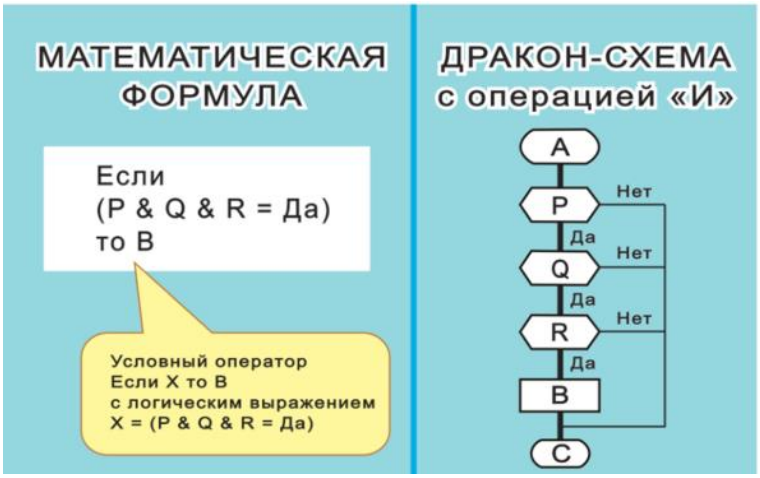
Графические формы записи алгоритмов.

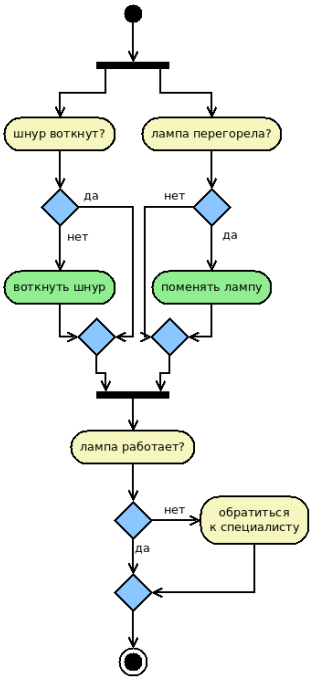
1. Дракон-Схемы.



Основные элементы: прямоугольник, ромб, “молния”, слияние, вопрос.
Особенности и основные моменты:

- Ясность и однозначность.
- Понятен не только программистам.
- Линейность: действия идут сверху → вниз.
- Уникальные обозначения, позволяющие описывать сложные действия.
- Используется в системах, где важна надежность и понятность.

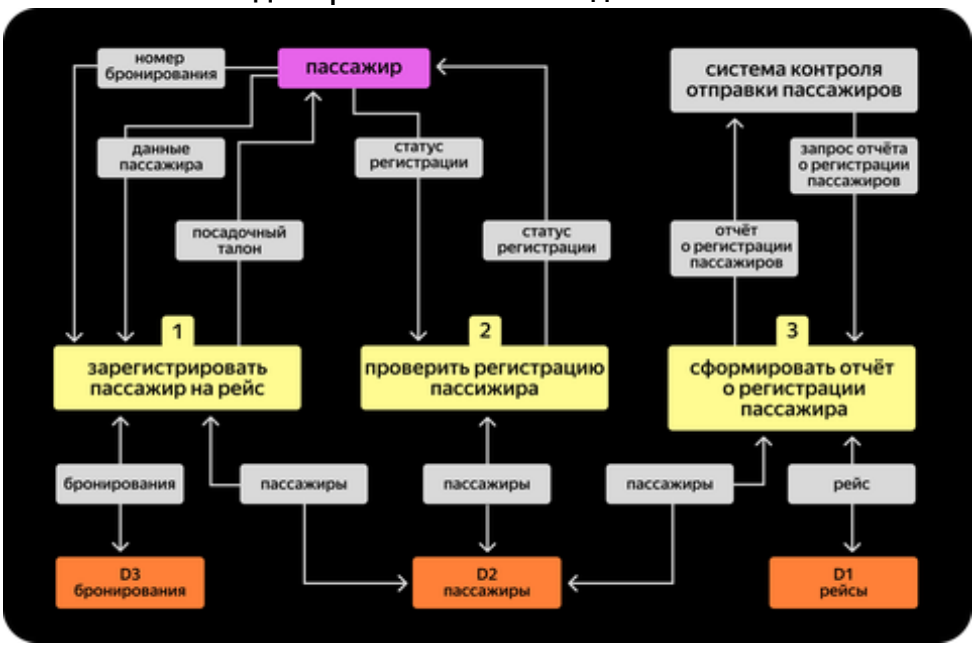
2. Диаграммы деятельности UML.



Основные элементы: Действия(узлы), Переходы (Стрелки), Ветвления(ромбы), Начало и конец (Черные круги), Параллельные секции(разделители).
Особенности и основные моменты:

- Подходит как для простых, так и для сложных алгоритмов.
- Позволяет выделить основные этапы и зависимости между ними.
- Можно моделировать одновременное выполнение задач.
- Легко совмещается с другими типами UML-диаграмм.
- Широко применяются в программной инженерии для документирования систем.

3. Диаграммы потоков данных.

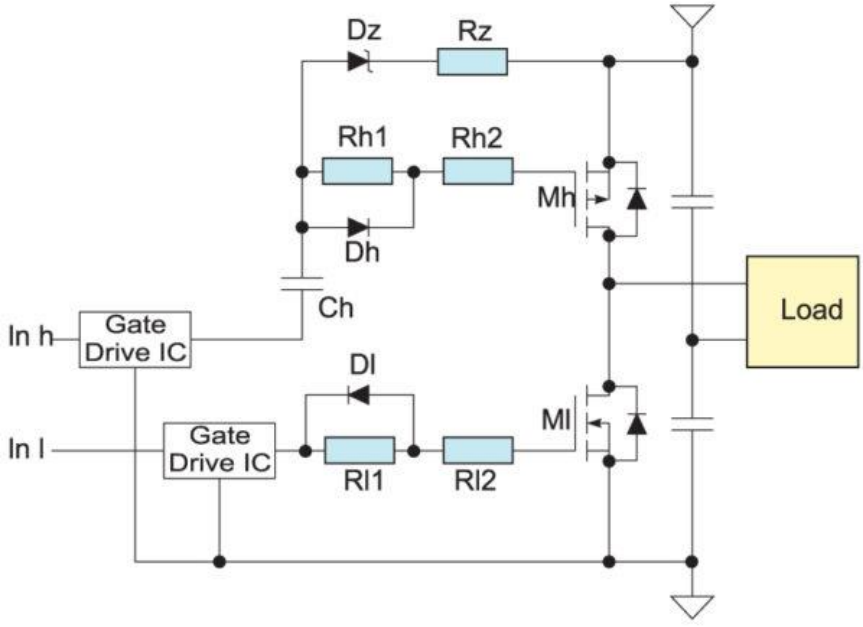


Основные элементы: процесс, поток данных, хранилище данных, внешняя сущность.

Особенности и основные моменты:

- Показывает какие данные используются и как они изменяются.
- Поддерживает декомпозицию процессов на более мелкие.
- Демонстрирует, как компоненты системы обмениваются данными.
- Описывает процессы, не привязываясь к конкретной платформе.
- Используется при проектировании баз данных и информационных систем.

4. Запись алгоритмов с помощью Р-схемы.

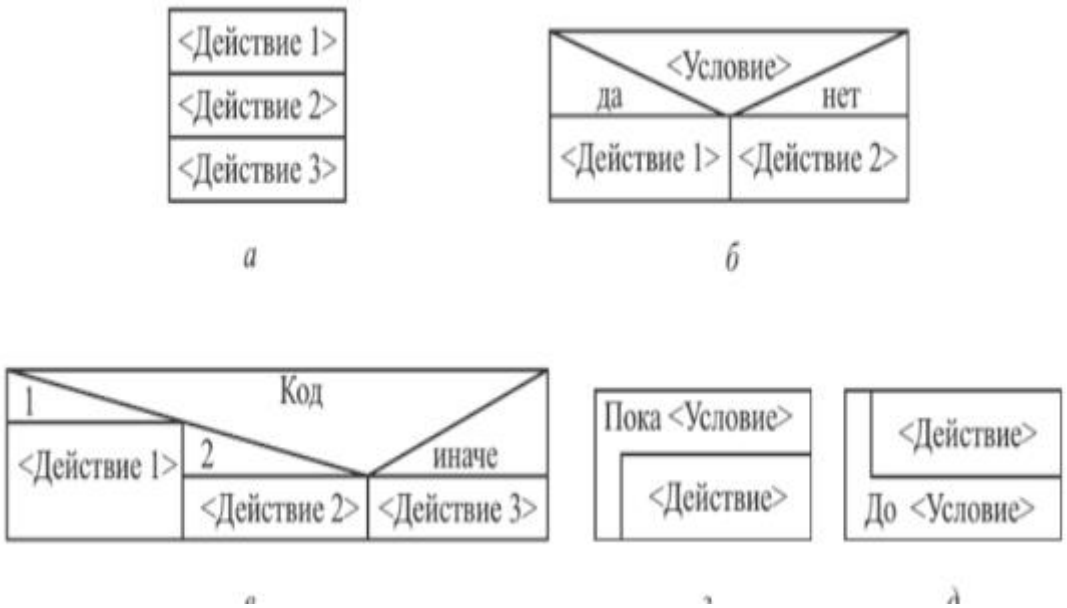


Основные элементы: Вершина, Вершина специальная, Дуга, Дуга специальная, Линия соединительная, комментарий.

Основные моменты и особенности:

- Четкая последовательность выполнения шагов.
- Используются для представления алгоритмов любой сложности.
- Отображают алгоритм в логике, аналогично коду.
- Алгоритм можно разбить на подзадачи и модули.

5. Диаграммы Нэсси-Шнейдермана.



Основные элементы: Блок действий, Блок выбора, Циклы.

Особенности и основные моменты:

- Основаны на концепции структурного программирования.
- Исключает хаотичные связи и упрощают восприятие.
- Отражают конструкции языков программирования.
- Для сложных алгоритмов диаграмма может быть слишком большой.
- Популярны в образовательных целях для объяснения базовых алгоритмов.