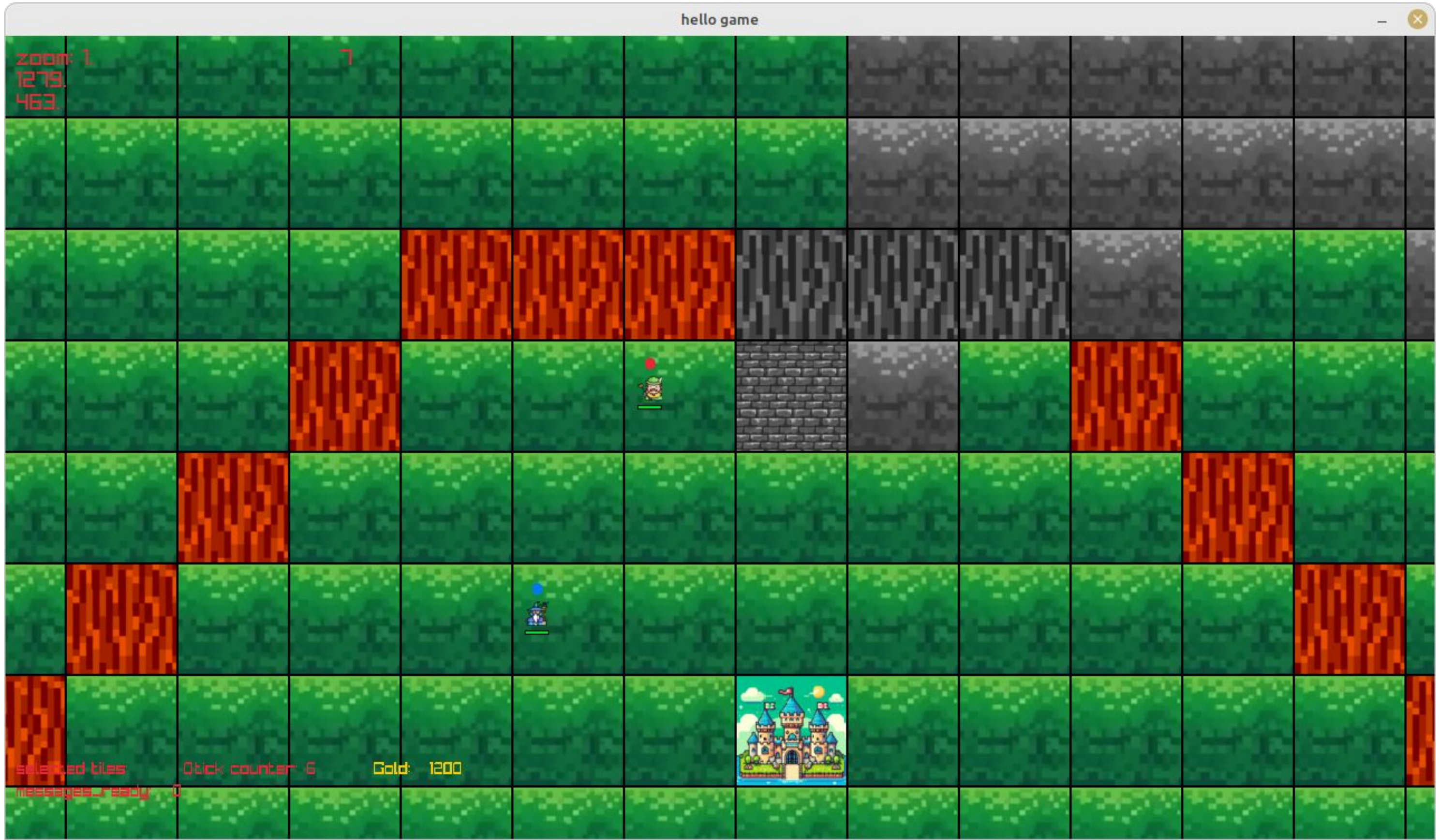
Projet Long

Mathieu Crocombette--Pasquet
Philippe Hinault

Présentation général du jeu

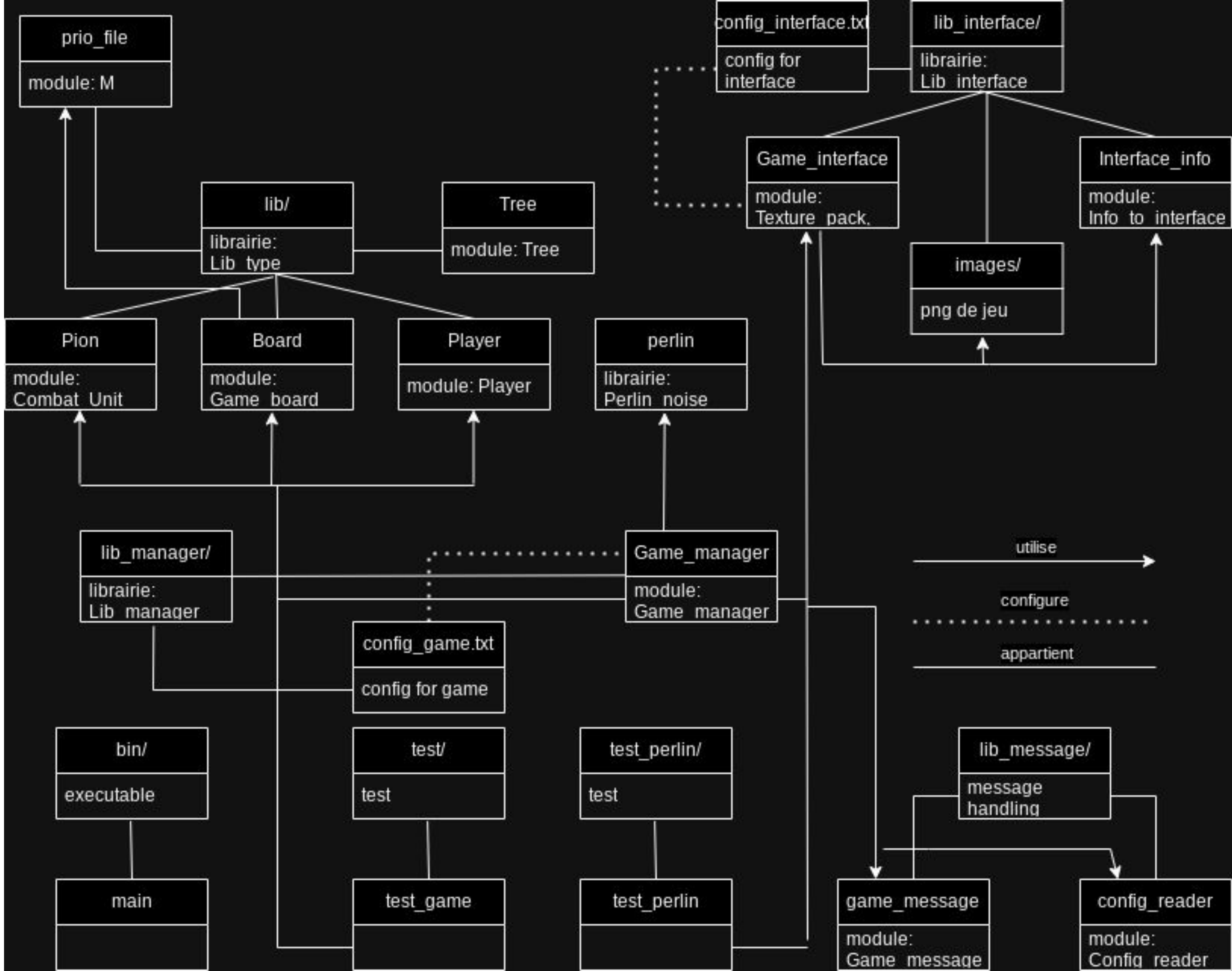


```
zoom: 1.          7
1279.
463.

selected tiles        0tick counter 6        Gold:  1200
messages_ready   0
```

Architecture

Diagramme de classe



Programmation :

Evolutive

- Multijoueur
- Génération aléatoire

Robuste

- Code testé
- Code sûr

Modulaire

- Structure de données

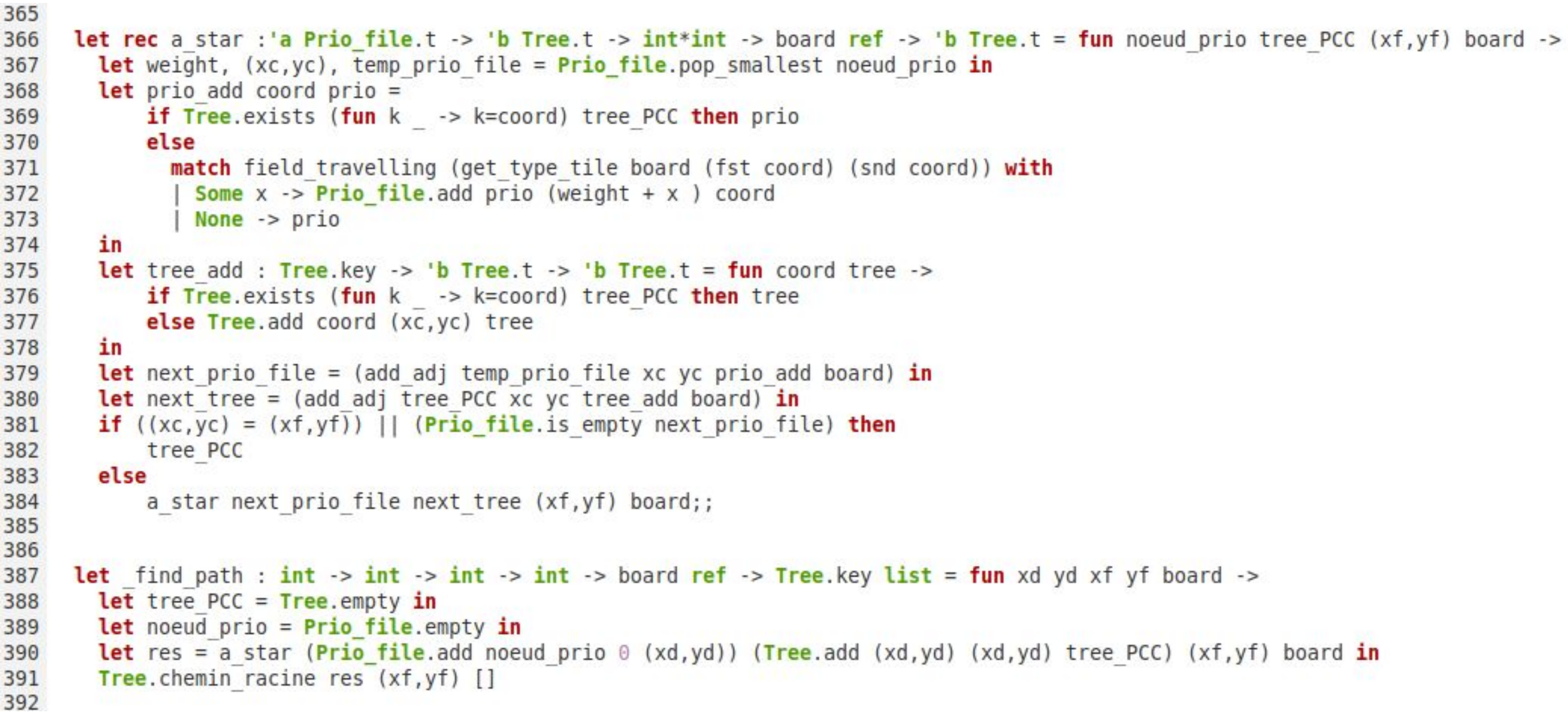
Architecture

Difficultés rencontrées

- Redondance des données

- Gestion de l’interface utilisateur

Programmation



```ocaml

let rec a_star :'a Prio_file.t -> 'b Tree.t -> int*int -> board ref -> 'b Tree.t = fun noeud_prio tree_PCC (xf,yf) board ->
 let weight, (xc,yc), temp_prio_file = Prio_file.pop_smallest noeud_prio in
 let prio_add coord prio =
   if Tree.exists (fun k _ -> k=coord) tree_PCC then prio
   else
    match field_travelling (get_type_tile board (fst coord) (snd coord)) with
    | Some x -> Prio_file.add prio (weight + x ) coord
    | None -> prio
 in
 let tree_add : Tree.key -> 'b Tree.t -> 'b Tree.t = fun coord tree ->
   if Tree.exists (fun k _ -> k=coord) tree_PCC then tree
   else Tree.add coord (xc,yc) tree
 in
 let next_prio_file = (add_adj temp_prio_file xc yc prio_add board) in
 let next_tree = (add_adj tree_PCC xc yc tree_add board) in
 if ((xc,yc) = (xf,yf)) || (Prio_file.is_empty next_prio_file) then
   tree_PCC
 else
   a_star next_prio_file next_tree (xf,yf) board;;


let _find_path : int -> int -> int -> int -> board ref -> Tree.key list = fun xd yd xf yf board ->
 let tree_PCC = Tree.empty in
 let noeud_prio = Prio_file.empty in
 let res = a_star (Prio_file.add noeud_prio 0 (xd,yd)) (Tree.add (xd,yd) (xd,yd) tree_PCC) (xf,yf) board in
 Tree.chemin_racine res (xf,yf) []

```

Conclusion

Ce que nous ajouterions

- Une IA avec un algorithme d’élagage
- Un mode multijoueur

Ce que nous changerions

- Supprimer la redondance des données
- Meilleur organisation du travaille