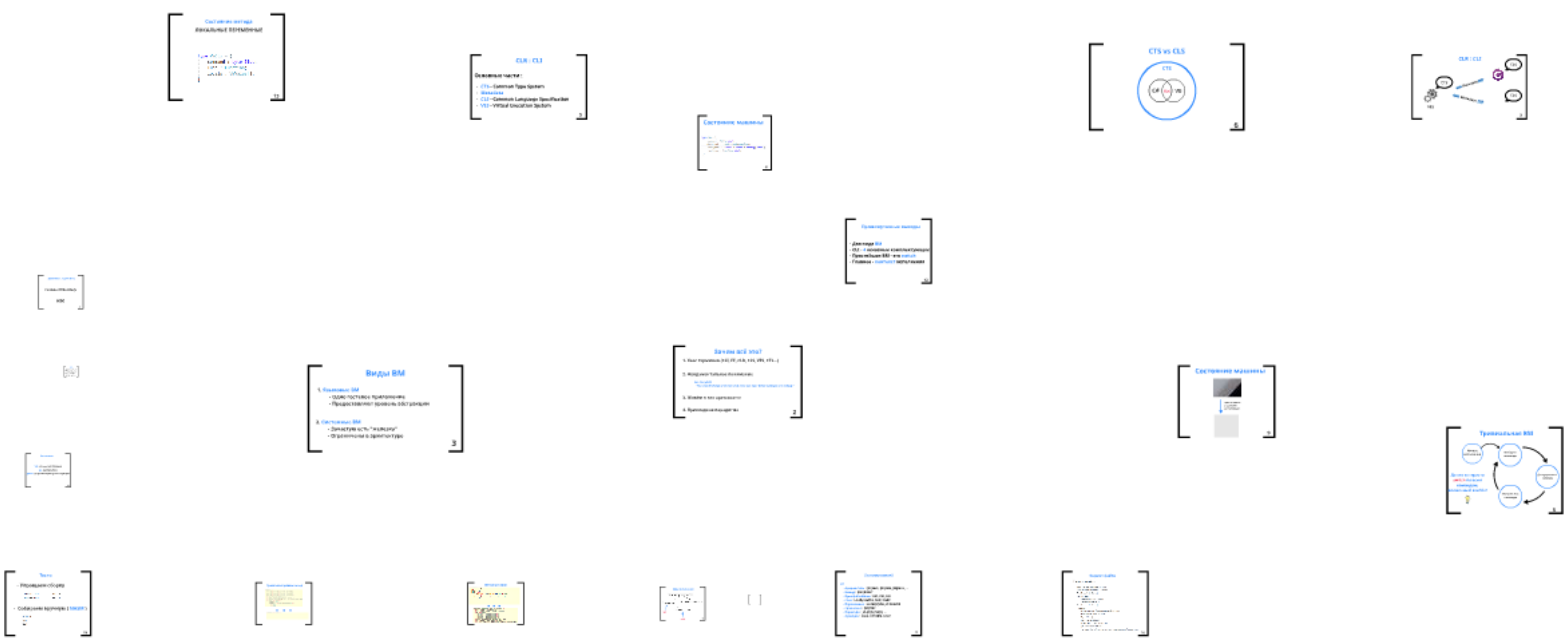
VM via F#

Ю. Волков, ИТМО



Зачем всё это?

1. Хаос терминов (CLI, EE, CLR, CLS, VES, CTS...)

2. Фундаментальное понимание

> *Lee Campbell:*
> *"You should always understand at least one layer below what you are coding."*

3. Живём в век open-source

4. Прикладная парадигма

Виды ВМ

1. **Языковые ВМ**
   - Одно гостевое приложение
   - Предоставляют уровень абстракции

2. **Системные ВМ**
   - Зачастую есть "железка"
   - Ограничены в архитектуре

Занятные моменты

Console.WriteLine()

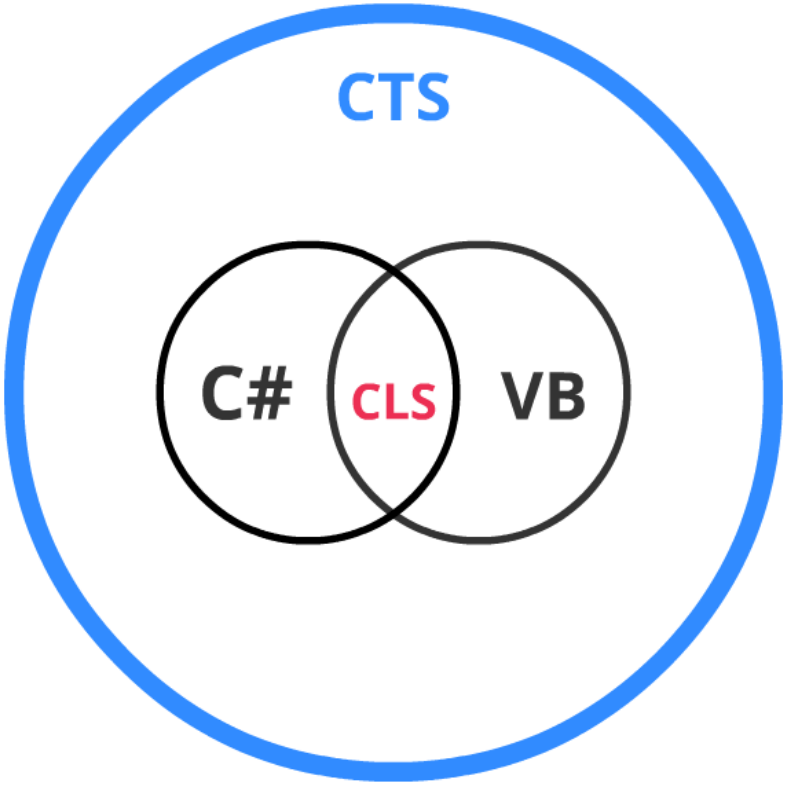
OISC

CLR : CLI

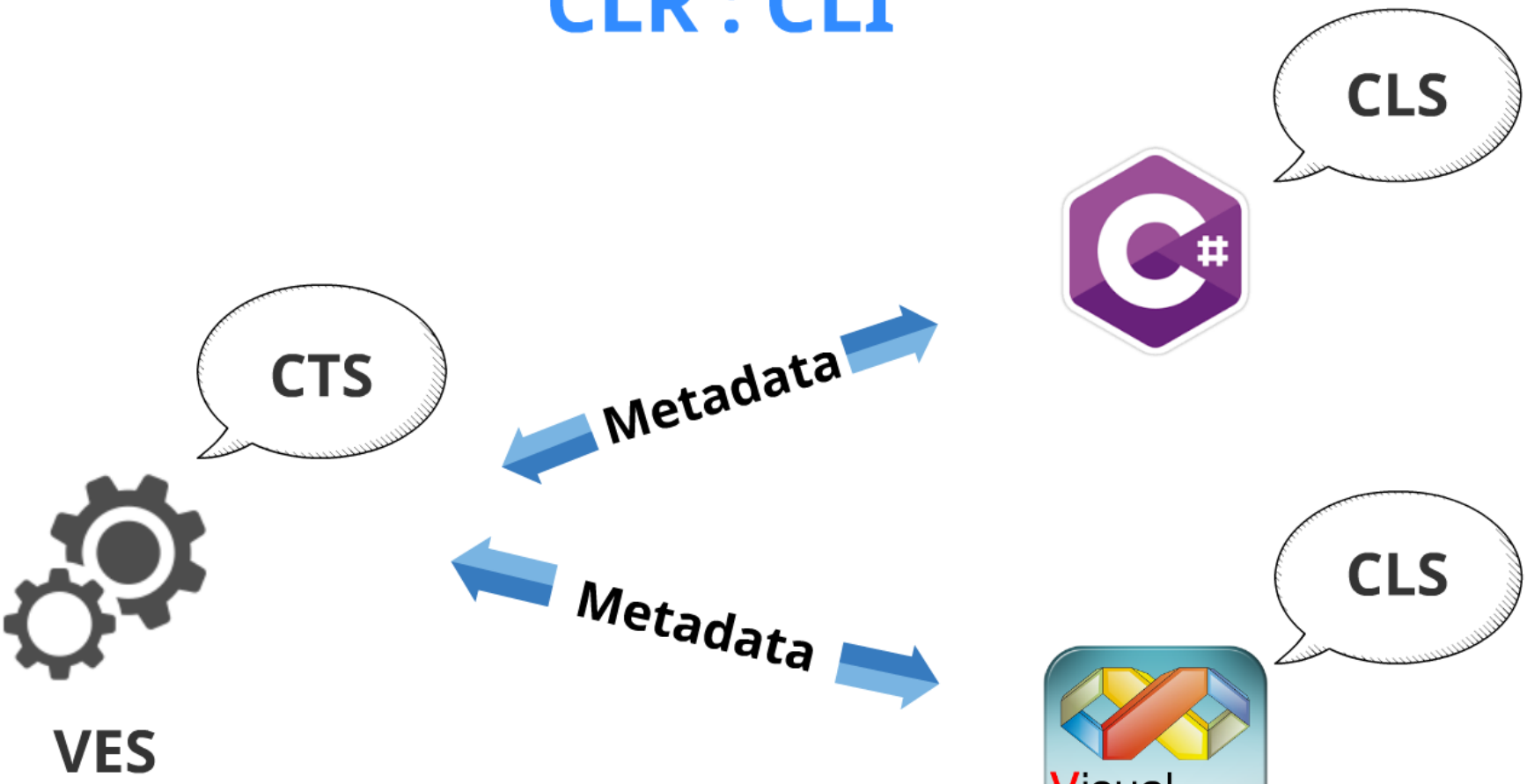
Основные части :

- **CTS - Common Type System**
- **Metadata**
- **CLS - Common Language Specification**
- **VES - Virtual Execution System**

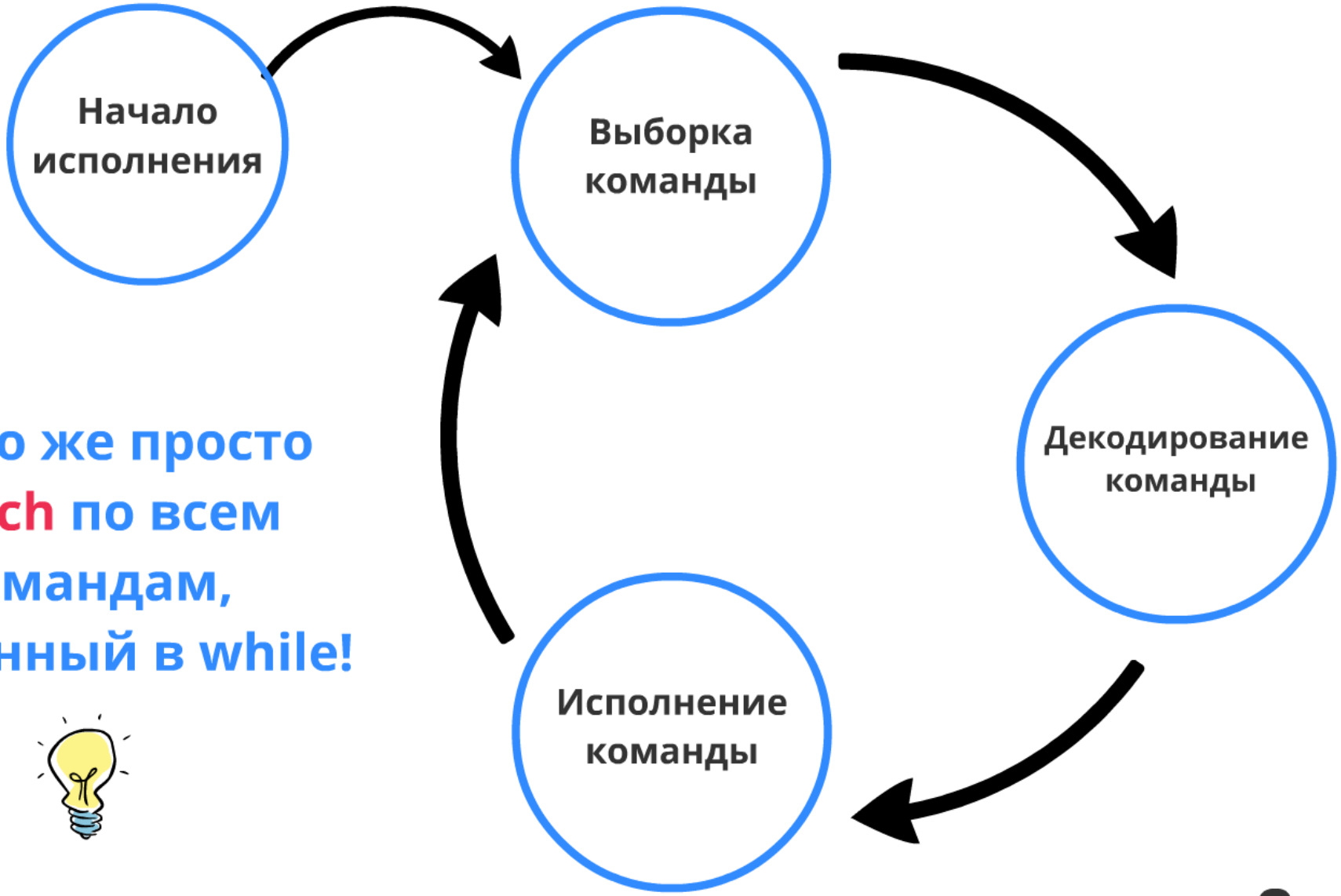
CTS vs CLS



CLR : CLI



Тривиальная ВМ

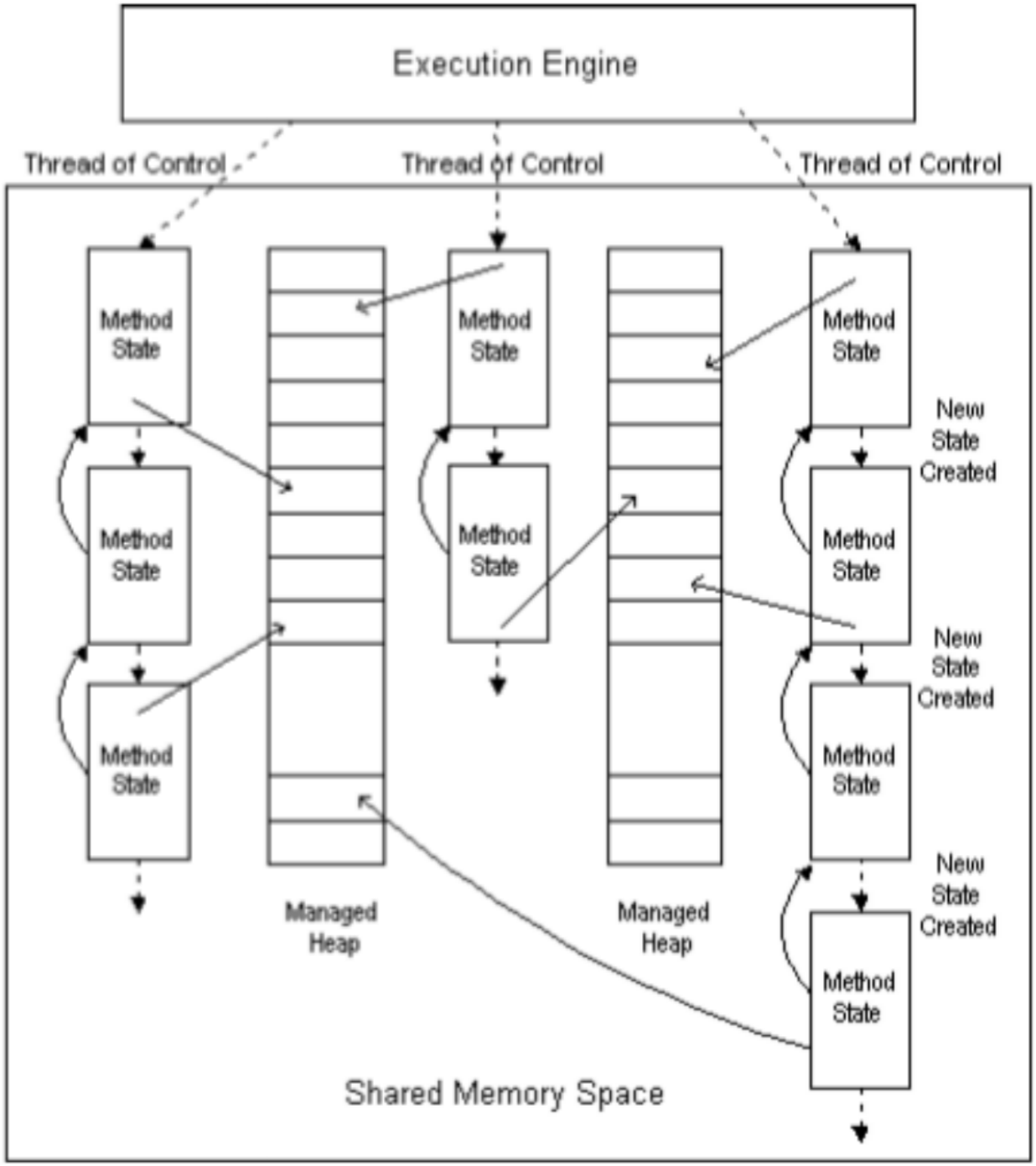


Да это же просто switch по всем командам, вложенный в while!

Состояние машины

Одна машина
с разными
состояниями

Состояние машины



Состояние машины

```
type Vm = {
    context : VmCtx list;
    dataStack : Stack.stack<VmValue>;
    stringPool : (int64 * int64 * string) list ;
    functions : Function list;
}
```

Промежуточные выводы

- Два вида ВМ
- CLI - 4 основные комплектующие
- Простейшая ВМ - это switch
- Главное - контекст исполнения

Состояние метода

ЛОКАЛЬНЫЕ ПЕРЕМЕННЫЕ

```
type VmCtx = {
    command : byte list;
    func : Function;
    locals : VmValue[];
}
```

Функция

```
type VmCtx = {
    command : byte list;
    func : Function;
    locals : VmValue[];
}
```

```
type Function =  {
    fileId : int64
    nameId : int64
    localsCount : int64
    flags : int64
    argsCount : int64
    args : byte list
    byteCodeSize : int64
    code : byte list
}
```

Единый тип данных

```
type VmValue = { arr : byte[] }
    with
        static member Cons (arr : byte[]) = {arr=arr}
        member v.ToInt16()  = BitConverter.ToInt16(v.arr,0)
        member v.ToInt32()  = BitConverter.ToInt32(v.arr,0)
        member v.ToInt64()  = BitConverter.ToInt64(v.arr,0)
        member v.ToBoolean()= BitConverter.ToBoolean(v.arr,0)
        member v.ToChar()   = BitConverter.ToChar(v.arr, 0)
        member v.ToSingle() = BitConverter.ToSingle(v.arr, 0)
        member v.ToDouble() = BitConverter.ToDouble(v.arr, 0)
        member v.ToUInt16() = BitConverter.ToUInt16(v.arr, 0)
        member v.ToUInt32() = BitConverter.ToUInt32(v.arr, 0)
        member v.ToUInt64() = BitConverter.ToUInt64(v.arr,0)
        member v.ToStrPtr() = (BitConverter.ToInt32(v.arr,0),
                               BitConverter.ToInt32(v.arr,4))
```

Унификация - это круто!

Формат файла

Program structure :

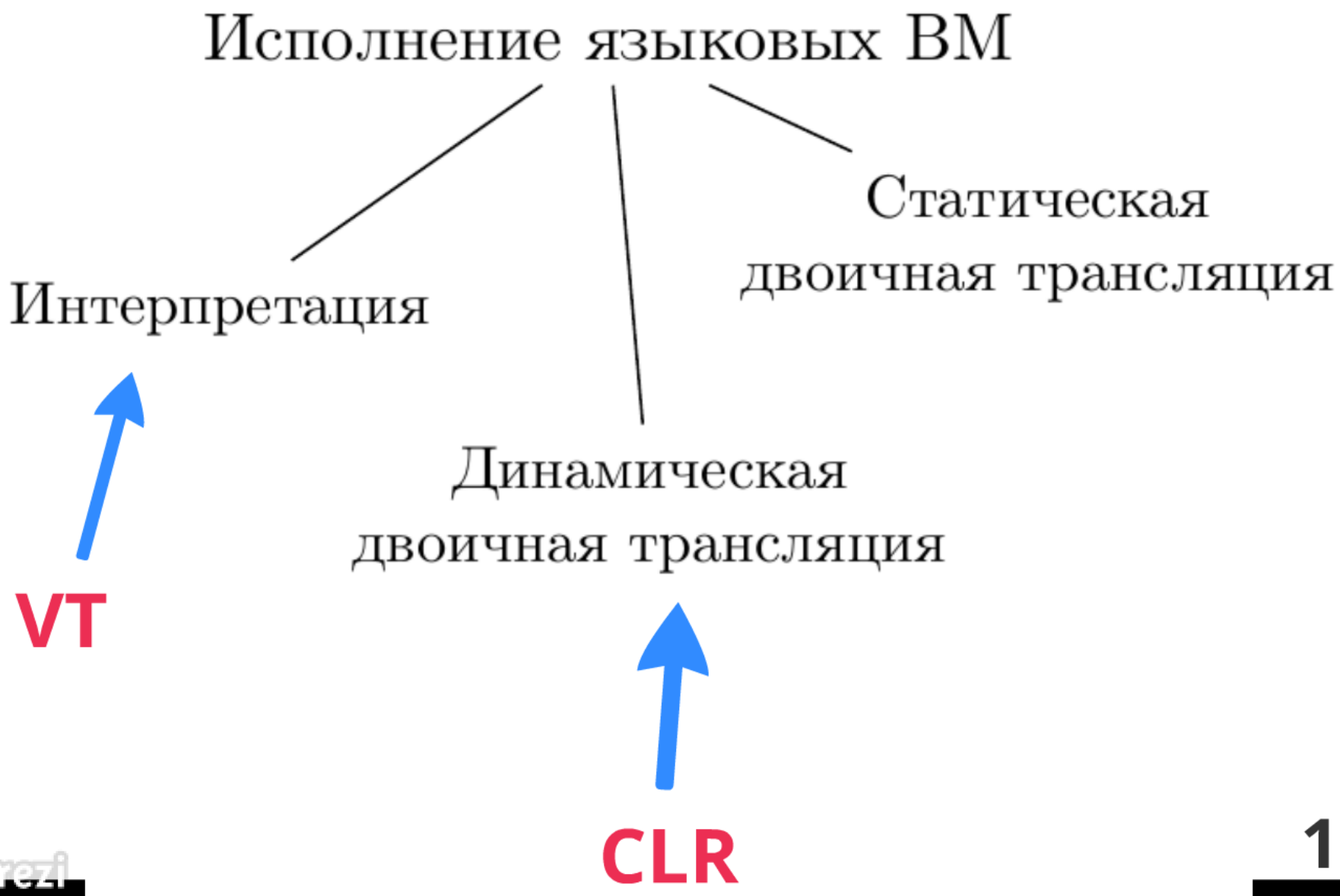
- **Magic numbers** : 0xba 0xba (2 bytes)
- **Required machine version** : 1 (8 bytes)
- **Functions offset** : (8 bytes)
- **Constant pool** :
  - **Constant pool size** : (8 bytes)
  - **Constants divided by \0**
- **Functions count** : (8 bytes)
- **Functions** :
  - **Function name ID (from constant pool)** : (8 bytes)
  - **Local variables count** : (8 bytes)
  - **Flags** : (8 bytes)
  - **Count of args** : (8 bytes)
  - **Type of args** : (1 byte for each arg)
  - **Byte code length** : (8 bytes)
  - **Byte code** : (1 byte for each command or required amount for parameter)

Система команд

BT

- **Арифметика : [ID]ADD, [ID]SUB, [ID]MUL, ...**
- **Вывод : [ISD]PRINT**
- **Преобразование : I2D, D2I, S2I**
- **Стек : LOAD, LOADS, POP, SWAP**
- **Переменные : LOAD[S]VAR, STOREVAR**
- **Сравнение : [DI]CMP**
- **Переходы : JA, JZ[I], JNZ[I], ...**
- **Функции : CALL, RETURN, STOP**

Виды исполнения



Тесты

- **Упрощаем сборку**

```
%define DADD        db    3
%define IADD        db    4
```

- **Собираем вручную (NASM)**

```
LOADS 5
SPRINT
DUMP
```

Выводы

- Многие сущности описываются в виде ВМ
- ВМ описывается текущим состоянием (контекстом)
- CLI состоит из 4х основных частей
- Thread в .Net представляет собой связный список контекстов методов
  - Именно поэтому локальные переменные не затераются
- Подобная парадигма бывает полезна на практике

Контакты

VK : vk.com/id127693696

tg : @yurijvolkov

github : yurijvolkov/sysProgrammingProject

VM via F#

Ю. Волков, ИТМО