Программная инженерия. Лекция 6
Введение в Agile и диаграммы

Михаил Пожидаев

20 марта 2024 г.

Agile

Гибкая методология разработки

Agile software development — это набор *принципов* и правил, в рамках которого осуществляется разработка ПО.

не единственный вариант

Agile — это семейство процессов, а не один конкретный подход.

Agile не описывает конкретных практик, а фиксирует ценности и принципы, которыми руководствуются команды.

Agile Manifesto

Ценности и принципы

Суть Agile методологии сформулирована в документе «Agile Manifesto». разработан и принят 11–13 февраля 2001 г. на лыжном курорте The Lodge at Snowbird в шт. Юта. Подписали представители следующих методологий:

- Extreme programming;
- Scrum;
- DSDM;
- Adaptive Software Development;
- Crystal Clear;
- Feature-Driven Development;
- Pragmatic Programming.

Короткие циклы

Снова итерации, куда без них?

Гибкие методологии сфокусированы на минимизации рисков путём сведения разработки к серии коротких циклов, называемых итерациями, ориентированными обычно на 1–2 недели. Итерация воспринимается как минимальный проект и включает все задачи, направленные на получение инкремента: планирование, анализ требований, проектирование, реализацию и тестирование.

Всегда готов!

Хотя отдельная итерация, как правило, недостаточна для выпуска новой версии продукта, подразумевается, что гибкий программный проект готов к выпуску в конце каждой итерации.

После окончания итерации производится ревизия целей разработки.

Ориентирование на небольшие команды

Взаимодействие

Методы Agile делают упор на живое общение участников разработки. Большинство команд расположены в одном офисе, иногда называемом bullpen.

Вместе с заказчиком!

Эту роль может выполнять менеджер проекта, бизнес-аналитик или клиент.

В офисе также присутствуют:

тестеры, дизайнеры интерфейсов, технические писатели и менеджеры.

Ценности

Результат — это работающий программный продукт.

Воспринимая работающий программный продукт как единственный показателя работы команды за период времени, сформулированы следующие ценности:

- личности и их взаимодействия важнее, чем процессы и инструменты;
- работающее программное обеспечение важнее, чем полная документация;
- сотрудничество с заказчиком важнее, чем контрактные обязательства;
- реакция на изменения важнее, чем следование плану.

Принципы

Первая половина принципов Agile

1. Удовлетворение клиента за счёт ранней и бесперебойной поставки ценного ПО.
2. Приветствие изменения требований, даже в конце разработки.
3. Частая поставка рабочего ПО (каждый месяц или неделю или ещё чаще).
4. Тесное, ежедневное общение заказчика с разработчиками.
5. Проектом занимаются мотивированные личности, которые обеспечены нужными условиями работы, поддержкой и доверием.
6. Рекомендуемый метод передачи информации — личный разговор.

Принципы (продолжение)

Вторая половина принципов Agile

1. Работающее ПО — лучший измеритель прогресса.
2. Спонсоры, разработчики и пользователи должны иметь возможность поддерживать постоянный темп на неопределенный срок.
3. Постоянное внимание на улучшение технического мастерства и удобный дизайн.
4. Простота — искусство НЕ делать лишней работы.
5. Лучшие архитектура, требования и дизайн получаются у самоорганизованной команды.
6. Постоянная адаптация к изменяющимся обстоятельствам.

Scrum

Пример Agile с конкретизации практик

Scrum («свалка» или «драка») делает акцент на качественном контроле процесса разработки. это набор принципов, на которых строится процесс разработки, позволяющий в фиксированные промежутки времени (спринты, 2–4 недели) предоставлять конечному пользователю работающее ПО с инкрементом.

1. Функционал в очередном спринте определяется до его начала на этапе планирования и не может далее изменяться.
2. Строго фиксированная небольшая длительность спринта придаёт процессу разработки предсказуемость и гибкость.

UML

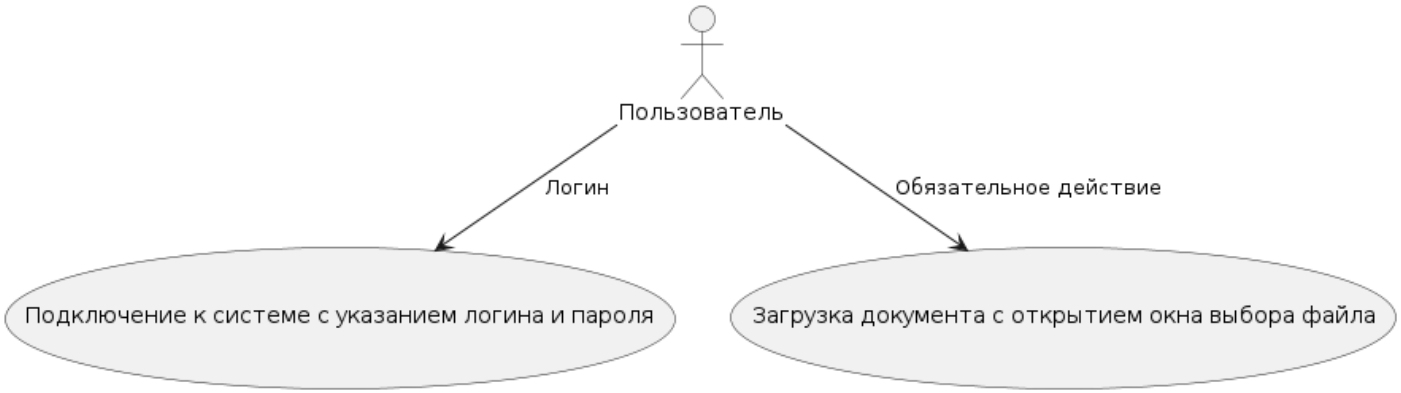
Unified Modelling Language

UML — система соглашений и графических нотаций, предназначенная для описания структурных, функциональных, поведенческих и других характеристик программной системы, а также для описания бизнес-процессов.

Ивар Якобсон

Варианты использования

Диаграмма вариантов использования UML



Варианты использования

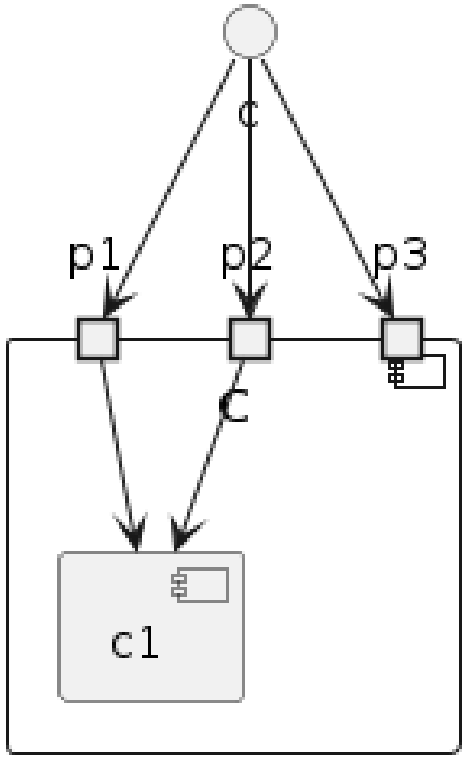
Диаграмма вариантов использования UML

диаграмма прецедентов — альтернативное название диаграммы вариантов использования со следующими свойствами:

- описывает принципиальные отношения между акторами и вариантами использования;
- определяет функциональный облик системы;
- вводит ясное разделение между периметром системы и её окружением.

Компоненты

Диаграмма компонентов UML



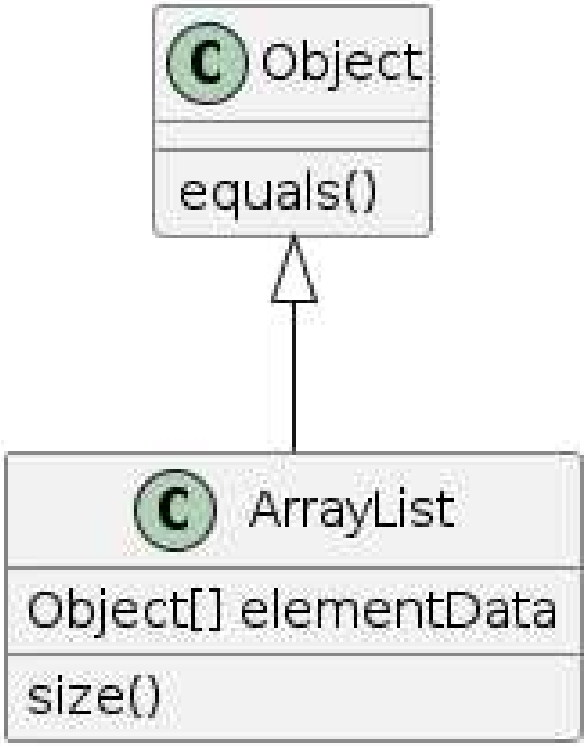
Компоненты

Диаграмма компонентов UML

Статическая структурная диаграмма со следующими свойствами:

- формируется структурный облик системы с акцентом на структурные блоки и заимосвязи между ними;
- компоненты имеют порты — точки взаимодействия с другими компонентами;
- структура компонентов описывается не только внешняя, но также изображается и внутренняя структура.

Диаграмма классов



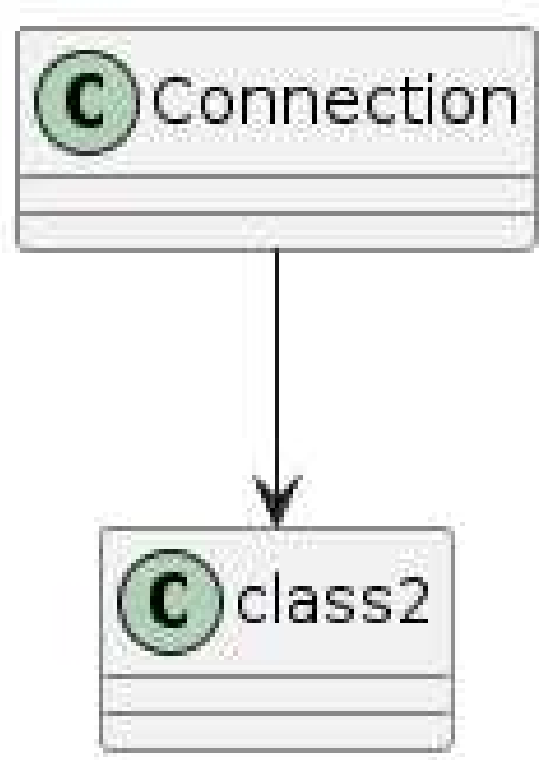
Классы

Отношения между классами в UML

1. Наследование: новый класс получает все свойства родительского.
2. Реализация: полное содержание неполного класса.
3. Композиция: неразделимые отношения между целым и частью.
4. Агрегация: разделимые отношения между целым и частью.
5. Ассоциация: ссылка на экземпляр другого класса.
6. Зависимость: отношения со влиянием изменений между классами.

Последовательность

Диаграмма последовательности UML



Последовательность

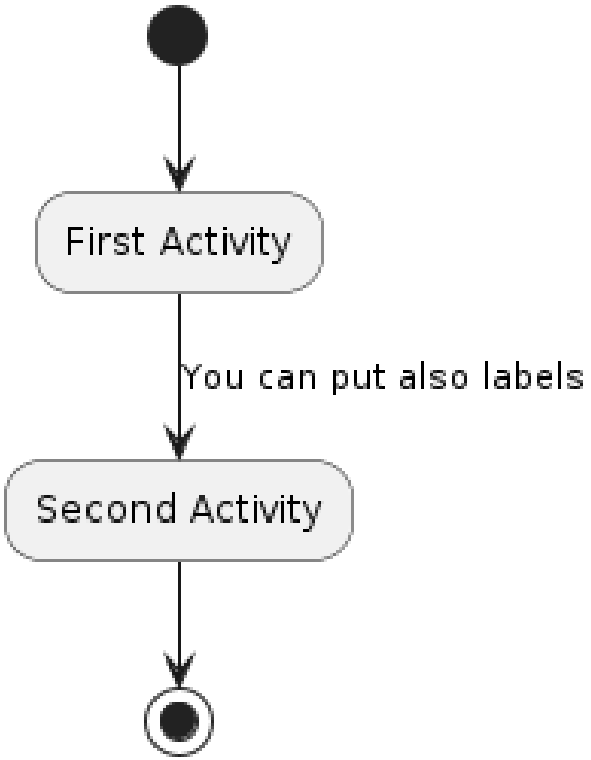
Диаграмма последовательности UML

Свойства диаграммы последовательности:

- изображается направленный обмен сообщениями или действиями между участниками процесса;
- все сообщения или действия изображаются на общей шкале времени;
- ориентирование на изображение жизненного цикла некоторого объекта;
- различаются синхронные и асинхронные сообщения.

Деятельность

Диаграмма деятельности UML



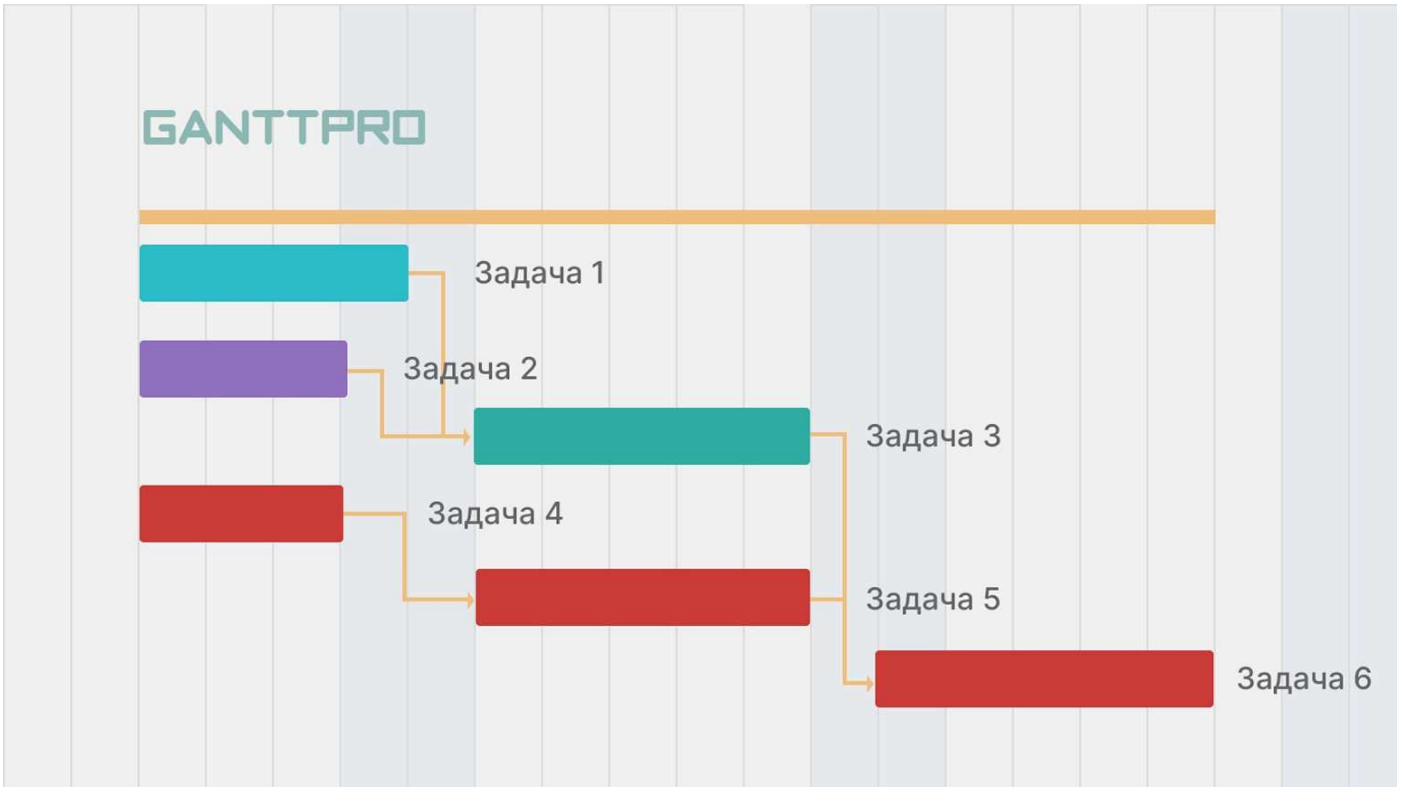
Деятельность

Диаграмма деятельности UML

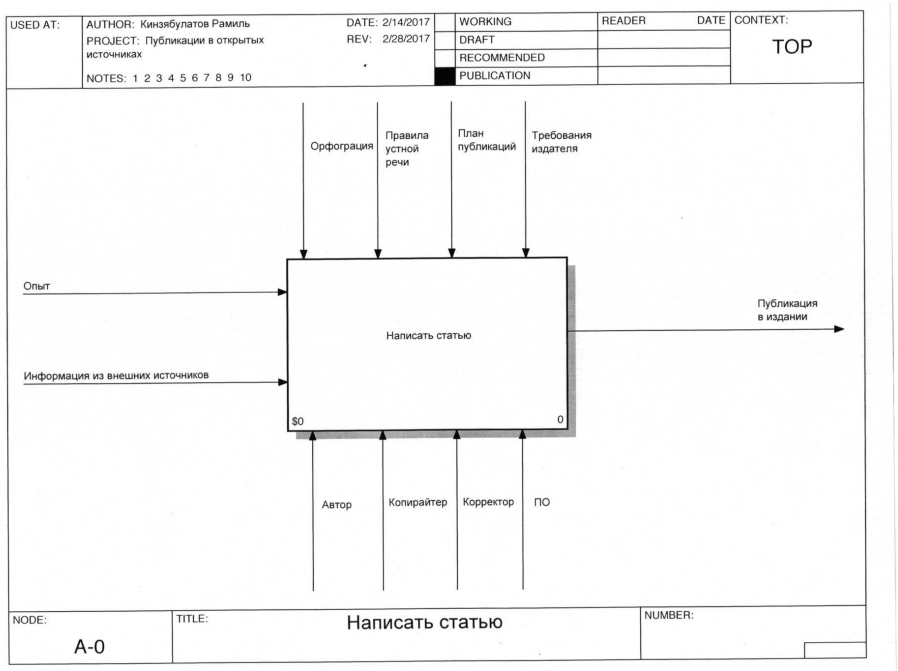
диаграммы деятельности используются преимущественно для моделирования бизнес-процессов со следующими свойствами:

- центральную роль играют действия и их последовательности;
- могут присутствовать узлы принятия решения, в которых происходит ветвление обработки;
- обязательно присутствие начального узла и одного или нескольких конечных узлов.

Диаграмма Ганта

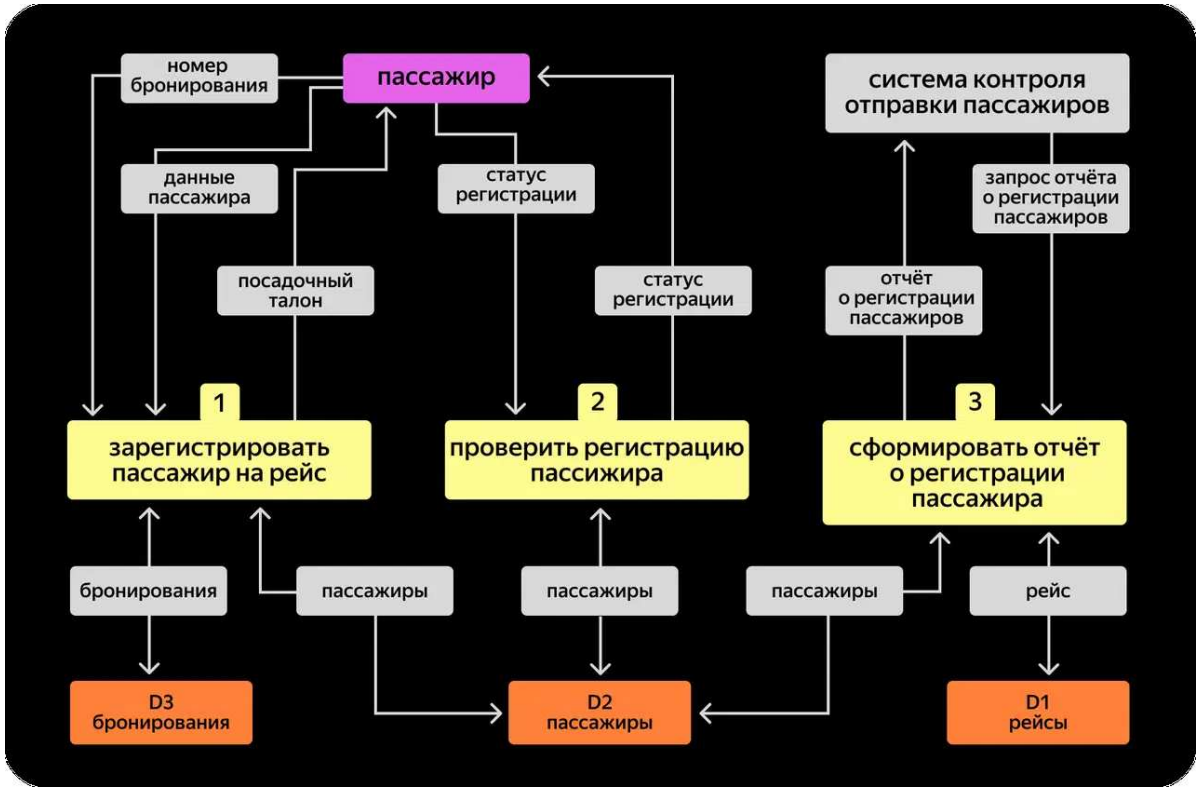


IDEF0


USED AT:
AUTHOR: Кинзябулатов Рамиль
DATE: 2/14/2017
PROJECT: Публикации в открытых источниках
REV: 2/28/2017
NOTES: 1 2 3 4 5 6 7 8 9 10
WORKING
DRAFT
RECOMMENDED
PUBLICATION
READER
DATE
CONTEXT:
TOP
Орфография
Правила устной речи
План публикаций
Требования издателя
Опыт
Информация из внешних источников
Написать статью
$0
0
Публикация в издании
Автор
Копирайтер
Корректор
ПО
NODE:
A-0
TITLE:
Написать статью
NUMBER:

DFD

data flow diagrams



Спасибо за внимание!

Всё о курсе: https://marigostra.ru/materials/engineering.html
E-mail: msp@luwrain.org
Канал в Телеграм: https://t.me/MarigostraRu