An Experiment in CAD Tool Formal Specification

UNU/IIST — Seminar n.º 2
5–16 May 1997
Macau

J. Nuno Oliveira
University of Minho / INESC (Group 2361)
Braga
Portugal

email: jno@di.uminho.pt
http://www.di.uminho.pt/~jno

Abstract

- This paper illustrates the application of **formal methods** to the step-wise development of a software system for building-specification and automatic building-plan plotting.
- A few iterations of a **building-description language** are formally specified and "animated" using a rapid prototyping shell supporting formal specification techniques.
- A prospect of a final implementation is given which not only will provide the system as a "plug-in" extension of a CAD package but also will connect it to an underlying **production database** allowing for automatic housing project materials planning and budget control.

Keywords

Formal methods
Rapid prototyping
Computer graphics
CAD tools

Background

- **3D Graphics**, **multimedia** and **virtual reality** available thanks to enormous advances in hardware support
- Overwhelming offer in the CAD market for architects and civil engineers — **AutoCAD**, Softdesk **AutoARQ**, **AutoVision**, **3D Studio**, **PowerPak**, Bentley's **MicroStation**. . .

. . . but one easily forgets that all these marvels have been made possible, in the first place, by basic research in **Mathematics** (numerical analysis, vector space algebra, spline theory, *etc.*)

However...

! Commercial tools may run slowly on modest hardware platforms

Questions

- Are the tools available in the market working at the **right level of abstraction**?
- Are building descriptions doomed to be represented by (huge) collections of **low-level** graphical entities?
- Or do they possess an intrinsic, high-level **algebraic structure** which is “simple” and can be reasoned about?

Our Targets

- House designs can be described by **mathematical expressions**
- House design **algebra**
- House design retrieval supported by a formal **classification** schema

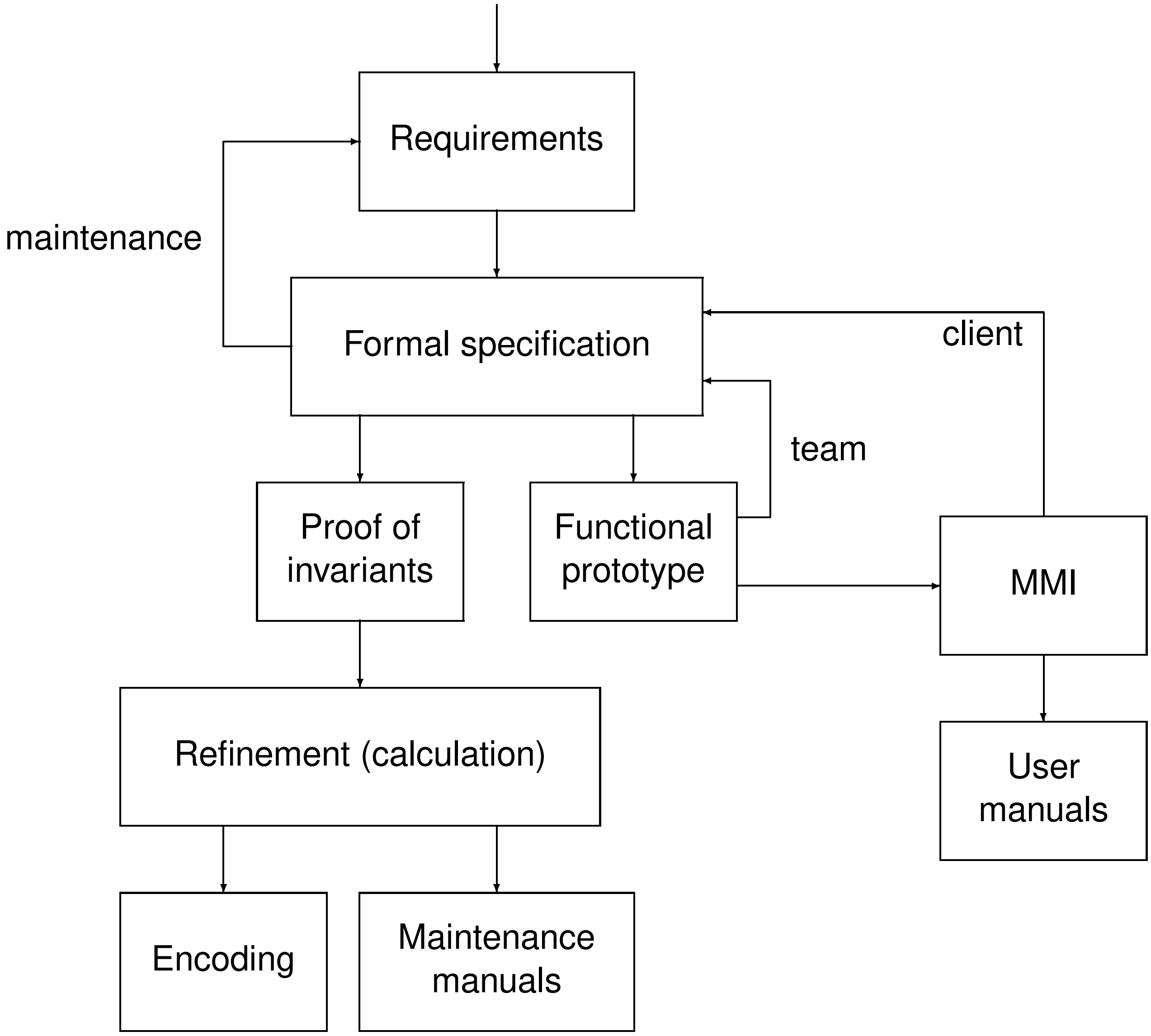
Our Strategy? Formal Methods

Formal methods adopt the “universal problem solving” strategy which one is taught at school:

- **understand** your problem
- build a **mathematical model** of it
- **reason** in such a model
- **upgrade** your model, if necessary
- **calculate** a final solution and implement it.

Formal Methods Software Life-cycle

The main target of the formal methods trend is to drive software production into good engineering standards:



Requirements

maintenance

Formal specification

client

team

Proof of invariants

Functional prototype

MMI

Refinement (calculation)

User manuals

Encoding

Maintenance manuals

Our Environment

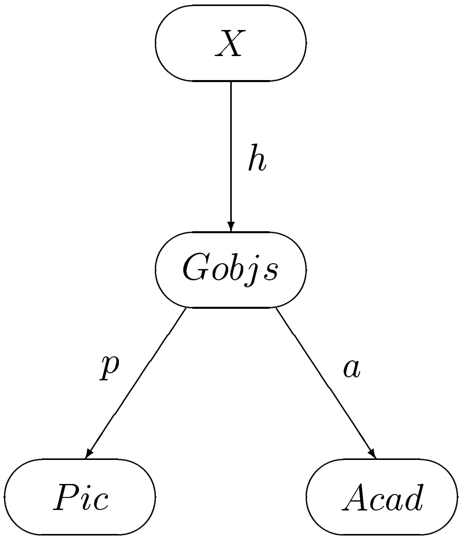
- SETS formal specification language
- CAMILA shell for “animating” formal specification prototypes
- LaTeX for graphical output drafting

- **AutoCAD**™ as final target enviroment

Outline of Project Plan

This paper reports on the formal specification phase of a structured **building description language** processor intended to be incorporated, in a later implementation phase, into commercial CAD systems for drawing office practice.

Block Diagram of the Building Language Processing Unit



X — abstract building description level
$GObjs$ — intermediate graphical object level
Pic — LaTeX `picture` command level
$Acad$ — **AutoLISP** command level

The main purpose of the $GObjs$ layer is to isolate both the language definition (X) and its processor (h) from specific requirements of commercial CAD tools.

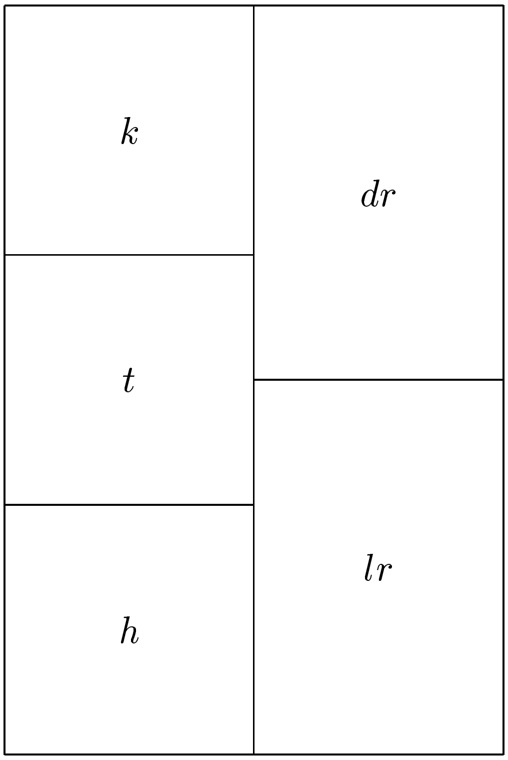
What is a Building Plan?

To **specify** a building consists of abstractly recording **what** the architect wants to build rather than **how** it is actually built in terms of civil engineering materials.

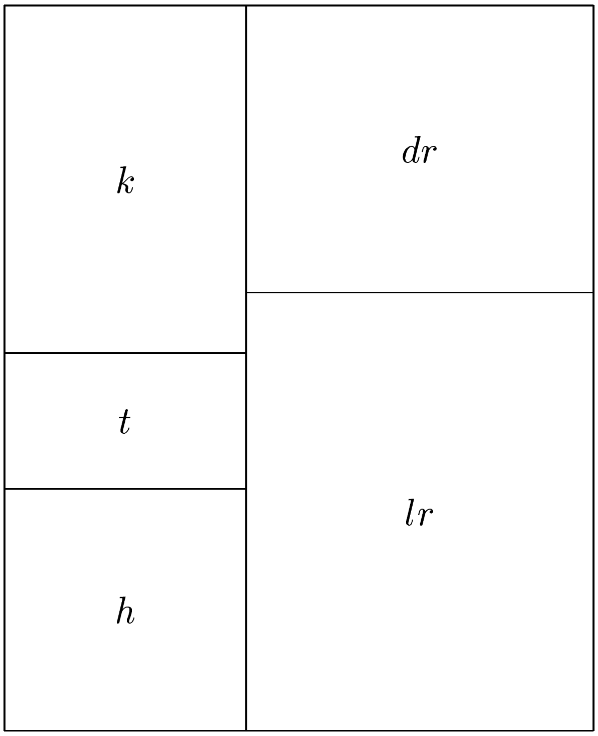
Requirements

> *I want the living and dining rooms on the right, thus facing east, the latter communicating with the kitchen on its left and the former communicating with the entrance hall which, in turn, should have discrete access to some toilets by the kitchen and will be partly occupied by the staircase leading to the second floor.*

Draft “Minimal” Building Plan

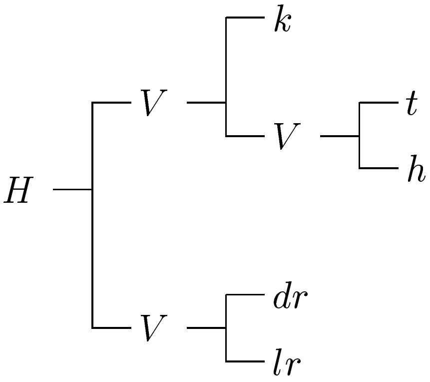


Resizing of Rooms



Important

"Similarity" between both house plans is captured by the following **expression tree**:



that is

$$H(V(k, V(t, h)), V(dr, lr))$$

H and V: operators for horizontal or vertical composition of space units.

The Algebra of Building Specification

Operators

$$\begin{aligned} H &: X \times X \to X \\ V &: X \times X \to X \end{aligned}$$

$$Box : \mathbb{N}_0 \times \mathbb{N}_0 \to X$$

Axioms

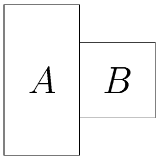
$$V(x, Box(0,0)) = V(Box(0,0), x) = x$$
$$H(x, Box(0,0)) = H(Box(0,0), x) = x$$

$$\begin{aligned} V(x, V(y,z)) &= V(V(x,y), z) \\ H(x, H(y,z)) &= H(H(x,y), z) \end{aligned}$$

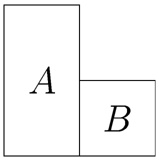
etc

Algebra Specialization

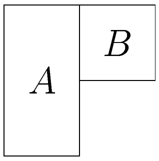
Because we cannot distinguish between



and



or



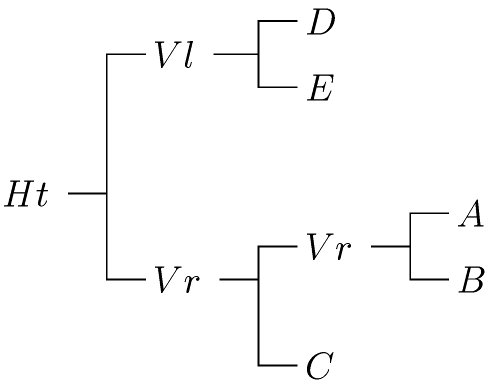
- H specializes in Ht or Hb, respectively for horizontal composition *aligning boxes at the top* and horizontal composition *aligning boxes at the bottom*
- V specializes in Vl (vertical composition with *left alignment*) or Vr (vertical composition with *right alignment*).

Example (a)

Given atomic boxes

$$\begin{aligned} A &= Box(400, 200) \\ B &= Box(250, 200) \\ C &= Box(300, 200) \\ D &= Box(200, 400) \\ E &= Box(150, 200) \end{aligned}$$

write the following expression:

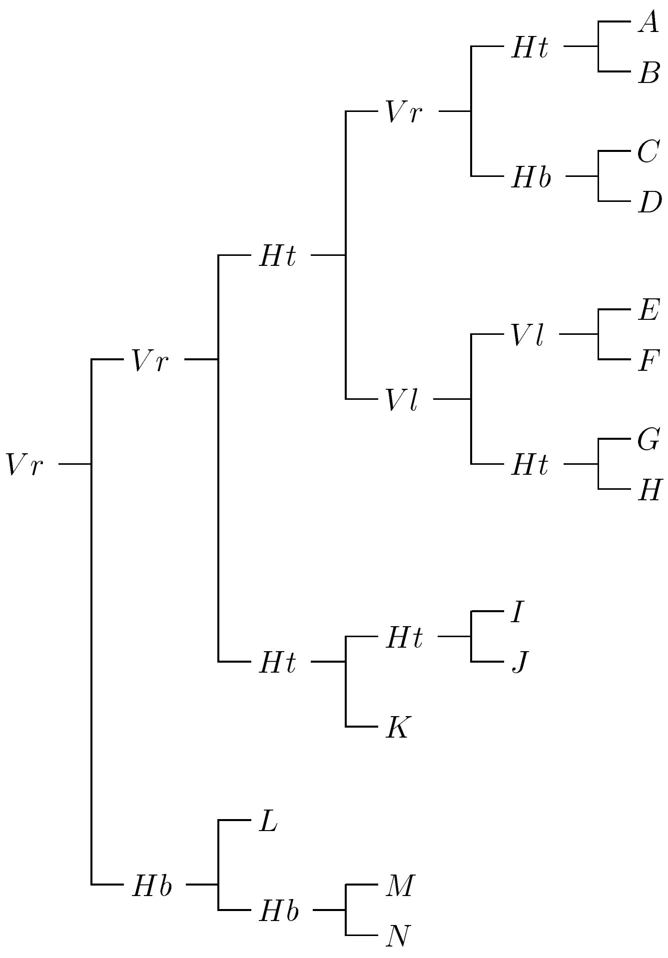


Example (b)

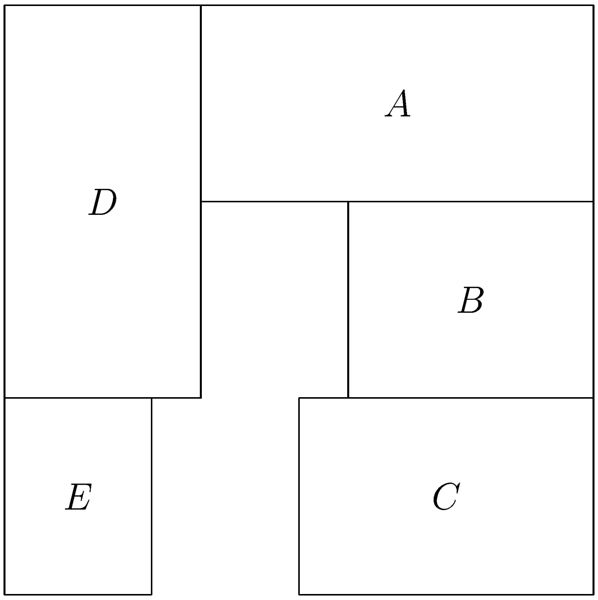
Given atomic boxes

$$
\begin{array}{rcl@{\qquad}rcl}
A & = & Box(170, 350) & H & = & Box(110, 280) \\
B & = & Box(330, 350) & I & = & Box(300, 320) \\
C & = & Box(480, 130) & J & = & Box(360, 320) \\
D & = & Box(150, 390) & K & = & Box(530, 320) \\
E & = & Box(350, 220) & L & = & Box(130, 150) \\
F & = & Box(350, 240) & M & = & Box(450, 150) \\
G & = & Box(240, 280) & N & = & Box(500, 150)
\end{array}
$$

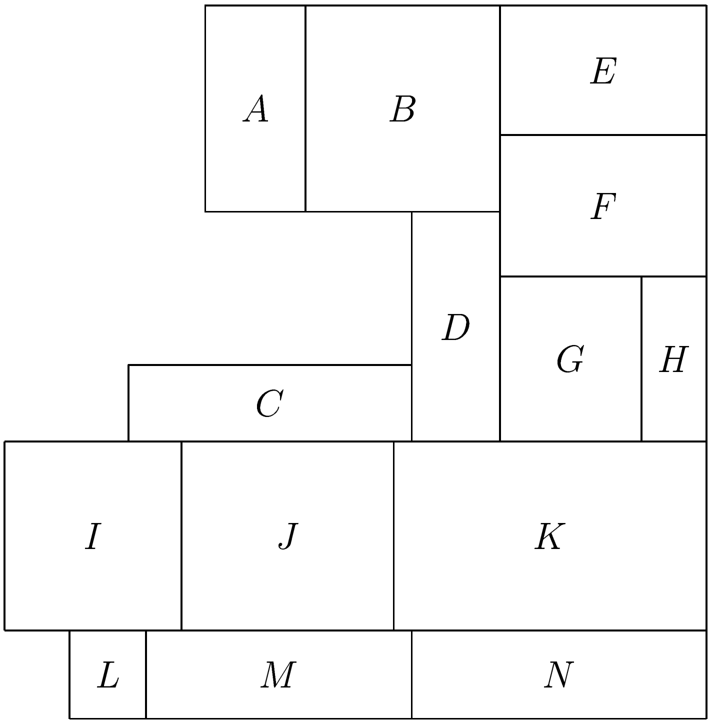
write the following expression:



Graphical outputs



(a)



(b)

Derived Operators

Clockwise and anticlockwise rotation operators:

$$\begin{aligned} Rc &: X \to X \\ Ra &: X \to X \end{aligned}$$

Vertical and horizontal mirror operators:

$$\begin{aligned} Mv &: X \to X \\ Mh &: X \to X \end{aligned}$$

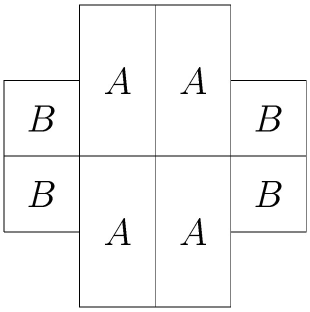
(see [*Oliveira 95*] for many properties involving these and the basic operators).

Reasoning About Abstract Building Plans

The following expression,

$$Vl(Hb(B, Hb(A, Hb(A, B))), Ht(B, Ht(A, Ht(A, B))))$$

whose graphical effect is



can be proved to reduce to

$$\begin{array}{ll} let & x = Hb(A, B) \\ & x' = Hb(Mv(x), x) \\ in & Vl(x', Mh(x')) \end{array}$$

(cf. graphical symmetry).

The Building Specification Language

Abstract Syntax 0: The Bottom-up Style

$$\left\{\begin{array}{rcl} X & \cong & Box + Vl + Vr + Ht + Hb + \\ & & Rc + Ra + Mv + Mh \\ Vl & \cong & X \times X \\ Vr & \cong & X \times X \\ Hb & \cong & X \times X \\ Ht & \cong & X \times X \\ Rc & \cong & X \\ Ra & \cong & X \\ Mv & \cong & X \\ Mh & \cong & X \\ Box & \cong & \mathbf{N}_0 \times \mathbf{N}_0 \end{array}\right.$$

Abstract Syntax 1: The Top-down Style

See paper.

Abstract Syntax 2: Specializing Boxes into Rooms

$$Room \cong \mathbf{N}_0 \times \mathbf{N}_0 \times (WallO \hookrightarrow Open)$$

$$\begin{array}{rcl} WallO & \cong & Bw + Tw + Lw + Rw \\ Bw & \cong & \mathbf{N}_0 \\ Tw & \cong & \mathbf{N}_0 \\ Lw & \cong & \mathbf{N}_0 \\ Rw & \cong & \mathbf{N}_0 \end{array}$$

$$Open \cong Wind + DDoor + LDoor + RDoor + NDoor$$

$$\begin{aligned} Wind &\cong \mathbb{N} \\ DDoor &\cong \mathbb{N} \\ LDoor &\cong \mathbb{N} \\ RDoor &\cong \mathbb{N} \\ NDoor &\cong \mathbb{N} \end{aligned}$$

Invariant Constraints

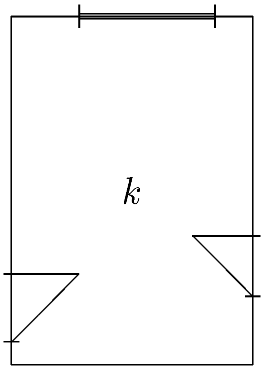
Some space definition constructs of the language are subject to formal **invariant constraints** which should be preserved for building well-formedness guarantee.

Example

Formal expression:

$$\begin{array}{l} k = Room(\\ \qquad 320, \\ \qquad 460, \\ \qquad \{ \\ \qquad\qquad Lw(30) \mapsto RDoor(90), \\ \qquad\qquad Tw(90) \mapsto Wind(180), \\ \qquad\qquad Rw(90) \mapsto LDoor(80) \\ \qquad \} \\ \quad) \end{array}$$

Output:

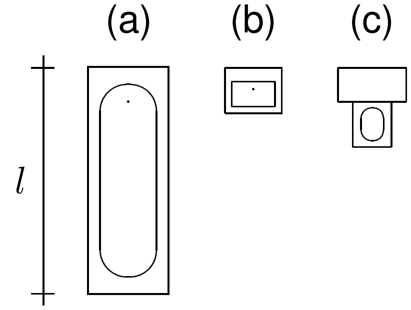


Abstract Syntax 3: Room Internal Specialization

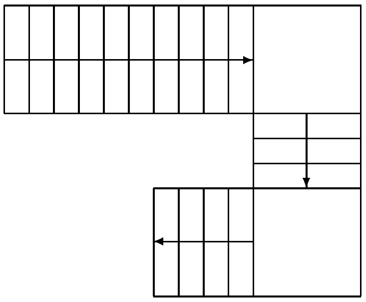
Here, a room becomes a "functional room" ($FRoom$) wherever its specification is that of a $Room$ enriched with a collection of "house objects" ($HObj$) the position of which is once again specified by a wall offset:

$$
\begin{array}{rcl}
FRoom & \cong & Room \times (WallO \hookrightarrow HObj) \\
HObj & \cong & Bathtub + \cdots + WashB + CStairs + \\
 & & AStairs + \cdots
\end{array}
$$

$HObj$ encompasses objects such as bathtubs (a), washbasins (b), WC sinks (c),



staircases,



etc.

Pictorial Semantics

Automatic generation of building location drawings from abstract specifications at three levels of drawing sophistication:

1. simple box outline

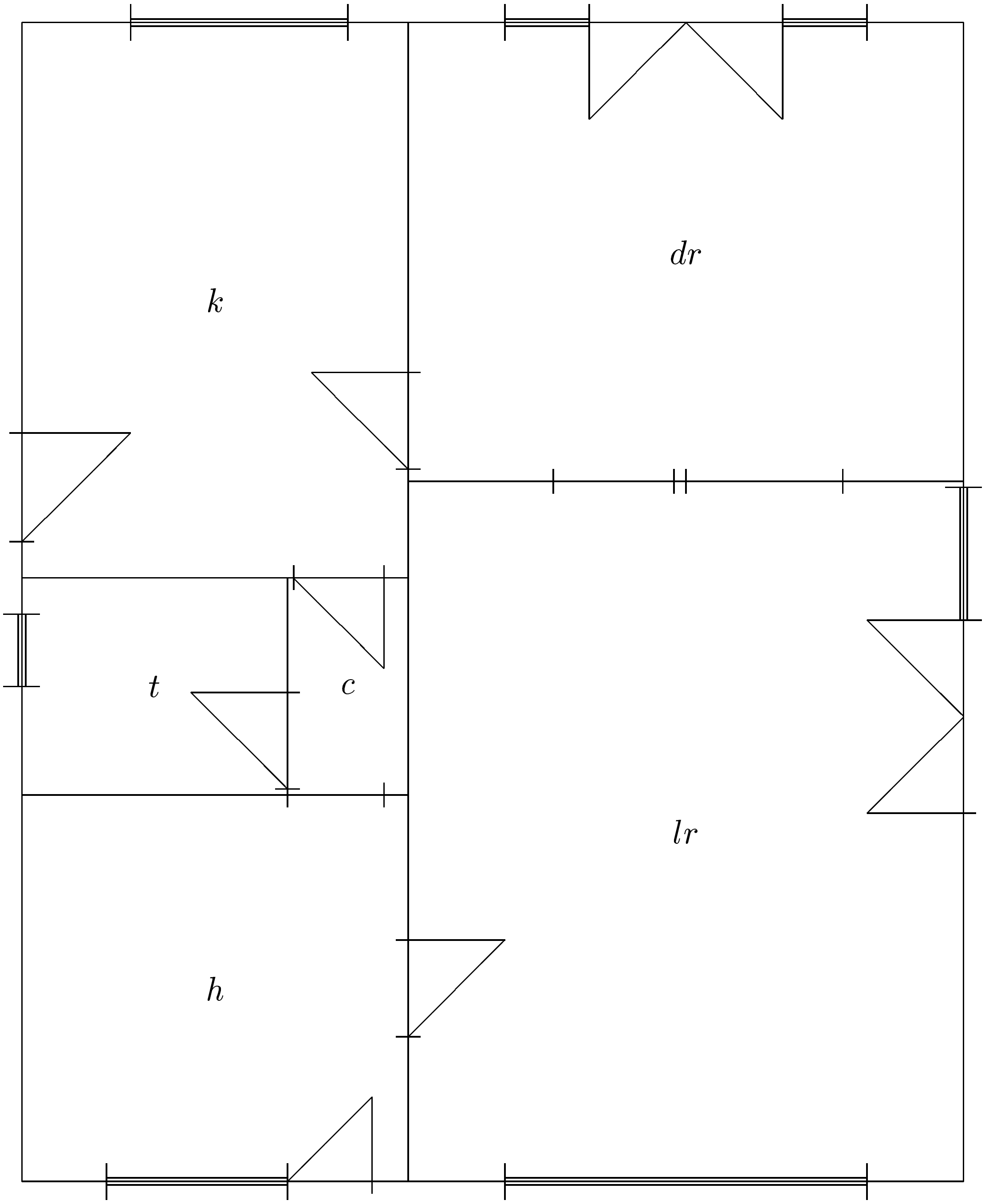
2. building plan sketch omitting internal objects
3. detailed location drawing.

Example

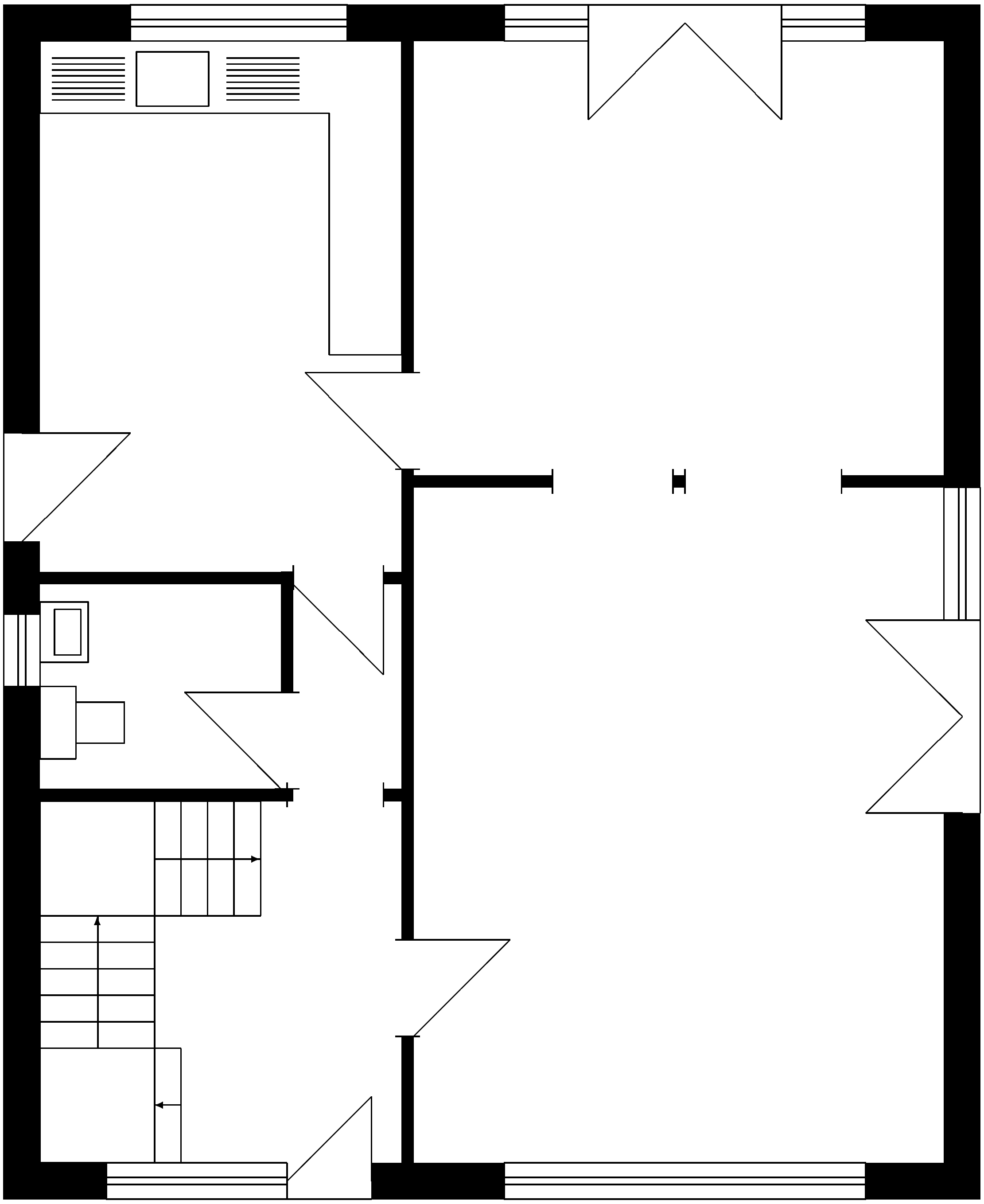
Source (in CAMILA format):

```
Hb(
   Vl(
       FRoom(320,460,"k",[Lw(30)->RDoor(90),
                          Tw(90)->Wind(180),
                          Rw(90)->LDoor(80)],
                         [Tw(15)->KSink(sub(320,20),260,80)
                         ]),
       Vl(
           Hb(
              FRoom(220,180,"t",[Lw(90)->Wind(60),
                                 Rw(5)->LDoor(80)],
                                [Lw(30)->Sink(60),
                                 Lw(110)->Basin(50)]),
              Room(100,180,"c",[Tw(5)->RDoor(75),Bw(0)->NDoor(80)])
             ),
              FRoom(320,320,"h",[Bw(70)->Wind(150),
                                 Bw(220)->LDoor(70)],
                                [Bw(15)->CStairs(95,1,<5,4>)
                                ])
         ),
     ),
   Vl(
       Room(460,380,"dr",[Tw(80)->Wind(70),
                            Bw(120)->NDoor(100),
                            Bw(230)->NDoor(130),
                            Tw(150)->DDoor(160),
                            Tw(310)->Wind(70)]),
       Room(460,580,"lr",[Rw(305)->DDoor(160),
                            Lw(120)->RDoor(80),
                            Bw(80)->Wind(300),
                            Rw(465)->Wind(110)])
     )
 )
```

Output Drawing at level 2 (LaTeX)



Output Drawing at level 3 (LaTeX)



Output Drawing at level 3 (AutoCAD)

(Via **AutoLISP**):

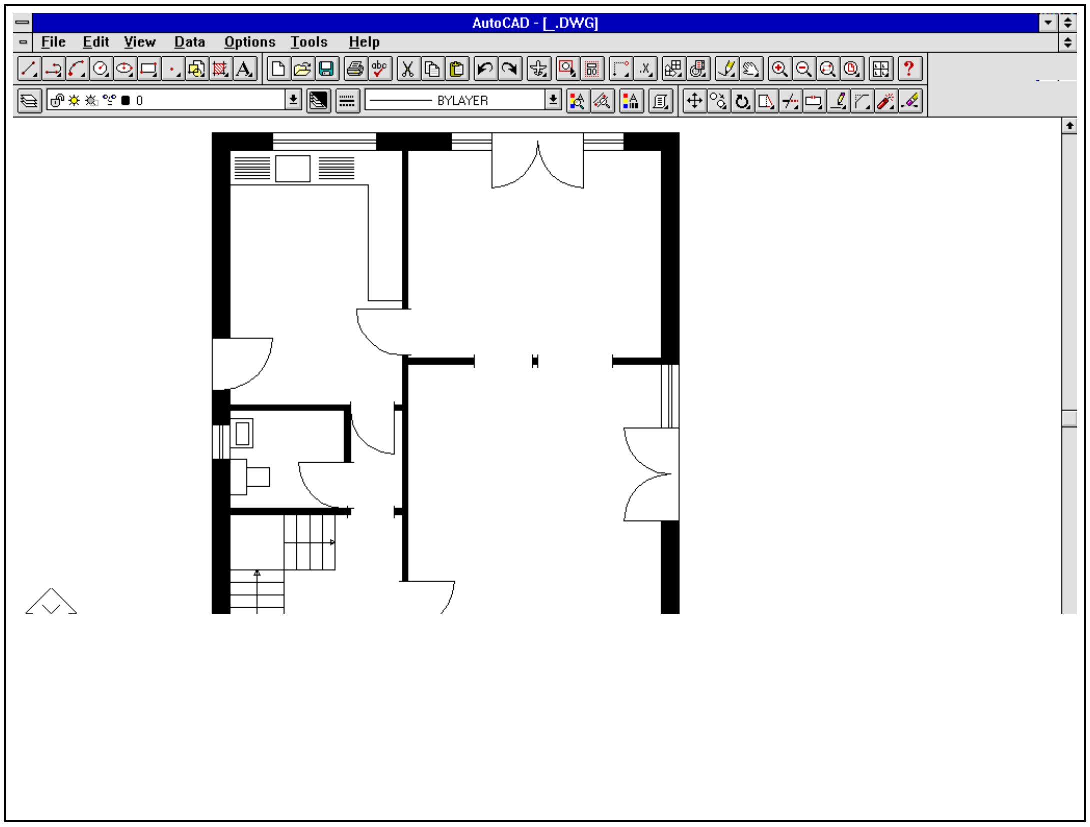
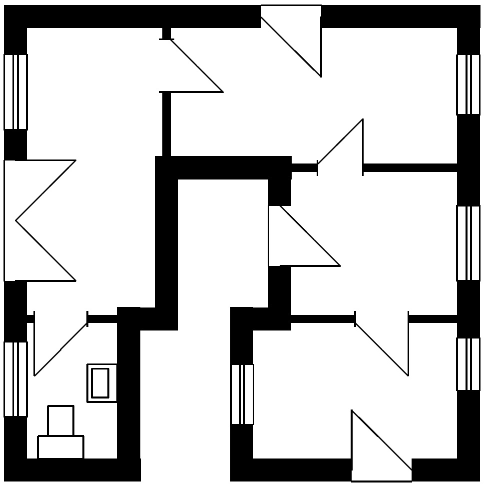
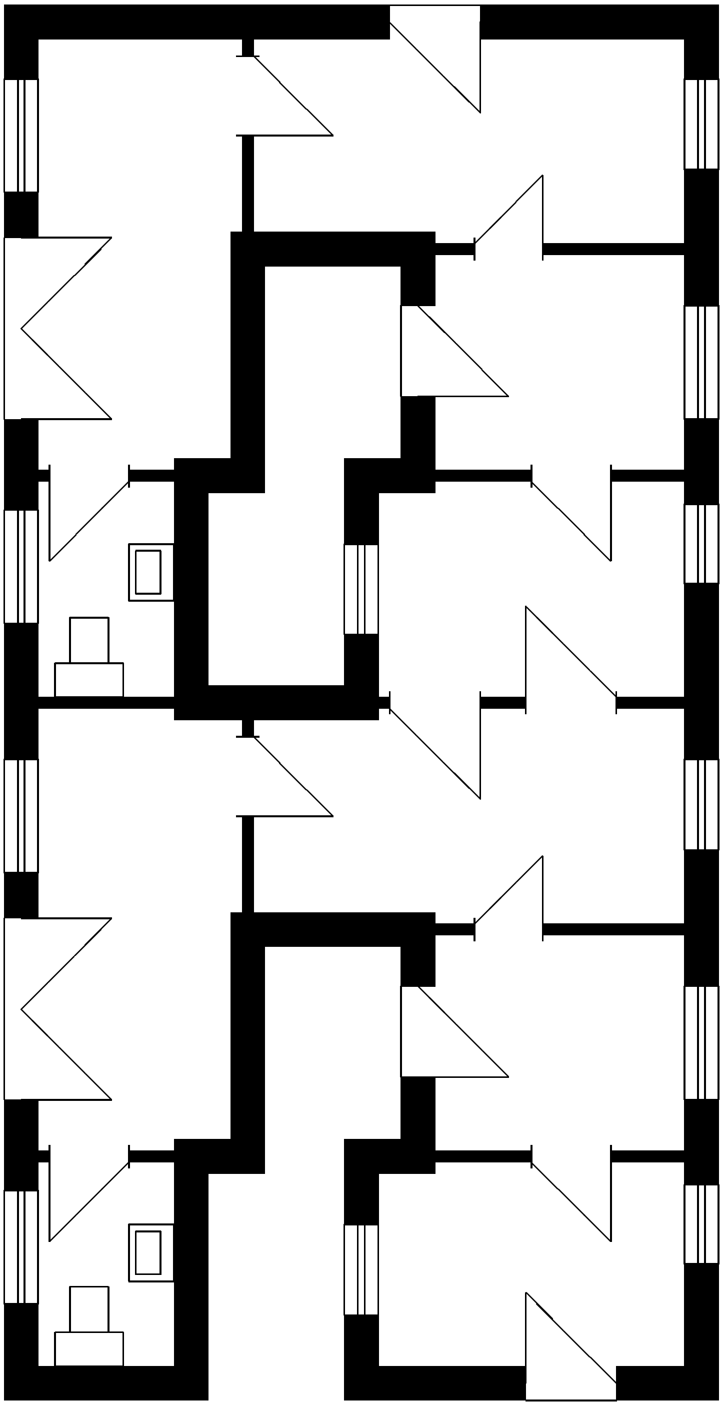


Illustration of Wall Interference Mechanism

(Provided at level 3). Let x be an expression denoting the following house plan:



The replication of this building twice in the vertical direction is specified simply by writing $Vr(x, x)$, yielding (LaTeX output):



Note how some wall segments of x have been automatically converted into inner wall segments, after the replication.

Recent developments

- Group of final year students built EdiSpec^{32}, a DELPHI user interface hiding the original 1600 line CAMILA prototype (version 1.05).
- **AutoCAD** could not be used as a server (!) — a DELPHI previewer was developed instead.
- An Oporto architect company is experimenting with this prototype system.
- Final implementation is likely to become available as a plug-in extension to Bentley's MicroStation ™software.

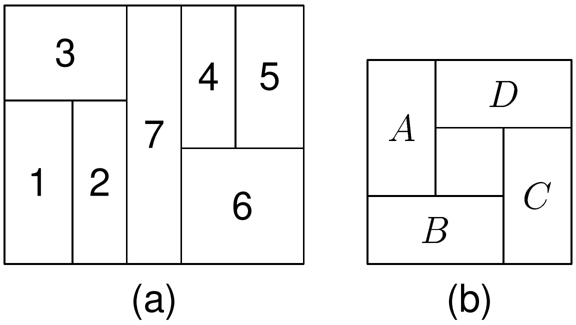
Conclusions

- The adoption of **formal methods** during this work has greatly increased **confidence** on the correctness, expressive power and conciseness of the design.
- The adopted **rapid prototyping** strategy provided a lot of insight on the design, at a **very low cost** (no effort has been put on the final implementation yet).

Future Work

- What has been done so far is the bare start point for the development of a product which should be easily incorporated into available commercial CAD packages.

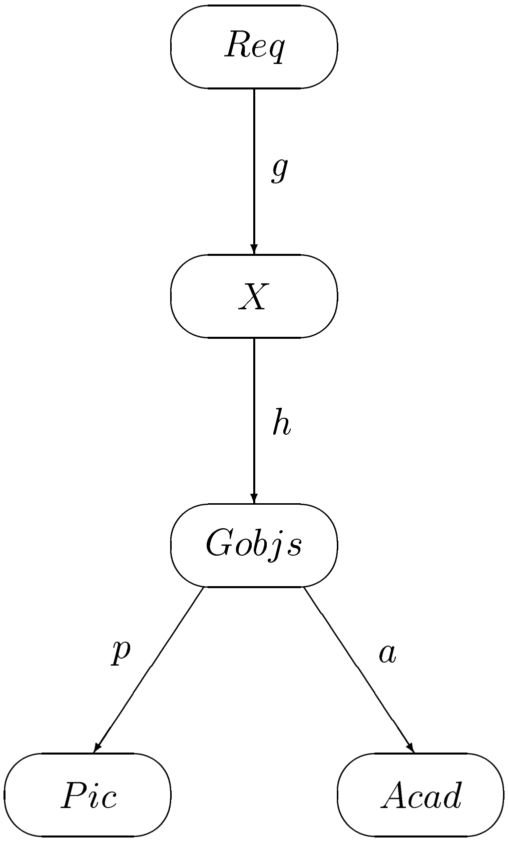
- It should be easy to connect it to an underlying **production database** allowing for automatic housing project materials planning and budget control (see [*Oliveira 95*]).
- **Language extensions**: "L-shaped" room-processing and its relation with VLSI non-slicing floorplanning, eg. (b) in:



(a) (b)

See [*Horta & Lima 97*] for a direct extension of the above model and its application to VLSI desin.

- **Formal classification scheme**: the abstract-term instantiation order can be used to classify building design blocks archived in design repositories (reusability).
- Relation to Virtual reality languages such as VRML.
- **Pictorial Semantics**: exploit better the available technology and libraries (*e.g.* MICROSTATION MDL functions) and introduce building-plan styles.
- **Automatic generation**: nowadays building design is so highly standardized that one is tempted to think of the building specification language as the target of a higher level, in which X-expressions could be generated from loose, constraint-based *requirements*, possibly under the scrutiny of genetic-algorithm evaluation functions:



References

[Oliveira 95] Oliveira J.N. *Formal Specification and Prototyping of a Building Description Languag*. In *Developments in Computer Aided Design and Modelling for Structured Engineering*, 37-51, Volume B of the Proc. of the 4th International Conference on The Application of Artificial Intelligence to Civil and Structural Engineering, Cambridge, UK, 28-30 August 1995.

[Oliveira 97] Oliveira J.N. *An Experiment in CAD Tool Formal Specification.* Submitted to *Computing Systems Engineering”.*

[Horta & Lima 97] Horta C.I., Lima J.A. *Formal Algebraic Specification of Floorplan Topologies.* University of Minho, 1997 (submitted for publication).