Rathaxes

A DSL for Driver Generation

Today’s specials

- Why ?
- How is it possible ?
- Does it work ?
- How is it working ?
- What’s next ?

Why Rathaxes?

Driver development is :

- 7 time more crash prone
- Requires a double knowledge
- Lengthy training on each OS
- Time consuming

Why Rathaxes?

70 %

Of Operating Systems code

Why Rathaxes?

- Time consuming
- Critical part of a system
- Must be cross platform
- Needed by the operating system

How is it possible?

- Generating Drivers : utopia?

  → LibOs, Devil, NDL, Rosetta....

- Focus on shared concepts

- Abstract Operating System differences

How is it possible?

Driver Anatomy

DEVICE DEPENDANT

ALGORITHMS

REGISTERS

OS DEPENDANT

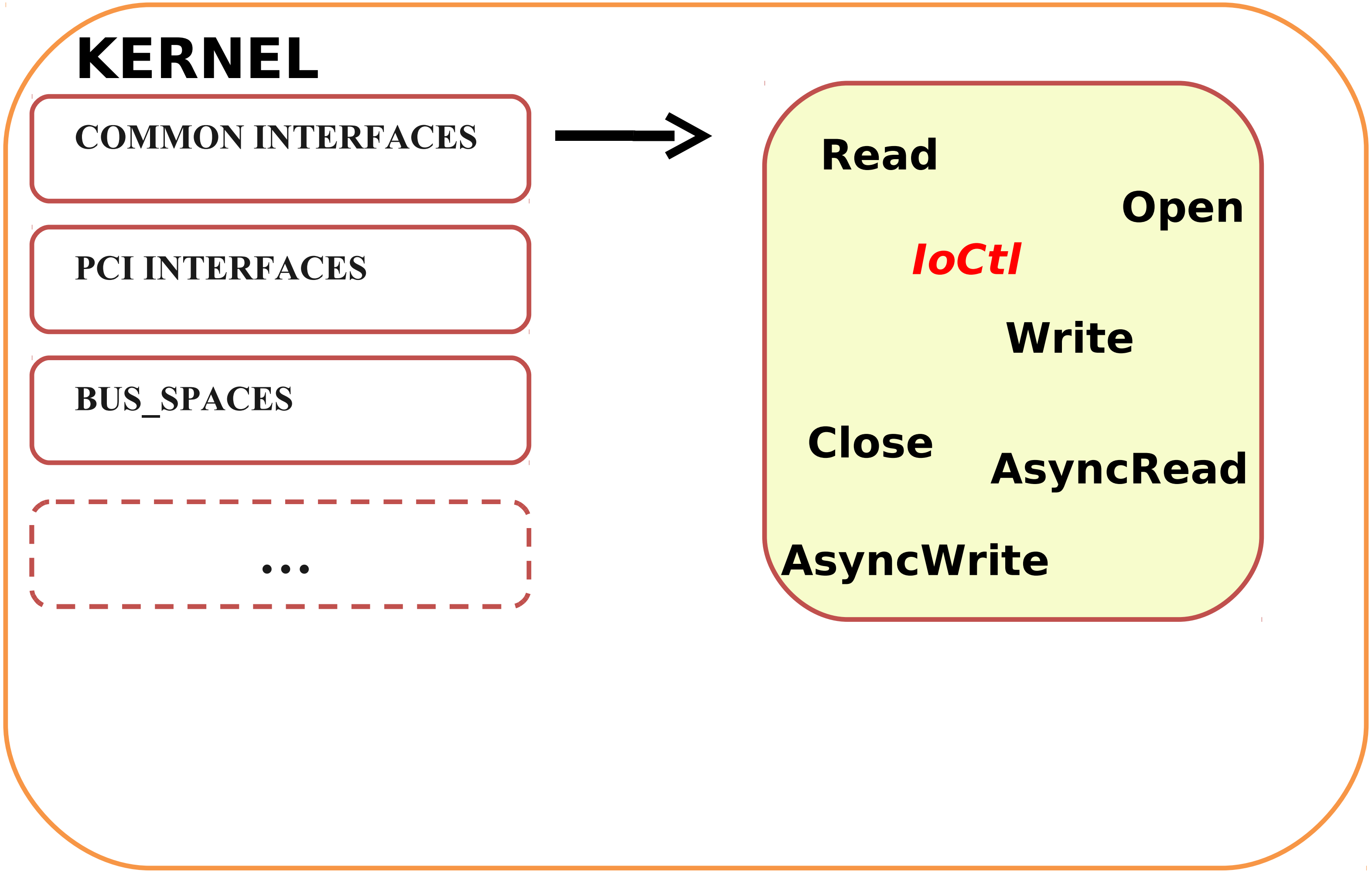
KERNEL INTERFACES LIBRARY/BUS

CONFIGURATION

LKM

How is it possible?

- OS Dependant concepts



KERNEL

COMMON INTERFACES

PCI INTERFACES

BUS_SPACES

...

Read

Open

IoCtl

Write

Close

AsyncRead

AsyncWrite

How is it possible ?

Device Dependant concepts

Standard Algorithms:

→ WAIT(Register, value)

→ SET(Register|variable, Register|variable|value)

→ COPY(Buffer, Register|Buffer)
PRE/POST/INVARIANT {Algoritms}

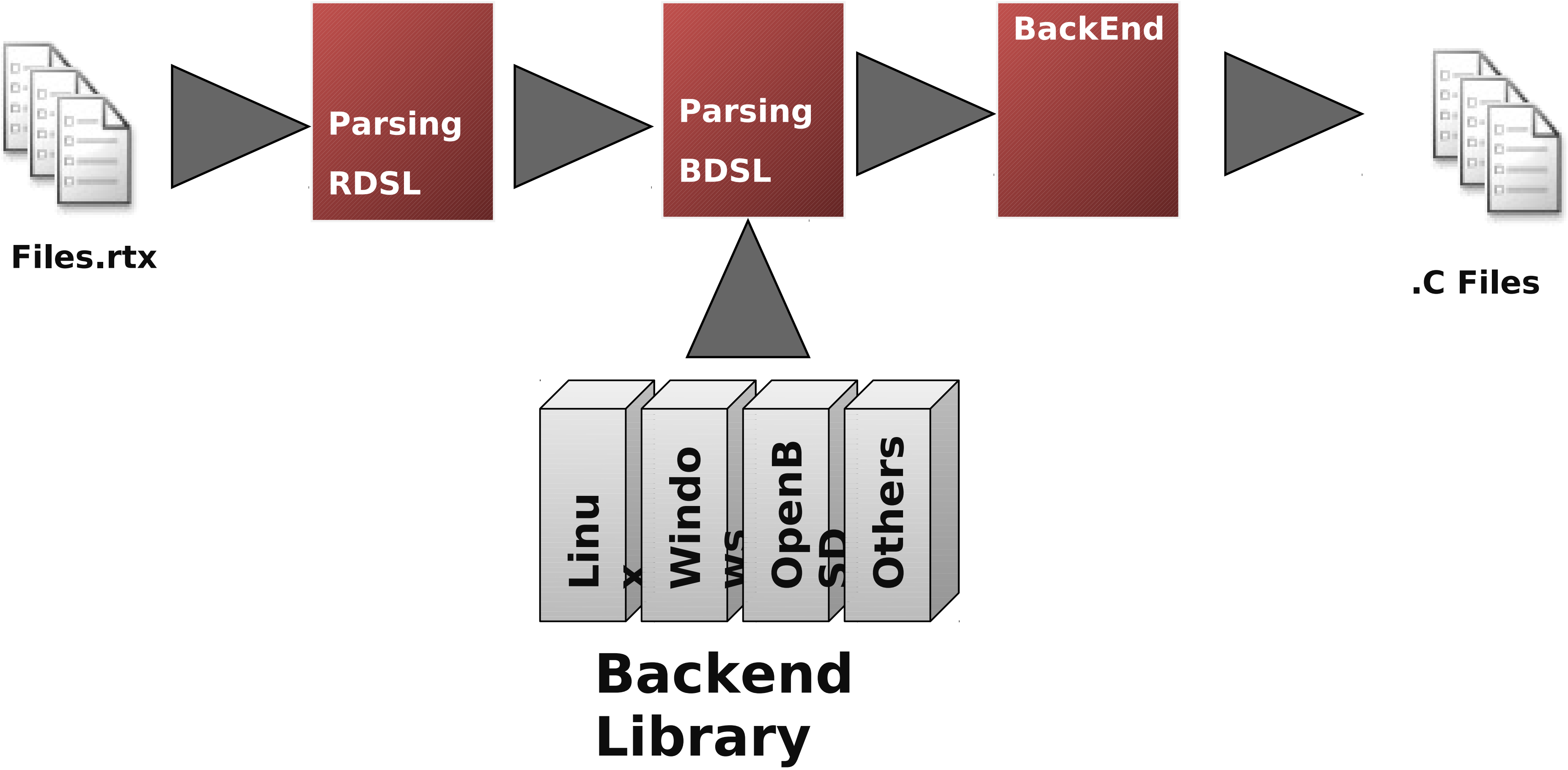
→ CONCAT(Buffer, Register|Buffer)
PRE/POST/INVARIANT {Algoritms}

Does it work ?

How is it working ?

- Domain Specific Language
- Compiler
- Backend Library

How is it working ?



Files.rtx

Parsing

RDSL

Parsing

BDSL

BackEnd

.C Files

Linu
x

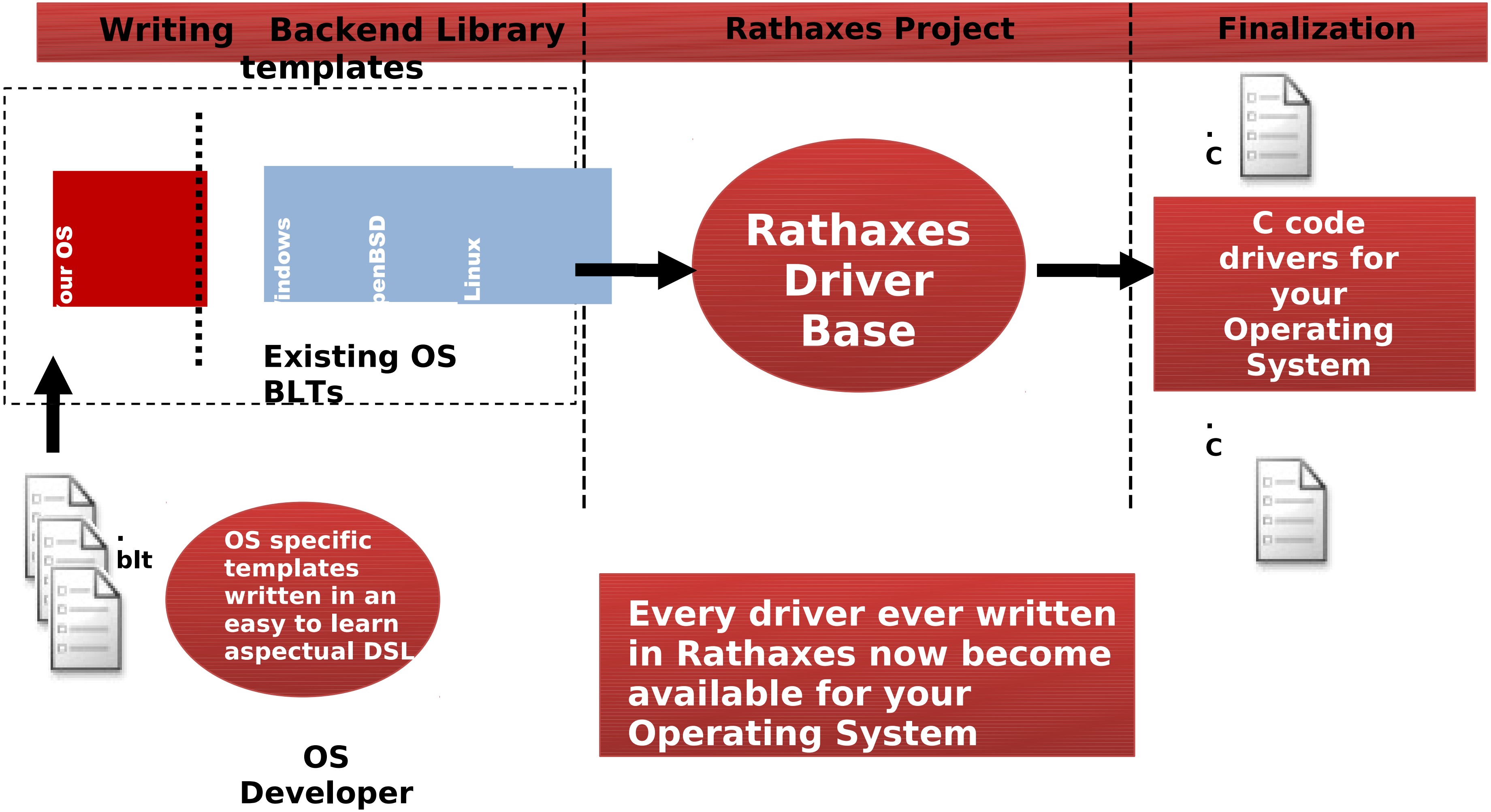
Windo
ws

OpenB
SD

Others

Backend
Library

How is it working ?



Writing Backend Library templates

Rathaxes Project

Finalization

Existing OS BLTs

Rathaxes Driver Base

.c

C code drivers for your Operating System

.c

.blt

OS specific templates written in an easy to learn aspectual DSL

OS Developer

Every driver ever written in Rathaxes now become available for your Operating System

How is it working ?

- BLT language

```
ADVICE register PART_OF registers
{
    struct test_struct
    {};
    < addElement(« int foo », « test_struct »)  >
    < addElement(« struct bar », « test_struct ») >

    JOINPOINT values IN registers.values;

    < addDeclaration(« struct $register.name »)>
    < foreach node in register.members>
       <addMember(« int $node.name : $node.size », $register.name);>
    < foreach >
};
```

What’s next ?

- Expend RTX semantics:
  - (BUS, DMA, algorithm)
- Expend BLT semantics:
  - A real aspectual C language
- BLT checking:
  - Help blt developer
- More samples of drivers
- More kernels BLT

Conclusion

Rathaxes is for :

- Driver Devs
- OS devs
- language enthusiasts (O.o)

Our tool :

- Opensource Compiler Generator
- Free Fast and reliable

→ www.codeworker.org

Questions

contact@rathaxes.org

rathaxespublic@googlegroups.com

www.rathaxes.org

LSE Security System

Laboratory of {Epita, Epitech}

{EPITECH INNOVATIVE PROJECTS}

{EPITECH. L'EXPERTISE INFORMATIQUE}