# FoCDoc: The Documentation System of FoC

Manuel Maarek[1,2]
Virgile Prevosto[1,3]

[1] SPI – LIP6 – University Paris 6

[2] ULTRA – MACS – Heriot-Watt University

[3] Projet Moscova – INRIA

## Mathematics on Computers

- Representation of data
- Performing calculations
- Automated reasoning
- Interaction with the users
- Interaction with other systems

# The Computer Algebra System FoC

**Formel Ocaml Coq**

- Specification

  FoC language

- Implementation

  Compilation to OCaml

- Proofs of correctness

  Compilation to Coq

# The FoC language

| | | |
|---|---|---|
| `species` | abstract | **Mathematical structures** |
| `collection` | implemented | |
| | | **Components** |
| `rep` | abstract or defined | carrier type |
| `sig` | declared | functions |
| `let` | defined | |
| `property` | stated (and proved) | properties |
| `theorem` | stated and proved | |
| inheritance | | **Constructions** |
| parametrization | | |

# Integration of FoC in a Mathematical Network

## Data output

- **Sharing the content of a FoC program**

  Documentation system

- **Presenting FoC computation results**

  Output functions
  for standard representation

## Data input

- **Integration of external data into a FoC program**
  - Using syntax extensions
  - Sharing a common interface

# A documentation system for FoC

## Goals

- **The library** — **Browsing**
- **Mathematical knowledge** — **Sharing**
- **Publication** — **Rendering**

first attempt: **OMDoc**

# Open Mathematical Document[a]

- Formula level

  uses OpenMath

- Statement level

  defines components

- Theory level

  describes structures

- Meta information

[a] http://www.mathweb.org/omdoc/

# Choosing OMDoc

## Experience

Systems interfaced with OMDoc – existing OMDoc documents

## Development

MathWeb project

MBase – MathWeb Software Bus

## Open specifications

Reliability – evolution

# A FoC program

```
species setoid =
 rep;
 sig eq in self -> self -> bool;
 let neq (x,y) = #not_b(self!eq(x,y));
 property eq_refl: all x in self, self!eq(x,x);
 theorem neq_nrefl: all x in self, not(self!neq(x,x))
    proof: ...
end
```

## A translation in OMDoc

```
<theory id="setoid">
  <symbol id="rep" kind="type" scope="local">
  <symbol id="eq" kind="object" scope="global">
    <type system="FoC">(self)-> (self)-> bool
  <symbol id="neq" kind="object" scope="global">
    <type system="FoC">(self)-> (self)-> bool
  <definition for="neq" type="simple"> ...
  <assertion id="eq_refl" type="proposition">
    <FMP>∀x ∈ self, x = x
  <assertion id="neq_nrefl" type="theorem">
    <FMP>∀x ∈ self, ¬(x ≠ x)
  <proof for="neq_nrefl"> ...
```

# Relations between FoC and OMDoc

Structures – Components – Constructions

| FoC | OMDoc |
|---|---|
| `species` and `collection` | `<theory>` |
| `rep` | `<symbol>` |
| `sig` and `let` | `<symbol>` (`<definition>`) |
| `property` and `theorem` | `<assertion>` (`<proof>`) |
| `inherits` and `implements` | `<imports>` |
| parameter | `<symbol>` |
| inheritance with parameters | `<imports>` with `<morphism>` |

# Simple inheritance

```
species setoid =                <theory id="setoid">
 ...
end

species monoid                  <theory id="monoid">
  inherits setoid =               <imports from="setoid">
 ...
end
```

## Parametrization by a value

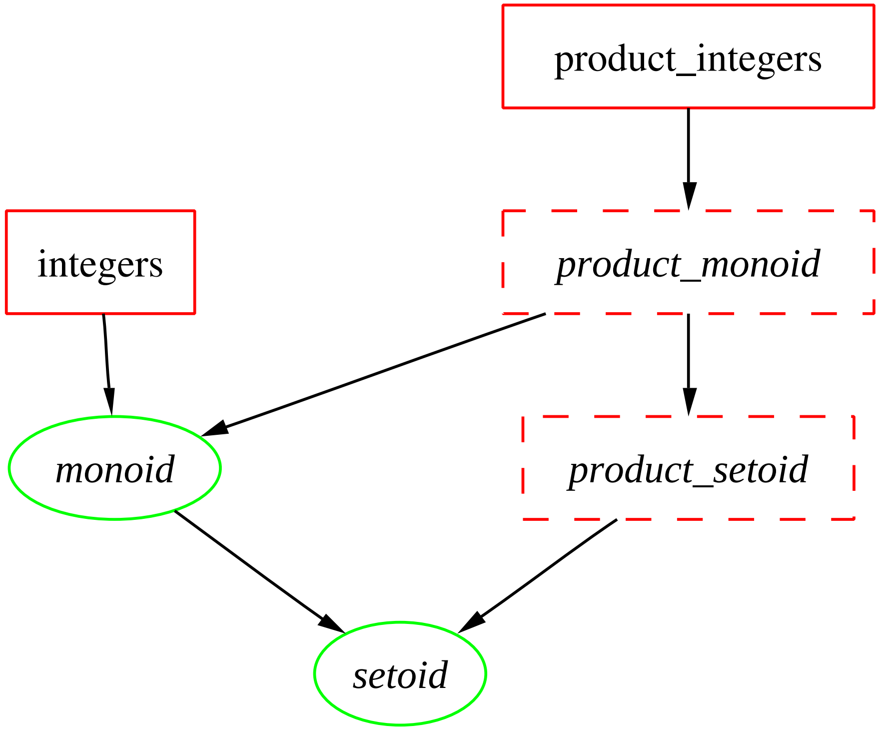
```
species modulo (n in int)     <theory id="modulo">
  inherits ring =               <imports from="ring">
 ...                            <symbol id="n">
end                               <type system="FoC"> int

species modulo2               <theory id="modulo2">
  inherits modulo(2) =          <imports from="modulo">
 ...                              <morphism>
end                                 <requation>
                                      <pattern> n
                                      <value> 2
```

# Parametrization by a collection

```
species fractions
  (a is gcd_domain) =
 ...
end

collection rationals
  implements
    fractions(integers) =
 ...
end
```

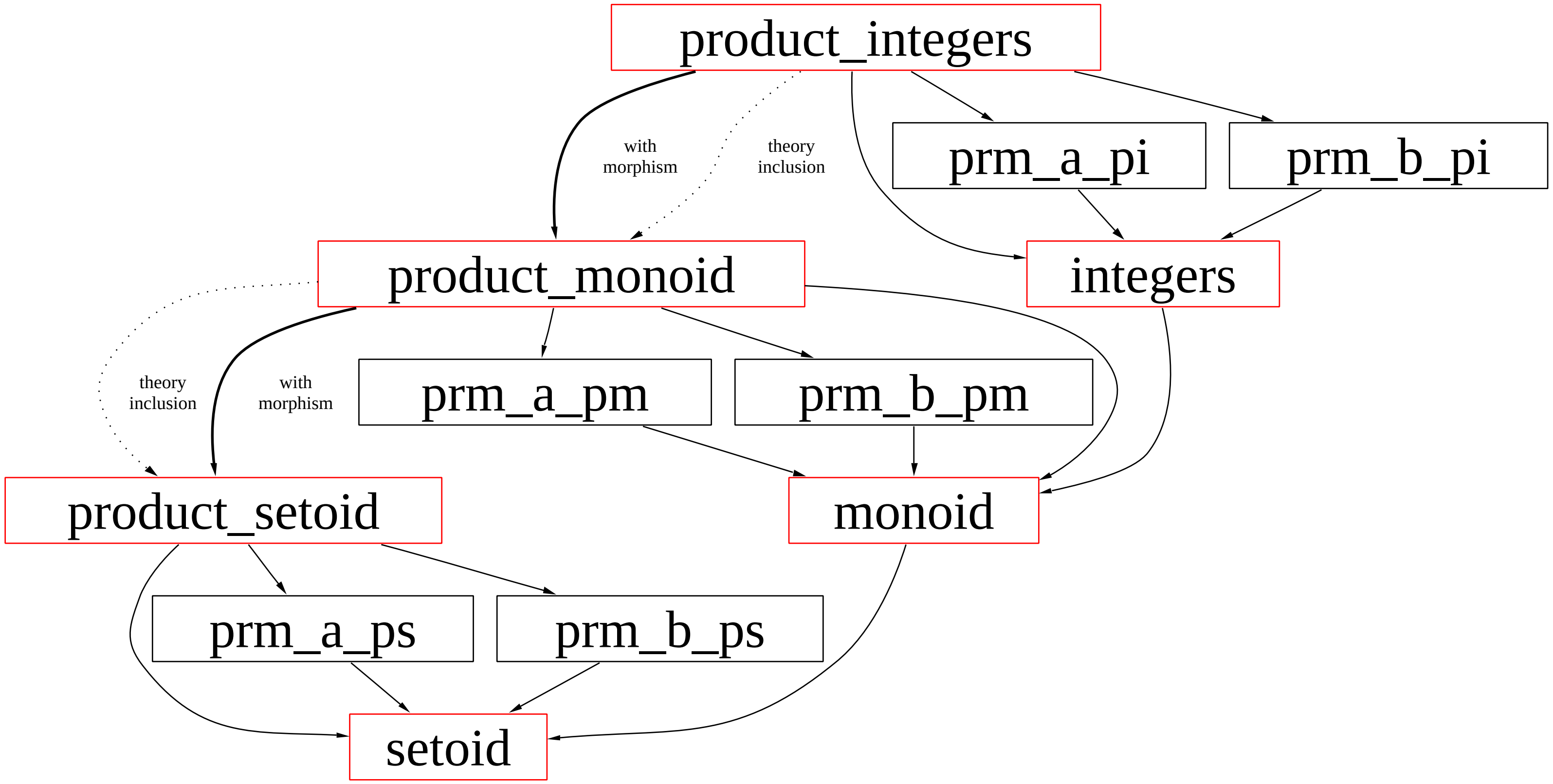
```
<theory id="fractions">
  <imports from="gcd_domain">

<theory id="rationals">
  <imports from="integers">
  <imports from="fractions">
    <morphism> <requation>
      renaming of each gcd_domain's
      symbol into integers'

<theory-inclusion from="fractions"
                  to="rationals">
  <morphism base="gcd_dom-int-morph">
```

# Inheritance of parametrized species




```
species product_setoid (a is setoid, b is setoid)
  inherits setoid =
 ...
end
...
```

# Actualization



# Limits

## Usability

The distinction between abstract species and implemented collection is lost.

## Expressivity

Inheritance of parametrized species
and OMDoc actualization

## Solutions

- Add specific FoC information in the OMDoc document

**Extending OMDoc**

- Use our own documentation language

**FoCDoc**

# FoCDoc

FoC program

- FoC code
- Structured comments
- Types and dependencies from compilation

$$\downarrow$$

FoCDoc (XML syntax)

$$\downarrow$$

HTML/MathML

OMDoc/OpenMath

LaTeX

# Integration of external data

# Syntax extensions

```
let five =
  <OMI>5</OMI>;;
let pair =
  <OMOBJ>
    <OMA>
      <OMS cd="cart" name="pair"/>
      <OMV name= "#five"/>
      <OMV name= "#five"/>
    </OMA>
  </OMOBJ>;;
```

# Sharing a common interface

## Preparation

Defining an interface for polynomial with

- an OMDoc theory `poly`
- an abstract FoC species `poly`

sharing same symbols.

```
<theory id="poly">
  <symbol id="add">
  <symbol id="monomial">
  <symbol id="constant">
  <symbol id="mult">
```

# Sharing a common interface

## Use in FoC

- Choosing the representation for polynomials
- Loading polynomials in the FoC program

```
change_rep(poly, my_poly);;
let p = <OMOBJ> <OMA>
          <OMS cd="poly" name="add"/>
          <OMA>
            <OMS cd="poly" name="monomial"/>
            <OMSTR>X</OMSTR> <OMI>2</OMI>
          </OMA>
          <OMA>
           <OMS cd="poly" name="constant"/>
           <OMI>5</OMI>
          </OMA>
        </OMA> </OMOBJ>;;
```

## Conclusion – Future work

- **Front-end for the FoC user**
- **Future integration in MathWeb**
- **Documentation of the FoC library**