



Méthode formelle de conception : Système de contrôle du trafic aéroportuaire

Jules HELLER - Abdelkader BELDJILALI



Table des matières

Introduction	2
1 Généralités	2
2 Expression textuelle des exigences	3
2.1 Description du système et de son environnement	3
2.2 Analyse des spécifications	3
2.3 Reformulation des exigences	4
3 Première modélisation abstraite	4
4 Premier raffinement	6
5 Second raffinement	6
Conclusion	6

Introduction

L'objet de ce TD est de modéliser un contrôleur automatique de trafic sur aéroport en interaction avec son environnement et de prouver formellement la correction du modèle formel à l'aide de l'outil rodin associé au plugin Atelier B

1 Généralités

Le système est modélisé sous forme d'une machine à états-transitions. Les états sont caractérisés par des constantes respectant des axiomes et des variables respectant des conditions nommées "invariants". Les constantes caractérisent le "contexte" du système i.e. l'aspect statique. Les variables décrivent l'aspect dynamique. Les transitions sont désignées par le terme event. Les events sont des actions conditionnées ou non par une garde. Une garde est une condition qui autorisera ou non l'action du système. La transition du système est supposée instantanée et se produire dès que la garde est vraie. Il peut survenir des cas d'indéterminisme externe lorsque plusieurs gardes sont vraies simultanément. Si elles sont toutes fausses, le système est bloqué. Si, à tout instant, une garde est vraie, le système ne se termine jamais.

La démarche itérative proposée dans l'exercice consiste à écrire les exigences a partir des spécifications ; chaque exigence est identifiée par un label qui permet la traçabilité et le classement en catégories notamment fonctionnelle et environnementale. On part d'un modèle abstrait simple, formalisé dans l'outil rodin. Des preuves notamment d'invariance sont effectuées. Le plugin Atelier B génère des obligations de preuves dans un langage correspondant à la logique du premier ordre associée à la théorie des ensembles. Atelier B utilise notamment une base de règles d'inférence associée à un moteur contenant des heuristiques et un prouveur de type SAT.

Le modèle formel est corrigé le cas échéant par ajout de gardes ou de conditions sur les gardes si on cherche à éviter les blocages. Les exigences sont complétées si nécessaire. Enfin par raffinement successifs, on passe d'une vision synoptique du système à une vue plus détaillée qui peut conduire jusqu'à la phase d'implémentation.

2 Expression textuelle des exigences

2.1 Description du système et de son environnement

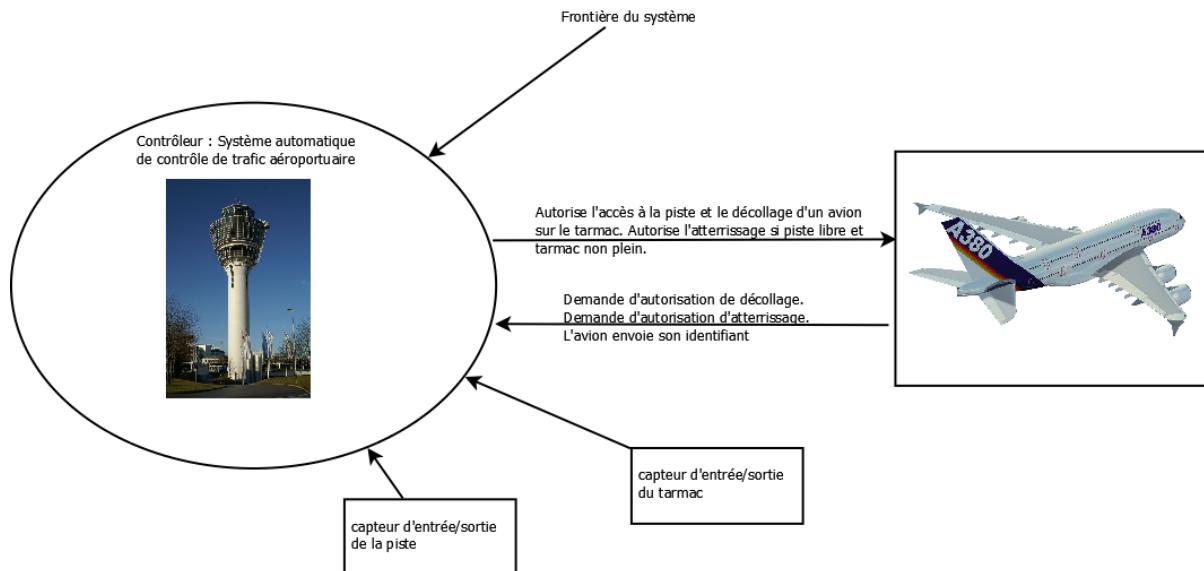


FIGURE 1 – Diagramme hiérarchique des fonctions

2.2 Analyse des spécifications

Le système à concevoir est un **contrôleur aérien automatique** chargé de contrôler le trafic sur le tarmac et d'autoriser l'accès à la piste des avions prêts au décollage.

Ce système est un logiciel qu'on appellera le Contrôleur. Ce contrôleur interagit avec son environnement à savoir les pilotes et des capteurs non précisés dans l'énoncé. Les spécifications sont incomplètes car le contrôleur doit forcément autoriser ou refuser les avions à l'atterrissement également. Il est nécessaire d'avoir un capteur qui permet de compter les avions qui entrent et sortent du tarmac et un autre capteur qui assure le comptage des avions sur la piste. Les avions ne peuvent atterrir que si l'aéroport n'est pas plein, 20 places maximum sont disponibles.

On suppose que les clearances sont données aux avions qui se trouvent sur le tarmac. Dès l'autorisation de décoller donnée, l'avion se rend sur la piste et décolle. Il n'y a pas d'étape intermédiaire. Un système de communication sécurisé de type data-link doit exister pour donner les clearances et recevoir les identifiants et les demandes des pilotes.

On suppose qu'il ne peut y avoir qu'un seul avion sur la piste ; pas de possibilité de mettre deux avions au décollage en même temps.

2.3 Reformulation des exigences

Label	Exigence
FON-1	Le contrôleur doit autoriser les avions à décoller et atterrir
FON-2	Le nombre d'avions sur le tarmac est limité à 20 y compris ceux en attente
FON-3	Des avions entrent et quittent la piste d'atterrissement décollage
FON-4	Des avions entrent sur le tarmac et le quittent
FON-5	La piste ne peut être occupée par plus de un avion
FON-6	Le nombre de décollages ou atterrissages successifs n'est pas limité
FON-7	Le contrôleur doit fixer et délivrer les clearances à l'avance
FON-8	Le contrôleur ne doit autoriser l'avion qu'après l'envoi de son identifiant
FON-9	Le contrôleur doit soit refuser, soit accepter, soit mettre en attente l'avion demandeur
FON-10	Le contrôleur doit refuser la clearance après 10 de mise en attente.
ENV-1	Tout avion se dirigeant vers la piste doit avoir une autorisation de décoller
ENV-2	Le système est muni d'un capteur qui permet de compter les avions sur la piste
ENV-3	Le système est muni d'un capteur qui permet de compter les avions sur le tarmac

TABLE 1 – Tableau des exigences

3 Première modélisation abstraite

On veut initialement considérer très peu d'exigences. L'énoncé nous indique que seules les entrées et sorties de l'aéroport sont étudiées. On effectue donc une étude formelle sur un système très simplifié où la piste et le tarmac ne sont pas pris en compte. Nous étudions uniquement les échanges d'avions entre l'aéroport et l'extérieur ce qu'on modélise en utilisant deux events ou transitions :

- extout lorsqu'un avion sort de l'espace extérieur pour rentrer dans l'espace aéroportuaire,
- extin lorsqu'un avion entre dans l'espace extérieur.

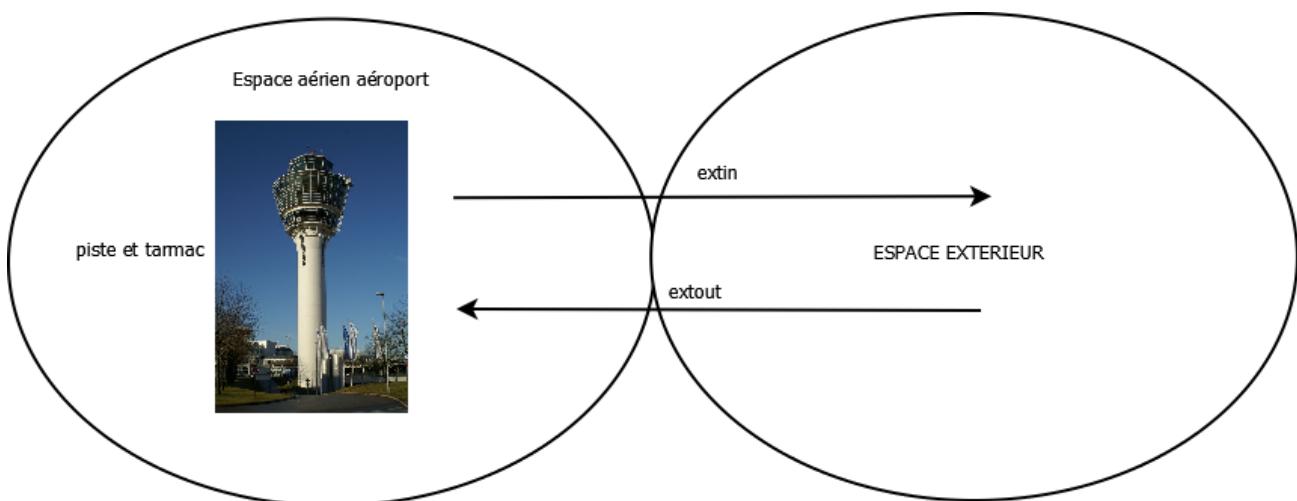


FIGURE 2 – Modèle abstrait

On fait abstraction de la piste. Ainsi, la seule variable à considérer est le nombre d'avions présents sur le tarmac à un instant donné. On note n_t cette variable.

Etat static ou contexte du système : L'état statique est caractérisé par ntmax :

```
CONTEXT
  context0
CONSTANTS
  ntmax
AXIOMS
  axm0_1  :  ntmax ∈ N
  axm0_2  :  ntmax = 20
END
```

FIGURE 3 – Etat static du système

Etat dynamique du système : L'état dynamique du système n'est constitué que de cette seule variable. Elle est définie au moyen de deux conditions ou *invariants* :

```
MACHINE
  machine0
SEES
  context0
VARIABLES
  nt
INVARIANTS
  inv0_1  :  nt ∈ N
  inv0_2  :  nt ≤ ntmax
EVENTS
  INITIALISATION  ≡
  STATUS
    ordinary
  BEGIN
    act1  :  nt = 0
  END

  extout  ≡
  STATUS
    ordinary
  BEGIN
    act1  :  nt = nt + 1
  END

  extin  ≡
  STATUS
    ordinary
  BEGIN
    act1  :  nt = nt - 1
  END

END
```

FIGURE 4 – Etat dynamique du système

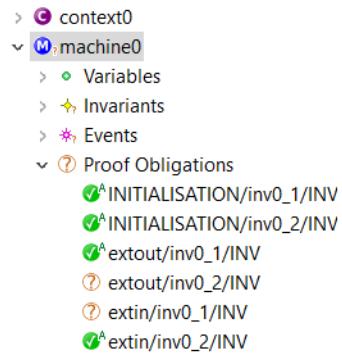


FIGURE 5 – Echec de deux preuves obligatoires

4 Premier raffinement

5 Second raffinement

Conclusion

