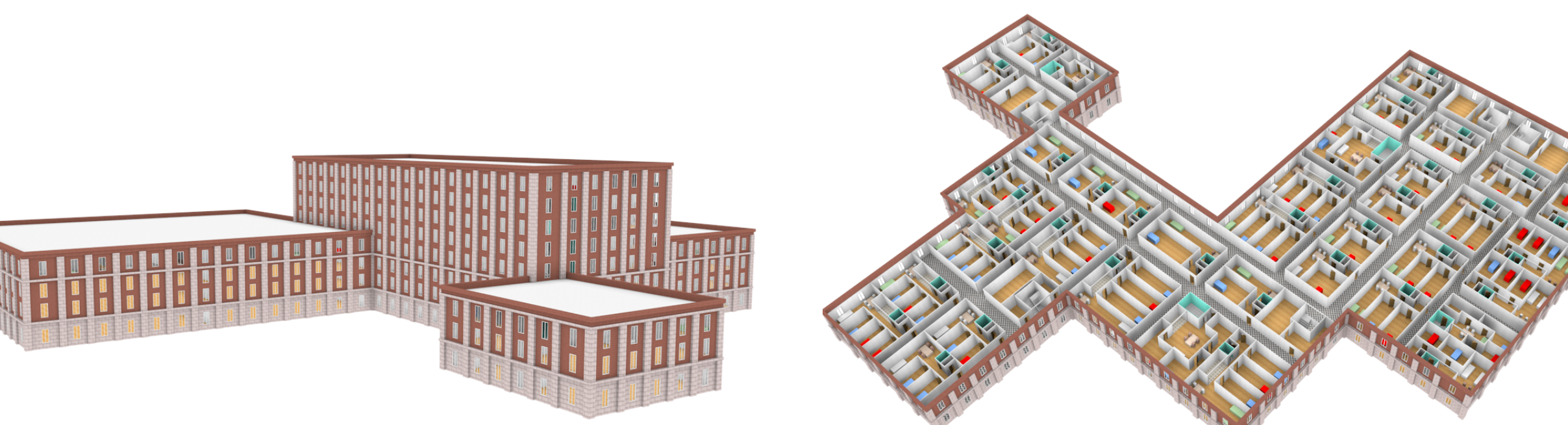
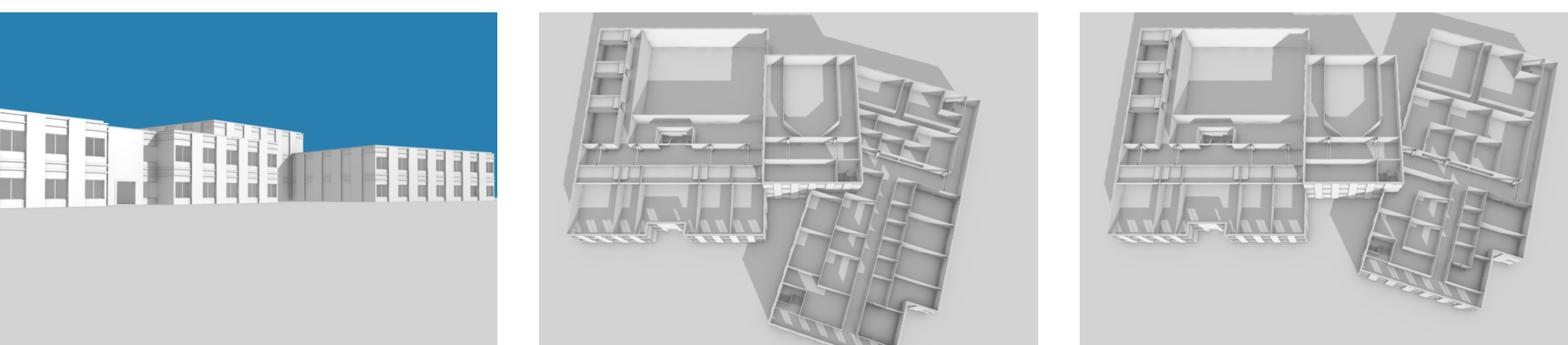
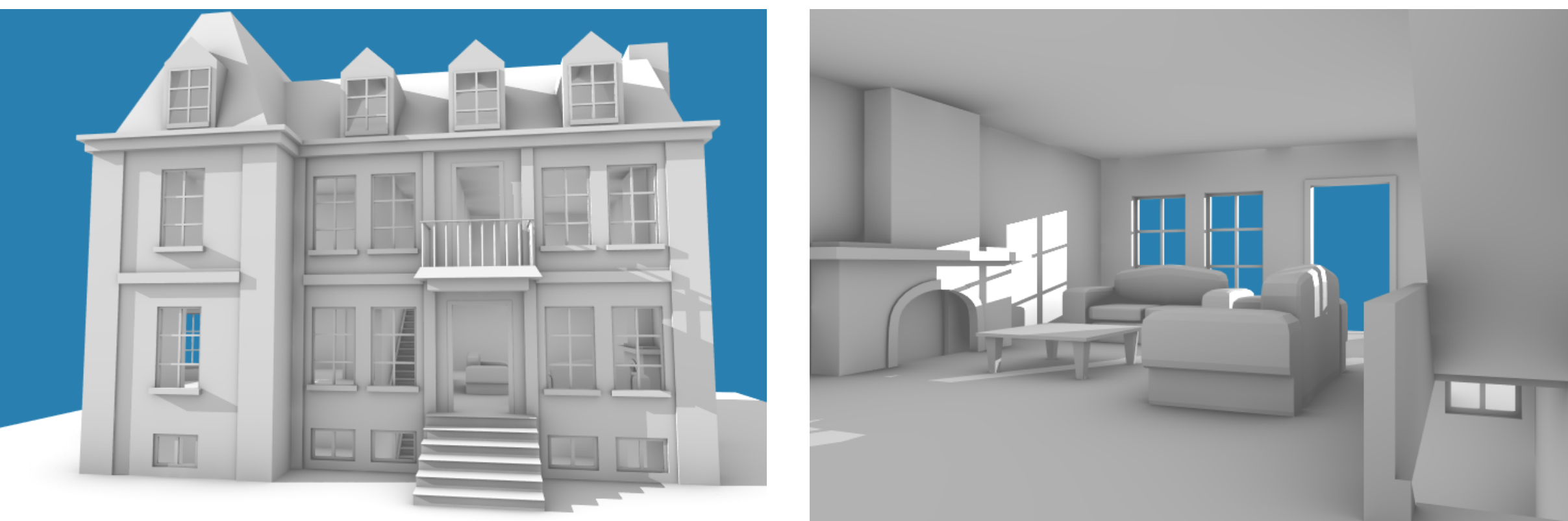
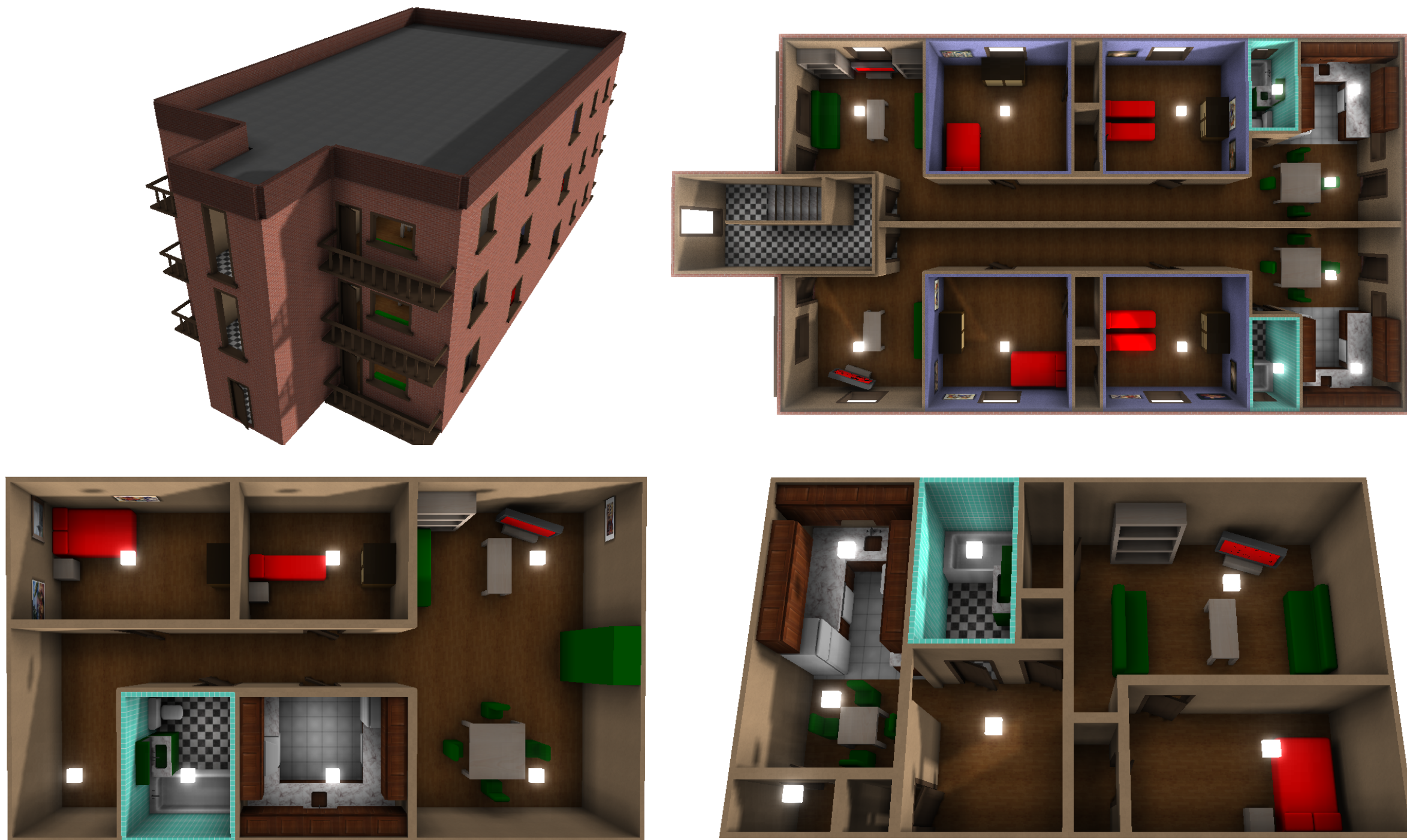


## Goal

Procedural generation of buildings with coherent interiors

## Why?

- Multiple variations
- Easy modifications
- Quick results



## Before

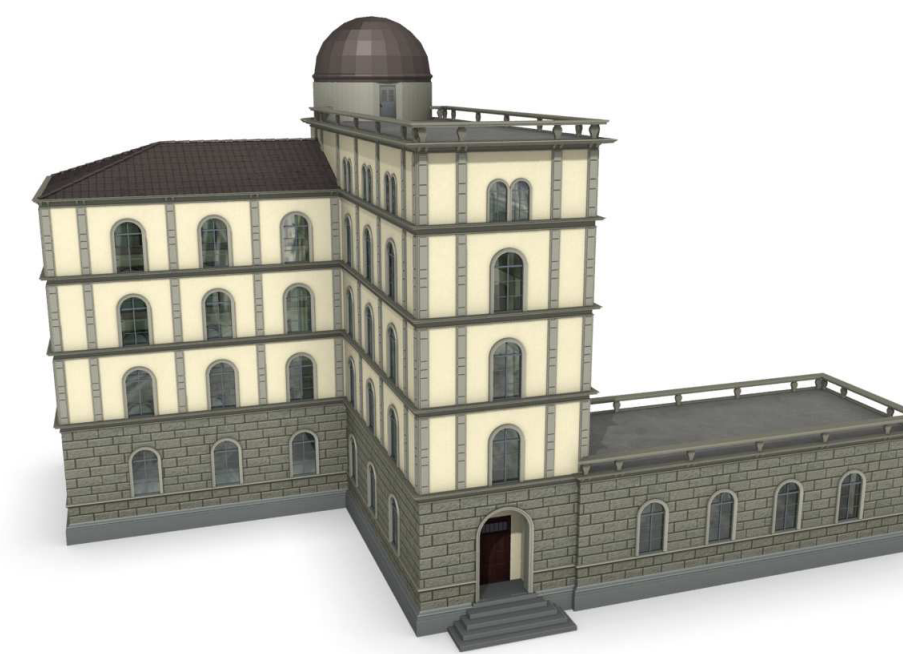
### (Shape) Grammar

Defined by a tuple (N,E,S,R), where  
N is a set of nonterminal symbols  
E is a set of terminal symbols  
 $S \in N$  is the starting symbol  
R is a set of production rules:

$$S \rightarrow aSb$$

$$S \rightarrow \epsilon$$

ab, aabb, aaabbb, ...

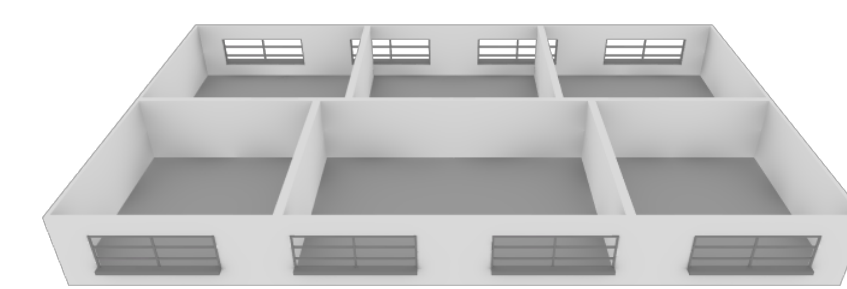


## Changes

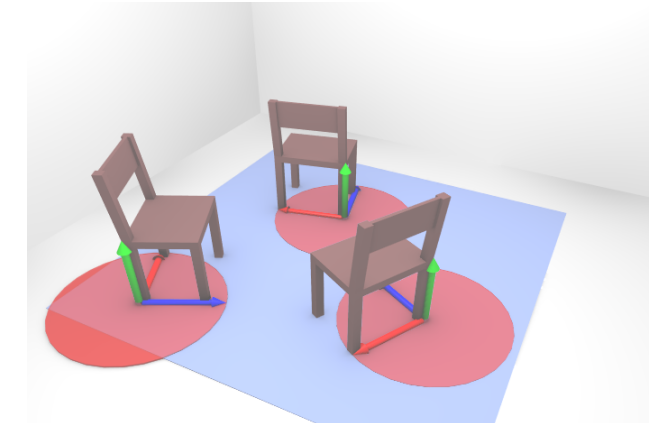
- order of rules execution
- rules become queries (any symbol subset)
- operator execution (each, all)
- symbol becomes a component (persistent)
- any operator

## Operators

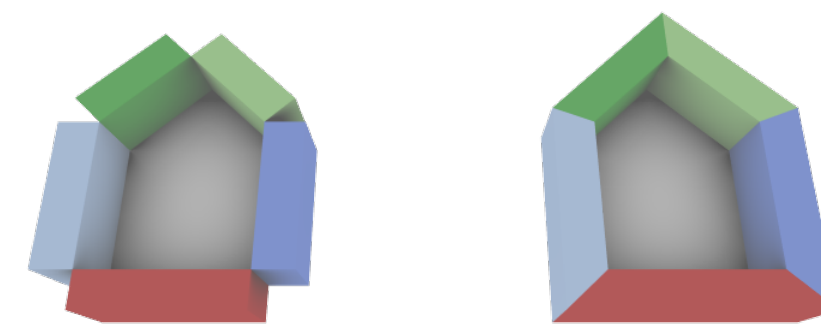
### Split and slice



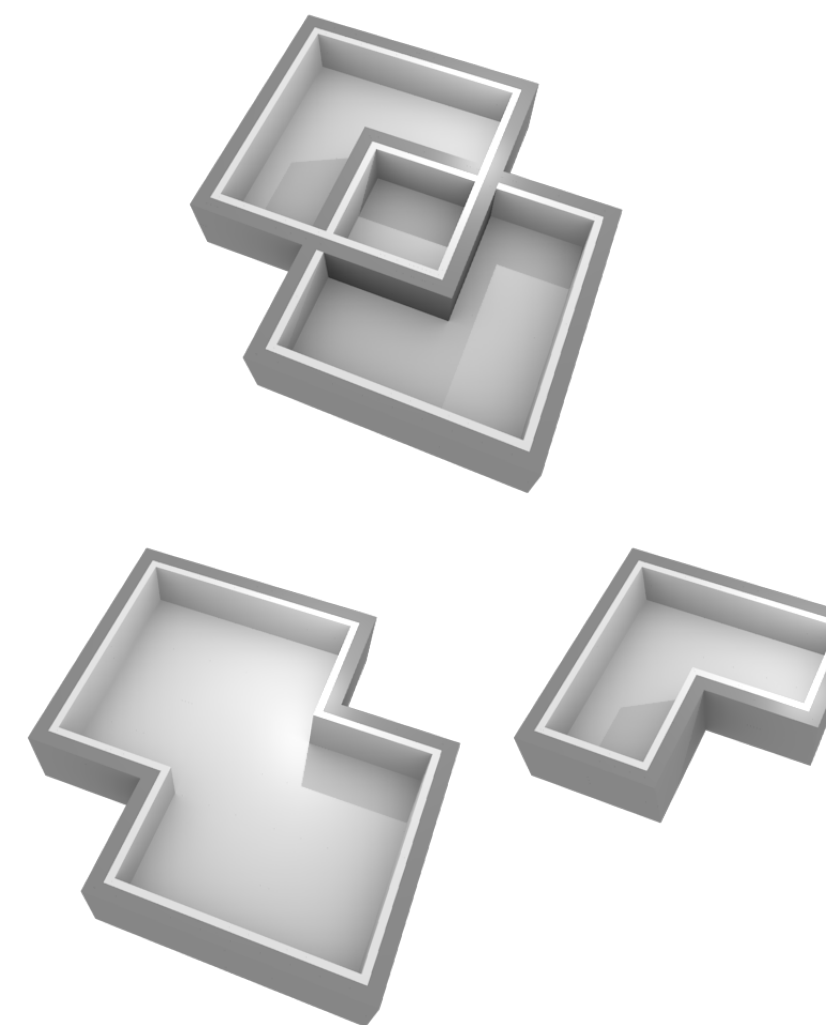
### Connection



### Extrusion



### Boolean



## Solution

### Component

- boundary and bounding box
- labels
- user attributes (inheriting)
- child components
- regions
- connector

### Program

```
// Main component.
component( label="floor", size={10, 2.5, 10} )

// Creation of the apartments.
for c in query( "floor" ) do
  split( c, "Z", { label="living space", rel=1 },
        { label="corridor" , abs=2 },
        { label="living space", rel=1 } )
end

for c in query( "living space" ) do
  split( c, "X", { label="apartment", rel=1 },
        { label="apartment", rel=1 } )
end

// Creation of the elevator shaft (A).
component(
  label = {"elevator", "room"},
  size = {2, 2.5, 2},
  position={4, 0, 2}
)

// Creation of rooms cut by the elevator shaft (B).
for c in query( "apartment" or "corridor" ) do
  subtract( c, query( "elevator" ), { label="room" } )
end

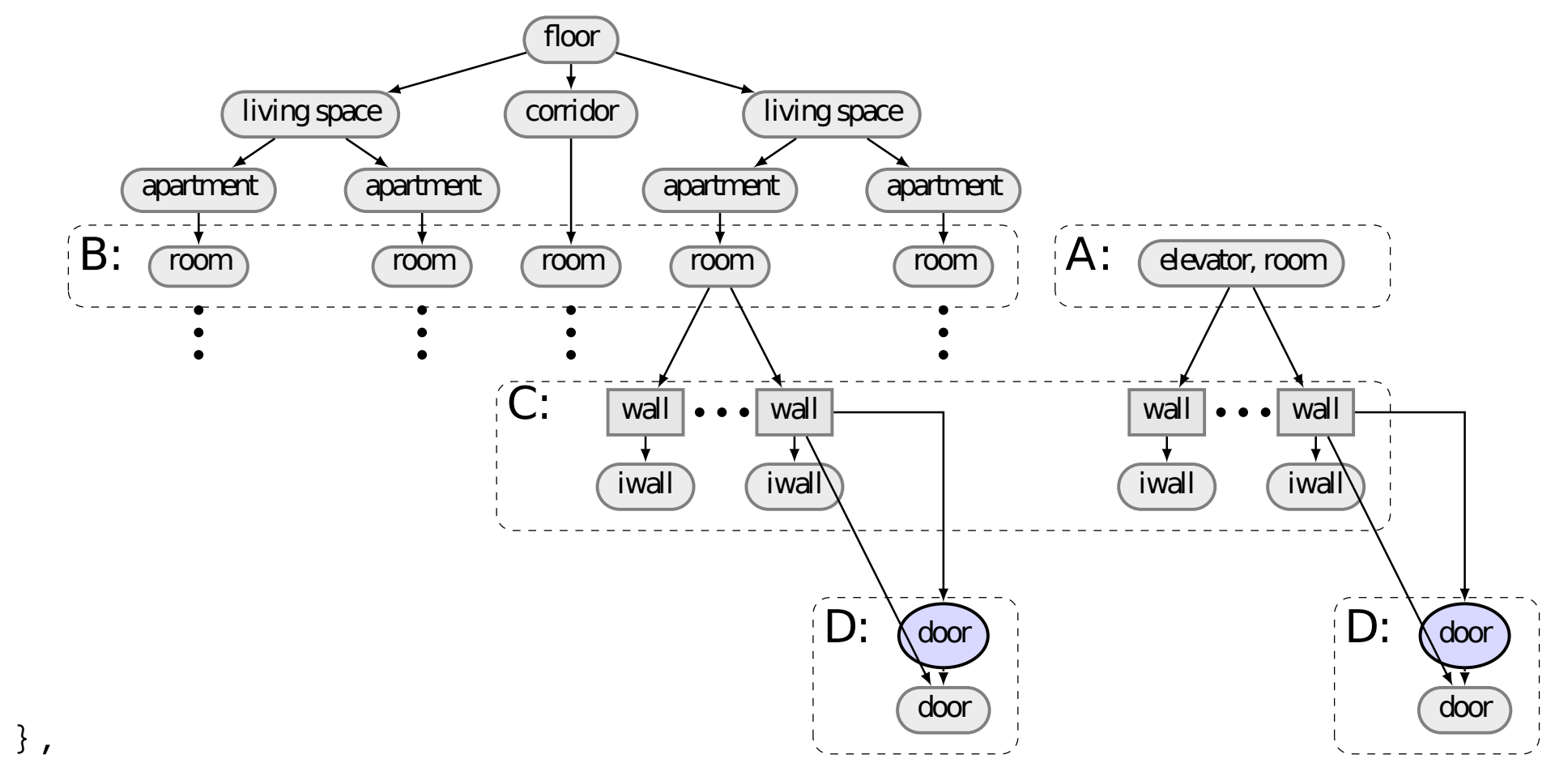
// Extrusion of room walls, with a color attribute (C).
var i = 0
for c in query( "room" ) do
  i = i + 1
  for f in fquery( c, "SIDE" or "BOTTOM" ) do
    component( c, label="wall", boundary=f )
  end
  extrude(
    query( c, "wall" ),
    -0.05, { label="iwall", color=i }
  )
end

// Creation of doors by using regions (D).
for c in query( "wall" and not parent("corridor")
  and occlusion("corridor") > 0 ) do
  region( c, label="door" )
end

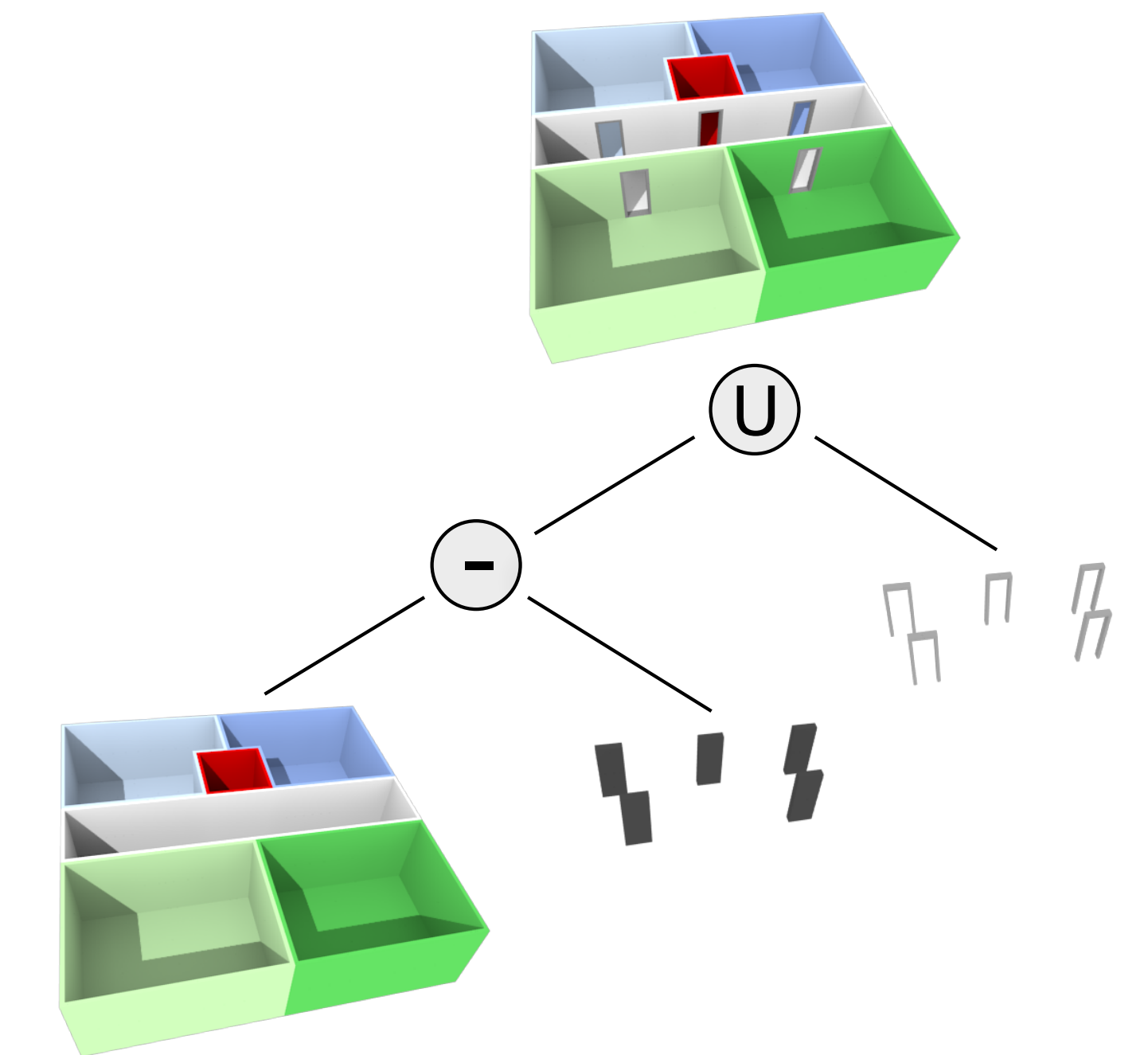
for r in rquery( "door" ) do
  connect( componentFromFile( "door01" ), r )
end

// Creation of the actual geometry.
for c in query( "iwall" ) do
  solidGeometry( c, c.color )
end
```

### Component Graph



### Geometry Graph (CSG)



## Conclusion

- Programming environment
- Flexible and powerful
- Complex task

## Future Work

- Higher level interface
- Optimization of space partitioning

