Оберон: семейство языков и программных сред/ОС. Практика их применения

Илья Ермаков

https://iermakov.ru/

28 июня 2019 года

- **Oberon** – семейство языков программирования, ОС и сделанных на их основе интегрированных сред разработки/выполнения (IDEE)

- «Гнездо» научно-технологической школы: Университет **ETHZ**, Цюрих, Швейцария

- Никлаус Вирт и Юрг Гуткнехт

- Вирт – лауреат премии Тьюринга («нобелевка» в ИТ)

Классический Oberon

- **Oberon** Никлауса Вирта: редакции от автора 1989, 2007, 2013, 2017… О Project Oberon / OS Oberon в целом – позднее.
- Современные реализации называют **Oberon-07** (независимо от наличия мелких изменений более поздних лет).
- Сверхкомпактный модульный строго статически типизированный безопасный язык системного программирования (ОС, микроконтроллеры…), со сборкой мусора (необязательной)
- Является ядром для расширения в других диалектах
- Поддерживает ООП-стиль, но не содержит статических ООП-средств (методов, статически привязанных к типу данных)
- Более десятка реализаций под разные платформы, разного качества, для распространённых 32-битных микроконтроллеров — используют активно

Component Pascal / BlackBox

- Oberon-2 – расширен для прикладных задач «классическими» методами. В чистом виде почти не используется (Oxford Oberon-2 – учебный).
- **Component Pascal** – развитие Oberon-2 компанией Oberon Microsystems в рамках среды BlackBox Component Builder, видимо, наиболее активно применяемый в прикладных задачах язык. О нём – позже. *Сложился к началу 2000-х.*
- Основная реализация: среда BlackBox. В 2013-м BlackBox 1.6 выпущен под BSD-like лицензией, далее – несколько community-веток.
- Вторая: GPCP (Gardens Point Component Pascal) – для .NET и JVM. Не развивается.
- Дискуссии о смене названия на вариант с CP на вариант с Oberon

Active Oberon, OC A2

- Развивается группой исследователей в ETHZ под руководством Юрга Гуткнехта с 1997 года
- Модель активных объектов (многопоточность и синхронизация).
- Основная реализация – ОС A2 (ранее – BlueBottle). Может работать поверх Win и Lin
- Мягкое реальное время
- Графическая система zoomable world
- Встраиваемые мультимедиа-применения
- Постоянно развивается в разных направлениях аспирантами ETHZ и другими разработчиками (минус: без централизованного документирования, сложно отследить)
- Как и все Оберон ОС: работа в едином адресном пространстве, защита языковыми средствами (→ сложно изолировать исполнение недоверенных компонентов. Т.е. не многопользовательская ОС, встраиваемые и серверные применения либо «тонкий клиент»).

Исследовательские системы

- Евгений Зуев во время работы в ETHZ – совместные проекты с Microsoft Research:
- Active Oberon for .NET
- Язык Zonnon. Сильно расширен конструкциями, вариантами модульности, введены средства обмена сообщениями и описаний протоколов грамматиками (контроль компилятором и во время выполнения)
- В ETHZ – система Composita (параллелизм на сообщениях, особая «шитая» компилятором многопоточность – миллионы параллельных сущностей)
- Последствия в MS Research: система Axum, Oberon Script, OS Singularity и др.

		Фортран и Алгол		LISP (1958)
1970	Pascal		C	
1980	Язык Модула-2, ПК Lilith, ОС Medos	Победа в крупных системах высокоуровневых языков: Модула-2, Ада, Эль-76, IBM AS/400...	Развитие микрокомпьютеров. BASIC и C. Ну и Turbo Pascal.	Smalltalk
1984-1989	**Project Oberon:** Язык и ОС Оберон, ПК Ceres			
90-е	BlackBox, A2	Java ООП-архитектуры: COM, CORBA **Borland Delphi**	C++ (на «плечах» MS, затем и волна GNU)	ФП
2000-е	BlackBox, A2	**.NET, Java** **Enterprise ООП-архитектуры**	Уход C/C++ из прикладных задач	Динамические языки для Web

- Древнейшие «холиворы»: высокоуровневый подход vs. низкоуровневый; статическая vs. динамическая типизация
- Сегодня: отдельный «холивор» у системщиков (высокоуровневые системные языки – Модула-2, Оберон, Ada, Go, Rust vs. «всё хорошо и на Си, и никто не ограничивает»). Проблемы переполнения буфера и разрушения памяти → киберуязвимости.
- И отдельный «холивор» у прикладников: универсальные языки программирования «слишком близки к машине», «быстродействие неважно», «даёшь максимальные абстракции и синтаксический сахар».
- Статическая типизация всё же постепенно побеждает (ФП, Go для Web, Dart и его экономика в Google)

- Итак, в контексте 1989-го: что такое «крупная задача»?
- Это, всё же, преимущественно консолидированная система из сотен и тысяч модулей в одном адресном пространстве. Сложность и архитектура контролируется языковыми средствами. Из крупных подходов: Ada, Эль-76; из лёгких: Modula-2, Oberon.
- Вычислительное ПО, системы моделирования, системы управления, финансовые системы, сложные настольные системы – САПР и др. инженерный софт, SCADA, редакторы и т. п. Всё это никуда не ушло, но это «элитные задачи», массовое ИТ занимается другим. Свежий пример на BlackBox: система промышленного моделирования Mobatec (https://www.mobatec.nl/)
- Java Enterprise. Даже в системах над БД – всё равно некая «толстая» объектная языковая модель.
- Архитектурно: ООП-паттерны. Фаулер «Архитектура корпоративных приложений». Эванс «Предметно-ориентированное проектирование» (Domain-Driven Design)
- В общем, для этих задач в плане инструментов и методов «всё хорошо». Особо и улучшать нечего с 90-х. Только упрощать и «подстригать разросшиеся кусты» (как в случае от Java и .NET к тому же Go).

- Универсальный язык в понимании 80-х-90-х (Ada, Modula-2 – признанные)
- должен закрывать задачи Фортрана (тупая и эффективная трансляция в машинный код, многомерные массивы, размещение на стеке и т.п.)
- статическая типизация, для описания абстрактных типов данных для сложных проектов. Строгая – для предотвращения ошибок.
- модульность, для управления сложностью.
- переносимость (небольшое количество изолированных системных средств).
- *Добавки 90-х:* – расширяемость и обобщение через ООП
- безопасность и автоматическое или частично автоматическое управление памятью.
- Оберон «отжимает» это до миниатюрного базиса. Кто-то считает, что слишком миниатюрного (ну, той же критике подвергается Go).

- Java и C# в роли универсальных языков?
- Ср. Фортран/C/C++ и Модула-2/Ada/Oberon. Очевидно, что вторые – «можно всё то же и так же быстро, но аккуратнее и надёжнее».
- Ср. с Java и .NET: прослойка JVM или IL, недоработанная модель типов.
- Java – только динамические массивы и структуры, только указательная семантика для них (не присвоить и не передать «по значению»). Язык с достаточно случайной эволюцией.
- Синтаксический сахар из мира ФП и скриптов – ещё дальше от уровня универсального языка в классическом понимании.
- Поэтому и появился Go — «подстригание» Java и C#. Оберон назван явно предком. См. лекцию Роберта Гризмера («Oberon is extemely effective»).
- Пара слов о вычислительно-научных задачах: как было, так и есть – библиотеки на Фортране и C++. Питон – как оболочка.
- Концепция Оберона как языка и среды – и язык не медленнее Фортрана, и его среда – «почти интерпретатор» + документы и графика. Но подводит то, что компилятор – старый.

Классический Oberon

90-е	BlackBox, A2	Java ООП-архитектуры: COM, CORBA **Borland Delphi**	C++ (на «плечах» MS, затем и волна GNU)	ФП
2000-е	BlackBox, A2	**.NET, Java** **Enterprise ООП-архитектуры** Возникновение крупных Web-сервисов. Enterprise (Java и т. п.) – не тянут. **«Крупная задача» теперь – это интеграция сервисов и потоков данных.** **Функциональные и динамические языки – как бы, удобнее.**	Уход C/C++ из прикладных задач Микроконтроллеры. Повторение истории с ПК: опять ограниченные ресурсы, в ущерб абстракции и надёжности	Динамические языки для Web Триумф Node.js по производительност и для массового обслуживания
2010	BlackBox, A2	Синт. сахар в Java и C# из опыта ФП и скриптов Go	Попытки использования на системном уровне Java, C#, Python… Эксперимент Rust	Dart и его «экономика» (выигрыш 40% относительно JS)

- **Project Oberon (1989)**
- Вдохновлен отчасти Smalltalk, но воспроизведён на статически типизированном эффективно компилируемом языке
- Первый компилируемый статически типизированный язык с автоматическим управлением памятью
- Интеграция такого языка со средой и интерактивными возможностями («почти как интерпретатор»)
- Определяющее влияние на Java и .NET
- **Project Oberon (2013)** – плюс процессор на ПЛИС

Средства классического Оберона

- Отшлифованный Паскаль-синтаксис (нет лишних BEGIN и т. п.)
- Модульность, импорт-экспорт. Обязательная полная квалификация имён. Модуль – единица компиляции и загрузки-выгрузки. Хранит машинный код и метаинформацию о типах и переменных, для рефлексии и метапрограммирования.
- В языке всё заточено под динамическое связывание. Нельзя, чтобы требовалась перекомпиляция других модулей из-за изменений внутри модуля. Поэтому, например, нет Enum и т. п. (неясно, как поддерживать обратную совместимость других модулей без перекомпиляции).
- Для ООП: расширяемые записи. Быстрый IS/WITH (3 сравнения).
- Вся трансляция прозрачна на маш. код (любая конструкция разворачивается в детерминированную последовательность инструкций) – можно эффективно оптимизировать и гарантировать жёсткое реальное время)
- Использование стека (VAR-параметры, открытые массивы — в том числе по копированию).
- Component Pascal добавляет IN и OUT-параметры.
- Паттерн «сообщения Гуткнехта»
- Динамическая память – динамические записи и массивы, сборка мусора
- ASSERT и защитный стиль программирования (предусловия – везде)

```
MODULE MyprojModule1;
 IMPORT  Log := StdLog,  In := ipuiK161, Math;

 PROCEDURE Do*;
  VAR x: REAL;
 BEGIN
  In.Open;
  In.Real(x);
  ASSERT(In.done);
  Log.Real(Math.Sin(x)); Log.Ln
 END Do;

 PROCEDURE InternalProc;
 BEGIN

 END InternalProc;

END MyprojModule1.
```

❗MyprojModule1.Do 0.5

IF ... THEN

ELSE

END

WHILE ... DO

END

(опускаю у типов и полей метку *
экспорта)
Классический Оберон:

```
TYPE
  Message = RECORD END;

  SetRecMsg = RECORD (Message)
    view: Views.View
    l, t, r, b: INTEGER
  END;

  InstallSubView = RECORD (Message)
    view, subView: Views.Vi
    x, y: INTEGER
  END;
```

```
PROCEDURE HandleMsg (VAR msg: Message);
BEGIN
  WITH msg: SetRecMsg DO
    ...

  | msg: InstallSubView DO

  ELSE
    OtherMod.HandleMsg(msg)
  END
END HandleMsg;
```

```
PROCEDURE Do;
  VAR msg: SetRecMsg; (* НА СТЕКЕ! *)
BEGIN
  msg.l := 10; msg.t := 15; ....
  HandleMsg(msg)
END Do;
```

В Компонентном Паскале добавляются доп. ключевые слова:

```
TYPE
  Message = ABSTRACT RECORD END;

  SetRecMsg = EXTENSIBLE RECORD (Message)
    view: Views.View
    l, t, r, b: INTEGER
  END;

  InstallSubView = EXTENSIBLE RECORD (Message)
    view, subView: Views.View;
    x, y: INTEGER
  END
```

ООП в классическом Обероне:

```
TYPE
  View = POINTER TO RECORD
    HandleMsg: PROCEDURE (v: View; VAR msg: HandleMsg)
  END;

  v.HandleMsg(v, msg)
```

Открытые массивы

```
PROCEDURE Proc (VAR cube: ARRAY OF ARRAY OF ARRAY OF REAL);
  VAR i, j, k: INTEGER;
BEGIN
  FOR i := LEN(cube, 0) DO
    FOR j := LEN(cube, 1) DO
      FOR k := LEN(cube, 2) DO
        ...

VAR staticCube: ARRAY 128, 128, 128 OF REAL;
Proc(staticCube);

VAR dynCube: POINTER TO ARRAY OF ARRAY OF ARRAY OF REAL;
NEW(dynCube, 256, 256, 256);
Proc(dynCube);
```

PROCEDURE Proc (~~VAR~~ cube: ARRAY OF ARRAY OF ARRAY OF REAL);
- будет создаваться автокопия массива на стеке

Component Pascal
в реализации BlackBox от Oberon Microsystems

- Синхронизирован по базовым типам с Java
- Esmertec JBed – изначально биязыковый проект (внутри и КП, и Java вместе)
- От Оберон-2 – методы (называется «связанные с типом процедуры», как в Ada).
- Добавлены спецификаторы для ООП-паттернов. Кроме ABSTRACT, есть и другие (LIMITED, NEW)
- Ядро (рантайм) – < 3 тыс. строк. Метаинформация/рефлексия, динамическая загрузка/выгрузка бинарных модулей, управления памятью и сборка мусора, поддержка исключений (конструкции языка нет, библиотечный TRY есть)
- Компилятор – 11 тыс. строк.
- Параллелизм для ББ – свои «зелёные потоки» (KrlTasks Ермакова, сопрограммы Дагаева)

Относительное позиционирование Оберон-языков

- Оберон и C/C++
- Оберон и Java/.NET
- Оберон и Go
- Оберон и Rust
- Оберон и Ada
- Оберон и математические пакеты (Mathcad и т.п.)
- Оберон и связка C++/Python и C++/Lua

- Сравнивать с динамическими или функциональными языками имеет смысл сначала как в целом представителя семейства классических универсальных компилируемых языков

Ссылки

- Обзор истории языков и подходов программирования (осторожно – «Обероноцентричный») http://www.iermakov.ru/study/Languages.pdf
- Курс программирования «с нуля» на КП/ББ – пока скидываю по запросу
- Пфистер «Компонентное ПО»
- Про паттерны циклов
- Одна задача – много вариантов
- Роберт Мартин Будущее программирования. https://infostart.ru/public/975789/