



**HAL**  
open science

## Cast3M, a problem-solving software for structural and fluid analysis

Stephane Gounand, Pierre Verpeaux, Serge Pascal, François Di Paola,  
Clément Berthinier

► **To cite this version:**

Stephane Gounand, Pierre Verpeaux, Serge Pascal, François Di Paola, Clément Berthinier. Cast3M, a problem-solving software for structural and fluid analysis. ECCOMAS 2024 - 9th European Congress on Computational Methods in Applied Sciences and Engineering, Jun 2024, Lisbonne, Portugal. hal-04609241

**HAL Id: hal-04609241**

**<https://hal.science/hal-04609241>**

Submitted on 12 Jun 2024

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution - NonCommercial - NoDerivatives 4.0 International License

# Cast3M, a problem-solving software for structural and fluid analysis

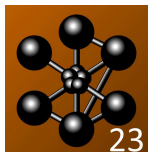
Stéphane GOUNAND<sup>1</sup>, Pierre VERPEAUX<sup>2</sup>, Serge PASCAL<sup>1</sup>, François DI PAOLA<sup>2</sup>, Clément BERTHINIER<sup>3</sup> & al.

<sup>1</sup> Université Paris-Saclay, CEA, Service de Recherche en Matériaux et procédés avancés (SRMA), 91191, Gif-sur-Yvette, France (stephane.gounand@cea.fr)

<sup>2</sup> Université Paris-Saclay, CEA, Service d'Études Mécanique et Thermique (SEMT), 91191, Gif-sur-Yvette, France

<sup>3</sup> CEA Marcoule, Service Analyses, Simulation et Production de Sources (SASP), 30207 Bagnols-sur-Cèze Cedex, France

May 31, 2024 version



## Outline

- 1 What is Cast3M?
- 2 Examples
- 3 Feature focus
- 4 Development model, openness
- 5 Future

## *What is Cast3M?*

- Matlab® for the Finite Element Method (generalized rule of three);
- **40 years old;**
- Integrated problem-solving environment:
  - User's programming language **Gibiane** and a developer's programming language Esope;
  - Inline full documentation (french and english);
  - Meshing, computation, post-treatment.
- Focus on non-linear implicit computational mechanics problems (vibration, plasticity, large deformation, contact-friction, fluids) from the nuclear industry;
- Some figures:  $\approx$  100 developers-users (5 at CEA currently), 1500 **test cases**, 9000 Google Scholar hits (Cast3M, Castem), 1.5M lines of code (Esope, half are comments), 170kloc (Gibiane), 170kloc (inline doc), 30Mo compressed (bzip2) executable.

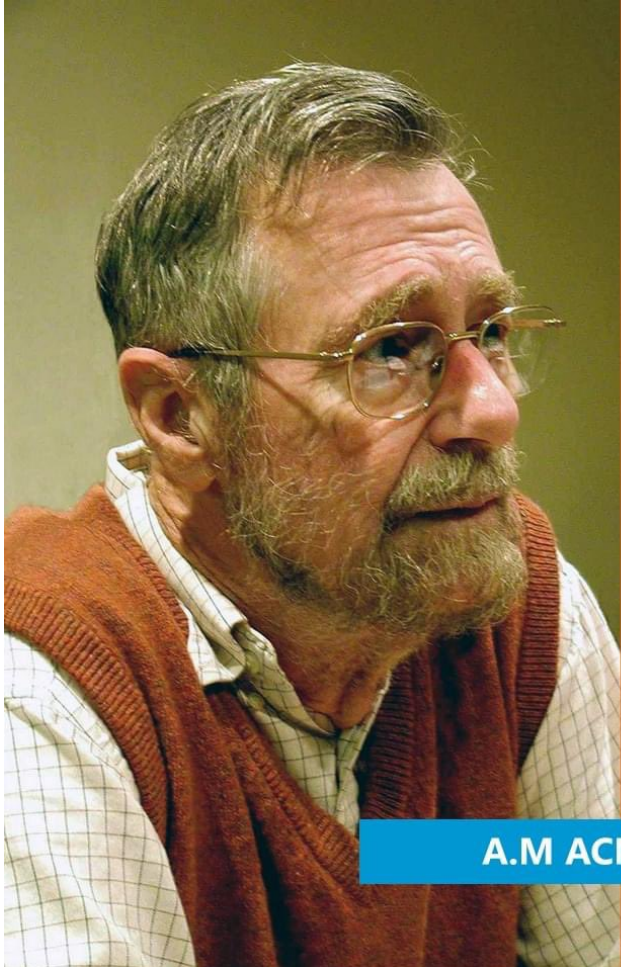
# Gibiane

- **Interactive** REPL (read-eval-print loop);
- Domain-Specific Language? Rather domain-specific objects: maillage (mesh), rigidite (matrix), modele (physical model), chpoint (fields on vertices), chamelem (fields on elements);
- Programming paradigm: **functional programming** (follow the maths) and referential transparency;
- Programming principles: **simplicity**, documentation, regularity, testing;
- Goals: **correctness**, quality.

Almost no syntax:

- operators: `objr = OPERATOR obj1 ... objn ;`
- statements: `STATEMENT obj1 ... objn ;` (Warning: side-effects!).

# Simplicity?



57  
acm

**“Simplicity is a great virtue but it requires hard work to achieve it and education to appreciate it. And to make matters worse: complexity sells better.”**

**Edsger Dijkstra**

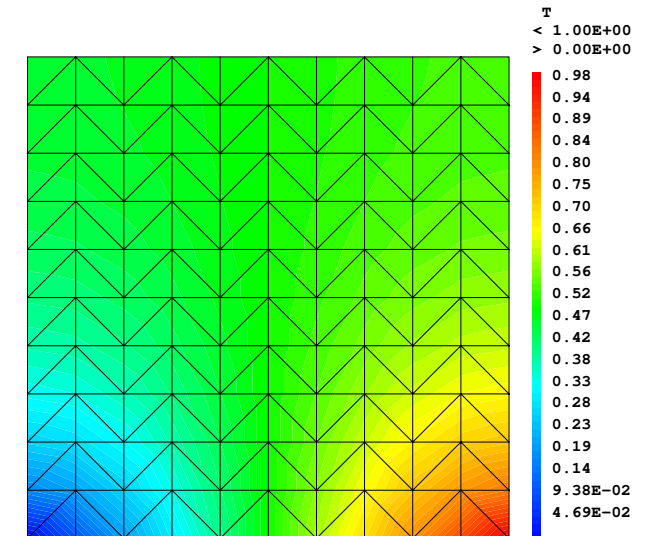
**A.M ACM Turing Laureate**

## Simple example

```

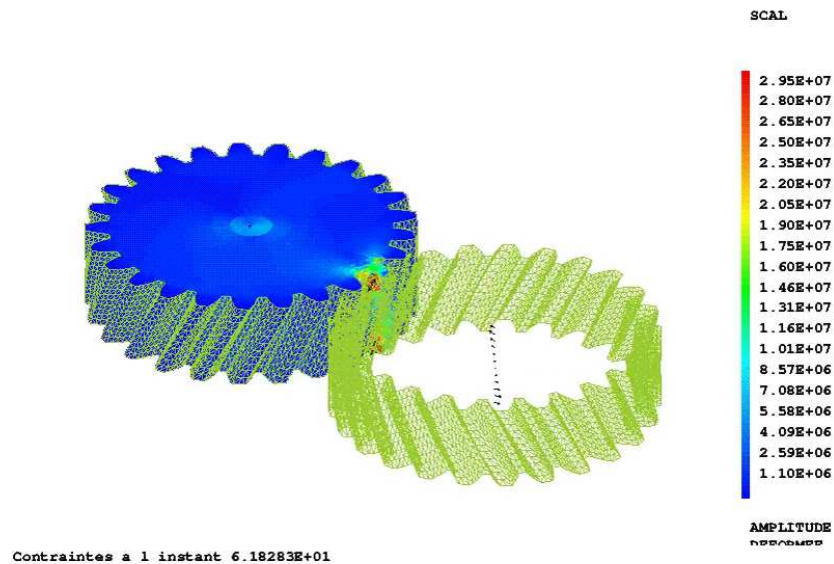
* Hello world FEM-style
* General options
OPTI DIME 2 ELEM TRI3 DENS 0.1 ;
* Mesh
p0 = 0. 0. ; p1 = 1. 0. ;      POINT
bot = DROI p0 p1 ;             MESH
msh = TRAN bot (0. 1.) ;      MESH
* Problem
mod = MODE msh THERMIQUE ;    MODEL
mat = MATE mod K 3.14 ;      ELEMENTWISE FIELD
* Matrices
cnd = COND mod mat ;        MATRIX
mbl = BLOQ T bot ;          MATRIX
vbl = NOMC T (COOR 1 bot) ;  NODAL FIELD
fbl = DEPI mbl vbl ;        NODAL FIELD
* Solve
mtot = cnd ET mbl ;         MATRIX
ftot = fbl ;                NODAL FIELD
sol = RESO mtot ftot ;      NODAL FIELD
* Plot
TRAC sol msh ;

```

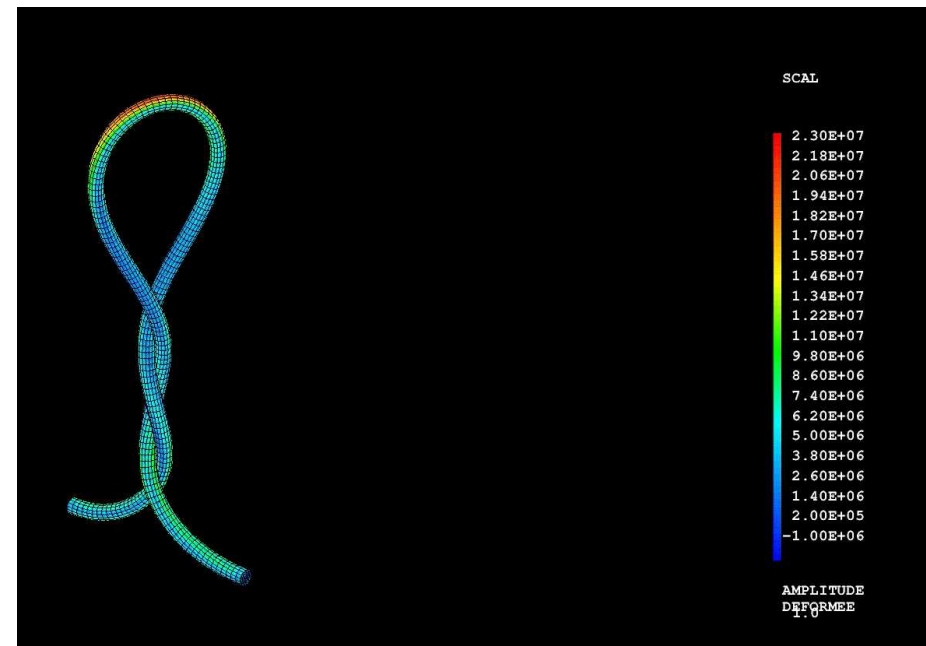


- **Few objects** ( $\approx 40$ );
- **Many operators** ( $\approx 500$ ).

# Contact-friction example (P. Verpeaux)



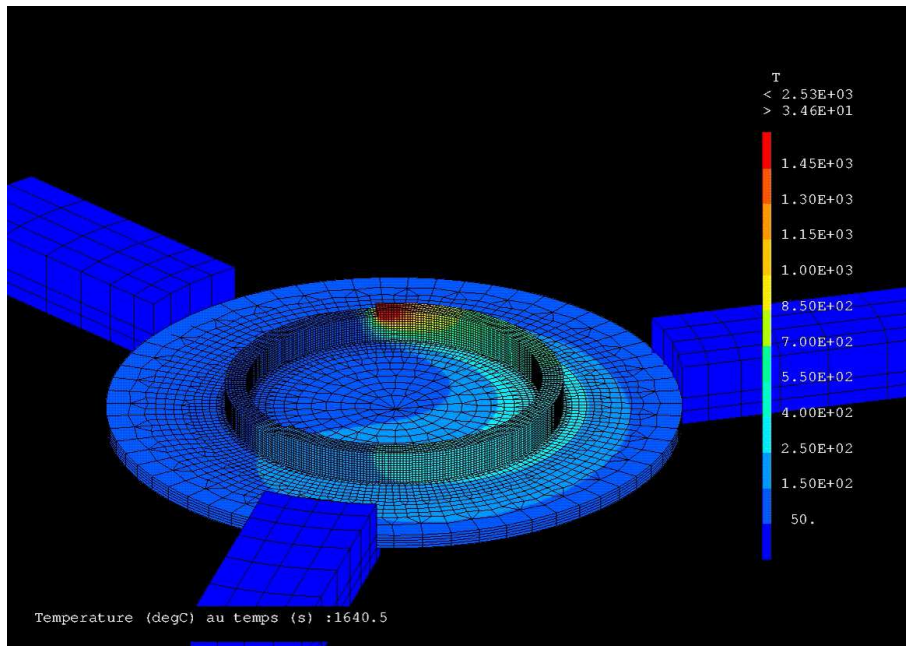
Gears mechanical simulation (local,web)



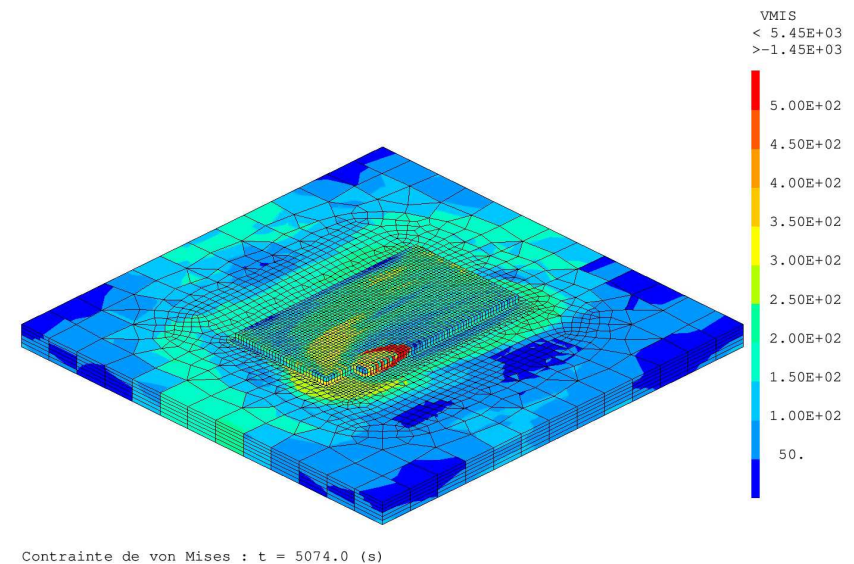
Torsion of an unstretched rope (local,web)



# Additive Manufacturing (AM) example (S. Pascal)

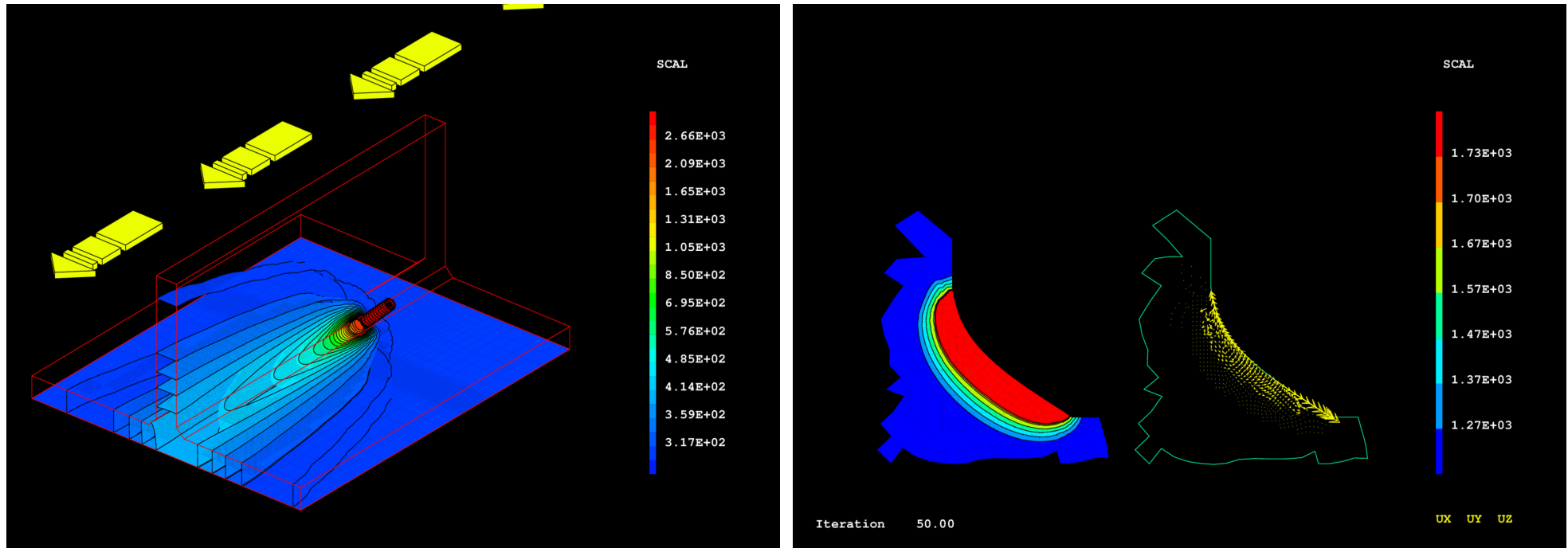


Thermal simulation of an AM-ed pipe  
(video)



Mechanical simulation of an AM-ed 316L layer  
(Von Mises) (video)

# Multiphysics Arc Welding simulation



Tee-like geometry Tungsten Inert Gas (TIG) Arc welding (local, web)

## *Lagrange multipliers everywhere*

Generic **simple** method for prescribing linear(ized) constraints: Dirichlet boundary conditions, periodicity, mortar, contact, incompressibility.

+ **Functional programming**: just assemble terms in the matrix and the RHS 
$$\begin{pmatrix} \boxed{A} & \boxed{B^t} \\ \boxed{B} & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \lambda \end{pmatrix} = \begin{pmatrix} \mathbf{f} \\ \boxed{g} \end{pmatrix};$$

+ **physical meaning** of the Lagrange multipliers  $\lambda$  (reaction, flux, pressure, **conservation**);

– more unknowns;

– **saddle-point** structure (stability, indefiniteness, no optimal algebraic solver);

– hard work in the (linear) solver:

- constraint elimination (static condensation);
- double Lagrange multiplier method (factor without pivoting).

## *Saving and Loading*

A simple way of saving your objects (your work!) at any time: `OPTI SAUV filename ; SAUV ;`

- portable binary or ASCII format;
- full or selective;
- incremental: just call `SAUV ;` multiple times;
- release memory for saved objects: `FANT` statement.

Loading is symmetric: `OPTI REST filename ; REST ;`

- at a given checkpoint in an incremental save;
- should load any savefile from older versions of Cast3M (**quality** requirement for safety studies).

## *Development strategy*

1. Monthly developer's meeting (was weekly at the beginning of the project):
  - discuss only the formal specification (documentation) of an operator, not the implementation:  
`objr = NEW\_OPERATOR obj1 ... objn ;`
  - **Do not document the code, code the documentation!**
2. Program a test case for the new operator (user's language Gibiane);
3. Program the new operator (Gibiane and/or Esope);
4. Integrate it in nightly version of Cast3M (continuous integration).

Traceability of developments since 1988 (web).

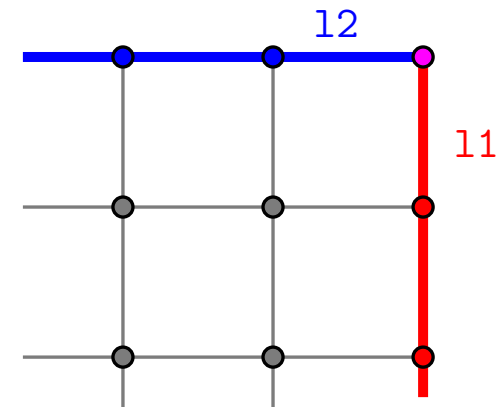
## Error Handling

- **Quality requirement:** an unhandled error is a bug;
- An example of error handling in the RESO solver:

---

```
bloq1 = BLOQ T l1 ; bloq2 = BLOQ T l2 ;
fb11 = DEPI bloq1 0. ;
fb12 = DEPI bloq2 0. or 1. ; REDUNDANT or INCOMPATIBLE
sol = RESO (mat ET bloq1 ET bloq2) (fb11 ET fb12) ;
```

---



- Cast3M warns and continues in the first case and exits on error in the second case;
- Linear algebra libraries generally won't handle this case gracefully,
- Cast3M uses **few external** libraries (fxdr, X11, openGL, pthread, MPI, med) for **quality** reasons: license, longevity, portability, testing, non-regression and **error handling**. Numerical libraries are preferably included (BLAS, LAPACK, ARPACK).

## Openness

- **License:** free for educational or research purposes, small fee for industry users;
- Windows, Linux and MacOS (arm) yearly version with bundled source code;
- **Website:** <http://www-cast3m.cea.fr/>
- Yearly **user's meeting** (last friday of November): free lunch for attendees and free polo for speakers!
- Free yearly **training sessions:** structural mechanics beginner and advanced (e-learning, english or at Université Paris-Saclay, french), additive manufacturing (classroom, french), developer (classroom, french). Training slides on the website;
- Integration of external developments done by CEA (ex: topological optimization, biomechanics of tumoral cells);
- Free Hotline.

## *Conclusion*

### Conclusion

- Ease of use of an integrated “white box” problem-solving language;
- Keep simplicity with useful generic tools: Lagrange multipliers, save-load feature, error handling;
- Document, test, code, integrate.

### Future

- Battle the curse of complexity;
- Handle more complex applications;
- Have fun formalizing and solving problems!



*The End*

Questions?

# Optimization

The real problem is that programmers have spent far too much time worrying about efficiency in the wrong places and at the wrong times; premature optimization is the root of all evil (or at least most of it) in programming.

Donald E. Knuth (1974)

*Computer Programming as an Art:  
ACM Turing Award Lecture*



©Creative Commons

## *The bad and the ugly?*

- No nodes object: shared and hidden (but a configuration object exists);
- Operator syntax is in french;
- Lack of regularity on the developer's side (duplication of utilities and of an object);
- Operators are ordinary objects (character string) and can be overloaded, name scoping is global in procedures;
- Very ambitious parallelism model (multiple operators-multiple objects).

## *Automatic parallelism*

### Asynchronous shared-memory parallelism

- `result = ASSI iassi OPERATOR obj1 ... objn ;;`
- Single instruction multiple data:  
`objp = PART mesh or modele ;`  
`result = ASSI TOUS OPERATOR objp ... ;`
- Automatic user-level parallelism: `OPTI PARA vrai ;;`
- Internal parallelism for some operators (`RESO`) with pthreads;

### Distributed-memory parallelism

- `COLL ENVOYER i obj1 ... objn ; and obj1 ... objn = COLL RECEVOIR i ;`
- Synchronous operator by default but asynchronous with `ASSI` ;
- Quite similar to save-load functionality;
- Not automatic!