



UNITÉ MIXTE DE RECHERCHE SUR LES
SYSTÈMES MANUFACTURIERS INNOVANTS



Méta-modélisation, configuration et optimisation de bâtiments dans un contexte d'industrie 4.0

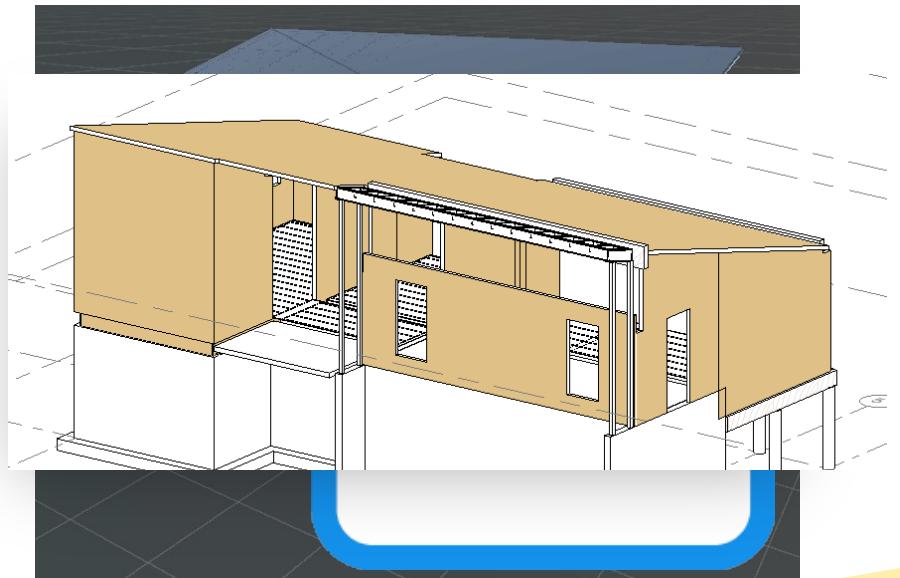
Sokio

Manuel Chastenay



Plan du séminaire

1. Présentation de Sokio - vision 4.0 et besoins pour y arriver
2. Introduction au concept de métamodèle
3. Configuration intelligente de produits
4. Automatisation des tâches / Optimisation
5. La suite des choses



Plan du séminaire

1. Présentation de Sokio - vision 4.0 et besoins pour y arriver
2. Introduction au concept de métamodèle
3. Configuration intelligente de produits
4. Automatisation des tâches / Optimisation
5. La suite des choses



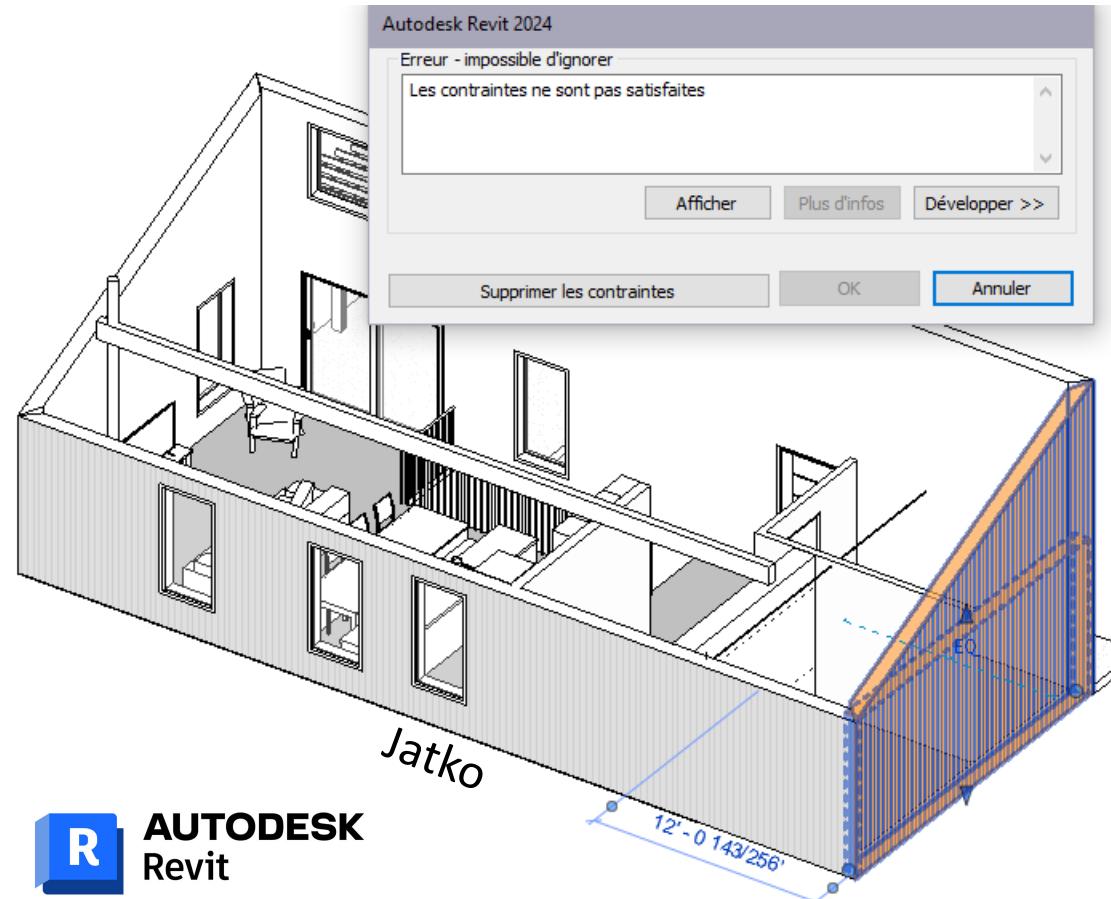
Construction non-traditionnelle

- Préassemblage en usine.
- Potentiel énorme d'automatisation et de liberté au niveau du design.



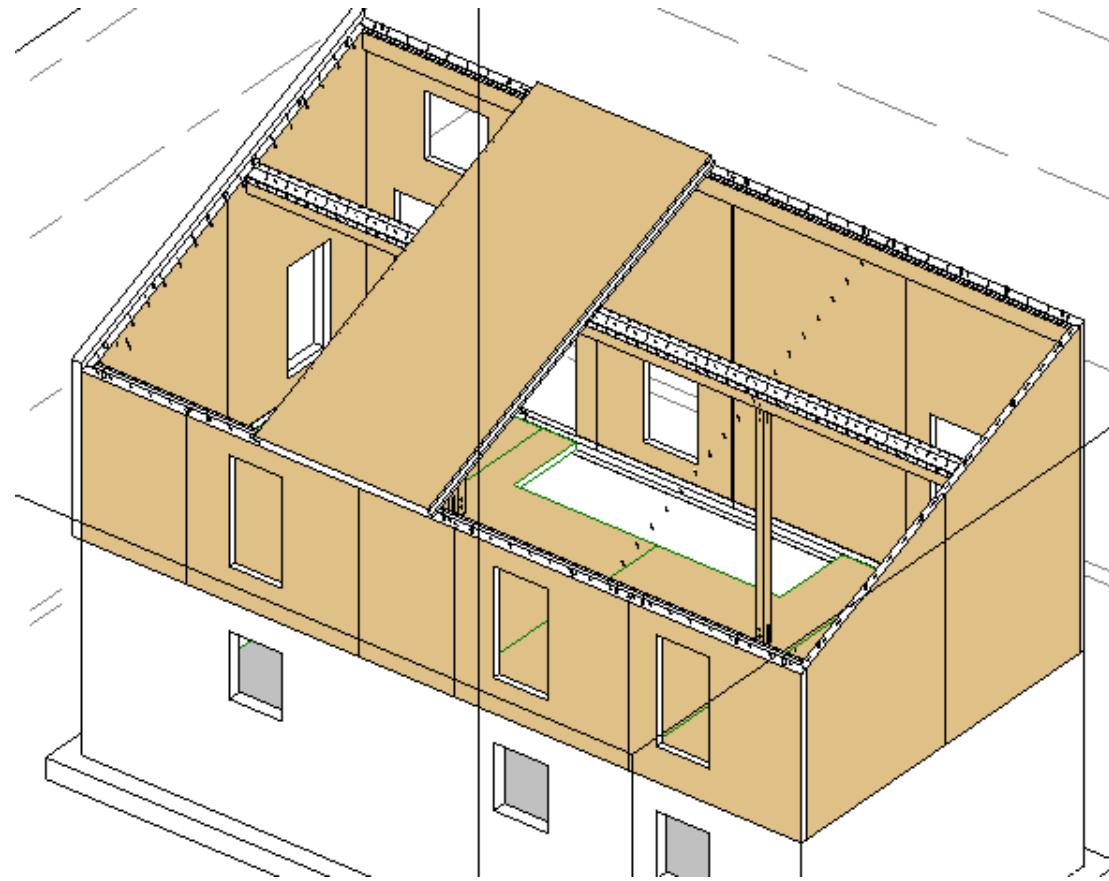
Fonctionnement actuel

- Développement d'un chalet avec l'aide d'un architecte.
- Petite banque de modèles pouvant être construits (\$\$).



Fonctionnement actuel

- Découpe manuelle des panneaux / isolant.
- Pas nécessairement optimisé (nombre de découpes, réutilisation des pertes des matériaux, ordre des manipulations en usine, etc.).
- Chaque modification implique de refaire le traitement entier.



Les buts

- Configuration simple des éléments du chalet.
- Optimisation des découpes de panneaux & isolant et minimisation des pertes en usine.
- Automatisation des trajectoires des robots.
- Automatisation de la création des plans.

Tout ces systèmes, unifiés par l'utilisation d'un langage unique: le **métamodèle**



Plan du séminaire

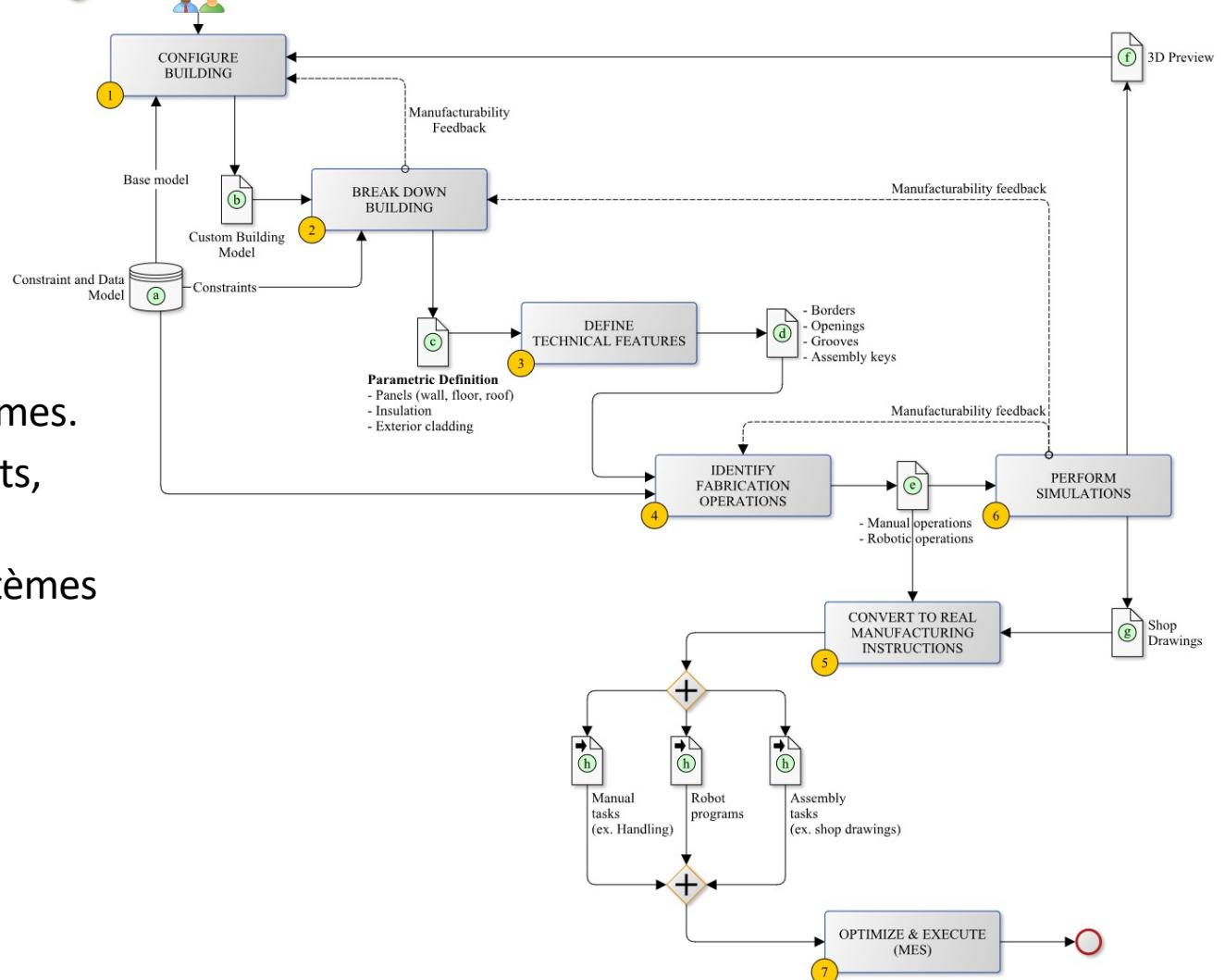
1. Présentation de Sokio - vision 4.0 et besoins pour y arriver
2. **Introduction au concept de métamodèle**
3. Configuration intelligente de produits
4. Automatisation des tâches / Optimisation
5. La suite des choses



KNOW-IT-ALL

Métamodèle

- Modèle d'intelligence centralisé partagé/bonifié par tous les systèmes.
- Définition d'une famille de produits, plutôt que d'un produit unique.
- Adaptabilité des données aux systèmes existants.



Métamodèle

Adoption de l'industrie 4.0

Modèle d'intelligence centralisé partagé/bonifié par tous les systèmes.

Ici, on vient régler le problème d'inter-opérabilité entre les différents logiciels CAD.

Dans le monde de la construction, le format IFC (Industry Foundation Classes) existe.

En pratique, son implémentation diffère dans chaque CAD.



Métamodèle

Adoption de l'industrie 4.0

Définition d'une famille de produits, plutôt que d'un produit unique.

Plutôt que de travailler avec un modèle unique, comme avec IFC, nous ajoutons une nouvelle couche d'abstraction, améliorant les possibilités.

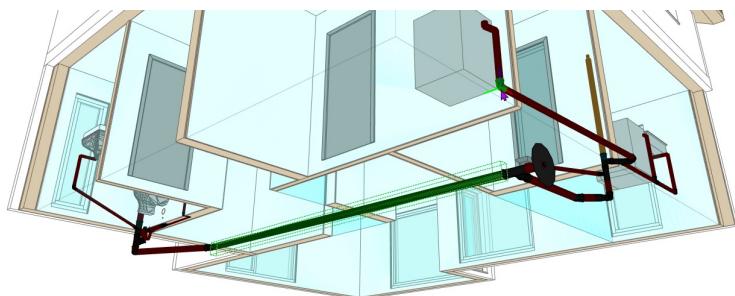


En pratique

Les développeurs de CADs bâtissent le logiciel, et implémentent les spécifications voulues du format IFC.

Si une partie du modèle n'est pas implémentée dans le logiciel, tant pis.

Réexporter le modèle viendra effacer les données qui ne sont pas prises en charge.



12

BIMvision

A	B	C	D	E
1 Ifc File	RME_2010_Trapelo const.ifc			
2 Ifc Directory	C:\Users\nobody\Documents\IFC\			
3 Excel File	C:\...\IFC\IFC4\Examples\Large\RN			
4 Timestamp	2016-01-23T18:19:30			
5 Total Entities	1103889			
6				
7 Entity	Count	IFC4		
8 IfcBuildingElementProxy	2437	Doc		
9 IfcBuildingElementProxyType	698	Doc		
10 IfcCovering	4	Doc		
11 IfcSpace	1	Doc		
12 IfcZone	1	Doc		
13 IfcAlarmType	16	Doc		
14 IfcDuctFitting	1568	Doc		
15 IfcDuctFittingType	522	Doc		
16 IfcDuctSegment	1585	Doc		
17 IfcDuctSegmentType	13	Doc		
18 IfcPipeFitting	438	Doc		
19 IfcPipeFittingType	146	Doc		
20 IfcPipeSegment	506	Doc		
21 IfcPipeSegmentType	6	Doc		
22 IfcValve	32	Doc		
23 IfcValveType	7	Doc		
24 IfcElectricAppliance	177	Doc		
25 IfcElectricApplianceType	4	Doc		
26 IfcLightFixture	1252	Doc		
27 IfcLightFixtureType	100	Doc		
28 IfcDistributionControlElement	175	Doc		
29 IfcDistributionPort	9754	Doc		
30 IfcRelServicesBuildings	793	Doc		
31 IfcArbitraryClosedProfileDef	717	Doc		
32 IfcArbitraryOpenProfileDef	270	Doc		
33 IfcArbitraryProfileDefWithVoids	12	Doc		
34 IfcCircleHollowProfileDef	10	Doc		
35 IfcCircleProfileDef	1475	Doc		
36 IfcExtrudedAreaSolid	4653	Doc		
37 IfcRectangleProfileDef	2703	Doc		
38 IfcSurfaceCurveSweptAreaSolid	264	Doc		
39 IfcMaterial	433	Doc		
40 IfcMaterialList	767	Doc		
41 IfcRelAssociatesMaterial	1194	Doc		
42 IfcPropertySet	2636	Doc		
43 IfcPropertySingleValue	727	Doc		
44 IfcRelDefinesByProperties	2537	Doc		
45 IfcClassification	1	Doc		

< > Summary Header IfcBuildingElementProxy IfcBuildingElementProxyType

En pratique

Nous faisons le contraire, en adaptant le logiciel à ce que le métamodèle nous offre.

Cela garantit l'intégrité des données, et centralise l'information!

Les données peuvent rester inutilisées, mais ne seront jamais effacées.

```
[Serializable]  
40 references  
public abstract class Element  
{  
    public string name;  
    public Types type;  
    public List<ObservableVariable> variables;  
    public List<Element> children;  
  
class Structure : Element  
└ class Panelizable : Element  
    └ class Panel : Element  
        └ class Insulation : Element
```

En pratique

Avant: les modèles se fient uniquement à ce qu'implémentent les systèmes

Maintenant: Les systèmes se fient au modèle, garantissant leur intégrité



Exemple d'instances

```
#499542=IFCPOLYLOOP((#498315,#498317,#498259));
#499543=IFCFACEOUTERBOUND(#499542,.T.);
#499544=IFCFACE((#499543));
#499545=IFCPOLYLOOP((#498258,#498316,#498259));
#499546=IFCFACEOUTERBOUND(#499545,.T.);
#499547=IFCFACE((#499546));
#499548=IFCPOLYLOOP((#498315,#498259,#498316));
#499549=IFCFACEOUTERBOUND(#499548,.T.);
#499550=IFCFACE((#499549));
#499551=IFCPOLYLOOP((#498315,#498118,#498317));
#499552=IFCFACEOUTERBOUND(#499551,.T.);
#499553=IFCFACE((#499552));
#499554=IFCPOLYLOOP((#498262,#498120,#498119));
#499555=IFCFACEOUTERBOUND(#499554,.T.);
#499556=IFCFACE((#499555));
#499557=IFCPOLYLOOP((#498314,#498119,#498118));
#499558=IFCFACEOUTERBOUND(#499557,.T.);
#499559=IFCFACE((#499558));
#499560=IFCPOLYLOOP((#498262,#498314,#498313));
#499561=IFCFACEOUTERBOUND(#499560,.T.);
#499562=IFCFACE((#499561));
#499563=IFCPOLYLOOP((#498313,#498263,#498262));
#499564=IFCFACEOUTERBOUND(#499563,.T.);
#499565=IFCFACE((#499564));
#499566=IFCPOLYLOOP((#498262,#498119,#498314));
#499567=IFCFACEOUTERBOUND(#499566,.T.);
#499568=IFCFACE((#499567));
#499569=IFCPOLYLOOP((#498315,#498314,#498118));
#499570=IFCFACEOUTERBOUND(#499569,.T.);
#499571=IFCFACE((#499570));
#499572=IFCPOLYLOOP((#498255,#498254,#498318));
#499573=IFCFACEOUTERBOUND(#499572,.T.);
#499574=IFCFACE((#499573));
#499575=IFCPOLYLOOP((#498255,#498318,#498321));
#499576=IFCFACEOUTERBOUND(#499575,.T.);
#499577=IFCFACE((#499576));
#499578=IFCPOLYLOOP((#498255,#498322,#498244));
```

```
{
  "$type": "SK_Metamodel.Wall, SK_Metamodel",
  "name": "mur_nord",
  "type": "Revetement",
  "variables": [
    {
      "$type": "SK_Metamodel.Transform, SK_Metamodel",
      "value": {
        "scale": {
          "X": {"Value": 36.163421630859375, "Unit": "Foot"},
          "Y": {"Value": 0.82291666666666652, "Unit": "Foot"},
          "Z": {"Value": 16.586512991556162, "Unit": "Foot"}
        },
        "position": {
          "X": {"Value": 0.0, "Unit": "Foot"},
          "Y": {"Value": 0.0, "Unit": "Foot"},
          "Z": {"Value": 0.0, "Unit": "Foot"}
        },
        "rotation": {
          "X": 0.0,
          "Y": 0.0,
          "Z": 0.0
        }
      },
      "control": {
        "scale": {
          "X": "[Jatko].[dimensionX]",
          "Y": "[Jatko].[covering_depth]",
          "Z": "[Jatko].[Walls].[mur_nord].[scale].__Z"
        },
        "position": {},
        "rotation": {}
      }
    },
    {
      ...
    }
  ],
  "utilityVariables": [ ... ],
  "children": [ ... ]
```

Métamodèle

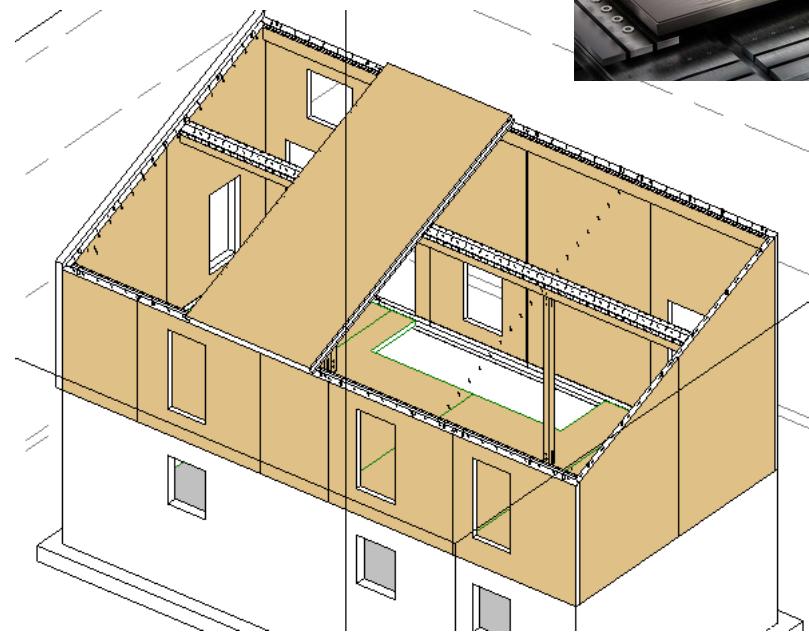
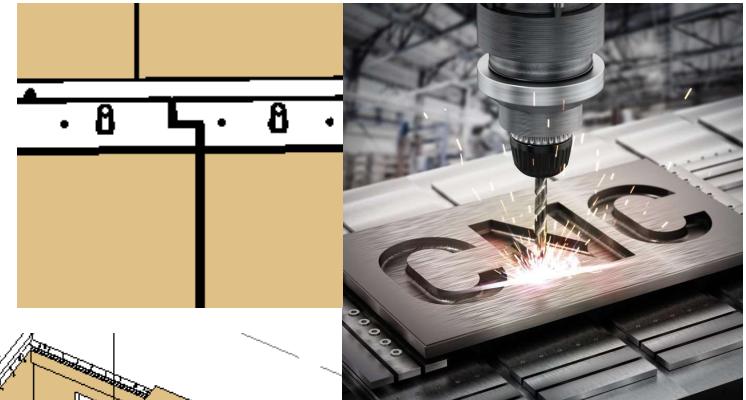
Adoption de l'industrie 4.0

Adaptabilité des données aux systèmes existants.

Avec le système de variables, chaque élément connaît sa représentation réelle.

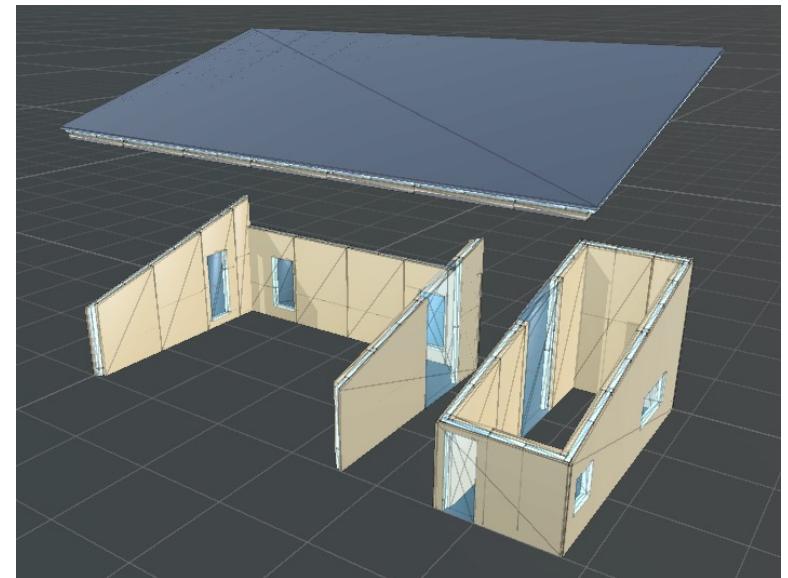
Si on désire découper l'élément à l'aide d'une CNC, on se sert de variables significatives, plutôt que de la géométrie 3D de l'objet uniquement.

Clé pour relier les panneaux.



Plan du séminaire

1. Présentation de Sokio - vision 4.0 et besoins pour y arriver
2. Introduction au concept de métamodèle
- 3. Configuration intelligente de produits**
4. Automatisation des tâches / Optimisation
5. La suite des choses



Gestion des variables

```
[Serializable]  
40 references  
public abstract class Element  
{  
    public string name;  
    public Types type;  
    public List<ObservableVariable> variables;  
    public List<Element> children;
```

Gestion des variables

Valeur simple.

```
{  
  "name": "covering_depth",  
  "value": {"Value":3,"Unit":"Inch"}  
},
```

Référence.

```
"scale": {  
  "X": "[Jatko].[dimensionX]",  
  "Y": "[Jatko].[covering_depth]"  
},
```

Valeur de base + contrôle.

```
{  
  "name": "dimensionX",  
  "value": {"Value":36,"Unit":"Foot"},  
  "control": "[Jatko].[Walls].[mur_est].[position]._Y_"  
},
```

```
{  
  "name": "is_facing_north",  
  "value": true  
},  
{  
  "name": "opposite_wall",  
  "value": "mur_sud"  
}
```

```
List<ObservableVariable> variables;
```

Variable simple.

```
public string name;  
public T value;  
public string control;
```

Variable complexe.

```
public TransformValue value;  
public TransformControl control;  
  
public TransformValue(  
    XYZ<Length>? position,  
    XYZ<double>? rotation,  
    XYZ<Length>? scale)  
{
```

Démo!

Plan du séminaire

1. Présentation de Sokio - vision 4.0 et besoins pour y arriver
2. Introduction au concept de métamodèle
3. Configuration intelligente de produits
- 4. Automatisation des tâches / Optimisation**
5. La suite des choses



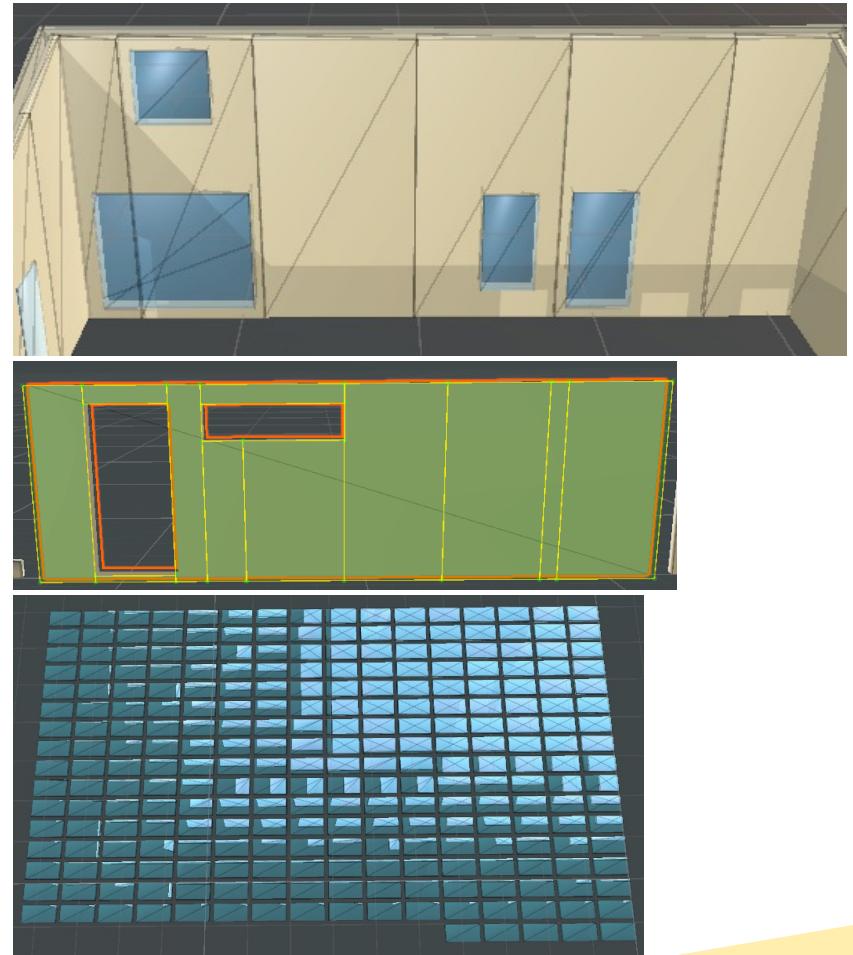
Automatisation

Certaines tâches sont à refaire au développement de chaque nouveau chalet:

- Dimensionnement des panneaux de CLT
- Découpe de l'isolant.

Certaines sont à refaire pour chaque chalet individuel:

- “Nesting” de l'isolant.

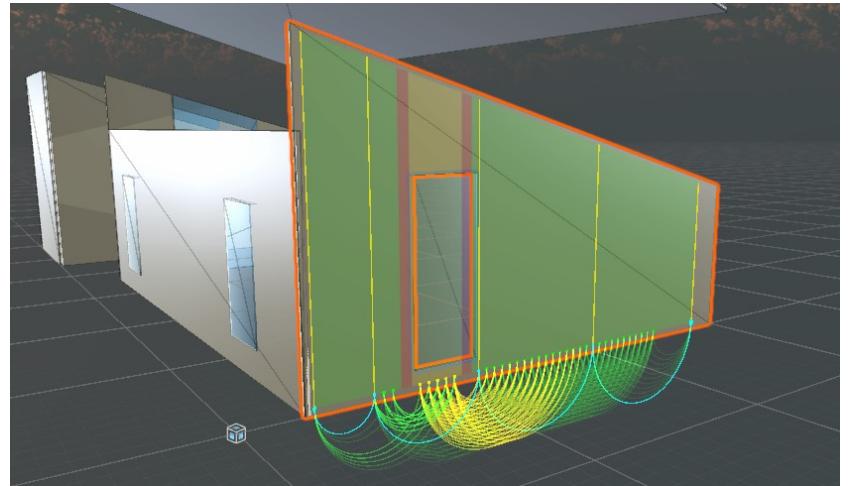


Panneaux

Puisque les panneaux sont visibles de l'intérieur, il est intéressant de pouvoir la visualiser en temps réel.

Donc, l'algorithme d'optimisation des panneaux doit pouvoir être exécuté à chaque modification, et ne doit pas avoir de latence.

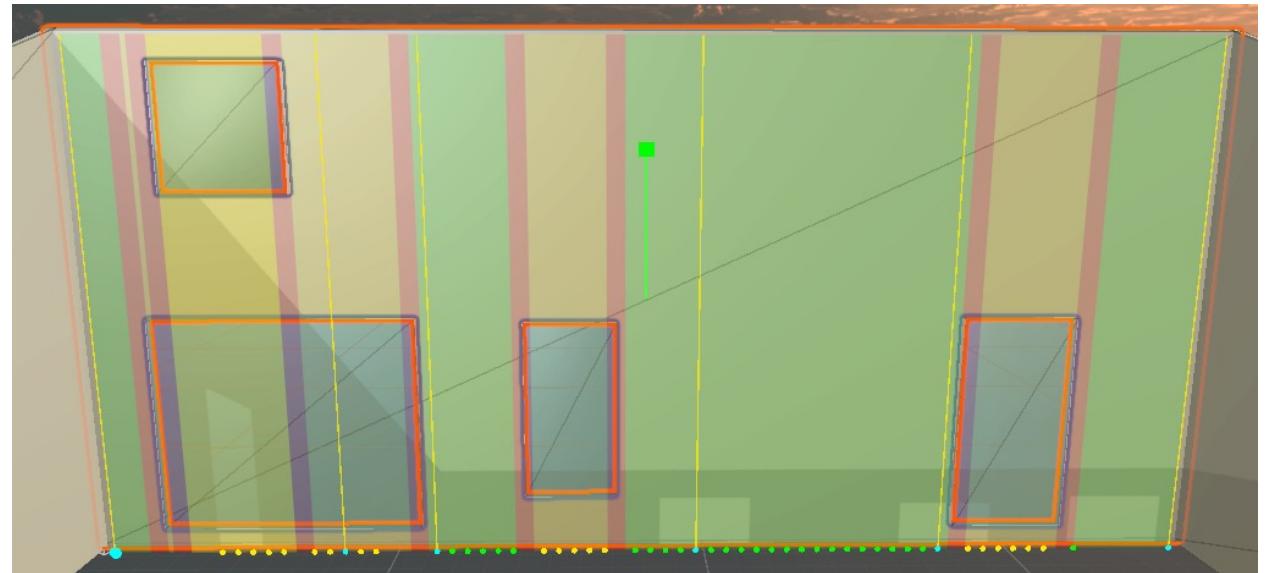
Comment procéder?



Panneaux

Modélisation du problème sous forme de recherche du plus court chemin dans un graphe!

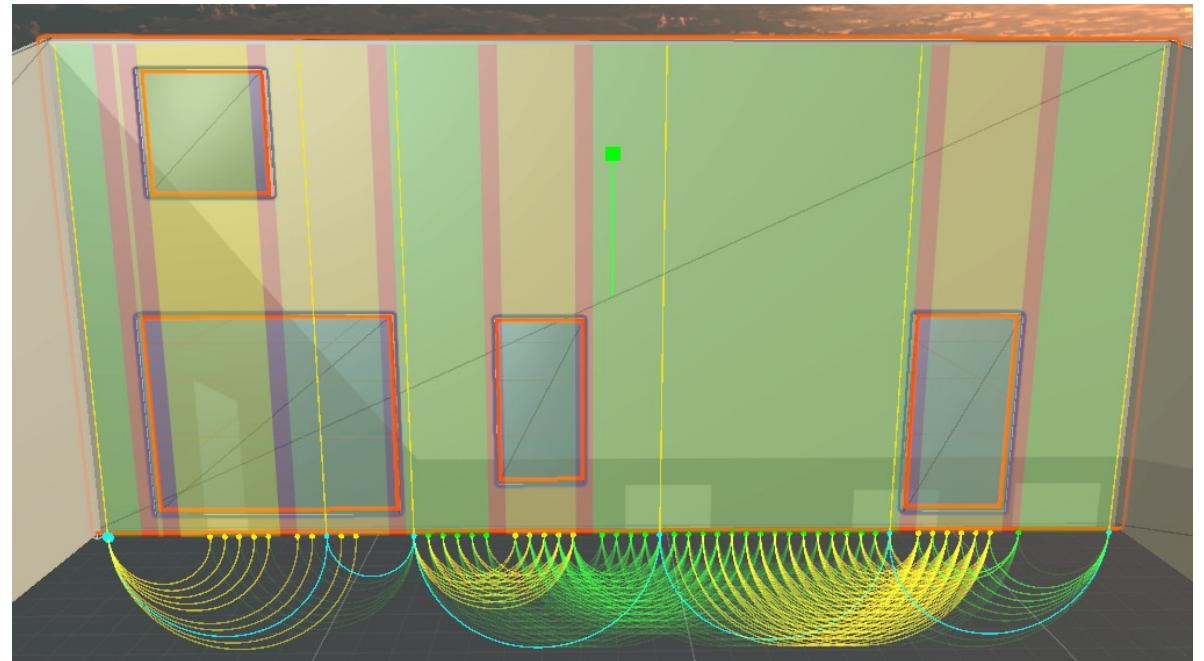
Chaque nœud (espacés de 6 pouces) représente une découpe possible.



Panneaux

Chaque arc représente un lien entre deux possibles découpes. Donc, un arc représente un panneau.

Les nœuds sont reliés au nœuds qui sont entre 2pi 6po et 8pi plus loins.

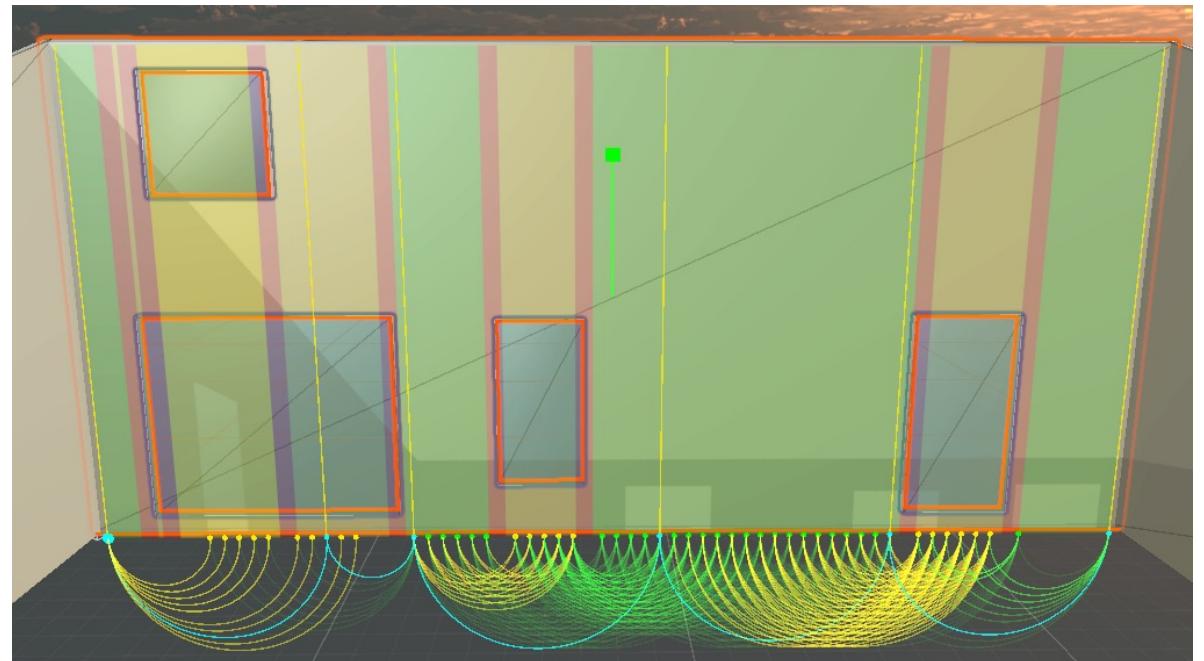


Panneaux

Le graphe est dirigé, acyclique et déjà trié.

Nous avons donc uniquement l'étape de relaxation à calculer pour trouver le plus court chemin.

L'étape de relaxation se calculant en $O(V + E)$, on a un algorithme très efficace!

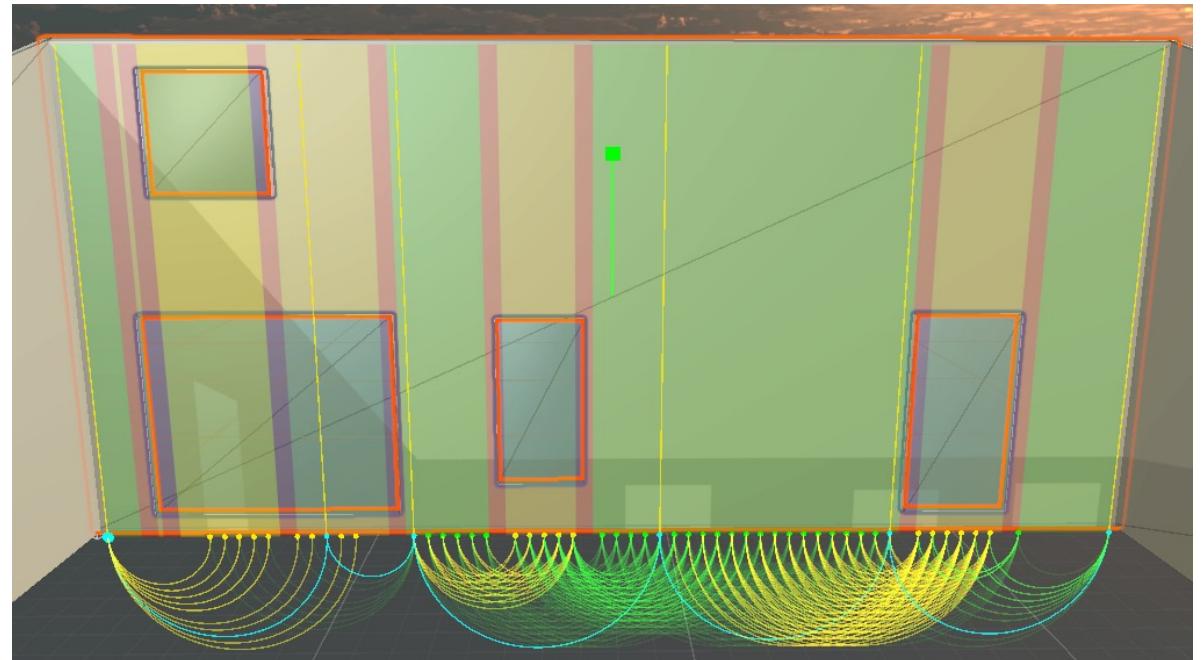


Panneaux

En théorie, les nœuds sont créés en $O(V)$ et les arcs en $O(V^2)$ en pire cas. L'algorithme serait donc en $O(V^2)$ plutôt qu'en $O(V + E)$.

En pratique, avec un espace de 6 pouces entre les nœuds, nous obtenons les mêmes résultats que ceux de Sokio assez rapidement pour être temps réel.

Un plus petit espace se fait ressentir!



Panneaux

Une fois les panneaux découpés, le métamodèle peut être exporté, et il les éléments ajoutés s'y trouveront.

Ils seront réutilisés dans les systèmes plus bas dans la chaîne!

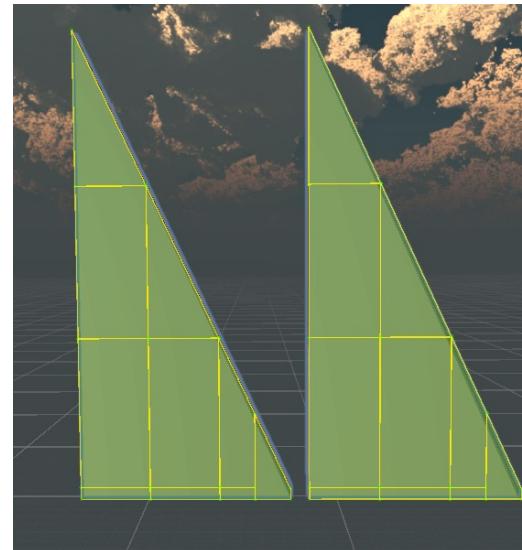
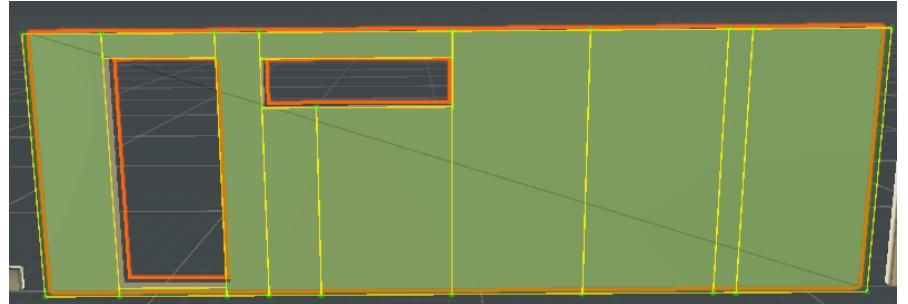
```
{  
  "$type": "SK_Metamodel.Panel, SK_Metamodel",  
  "name": "panel_1",  
  "type": "PanneauCLT",  
  "variables": [  
    {  
      "$type": "SK_Metamodel.AngleHeightTransform, SK_Metamodel",  
      "value": {  
        "scale": {  
          "X": {"Value": 102.0, "Unit": "Inch"},  
          "Y": {"Value": 6.0, "Unit": "Inch"},  
          "Z1": {"Value": 16.246059840842754, "Unit": "Foot"},  
          "Z2": {"Value": 12.387590799424128, "Unit": "Foot"}  
        },  
        "position": {  
          "X": {"Value": 9.0, "Unit": "Inch"},  
          "Y": {"Value": -9.0, "Unit": "Inch"},  
          "Z": {"Value": 0.0, "Unit": "Meter"}  
        }  
      }  
    },  
  ]  
}
```

Isolant

Pour la découpe de l'isolant, le problème est abordé de manière assez naïve.

Aucune optimisation pour l'instant, donc ne nous y attardons pas.

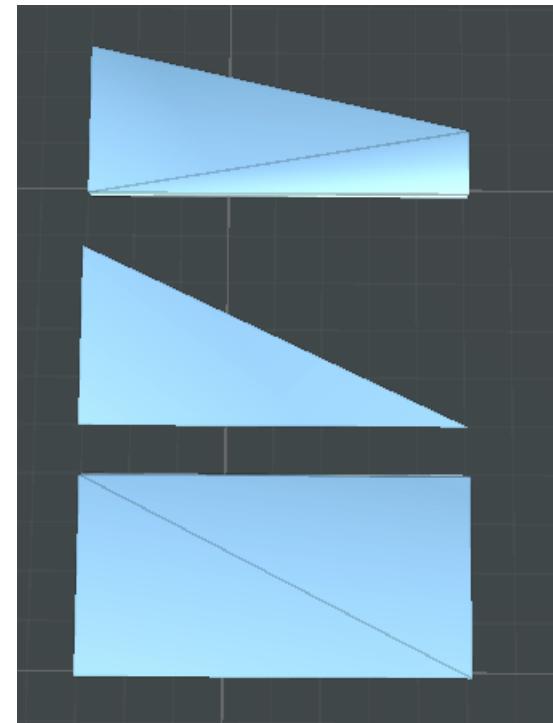
Cependant intéressant de s'attarder au type de forme découpé!



Isolant

L'isolant peut être découpé selon une forme respectant les conditions suivantes:

- Composante rectangulaire de hauteur ≥ 0
- Composante triangle-rectangle de hauteur ≥ 0
- Les deux composantes ne doivent pas avoir une hauteur de 0.

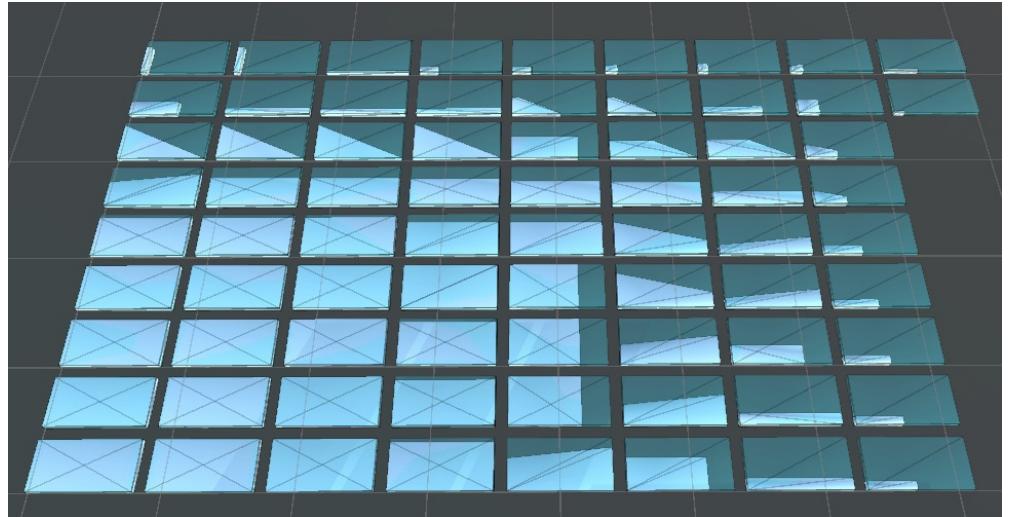


Isolant

Pour le nesting de l'isolant, le problème est assez différent, et beaucoup plus complexe!

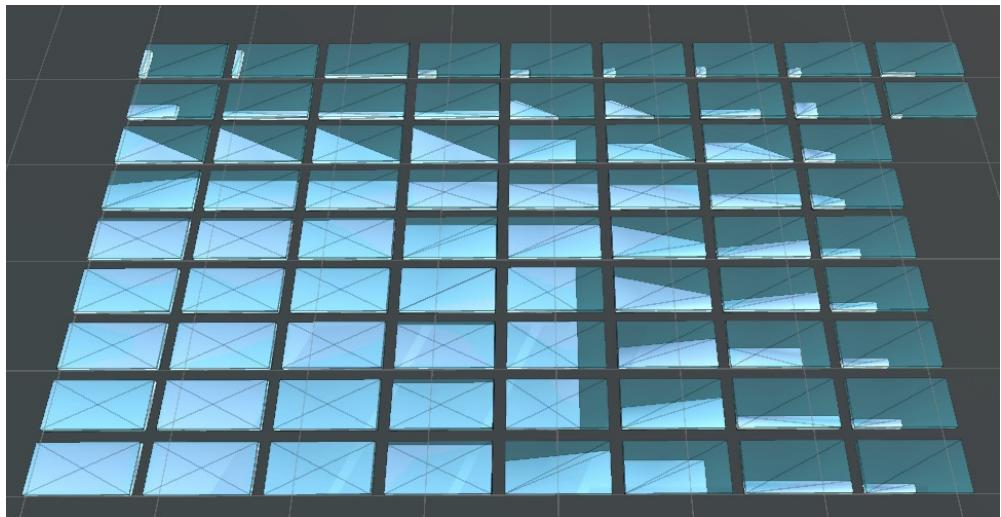
2 objectifs:

- Minimiser le nombre de feuilles utilisées
- Maximiser la réutilisabilité des pertes occasionnées



Isolant

Afin d'optimiser ces différents objectifs, nous utilisons deux modèles de programmation par contraintes Minizinc de manière séquentielle.



Minizinc

Un modèle Minizinc, c'est:

- Des variables
- Des contraintes
- Une fonction objectif
- Un solveur qui tente de trouver la solution avec la meilleure valeur objective
- Et une instance!

```
numPieces = 73;  
numLossesFromDatabase = 0;  
  
widths = [768, 768, 768, 768, 768,  
          768, 768, 544, 544, 524, 470, 441  
          117, 361, 1076, 382, 544, 633, 71  
heights = [1536, 1536, 1536, 1536,  
           1105, 547, 547, 546, 546, 1529, 1  
           1529, 172, 285, 1529, 1536, 1536,
```

```
array[PIECES] of int: widths;  
array[PIECES] of int: heights;  
array[PIECES] of var 0..containerWidth: x;  
array[PIECES] of var 0..containerHeight: y;
```

```
constraint diffn(x, y, rotatedWidths, rotatedHeights);  
constraint cumulative(x, rotatedWidths, rotatedHeights, containerHeight);
```

```
numSheetsUsed = max(sheets);  
solve minimize numSheetsUsed;
```

Chuffed (Chuffed 0.13.2)

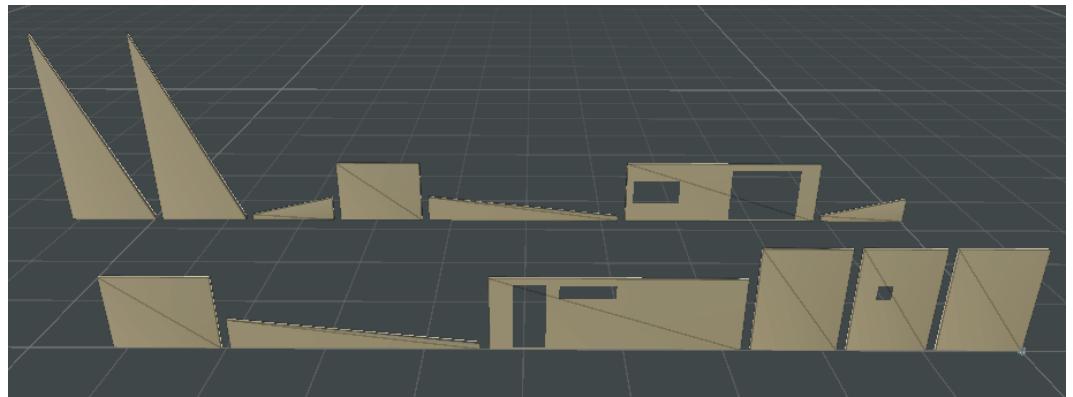
ORTools (OR Tools CP-SAT 9.11.4210)

Isolant

Minimisation de feuilles

Voici les résultats sur cette instance de 13 panneaux. Ici, Sokio ont (manuellement) calculé un minimum de 46 feuilles.

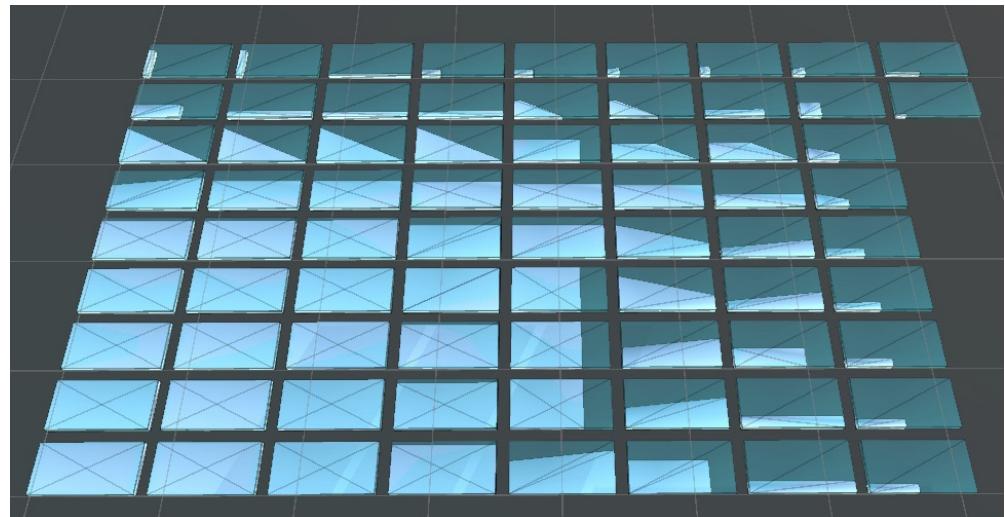
Pour ce calcul “benchmark”, uniquement de nouvelles feuilles ont été utilisées.



Isolant

Les contraintes utilisées dans le modèle Minizinc prennent en entrée des formes rectangulaires uniquement.

Il faut donc appliquer un prétraitement sur notre forme trapézoïdale.

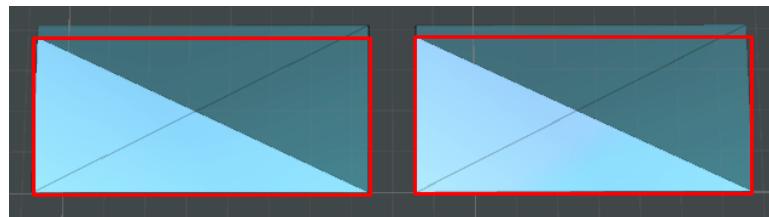


Isolant

Prétraitement 1 – Bounding boxes

Ici, on considère la forme comme son rectangle la bornant entièrement.

74 pièces, 44 feuilles avec optimal en 3 secondes.



Isolant

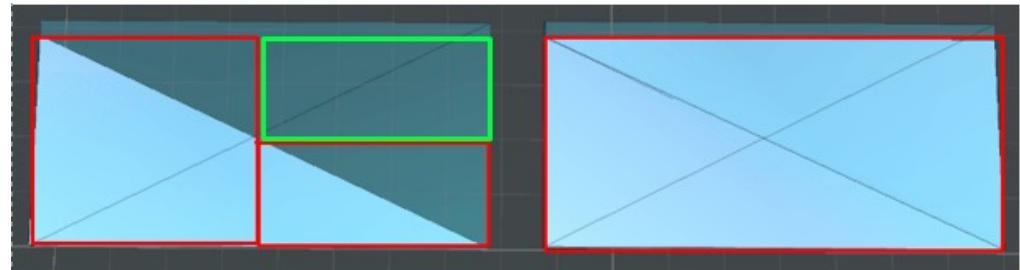
Prétraitement 2 – Rectangle subdivision

Ici, on considère une structure en escalier, avec un nombre de “steps” variable.

Les différentes boîtes sont reliées ensemble dans le modèle, elles ne peuvent pas bouger individuellement.

74 pièces, 44 feuilles avec optimal en 33 secondes - 2 subdivisions.

74 pièces, 43 feuilles avec optimal en 85 secondes - 4 subdivisions.

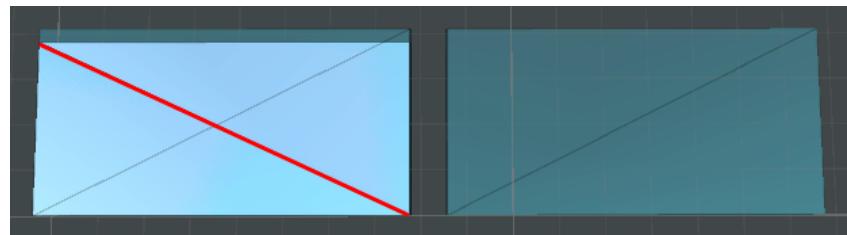


Isolant

Prétraitement 3 – Triangle matching

Ici, on crée un rectangle avec deux triangles, si leur pente coïncide. Les triangles restants sont considérés avec le traitement BB.

67 pièces, 41 feuilles avec optimal en 1 secondes.

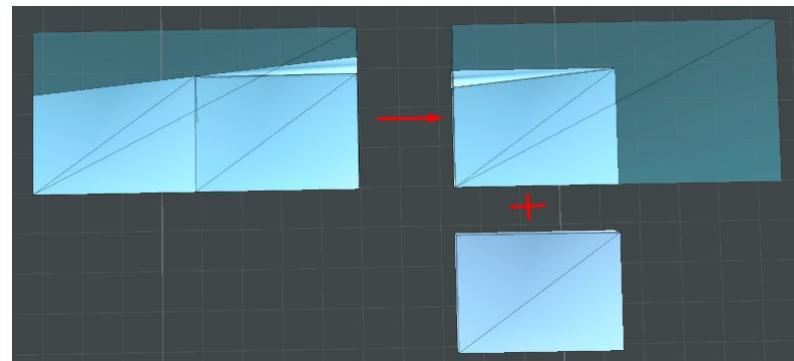
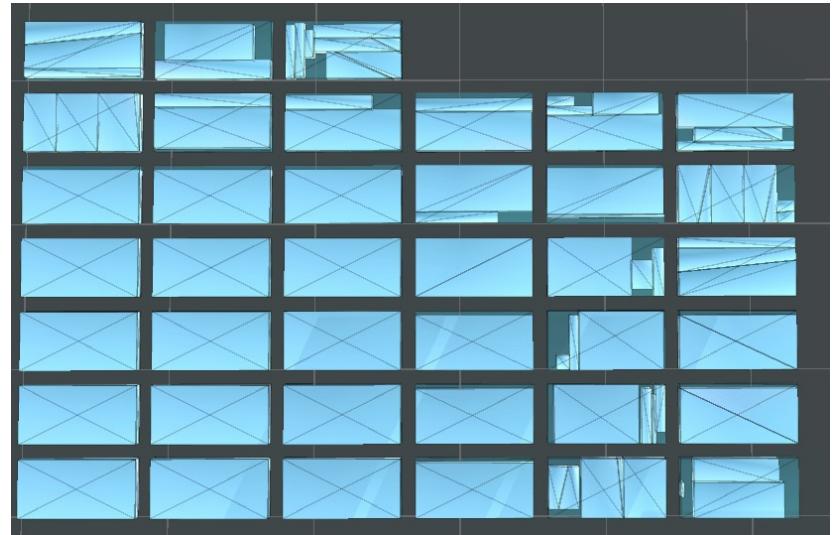


Isolant

Prétraitement 4 – Triangle matching ++

Ici, prétraitement 3, et on coupe les triangles restants pour venir créer des rectangles supplémentaires.

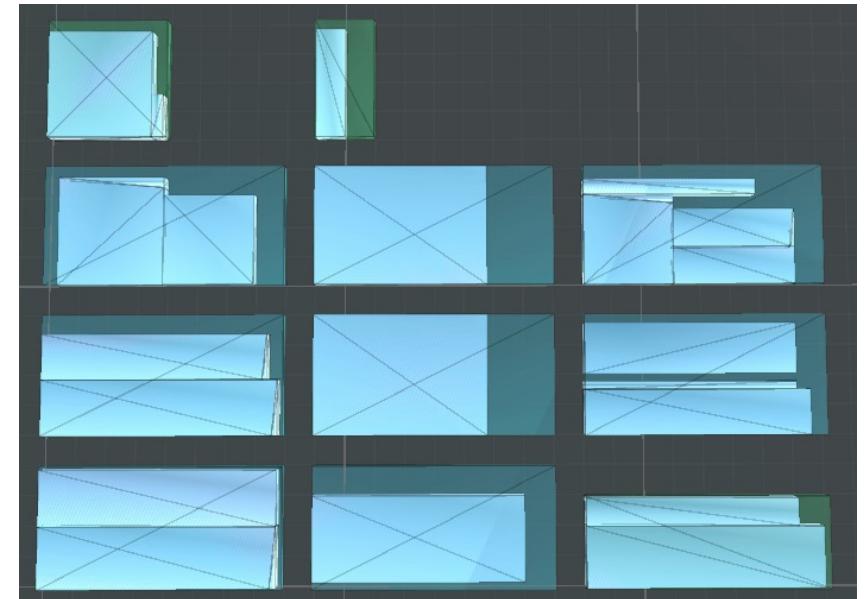
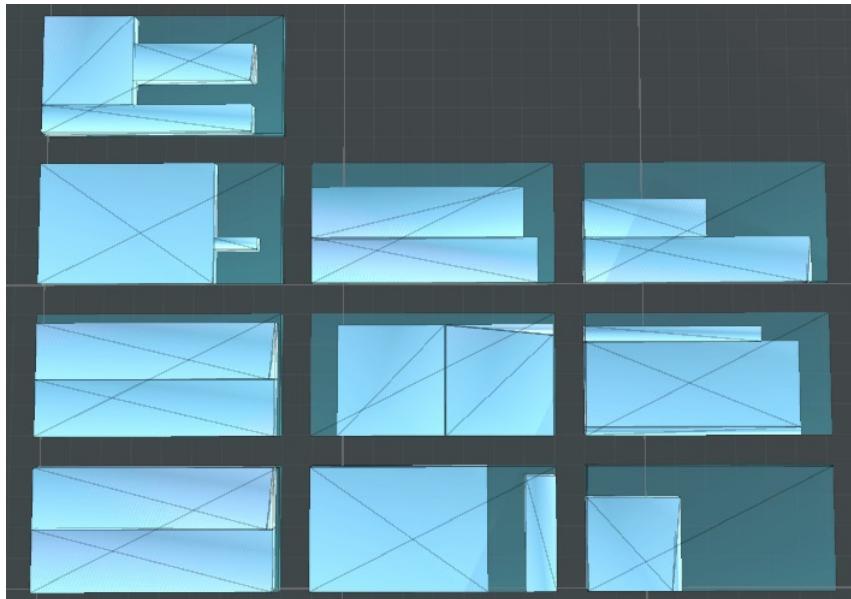
73 pièces, 39 feuilles avec optimal en 10 secondes.



Isolant

Réutilisation des pertes

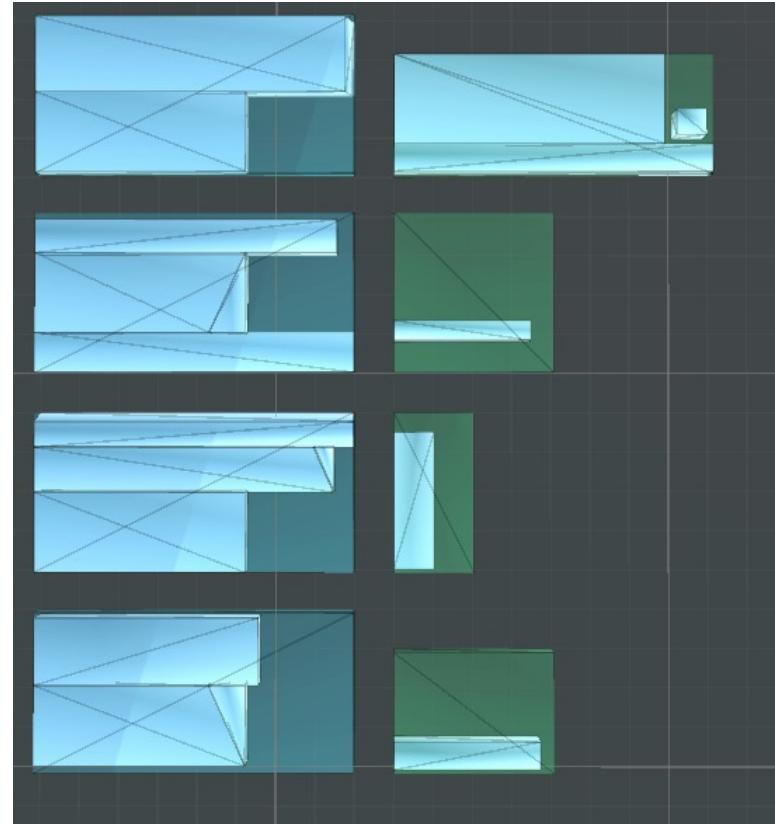
La découpe de l'isolant génère beaucoup de pertes, qui peuvent être réutilisées!



Isolant

Maximisation de la taille des pertes

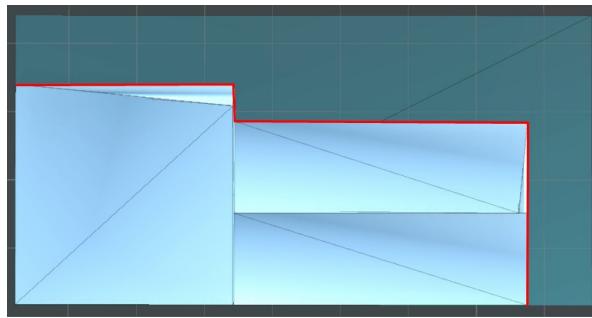
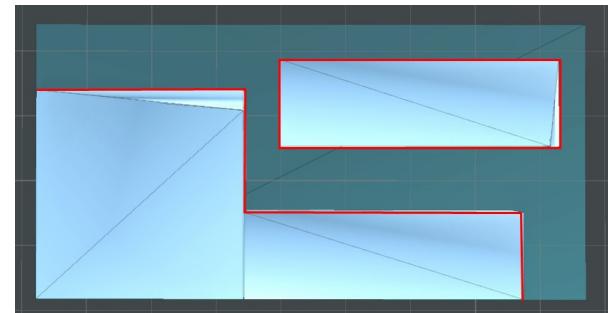
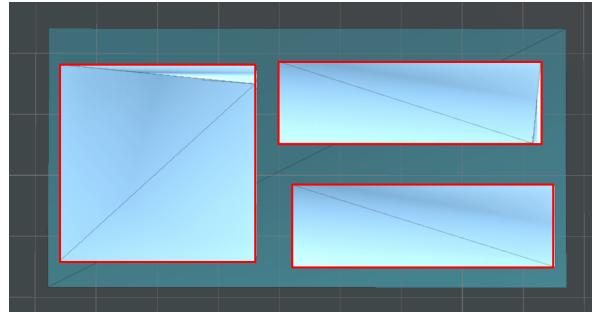
C'est donc intéressant de créer des pertes qui sont facilement réutilisables, d'où l'utilisation d'un 2^e modèle Minizinc.



Isolant

Maximisation de la taille des pertes

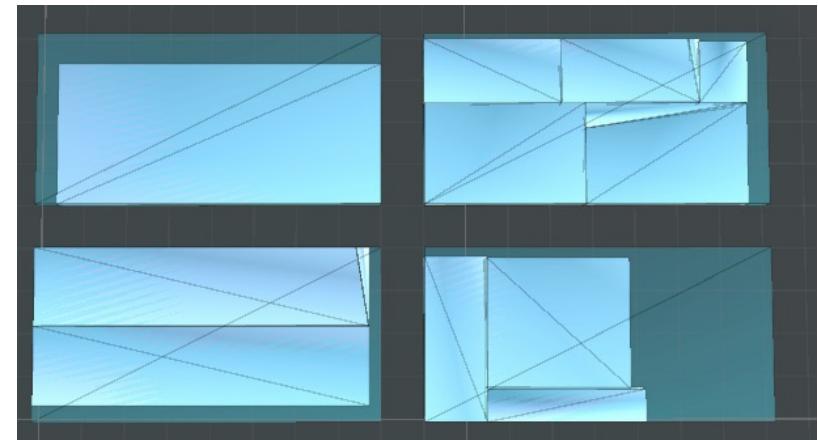
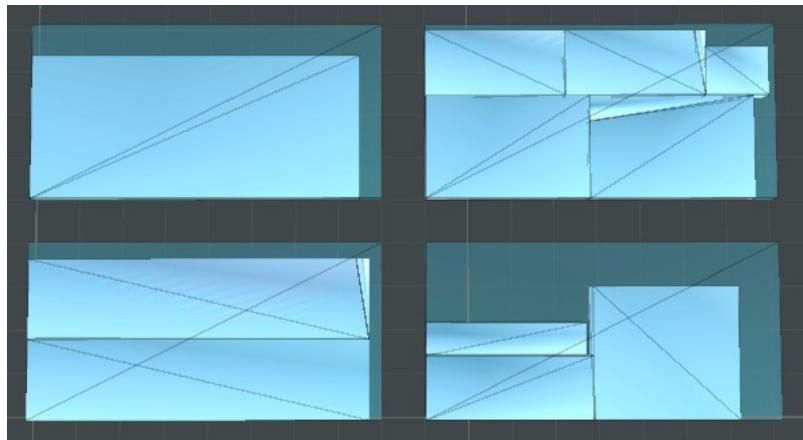
Pour s'y prendre, le modèle est le même, mais nous minimisons une valeur différente, le « périmètre interne » des pièces à découper.



Isolant

Maximisation de la taille des pertes

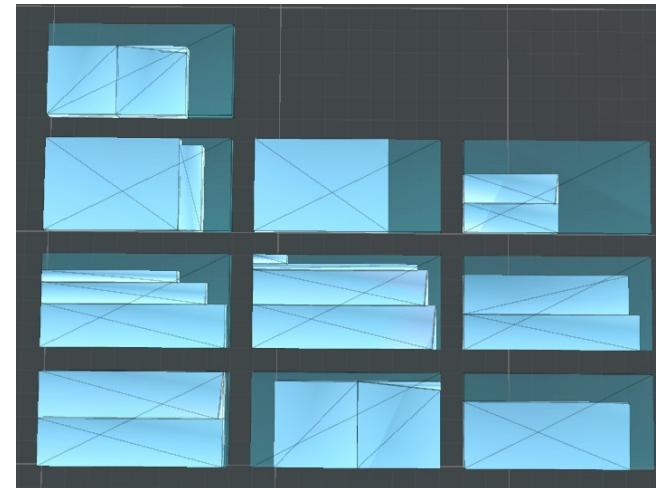
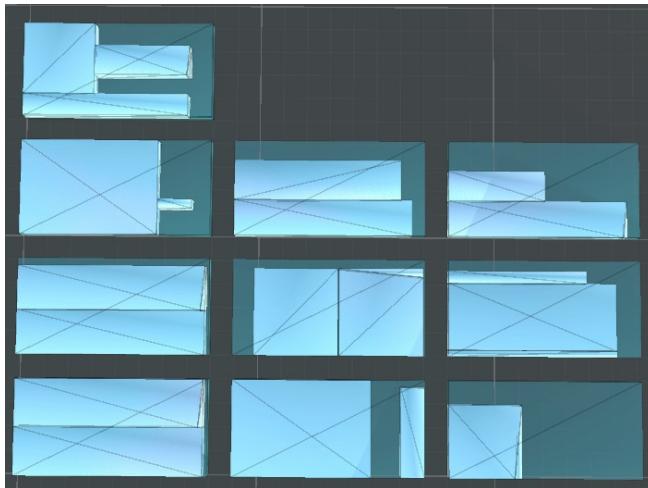
Au final, avec cette simple modification,
les résultats sont grandement améliorés!



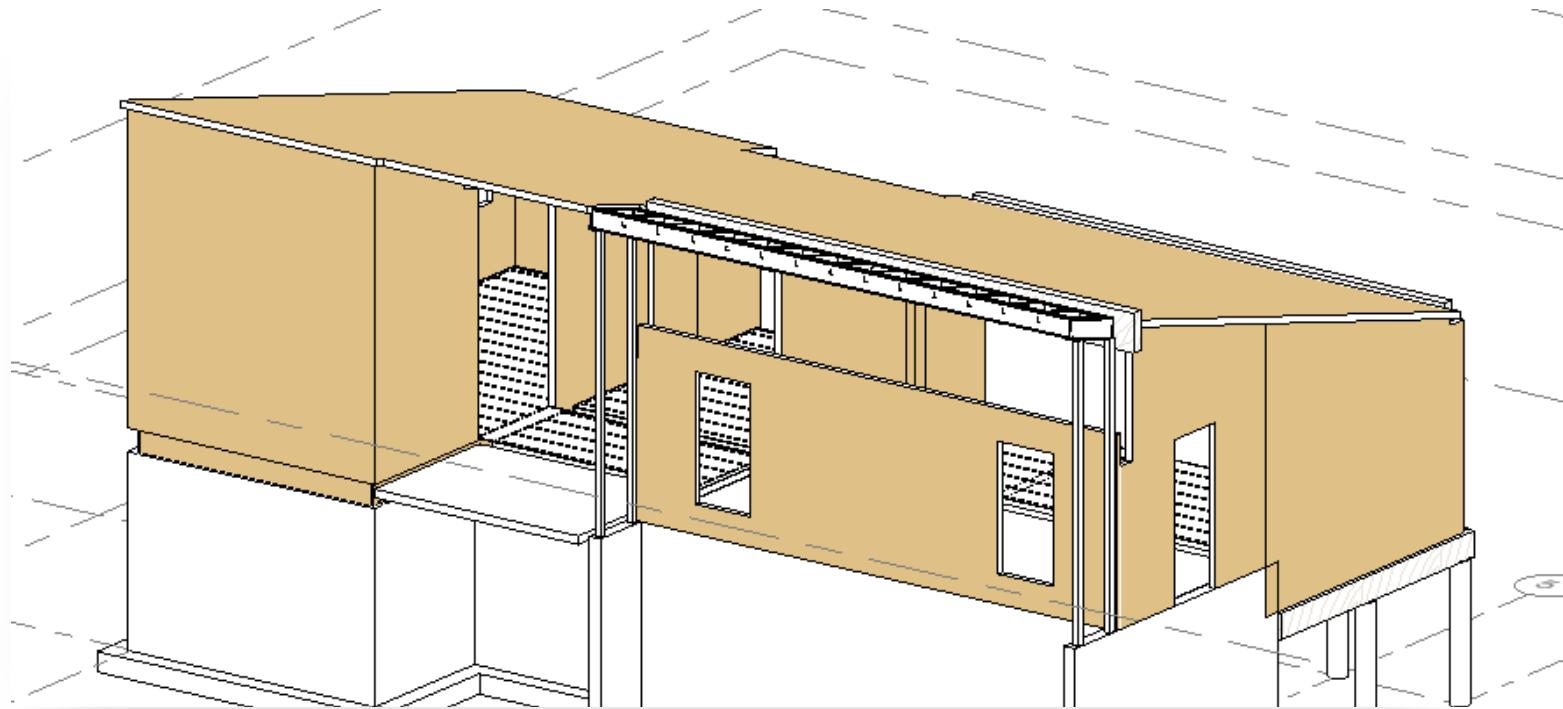
Isolant

Maximisation de la taille des pertes

Au final, avec cette simple modification,
les résultats sont grandement améliorés!



La suite?



- Exportation d'un bâtiment Revit sous sa forme paramétrique (métamodèle).
- Génération automatique des plans.

Questions / Discussion