Программная инженерия

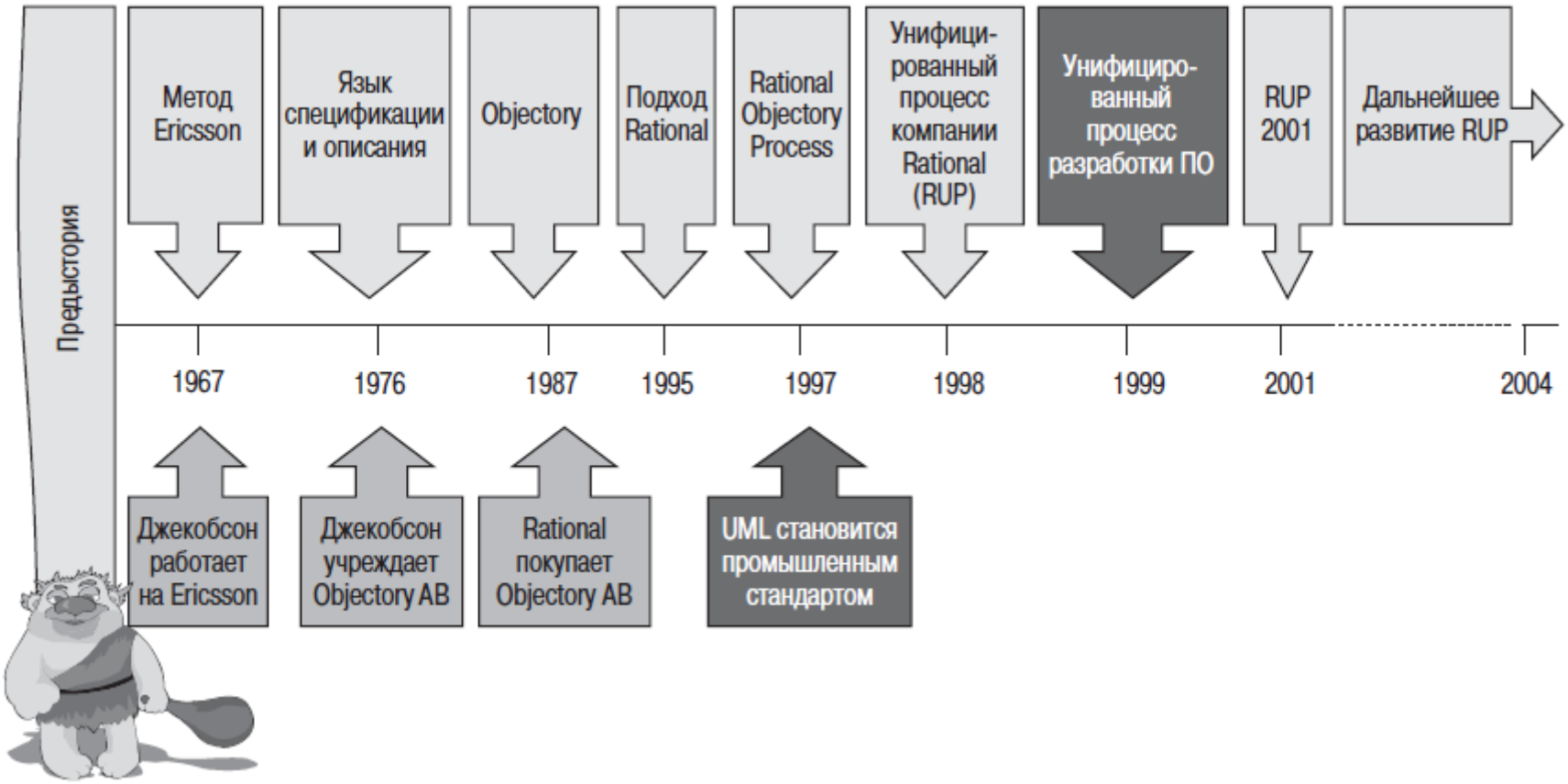
UP. Определение требований.

Унифицированный процесс (Unified Process, UP)

Унифицированный процесс

- Процесс производства программного обеспечения (Software Engineering Process, **SEP**), также известный как процесс разработки программного обеспечения (Software Development Process), определяет *кто, что, когда и как* в разработке ПО.
- **Унифицированный процесс разработки программного обеспечения** (Unified Software Development Process, USDP) – это SEP от авторов UML.

История UP


Предыстория
Метод Ericsson
Язык спецификации и описания
Objectory
Подход Rational
Rational Objectory Process
Унифици-рованный процесс компании Rational (RUP)
Унифициро-ванный процесс разработки ПО
RUP 2001
Дальнейшее развитие RUP
1967
1976
1987
1995
1997
1998
1999
2001
2004
Джекобсон работает на Ericsson
Джекобсон учреждает Objectory AB
Rational покупает Objectory AB
UML становится промышленным стандартом

UP и RUP

- Компания Rational развила подход Якобсона в «Унифицированный процесс компании Rational» (**Rational Unified Process, RUP**).
- UP и RUP очень тесно связаны. RUP – это коммерческий продукт, расширяющий UP
- Но RUP – это пропиретарный процесс, являющийся продуктом компании Rational, а UP – это открытый SEP от авторов UML.

Аксиомы UP

Унифицированный процесс является:

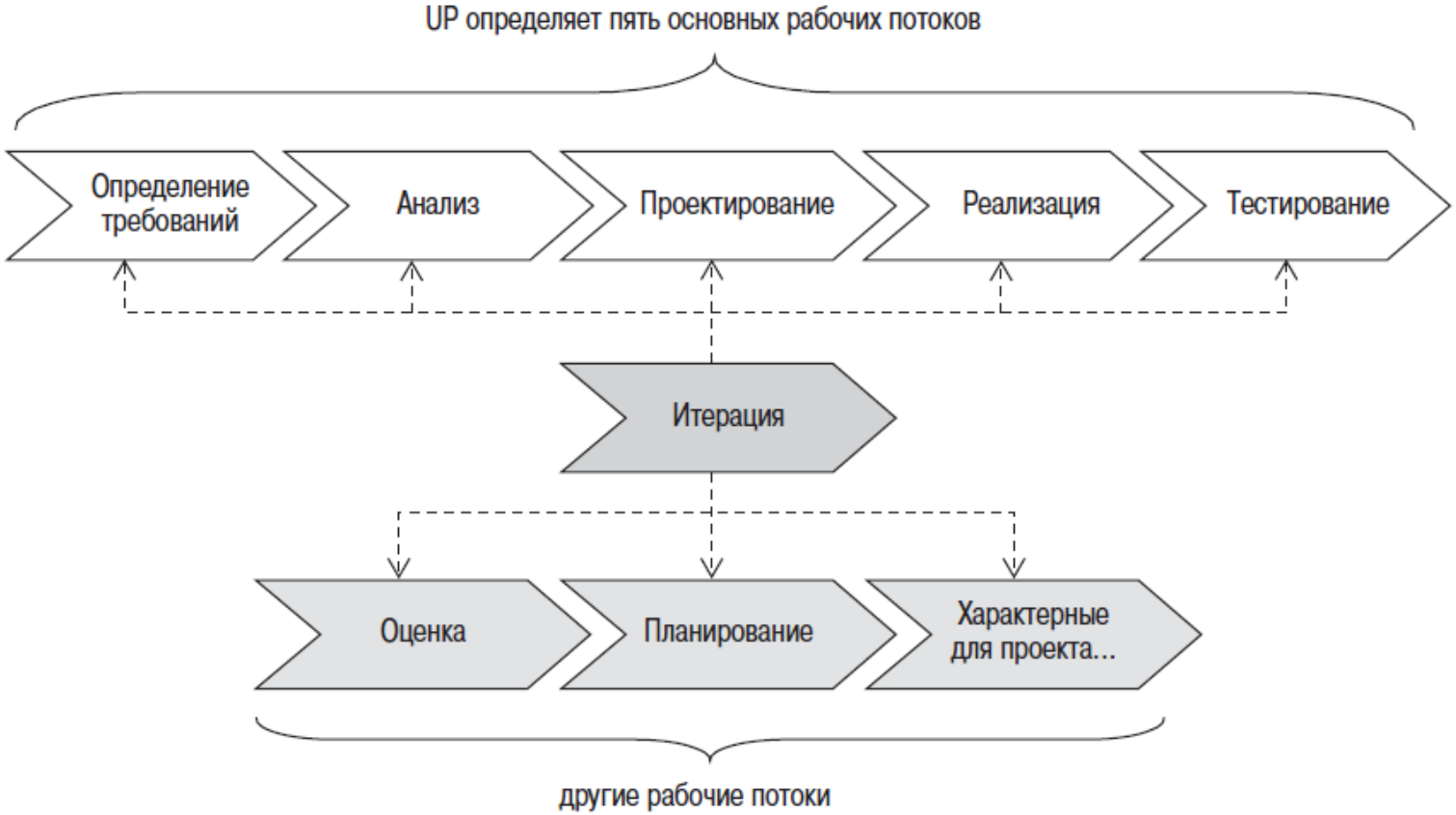
- **управляемым требованиями и риском**
  - требования и анализ возможных рисков закладывается в основу создания ПО
- **архитектуроцентричным**
  - подход UP заключается в создании надежной архитектуры системы
- **итеративным и инкрементным**
  - проект разбивается на небольшие подпроекты (итерации), которые обеспечивают функциональность системы по частям

Итерации UP

Каждая итерация включает ***все*** элементы обычного проекта по разработке ПО:

- **Планирование**
- **Анализ и проектирование**
- **Построение**
- **Интеграция и тестирование**
- **Версия для внутреннего или внешнего использования**

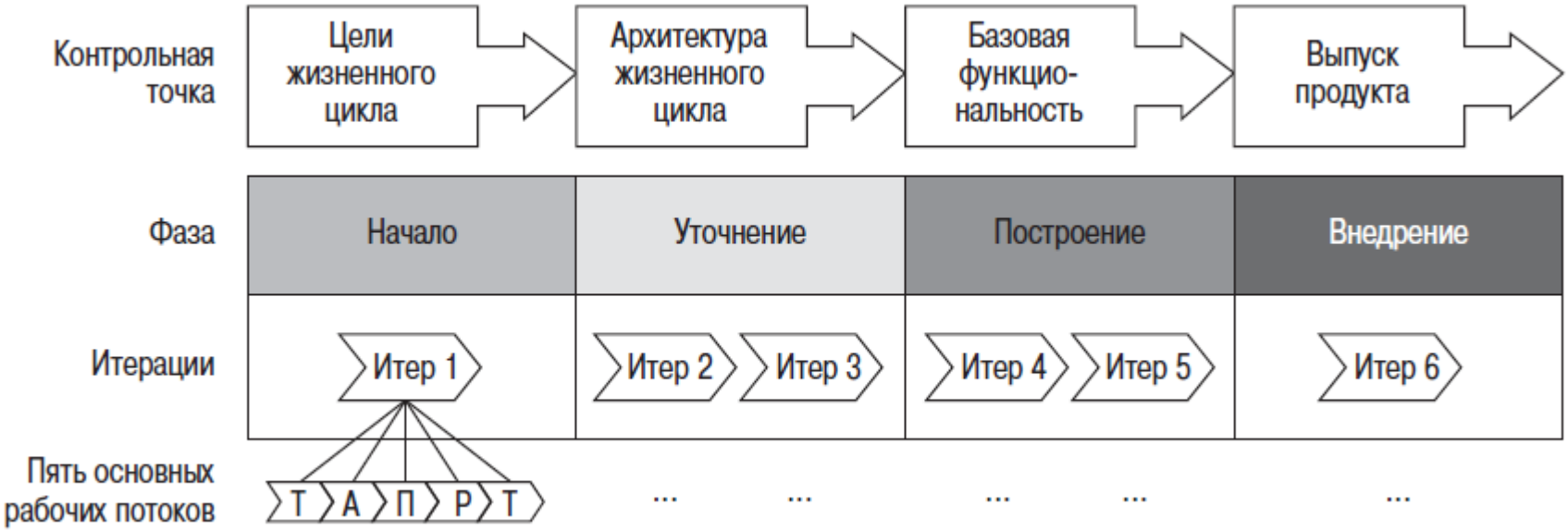
Основные рабочие потоки итерации



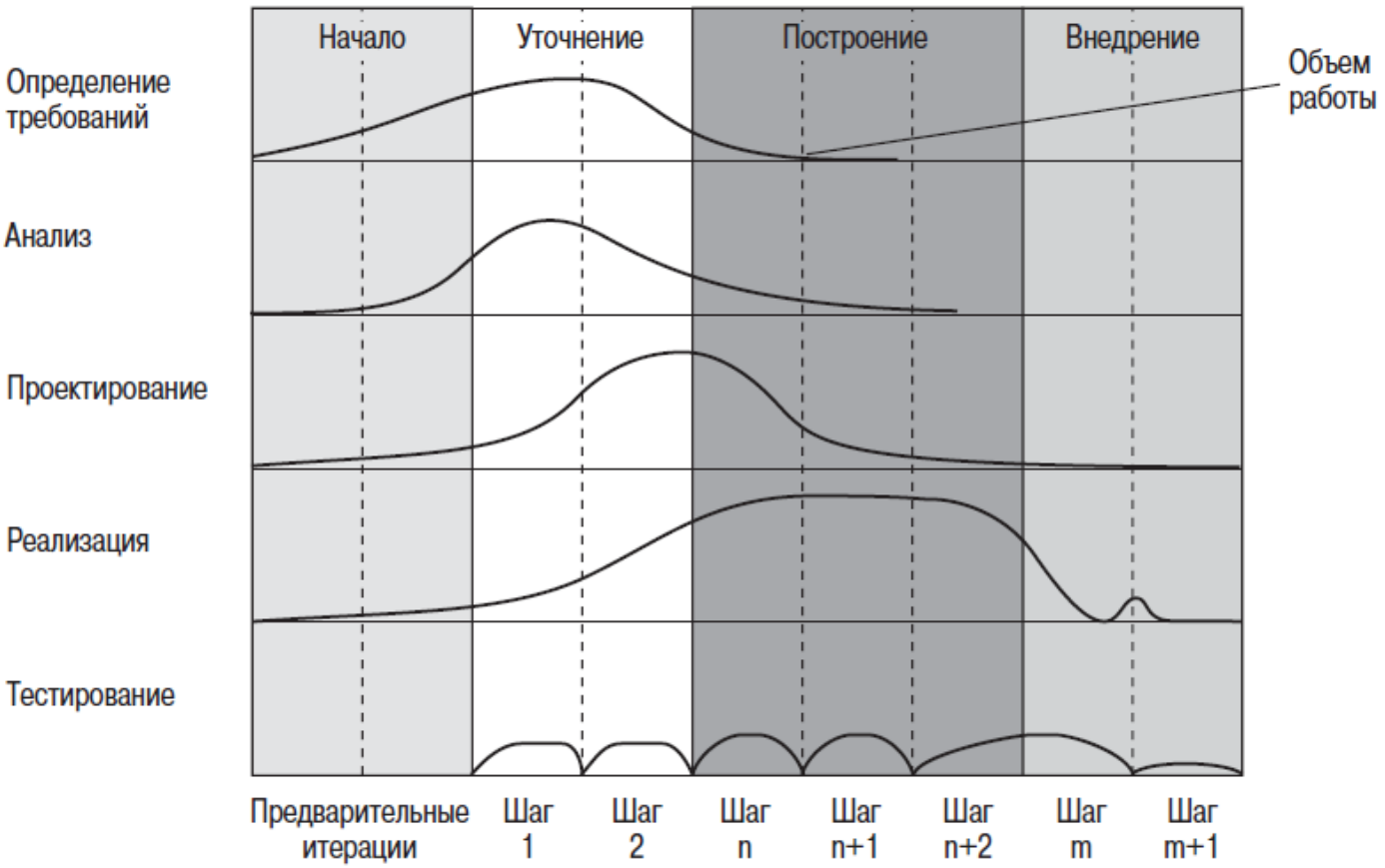
Фазы UP

- В UP четыре фазы, каждая из которых имеет свои контрольные точки. Каждая фаза может состоять из 1 или более итераций.
- **Начало (Inception)** – цели жизненного цикла;
- **Уточнение (Elaboration)** – архитектура жизненного цикла;
- **Построение (Construction)** – базовая функциональность;
- **Внедрение (Transition)** – выпуск продукта.

Структура UP



Распределение работ по фазам UP



Определение требований

Определение требований – основа проектирования системы

- Большая часть работы с требованиями проводится в начале проекта, в фазах Начало и Уточнение.
- До начала работы над ОО анализом и проектированием уже необходимо иметь представление о том, что должно получиться в результате.
- Выработка требований – это переговорный процесс, поскольку часто выдвигаются противоречивые требования, которые должны быть согласованы.

Требования и диаграмма прецедентов

- Некоторые курсы UML утверждают, что прецеденты UML являются единственным способом фиксирования требований.
  - Прецеденты могут отражать только функциональные требования, которые являются описанием того, *что будет делать система.*
- Но существуют требования которые не относятся к функциональности, но являются ограничениями, накладываемыми на систему (производительность, надежность)

Требование

- Требование – это подробное описание того, что должно быть реализовано.
  - Функциональные требования – какое поведение должна предоставлять система;
  - Нефункциональные требования – особое свойство или ограничение, накладываемое на систему

Требования указывают что должно быть построено, но не говорят как это сделать.

Традиционное описание требований

Примеры функциональных требований

- 1. Система ATM должна проверять действительность вставленной в банкомат карточки.
- 2. Система ATM должна проверять достоверность PIN-кода, введенного пользователем.
- 3. Система ATM должна выдавать по одной ATM-карточке не более $250 в сутки.

Примеры нефункциональных требований

- 1. Система ATM должна быть написана на C++.
- 2. Система ATM должна обмениваться информацией с банком, используя 256-разрядную кодировку.
- 3. Система ATM должна проверять действительность карточки ATM в течение не более трех секунд.
- 4. Система ATM должна проверять достоверность PIN-кода в течение не более трех секунд.

Атрибуты требований

Набор критериев важности «MoSCoW»

Значения атрибута Priority	Семантика
Must have (обязан иметь)	Обязательные требования, являющиеся фундаментальными для системы.
Should have (должен иметь)	Важные требования, которые могут быть опущены.
Could have (мог бы иметь)	По-настоящему необязательные требования (реализуются, если есть на это время).
Want to have (хотел бы иметь)	Требования, которые могут подождать до следующих версий системы.