Drakon IDE
Среда для обучения алгоритмизации

Валерий Лаптев

Астраханский
государственный технический университет

Причины

1. *Необходимость* обучать алгоритмизации с нуля слабых студентов

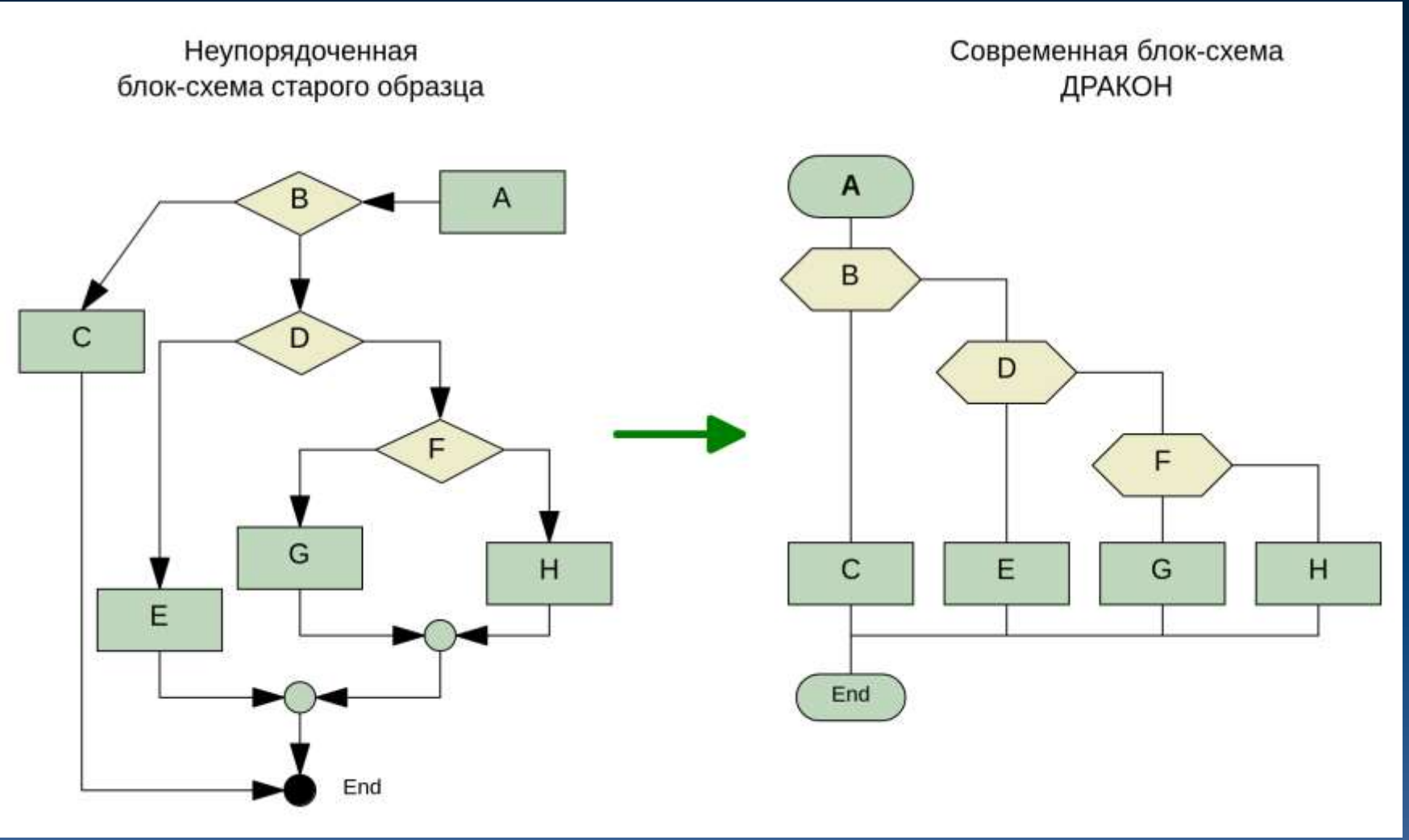
2. *Возможность* обучить сильных студентов реальной разработке

3. Существует *реальный заказчик*

ОКБ АМУР №3

АВТОМАТИЗАЦИЯ

Строгость языка ДРАКОН



СКБ АМУР

Пользователи - не программисты

Предпочитают графические схемы, а не код

Потребности - создавать *алгоритмы* для работы *микроконтроллеров*

Используемые системы

- CodeSys – мощная IDE

1. графические языки не позволяют наглядно представить алгоритм работы
2. алгоритмы – программа на языке программирования (код)

- Дракон-редактор (Тышов)
- **OWENLogic**

СКБ АМУР

Технология работы

1. Создание схемы алгоритма (Дракон)
2. *Конвертирование* в язык CodeSys
3. *Тестирование и отладка* в CodeSys
4. *Модификация* Дракон-схемы
5. Смотри пункт 2

Требуется разработка IDE

на основе Дракона

Анализ

Свойства среды

1. Работа с проектами
2. Редакторская работа со схемами
3. Накопление готовых компонент схем
4. Интерпретация (выполнение) схемы
5. Конвертирование (трансляция) схемы
6. Общее управление и операции

Требуется расширение

стандартного Дракона

Анализ

Работа с проектами

1. Создать проект
2. Открыть/закрыть проект
3. Сохранить проект
4. Экспорт/импорт проекта

Содержимое проекта

1. Стандартные компоненты (алгоритма)
2. Схемы
3. Библиотека компонент проекта

Анализ

Работа со схемами

1. Создать схему (виды схем)
2. Открыть/закрыть схему
3. Сохранить схему
4. Сохранить как компонент (вид компонента)
5. Экспорт/импорт схемы

Содержимое схемы

a) Переменные (виды и типы)

b) Иконы / макроиконы (Дракон)

c) Нестандартные компоненты ()

Анализ

Редактирование схем

1. Навигация по схеме
2. Масштабирование схемы
3. Вставить/удалить/заменить элемент
4. Копировать элемент
5. Разделить / объединить элементы
6. Работа с буфером обмена

Что еще ?

Анализ

Работа с библиотеками

Библиотека проекта и библиотека среды

Виды элементов библиотеки

1. Икона/макроикона
2. Последовательность икон
3. Схема
4. Ветка
5. Переменные (?)
6. Шаблон схемы (?)

Надо как-то называть и обозначать на схеме

Стыковка терминологии – программисты/разработчики

Анализ

Работа с библиотеками

Операции с библиотеками

1. Создать (автоматически в проекте ?)
2. Записать в библиотеку (вид компонента ?)
3. Удалить из библиотеки
4. Заменить компонент в библиотеке
5. Вставить компонент в схему
6. Список компонентов в библиотеке (и выборки)
7. Перенести/копировать в библиотеку среды
8. Перенести/копировать в библиотеку проекта)

Что-то еще ?

Анализ

Интерпретация (выполнение) схемы

Варианты выполнения:

1. Непрерывное
2. Непрерывное до точки останова
3. Пошаговое автоматическое (время задержки)
4. Пошаговое ручное

A. Без захода в компоненты
B. С заходом в компоненты

Работа с переменными

Требуется разработка **среды выполнения**

Анализ

Конвертирование схемы

Варианты:

1. Язык высокого уровня
   - Языки CodeSys
   - Язык Slang – *наши потребности*
   - C/C++ - *наши потребности*
   - JavaScript – *наши потребности*

2. Коды целевого процессора (заказчик)

Отдельная большая тема для разговора.

Анализ

Характеристика	ИС Дракон	DRAKON Editor	DrakonHub	Фабула
Построение схем	+	+	+	+
Конвертация схем	+	+	---	---
Интерпретация схем	---	---	---	---
Отладка схем	---	---	---	---
Работа с проектами	---	+	---	---
Библиотека компонент	---	---	+	---

Реализация

Две разработки
Windows / Linux

Первая
C++ / Qt – первоначальный вариант
Dart / Flatter – текущая разработка

Интерпретатор дракон-схемы - работает
Разрабатывается графический интерфейс

Разработка

Семантическая модель схемы (алгоритм)

а) последовательность *узлов* (примитив)

б) массив последовательностей *узлов* (силуэт)

Узел – элемент схемы с информацией

а) для интерпретации

б) для конвертирования

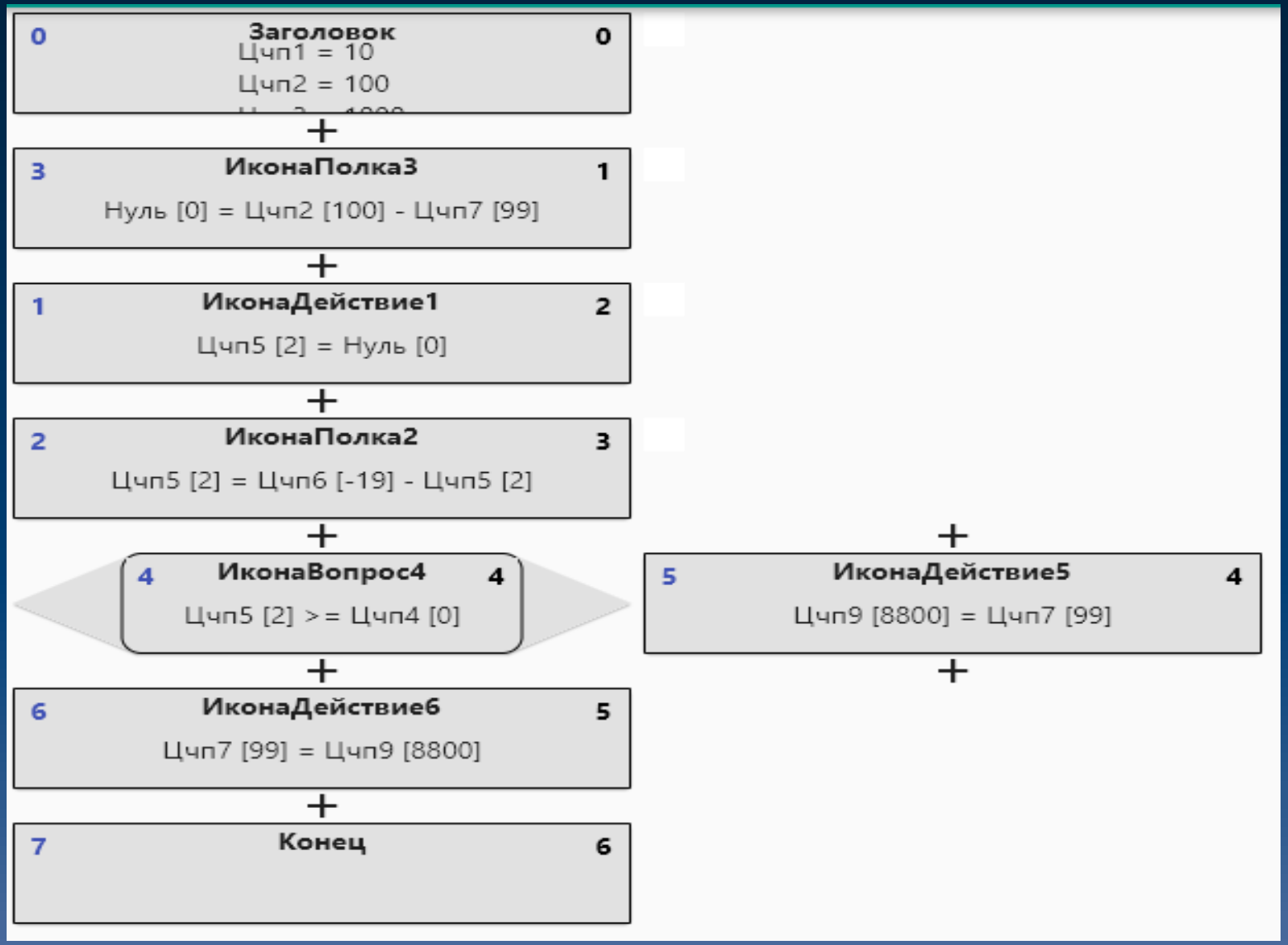
в) для изображения

Обход дерева

```
метод (Узел этот) Выполнить ()
  номер ветки = 1
  пока ветки есть
     если этот.След  != пусто():
        вызвать этот.След.Выполнить();
     конец ветвления;
     номер ветки += 1
  конец цикла
конец Выполнить;
```

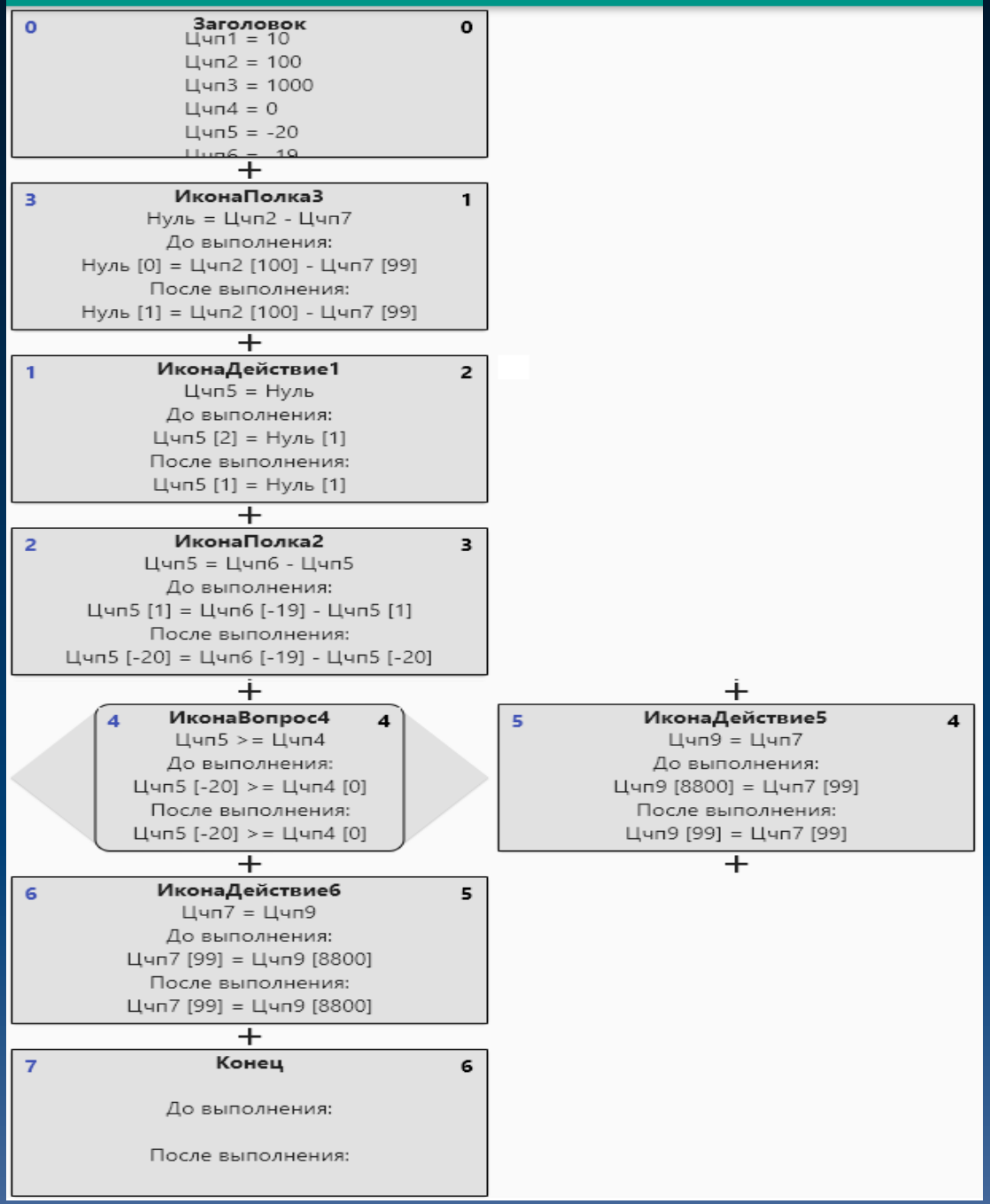
Выполнить() – Конвертировать()

Drakon IDE. Схема до выполнения



Drakon IDE.

Схема после выполнения



Drakon IDE

Две разработки
Windows / Linux

Вторая

- Typescript + Javascript + React
- Среда разработки – Visual Studio Code v1.53.2

Конвертер дракон-схемы - работает
Разрабатывается графический интерфейс

Лаптев Валерий Викторович

WLaptew@yandex.ru

8-905-361-55-24

Астраханский государственный технический университет, кафедра АСОИУ