HAL open science

Modélisation et validation formelle des règles d'exploitation ferroviaires

Rahma Ben Ayed, Simon Collart-Dutilleul, Philippe Bon, Yves Ledru, Akram Idani

To cite this version:

Rahma Ben Ayed, Simon Collart-Dutilleul, Philippe Bon, Yves Ledru, Akram Idani. Modélisation et validation formelle des règles d'exploitation ferroviaires. Approches Formelles dans l'Assistance au Développement de Logiciels, Jun 2014, France. 15p. hal-01066380

HAL Id: hal-01066380

https://hal.science/hal-01066380v1

Submitted on 19 Sep 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Modélisation et validation formelle des règles d'exploitation ferroviaires

Rahma Ben Ayed[1], Simon Collart-Dutilleul[1], Philippe Bon[1], Yves Ledru[2] et Akram Idani[2]

[1] Univ. Nord de France, IFSTTAR/COSYS-ESTAS, 20 rue Elisée Reclus, F-59650, Villeneuve d'Ascq, France
{rahma.ben-ayed,simon.collart-dutilleul,philippe.bon}@ifsttar.fr
http://www.ifsttar.fr
[2] UJF-Grenoble 1/Grenoble-INP/UPMF-Grenoble 2/CNRS, LIG UMR 5217, F-38041, Grenoble, France
{yves.ledru,akram.idani}@imag.fr
http://www.liglab.fr

Résumé Le système européen de surveillance du trafic ferroviaire (en anglais, European Rail Traffic Management System, ERTMS) est un système complexe de contrôle/commande et de signalisation ferroviaire mettant en œuvre des règles européennes d'exploitation ferroviaires. Cet article propose une étude de cas basée sur deux scénarios extraits de ces règles, un scénario nominal d'autorisation de mouvement et un scénario exceptionnel de franchissement d'un arrêt. En effet, on trouve dans ces scénarios des aspects fonctionnels et de sécurité. Ces aspects nécessitent, d'une part, une modélisation fonctionnelle enrichie par des modèles décrivant la politique de sécurité et les autorisations données aux agents agissant sur le système, et d'autre part, une validation formelle. Pour ce faire, nous avons utilisé la plate-forme B4MSecure, fondée sur l'approche IDM (Ingénierie Dirigée par les Modèles), produisant à partir des modèles UML des spécifications formelles B. L'objectif de ces spécifications résultantes est de valider ces scénarios à l'aide d'outils d'animation et de preuve de spécifications B afin de garantir une analyse rigoureuse de la fonctionnalité et de la politique de sécurité.

1 Introduction

La sécurité des systèmes critiques ferroviaires est un enjeu majeur des systèmes d'aujourd'hui du fait de leur complexité et des conséquences graves pouvant découler d'erreur de conception. C'est pourquoi, leur validation et leur vérification constituent des tâches d'envergure ayant une place prépondérante dans leur cycle de développement. Ce faisant, un éventail de méthodes formelles existe dans l'optique de mener rigoureusement ces activités. Fondées sur des bases mathématiques, ces méthodes peuvent pallier la complexité et l'ambiguïté des spécificités des systèmes critiques, dès lors qu'elles permettent la spécification et le développement de systèmes, ainsi que la validation et la vérification automatique des propriétés de sécurité ferroviaire.

Dans le cadre du projet ANR « Vers la formalisation des exigences ferroviaires et leur traçabilité » (Performing Enhanced Rail Formal Engineering Constraints Traceability, PERFECT), notre travail se focalise sur les systèmes ERTMS et s'oriente vers la modélisation et la validation de leurs aspects fonctionnels et de sécurité par la méthode formelle B. Plusieurs travaux de recherche ont été menés dans le cadre de la validation et de la vérification des spécifications ERTMS par des méthodes et des techniques formelles. Nous pouvons citer le projet *OpenETCS* mené par Systerel [1] grâce à son expertise dans la maîtrise des systèmes complexes et en particulier des méthodes formelles, telles que la méthode B. Il consiste à consolider l'ensemble des spécifications ERTMS avec des méthodologies formelles et des techniques de preuve.

Dans cet article, nous présentons la modélisation de deux scénarios d'autorisation de mouvement des trains et de franchissement d'un arrêt dans le système ERTMS. Cette modélisation comprend un modèle fonctionnel, qui décrit les principaux concepts de ces mouvements de trains, et un modèle qui précise les responsabilités de chaque intervenant (conducteur de train, agent de circulation, ordinateurs embarqué et au sol). Ce deuxième modèle, dit de sécurité, est exprimé comme un modèle de contrôle d'accès, en utilisant l'outil B4MSecure.

Dans la section 2, nous décrivons notre étude de cas comprenant deux scénarios de règles d'exploitation ferroviaires du système ERTMS/ETCS. Ensuite, nous présentons dans la section 3 le modèle fonctionnel et les modèles de sécurité, ainsi que leur transformation en spécifications B à l'aide de la plate-forme B4MSecure dans la section 4. La section 5 montre la validation formelle des spécifications résultantes en utilisant l'animateur ProB et le prouveur Atelier B. Finalement, la section 6 conclut et présente les améliorations que l'on pourrait apporter à notre travail.

2 Étude de cas

ERTMS [2] est un projet industriel majeur implémenté par huit membres d'UNIFE [3] en Europe. Ce projet vise à harmoniser la signalisation ferroviaire en Europe tout en garantissant la sécurité des circulations. En effet, chaque pays possède son propre système de signalisation ferroviaire implémenté et géré par des entreprises ferroviaires nationales. Chaque système est alors considéré comme indépendant et non-interopérable avec les autres systèmes, ce qui provoque un surcoût financier très important dédié au passage de frontières imposant par exemple le changement de locomotive et/ou du système de signalisation embarqué.

Le système ERTMS est composé du système européen de contrôle des trains (en anglais, European Train Control System, ETCS) qui est le système de contrôle/commande, ainsi que le système de communication GSM-R (en anglais, Global System for Mobile communications - Railways) pour la transmission de

1. SYSTEREL : http ://www.systerel.fr/
2. European Rail Traffic Management System : http ://www.ertms.net
3. European Rail Industry : http ://www.unife.org

données entre l'ETCS embarqué (le système à bord du train) et l'ETCS sol (le système au sol).

Afin de respecter des impératifs de sécurité et d'assurer la bonne gestion des circulations, des règles d'exploitation ferroviaires sont définies régissant la sécurité ferroviaire. Ces règles définissent l'ensemble des interactions entre les systèmes «temps réels» embarqués et les opérateurs tels que le conducteur et l'agent de circulation, notamment dans les modes dégradés.

Notre étude de cas est extraite des principes et des règles d'exploitation du système ERTMS/ETCS niveau 2 appliqués à la ligne de grande vitesse LGV-Est Européenne [2] et des spécifications décrites dans [1], disponibles sur le site d'ERA [4]. Nous avons choisi deux scénarios pour notre étude, un scénario nominal d'autorisation de mouvement (en anglais Movement Authority, MA) et un scénario exceptionnel de franchissement d'un arrêt ETCS (en anglais Override EOA). Le document de référence [2] est un document industriel qui n'est pas encore public et n'est pas dans sa version définitive. Seule l'utilisation du système ETCS niveau 2 est concernée par la présente étude.

2.1 Scénario nominal d'autorisation de mouvement (MA)

En ETCS niveau 2, le train reçoit une MA en « mode » nominal. Celle-ci est une autorisation donnée au train de circuler sur une distance donnée en tant que mouvement supervisé. La MA est mise à jour au fur et à mesure que le train avance.

Une MA est la traduction ETCS d'un itinéraire tracé sur l'infrastructure dont tout ou partie est affecté au train. Cette règle de signalisation se base notamment sur la position du train, sur l'occupation de l'infrastructure par d'autres trains, sur des règles d'exploitation de sécurité et sur les tables horaires de chacun des trains (par exemple l'heure d'arrivée en gare), elles-mêmes dépendantes de règles d'exploitation propres à chaque ligne.

La MA est caractérisée par (cf. paquet 15 de [1]) :

La section représente une distance par rapport au repère géographique du train. Elle est composée éventuellement de sous-sections. Une MA peut être appliquée sur une ou plusieurs sections. La dernière section est appelée la section de fin.

La fin d'autorisation de mouvement (End Of Authority, EOA) est le « lieu » jusqu'au quel le train est autorisé à se mouvoir, où la vitesse but indiquée sur l'interface conducteur-machine (Driver Machine Interface, DMI) est égale à zéro. Elle peut correspondre à un repère d'arrêt ETCS.

La vitesse cible à l'EOA est la vitesse autorisée à l'EOA. Lorsque la vitesse cible n'est pas nulle, l'EOA est appelé la limite de l'autorisation de mouvement (Limit Of Authority, LOA). Cette vitesse cible peut être limitée dans le temps.

4. European Railway Agency : http ://www.era.europa.eu

Le point de danger est un point au-delà de l'EOA qui peut être atteint par l'extrémité avant du train sans risque d'une situation dangereuse.

Le délai d'attente est un délai qui peut être attaché à chaque section. Il est utilisé pour la révocation de l'itinéraire associé lorsque le train ne l'a pas encore emprunté. Il peut être aussi attaché à la section de fin de la MA. Dans ce cas, il est utilisé pour la révocation de la dernière section quand elle est occupée par le train.

Ce scénario, décrit en détail dans [11], se déroule en interaction entre la machine responsable de la gestion de sécurité à bord du train appelée *OnboardSafetyManagement* qui demande une MA, la machine responsable de la gestion de sécurité au sol appelée *TracksideSafetyManagement* qui reçoit la demande et propose une MA après avoir effectué des vérifications et le conducteur qui peut lire la MA proposée via le DMI après sa validation par l'OnboardSafetyManagement. Le conducteur doit respecter les consignes à travers le DMI.

2.2 Scénario exceptionnel (Override EOA)

Override EOA est un scénario déclenché par le conducteur dans des situations dégradées spécifiques en absence de MA. Lorsqu'il est activé par le conducteur, ce scénario permet à un train de franchir un repère d'arrêt ETCS ou un EOA après avoir reçu de l'agent de circulation une autorisation Override EOA et de désactiver certaines protections. Cette autorisation est donnée sous forme d'un ordre écrit. Ce dernier est un message de sécurité délivré par l'agent de circulation au conducteur dans le but de fournir des instructions. Il peut être délivré physiquement ou bien faire l'objet d'une transmission verbale par téléphone ou par radio sol train selon les modalités d'application de la réglementation technique de sécurité relative à la communication. Il existe plusieurs types d'ordres écrits d'ETCS01 à ETCS07. Nous pouvons citer par exemple l'ordre écrit ETCS01 d'autorisation de franchir un EOA traité dans notre étude de cas, l'ordre écrit ETCS03 d'obligation de rester à l'arrêt, l'ordre écrit ETCS04 d'annulation de l'ordre écrit ETCS03, etc. Un ordre écrit contient au minimum le type de l'autorisation, le numéro de l'autorisation, l'heure et la date de sa délivrance, le poste qui le délivre, à quel train il s'adresse, à quel endroit il s'applique et une indication claire, précise et sans ambiguïté des actions à effectuer.

Le scénario d'Override EOA est effectué comme suit :

OverrideEOA.1 Le conducteur (Driver) demande un Override EOA à travers le DMI. Il peut aussi faire avancer, freiner, arrêter le train à travers le DMI.

OverrideEOA.2 La machine responsable de la gestion de sécurité à bord du train (OnboardSafetyManagement) traite la demande d'Override EOA et la transmet au système au sol (TracksideSystem).

OverrideEOA.3 L'agent de circulation (TrafficAgent) reçoit la demande du système au sol (TracksideSystem), crée un ordre écrit et l'autorise. Il peut aussi le modifier et/ou le supprimer.

OverrideEOA.4 L'agent de circulation (TrafficAgent) transmet l'ordre écrit autorisé au système à bord (OnboardSystem).

OverrideEOA.5 La machine responsable de la gestion de sécurité à bord du train (OnboardSafetyManagement) traite cette autorisation afin qu'elle soit affichée sur le DMI.

OverrideEOA.6 Le conducteur (Driver) suit les indications et les consignes affichées sur le DMI suite à cette autorisation.

Les ordres et les instructions, entre autres les instructions de MA et d'Override EOA, sont affichés sur le DMI sous forme de messages textuels ou de symboles. Le DMI permet alors la communication entre le système à bord du train et le conducteur. Ce dernier doit respecter les indications et acquitter les consignes sur le DMI. Il peut aussi informer le système en entrant des informations.

Notre étude de cas basée sur ces deux scénarios révèle l'existence d'interactions entre le système et les agents agissant sur le système. Pour bien modéliser ces interactions, une séparation de la fonctionnalité du système et de la politique de sécurité est nécessaire. Cette politique de contrôle d'accès permettra de modéliser qui est responsable de quelle action. Dans ce qui suit, nous décrivons la modélisation fonctionnelle du système renforcée par la politique de sécurité.

3 Modélisation

Le couplage des méthodes formelles, en l'occurrence la méthode B, et semi-formelles telles qu'UML est un sujet étudié depuis des années grâce à leurs complémentarités [7]. D'une part, UML possède un aspect structurel, intuitif et synthétique grâce à l'utilisation de ses diagrammes. D'autre part, la méthode B permet de construire des spécifications précises et détaillées et d'identifier les ambiguïtés mieux qu'en UML. Les diagrammes UML utilisés sont les diagrammes de classes permettant de décrire le système et ses fonctionnalités. L'aspect dynamique permettant la description du comportement du système n'est pas encore abordé dans nos travaux de recherche.

Les deux scénarios de la section précédente ne se réalisent qu'après le déroulement d'une séquence d'actions par des agents qui possèdent des rôles bien déterminés. Chaque action est autorisée pour un agent et ne peut être exécutée que par ce dernier. Ainsi, chaque agent n'a l'autorisation d'exécuter que des actions qui lui sont permises. De ce fait, des permissions associées aux actions et des rôles associés aux agents peuvent être mis en œuvre dans la modélisation de ces scénarios. Ces concepts peuvent découler des principes de contrôle d'accès à base de rôles (Role Based Acces Control, RBAC) [10] appliqués principalement dans les systèmes d'information. Pour cette raison, nous avons eu recours à l'emploi des principes de RBAC dans notre modélisation, et ce, au travers de l'outil B4MSecure.

B4MSecure [5] [9] est une plate-forme Eclipse fondée sur l'approche IDM. Cette plate-forme permet de modéliser graphiquement en UML un diagramme de classes fonctionnel et des diagrammes de classes pour la politique de contrôle d'accès en utilisant le profil RBAC. L'outil transforme automatiquement tous

5. B4MSecure : http ://b4msecure.forge.imag.fr/

ces modèles en des spécifications B dans le but de les valider formellement. Les travaux de recherche menés dans le cadre du projet Selkis [5], [6] et [7] montrent l'utilité et l'efficacité de cette plate-forme pour la validation formelle des scénarios dans le domaine d'un SI médical tout en détectant les séquences d'opérations pernicieuses.

3.1 Modèle fonctionnel

Le modèle fonctionnel permet de décrire les entités du système impliquées dans la réalisation des scénarios, les caractéristiques de chaque entité, les opérations effectuées sur chaque entité et les relations entre les entités. Les principes de modélisation des diagrammes de classes UML : classes, attributs de classes, méthodes de classes, les relations (agrégation, composition, généralisation/spécialisation), ainsi que les associations et les classes associatives permettent la modélisation fonctionnelle désirée. En effet, les classes sont utilisées pour la modélisation des entités. Les attributs de classes et les méthodes de classes sont utilisés respectivement pour la modélisation des caractéristiques des entités et les opérations effectuées sur les entités. Les relations (agrégation, composition, etc.), les associations et les classes associatives sont utilisées pour la modélisation des relations entre les entités.

La Fig. 1 représente le système ETCS (la classe ETCSSystem) contenant le sous-système à bord (la classe OnboardSystem) qui fait partie du train ETCS (la classe TrainETCS) et le sous-système au sol (la classe TracksideSystem). L'autorisation de mouvement et l'ordre écrit sont modélisés par deux classes (les classes MA et ETCSOrder) contenant les attributs qui les caractérisent. Chacun est associé à un train ETCS et est affiché sous forme de consignes sur le DMI, qui fait partie du système à bord, après son traitement et sa validation. Comme décrit dans la section 2.1, une MA est transmise par le systèmes au sol et elle est composée d'une ou plusieurs sections. Sur une section, au maximum un train peut circuler.

3.2 Politique de sécurité

Des permissions sont accordées aux usagers du système selon les rôles qui leur sont attribués. Les concepts de permission et de rôle sont modélisés conformément au modèle RBAC, et ce, au travers d'une extension d'UML inspirée du profil SecureUML [8]. En effet, une action n'est opérationnelle pour un rôle que si ce rôle a la permission de l'exécuter. Selon le profil RBAC, un rôle est représenté par une classe stéréotypée « Role » et une permission est représentée par une classe associative stéréotypée « Permission » entre le rôle et l'entité du système sur laquelle s'applique cette permission.

Les modèles de sécurité de notre étude de cas enrichissent le modèle fonctionnel par la modélisation de quatre rôles, à savoir le conducteur (Driver), l'agent de circulation (TrafficAgent), la machine de gestion de sécurité à bord (OnboardSafetyManagement) et la machine de gestion de sécurité au sol (TracksideSafetyManagement). La Fig. 2 est un modèle de sécurité décrivant les permissions

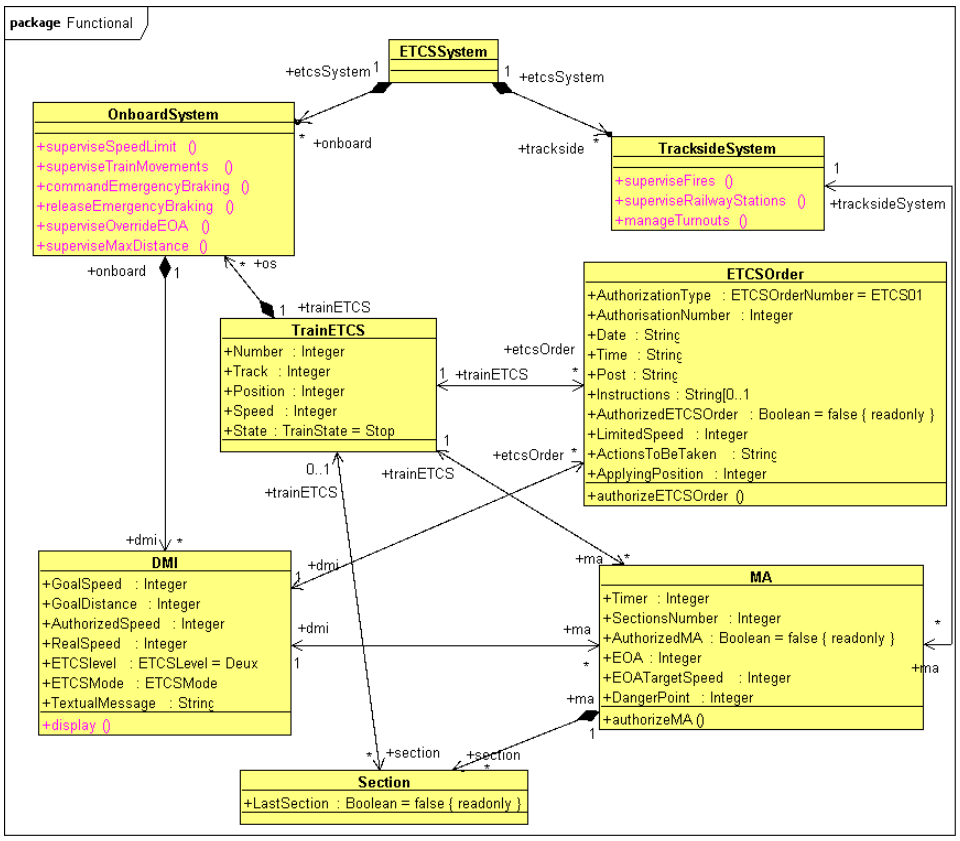

Figure 1. Le modèle fonctionnel

accordées aux rôles sur un ordre écrit. Le conducteur du train ne peut que lire un ordre écrit et suivre les indications appropriées sur le DMI alors que l'agent de circulation peut le créer, le modifier, l'autoriser et/ou le supprimer. De la même manière, nous décrivons pour chaque entité du modèle fonctionnel un modèle de sécurité contenant les permissions accordées aux rôles pouvant agir sur cette dernière.

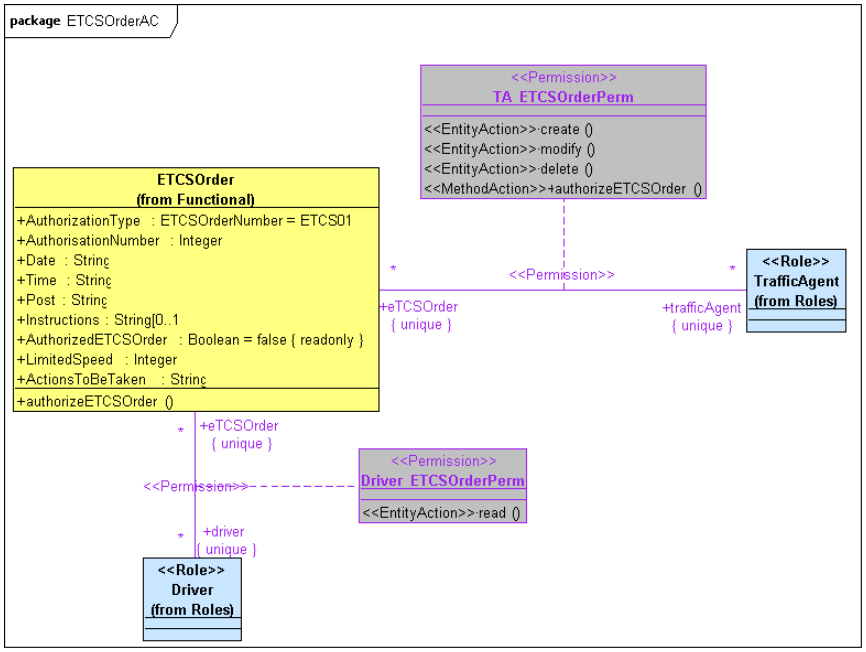

Figure 2. Permissions associées à la classe ETCSOrder

4 Transformation en spécifications B

La méthode B est une méthode de spécification formelle et d'analyse rigoureuse de la fonctionnalité et du comportement d'un système. Elle couvre toutes les phases du cycle de vie de développement logiciel allant de la spécification vers l'implémentation. Elle est basée sur des notations mathématiques fondées sur la logique du premier ordre et la théorie des ensembles où l'état du système est modélisé par des types abstraits de données prédéfinies. Elle permet la modélisation des aspects statiques et dynamiques du logiciel structurés dans des machines abstraites. L'aspect statique est caractérisé par les ensembles, les constantes, les propriétés, les variables et les invariants, tandis que l'aspect dynamique est décrit par l'initialisation et les opérations.

Dans le cadre du couplage des notations UML et B, la plate-forme dispose d'une base de règles de transformations de UML vers B. Dans cette section, nous illustrons la transformation du modèle fonctionnel ainsi que les modèles de sécurité en spécifications B.

La transformation automatique des modèles avec la plate-forme B4MSecure permet d'obtenir une unique machine B abstraite pour le modèle fonctionnel appelée « Functional » et une unique machine B pour tous les modèles de sécurité appelée « RBAC_Model ». La traduction du modèle fonctionnel est inspirée des approches de transformation existantes de UML vers B [7].

La spécification B décrit les ensembles (*SETS*), les variables abstraites (*ABSTRACT_VARIABLES*), les invariants (*INVARIANT*) et les opérations (*OPERATIONS*). La Fig. 3 donne quelques extraits du modèle fonctionnel. Les classes *MA* et *ETCSOrder* sont transformées respectivement en un ensemble *MA_AS* et un ensemble *ETCSORDER* (déclarés dans la clause SETS) qui représentent les instances possibles de ces classes et les variables abstraites *MA* et *ETCSOrder* (déclarées dans la clause *ABSTRACT_VARIABLES*) qui représentent les instances existantes de ces classes. Les invariants des lignes 15 et 16 montrent l'inclusion de *MA* dans *MA_AS* et de *ETCSOrder* dans *ETCSORDER*. Les attributs des classes (*MA__AuthorizedMA* et *ETCSOrder__AuthorizedETCSOrder*) et les associations (*MAOfTrainETCS* entre la classe *MA* et *TrainETCS* et *ETCSOrderOfTrainETCS* entre la classe *ETCSOrder* et *TrainETCS*) sont représentés sous forme de variables (lignes 9, 10, 11 et 12). Les invariants des lignes 17 et 18 sont relatifs aux attributs dans le but de spécifier leurs types (le type booléen « *BOOL* »). Les invariants des lignes 19 et 20 sont relatifs aux associations pour spécifier leurs multiplicités. Les méthodes des classes, ainsi que les constructeurs, les destructeurs, les getters et les setters sont transformés en opérations dans la partie dynamique du modèle B. Ces opérations respectent les invariants de types.

La spécification des opérations peut être enrichie par des pré-conditions et des substitutions, ainsi que la machine par des invariants. Cette tâche peut être préalablement effectuée dans le modèle UML afin de générer automatiquement des opérations bien spécifiées et des invariants de la machine autres que les invariants de types. En effet, la plate-forme B4MSecure permet d'ajouter des annotations en B sur le modèle UML pour spécifier des invariants et sur les méthodes des classes pour spécifier des pré-conditions et des substitutions d'opération. Les pré-conditions des lignes 31 et 36 et les substitutions d'opération des lignes 32 et 37 peuvent être ajoutées dans le modèle UML sous forme d'annotations respectivement sur les méthodes *authorizeMA* et *authorizeETCSOrder* des classes *MA* et *ETCSOrder*.

La machine associée au modèle de sécurité inclut via un opérateur *INCLUDES* la machine associée au modèle fonctionnel pour vérifier les propriétés de sécurité. Cette machine appelée « RBAC_Model » ajoute des variables relatives aux permissions. Par exemple, *PermissionAssignement* est une fonction totale de l'ensemble *PERMISSIONS* vers le produit cartésien *(ROLES * ENTITIES)*, *isPermitted* est une relation entre les deux ensembles *ROLES* et *Operations*. *PERMISSIONS*, *ENTITIES* et *Operations* sont des ensembles définis dans *RBAC_Model*,

```
Machine
 Functional
SETS
 MA_AS;
 ETCSORDER; ...
ABSTRACT_VARIABLES
 MA,
 ETCSOrder,
 MA__AuthorizedMA,
 ETCSOrder__AuthorizedETCSOrder,
 MAOfTrainETCS,
 ETCSOrderOfTrainETCS,
 ...
INVARIANT
 MA <: MA_AS
 & ETCSOrder <: ETCSORDER
 & MA__AuthorizedMA : MA --> BOOL
 & ETCSOrder__AuthorizedETCSOrder : ETCSOrder --> BOOL
 & MAOfTrainETCS : MA --> TrainETCS
 & ETCSOrderOfTrainETCS : ETCSOrder --> TrainETCS
 & ...
INITIALISATION
 MA := {}
 || ETCSOrder := {}
 ||  MA__AuthorizedMA := {}
 ||  ETCSOrder__AuthorizedETCSOrder := {}
 ||...
OPERATIONS
 MA__authorizeMA(Instance)=
  PRE Instance : MA
    & MA__AuthorizedMA(Instance) = FALSE
  THEN MA__AuthorizedMA(Instance) := TRUE
 END;
 ETCSOrder__authorizeETCSOrder(Instance)=
  PRE Instance : ETCSOrder
    & ETCSOrder__AuthorizedETCSOrder(Instance) = FALSE
  THEN ETCSOrder__AuthorizedETCSOrder(Instance) := TRUE
 END; ...
END
```

Figure 3. Machine « *Functional* »

alors que *ROLES* est un ensemble défini dans la machine incluse *UserAssignments*. Elle ajoute aussi des contraintes d'autorisation d'accès aux opérations du modèle fonctionnel. Elle associe une opération sécurisée correspondante à une opération fonctionnelle pour vérifier si l'utilisateur courant (le rôle) possède une permission d'effectuer cette opération. Afin de limiter l'accès à une opération fonctionnelle, un prédicat est ajouté dans l'opération sécurisée du modèle de sécurité qui vérifie si cette opération figure parmi les opérations permises par l'utilisateur courant *currentRole*. Ces prédicats apparaissent dans les lignes 23 et 29 de la Fig. 4 pour les opérations d'autorisation de *MA* et d'*ETCSOrder*.

```
1  Machine
2   RBAC_Model
3  INCLUDES
4   Functional, UserAssignments
5  SEES
6   ContextMachine
7  SETS
8   ENTITIES = {MA_Label, ETCSOrder_Label, ...};
9   Attributes = {MA_AuthorizedMA_Label,ETCSOrder_AuthorizedETCSOrder_Label...};
10  Operations ={MA_authorizeMA_Label, ETCSOrder_authorizeETCSOrder_Label...}...
11 VARIABLES
12  PermissionAssignement, isPermitted, ...
13 INVARIANT
14  PermissionAssignement: PERMISSIONS --> (ROLES * ENTITIES)
15  & isPermitted: ROLES <-> Operations ...
16 INITIAISATION
17  PermissionAssignement :=
18  {(OSM_MAPerm|->(OnboardSafetyManagement|->MA_Label)),
19   (TA_ETCSOrderPerm|->(TrafficAgent|->ETCSOrder_Label))...}
20 OPERATIONS
21  secure_MA__authorizeMA(Instance)=
22   PRE Instance: MA & MA__AuthorizedMA(Instance) = FALSE
23   THEN SELECT MA__authorizeMA_Label : isPermitted[currentRole]
24        THEN MA__authorizeMA(Instance)
25        END
26  END;
27  secure_ETCSOrder__authorizeETCSOrder(Instance)=
28   PRE Instance: ETCSOrder & ETCSOrder__AuthorizedETCSOrder(Instance) = FALSE
29   THEN SELECT ETCSOrder__authorizeETCSOrder_Label : isPermitted[currentRole]
30        THEN ETCSOrder__authorizeETCSOrder(Instance)
31        END
32  END; ...
33 END
```

Figure 4. Machine « *RBAC_ Model* »

5 Validation des modèles formels

La vérification et la validation de modèles basées sur la méthode B ont pour objectif de garantir que les modèles respectent bien les exigences de départ. Nous nous focalisons sur la validation formelle des spécifications B générées automatiquement par la plate-forme B4MSecure au moyen des outils ProB et Atelier B.

ProB supporte plusieurs techniques de validation telles que l'animation et le model checking. Nous l'avons utilisé en tant qu'animateur pour tester les deux scénarios de la section 2. Une telle animation simule l'évolution de l'état du système. Le premier état est calculé à partir de l'initialisation. Les états suivants dépendent des opérations dont la pré-condition est vérifiée.

La transformation des modèles UML en spécifications B génère des opérations B qui respectent les invariants de types (les types des attributs de classes, la multiplicité des associations entre les classes, etc.) dans le modèle fonctionnel et la politique de sécurité (les rôles, les permissions, etc.) dans le modèle de sécurité. L'animation avec ProB permet de valider, suivant les droits de chaque rôle, certaines opérations du modèle fonctionnel et de vérifier la séquence d'opérations qui construit les deux scénarios. Pour la validation de nos exigences de sécurité ferroviaire, il est judicieux d'ajouter des contraintes tant pour le modèle fonctionnel que pour le modèle de sécurité afin d'assurer la conformité des modèles B enrichis à ces exigences. L'animation de la séquence d'opérations avec ProB révèle le besoin d'ajouter des contraintes de sécurité ferroviaire à certaines opérations sous forme de pré-conditions ou bien à la spécification en sa globalité sous forme d'invariants. Ces contraintes peuvent être exprimées sous forme d'annotations en B dans les modèles UML afin qu'elles soient transformées automatiquement par la plate-forme B4MSecure dans les spécifications générées en B.

Comme mentionné dans la section précédente, le modèle de sécurité inclut le modèle fonctionnel et chaque opération sécurisée du modèle de sécurité est associée à une opération du modèle fonctionnel sur laquelle s'ajoutent des contraintes d'autorisation d'accès. De ce fait, l'opération du modèle de sécurité appelante doit contenir les mêmes pré-conditions de l'opération du modèle fonctionnel appelée.

La pré-condition ci-dessous a été ajoutée à l'opération « *DMI_ _ advance* » du modèle fonctionnel (Fig. 5) dans lequel les classes et leurs attributs sont spécifiés. De même, elle a été ajoutée à l'opération sécurisée correspondante « *secure_DMI_ _ advance* » du modèle de sécurité (Fig. 6). Cette opération sécurisée doit être exécutée par le conducteur du train à travers le DMI. Selon les règles ferroviaires décrites par notre étude de cas, le conducteur ne fait avancer le train qu'après avoir reçu l'autorisation de franchir un EOA ou bien une MA. Cette pré-condition est exprimée comme suit : Le conducteur ne peut avancer le train à travers le DMI que s'il existe un ETCSOrder (une instance déjà créée de ETCSOrder) qui a été autorisé (son attribut booléen *AuthorizedETCSOrder* est à *TRUE*) ou une MA (une instance déjà créée de MA) qui a été autorisée (son attribut booléen *AuthorizedMA* est à *TRUE*).

$$\exists\, \text{etcsorder}.(\text{etcsorder} \in \text{ETCSOrder} \wedge \text{ETCSOrder__AuthorizedETCSOrder}(\text{etcsorder}) = \text{TRUE}) \\ \vee\, \exists\, \text{ma}.(\text{ma} \in \text{MA} \wedge \text{MA__AuthorizedMA}(\text{ma}) = \text{TRUE})$$

Une autre contrainte a été ajoutée à la spécification B générée du modèle fonctionnel sous forme d'invariant qui exprime la propriété de sécurité ferroviaire suivante : sur une section de voie, au maximum un train peut circuler. Alors,

```
DMI__advance(Instance)=
   PRE
      Instance : DMI &
      #etcsorder.(etcsorder:ETCSOrder & ETCSOrder__AuthorizedETCSOrder(etcsorder)=TRUE)
or #ma.(ma:MA & MA__AuthorizedMA(ma) = TRUE)
   THEN
      TrainETCS__State(OnboardSystem_in_TrainETCS(DMI_in_OnboardSystem(Instance))) := Advance
   END
;
```

FIGURE 5. L'opération « *DMI_ _ advance* » du modèle fonctionnel

```
secure_DMI__advance(Instance)=
   PRE
      Instance : DMI &
      #etcsorder.(etcsorder:ETCSOrder & ETCSOrder__AuthorizedETCSOrder(etcsorder)=TRUE)
or #ma.(ma:MA & MA__AuthorizedMA(ma) = TRUE)
   THEN  SELECT
       DMI__advance_Label : isPermitted[currentRole]
       THEN
          DMI__advance(Instance)
       END
   END
;
```

FIGURE 6. L'opération « *secure_ DMI_ _ advance* » du modèle de sécurité

quelque soient les trains t1 et t2 appartenant aux instances existantes de la classe *TrainETCS* et au codomaine de la fonction *TrainETCSSection* et tel que le train t1 est différent du train t2 implique les images de t1 et t2 par la fonction inverse de *TrainETCSSection* (correspondantes aux sections occupées par ces deux trains) sont différentes. *TrainETCSSection* est une fonction partielle de l'ensemble *Section* vers l'ensemble *TrainETCS* qui correspond à l'association entre la classe *Section* et la classe *TrainETCS*.

$$\forall\ (t1,t2).(t1 \in \mathrm{TrainETCS} \wedge t2 \in \mathrm{TrainETCS} \wedge t1 \in \mathrm{ran}(\mathrm{TrainETCSSection}) \wedge t2 \in \mathrm{ran}(\mathrm{TrainETCSSection}) \wedge t1 \neq t2 \Rightarrow \mathrm{TrainETCSSection}^{-1}(t1) \neq \mathrm{TrainETCSSection}^{-1}(t2))$$

Cette contrainte est exprimée sous forme d'invariant qui doit être respecté par les opérations de la machine fonctionnelle, ainsi que la machine de sécurité qui l'inclut.

Nous avons utilisé l'Atelier B [6] pour prouver les spécifications B générées automatiquement et puis ces spécifications après l'ajout des contraintes. Atelier B est un outil industriel développé par la société ClearSy qui permet une utilisation opérationnelle de la méthode formelle B pour des développements de logiciels prouvés sans défaut. Il dispose d'un prouveur automatique pour la démonstration de la plupart des obligations de preuves justes et d'un prouveur interactif pour la détection des erreurs et la finalisation de la preuve. Nous avons alors utilisé le prouveur automatique et un ensemble de commandes du prouveur interactif pour prouver les obligations de preuves restantes dues aux contraintes ajoutées. Les obligations de preuves générées du modèle fonctionnel sont toutes prouvées. Par contre, des obligations de preuves du modèle de sécurité ne sont

6. Atelier B : http ://www.atelierb.eu/

pas encore prouvées à cause d'une limitation du prouveur qui ne parvient pas à raisonner sur toutes les définitions décrites dans le modèle de sécurité. Le modèle de sécurité est considéré comme un filtre qui ne propose pas de nouvelles opérations et n'ajoute pas un comportement supplémentaire. Il permet donc de limiter l'accès à certaines opérations en fonction du rôle. Les principaux invariants à respecter sont ceux qui sont liés aux concepts de RBAC, ce qui est déjà réalisé par la transformation automatique à l'aide de l'outil B4MSecure.

6 Conclusion

Notre étude de cas est basée sur deux scénarios extraits des règles d'exploitation ferroviaires ERTMS/ETCS appliquées sur la ligne à grande vitesse LGV Est-Européenne. Ces scénarios contiennent des aspects fonctionnels liés à la fonctionnalité du système (modèle fonctionnel) et des aspects de sécurité liés au contrôle d'accès (modèles de sécurité). Notre objectif est de pouvoir les modéliser et les valider formellement. Basée sur l'approche IDM, la plate-forme B4MSecure nous a permis, d'une part, de modéliser ces règles en diagrammes de classe UML renforcés par un profil d'extension UML pour la politique de contrôle d'accès, et d'autre part, de les transformer en spécifications B afin de les valider formellement au moyen de l'animateur ProB et du prouveur Atelier B. Une amélioration sur l'architecture des spécifications générées en B pourrait être intégrée à la plate-forme B4MSecure. Elle s'agit de remplacer l'inclusion du modèle fonctionnel par le modèle de sécurité par un raffinement du modèle fonctionnel. Le modèle de sécurité réduit l'espace d'état du modèle fonctionnel grâce à des règles de contrôle d'accès. De ce fait, considérer le modèle de sécurité comme un raffinement du modèle fonctionnel permet de respecter les invariants du modèle fonctionnel et de préserver les propriétés sans augmenter la taille des invariants à prouver ainsi que celle des obligations de preuves.

UML permet la modélisation avec des différentes vues graphiques des aspects différents du système grâce à l'utilisation des profils. Dans notre étude de cas, les scénarios décrits mettent en œuvre des actions et des interactions entre les rôles et les entités du système. Nous visons dans des travaux futurs à explorer les diagrammes UML dynamiques tels que les diagrammes de séquence de telle sorte que nous pourrions valider formellement la séquence des opérations après sa transformation en spécifications B. [3] propose une transformation des diagrammes UML de séquence en des spécifications formelles CSP afin de valider les exigences décrites par ces diagrammes. Les travaux réalisés dans [4] proposent aussi une approche pour valider des spécifications UML avec des diagrammes de classe et des diagrammes de séquence en les transformant en spécifications B définissant une nouvelle machine B de simulation.

L'exploitation de l'approche utilisant B4Msecure pour une nouvelle application s'intéressant à l'analyse de la sécurité ferroviaire pour une portion de règles d'exploitation est plutôt concluante. En ce qui concerne les aspects dynamiques du modèle, l'approche existante avoue quelques limites et des extensions sont à envisager.

Remerciements Ces travaux de recherche ont été soutenus par le projet Perfect (ANR-12-VPTT-0010) et en partie par le projet Selkis (ANR-08-SEGI-018) et l'ARC6 de la Région Rhône-Alpes.

Références

1. ALCATEL, ALSTOM, ANSALDO SIGNAL, BOMBARDIER, INVENSYS RAIL, SIEMENS : ETCS-Baseline 2, System Requirements Specification - Subset026. 7 chapters (2006)
2. RFF : Principes et règles d'exploitation du système ETCS - Particularités en cas de superposition à un autre système de signalisation. PROJET 0T Révision (2012) (to be published)
3. Rasch, H., Wehrheim, H. : Checking the Validity of Scenarios in UML Models. Proceedings of the 7th IFIP WG 6.1 International Conference on Formal Methods for Open Object-Based Distributed Systems, pp. 67-82. Springer, Heidelberg (2005)
4. Truong, N.T. and Souquières, J. : Test of object-based specifications using B notations. Hal-INRIA, hal-00015031. LORIA (2005)
5. Ledru, Y., Idani, A., Milhau, J., Qamar, N., Laleau, R., Richier, J.L., Labiadh, M.A. : Taking into Account Functional Models in the Validation of IS Security Policies. Advanced Information Systems Engineering Workshops. LNCS, vol. 83, pp. 592-606. Springer, Heidelberg (2011)
6. Milhau, J., Idani, A., Laleau, R., Labiadh, M.A., Ledru, Y., Frappier, M. : Combining UML, ASTD and B for the formal specification of an access control filter. Innovations in Systems and Software Engineering, vol. 7, pp. 303-313. Springer (2011)
7. Idani, A., Labiadh, M.A., Ledru, Y. : Infrastructure dirigée par les modèles pour une intégration adaptable et évolutive de UML et B. Ingénierie des Systèmes d'Information Journal, vol. 15, pp. 87-112 (2010)
8. Lodderstedt, T., Basin, D.A., Doser, J. : SecureUML : A UML-Based Modeling Language for Model-Driven Security. Jézéquel, J.-M., Hussmann, H., Cook, S. (eds.) UML 2002. LNCS, vol. 2460, pp. 426-441. Springer, Heidelberg (2002)
9. Idani, A., Ledru, Y., Labiadh, M.A. : B4MSecure : une plateforme IDM pour la modélisation et la validation de politiques de sécurité en Systèmes d'Information. Journées Francophones sur les Approches Formelles dans l'Assistance au Développement de Logiciels (AFADL), pp. 85-89 (2013)
10. Sandhu, R., Ferraiolo, D., Kuhn, R. : The NIST Model for Role-based Access Control : Towards a Unified Standard. Proceedings of the Fifth ACM Workshop on Role-based Access Control, pp. 47-63 (2000)
11. Ben Ayed, R., Collart-Dutilleul, S., Bon, P., Idani, A., Ledru, Y. : B Formal Validation of ERTMS/ETCS Railway Operating Rules. Y. Ait Ameur and K.-D. Schewe (Eds.) : ABZ 2014. LNCS, vol. 8477, pp. 124-129. Springer, Heidelberg (2014)