Boost.Proto
C++ Embedded Domain Specific Languages Made Easy

metascale
Software tools for high-performance computing

Joel Falcou

LRI, University Paris Sud XI

31/03/2011

Who am I

Part researcher ...

- Focus on C++ for HPC
- Application of Generic/Generative Programming
- Daily abuses and tortures compilers
- also "Head" of the French C++ UG

Who am I

Part researcher ...

- Focus on C++ for HPC
- Application of Generic/Generative Programming
- Daily abuses and tortures compilers
- also "Head" of the French C++ UG

Part entrepreneur ...

- CTO of MetaScale SAS
- Boost.SIMD, NT2
- Metaphore: high performance Matlab compiler

Context

In Application Development ...

- there is *Application*

Context

In Application Development ...

- there is *Application*
  - Design is domain driven
  - Users $\neq$ Developers
  - Users are reluctant to changes

Context

In Application Development ...

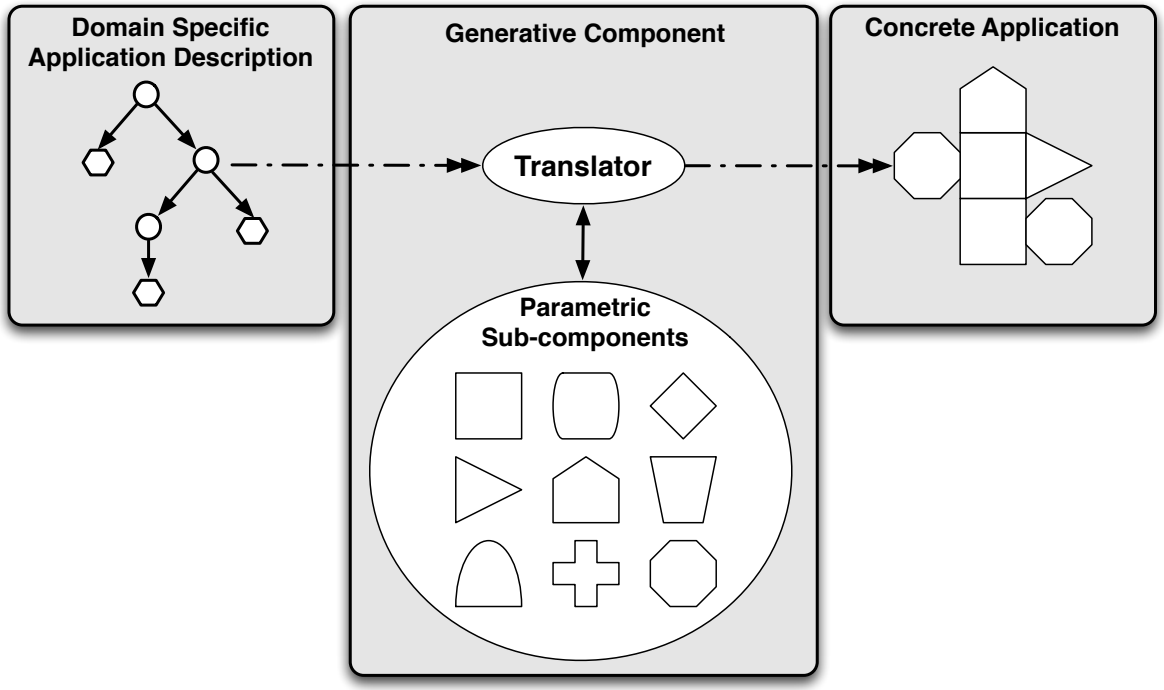
- there is *Application*
  - Design is domain driven
  - Users $\neq$ Developers
  - Users are reluctant to changes
- there is *Development*

Context

In Application Development ...

- there is *Application*
  - Design is domain driven
  - Users $\neq$ Developers
  - Users are reluctant to changes
- there is *Development*
  - Development requires meeting constraints ...
  - ... which implies specific tuning
  - ... which requires expertise
  - ... which may or may not be available

Generative Programming



Generative Programming as a Tool

Available techniques

- Dedicated compilers
- External pre-processing tools
- Languages supporting meta-programming

Generative Programming as a Tool

Available techniques

- Dedicated compilers
- External pre-processing tools
- Languages supporting meta-programming

Generative Programming as a Tool

Available techniques

- Dedicated compilers
- External pre-processing tools
- Languages supporting meta-programming

Definition of Meta-programming

Meta-programming is the writing of computer programs that analyse, transform and generate other programs (or themselves) as their data.

From Generative to Meta-programming

Meta-programmable languages

- `template` Haskell
- `metaOcaml`
- C++

From Generative to Meta-programming

Meta-programmable languages

- `template` HASKELL
- `metaOcaml`
- C++

From Generative to Meta-programming

Meta-programmable languages

- `template` HASKELL
- `metaOcaml`
- C++

C++ meta-programming

- Relies on the C++ `template` sub-language
- Handles types and integral constants at compile-time
- Proved to be Turing-complete

Embedded Domain Specific Languages

What's an EDSL ?

- DSL = Domain Specific Language
- Declarative language, easy-to-use, fitting the domain
- EDSL = DSL within a general purpose language

EDSL in C++

- Relies on operator overload abuse (Expression Templates)
- Carry semantic information around code fragment
- Generic implementation become self-aware of optimizations

Exploiting static AST

- At the expression level: code generation
- At the function level: inter-procedural optimization

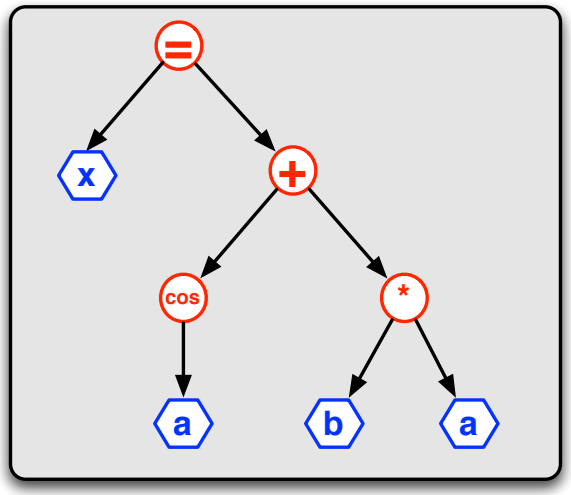
Expression Templates

```
matrix x(h,w),a(h,w),b(h,w);

x = cos(a) + (b*a);
```

```
expr<assign
    ,expr<matrix&>
    ,expr<plus
          , expr<cos
                 ,expr<matrix&>
                 >
          , expr<multiplies
                 ,expr<matrix&>
                 ,expr<matrix&>
                 >
         >(x,a,b);
```

Arbitrary Transforms applied on the meta-AST




```
#pragma omp parallel for
for(int j=0;j<h;++j)
{
  for(int i=0;i<w;++i)
  {
    x(j,i) = cos(a(j,i))
           + (  b(j,i)
              * a(j,i)
           );
  }
}
```