A DSL for Video Device Drivers: from Design to Implementation

Scott Thibault, Renaud Marlet, Charles Consel

IRISA / INRIA - Université de Rennes 1

GPL vs. DSL

GPL

- Low-level
- Programming
- All programs
- Compiler

DSL

- High-level
- Instantiation
- Small set of programs
- Application generator

Program Family

Set of applications having enough commonalities to be studied as a group.

DSL: pros & cons

Features

- Small high-level programs
- Reuse: design & implementation

Advantages

- Productivity
- Prototyping
- Debugging
- Maintenance
- Analyzability
- Mobility
- Usability

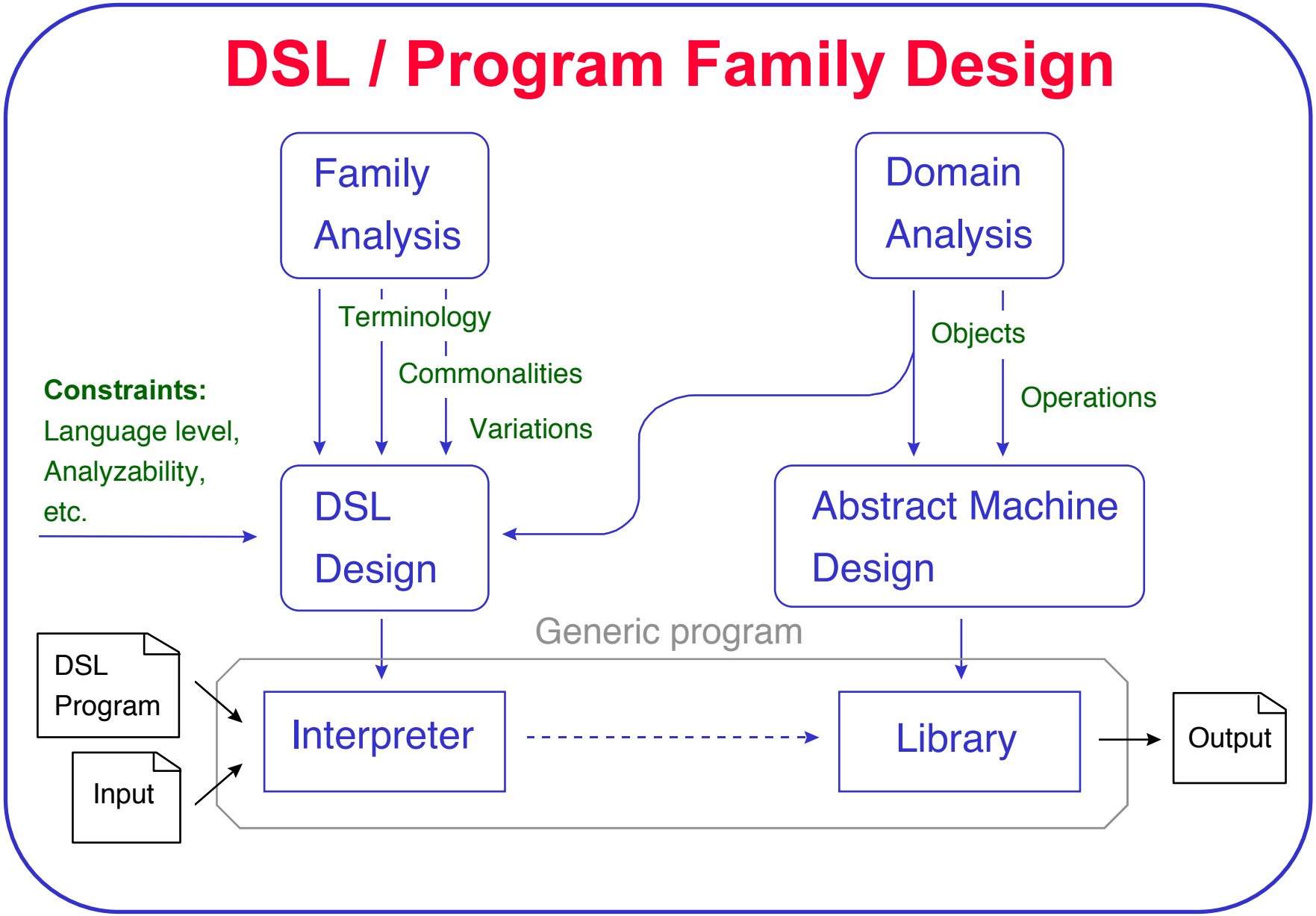
Drawbacks

- Compiler development
- Performance
- Compiler maintenance
- Limited functionality
- Compiler extension
- Standardization
- Training

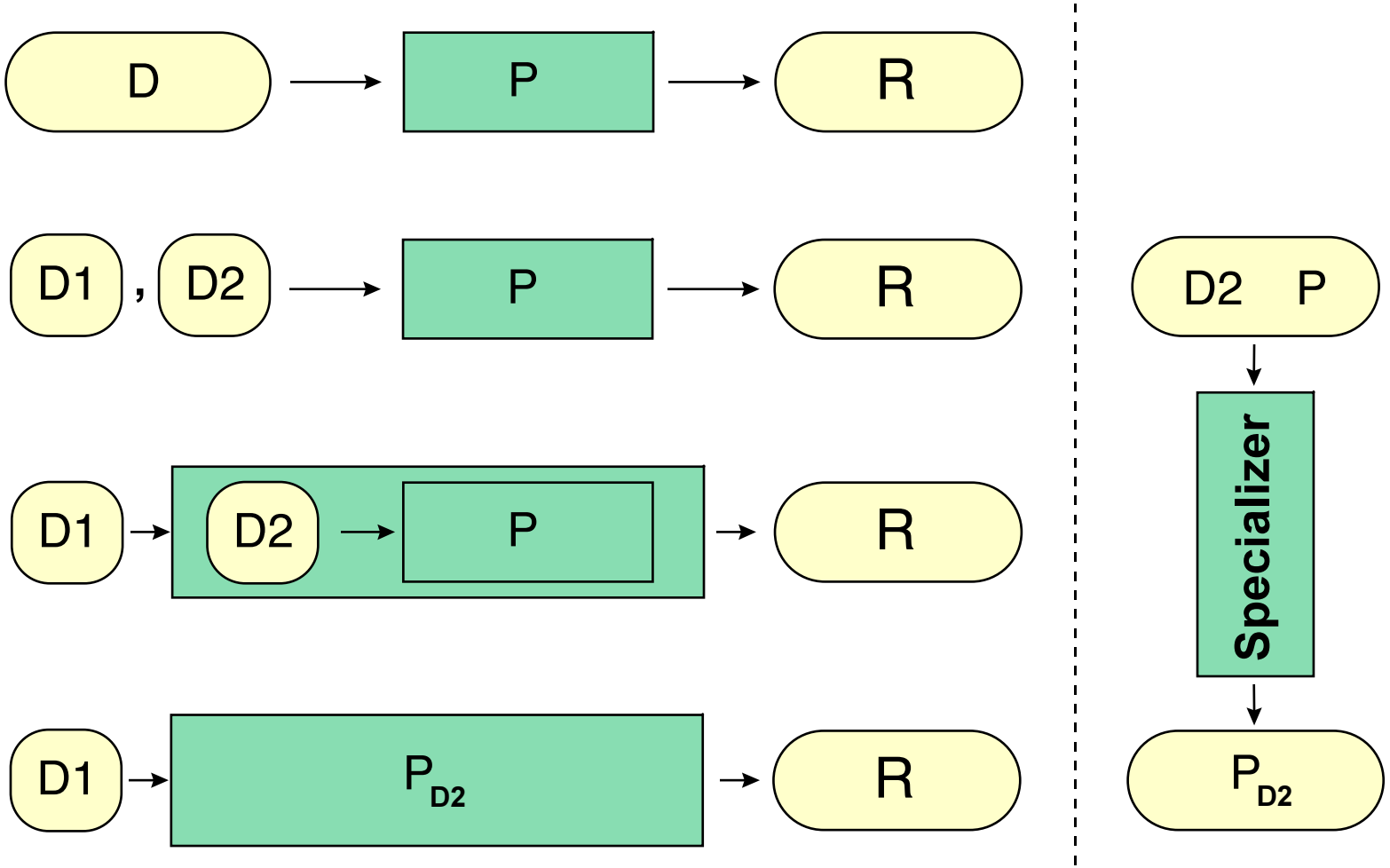
Contents

- Flexible framework of program family design
- Efficient implementation via partial evaluation
- Application: GAL, a DSL for video device drivers
- Assessment of GAL's pros and cons

DSL / Program Family Design



Partial Evaluation



Partial Evaluation Example

```
mini_printf(char fmt[], int val[])
{
  int i = 0;
  for(; *fmt; fmt++) {
    switch(*fmt) {
    case '%' :
      switch(*++fmt) {
      case '%' :
        putchar('%'); break;
      case 'd' :
        putint(val[i++]); break;
      default :
        abort(); break;
      } break;
    default :
      putchar(*fmt);
    }
  }
}
```

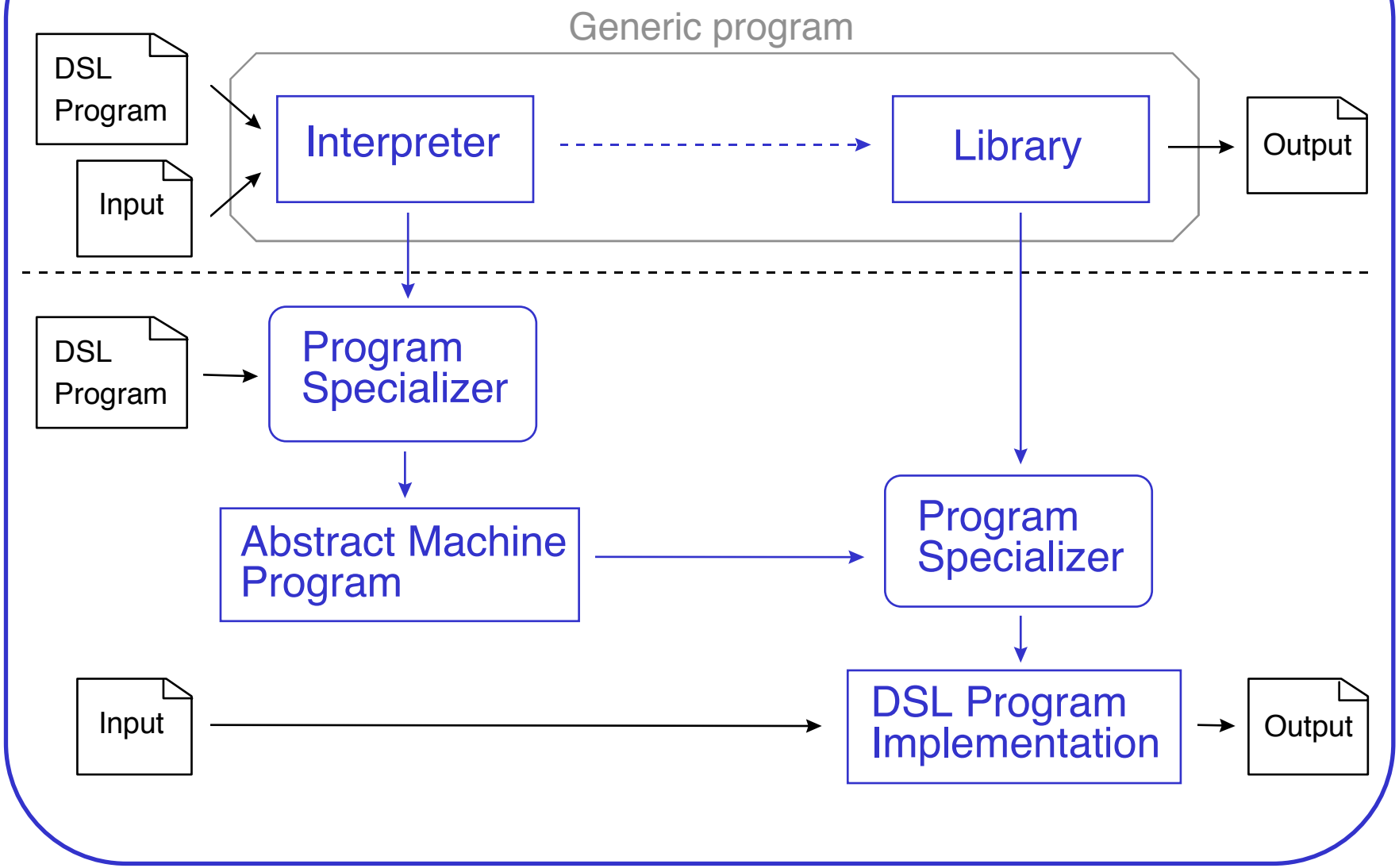
Specialized `mini_printf` with respect to `fmt = "n: %d"`:

```
mini_printf_fmt(int val[])
{
  putchar('n');
  putchar(':');
  putchar(' ');
  putint(val[0]);
}
```

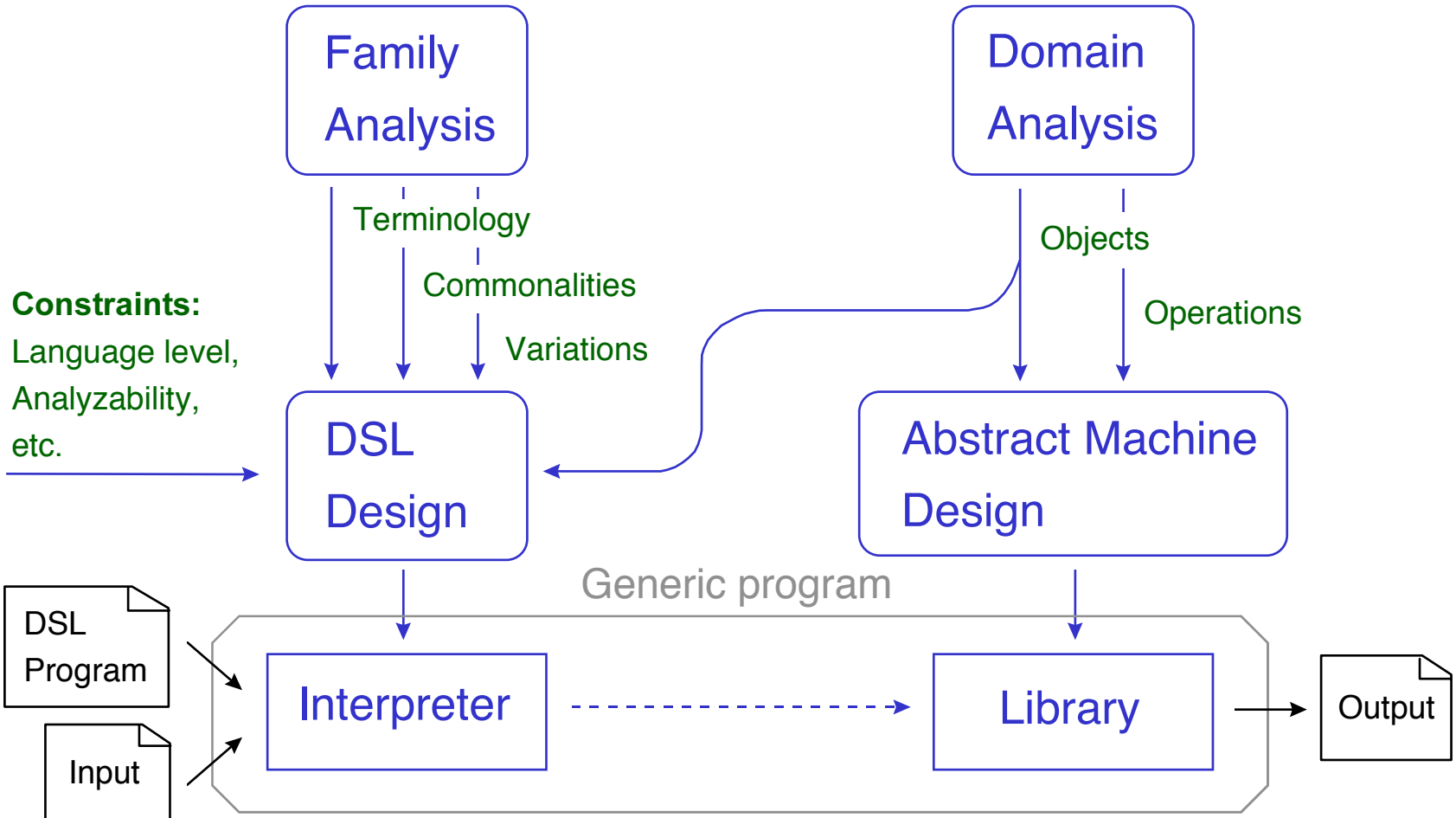
Some Issues in Partial Evaluation

- Speed
- Code size
- Termination
- Degree of specialization
- Use of specialized code

Instantiation Process



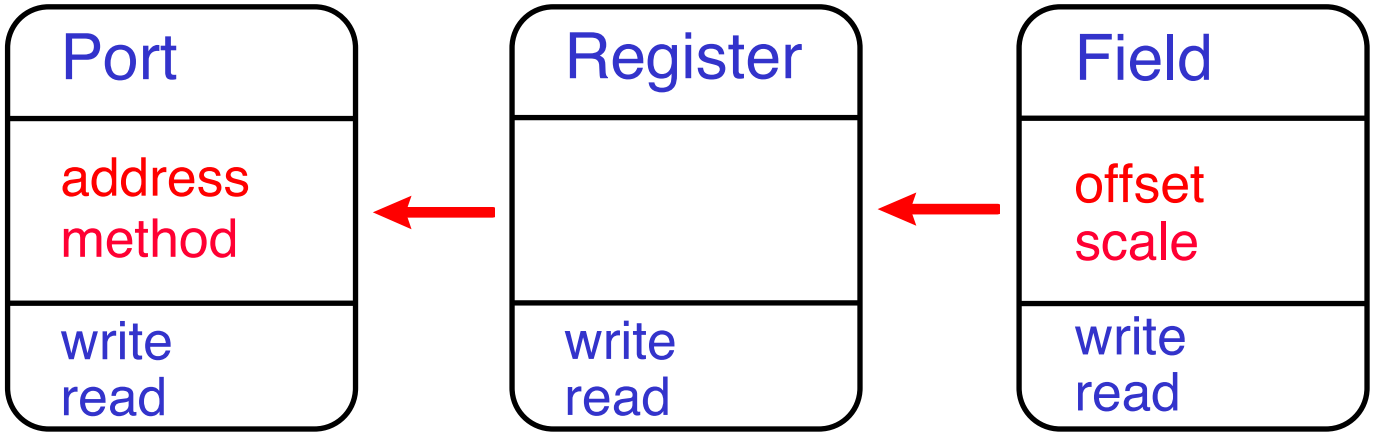
DSL / Program Family Design



Design of GAL / Abstract Machine

- Documentation of existing video cards
  - Objects / Terminology
  - Attributes related to variations
  - Declarative statements to specify attributes
- Inspection of existing device drivers
- Level of abstraction
  - No bitwise operations
  - Draw the line between the interpreter and the AM
- Level of restriction
  - No loops
  - Detection of common errors

Video Device Driver Domain




```
port svga indexed:=0x3d4;

register Offset:=svga(0x13);

field LogicalWidth:=Control2[5..4]#Pitch scaled 8
```

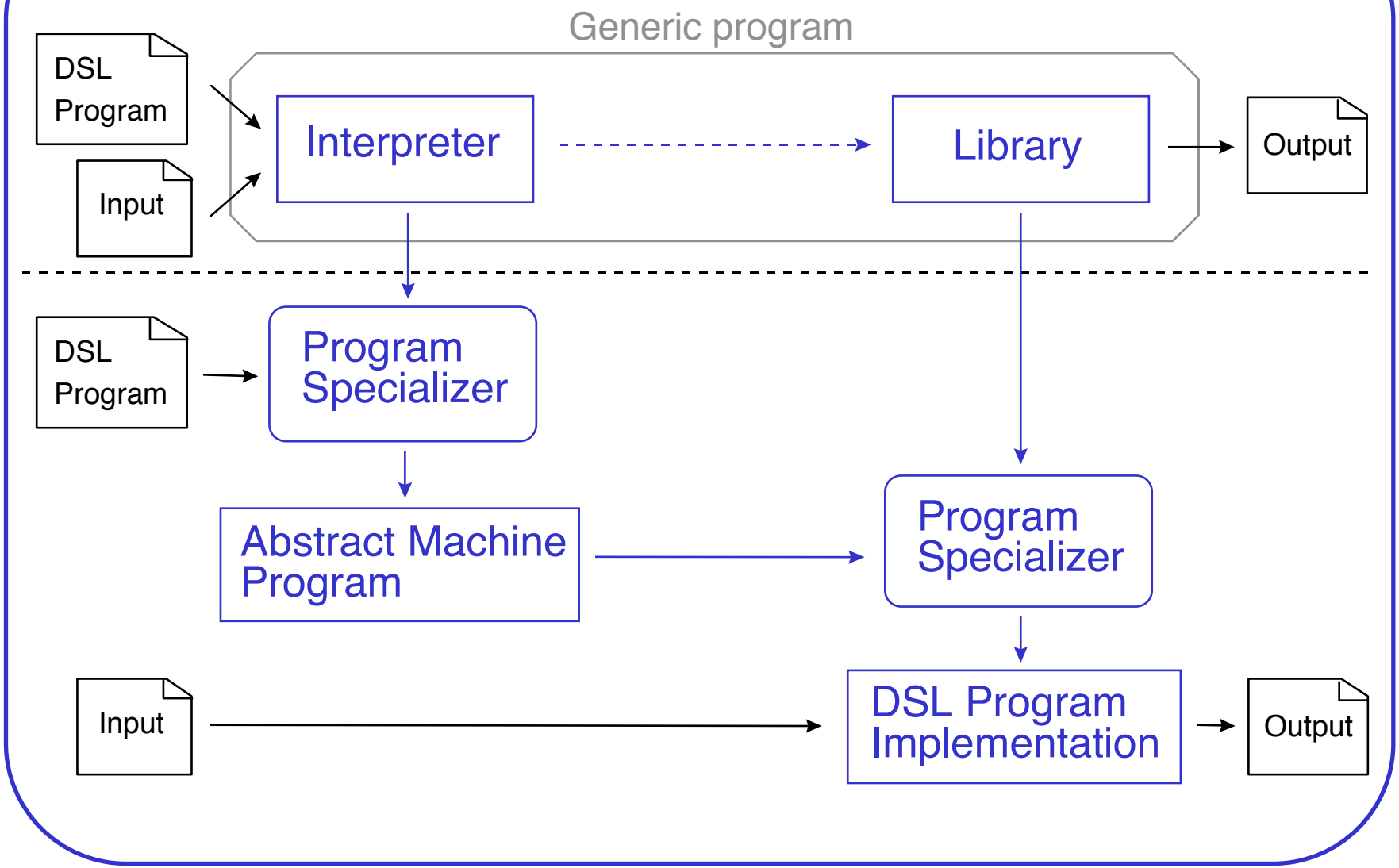
DSL Benefits Realized

- **Increased level of abstraction**
  - Less errors, more readable, easier to write
- **Design re-use**
  - Knowledge about communications with the card
- **Productivity gains**
  - Drivers 9 times smaller
- **Automatic analysis**
  - Verification, documentation

Example Analysis

```
Profile of S3_TRIO32 + S3_TRIO64

Maximum resolution: 4088x2047

Max. virtual screen with max. RAM: 3328x2520

Ram size: 512k-8192k

Clock range: 135-270MHz

Resolution limited to 2304x1728 by clock
(max. refresh 67Hz)
```

Instantiation Process

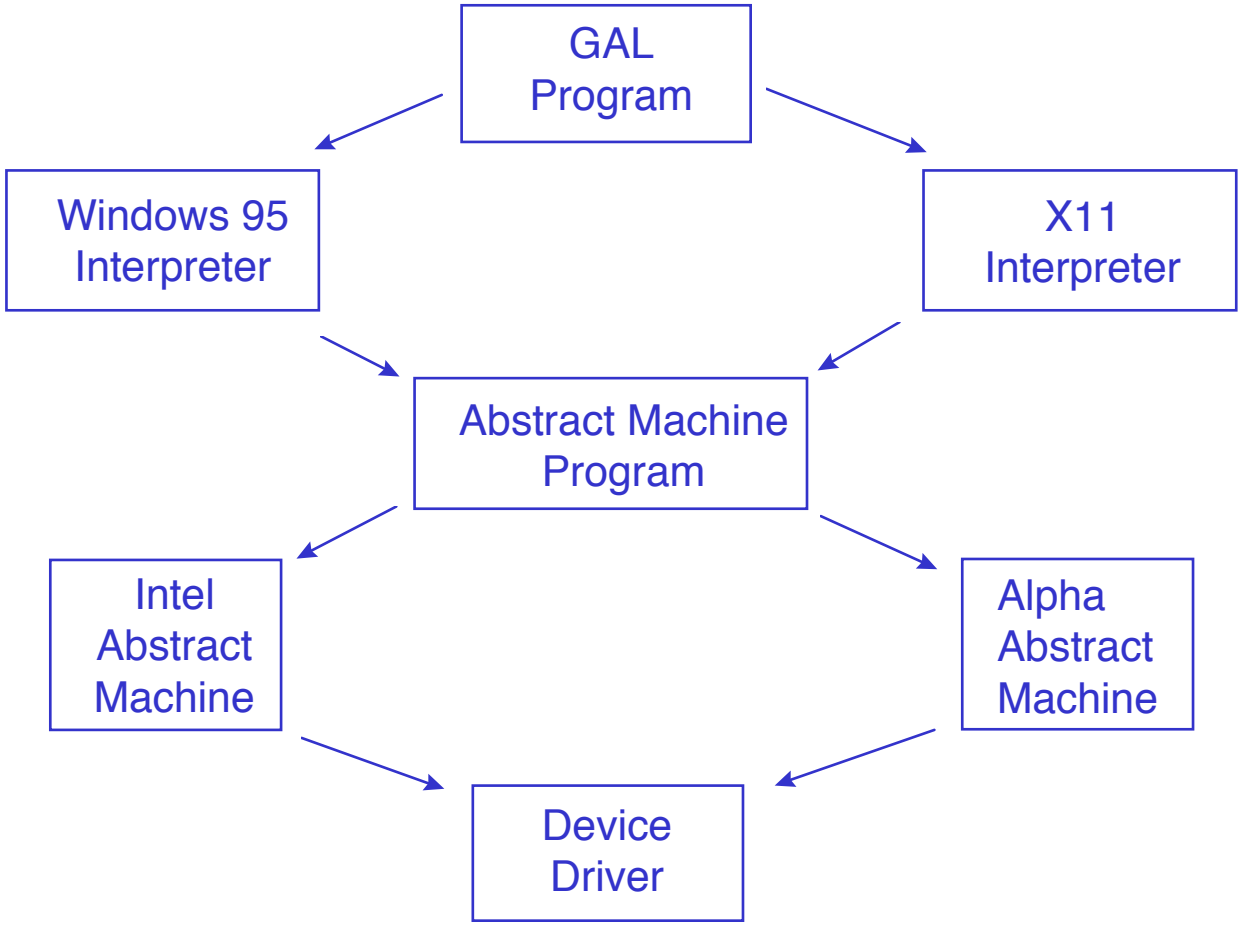


Achieved Benefits of the Framework

Application of Tempo, a partial evaluator for C

- **Reduced implementation costs**
  - Interpreter vs. compiler
- **Comparable performance**
  - Interpreter 10-100 times slower
  - AM 20% slower
  - Fully specialized driver ~ hand coded driver
- **Further possibilities of analysis**
  - Analysis of generation process
  - Analysis on AM programs
- **Multiple implementations at two levels**

Multiple Implementations



GAL: assessment

Features

- ✔ Small high-level programs
- ✔ Reuse: design & implementation

Advantages

- ✔ Productivity
- Prototyping
- ✔ Debugging
- Maintenance
- ✔ Analyzability
- Mobility
- ✔ Usability

Drawbacks

- ♥ Compiler development
- ♥ Performance
- Compiler maintenance
- ♣ Limited functionality
- ♥ Compiler evolution
- ✔ Standardization
- ♣ Training

Future Work

- Components for implementing interpreters & AM
- Techniques for language extension / composition
- Techniques for analysis extension / composition
- More applications

COMPOSE Home Page (and access to GAL):

http://www.irisa.fr/compose