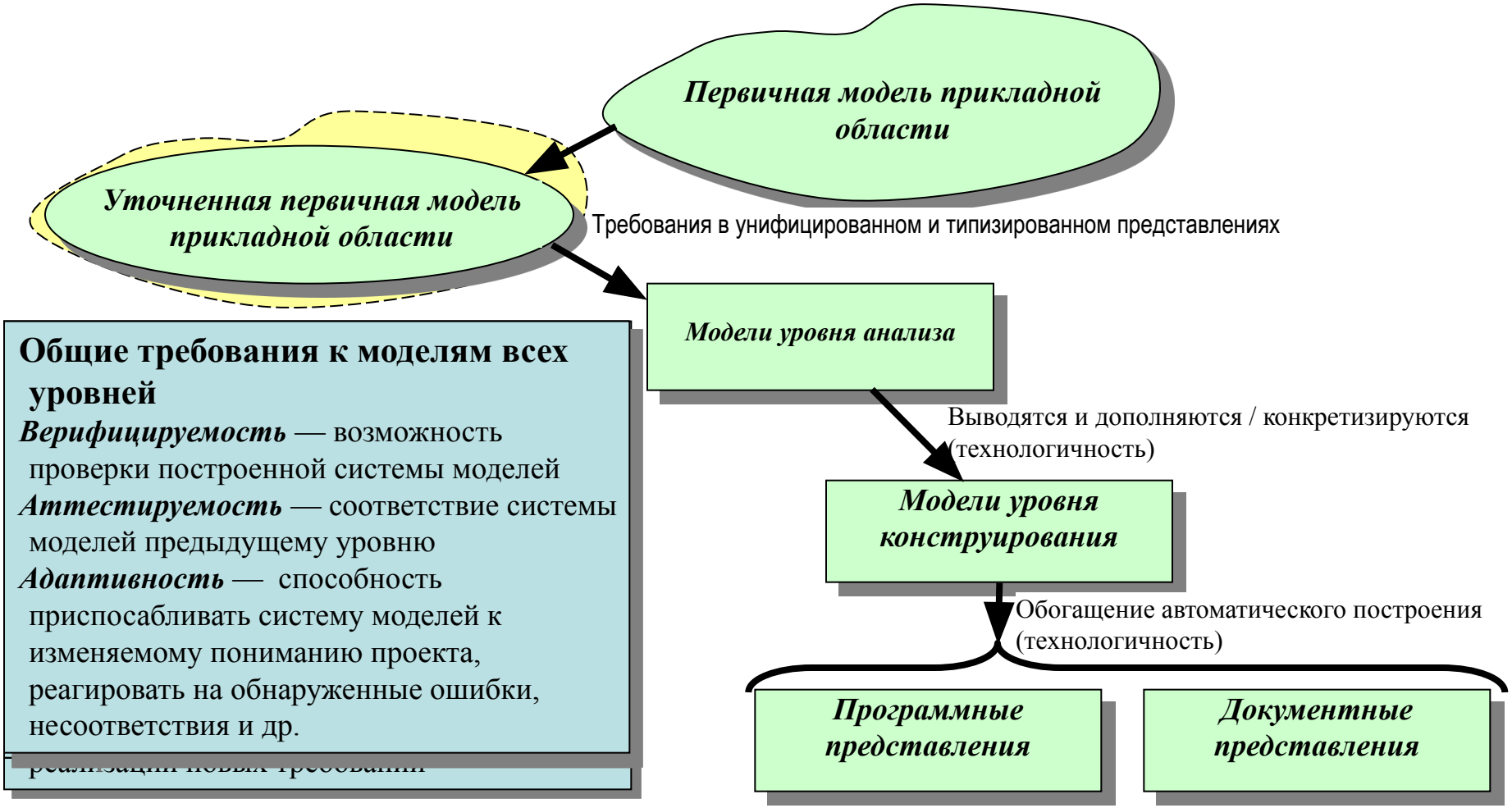
Приемы работы с требованиями

1. Анализ проблем
2. Понимание пользовательских потребностей
3. Преодоление сложности многофункциональности
4. Оперирование с многомерными требованиями
5. Определение системы
6. Управление областью применимости системы
7. Детализированное определение системы
8. Использование метафор
9. Моделирование требований
10. Управление изменениями требований
11. Сохранение истории изменений требований
12. Организация работ с требованиями

9. Моделирование требований

Моделирование требований — сокращение для более точного названия процесса построения моделей прикладной области по динамически изменяемой совокупности требований к программной системе

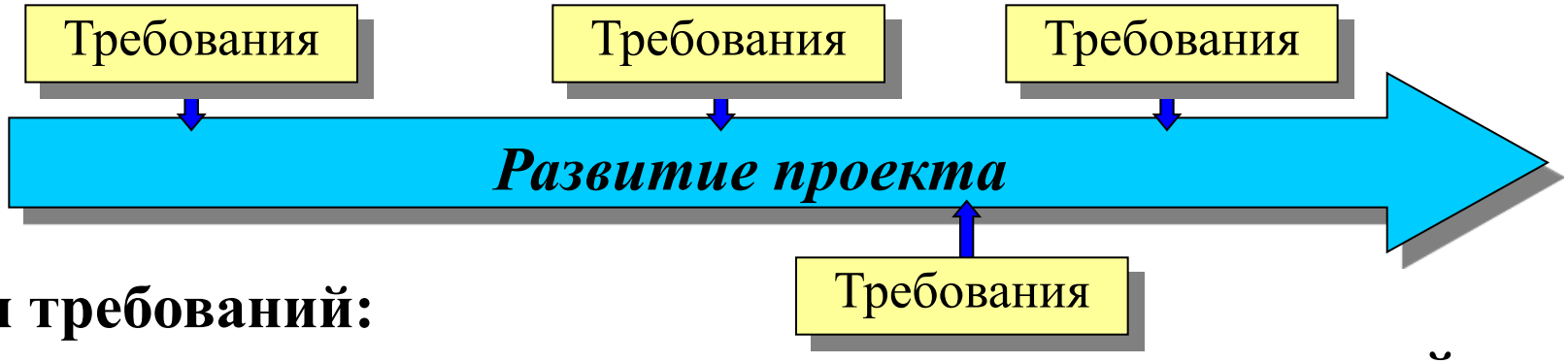
Адекватность, Полнота, Непротиворечивость, Расширяемость — свойства моделей



Приемы работы с требованиями

1. Анализ проблем
2. Понимание пользовательских потребностей
3. Преодоление сложности многофункциональности
4. Оперирование с многомерными требованиями
5. Определение системы
6. Управление областью применимости системы
7. Детализированное определение системы
8. Использование метафор
9. Моделирование требований
10. Управление изменениями требований
11. Сохранение истории изменений требований
12. Организация работ с требованиями

10. Управление изменениями требований



Виды требований:

- ***дополнительное*** **— отражает ранее не рассмотренный аспект системы;**
- ***модифицирующее*** **— изменяет одно или несколько требований;**
- ***отменяющее*** **— принятие его исключает одно или несколько требований.**

Различия анализа нового требования в контексте существующих соглашений. *Цель* такого анализа — поддержка целостности системы требований проекта.

Принять или отвергнуть требование для данного проекта?

Приемы работы с требованиями

1. Анализ проблем
2. Понимание пользовательских потребностей
3. Преодоление сложности многофункциональности
4. Оперирование с многомерными требованиями
5. Определение системы
6. Управление областью применимости системы
7. Детализированное определение системы
8. Использование метафор
9. Моделирование требований
10. Управление изменениями требований
11. Сохранение истории изменений требований
12. Организация работ с требованиями

11. Сохранение истории изменений требований

Набор требований и их атрибуты меняются весьма произвольно

Изменения зависят от изменений в окружении, от используемых методов разработки, от политики компании и заказчика, от успеха или неудачи очередного релиза, от множества других факторов

История изменений требований используется

- для отката проекта к пройденному состоянию
- для отслеживания аналогичных ситуаций
- для поддержки управления версиями
- для будущих учебных целей

Для хранения истории нужно обеспечить

- протоколирование появления данных
- обратимость изменения данных
- фиксацию времени каждой модификации данных

Результат: **совокупность исторических сведений, которая хранится для данного проекта и используется при организации дальнейших работ.**

Пример протокола и система типовых запросов к истории

```
.....
0627 (12.12.1999):ТР0 поступило,    Иванов
.....
0632 (13.12.1999):        ТР1 получено из ТР0,    Петров
0633            ТР2 получено из ТР0,    Петров
0634:           ТР3 получено из ТР0,    Петров
.....
0651 (14.12.1999):        ТР1 принято,            Петров
0652:           ТР2 отклонено,          Петров
0653:           ТР3 принято,            Петров
0634:           ТР1У получено из ТР1,   Петров
0634:           ТР3У получено из ТР3,   Петров
.....
0681 (15.12.1999):        М01 расширено ТР1У, ТР3У,   Петров
0682:           ТР3У отклонено,         Петров
0683:           ТР3 отклонено,          автоматически 0682
0684:           М01 восстановлено по 0681,   автоматически 0682
0685:           М01 расширено ТР1У,     Петров
.....
```

Задача конкретизации сохранения истории изменений требований для данного проекта:

- прагматический уровень +
- оптимизация общей постановки

Примеры семантически сложных запросов (временной параметр запроса!):

- выдать набор всех требований, поступивших в течение определенного периода от заданного инициатора работ и касающихся интерфейса;
- показать требования, приятые к некоторому моменту и связанные с изменением заданного требования;
- показать ретроспективу изменений заданного представления заданного требования.

Приемы работы с требованиями

1. Анализ проблем
2. Понимание пользовательских потребностей
3. Преодоление сложности многофункциональности
4. Оперирование с многомерными требованиями
5. Определение системы
6. Управление областью применимости системы
7. Детализированное определение системы
8. Использование метафор
9. Моделирование требований
10. Управление изменениями требований
11. Сохранение истории изменений требований
12. Организация работ с требованиями

12. Организация работ с требованиями

- Требования разделяются на принимаемые и отвергаемые
- Для отвергаемого требования — мотивированное заключение
- Для принимаемого элементарного составляющего требования определяется, на какой итерации оно может (должно) быть удовлетворено (критерии — приоритетность, зависимость …)
- Простые требования реализуются в момент утверждения
- Сложные требования откладываются до завершения конструкторских работ данной итерации
- Из-за откладывания требований — корректировка общего плана
- Учитывается, что ранее принятое требование может оказаться отвергнутым вследствие принятия нового требования
- Изменения планов всегда согласуются с заказчиком.

Результат управления изменениями требований *(перманентный):* **определение целесообразного порядка адаптации проекта к меняющимся условиям эксплуатации разрабатываемого программного изделия**

Управление требованиями

Приемы 1 – 12 — необходимое, но не достаточное условие

Нужна специальная работа, которая

- адаптирует приемы к конкретным условиям ведения разработки,
- организует мероприятия, соответствующие приемам,
- определяет, как и когда активизируются эти мероприятия,
- встраивает мероприятия в планы развития проекта.

В результате определяются процессы *управления требованиями проекта*

10. Методическая поддержка оперирования требованиями. Примеры: работа с резюме, прием на работу

Исходные требования

Первичные требования

1. Нужно создать систему, которая позволила бы собирать резюме из разных источников.
   Автор резюме — кандидат на прием на работу, сотрудник или тот, о ком есть какие-то сведения.
2. Нужно иметь черный список, т.е. тех, кого рассматривать надо в случае крайней необходимости.
3. Нужно уметь просматривать резюме, отбирать тех, кто удовлетворяе
4. ……

отвергается из-за того, что не хранится атрибут. Однако, если надо, но придется добавить и тогда рассчитать работы по модификации.

Дополнительные требования

5. *Я хочу уметь отбирать по национальному признаку*
6. *Я хочу отбирать тех, кто не только имеет нужный навык, но и работал с чем-то определенное время*.

сказать заранее, т.е. до разбиения на элементарные составляющие, унификации и типизации, ничего нельзя.

Унифицированные требования

Выделение существенного для предметной области

Участники-пользователи системы:

1. *Автор резюме*: составляет, редактирует, изменяет /новые сведения/
2. *Обработчик резюме*: регистрирует, наводит статистику
3. *Работодатель по резюме*: отбирает по навыкам.

Объекты и действия:

1. Резюме. Имеет атрибуты (по ним можно отбирать):
   1. Идентификационные;
   2. Навыки;
   3. Стаж;
   4. Другие атрибуты.

Работы:

1. Составить резюме — внешняя
2-5. См. права
6. Составить черный список

2. Персоны:
   1. Автор;
   2. Регистратор;
   3. Работодатель.
3. Права:
   1. Составлять резюме;
   2. Читать резюме;
   3. Регистрировать автора;
   4. Регистрировать резюме;
   5. Регистрировать работодателя
4. Действия: …

До этого проводить типизацию не получится!

Требования к объектам, с которыми работают субъекты

Вариант «обстоятельство, сказуемое, подлежащее»

Перечень унифицированных требований

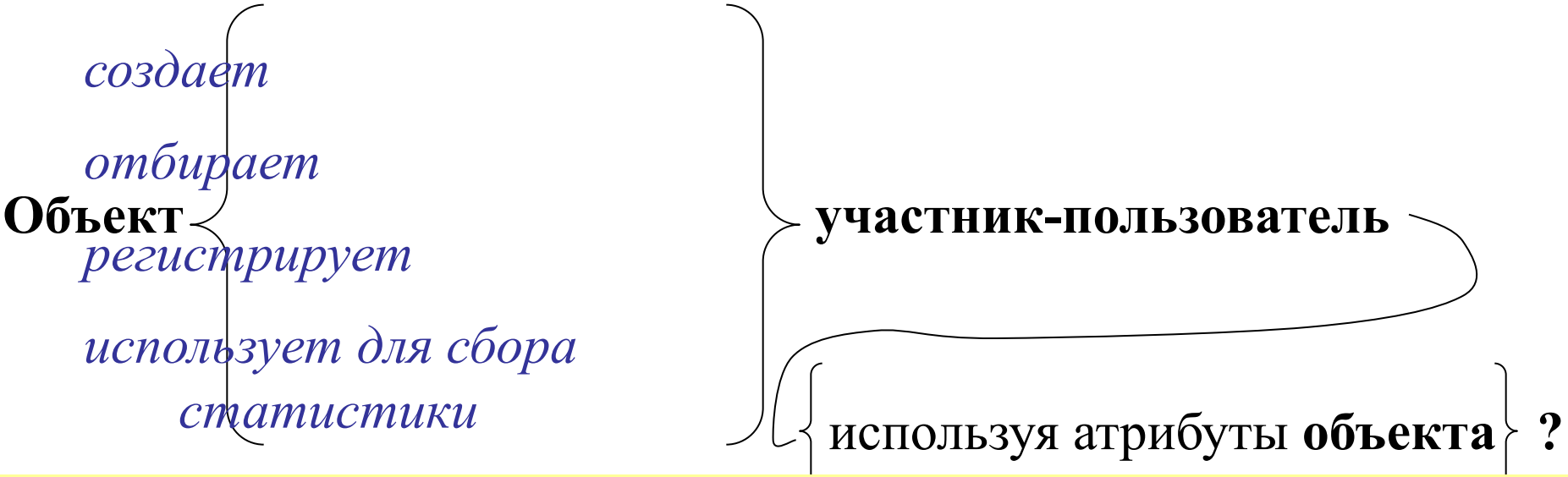
1. Резюме *создает*
   Автор резюме, Обработчик резюме, Работодатель по резюме {последние двое — для тех, «о ком есть какие-то сведения» → Автор резюме из (1) — это из-за отсутствия строгости}.
2. Черный список *создает*
   Обработчик резюме, Работодатель по резюме.
3. Авторов резюме *отбирает*
   Работодатель по резюме.
4. Авторов резюме, Обработчиков резюме, Работодателей по резюме *регистрирует*
   выделенный Обработчик резюме {администратор}
5. *Придумать про статистику!*

Вариант «подлежащее, сказуемое, обстоятельство»

Возможны и другие варианты

Все аналогично, но **нельзя смешивать варианты!**

Приведение к типизированной форме



Здесь система типов строится по отношению к объектам. Можно ее строить и по отношению к действиям, и по отношению к субъектам. Можно выстраивать иерархии. Т.е.

все делать так, как удобно в конкретной ситуации!

Получился граф из трех (и более, если иметь ввиду атрибуты) видов вершин со специальными связями между вершинами разных видов (не все они осмысленны). Можно изучать его формально с целью выделения нужных обобщений (см. ниже).

Что надо делать, чтобы получить объективный результат

- Выделить всех субъектов-пользователей
- Постараться определить все объекты
- Выстроить функции-взаимодействия, определить их параметры и результаты

Все это в данном подходе делается путем изучения исходных текстов требований (отправная точка) → ***лингвистически стандартизованная форма*** элементарных требований.

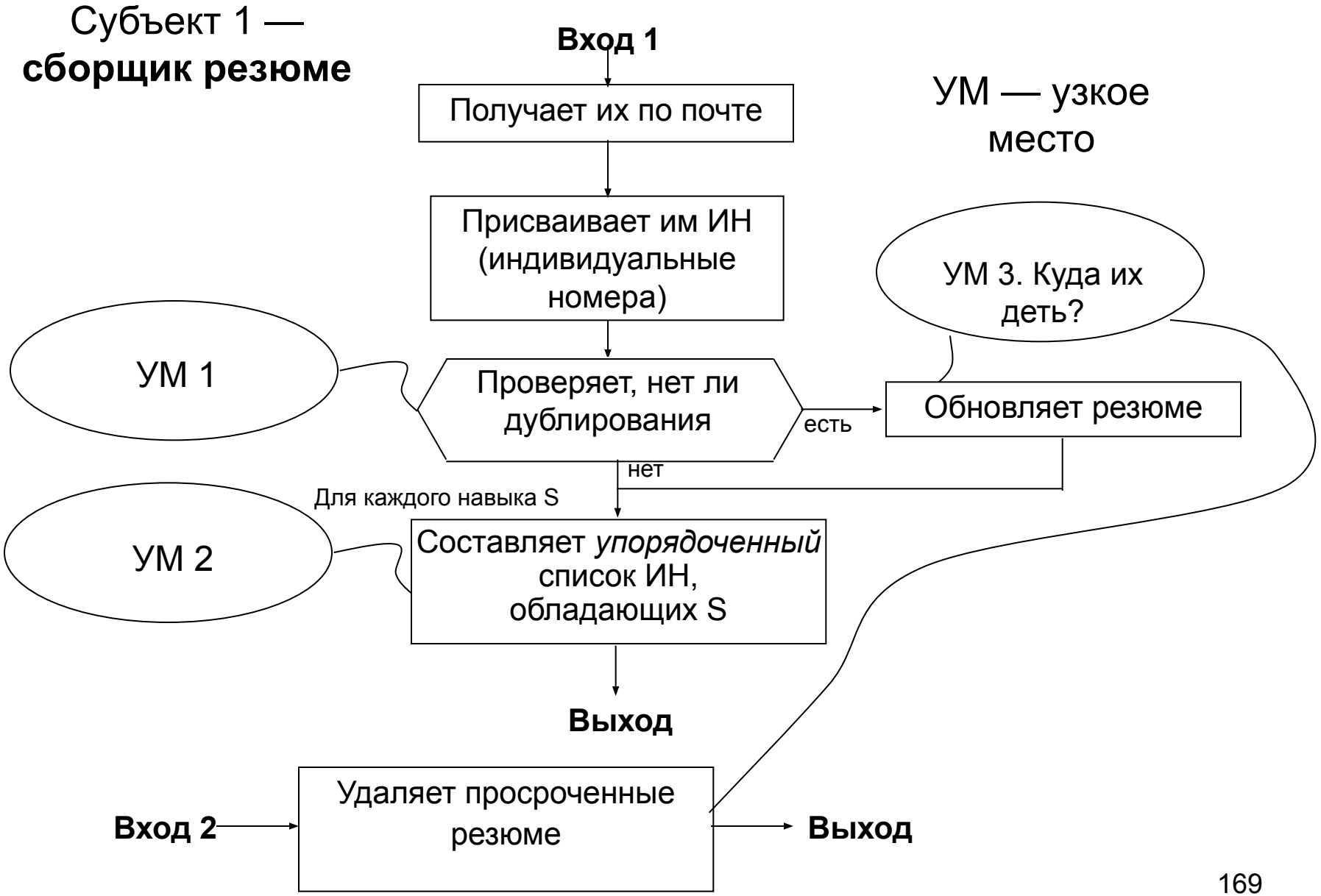
- Обобщить элементарные требования на основе абстракции:
  - *субъектов*,
  - *объектов*,
  - *операций*

  } *кандидаты на основание типизации требований*
- Помощь в выборе обобщения: интервью, консультации, анкетирование и др. задачи эксперта предметной области.

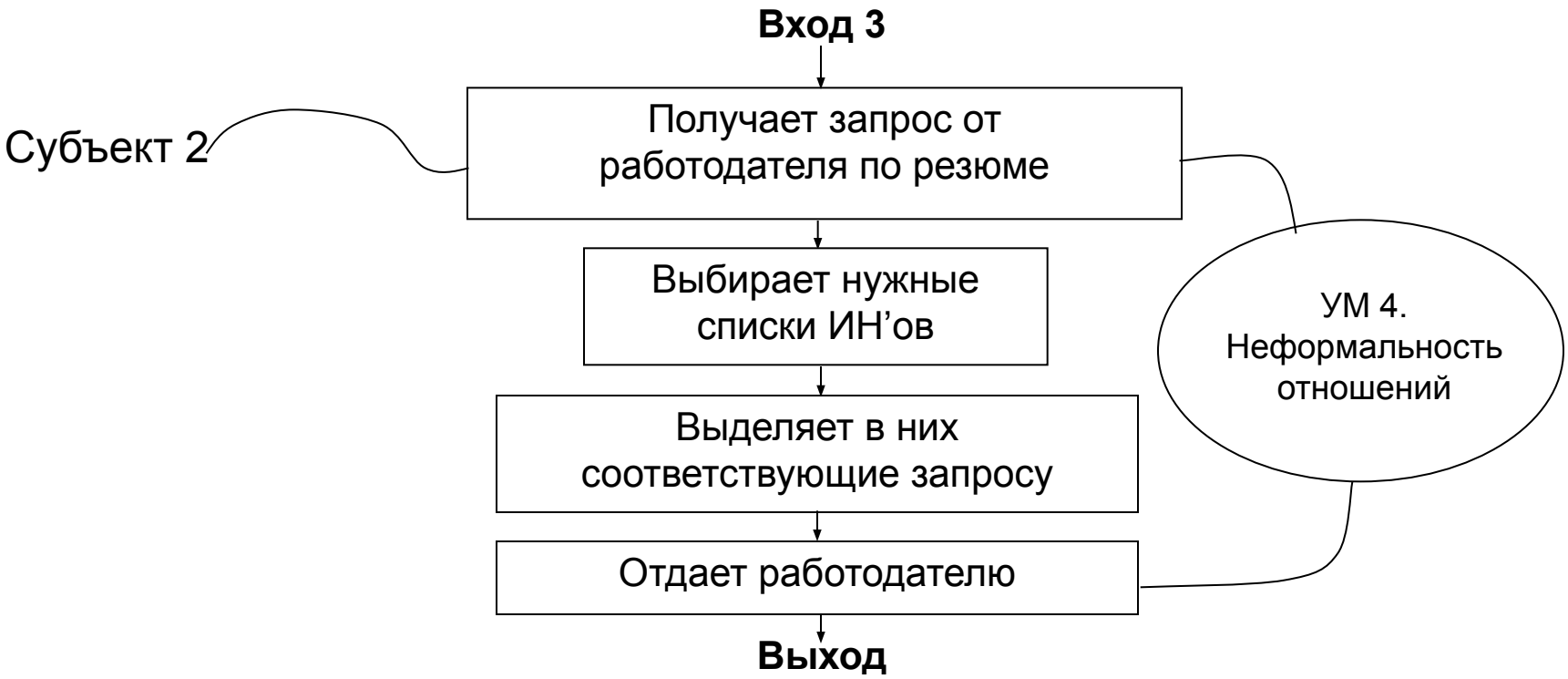
  Это обратная связь с инициаторами работ
- Результаты всего фиксируются в глоссарии
- Задача организации (неформального) обратного перевода.

Это был **лингвистический подход к анализу**

Реконструкция деятельности



Реконструкция деятельности



Когда все маршруты составлены/изучены, возможна систематизация:

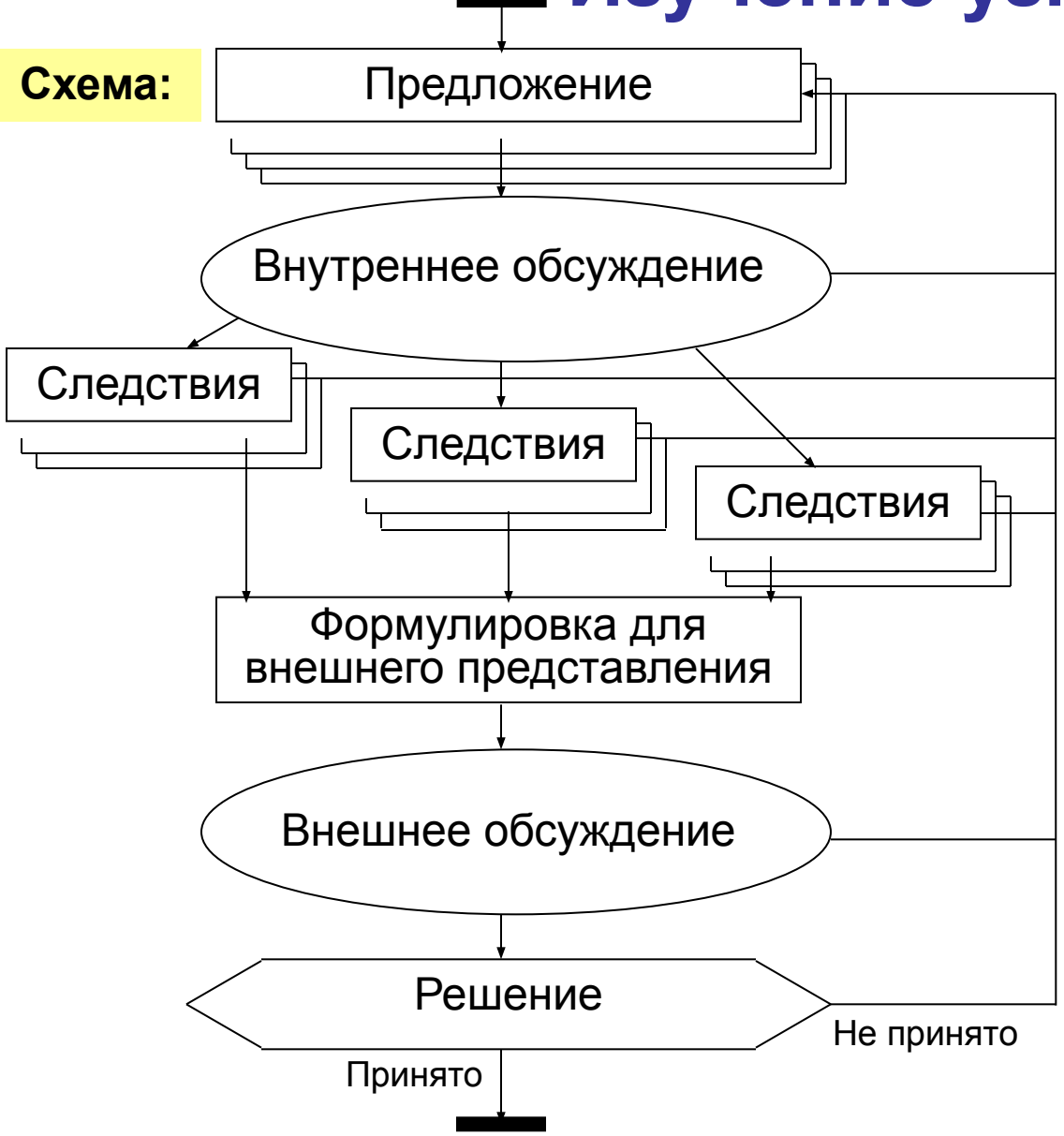
- Объекты (здесь сложнее)
- Функции (здесь понятнее)

Далее все как в лингвистическом подходе

Это подход **анализа деятельности пользователей**

Изучение узких мест

Схема:



Приемы:

- Протоколирование (всего);
- Мозговые штурмы (сбор и анализ предложений).
- *Представление предложения*;
- Обзор следствий;
- Экспертиза;

- Критика + альтернатива;
- Назначение ответственных;
- Выработка решения;
- Техническое задание.

Это схема обсуждения любых предложений

Путь от проблем пользователя

- УМi — пытаются синтезировать обобщенные УМi (как УМ3)
- *Вопросы*: что вызывает затруднения. Попытки выразить проблему в терминах объект, субъект, функция
- Если проблема обозначена, то пытаться строить ее декомпозицию (какие подпроблемы нужны для решения)
- Не забывать про горизонтальный анализ
- Не забывать про синтез

Все это с целью подготовки среды, в которой происходят действия с системой (т.е. переход к восходящему построению)

Это путь **системного анализа**

Сценарии

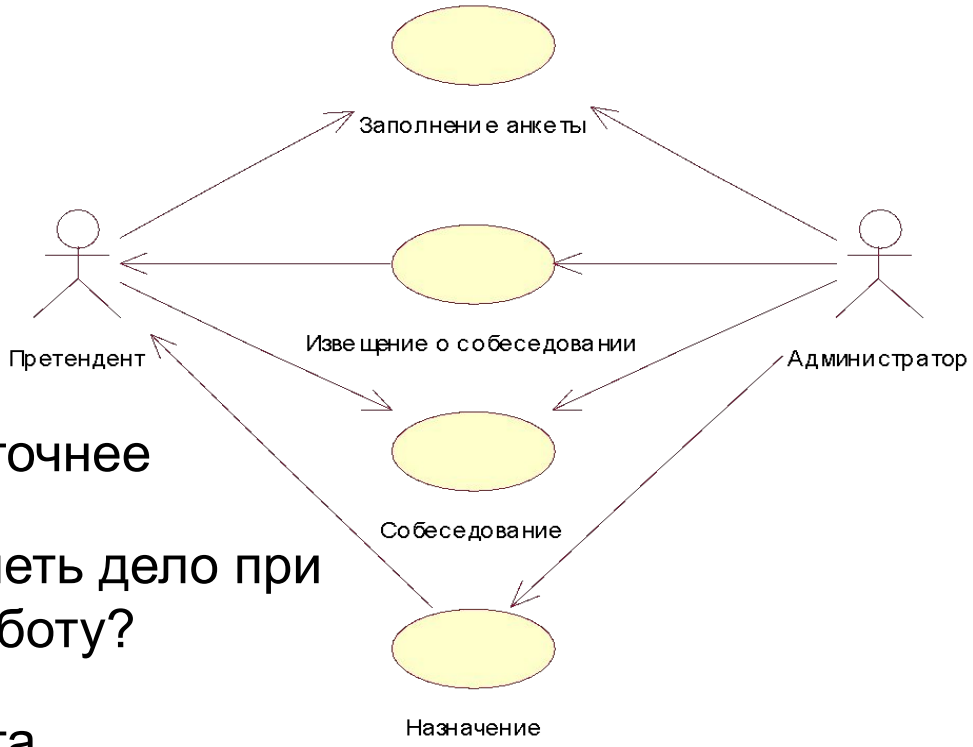
Атомарных действий не хватает. Нужны последовательности действий, т.е. *сценарии*.

Следует проанализировать:

- условия, выполнение которых необходимо, чтобы те или иные действия объектов и субъектов были бы (а) осмысленны и (б) корректны — pred-условия;
- как то или иное действие изменяет среду функционирования объектов и субъектов — post-условия;
- как то или иное действие изменяет среду функционирования объектов и субъектов — post-условия;
- какие существуют связи между объектами и субъектами, упорядочивающие действия;
- линии жизни объектов и субъектов и их переплетение между собой (принятая сегодня для этого парадигма — передача сообщений между объектами);
- варианты статусов сообщений (зависят от проекта) с целью выявления дополнительных объектов;

Диаграммы ситуаций использования

Здесь придется многое домысливать



Здесь все выражено точнее

С какими объектами придется иметь дело при составлении сценария приема на работу?

- Претендент
- Администратор

} два субъекта

- Анкетные данные
- атрибуты Назначения
- Извещения о собеседовании — локально
- Объект, представляющий Собеседование — вне модели

Диаграммы взаимодействий

Модели ситуаций использования описывают, что делает система при взаимодействии с ней пользователей. Активизация действий, указываемых в них, достигается за счет того, что участники процесса — субъекты моделей, а также иные объекты системы — способны:

- производить (порождать) события;
- воспринимать события (сообщения о них);
- реагировать на события, т.е. выполнять другие операции, приводящие к определенным результатам.

Участники взаимодействуют между собой.

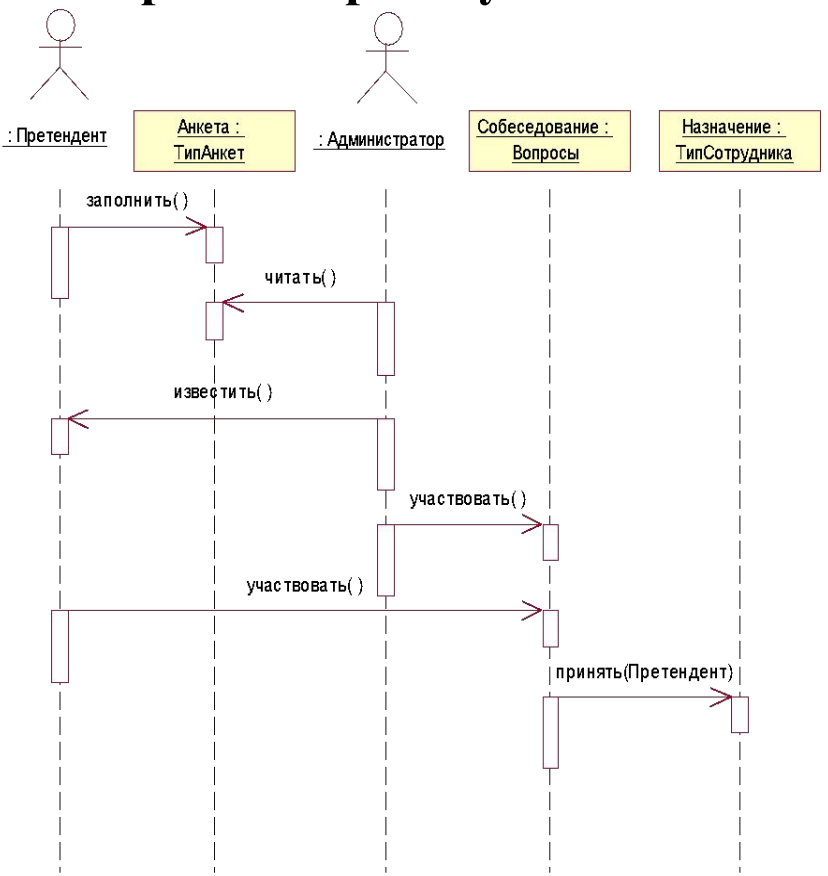
Правила, в соответствии с которыми *взаимодействия выстраиваются в последовательности* для целенаправленного достижения *результата*, называются ***сценариями поведения системы***.

Диаграммы последовательностей, кооперативные диаграммы, диаграммы деятельностей, диаграммы состояний

Это про *функциональность*, а не про *интерфейс* (особая задача).

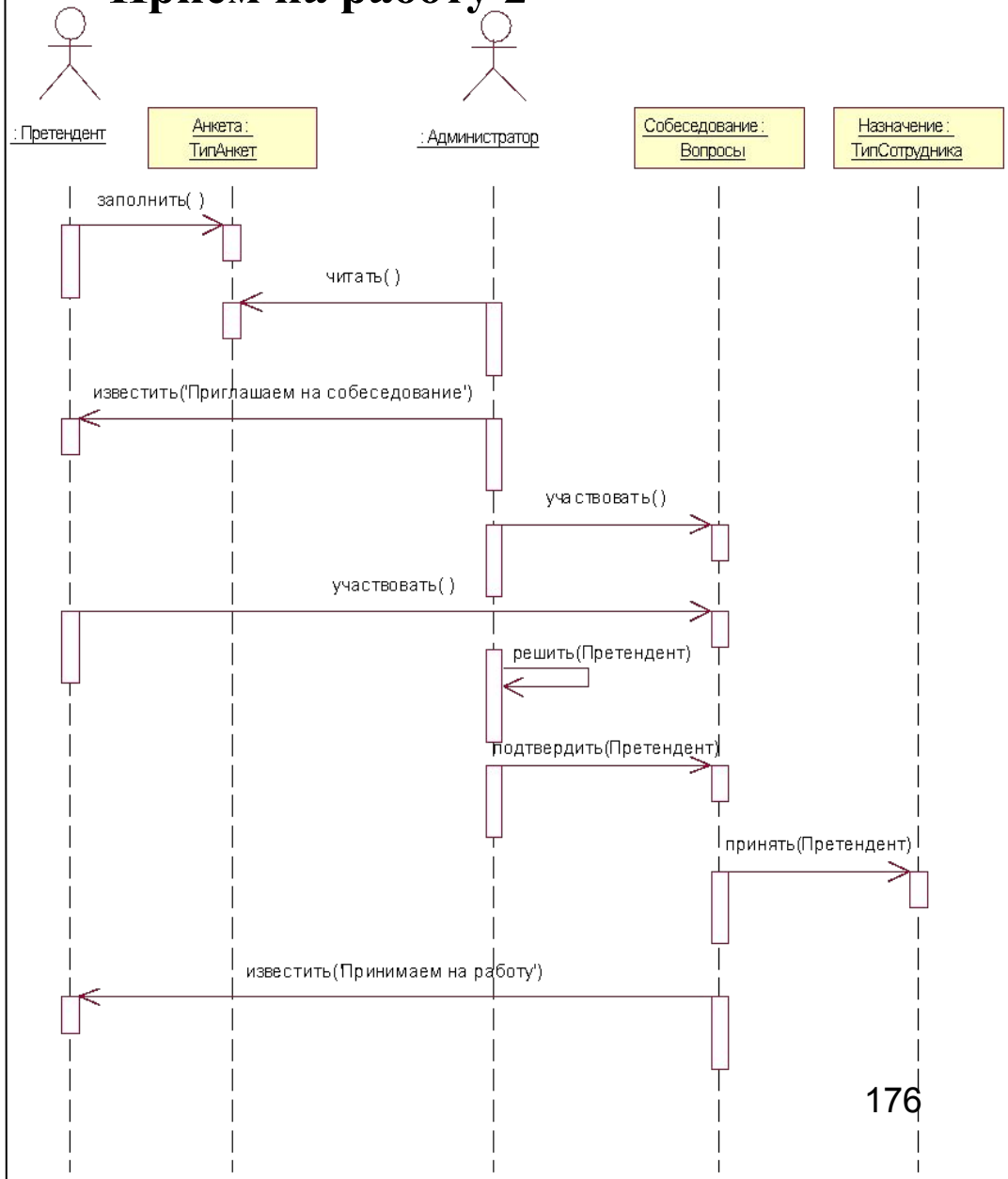
Диаграммы последовательностей и взаимодействий

Прием на работу 1



- Прием не зависит от результатов Собеседования
- Не предусмотрено извещение Претендента о приеме на работу.

Прием на работу 2



Прием на работу 2

Сценарий не отражает условности выполнения действия *подтвердить* и последующих за ним событий. Для учета этого обстоятельства можно поступить двумя путями:

- сопровождать сообщения условиями, показывающими поведение объектов в разных случаях (в примере
- обусловленность вызова метода *подтвердить*), и строить разные диаграммы для альтернативных вариантов поведения.

Когда какой подход применять:

- условие естественно укладывается в составляемую схему — первый путь;
- условность можно трактовать как отдельные сценарии — второй путь;
- использование диаграмм других видов (третий путь).

Анализ дополнительного требования (группы требований)

Получить унифицированное представление элементарных составляющих группы требований

Это делается путем погружения в систему существующих понятий и моделей.

При этом может потребоваться пополнение набора объектов, ревизия сценариев и прочее ⇒ нужна *трассировка новых требований* (пройти обычный путь)

Цель: выявить непротиворечивость или возможность непротиворечивости с существующим.

Да —> что-то принимается и решаются вопросы когда, какие ресурсы требуются для реализации. **Коррекция планов**.

Нет —> мотивируется отклоняющее решение.

11. Результативность программистской проектной деятельности

Понятие результативности проектной деятельности

Программистская деятельность в целом — производство средств автоматизации пользовательской деятельности.

- В какой мере обеспечивается требуемая автоматизация?
- Удовлетворяет ли она тех, для кого предназначена?
- Какой ценой, т.е. за счет расхода каких ресурсов это достигается?
- В какой мере опыт ведения конкретного проекта может распространяться на другие проекты?

Внутренний и внешний аспекты (по отношению к проекту) показателей результативности

Результативность — *оценочная, но не только итоговая характеристика проекта*

Способность компании организовывать результативные процессы разработки проектов → понятие *уровней зрелости компании*

Рабочие продукты

Рабочие продукты — все полезное, что создается в ходе развития проекта

- *Познаваемость продукта* — необходимо, чтобы пользователь продукта был в состоянии понимать:
  - *что можно* получать, используя предлагаемую систему, документацию и пр.
  - *чего нельзя* ожидать от них
  - *как активизировать* функции системы,
  - *как трактовать* (для применения) получаемые результаты;
- *Познаваемость процесса разработки продукта* — необходимо, чтобы разработчики были в состоянии:
  - развивать проект,
  - принимать согласованные решения,
  - сбалансированно выполнять коллективную работу.

- Программные результаты
  - предлагаемая пользователю система
  - сопутствующие инструменты
  - демонстрации и др.
- Документные результаты
  - Что это за документы?
  - Могут ли и должны ли они использоваться за пределами проекта?
  - Как минимизировать затраты на их производство?

Документные и программные рабочие продукты

- *Метафора идеальной документации*: это разработчик программного продукта; только он в состоянии давать адресные и самые квалифицированные консультации, конструктивно отвечать на вопросы пользователей
  - Документация — это *заместитель* разработчика при используемой программе →
  - *Критерий качества документации*: она тем лучше, чем точнее имитирует непосредственное взаимодействие разработчиков с пользователями
  - В разных ситуациях использования требуются различные консультации и разъяснения → и разные варианты документов
    (в частности: система помощи не заменяет документацию!)
- Виды рабочих продуктов проекта в первом приближении:
  - *Программные продукты*: целевая программная система, компоненты ПО для внешнего применения (библиотеки, библиотеки классов и др.), инструменты, появляющиеся в рамках проекта для тех или иных целей (например, для построения специальных структур данных);
  - *Документы для пользователей* — все бумажные и электронные текстовые материалы (возможно, с рисунками, схемами и т.д.);
  - *Документы для сопровождения и развития* — для поддержки работ, связанных с обслуживанием пользователей программного продукта.

Группы рабочих продуктов

- Все ли рабочие продукты перечислены? Безусловно НЕТ!
- Это лишь первая группа — ***целевые продукты***
- Вторая группа — ***продукты, отражающие процесс развития проекта***:
  - *Планы, контрольные мероприятия* и другие материалы, с помощью которых осуществляется управление развитием проекта;
  - *Задания, отчеты об их выполнении, технические предложения* и другие материалы, формирующие процесс выполнения проекта как коллективную деятельность;
  - *Соглашения, правила и регламенты коммуникаций* между разработчиками, которые используются в ходе выполнения проекта.
- Третья группа — ***продукты, отражающих наблюдение за проектом***:

Выделено три группы рабочих продуктов, отражающие три аспекта проекта:

- цель,
- процесс и
- наблюдение за процессом.

Эту классификацию не следует догматизировать

Наша квалификация используется для обсуждения уровней зрелости процессов разработки программного обеспечения

Уровни зрелости процессов разработки программного обеспечения

- Для гарантированной результативности выполнения проектов нужно иметь возможность убедиться в этом → на уровень рабочих продуктов обязательно выносятся не только целевая группа, но и описания процессов в разных формах, доступных для независимого от проектной деятельности изучения
- Отслеживание результативности — задача специальной деятельности, обеспечивающей процедуру сертификации организаций, выполняющих проекты, достоверной информацией.
  - *Исходным материалом* для этой деятельности являются все рабочие продукты проектов, в частности, документы, отражающие процесс развития и контроля реализации проекта.
  - *Основной сертификационный результат* — отнесение организации к одному из пяти уровней зрелости
- Модель зрелости процессов разработки программного обеспечения SW-CMM

1. *Начальный* (initial) *уровень*

- Находясь на начальном уровне, организация обычно не может обеспечить устойчивый процесс разработки и сопровождения программного обеспечения. В организации отсутствует культура управления, из-за неэффективного планирования и плохого согласования работ продуктивность производственного процесса непредсказуема
- Успешные проекты возможны, но как рабочие продукты оформляются лишь результаты целевой группы

Большинство start up групп находятся на начальном уровне. По мере развития возникает потребность быть более зрелыми

2. *Повторяемый* (repeatable) *уровень*

- Планирование и управление новым проектом базируются на опыте работы с подобными проектами
- Возможность воспроизведения успешных практик прежних проектов
- Если в организации как рабочие продукты оформляются лишь результаты целевой группы, то единственная возможность для такого воспроизведения — это назначение на новые проекты менеджеров, которые уже хорошо зарекомендовали себя в предыдущих проектах.
- Процессы развития проекта и наблюдения не отражаются в рабочих продуктах, а представлены лишь в фольклорном варианте.

Осознание неустойчивости такого положения приводит к стремлению все более точно фиксировать опыт документально.

В конечно итоге, повторяемый уровень диктует необходимость перехода к формированию рабочих продуктов для всех трех групп.

3. *Определенный* (defined) *уровень*, или *уровень становления*

- В организации принят стандарт проведения разработки и сопровождения программного обеспечения, в рамках которого обеспечена интеграция процессов построения и управления.
- Разработка и сопровождение полностью документированы, что позволяет организовать наблюдение и контроль выполнения проектов. (в CMM такой стандарт называется *стандартным производственным процессом организации*)
- Для конкретных проектов стандартный производственный процесс адаптируется с целью учета его особенностей:
  - определяются критерии готовности, качества и т.д., а также механизмы контроля.
- Руководство получает точную картину развития проектов
- Продуктивность производственного процесса характеризуется как *стандартная* и *согласованная*.
- К рабочим продуктам каждой из групп предъявляются требования:
  - они должны быть представлены таким образом, чтобы *специфика проекта явно отделялась от стандартизованного содержания*,
    то есть чтобы при производстве рабочих продуктов максимально повышалась возможность их независимого от проекта применения

4. *Управляемый (managed) уровень*

- Характеризуется внедрением в организацию количественных показателей качества как для программных продуктов, так и для процессов их разработки
- Производственные процессы оснащены средствами для проведения четко определенных и согласованных измерений
- Продуктивность производственного процесса характеризуется как *предсказуемая* (процесс функционирует в заданных и измеряемых пределах)
- Предполагается, что за счет количественных показателей качества удается организовать наблюдение за любой проектной деятельностью, которое позволяет удерживать траекторию в области допустимости.
- Дополнительные требования к рабочим продуктам этого уровня:
  - конкретизируется и получает статус обязательного стандарта группы продуктов, отражающих процесс развития проекта, а также измерения и наблюдение за проектом.

Количественные показатели и измерения полезны для оценки результативности. Но что и как нужно измерять? Попытки ответить так, чтобы в показателях качество отражалось с функциональной точностью к ощутимым результатам не привели. Неудачи обычно связывают с тем, что

- понятие «качество» не определено строго
- оно меняется от методологии к методологии
- субъективная составляющая качества превалирует над объективной.

Но причины глубже и принципиальнее: коренные причины проблем количественных показателей обусловлены существенной креативностью программистской деятельности, а надежных критериев качества любой деятельности такого рода просто не существует.

Вместо точных *измерений* в практике прибегают к *оценкам*, что зачастую (но далеко не всегда!) оказывается достаточным

5. *Оптимизирующий* (optimizing) *уровень*

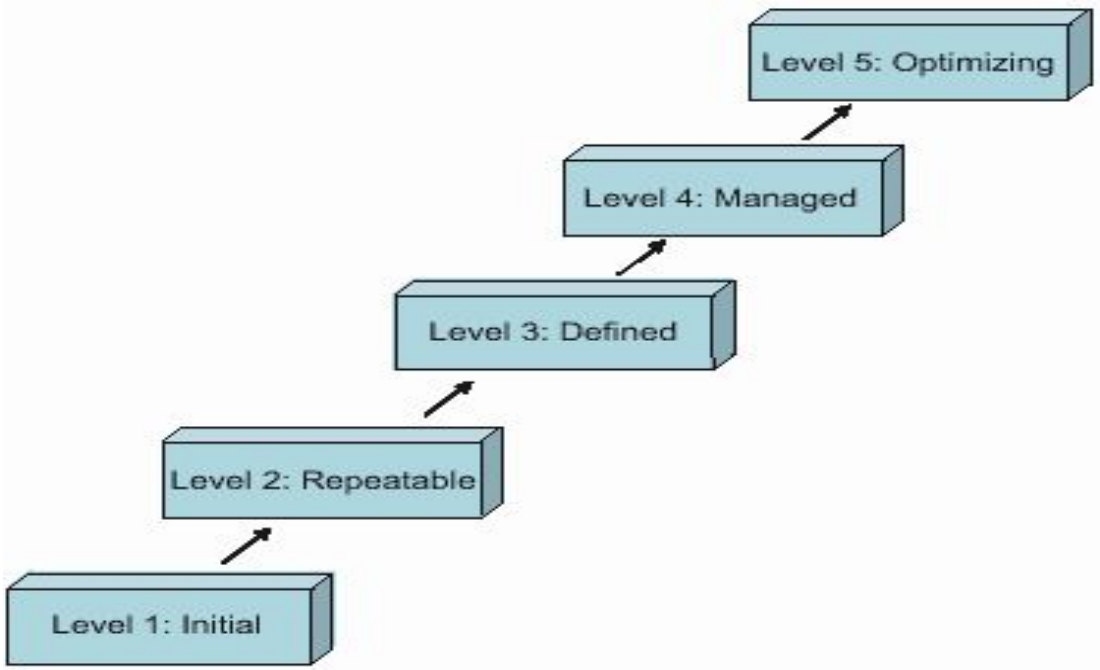
- Работа над проектами ведется как на управляемом уровне, но организация сосредоточена на *усовершенствовании производственного процесса*.
- Есть средства выявления слабых мест процесса и его улучшения с целью предотвращения дефектов.
- Для улучшения качества разработок проектов внедряются и распространяются новшества, передовые методы программной инженерии
- Налажены механизмы оценки не только выполненных проектов (это идет от предыдущего уровня), но и возможных новаций во всех аспектах проектирования

Таким образом, ассортимент используемых рабочих продуктов не ограничивается тем, что появляется по ходу выполнения целевых продуктов

Необходимые для оптимизирующего уровня продукты могут разрабатываться специально

Оптимизирующий уровень иногда называют «*нирваной проектной деятельности*»

Лестница развития зрелости организации



- Модель CMM предполагает, что организация, согласившаяся с принятым подходом, берет обязательства продвигаться вверх по лестнице развития зрелости
- Эволюция представления о том, какими должны быть рабочие продукты

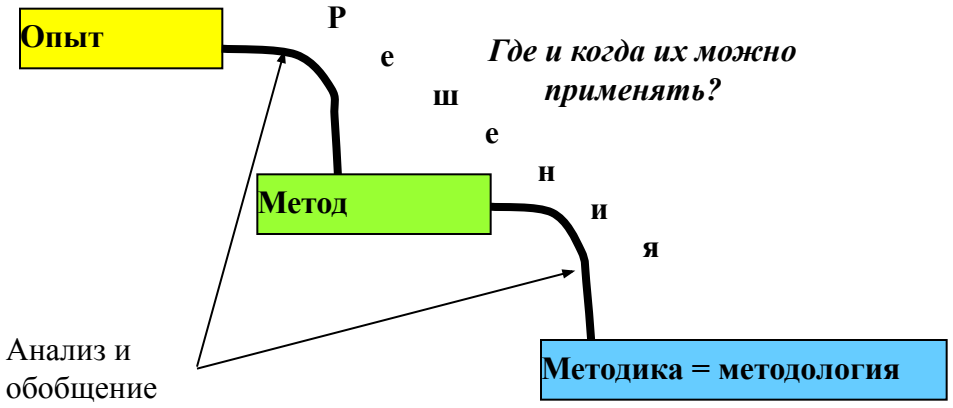
Критика: стремление втиснуть все схемы менеджмента в прокрустово ложе унифицированных процедур слежения за развитием проектов с помощью раз и навсегда «хорошо себя зарекомендовавших» практик

Что ожидают от модели развития зрелости организации CMM?

- Успешными чаще оказываются проекты, в которых уделяется внимание сопутствующим продуктам.
  Отсюда *впечатление о причинно-следственной связи, зависимости успешности от дополнительных продуктов*
- Стремление к технологизации на основе опыта (+ возможность «объективного» сравнения → независимого контроля)

- Эту попытку демонстрирует модель CMM
  - Предполагается, что стандартизация процессов, развивающаяся по мере развития организации, способствует качеству этих продуктов и совершенствованию методик их разработки.
  - На деле оказывается, что
    опыт хороших решений в одних ситуациях совершенно ничего не дает в условиях других проектов

Формирование методов и методик путем анализа и обобщения решений



Проблемы этого процесса:

1. Искушение распространения удачного опыта за пределы его области применимости
2. Многие методы противоречивы и при их объединении в методике вместо ожидаемого дополнения достоинств происходит их нейтрализация, и пышным цветом расцветают недостатки объединяемых методов
3. Груз стандартов

Комментарий к 1

• Перенос опыта или метода

Комментарий к 3

• Успешность применения ведет

Комментарий к 2

Примеров множество:

– несовместимость стилей (языки программирования),
– поддержка противоречащих методик (CASE-системы)
– требование измерения качества без определения того, что это для данного проекта (методологии)

нотации позволит решать проблемы

реализационные диаграммы классов UML

12. Управление рисками

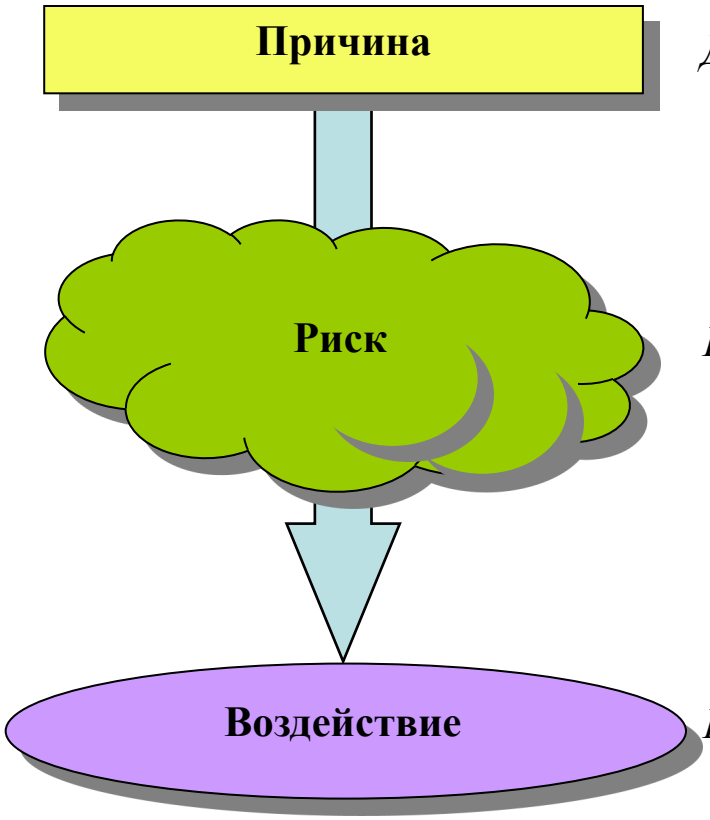
Понятия, связанные с рисками

- *Причины риска* — определенные события или обстоятельства в проекте или в его окружении, которые вызывают *неопределенность*
  - технические (сбои и задержки поставок и др.);
  - политика руководства фирмы и заказчика (компетентность, расчет на удачу, кредитоспособность, репутация и др.);
  - рыночная конъюнктура;
  - внутренние причины рисков:
    - сбои в используемом окружении,
    - неточность предъявляемых требований,
    - ненадежность подрядчиков,
    - ненадежность кадров.
- *Риск* — может привести к воздействию на процесс достижения целей проекта:
  - к негативному воздействию, если в результате траектория проектной деятельности выходит из области допустимости,
  - к нейтральному воздействию, если траектория не выходит из этой области,
  - к позитивному воздействию, новая траектория не только остается допустимой, но и по разным причинам оказывается предпочтительнее других траекторий, которые могли бы реализоваться, если бы рисковая ситуация не возникла;
- *Последствия* — незапланированные отклонения траектории выполнения проекта, возникшие в результате того, что событие или обстоятельство, квалифицированное как риск, произошли.

Управление рисками

- Project Risk Management — процессы, обеспечивающие
  - планирование возможности рисков,
  - их идентификацию,
  - анализ,
  - разработку откликов и
  - контроль в течение жизненного цикла проекта
- Риски — неопределенные события или обстоятельства, возникновение которых может привести к воздействию на процесс достижения целей проекта
- Реальные риски — это те события, которые действительно характеризуются неопределенностью
- План управления рисками — идентификация рисков данного проекта и мероприятия, снижающие зависимость его от рисков:
  - Идентификация рисков ⇒ список идентифицированных рисков
  - Выставление приоритетов рискам ⇒ определяются риски, которые будут отслеживаться в проекте. Остальные не отслеживаются, но игнорируются обоснованно (реально можно отследить не более 10-15 рисков)
  - Установление возможности влияния на риски (на причины и на воздействия)

Схема связей составляющих риска



- ***В результате (по причине)***
  <определенное событие>
  - вместо этого нужно подставить конкретную причину идентифицируемого риска
- ***может случиться***
  <риск>,
  - нужно подставить наименование риска
- ***что может привести к***
  <воздействию>
  - нужно подставить варианты того, что может произойти

Уровни влияния на риски

1. ***Исключение риска*** (avoid). Некоторые рискованные ситуации могут быть исключены полностью. К сожалению, невозможно исключение всех рискованных ситуаций.

   Исключение риска осуществимо не всегда, но при планировании работы с рисками для каждого из них следует попытаться найти варианты исключения (преодоление выявленных причин возникновения риска, т.е. создание условий, которые приведут разрыву связи причина — риск).

2. ***Переключение риска*** (transfer) — частный случай исключения, когда риск переносится из сферы влияния проекта на окружение. К этому уровню относятся все варианты контрактных соглашений, но не только они. Оперируя рисками нужно стараться предусмотреть способы переключения.

   К сожалению, эта стратегия является исключением риска только для разработчиков, но не для проекта в целом.

3. ***Уменьшение риска*** (mitigate). Если риск не может быть исключен, можно постараться уменьшить вероятность его появления на практике (оперирование с причинами). Нужно, чтобы подобные решения делались не в ответ на ситуацию, а заранее.

   Пример уменьшения риска — объявление (для заказчика, руководства и коллектива) о пересмотре требований, когда становится понятным, что график выполнения работ может быть сорван

4. ***Предупреждение ущерба от риска*** (accept). Когда не получается удовлетворительно уменьшить риск, следует подготовиться к встрече неприятности так, чтобы минимизировать потери, т.е. снизить вероятность негативного воздействия и уменьшить само воздействие (оперирование с последствиями). Если это удается, то продолжение проекта во многих случаях оказывается успешным

Планируемые *отклики на риски* — мероприятия на указанных уровнях в различной комбинации, возможно, дополненные более тонкой реакцией на возникновение риска

В результате рискового события, влияния на него, а также его воздействия на проект возможно появление, так называемых, *вторичных рисков*, т.е. тех, которые без этого не могли бы возникнуть. Их также следует оценивать, для них также нужно предусматривать влияния.

Исполнители должны быть осведомлены о возможных рисках и о плане управления ими.

Ситуации, которые нужно избегать и учитывать, составляя план управления рисками

- Задержка начала проекта никогда не компенсируется;
- Если сетевой график выполняется с большими нарушениями сроков, трудно ожидать создание хорошего продукта;
- Если пользовательский интерфейс не является интуитивно понятным и превышает уровень компетенции потребителя изделия, продукт будет плохо распространяться;
- Упущенные возможности требуют дополнительных усилий при их более поздней реализации и увеличивают затраты;
- Не протестированный продукт снижает репутацию разработчиков;
- Не стоит рассчитывать на постоянство пользовательских намерений, никогда нельзя знать, что хочется пользователю и чего на самом деле ему нужно.
- Как следствие, необходимо планировать время на переделку того, что в начале проекта казалось приемлемым.

Управление качеством проекта

12. Организация коллективной работы

Методы организации труда программистского коллектива

Примитивная организация:
Задание — (выдается менеджером)
Выполнение — (деятельность работника)

- ***схемы с разделением ответственности*;**
- ***деперсонифицированные схемы*;**
- ***смешанные схемы***

распределяются между

***Сферы ответственности*:**

- охватывают все задачи проекта
- включает задачи проекта, тематически связанные по методам решения, либо задачи, охватывающие разработку обособленных единиц декомпозиции.

(часто конкретизируются как программный компонент, функция и др.)

ответственными исполнителями — назначаются для руководства сферой ответственности исходя из ролей, квалификации, занятости и др.

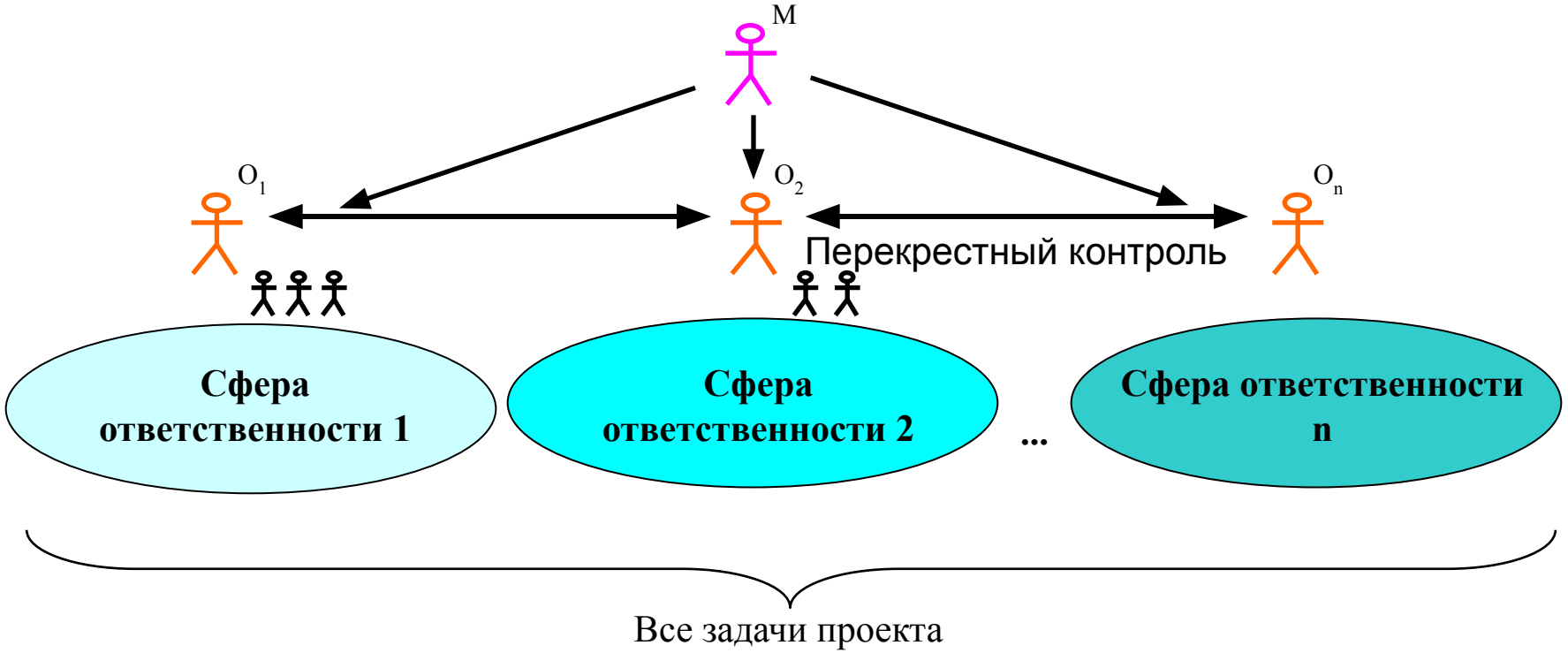
Понятие сферы ответственности

Сферы ответственности (коллектива, работника и др.) ≠ проектное распределение работ

- Работник, получивший сферу ответственности, должен стать экспертом для задач данной сферы.
- Способ решения задач — в рамках выделенных ресурсов.
- Знание других сфер сверх необходимого — выходит за рамки проектного разделения труда, но полезно.
- ***Контакты менеджера с группой ограничиваются контактами с ответственным исполнителем* (ОИ).**
  - Нет нужды углубленно вникать в задачи сферы.
  - ОИ свои действия согласовывает с менеджером.
- ***За проект в целом по-прежнему отвечает менеджер!***

Степень риска проекта возрастает ⇒
нужны мероприятия, направленные на снижение рисков

Схема с разделяемой ответственностью

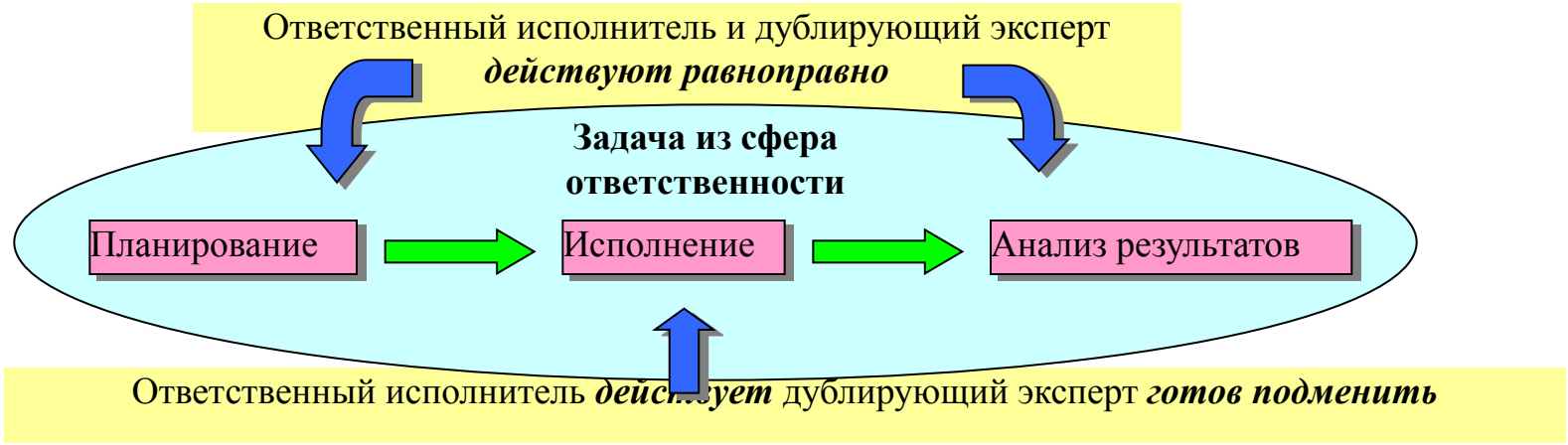


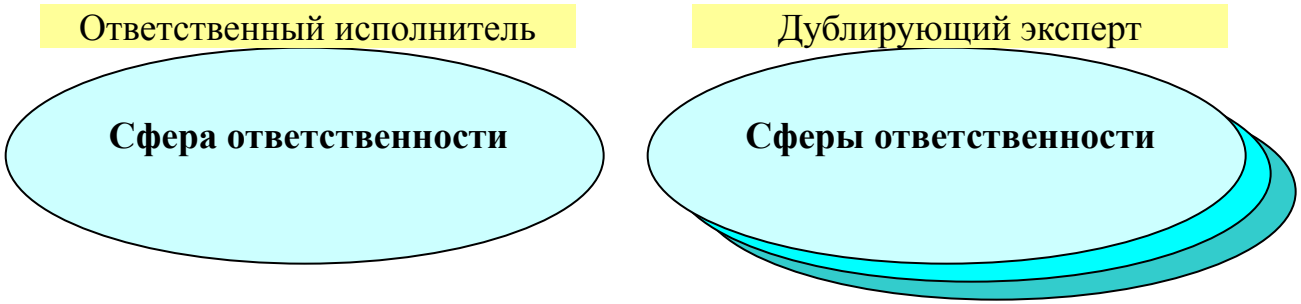
Мероприятия, направленные на снижение рисков:

- перекрестный контроль,
- ответственный исполнитель и дублирующий эксперт:
  - *первый и второй пилоты,*
  - *хирургическая бригада* (специализация работников).

/см. например, кн. Ф. Брукс «Мифический человекомесяц»/

Взаимодействие ответственного исполнителя и дублирующего эксперта





Команда летчиков и хирургическая бригада

Ответственный исполнитель — первый пилот

Дублирующий эксперт — второй пилот

- первый пилот *принимает решения*;
- второй пилот отслеживает действия первого, помогает советами, *находится в постоянной готовности подменить* первого пилота, который является ответственным исполнителем;
- остальные работники *находятся в подчинении первого пилота* за исключением случаев
  - подмены первого пилота и
  - делегирования второму пилоту некоторых функций ответственного исполнителя

Ответственный исполнитель — главный хирург

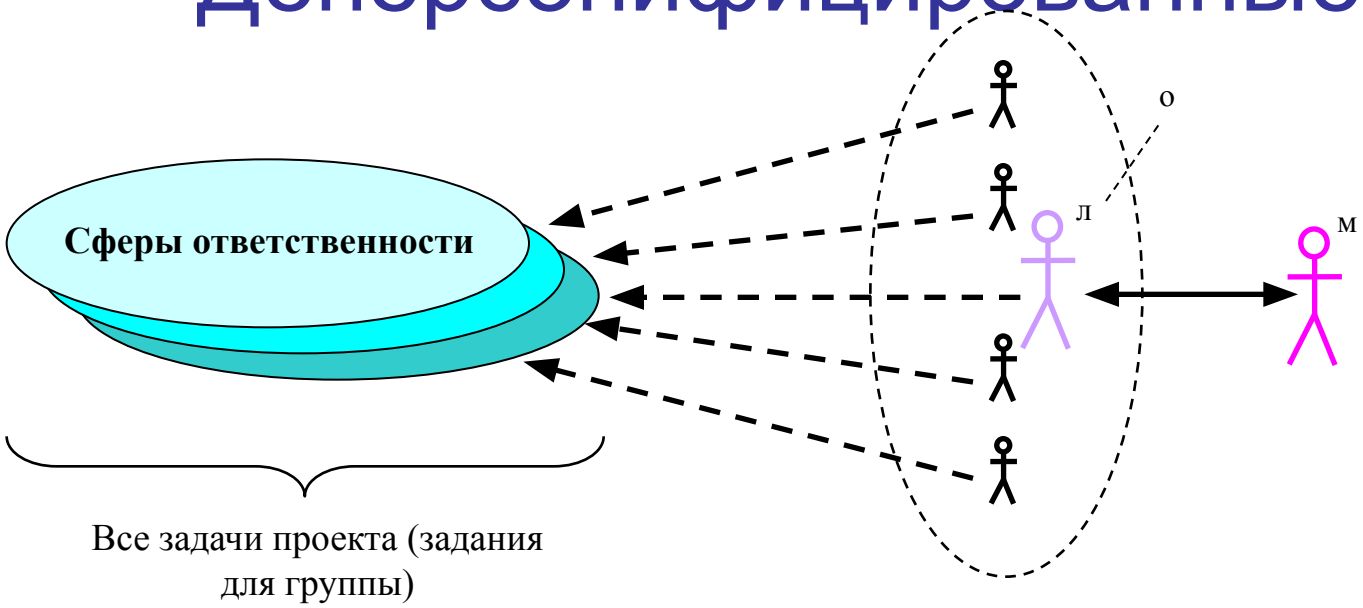
Дублирующий эксперт — ассистент

- непосредственные оперативные действия выполняет *главный хирург*;
- *ассистент*, готовый в любую минуту заменить главного хирурга, наблюдает за процессом;
- *другие работники* выполняют не персонифицированные приказы главного, а специализированные распоряжения, строго соответствующие фиксированному распределению ролей.

В обоих случаях соблюдается принцип равноправия

Функции дублирующего эксперта можно получать лидеру

Деперсонифицированные схемы



Проекты, требующие концентрации «умственных ресурсов» (наукоемкие, экспериментальные и др.) — область, где схема может хорошо работать

Проекты, которые требуют концентрации «умственных ресурсов» (наукоемкие, экспериментальные и др.) — область, где эта схема может хорошо работать

- Проектные решения принимаются коллективно (не голосование!),
- Каждый участник вникает во все ключевые задачи проекта и может квалифицированно аргументировать выбор принятых решений.
- ⇒ *Ответственность за части проекта распространяется на весь коллектив независимо от авторства решений и их реализации*.

Сравнить с рабочей группой MSF

Ответственность за проект по-прежнему лежит на менеджере!

Условия применимости деперсонифицированных схем

+ Сработанность коллектива,
+ Психологическая совместимость работников,
+ Общий интерес и ***единая целевая установка*** на выполнение проекта с максимально высоким качеством,
+ Каждый работник может рассчитывать на всех остальных,
+ Все могут рассчитывать на каждого
+ Наличие признанного лидера коллектива (снижает риск невыполнения). Лидеру поручается
  - организация форм коллективной ответственности
  - распределение группового задания
  - выполнение внутренних для проекта (задания) функций менеджера
− Велик риск невыполнения (больше, чем при разделении ответственности)
− Пригодны для небольших коллективов (до 10 человек)
− Не очень большое (обозримое) время
− Проблема карьерного роста

Но они весьма эффективны при воплощении новых идей, методов!

Проектная рабочая группа MSF и деперсонифицированная разработка

MSF вариант деперсонифицированной схемы выделяется тем, что

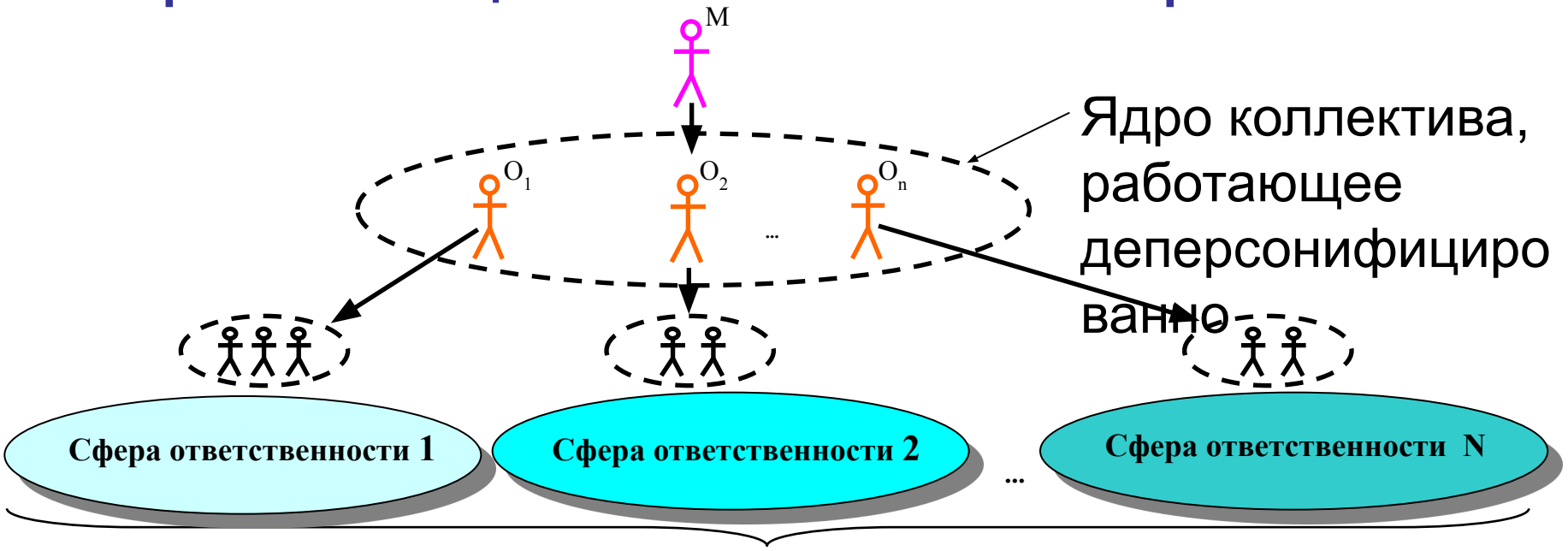
- Внешний менеджмент «видит» единого исполнителя
- Вместо ролей — ролевые кластеры, маскирующие фактическую структуру группы
- Плохо проработана кооперация групп

Понятие сфер ответственности сохраняет свободу выбора вариантов схемы

Характерное для всех вариантов

- не обсуждается, как формируется проектная рабочая группа;
- взаимоотношения в группе остаются за рамками рассмотрения.

Смешанные схемы и планирование организации коллективной работы



Иерархическое распределение менеджерских задач по группам исполнителей, у которых складывается самостоятельная организация коллективной деятельности (в явном виде это постулируется в предложениях MSF).

Из определения следует возможность различных смешанных схем, любая из них отражает некоторую иерархию взаимоотношений в коллективе.

Оформление взаимоотношений по смешанной схеме

Как появляется:

- Из деперсонифицированной схемы
- Предусматривается для проекта заранее

Схема планируется исходя из информации о проекте и его исполнителях:

- содержание, сложность и объемность проекта;
- фактический состав коллектива исполнителей:
  - наличие или отсутствие слаженных команд,
  - кадровая укомплектованность,
  - квалификационная избыточность/недостаточность проекта,
  - другое;
- сведения о занятости ключевых ролей и о персоналиях, занимающих эти роли
  - укомплектованность,
  - квалификация,
  - психологические особенности.

Другие варианты смешанной схемы возможны

Выбор типа организации коллективной работы – задача менеджера

Стихийный подход или подход по предписанию «сверху» противопоказаны

Перечисленные факторы, влияющие на выбор, оставляют для менеджера весьма немного вариантов

Априорно существующие иерархии в коллективах

Иерархии и отношения, образующие иерархии

1. Общий проект: обязательства, общие и специальные соглашения, связи работ
   ⇒ Иерархия по отношению к атрибутам проектной деятельности — ***проектная иерархия***

   Отношение *использования* задачами и заданиями + между исполнителями (роли)
   - Динамичность проектной иерархии
   - Подчиненность управленческой деятельности и текущим целям проекта
2. Административное подчинение сотрудников (отношение *субординации*) ⇒ ***административная иерархия***
   - Карьерный рост,
   - Назначения работника,
   - Административная отчетность
   - Сотрудник вынужден отвлекаться на работы, выходящие за пределы проектных задач
   - Консервативность
   - Перемещения планируются ⇒ можно к ним подготовиться ⇒ влияние менеджера необходимо
3. Структура базирующаяся на взаимоотношениях, которые есть следствие архаических качеств индивидуумов ⇒ ***стайная иерархия***

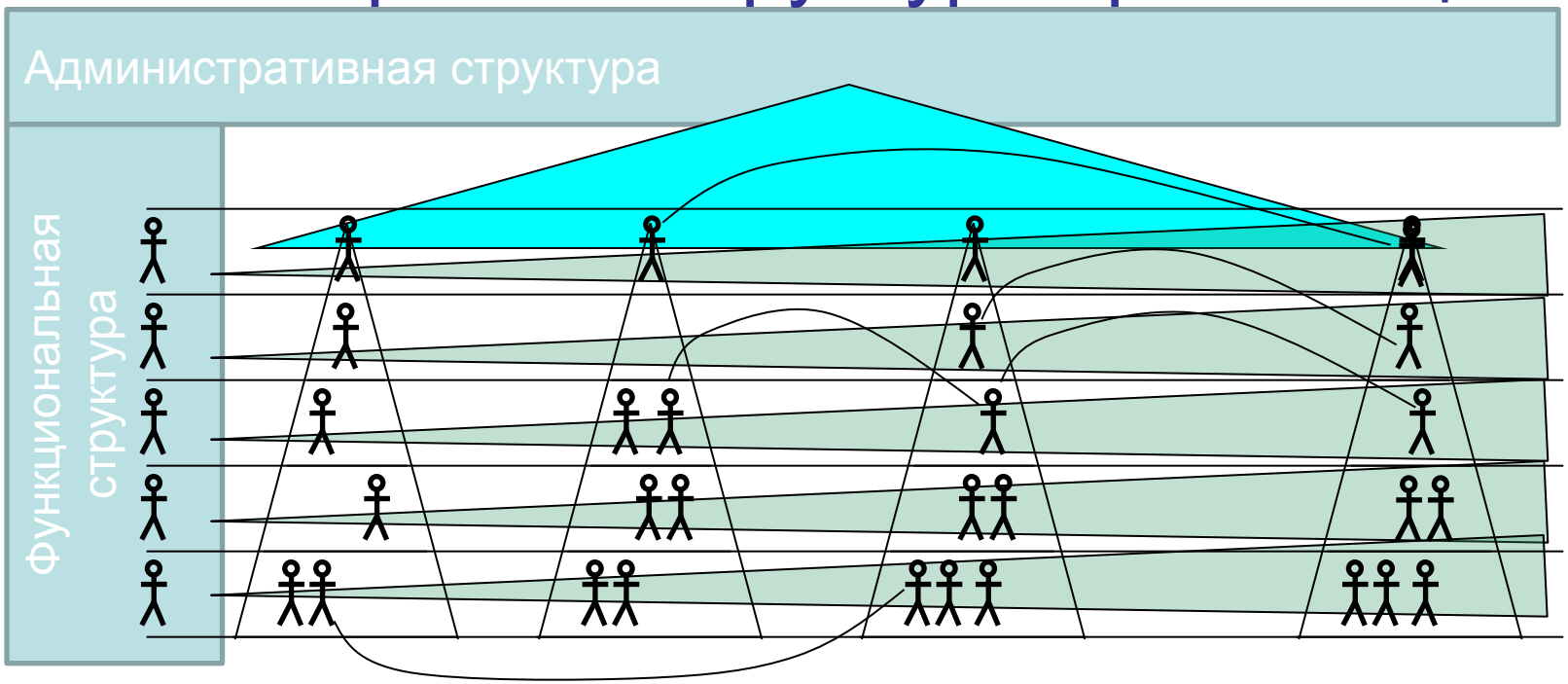
   Отношения *стремления особей* к лидерству и др. позициям в стае
   - Не может способствовать сплочению коллектива вокруг целей проекта (деструктивность) ⇒ нужно предпринимать определенные шаги
   - Сильные и слабые стайные качества индивидуумов
   - Динамичность: чередование периодов формирования, стабильности и разрушения
   - Роль неформального лидера

Игнорирование базовых иерархий

- Проектные и административные иерархии часто и даже не различаются
  - Карьерный рост, и административное подчинение определяются успехами проектов
  - Но проектные целевые установки не совпадают с административными интересами сотрудников

  ⇒смешивание двух структур или игнорирование одной из них (как правило, административной)
- Утверждается, что административная структура должна способствовать проектной деятельности, но она складывается исторически ⇒ возможно противоречие с проектной иерархией
- *Банда*: команда собирается на «дело», потом разбегается.
  - Для разовых и небольших проектов может быть устойчива
  - Но есть проблема развития, когда теряется управляемость (какая?)
- *Функциональная организация*: производственные функции проекта становятся подразделениями, ответственными за эти функции
  - Команда для нового проекта из представителей разных подразделений, отчетность перед функциональными менеджерами-администраторами.
  - Менеджмент проекта распределяется по ответственным за выполнение функций ⇒ потеря главенствующей роли проектной иерархии.
  - противодействие — *матричная организация* (варианты)

Матричная структура организации



На первых порах схема дает результат, но в случаях работы над программистскими проектами она быстро превращается в бюрократическую надстройку

Альтернатива – использование неформально структурированных рабочих групп, деперсонифицировано отвечающих за проект перед вышестоящей инстанцией.

- Здесь образовываются не административные, а проектные структуры, иерархия которых для организации считается непринципиальной.
- Это хорошо работает, когда группа отделена от административной иерархии

Подтверждение непродуктивности смешивания двух видов иерархий

Взаимодействие со стайной иерархией

- Деструктивность стаи
- Но если заранее знать сильные и слабые стороны членов команды как индивидуумов, то
  - можно использовать их для поддержки развития
    - проекта,
    - коллектива,
    - находить противодействие деструктивному поведению.
  - Динамичность стайной иерархии имеет характерную специфику: чередование периодов *формирования*, *стабильности* и *разрушения*.
    - В период формирования можно существенно повлиять на расклад сил в коллективе,
    - В период стабильности можно только учитывать его.
    - В период разрушения жизненно необходимо четко осознавать потери и ростки новой иерархии.
  - Воздействия, направленные на сохранение недеструктивной стабильности
  - Шаги для разрушения деструктивной иерархии (решительные или сдержанные)
  - при благоприятных обстоятельствах можно с успехом переложить часть работы со стаей на ее лидера

Влияние основных иерархий на деятельности исполнителей проекта

- Проектная, административная и стайная иерархии в большей степени, чем все другие взаимоотношения в сообществах влияют на проектную деятельность
- Они первую очередь должны приниматься во внимание при работе руководителя
- Учет других отношений и иерархий (например, возрастной структуры, уровня образованности и др.) также необходим, но он вторичен и осуществляется в рамках, которые задают три основные иерархии.
- В случае согласованности целей и интересов они дает самый существенный импульс для Дела,
- Именно они препятствуют успеху, когда их интегральное воздействие на проект оказывается разнонаправленным.
- Учет человеческого фактора, создание мотиваций для работников и др. не работают, если возникает противоречие основных иерархий коллектива
- **Первичная задача руководителя** — организация коллективной работы, согласованной с устремлением всех трех иерархий
- *Большинство рекомендаций дается с доминированием проектной иерархии, иногда говорят об административной иерархии, и практически все «забывают» о глубинной основе взаимоотношений — об архаических инстинктах и связанных с ними устремлениях*

Почему надо учитывать взаимоотношение иерархий

- Управление, а значит, и проектная иерархия объективно занимает главенствующее положение среди других взаимоотношений.
  - Есть опасность, что не обращать внимание на другие стороны руководства, отдав их стихийному способу формирования;
- Программные проекты по своей креативной сути практически всегда развиваются недетерминировано.
  - Это требует большего внимания при проведении корректирующих воздействий на ход развития проекта:
  - Сводить такие воздействия лишь к управляющим мероприятиям не только неразумно, но во многих случаях губительно для формирования требуемого микроклимата в коллективе;
- Для руководителя формирование слаженного коллектива, возможно, даже более значимая задача, чем успешность развития проекта (последнее — положительный побочный эффект).
  - Такой коллектив рассматривается окружением как весьма влиятельная сила; ему могут поручать все более ответственные задания.
  - Это обстоятельство само по себе сказывается положительно на многих аспектах проектной деятельности.

Творчество vs. технологии и базовые иерархии

- Творчество не может быть коллективным. Это всегда авторский процесс:
  - 1 автор или соавторы
    - сплав, «личный вклад» – резать по живому
    - автор как соавтор сам себе /исправление сделанного/
    - 1, 2 автора – устойчиво, 3 – бывает, 4 – очень редко (начинается ролевая дифференциация)
    - Учитель *никогда* не приписывает себя соавтором учеников, *редко* (при необходимости) приписывает соавторами учеников, *всегда* указывает явное соавторство
  - Автор «заражает» исполнителей на творчество /не только/ (в хорошей пьесе при блестящей постановке актеры «выкладываются») → иллюзия коллективного творчества
- Стайное + проектное = вектор созидания
  инстинкты | замыслы | развитие
  Административное – лучше не мешать | поддержка | (часто это медвежьи услуги, благие пожелания, но бывает и помощь)
- Технология – разделение труда и функций. Для успешности даже ее цели знать не обязательно
- Креативность противопоказана, но остается при неопределенности:
  - обучение
  - управление рисками
- Она может способствовать творчеству – рационализаторство /другая деятельность/
- Менеджмент – не автор продукта, а «надзиратель» /другая деятельность/:
  - стабильность
  - трансляция указаний (вниз) и результатов (вверх)
- Ответственному исполнителю *предписано* указывать всех исполнителей проекта, задачи и др., иногда требуются сведения о «личном вкладе»
- Общая направленность проектного и административного способствует успеху. Стайное, как правило, подавляется; лучше, если переносится в другую сферу (определения инициативности и др. цели)

14. Взаимодействия разработчиков проекта

Понятие локальных взаимодействий

Общение сотрудников, цели которого связаны с выяснением**:**

- что и как надо делать,
- что и как из результатов коллег надо использовать,
- какие связи в проекте влияют на работу,
- а также многого другого, без чего проект не мог бы развиваться.

Источники сведений:

- Документы
- Приказы и распоряжения (документы особого рода)
- Контакты разработчиков

Решения:

- Положения, *утвержденные руководителем* как соглашения и регламенты, предписания и приказы, иные материалы

Решение в процессе их формирования:

- Предварительное решение (предложение)
- Предложение в согласованной форме
- Окончательное решение (может быть изменено)
- Утвержденное решение (может быть отменено другим решением)
- Исполняемое решение (регламент выполнения некоторых деятельностей)

Принцип осмысленности действий (напоминание)

Принцип: всякий раз, когда работнику приходится выполнять какую-либо работу, он должен четко осознавать, зачем он это должен делать и делает

Почему он далеко не всегда выполняется?

Подмена целей

- Тот, от кого исходит предложение или приказ, скорее всего, осознает, зачем ему это нужно
- Но надо еще сопоставить свое «*зачем*» с системой целей и приоритетов работника, которая, очевидно, не совпадает с его целями.
- Предлагающий работу часто полагает, что это не так, и он может быть искренне убежден, что, раз уж работнику объяснили нечто, он это обязательно воспринял, и обязательно будет соблюдать предписанные правила.

Осмысленность и целенаправленность

- **Целенаправленность**: цель действий определена и известна работнику
- **Осмысленность**: соответствие целей задания (внешних) внутренним целям и установкам (индивидуальным). *Подмена целей — когда это не так*.

Мотивация к цели имеет косвенное отношение (может сформировать цель), но не обязательно соответствующую цели задания. Опасность вместо выполнения получить его имитацию.

Необходимы **приемы и методы** доведения заданий до разработчиков.

Нужно учитывать **варианты подмены целей**, включая как простое недопонимание, с одной стороны, а с другой — прямой, но невысказанный саботаж.

Требуются **особые подходы**. Это один из существенных аспектов квалификации.

Принцип осмысленности действий не сводится *только* к индивидуальным воздействиям:

- коллективное обсуждение,
- открытое распределение обязанностей и др.
- **Но не в случае авторитарного и директивного руководства.**
- Осознанность – это, когда решения приняты на индивидуальном уровне, всеми участниками

эффективно стимулируют осмысленность действий («на миру и смерть красна!»).

Мы ратуем за консенсус в руководстве коллективом.

Цели производственных контактных мероприятий

Требуется представить рекомендации, отвечающие на следующий вопрос:

как должны быть организованы контакты в коллективе,

чтобы проектные решения

- ***принимались точно***,
- ***корректно понимались*** разработчиками и
- ***выполнялись с максимальным*** производственным ***эффектом***.

Ситуации принятия проектных решений

1. Решение сформулировано, принято и передано заинтересованным лицам — ***целевая ситуация решения, переданного для выполнения***;
2. Решение сформулировано и принято менеджером, но передано не всем заинтересованным лицам — ***ситуация распространения подготовленного решения***;
3. Решение сформулировано и принято менеджером, но не передано заинтересованным лицам — ***ситуация принятого решения***;
4. Решение не сформулировано, но есть основание считать, что для выработки его нужно прислушаться к мнению некоторых из заинтересованных лиц — ***ситуация формулировки решения***;
5. Решение не может быть сформулировано по причине затруднений — ***ситуация неопределенности решения***.

+ *Утверждение решения* (оно становится окончательным)

Мероприятия для поддержки принятия проектных решений

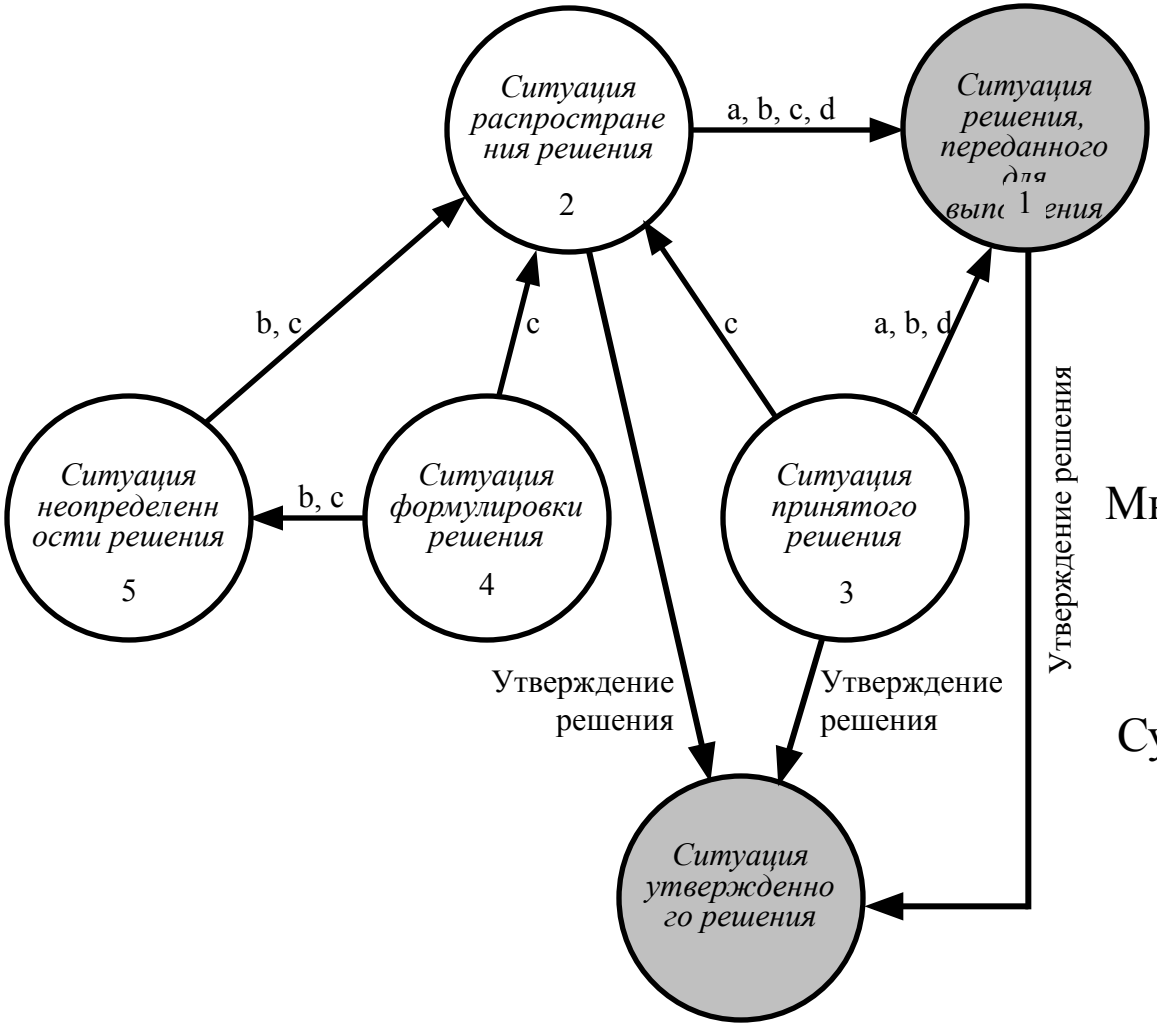
a) *оперативное совещание* — собрание заинтересованных лиц для извещении о решении (когда не требуется обсуждение);

b) *коллективное обсуждение* — собрание заинтересованных лиц с целью получить их мнение;

c) *индивидуальные обсуждения* — беседы, проводимые, когда мнение о решении легче получить без привлечения коллектива (часто предшествуют коллективным обсуждениям или совещаниям);

d) *поручение* ознакомленным сотрудникам оповестить заинтересованных лиц. Используется, когда можно обойтись без участия менеджера в распространении решения.

Схема возможных изменений ситуаций при принятии проектных решений



a) *оперативное совещание*

b) *коллективное обсуждение*

c) *индивидуальные обсуждения*

d) *поручение*

Много форм коллективного обсуждения: от никак не организованной беседы до четко регламентированного собрания

Существенными (с точки зрения организационных методик) являются только те обсуждения, рамки проведения которых повышают результативность мероприятия

Мозговой штурм

Коллективное обсуждение проблемы, при котором собирается и фиксируется как можно больше мнений по обсуждаемому вопросу для последующей обработки.

Особенность — абсолютное равноправие участников. Если кто-то своим авторитетом, темпераментом и т.п., может мешать равноправию, то нужны компенсирующие меры (давать ему слово в самом конце обсуждения и др.), а когда это не получается, лучшее решение — удалить его из числа участников.

Правила организации мозгового штурма

- Равноправие
- Секретарь сессии следит за регламентом:
  - общий лимит времени сессии,
  - лимит времени одного выступления,
  - соблюдение общих правил и соглашений о предмете разговора
- Коллекция мнений
- Порядок:
  - начинается с объявления предмета и целей, знакомство с регламентом сессии;
  - Краткие выступления каждого участника для изложения одной идеи (допускаются и приветствуются любые высказывания, кроме дискуссионных и критических);
  - Все высказывания фиксируются;
  - Если цель оценить что-либо, оценки за и против фиксируются раздельно;
- Обработка результатов сессии
- Сокращение предложений: сортировка, голосование или взвешенная сортировка:
  - экспертам предлагается распределить идеи на бесполезные, сомнительные, интересные, существенные, важные и т.п.;
  - каждой категории приписывается вес: 1, 2, ...;
  - для каждой идеи суммируются веса и упорядочиваются собранные высказывания.
- Выработка заключения о сессии

Ролевые игры

Ролевая игра — коллективное мероприятие, в котором участникам назначаются определенные роли

- Правила и регламенты оформляют коллективную игровую деятельность (возможно, несколько) и совокупность ролей, выполняемых участниками.
- С каждой ролью связывается линия поведения, участники не имеют права выходить за пределы, установленные ролью.
- Иногда используются материальные атрибуты (различные предметы, транспаранты и др.). Это средства игровой деятельности, знаки понятий и реальных объектов, моделей таких объектов (способствует установлению ассоциативного ряда у участников $\Rightarrow$ эффективность мероприятия повышается (если удается чем-то компенсировать отвлечение внимания на знаки).
- Ролевые игры оживляют обсуждения, создают стойкие образы игровых ситуаций, связывая их с конкретными персоналиями, моделируют реальные взаимоотношения. $\Rightarrow$ методики ролевых игр часто применяются при обучении и на тренингах
- Деловая и ролевая игры — вопрос терминологии
- Множество типов ролевых игр

Антагонистические ролевые игры

Обсуждаются *варианты* (решения, плана и пр.) с цель поиска принимаемого варианта как *решения*

Разделение участников на две или три группы:

- *Сторонники* каждого из вариантов. *Разрешено только позитивное отношение к своему варианту, положительные следствия его принятия, но запрещена критика*;
- *Противники* каждого из вариантов. *Вменяется в обязанность искать и делать достоянием собравшихся только недостатки вариантов*;
- *Наблюдатели* (желательная группа, которой можно пожертвовать при недостаточном числе участников). Задача — *сравнивать доводы сторонников и противников. Им разрешено высказывать мнения, но лишь на основании аргументов сторонников и участников*.
- *Секретарь* игры. Следит за соблюдением регламента и контролирует:
  - общий лимит времени сессии,
  - лимит времени одного выступления,
  - соблюдение правил игры и соглашений о предмете разговора,
  - соблюдение правил поведения сторонников, противников и наблюдателей.
- *Совмещение ролей*:
  - секретаря и наблюдателя возможно,
  - секретаря с другими ролями — потеря качества,
  - другие совмещения невозможны принципиально

Общие правила организации антагонистических игр

- Правила организации антагонистической ролевой игры могут варьироваться (предмет обсуждения, число участников, количества тем, лимит времени и др.)
- Игра эффективна, когда для обсуждения выносятся хорошо определенные варианты решения. В этом случае
  - Для каждого варианта должен быть по крайней мере один сторонник.
  - Противники могут быть у каждого варианта свои, но допускаются и объединенные противники.
  - Наблюдатели не могут быть ни сторонниками ни противниками вариантов.
  - При выполнении роли сторонников и противников личные пристрастия подавляются.
- Требование мозгового штурма — равноправие участников — смягчается, но учитывается при распределении ролей

Схема сценариев антагонистических игр

- Сессия разбивается на *сеансы*, для которых фиксируется обсуждаемый вариант и распределение ролей между участниками
- Сеанс состоит из *шагов*
  - *серия выступлений* сторонников и противников
  - *фиксация вариантов* наблюдателями с выступлениями или без них.
  - *обмен ролями* сторонников и противников варианта (наблюдатель может присоединиться к одной из групп)
  - *решение о новом сеансе или о завершении сессии*
- Завершение сессии
  - Подведение итогов сессии:
    - формирование списков доводов «за» и «против» и их упорядочивание (возможно приписывание весов для доводов)
    - Определение предпочтительного варианта (разные способы)
    - Уточнение предпочтительного варианта
  - Если предпочтительный вариант принимается как *решение*, то
    - формулировка решения и окончание обсуждения
  - иначе доводов оказалось недостаточно и нужна дополнительная проработка:
    - постановление о продолжении обсуждения: назначение (а) новой сессии игры, (б) мозгового штурма или (в) их комбинации
    - постановление о дополнительном изучении ситуации

Принципы контактных мероприятий

- **Общие принципы**:
  - *целенаправленность*;
  - *планирование* (подчиняется цели контакта);
  - *упорядоченность*;
  - *краткость*;
  - *убедительность*;
  - *результативность*;
  - *отношение к традициям и стереотипам*
  - *следование этическим нормам поведения*.

Любая деловая встреча, иной контакт должны

Эффективность контакта повышается, если перед ним инициатор разработает детальный план

Следование составленному плану и выполнение

Никогда не следует уделять внимание вопросам

Протоколирование и оценка результатов
Запись хода встречи, обсуждения и даже беседы

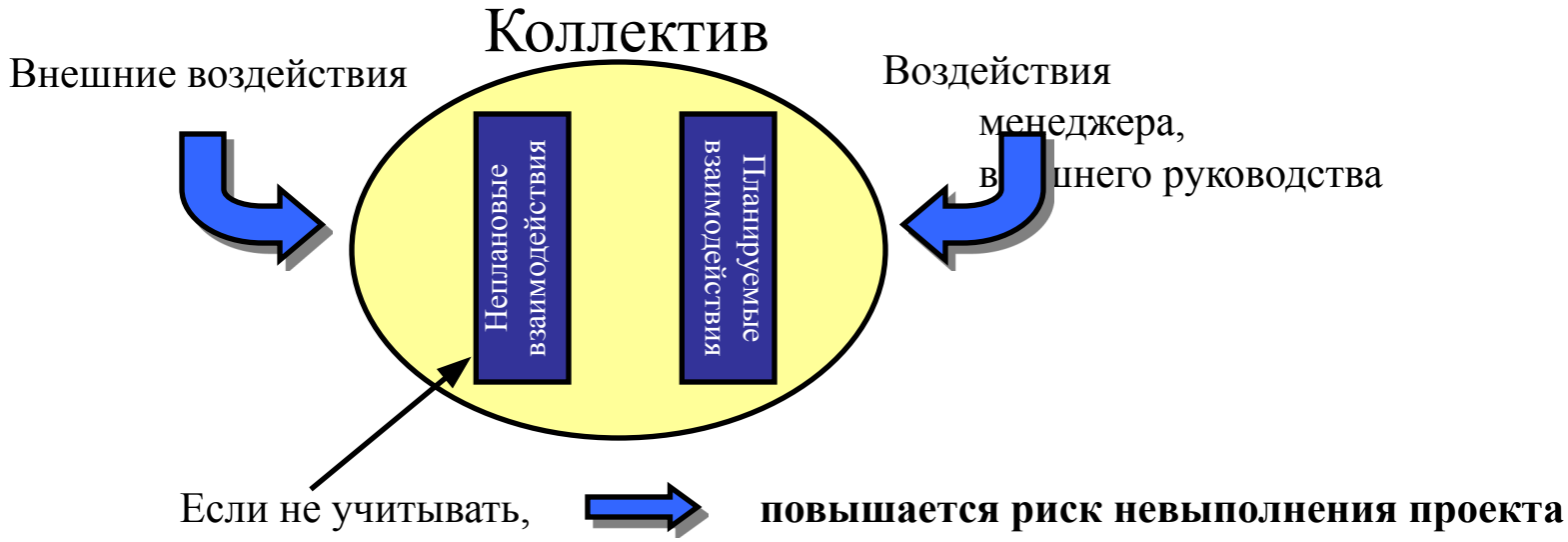
Не следует ломать традиции и стереотипы как коллективного, так и индивидуального

Создает благотворный микроклимат в коллективе, который способствует результативности ведения как данного проекта, так и будущих проектов коллектива

- **Специальные принципы** обусловлены особенностями
  - этих мероприятий,
  - коллектива,
  - конкретной методологии разработки проекта.
- Целесообразно до начала проекта зафиксировать процедуру проведения специальных мероприятий (лучше в документальной форме). Это избавит от бессодержательных обсуждений того, как должны проходить мероприятия. По крайней мере, это будет способствовать минимизации таких контактов

Неплановые взаимоотношения в коллективе

Далеко не все контакты в группе исполнителей проекта сводятся к отношениям, связанным с проектом



Неплановые взаимодействия

для нового коллектива → стихийно формируются

для устоявшихся групп → сглаживаются за счет учета индивидуальности работников

Лидер ответственен за

минимизацию нежелательных и

поддержку полезных неплановых взаимоотношений

Типичные виды взаимоотношений

- **Совместимость работников** (образуются *неформальные группы*);
- **Стихийное разделение труда** (стоит обращать на это внимание и использовать);
- **Отношение соперничества** (может быть позитивным или деструктивным);
- **Критическая позиция работника** (плохо, если она критиканская!);
- **Руководящая позиция работника** (имеется ввиду стихийное формирование позиции, а не прямое назначение);
- **Исполнительская позиция работника** (он может входить в работоспособную неформальную группу);
- **Обучающая позиция работника** (охотно разбирается в новых вопросах и полученные знания и умения передает окружающим

  Но **поучающая** позиция — источник конфликтов!)
- **Катализатор проектных работ** (материализуемой пользы от него не видно /по разным причинам/, но его участие в работах способствует эффективности работы других).
- **«Серый кардинал»** (старается вести «подпольную» работу. Плохо, когда он не распознан, но может быть хорошо, когда его удается использовать «по назначению»)

Это характеристики взаимоотношений в коллективе, а не индивидуальные качества работника

Совместные непроектные мероприятия

Это совместный отдых, походы на обед (разовые и постоянные), экскурсии и пр.

- Для менеджера не бывает взаимоотношений, несвязанных с выполнением проекта
- Совместные непроектные мероприятия говорят наблюдательному менеджеру:
  - кто с кем находится в контакте,
  - степень близости таких контактов,
  - кто может, а кто нет руководить,
  - кому комфортнее оставаться в тени и др.
  - как влияет на поведение коллектива присутствие руководителя: если оно незначительно, контакт менеджера и команды установлен, иначе — это симптом возможного нарушения желательных взаимоотношений.

Это уместно использовать для кадровых назначений и при распределении работ.

Критерий присутствия полезен для выяснения неформального места в коллективе не только менеджера, но и других работников

- Информация из непроизводственных контактов, является лишь дополнительной и вспомогательной и не должна подменять сведений о квалификации, опыте и других явных факторов, прямо влияющих на эффективность использования работника на том или ином месте.

15. Конфликты в проектном коллективе

Конфликты с позиций коллектива в целом

Потенциальная причина почти всех конфликтов, не связанных с неуправляемыми внешними возмущениями, — **ошибки в расстановке кадров**

Случай соперничества — не исключение:

Нужно определить цели и критерии, благодаря которым соперники будут стремиться к объективно более качественным решениям, а не к удовлетворению своих амбиций. Это возможно, *если*

- соперники потенциально совместимы (возможно, что именно на почве соперничества!),
- когда успех проекта для них значит больше, чем победа в личных взаимоотношениях.

Если есть риск, что это не так, спокойнее дать соперникам непересекающиеся области деятельности. Крайняя мера, на которую можно пойти, когда другие приемы не срабатывают, — отстранение деструктивно действующих работников от проекта.

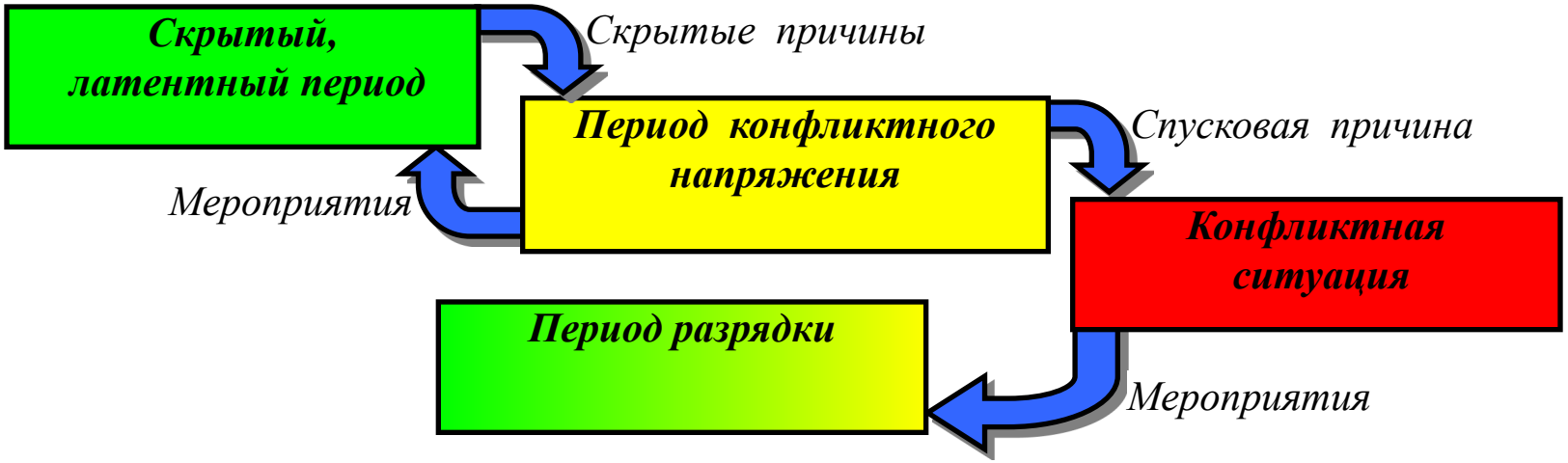
Как внутренние, так внешне возникающие конфликты компенсируются мерами, повышающими устойчивость команды к ним (чуть позже).

Конфликт с позиций рассмотрения отдельного работника

Здесь такие причины конфликтов:

- *Несоответствие притязаний* работника положению, которое он занимает в коллективе:
  - *Завышенная самооценка* — провоцирует необъективно низкую оценку результатов, полученных коллегами
  - *Заниженная самооценка* — чревата стрессом, когда работнику кажется, что он не справляется или не справится с заданиями (это уже сама по себе не способствует успеху)
- *Завышенные ожидания* от проекта (обычное следствие предыдущего) — работник рассчитывает на более высокие доходы, продвижение по службе и т. п., чем он того заслуживает
- *Несоответствие занимаемого положения* тому, которое он объективно заслуживает — положение работника оказывается ниже или выше.
  - Если это заметно коллективу $\Rightarrow$ вполне обоснованные претензиями к менеджеру: нерациональное использование кадров + сомнения в квалификации
    - Ниже: дискомфорт, другие примеряют ситуацию к себе
    - Выше: работу вынужден делать кто-то другой, это прецедент для других: можно занимать положение не в соответствии с деловыми качествами

Последовательное развитие конфликта



- Эти периоды можно выделить в развитии любого конфликта
- Относительная длительность их и общее время конфликта различаются
- Самый существенный по времени и влиянию на конфликт — латентный период, ⇒ объект наиболее пристального внимания менеджера до того, как сами работники поймут о возможности конфликта
  чтобы не допустить разрастания его до конфликтной ситуации.

Конфликты — это один из типов рисковых ситуаций проекта и к ним применимы стандартные приемы управления рисками:

1. Исключение риска
2. Уменьшения риска
3. Предупреждение ущерба
4. Стратегия действий при проявлении риска

Управление риском конфликтов

1. ***Исключение риска***. Применительно к конфликтам это означает минимизацию возможных и скрытых их причин.
   - взаимная совместимость менеджера и ключевых работников;
   - комфортные условия для ключевых работников (перспективы);
   - возможность для каждого ключевого работника привлечения к проекту тех, с кем он имел опыт совместной работы;
   - признание необходимости единодушия приема на работу.
2. ***Уменьшение риска конфликтов***. Минимизация скрытых причин в латентный период и в период конфликтного напряжения + демпфирование внешних причин:
   - планирование дополнительных работ,
   - создание резервного фонда проекта,
   - частичное перераспределение финансовых средств через общественные фонды

   Внешние причины и претензии к менеджеру.
3. ***Предупреждение ущерба от конфликтов***.
   - Компенсировать причины.
   - Проведение мероприятий *до* появления конфликтной ситуации.
   - Выход из нее с минимальными потерями.
   - Не порождать новых скрытых причин конфликтов.
   - Стремление, чтобы во всех возможных проявлениях любого конфликта он не оказался неожиданностью для менеджера + целевые установки:
4. ***Стратегия действий в конфликтных ситуациях***.

   Общее для всех стратегий — стремление, чтобы во всех возможных проявлениях любого конфликта он не оказался неожиданностью для менеджера и для коллектива.

   Целевые установки:
   - Минимизация конфликтной ситуации

     $\Rightarrow$ действовать нужно как можно раньше
   - сохранение работоспособности коллектива.

Конфликты с заказчиком

Особое положение таких конфликтов:

- судьба проекта в большей степени зависит от взаимоотношений с заказчиком, чем от других факторов;
- заказчик всегда прав, даже если допускает очевидные ошибки, несправедливости;
- не межличностный характер конфликтов между заказчиком и исполнителями

В конфликте с заказчиком вторая сторона — это менеджер проекта

Категории возможных конфликтов с заказчиком:

- *тупики согласования* уточнений постановок задач, принимаемых решений, используемых методов и др.;
- *нарушение плановых сроков* этапа (проекта, итерации, релиза, а также поставок оборудования и/или программ исполнителями);
- *претензии к качеству продукции* со стороны заказчика. Несоответствие ранее зафиксированным требованиям или дополнительные требования;
- *несогласованные нарушения заказчиком* графика финансирования или поставок, несвоевременное предоставление исполнителям нужных сведений и т.д. (сюда попадает и немотивированный отказ от проекта);
- *изменение заказа* в ходе выполнения проекта может быть конфликтным;
- *претензии исполнителей к заказчику*.

16. Планирование и контроль развития проекта

Планирование и оценка проекта

План — основа для оценки разработки с точки зрения соответствия ему полученных результатов.

Но есть и другие аспекты оценки:

- Оценка продукта безотносительно его производства (его эффективности с точки зрения автоматизации деятельности);
- Оценка побочных продуктов производства;
- Оценка удовлетворения требованиям (первичным, поступающим в ходе выполнения проекта и после начала использования);
- Оценка удовлетворения пользовательской потребности;
- Оценка соответствия спросу;
- Оценка соответствия рыночным потребностям;
- Оценка качества.

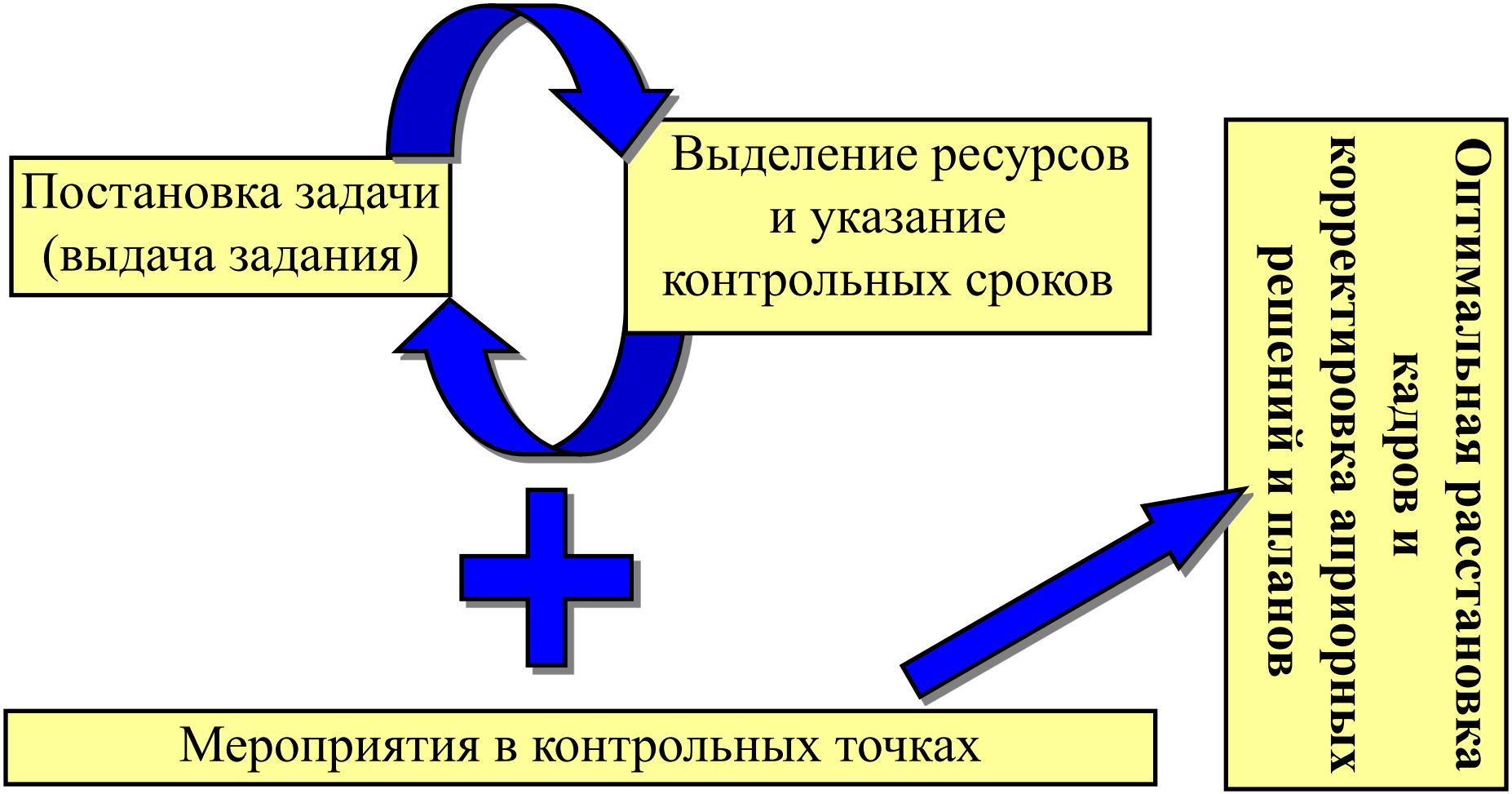
Когда, как и для чего нужно определять эти оценки?

Оценка процесса разработки и ее планирования

Есть еще оценка качества планирования как производственной функции и процесса разработки

- Оценка соответствия графику запланированных работ;
- Оценка проводимых мероприятий;
- Оценка коллектива:
  - квалификация работников,
  - рост квалификации работников,
  - стабильность кадров,
  - слаженность,
  - распределение обязанностей и разделение труда.
- Оценка реалистичности плана;
- Оценка выполнения каждого из видов плана

Цикл управления при разработке проекта



Подготовка к проекту

Цель — **априорное *распределение ресурсов*** (технических, кадровых, временных и финансовых) и ***обоснование проекта*** (соглашения о том, как должен развиваться проект)

Свод аргументов для

Уточненный заказ

варианты:

Концептуальная база проекта:

- концепции развития проекта,
- план релизов,
- стратегия минимизации рисков,
- стратегия управления качеством,
- методика тестирования,
- методика измерений и
- соглашение об отслеживаемых существенных связях.

Единый документ

Комплект документов

Ссылки на рабочие продукты

***Статические* и *изменяемые* (дополняемые) в дальнейшем документы**

Категории материалов проекта (уровни согласования)

- ***индивидуальные*** — материалы, с которыми имеет дело менеджер, для поддержки собственной деятельности;
- ***рабочие*** — материалы, которые готовятся для использования работниками коллектива, выполняющими проект;
- ***внутрифирменные*** — материалы, которые предъявляются руководству фирмы;
- ***официальные*** — материалы, требующие согласования на внешнем уровне (как с руководством фирмы, так и с заказчиком).

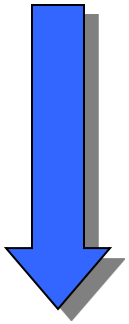
Необходимо знать:

Масштаб проекта и состав и объем материалов

Потребности ресурсов — *оценка*

+

концептуальная база проекта



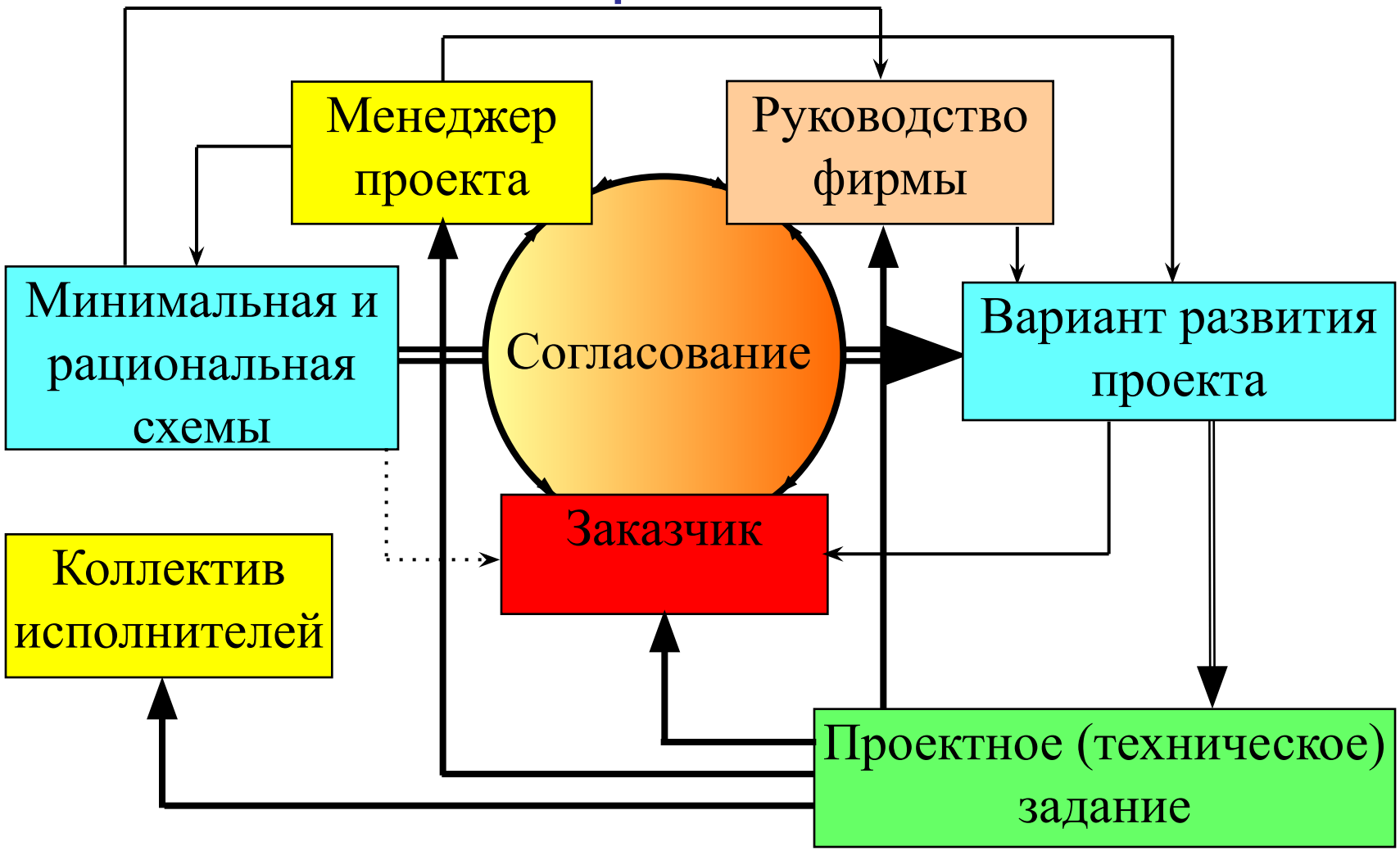
Две схемы распределения ресурсов:

Минимальная

Рациональная

Варианты развития проекта и проектное (техническое) задание

Выработка согласованного варианта развития проекта



Предъявление проектных материалов

Технические ресурсы проекта

- Поставляемое пользователям оборудование,
- Оборудование для разработчиков,
- Поставляемое внешнее ПО (а не разрабатываемое, т.е. стандартное),
- Используемое инструментальное ПО (также стандартное).

Как правило, потребность проекта во всех этих видах ресурсов вполне определена. *Однако есть моменты, на которые стоит обратить внимание*.

Какое оборудование и ПО должно поставляться, зависит от условий проекта (использование имеющегося, поставка или upgrade; возможность перенесения на проект стоимости инструментария разработчиков; вариантность решений).

График поставок оборудования и программного обеспечения

Время для:

- поиска и приобретения запрашиваемых ресурсов,
- установки и наладки,
- настройки у разработчиков с передачей потребителю.

Привязка поставок к результатам!

Это нужно как для отстаивания своей точки зрения на проект, так и для составления отчетного документа о распределении ресурсов.

(до начала проекта эта информация остается в распоряжении менеджера

Кадровые ресурсы проекта

1.	До начала выполнения проекта менеджер должен уяснить:	• кадровые потребности проекта и оценки их распределения по времени • возможности подбора кадров
2.	Информация о кадровой потребности проекта используется	• для составления графика привлечения сотрудников к проекту • для графиков минимально необходимой и рациональной схем
3.	Руководству предлагаются:	• графики и обобщающие характеристики (текущее состояние)
4.	Заказчику предлагаются (если он пожелает)	• число работников проекта • квалификационное распределение работников • распределение работников по срокам привлечения к проекту • другое
5.	Деятельность менеджера по заполнению вакансий	• приглашение кандидатов • определение лидера • уточнение текущих критических вакансий • прием на работу

Определение финансовых ресурсов

Определение финансовых потребностей	Оценка требуемых средств следует из оценки технических и кадровых ресурсов
Распределение предоставляемых финансовых средств	Оценка требуемых средств следует из оценки технических и кадровых ресурсов
Оценка вероятных доходов от разработки проекта	Прямые и косвенные доходы, расходы. Убыточные проекты

Заказанные продукты

Внутренние продукты

Переиспользование

Опыт разработчиков

Планируемая убыточность

Где и чем компенсируется

Схема формирования финансовых документов

- Менеджер составляет *основу документа*;
- Основа документа согласовывается с заинтересованными сотрудниками фирмы;
- Бухгалтер рассматривает основу и дополняет ее (расписывание средств по балансовым статьям прихода/расхода фирмы, указание дополнительных расходов);
- Результирующий документ
  - утверждается,
    - отвергается
      - отправляется на доработку к менеджеру (на этом шаге возможна передача документа другим лицам).

- Дорабатывается

…

Исполнение

Распределение предоставляемых финансовых средств

Основная ***задача*** — максимально быстрое получение результатов.

Календарный план, сетевой график и финансовые потребности проекта ⟹ ***смета проекта с привязкой затрат к срокам***.

1. При обнаружении в ней разного рода ошибок;
2. При нарушении общего баланса: суммарные расходы превышают суммарные плановые доходы;
3. При локальном (в какие-либо периоды) расхождении предоставляемых средств с объявленной в смете потребностью.

Если смета не утверждается

Дополнительные (сверхсметные) расходы — кто должен их нести? (фирма, заказчик, акционеры и др.)

Стратегии распределения времени

Сроки, фиксированные в заказе — это ***общий ресурс проекта***, предоставляемый в распоряжение менеджера для распределения.

Общепризнанные методики:

- составление календарных планов
- ведение графиков сетевого планирования (диаграммы Гантта и диаграммы зависимостей работ)

Календарный план

Достоинства	Недостатки
• Соответствие техническому заданию; • Дополнение новыми рубриками не вызывает трудностей; • Наглядность	• Тенденция к разрастанию • Плохой учет использования ресурсов • Рубрикация противоречит распараллеливанию работ, привязки работ и поставок к срокам • трудно увидеть все нужные показатели на определенный момент

Сетевое планирование

Два вида **графов зависимостей работ**:

(1). Вершины — работы, дуги — зависимости работ +

Атрибуты вершин, отражающие ***длительности*** работ

(2). Дуги — работы, вершины — зависимости между работами, +

Атрибуты дуг — ***длительности*** работ, требуемые ресурсы и др., атрибуты вершин отражают характеристики зависимостей

Диаграммы Гантта

Изображение графа зависимостей (1) в привязке к временной оси называется ***сетевым графиком выполнения работ*** в виде ***диаграмм Гантта***

Параметры, которые **нужно отражать** на сетевом графике:

— ***длительность работы*** — параметр, образующий график;

— ***минимальная ресурсная потребность***;

— ***максимально возможная ресурсная потребность***;

— ***минимально необходимое время*** выполнения работы.

Параметры, определяемые после построения графика:

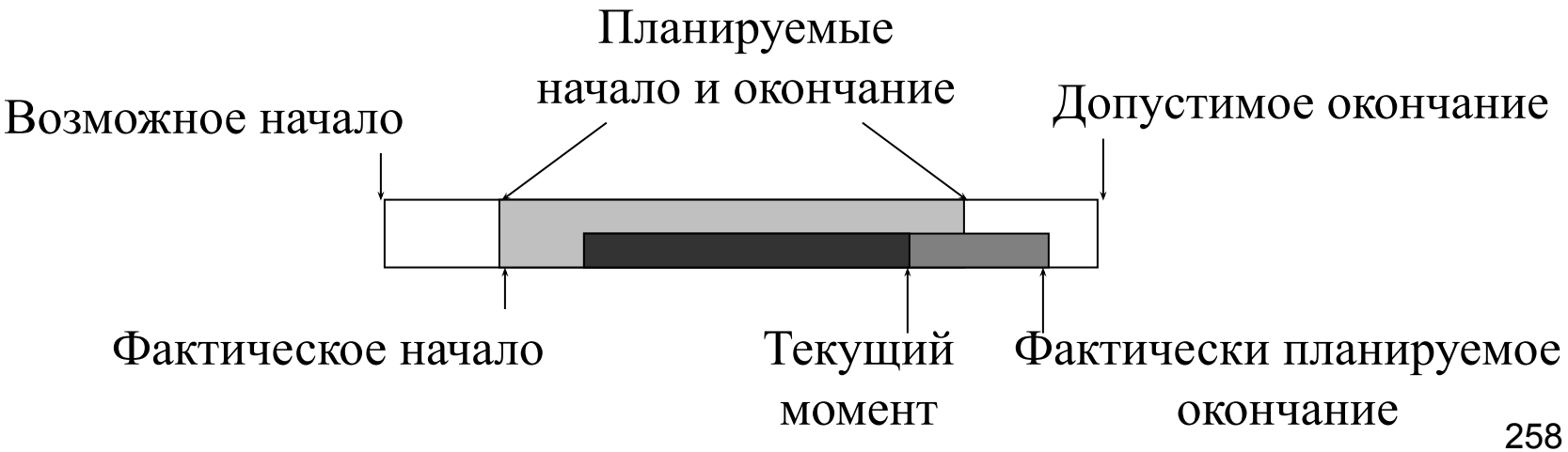
— ***время возможного начала работы*** — время, когда данная в работа принципе может начаться

— ***время допустимого конца работы*** — время, позднее которого данная работа не должна продолжаться

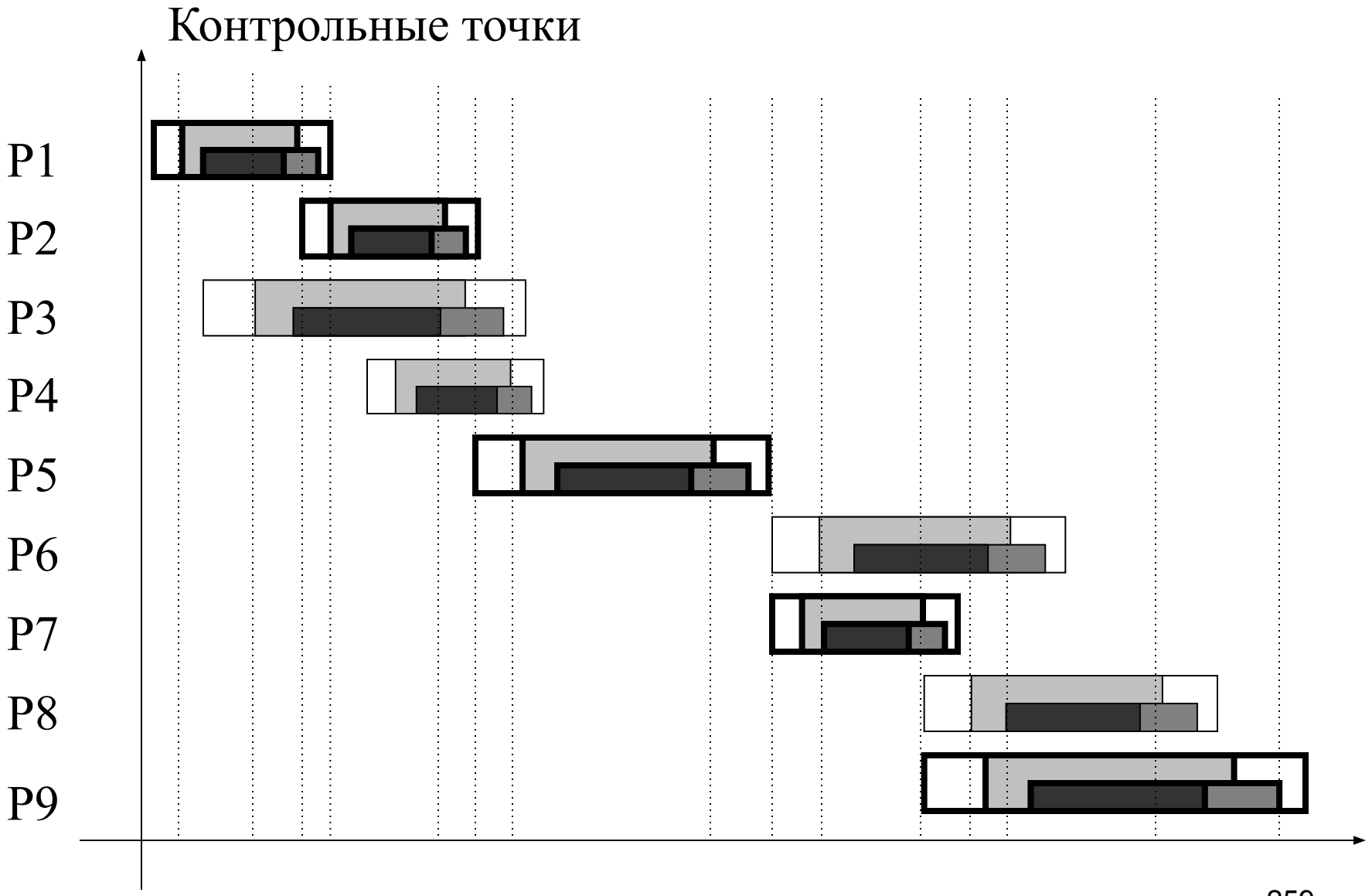
Что можно получать из диаграмм Гантта?

- Параметры, определяемые ходе выполнения проекта, которые указываются на графике:
  - ***время фактического начала работы***
  - ***время текущего планового завершения работы***
  - ***время фактического завершения работы***
- Параметры текущего момента:
  - текущая ***ресурсная обеспеченность*** (доля от максимума)
  - ***объемы работы*** — выполненный и оставшийся

Список параметров адаптируется к конкретным условиям проекта



Условный пример диаграммы Гантта



Общие положения сетевого планирования

- Сетевой план можно строить как для ***проекта в целом***, для ***отдельных этапов***, для ***групп исполнителей*** и ***отдельных исполнителей*** (большие проекты);
- Целесообразно варьировать ***уровень детализации работ*** и ***отслеживаемых параметров***, а также на отдельных операционных маршрутах. Большей детализации требуют текущий и ближайший следующий этапы, больше отслеживаемых параметров требуется для критического маршрута;
- Дуги графа зависимостей работ являются важной, но менее информативной частью сетевого графика по сравнению с выстраиваемой последовательностью работ. Гораздо важнее изображать ***временную вариантность выполняемых работ***;
- Явно на графике ***выделяется критический маршрут***, а также ***наиболее важные*** с точки зрения менеджера ***работы***;

Обычно ***наименования работ*** выносятся ***слева от графика*** и упорядочиваются в точном соответствии с календарным планом;

На графике ***явно отмечаются этапы выполнения проекта***

Сетевой план используется для текущего контроля выполнения проекта

- распределяются ресурсы (контрольная точка 1 жизненного цикла), строится **сетевой график верхнего уровня**;
- перед началом этапа в график вносятся **все работы этапа**;
- в ходе этапа отслеживаются временные характеристики работ, отмечается фактическое положение дел;
- если какая-либо **работа затягивается**, то выясняются **причины** этого и принимаются соответствующие **меры**;
- при досрочном выполнении работы оперативно перераспределяются высвобождающиеся ресурсы.

Своевременно корректировать график,

принимать меры (планировать их заранее)!

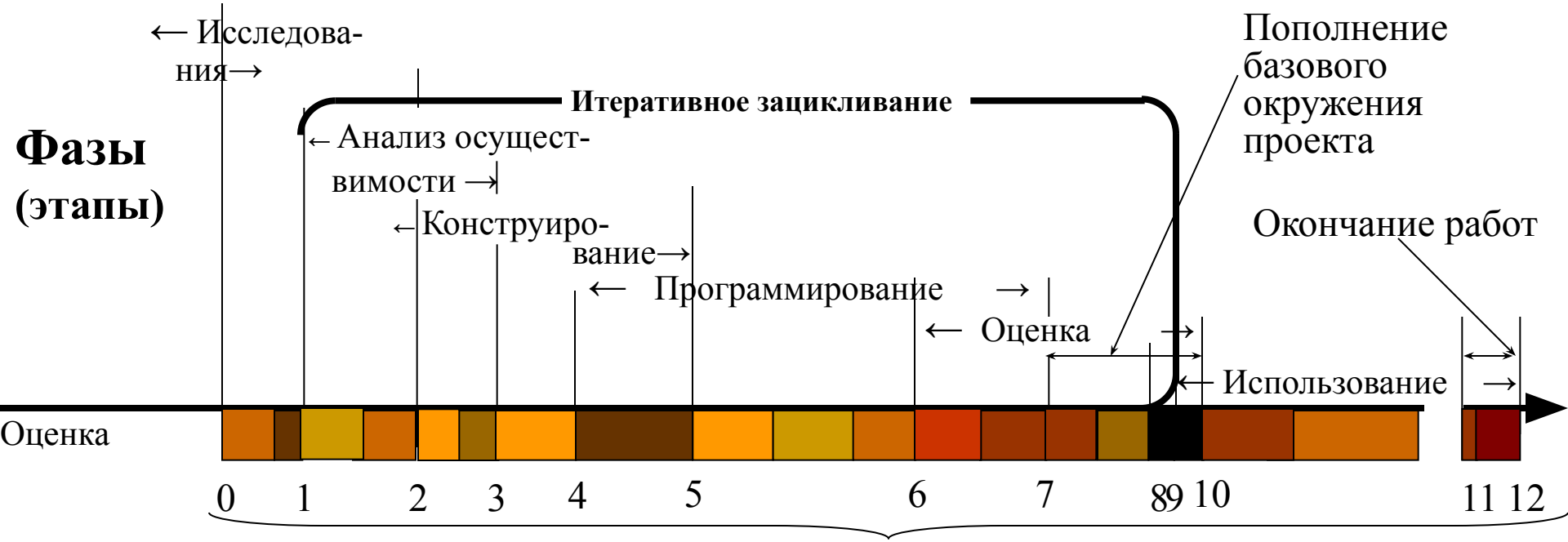
Ситуации, когда требуются дополнительные приемы работы с сетевым графиком

Что отслеживается: ***начатая работа замедляется или приостанавливается*** по причине того, что ***другая работа не позволяет*** ей выполняться

- работы расходуют общий ресурс, динамически распределяя его между собой;
- работы совместно используют общий ресурс;
- работы технологически объединены общим результатом (в частности, при конвейерном взаимодействии).

Приемы: ***укрупненное рассмотрение работ;*** изображение связанных работ как ***параллельно исполняемых*** + ***специальные обозначения***

Оценка как технологическая функция



Контрольные точки (события)

Стоимостная оценка продуктов

- соответствие спросу;
- соответствие потребности;
- своевременность;
- качество;

- трудозатраты;
- затраты на ресурсы;

соотнесенные к:

- тиражу;
- времени жизни;

Зависимость от масштабов проекта

Нормирование:
(а) объем кода;
(б) инвестиции;
(в) число исполнителей;
(г) другие характеристики

Цель: установить аналогию для оценки

Измерения

- *показатели размера* — характеристики, отражающие объемные параметры проекта, его сложность, а также количественные данные об исполнителях, занятых в проекте

К объемным показателям относятся: размеры отлаженного кода и средний размер модулей, классов и т.п., число модулей и классов.

К показателям сложности относятся различные отношения: числа циклических и разветвляющих конструкций, числа описаний к числу операторов. К ним относят глубину и разветвленность иерархии классов средний размер кода;

- *показатели продуктивности* — характеристики, отражающие производительность труда по категориям работников и индивидуальную производительность труда
- *показатели качества;*
- *показатели переиспользования.*

Назначение измерений в том, чтобы сопоставить их с априорными (экспертными) значениями, нежели для оценки труда коллектива.

Измерения: принципы

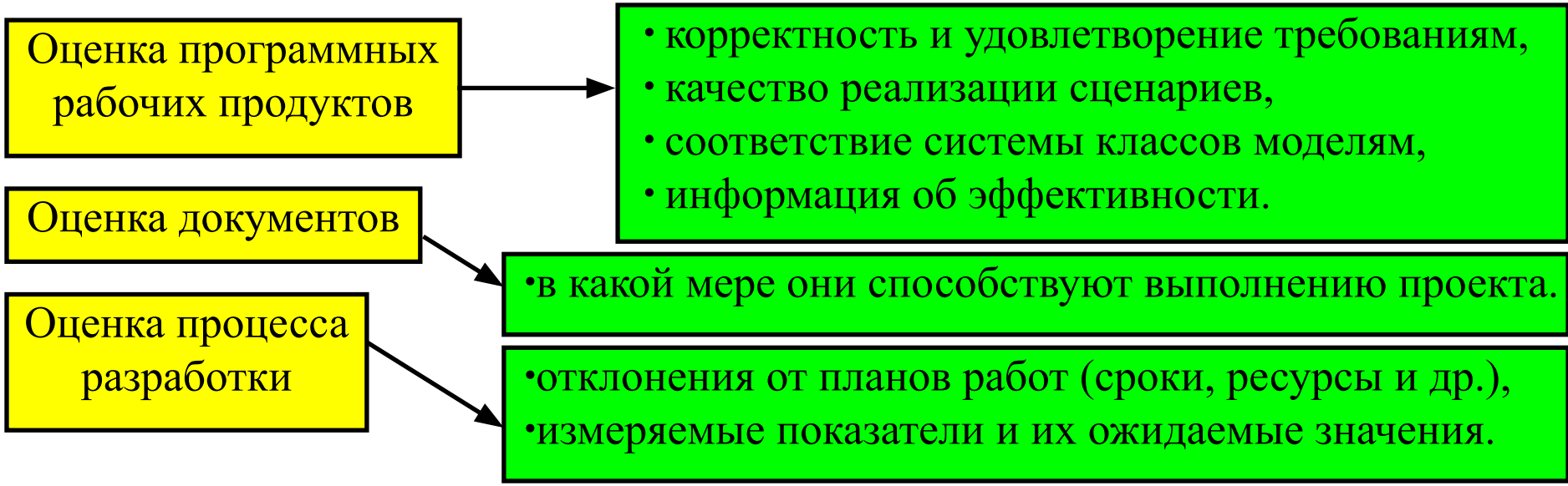
Это не самоцель —> ПРОСТОТА!!!

Нельзя смешивать измерения и оценку —> выводы прежде всего для себя, затем мероприятия, а НЕ ФОРМАЛЬНЫЙ ПОДХОД!!!

Любые формализованные характеристики, применяемые для оценки текущей работы, в конечном итоге могут подменить цели разработчиков: вместо достижения реальных результатов они станут стремиться к получению наиболее выгодных значений формальных показателей.

Эта ошибка типична при выполнении больших проектов, когда проявляется тенденция превращения разумных характеристик в формальные показатели оценки труда индивидуальных работников и групп.

Оценка хода выполнения проекта: оценка итерации



Основные работы

Испытание системы и анализ использования. Результаты проверки с точки зрения предстоящих работ. Определяется порядок корректировки ошибок, ситуации использования и сценарии, планируемые для следующих итераций;

Проведение экспертизы. Собираются сведения об использовании рабочих продуктов, полученных на данной итерации и о процессе выполнения итерации;

Измерения. Определяются запланированные для измерения показатели размера, продуктивности, качества и переиспользуемости;

Корректировка рабочих продуктов. Исправляются ошибки программ, модифицируются планы и другие документальные рабочие продукты;

Организация переиспользования;

Реорганизационные мероприятия. Исправляются ошибки в организации работ;

Обзор выполнения итерации. Подводятся итоги: фиксируются результаты и намечаются перспективы дальнейшего развития проекта;

Цель — в контексте проекта в целом. Оцениваются рабочие продукты и процесс разработки. Выясняется, как соответствуют полученные результаты целям итерации, как решается задача переиспользования.

Задача перераспределения ресурсов

1. Выделить работы, которые можно отложить, не нарушая сетевого графика проекта
2. Выделить работы, которые можно отложить, с нарушением графика и за счет этого сократить ресурсную потребность. Согласовать с заказчиком перенос сроков
3. Выделить замкнутые части операционных маршрутов ранжировать их по ресурсоемкость
4. Произвести переоценку значимости достигаемых целей и соответствующих им работ
5. Выделить максимально большую часть работ проекта, которая выполнима при заданном финансировании

Максимально полно обеспечить финансовую потребность решения ближайших задач проекта!

Оценка вероятных доходов от реализации проекта

Предположения о проекте + анализ ***ситуаций использования*** (оценка доходности вариантов применения).

Подсчет затрат на разработку и их ***разнесение*** на продукцию при тиражировании (какая стоимость продукта может обеспечить компенсацию затрат)

Разграничение проплат, которые

- обязуется сделать заказчик, и
- расходов, обеспечение которых берет на себя фирма.

В соответствии с этой пропорцией распределяется и будущий доход от реализации.

Соглашение о разделе продукции — часть договора между заказчиком и разработчиками