Масштабируемые Оберон-технологии как средства обеспечения защищенного ПО критически важных систем

Дагаев Дмитрий Викторович,
Главный Эксперт,
АО «Русатом Автоматизированные системы управления»
Консультант проекта Информатика-21

Прогресс “от ограничения” (Роберт Мартин)

Ограничение прямой передачи управления привело к созданию структурного программирования.

~~GOTO next~~

Ограничение присваивания привело к созданию функционального программирования.

~~Y := X~~

Ограничение косвенной передачи управления - к созданию объектно-ориентированного программирования.

~~((Method)*handler)(obj, arg)~~

Продать «смотрите, что я ради вас сделал» куда легче, чем «смотрите, от чего я ради вас уклонился» (Нассим Талеб)

Можно ли монетизировать «наличие отсутствия»?

МЭК 60880 – обеспечение функциональной безопасности систем для АЭС категории А

B.2cb Избегать использования универсальных ОС ~~std OS~~;

B.2cd OS должна содержать только необходимые функции ~~stdlib~~;

B.4ag Циклы только с постоянными максимальными областями значений переменной цикла ~~WHILE REPEAT LOOP~~;

B.2dd Время прогона не должно существенно меняться от изменения входных данных ~~IF~~;

B.2ee Применение или блокирование прерываний должно быть тщательно оформлено документально ~~INTR~~;

B.4dc Массивы должны иметь фиксированную длину ~~NEW~~;

Обеспечение соответствия требованиям ФБ определяется ***отсутствием***, а не наличием функциональности.

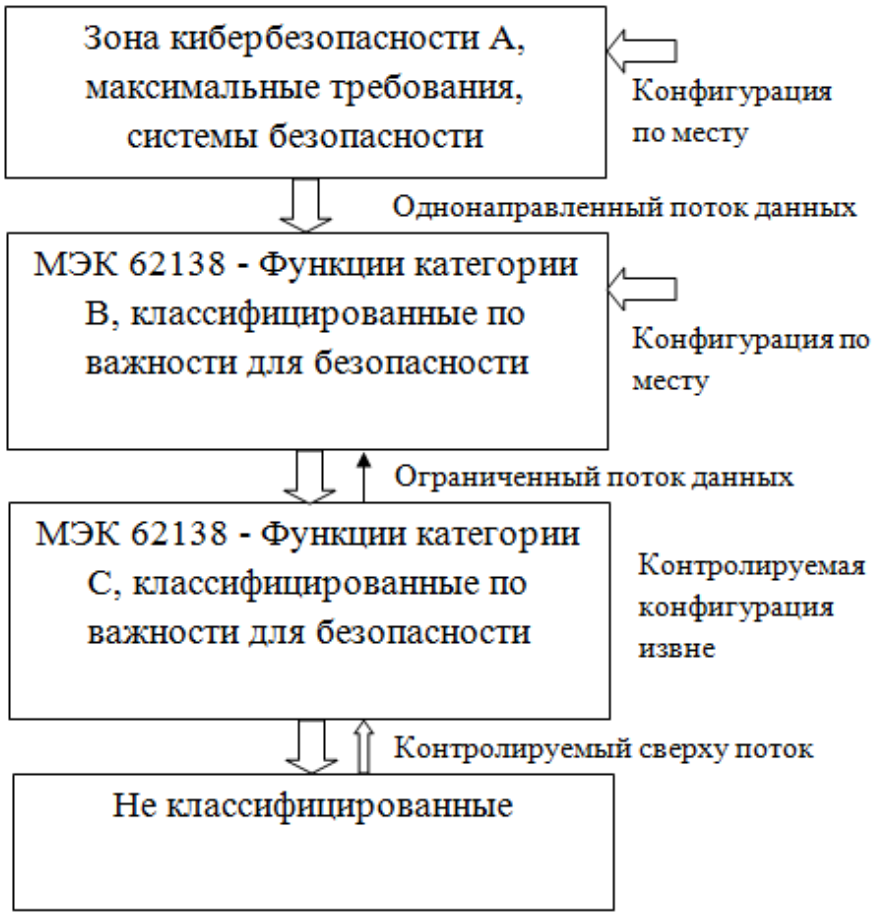
МЭК 62645 – Требования по защищенности программ СКУ

Разбиение на зоны безопасности, разделенные шлюзами.

Отсутствие ~~удаленного обновления~~ ПО (A, B);

Отсутствие ~~удаленной записи~~ данных: уставки, параметры БД (A);

Ограничение удаленной записи данных: уставки, параметры БД (B);



РД ФСТЭК контроль отсутствия НДВ и показатели защищенности от НСД

3.5.3НДВ семантический контроль отсутствия заданных конструкций в исходных текстах ПО из списка (базы) потенциально ~~опасных конструкций~~;

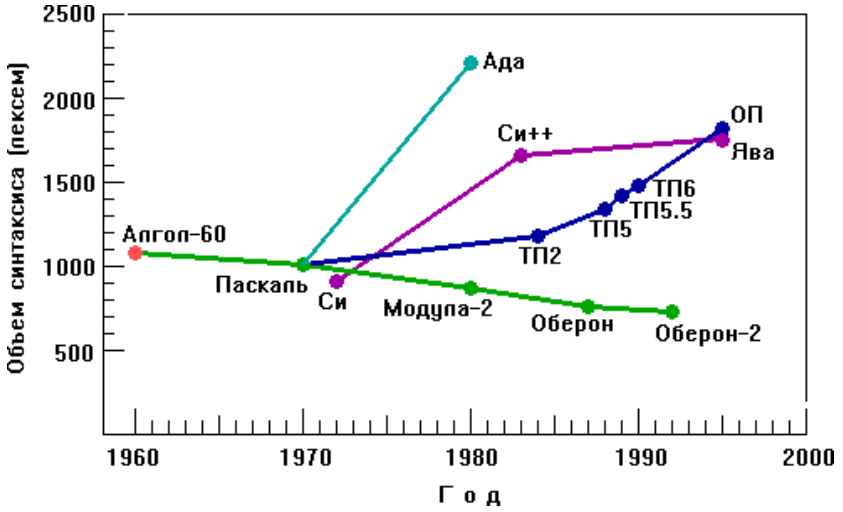
3.4.3НДВ контроль выполнения функциональных объектов (ветвей);

3.3.3НДВ контроль связей функциональных объектов (модулей, процедур, функций) по управлению ~~(нет переполнения буфера)~~;

2.5.3НСД очистка оперативной и внешней памяти должна производиться путем записи маскирующей информации в память при ее освобождении (перераспределении). Гарантирует ~~(отсутствие остаточной информации)~~;

Обеспечение соответствия требованиям КБ определяется гарантиями ***отсутствия***.

Оберон и борьба с избыточной сложностью



Сложность синтаксиса языков программирования (С.З.Свердлов).

Только Оберон развивается в сторону уменьшения сложности.

Когда мощность системы измеряется числом ее возможностей, количество становится более важным, чем качество» (Н.Вирт)

Долой “жирные” программы (Н.Вирт)

Я предлагаю отныне строго придерживаться разработки и реализации интеллектуально-управляемых программ (Э.Дейкстра)

Проект Оберон (Н.Вирт, Ю.Гуткнехт) 1986-1989

ОС: управление задачами, динамические модули, файловая система и сериализация, драйверы устройств, сети, серверные задачи;

Однопроходный компилятор;

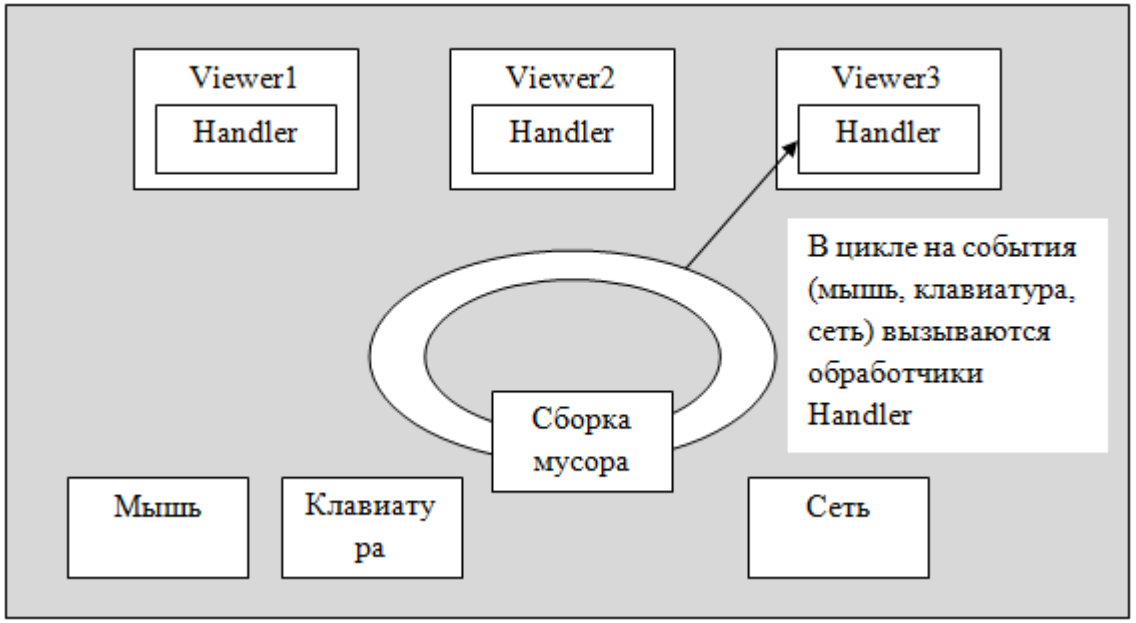
Графическая система и система гипертекста.

Содержит < 20 тыс строк, 120 Кбайт.

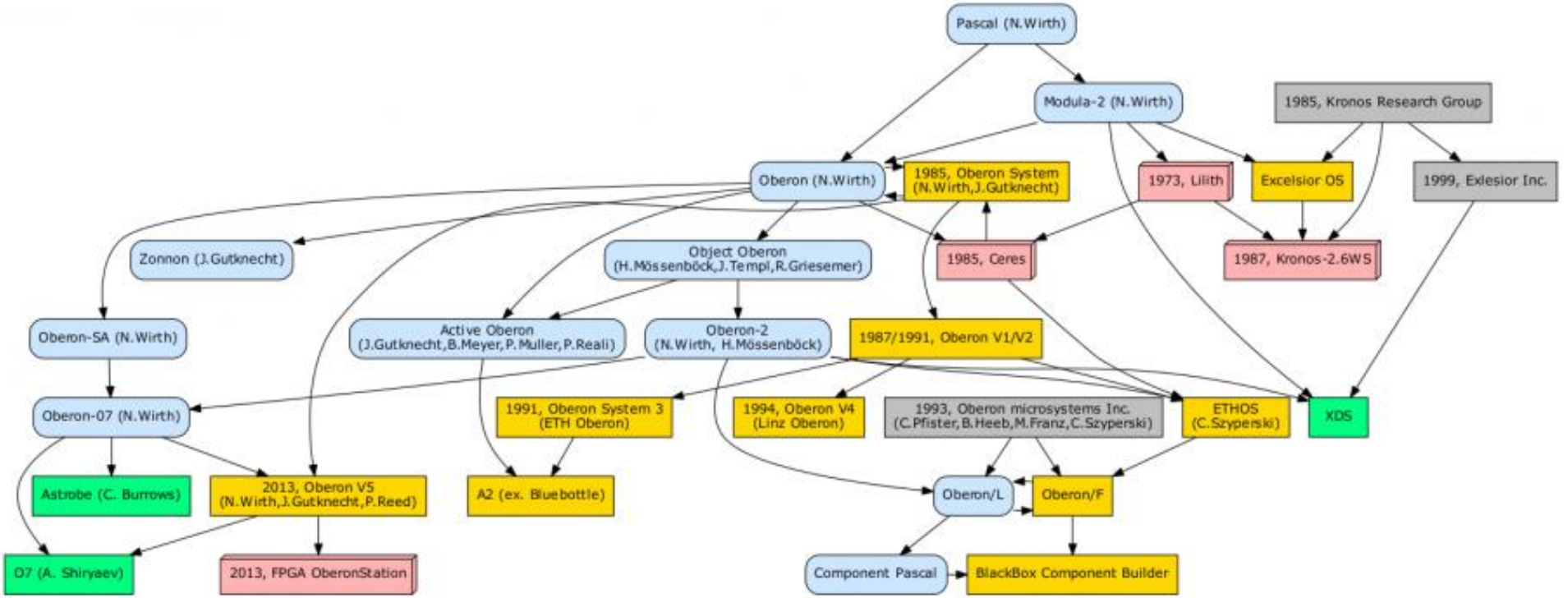
Трудозатраты 5 ч/лет

ОС Оберон имеет только один основной цикл обработки событий Loop;

Атомарность реализована до вызова процедуры.



Генеалогия Оберонов (И.А.Денисов)



Данное дерево хотя и не полно, но множество реализаций демонстрирует, что Н.Вирту удалось создать ***инженерную школу*** и оказать влияние на многих (например, проект Kronos, Excelsior OS и компилятор XDS для Оберон-2 и Модула-2).

Ветвь развития Оберона Н.Вирта

Oberon Strong-ARM → Oberon-07 → Oberon-2016

- WHILE поднят до ЦД, ~~LOOP~~, изменен синтаксис CASE;
- Исключены ~~WITH, HALT, SHORTINT, LONGINT, LONGREAL~~;
- Синтаксис ~~RETURN~~ только в конце функции;
- Запрет передачи ~~структурированных параметров по значению~~;
- Импорт ~~переменных по записи~~ – исключить;
- Указатели ~~на массивы~~ – исключить.

Oberon 2013

- Перенос с NS32032 на FPGA Xilinx Spartan-3;
- Переработка компилятора: однопроходный, 2900 строк для RISC-архитектуры, 3 и 10 сек сборки компилятора и всей ОС;
- SD-card вместо диска и PS-2 для клавиатуры и мыши.

Реализации Oberon-07

Astrobe (C.Burrows) - Коммерческий продукт для встроенных применений
www.astrobe.com

O7 (А.Ширяев) - Дистанционно-пилотируемые БПЛА, АО «НЦПЭ»

Без ОС, поверх «голого» железа, жесткое реальное время от таймера.

Корректировка механизма управления динамической памятью.
Без ~~сборщика мусора~~.

Процедуры - обработчики прерываний

```
PROCEDURE Timer1Handler[IRQ].
```

RESTRICT – Инициатива гарантий использования

RESTRICT дает профилактическую защиту ПО против «хирургии последствий».

Операция RESTRICT ‘–’ отключает операторы, ‘+’ восстанавливает отключенные, а ‘*’ ограничивает их использование заранее определенным образом.

Для адаптирования к стандарту 60880 потребуется нижестоящее определение и вариант бэкенда, который это поддерживает. Отключаем ~~NEW, WHILE, LOOP, REPEAT, CASE/ELSE~~. Отключаем ~~рекурсию~~.

```
RESTRICT –NEW –WHILE –LOOP –REPEAT –ELSE(CASE)
  +EXIT(FOR)-”Recursion”;
```

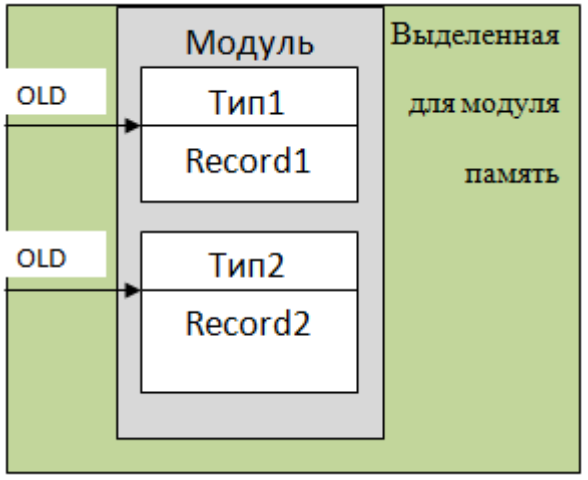
Близкая аналогия – Ada Restrictions.

```
pragma Restrictions (No_Recursion);
```

Предложения по системе гарантий памяти и времени

Замена динамической памяти ~~NEW~~ на внутримодульную через функцию OLD.

```
Rec = RECORD [typed].. END;

RPtr = POINTER TO Rec;

VAR myRec: Rec; myPtr: RPtr;

myPtr := OLD(myRec);
```



Гарантии времени требуют использовать циклы только с постоянными максимальными областями значений переменной цикла.

Циклы ~~WHILE, LOOP, REPEAT~~ исключаются.

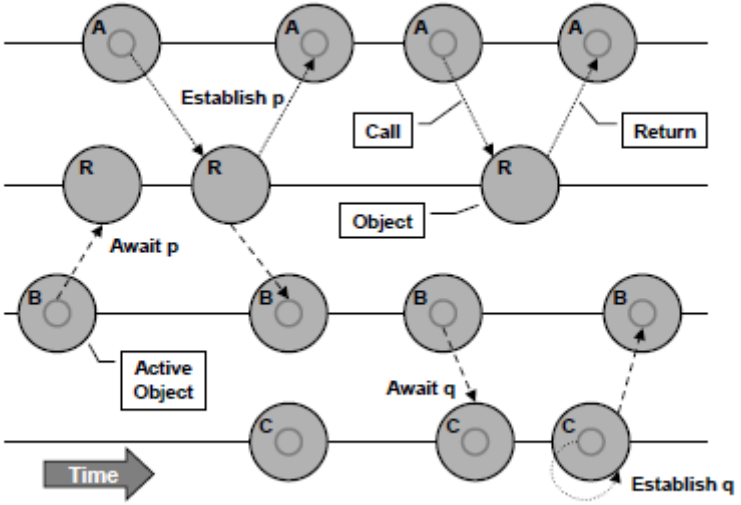
```
FOR j := 0 TO N-1 DO

   IF in.value[j].quality =
    Q_BAD_COMM THEN

      EXIT

   END

END
```

Активный Оберон и A2

A2 - github.com/metacore/A2OS

Активные объекты со средствами синхронизации вместо процессов.

Реализация многозадачности на основе легковесных активных объектов в общей памяти.




```
Synchronizer = OBJECT
   awake: BOOLEAN
   PROCEDURE Wait;
   BEGIN {EXCLUSIVE}
    AWAIT(awake);
    awake := FALSE
   END Wait;
   PROCEDURE WakeUp;
   BEGIN {EXCLUSIVE}
    awake := TRUE
   END WakeUp;
END Synchronizer;
```

Ariane 5 Flight 501 и принцип элегантной деградации

Ariane 5 - Отказ основного и резервного компьютера, приведший к потере спутника 4 июня 1996.

Преобразование данных из 64-битного действительного числа в 16-битное целое привело к переполнению и вызвало аппаратное исключение.

```
P_M_DERIVE(T_ALG.E_BH) := UC_16S_EN_16NS
   (TDB.T_ENTIER_16S ((1.0/C_M_LSB_BH) *
   G_M_INFO_DERIVE(T_ALG.E_BH)));
```

Система выявила и распознала ошибку. Однако, спецификация механизма обработки ошибок не соответствовала ситуации, что вызвало выход ракеты на запредельную траекторию.

Элегантная деградация в A2 - многоуровневая обработка ошибок и реконфигурирование:

- На уровне переменных – установка кодов качества и обработка с учетом кодов;
- Иначе на уровне процедур – обработка исключений процедур;
- Иначе на уровне активных объектов – безопасный рестарт с восстановлением переменных;
- Иначе на уровне модуля – рестарт секции инициализации модуля с восстановлением глобальных переменных.
- Иначе – перезагрузка.

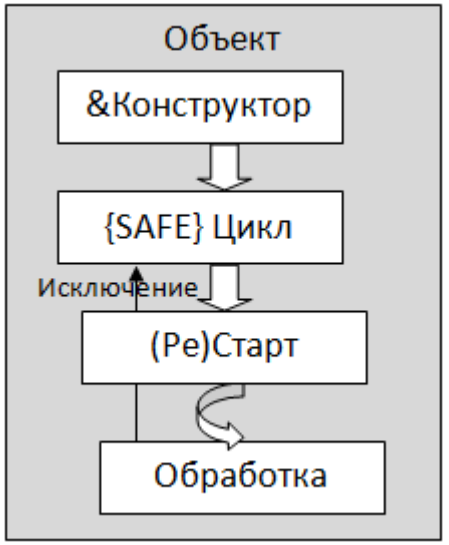
Гарантии процедур и объектов в A2

Процедуры обработки реального времени с флагом {REALTIME} добавляют свою систему ограничений:

- Запрет прерываний ~~INTR~~;
- Запрет выделения динамической памяти ~~NEW~~ ;
- Можно предложить добавление гарантий времени выполнения.

В активных объектах с флагом {SAFE} исключения приводят к рестарту активного объекта. Процедуру рестарта предлагается дополнить запретом выделения динамической памяти ~~NEW~~.

Выделение динамической памяти требуется только в конструкторе.



Oberon+ETHOS → Oberon/L+Oberon/F → Component Pascal+BlackBox

Создание парадигмы компонентно-ориентированного программирования:

- сокрытие информации в рамках модульности;
- позднее связывание и механизмы мета-программирования;
- безопасность статической типизации, указателей и границ массивов;
- ограничения наследования в части наследования реализации с исключением ~~**хрупкого базового класса EXTENSIBLE**~~(В C++ и Java сложно отследить наличие ключевого слова final в наследованиях реализации);
- модульное программирование и динамическая загрузка.

BlackBox Component Builder - blackboxframework.org

Проблема наследования реализации и BlackBox-решение

```
class Super { private int
   counter = 0;

   void inc1() {

       inc2(); // counter++;

   }

   void inc2() { // final

       counter++;

   }

}

class Sub extends Super {
   void inc2() {

       inc1();

   }

}
```

- Проблема хрупкого базового класса – бесконечная рекурсия при `inc2()`
- Решение – явное указание запрета наследования реализации.

```
RESTRICT –EXTENSIBLE;

PROCEDURE (s: Super) inc2(),
   EXTENSIBLE, NEW;

BEGIN

   INC(s.counter);

END inc2;

PROCEDURE (sb: Sub) inc2();

BEGIN END sub2;
```

Восстановление кода

О необходимости восстановления кода.

SpaceIL авария аппарата «Берешит» 11.04.2019:

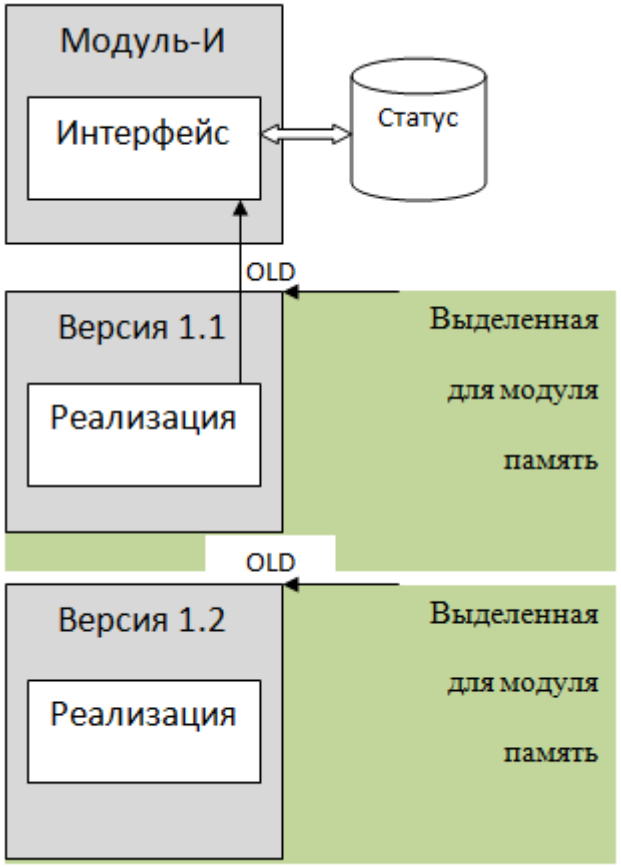
- Larger number of computer resets than expected;
- Several software “patches” were sent to the spacecraft to handle and mitigate problems.

О возможности горячей замены кода.

- «Statelessness makes hot code swapping easy in Erlang(Joe Armstrong)»;
- Без состояния означает отсутствие доступа ~~на запись глобальных переменных и переменных-членов класса~~.
- Должно быть обеспечено начальное состояние объекта при старте (замещение из LSP).

Рестарт модуля и замена кода

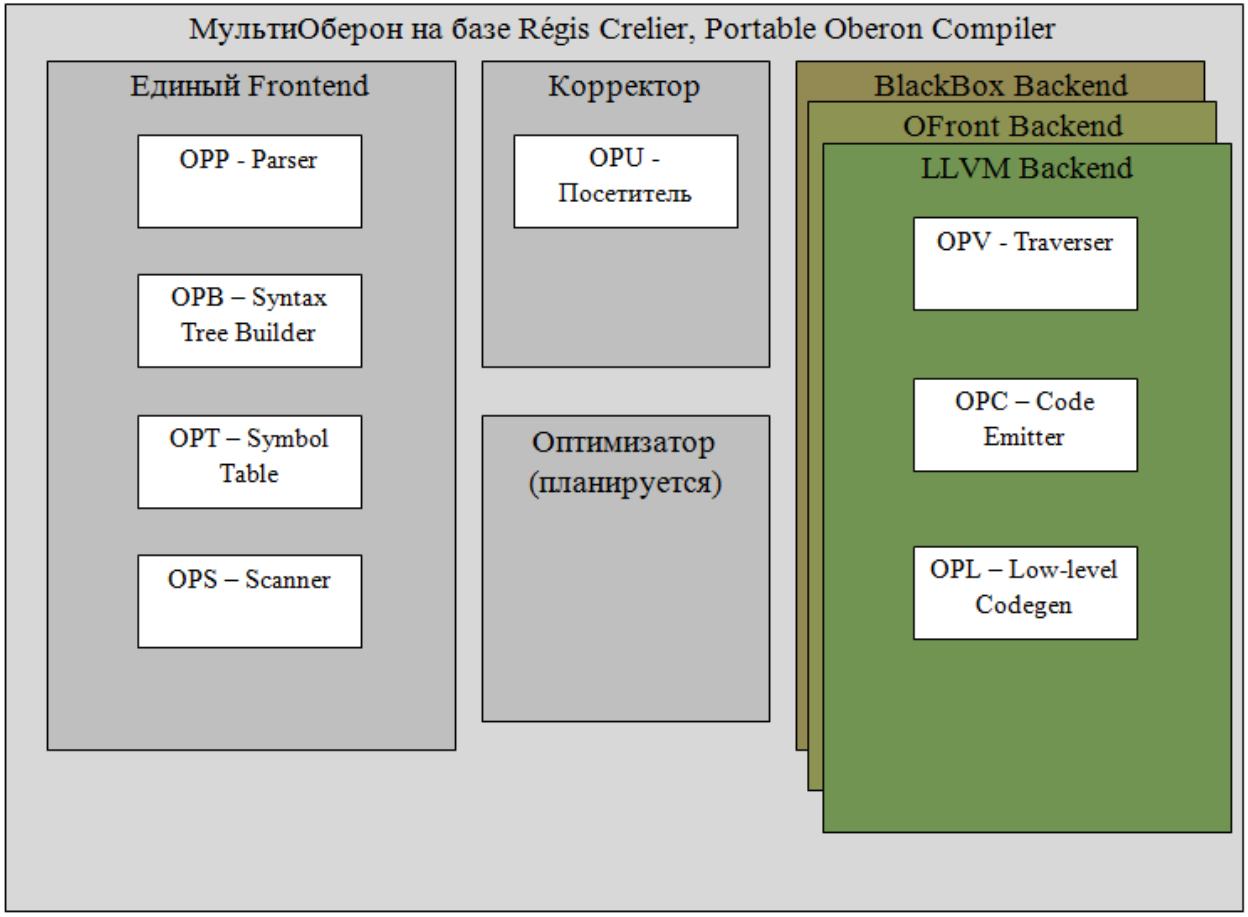
Безопасный рестарт возможен в рамках компонентно-ориентированного ПО.



Ограничения безопасного рестарта программного модуля:

- ~~Нет состояния~~, все состояние реализации выведено в область статуса;
- Нет динамической памяти ~~NEW~~, заменена на внутримодульную, нет фабричных объектов;
- Инициализация глобальных переменных модуля и членов класса осуществляется только в секции инициализации модуля. Нет ~~обновления глобальных переменных и членов класса~~.
- Рестарт программных модулей должен осуществляться на фиксированные адреса в выделенной для модуля памяти.

МультиОберон – решение со сменными backend'ами



Корректор реализован для проверки соответствия ***ограничений*** использования RESTRICT и для контроля ***отсутствия*** ПО в перечне ~~потенциально-опасных конструкций~~.

Масштабирование и многоцелевое использование

Применение	Операционная система	Выходной код
Портабельное ПО широкого применения	Без ограничений	Транслированный код C
Портабельное ПО, требующее высокой степени оптимизации	Многопользовательская ОС с поддержкой POSIX	Код LLVM
Надежное ПО быстрой разработки и динамически изменяемое ПО	Многопользовательская ОС с поддержкой POSIX	Нативный код BlackBox
Встроенное графическое и управляющее ПО категорий A, B	ОС на основе однопользовательской системы A2	Код объектов A2
Встроенное управляющее ПО категории A	Без ОС, цикл от Таймера	Нативный код встроенной системы

При демасштабировании (scale down, движение вниз) сложность системы уменьшается, вместе с этим улучшается надежность и функциональная безопасность.

Принцип подстановки и замещение компиляторов

Формулировка LSP Б.Лисков:

*Пусть **q(x)** является свойством, верным относительно объектов **x** некоторого типа **T**. Тогда **q(y)** также должно быть верным для объектов **y** типа **S**, где **S** является подтипом типа **T**.*

Замещение компиляторов по свойству допустимости входного кода:

*Если (компилятор) **S** является подтипом (компилятора) типа **T** , то возможно замещение компилятора **T** на компилятор **S.***

Отношение подтипов при ***T*** → ***S*** для одной и той же базы программ можно установить в том случае, когда имеется один frontend с переменной системой ***ограничений***.

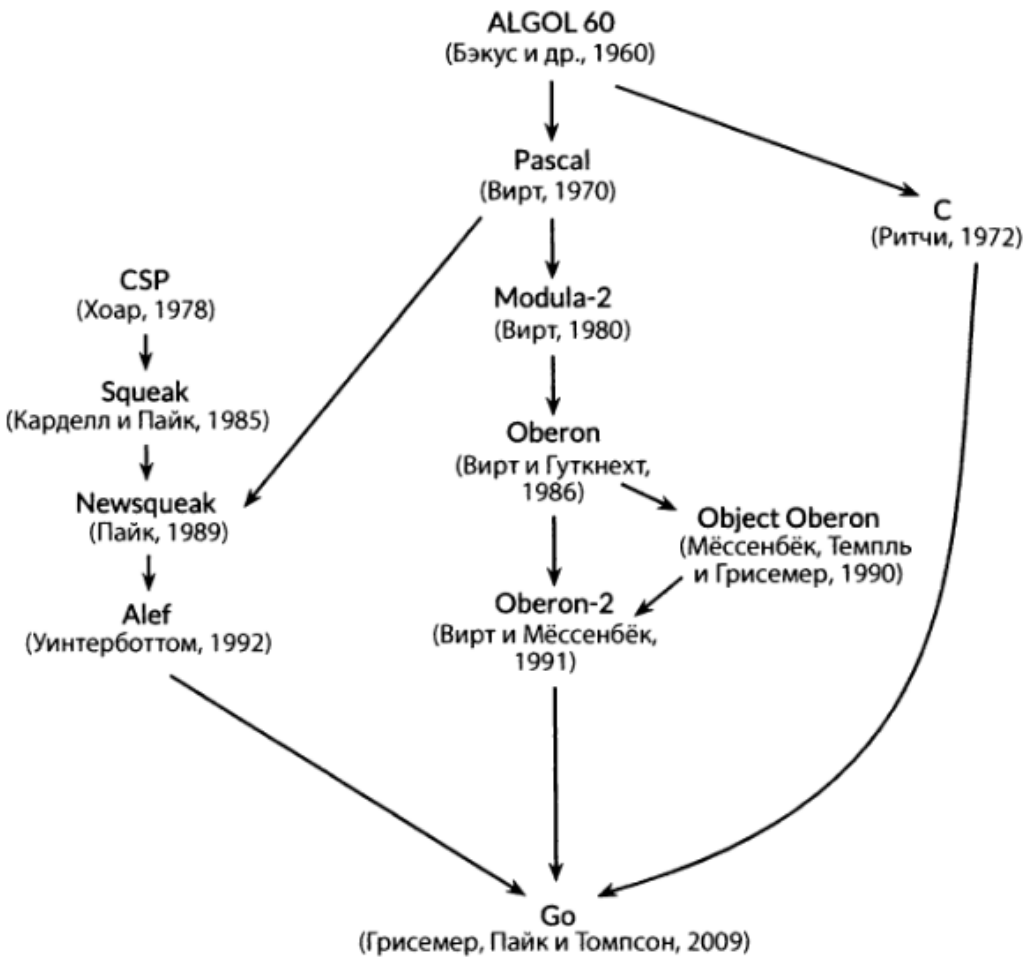
Есть такая IT-организация?

Вопросы по докладу …

Дагаев Дмитрий Викторович,
Главный Эксперт,
АО «Русатом Автоматизированные системы управления»
Консультант проекта Информатика-21
forum.oberoncore.ru
www.inr.ac.ru/~info21/

dvdagaev@mail.ru

Приложение 1 – Генеалогия Golang



Генеалогия Go (из книги А.Донована и Б.Кернигана).

Один из авторов Robert Griesemer является Phd учеником Н.Вирта, работал над компилятором Oberon-V для векторных компьютеров.