ORQA : un canevas logiciel pour la gestion de l'énergie dans les véhicules électriques

Borjan Christophe-Tchakaloff

THÈSE / UNIVERSITÉ DE BRETAGNE OCCIDENTALE
sous le sceau de l'Université européenne de Bretagne

pour obtenir le titre de
DOCTEUR DE L'UNIVERSITÉ DE BRETAGNE OCCIDENTALE
Mention « Sciences et Technologies de l'Information et de la Communication »
École Doctorale Santé, Information-Communications, Mathématiques, Matière (SICMA)

présentée par
Borjan Christophe-Tchakaloff
Préparée au laboratoire CNRS Lab-STICC et au sein du pôle Systèmes et Énergies Embarqués pour les Transports d'ESTACA'Lab

ORQA : un canevas logiciel pour la gestion de l'énergie dans les véhicules électriques

Thèse à soutenir le 13 janvier 2015
devant le jury composé de :

Sébastien Gérard
Docteur-HDR, CEA Saclay / *rapporteur*

Lionel Seinturier
Professeur, Université de Lille 1/ *rapporteur*

Antoine Beugnard
Professeur, ENST Bretagne / *examinateur*

Jean-Marc Jézéquel
Professeur, IRISA / *examinateur*

Mickaël Kerboeuf
Maître de conférence, Université de Bretagne Occidentale / *examinateur*

Jean-Philippe Babau
Professeur, Université de Bretagne Occidentale / *directeur de thèse*

Sébastien Saudrais
Enseignant-Chercheur, ESTACA / *encadrant*

So Long, and Thanks for All the Fish.

Remerciements

Je tiens tout d'abord à remercier mes encadrants pour m'avoir accompagné tout au long de ces trois années. En premier lieu, Sébastien Saudrais pour m'avoir supporté, encouragé mais aussi subi au quotidien. Ses remarques et ses encouragements ont été primordiaux durant le déroulement de cette thèse. Aussi, un grand merci à mon directeur de thèse Jean-Philippe Babau d'avoir continuer à me conseiller et ré-orienter quand cela était nécessaire.

Je tiens ensuite à remercier les membres du jury qui me font l'honneur de participer à la soutenance de cette thèse. Mes remerciements les plus sincères vont à Sébastien Gérard et Lionel Seinturier pour avoir accepter de rapporter ma thèse.

Il est également important que je remercie mes collègues de l'ESTACA pour leur accueil chaleureux et leur support durant mon passage parmi eux. La communauté des doctorants m'a plus particulièrement encouragé et porté, merci. Je remercie bien entendu l'équipe Systèmes Embarqués, maintenant intégrée au pôle S2ET, pour m'avoir fourni un environnement de travail propice.

Enfin, malgré peu de visites parmi eux, les membres de l'équipe du LISyC de l'UBO, désormais intégrés au département informatique du Lab-STICC, m'ont toujours bien accueilli et fourni de nombreux conseils.

Je ne peux évidemment pas oublier de remercier ma famille et mes amis pour leur soutien incommensurable et inconditionnel.

Résumé

Les véhicule électriques présentent désormais une alternative crédible aux véhicules équipés de moteurs à combustion interne. Ils sont plus propres et plus confortables à l'utilisation. La gestion de l'énergie d'un véhicule électrique est actuellement focalisée sur le fonctionnement du moteur, principal consommateur d'énergie, sans tenir compte du confort de l'utilisateur.

Le travail présenté dans cette thèse intègre la prise en compte des préférences utilisateur au sein de la gestion de l'énergie des véhicules électriques. La contribution est réalisée par un canevas logiciel nommé ORQA. Dans un premier temps, les organes du véhicule sont caractérisés par leurs besoins énergétiques et par les niveaux de qualité de service qu'ils proposent. Un gestionnaire d'énergie est ensuite intégré au système logiciel du véhicule. Il se base sur les caractéristiques des organes et propose à l'utilisateur une solution de configuration de trajet prenant en compte ses préférences d'utilisation ainsi que d'éventuelles contraintes de temps et d'énergie. Cette solution définit des contraintes d'utilisation sur le moteur et les organes de confort lors du déroulement du trajet.

Le gestionnaire d'énergie est exécuté au sein des systèmes embarqués du véhicule dont les plates-formes sont fortement contraintes. Pour satisfaire celles-ci, il est primordial de proposer une configuration efficace du gestionnaire d'énergie. L'intérêt et la validité de l'approche employée sont démontrés au travers de la comparaison entre la configuration originelle et une configuration optimisée sur deux véhicules exemples.

Mots-clefs

Gestion d'énergie, qualité de service, véhicule électrique, ingénierie dirigée par les modèles, systèmes embarqués.

Abstract

Electric vehicles are nowadays a viable alternative to vehicles built around an internal combustion engine. They offer a cleaner and more comfortable driving thanks to the electric engine. The energy management of an electric vehicle is usually focused on the engine operation, the biggest energy consumer, thus ignoring the user comfort.

The contribution presented in this thesis allows for the consideration of the user preferences inside the energy management of electric vehicles. It takes shape with a framework named ORQA. First, the vehicle devices are characterised by their energy requirements and the quality levels they offer. An energy manager based on the devices characteristics is then integrated into the software system of the vehicle. The manager offers the user a solution to configure a trip request. The configuration is based on the usage preferences and optional constraints over duration and consumption. The solution defines a set of constraints on the motor and on the comfort-related devices during the trip execution.

The energy manager is executed in the embedded systems of the vehicle which platforms are highly constrained. Thus, the energy manager's implementation is optimised to satisfy the execution platform constraints. The interest and the validity of the chosen approach are attested by the comparison of the original configuration and an optimised configuration on two example vehicles.

Keywords

Energy management, quality of service, electric vehicle, model based engineering, embedded systems.

Table des matières

Introduction

Contexte

Les véhicules électriques domestiques font désormais partie du paysage automobile mondial. Même s'ils existent depuis plus de cent ans, ce n'est que récemment que les technologies de stockage d'énergie ont suffisamment évoluées pour permettre un usage quotidien de ce type de véhicule. Les moteurs électriques sont plus efficaces que leurs homologues à explosion, ainsi la propulsion d'une voiture électrique avoisine les 90% d'efficacité contre 20% pour une voiture à essence [LL12]. Les véhicules électriques apportent également un plus grand confort, le véhicule n'est plus propulsé par des explosions (engendrant ainsi vibrations et bruit) mais circule grâce aux champs électromagnétiques. Dans le cadre de cette thèse, nous nous concentrons sur les voitures tout électriques accessibles aux particuliers.

De manière simplifiée, un véhicule électrique possède une source d'énergie et des consommateurs d'énergie. La source est le système de stockage d'énergie, fréquemment remplacée par le terme *batterie*. Même si d'autres sources existent (comme les super-capacités et les piles à combustible), la batterie est actuellement la technologie la plus répandue. L'autonomie des véhicules électriques est souvent critiquée comme étant insuffisante. La crainte alors formulée est l'angoisse du faible rayon d'action (*range anxiety*), la peur de tomber en panne. Les voitures électriques urbaines indiquent une autonomie d'environ 150km à l'usage alors que les véhicules conventionnels de gamme équivalente offrent une autonomie de plus d'un millier de kilomètres. Le rapport entre la capacité énergétique d'un véhicule électrique avec celle d'un véhicule équipé d'un moteur à combustion est donc de l'ordre de 1 pour 10.

Au sein d'un véhicule électrique, le consommateur le plus important est le moteur électrique comme l'illustre la figure 1. Les autres principaux consommateurs équipant communément les véhicules électriques sont le système de climatisation, le système d'éclairage, le système de divertissement ainsi que les systèmes de sécurité et le système électronique (regroupés sous *autres* dans la figure). Ainsi, et de manière logique au vu de la répartition des consommateurs, la gestion énergétique d'un véhicule électrique est focalisée sur la motorisation. Dans cette thèse, nous

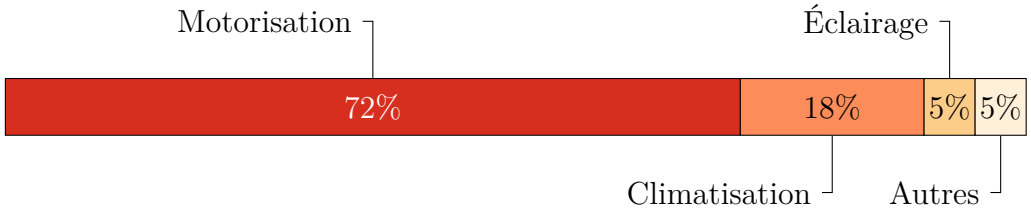


FIGURE 1 – Répartition moyenne des consommations d'un véhicule électrique en zone urbaine. Exemple basé sur les données présentées au chapitre 4, page 87.

nous intéressons aussi aux autres systèmes équipant un véhicule, appelés par la suite équipements ou organes.

Les algorithmes de contrôle des équipements, d'optimisation de leur utilisation et ceux d'aide à la conduite sont réalisés par des systèmes embarqués. Ces derniers regroupent un ensemble de plates-formes d'exécution, ou calculateurs. Afin de permettre une vue systémique et de promouvoir la réutilisation des algorithmes et implémentations, les constructeurs et fournisseurs majeurs du secteur automobile ont créé le consortium AUTOSAR [AUT]. Celui-ci a établi une méthodologie standard éponyme de modélisation des systèmes embarqués. Elle est basée sur le paradigme des composants logiciels, promouvant ainsi une encapsulation des fonctionnalités du véhicule.

Du freinage à la direction électrique en passant par la boite de vitesses robotisée, le système logiciel contrôle de plus en plus de fonctionnalités jusqu'alors purement mécaniques. Elles apparaissent ainsi dans la modélisation des systèmes embarqués qui sont désormais au cœur des véhicules modernes. De plus, ces systèmes embarqués évoluent vers des systèmes communiquant avec leur environnement. Nous sommes habitués aux systèmes de localisation (*Global Positioning System*, GPS), d'info-trafic (*Traffic Message Channel*, TMC) ou encore d'aide au parking mais les technologies de communication entre véhicules (*vehicle-to-vehicle*, V2V) ou véhicule et infrastructure (*infrastructure-to-vehicle*, I2V ; *vehicle-to-infrastructure*, V2I) sont à l'étude depuis plusieurs années [PdE+09]. Celles-ci ont pour objectifs majeurs d'améliorer la sécurité des usagers de la route et d'augmenter l'efficacité des transports à travers un partage d'informations inaccessibles (hors du champ de visibilité des capteurs) et d'approches coopératives. Les applications mises en avant sont par exemple la fluidification du trafic (bouchons et accidents) et la prévention d'accidents (détection de comportements dangereux par exemple).

Problématiques

La gestion énergétique des véhicules électriques actuels, de plus en plus complexes, est focalisée sur l'utilisation de la motorisation. Une gestion basique et répandue est la suivante : à partir du moment où le niveau d'énergie embarquée dans le véhicule atteint un seuil critique, la vitesse maximale du véhicule est bridée, son accélération limitée et ses équipements désactivés. Une telle gestion ne tient pas compte des attentes de l'utilisateur. Si le conducteur attend de rejoindre une destination grâce à son véhicule, il doit pouvoir être assurer d'y arriver mais également choisir *comment* il y arrive. C'est à dire que les fonctionnalités du véhicule, particulièrement si elles possèdent une dimension qualitative, doivent être gérées finement avec un point de vue global.

Les fonctionnalités logicielles – ou contrôlées logiciellement – sont réalisées par les systèmes embarqués, modélisés avec AUTOSAR. Ces modèles permettent de représenter les opérations réalisables par le véhicule, leurs causalités et leurs dépendances. En revanche, les propriétés extra-fonctionnelles – telles que les besoins énergétiques et les qualités de service – ne sont pas représentables dans les modèles AUTOSAR. De plus, comme le système logiciel est responsable d'un nombre croissant de fonctionnalités, il est nécessaire d'assurer sa sûreté de fonctionnement ainsi que son comportement. Dans AUTOSAR, l'architecture logicielle ne peut pas être adaptée à l'exécution, il est nécessaire de prévoir et d'organiser les différents modes de fonctionnement à la conception.

Dans ce contexte, l'objectif de cette thèse est d'optimiser *via* le logiciel la gestion de la ressource énergétique – du point de vue utilisateur – au sein des véhicules électriques en considérant l'ensemble des équipements et les préférences utilisateur.

Contributions

Pour répondre à ce défi, nous proposons dans un premier temps d'acquérir les connaissances de besoins énergétiques et de niveaux de qualité en caractérisant tous les organes du véhicule. Puis, à l'aide d'un gestionnaire d'énergie basé sur ces connaissances, le conducteur est assuré d'arriver à destination avec le niveau de qualité qu'il désire. Les contributions de cette thèse sont donc :

La caractérisation des organes du véhicule et des préférences utilisateur. Un métamodèle propre à la caractérisation énergétique et qualitative est défini. Chaque organe est caractérisé par ses modes de fonctionnement, ses besoins énergétiques ainsi que par ses niveaux de qualité. Les connaissances modélisées sont intégrées aux modèles AUTOSAR existants par le biais d'un gestionnaire d'éner-

gie lui-même intégré à l'architecture des systèmes embarqués. Comme il n'est pas possible de représenter les propriétés extra-fonctionnelles au niveau du modèle AUTOSAR, ces connaissances sont intégrées lors de l'implémentation du gestionnaire d'énergie.

Le gestionnaire d'énergie optimise en ligne l'utilisation des organes du véhicule selon les attentes du conducteur pour un trajet. Le gestionnaire d'énergie répond à une requête du conducteur pour atteindre une destination. Cette requête contient les préférences d'utilisation du conducteur pour son véhicule ainsi que d'éventuelles contraintes sur la durée du trajet et sur le niveau d'énergie à conserver. Le gestionnaire d'énergie détermine la meilleure stratégie de conduite combinant une allure de conduite et une utilisation fixe des organes selon la politique de consommation demandée par le conducteur. L'utilisation des organes telle que définie dans la stratégie permet d'établir la consommation globale du véhicule mais aussi le niveau de qualité qu'il assure pour ce trajet. Comme il n'est pas possible de reconfigurer dynamiquement les fonctionnalités logicielles pour satisfaire la stratégie choisie, le gestionnaire d'énergie prend le contrôle de certains organes du véhicule par le biais de courtiers ajoutés dans l'architecture lors de l'intégration du gestionnaire d'énergie. La stratégie est réalisée et sa qualité de service ainsi assurée.

Le canevas logiciel ORQA est défini pour assurer la réalisation de ces deux parties, l'une à la conception et l'autre à l'exécution. Il s'intègre à la phase de conception des systèmes embarqués, une fois que sont modélisées les fonctionnalités du système. La gestion de la qualité de service est ensuite ajoutée à l'architecture des systèmes embarqués pour être finalement utilisée en ligne par le conducteur. Celle-ci permet au conducteur de décider sous quelle politique doit être réalisé son trajet tout en améliorant la dépense énergétique globale du véhicule.

Plan

Cette thèse comprend quatre chapitres. Le premier chapitre présente les travaux liés aux problématiques étudiées à travers un état de l'art.

Le second chapitre présente la contribution, le canevas logiciel ORQA, et détaille ses objectifs. La modélisation des organes consommateurs du véhicule y est abordée. Nous y détaillons leur comportement, leurs besoins énergétiques ainsi que les qualités de service offertes. Puis, nous abordons le fonctionnement du gestionnaire d'énergie par la recherche (et sélection) d'une solution à une requête du conducteur. Comme ce gestionnaire cible les systèmes embarqués, nous tenons compte des ressources limitées en proposant une optimisation de sa configuration.

Nous présentons enfin l'architecture réalisant le gestionnaire d'énergie par le biais de composants logiciels.

Le troisième chapitre décrit la mise en œuvre du canevas logiciel. Nous détaillons tout d'abord le métamodèle final et le langage textuel associé, proposé pour faciliter la manipulation des modèles par les concepteurs. Nous détaillons également les modèles génériques des principaux organes consommateurs inclus dans le canevas logiciel sous forme d'une bibliothèque. Le prototype AUTOSAR permet ensuite d'illustrer la mise en place du gestionnaire d'énergie et la génération de son implémentation. Nous abordons finalement les algorithmes permettant d'optimiser la configuration du gestionnaire d'énergie pour réduire son coût de fonctionnement.

Le quatrième chapitre valide l'utilisation du canevas logiciel ORQA avec deux véhicules. Deux scénarios sont étudiés et permettent de visualiser les différences induites par la configuration du gestionnaire d'énergie.

Enfin, une conclusion générale du travail présenté est dressée ainsi que des perspectives d'évolution par rapport aux contributions apportées.

Chapitre 1

État de l'art

Sommaire

Ce chapitre présente les solutions existantes dans la littérature liées à nos problématiques, c'est à dire la gestion de l'énergie, la qualité de service, et les architectures logicielles. Ces problématiques sont des domaines à part entière et nombre de travaux existent à ce sujet dans la littérature. Malgré le fait qu'elles puissent être étudiées de manière autonome, elles ne prennent tout leur sens que lorsqu'elles sont combinées.

Nous décrivons premièrement notre contexte de modélisation. Nous présentons ensuite des travaux liés à chacune des problématiques.

1.1 Modélisation et composants logiciels

Les systèmes et applications informatiques deviennent de plus en plus complexes afin d'offrir plus de services, une meilleure robustesse mais aussi d'améliorer le traitement des données dont le volume ne cesse de croître. Une formalisation des processus de développement est mise en place depuis les années 1990 : l'ingénierie dirigée par les modèles (IDM ou *Model Driven Engineering*, MDE).

1.1.1 L'ingénierie dirigée par les modèles

Cette approche permet, à partir de modèles de haut-niveau, de générer des modèles raffinés plus proches de l'implémentation. Ce sont par exemple le code et la documentation. L'idée est de capturer et d'automatiser les bonnes pratiques souvent implicites et propres à un domaine. Au lieu de passer d'un niveau d'abstraction à un autre en reprenant les concepts précédemment décrits et de les raffiner « à la main », le processus est automatisé. Le cœur de cette approche repose sur les transformations de modèle qui permettent de passer d'une représentation à une autre. À un même niveau d'abstraction, il est par exemple possible d'exprimer des besoins différents (sécurité, fiabilité, efficacité, etc.) de différentes façons (un besoin = un modèle) tout en partageant une base commune. À différents niveaux, les transformations servent à généraliser (abstraire) ou à raffiner (implémenter) les concepts.

Au début des années 2000, l'OMG (*Object Management Group*) [OMG] a poursuivi sa quête de standardisation à base de consensus débutée dans les années 1990 avec CORBA (*Common Object Request Broker Architecture*) [OMG91] et la première version d'UML (*Unified Modeling Language*) [OMG97]. L'organisme a introduit l'ingénierie dirigée par l'architecture (*model driven architecture*, MDA) [Sol00] comme un moyen de contrôler le nombre florissant d'intergiciels et de langages de programmation. MDA est une spécialisation de l'IDM au niveau architectural où sont principalement mis en avant les aspects de déploiements spécifiques aux plates-formes d'implémentation.

1.1.2 Les composants logiciels

Parallèlement à l'avènement des processus à base de modèles, la notion de composant logiciel a été formalisée [NGT92, HC01, Szy02] pour pallier à certains manques des objets. Bien qu'ébauché précédemment [MBNR68], le paradigme des composants logiciels n'est réellement utilisé à grande échelle que depuis une dizaine d'années avec les modèles à composants (p. ex. Fractal [OW202] et COM [WK94]) et canevas logiciels à composants (p. ex. Think [FSLM02] et Julia [BCL+06] pour Fractal, *Enterprise Java Beans* [MH98] pour Java et CCM [OMG02] basé sur Corba).

Ce paradigme permet de définir explicitement les interfaces fournies et les interfaces requises par chacune des briques applicatives d'un système. Par la promotion de l'indépendance de ces briques logicielles, il met en avant une réutilisation plus sûre qu'avec les objets qui sont souvent interconnectés de manière implicite. La définition des composants énoncée par Clemens Szyperski dans [Szy02] tient lieu de référence :

> "A software component is a unit of composition with contractually specified interfaces and explicit context dependencies only. A software component can be deployed independently and is subject to composition by third parties."

Ainsi, les propriétés suivantes d'un composant sont distinguées :

Composition La notion de composition de composants est primaire. Un composant peut être défini à partir d'autre composants, il est alors appelé composite (ou primitif dans le cas contraire). Du point de vue extérieur, c'est un composant au même titre qu'un composant primitif. Le système global étant une composition de composants (primitifs ou composites), il est lui même un composant. Cette propriété met en exergue l'assemblage et la réutilisation des composants comme briques.

Définition des interfaces Un composant est accompagné de la définition de ses interfaces, c'est à dire de ses points d'entrée (les services qu'il offre) et de ses dépendances (les services qu'il requiert). La définition des interfaces permet d'assurer un assemblage correct par vérification de type.

Déploiement Un composant est une brique logicielle indépendante. Il peut être utilisé par n'importe quel système du moment que ses interfaces sont respectées. Cette propriété est particulièrement utile pour les systèmes ouverts où la configuration d'exécution est évolutive. De nouveaux composants peuvent ainsi être ajoutés ou d'anciens mis à jour pour supporter l'évolution du système.

Les interfaces des composants logiciels peuvent être étendues pour considérer des propriétés (extra-)fonctionnelles comme la qualité de service [DJP04]. Ces

propriétés peuvent être assurées par le biais de contrats auxquels le fournisseur (le composant implémentant le service) se soumet. La définition acceptée des contrats pour les composants est celle présentée par Beugnard et coll. dans [BJPW99]. Les auteurs y distinguent quatre niveaux de contrats à la négociabilité croissante :

Niveau syntaxique Ce niveau correspond aux interfaces du composant, les services qu'il offre, ceux qu'il requiert ainsi que ses potentielles exceptions. La vérification de ce niveau peut être réalisée statiquement *via* l'architecture des composants et les connexions qui les relient. Il est également possible de vérifier de manière dynamique la conformité d'un composant en surveillant les types des données échangées.

Niveau comportemental Ce sont ici les pré- et post-conditions ainsi que les invariants qui sont définis. Ce sont des assertions booléennes permettant d'assurer un comportement (post-conditions) après l'exécution d'un service si les conditions d'entrées sont respectées (pré-conditions). Les invariants assurent l'intégrité d'une instance de composant. Les services d'un composant sont considérés de manière atomique. Ce niveau a précédemment été décrit comme *design by contract* dans Eiffel [ISO06, Mey97]. Il peut être mis en œuvre en UML avec OCL (*Object Constraint Language*) [ISO12, WK98].

Niveau de synchronisation Ce niveau spécifie le comportement global d'un composant en terme de synchronisation entre les appels de services. Les dépendances entre les services fournis par le composant sont exprimées en tant que séquence, parallélisme ou choix arbitraire. La manière dont le composant sert ses clients est ainsi définie explicitement.

Niveau qualité de service C'est la quantification du comportement attendu du composant qui est explicitée par le biais de propriétés (fonctionnelles ou non). Par exemple, dans le cadre des systèmes embarqués, les propriétés communément considérées sont temporelles (temps de réponse moyen/maximal et gîte). Elles peuvent également être qualitatives (niveau arbitraire, qualité du résultat). Ce niveau est le plus négociable mais aussi le plus difficile à vérifier. Ceci est dû au fait que ces propriétés sont généralement obtenues auprès de tierces parties (mesures, spécifications) n'offrant pas de garantie de performance.

Les logiciels embarqués dans le domaine automobile n'échappent pas à la croissance de complexité. Pour faciliter la définition et l'usage des systèmes embarqués, l'IDM et les composants sont désormais utilisés par les constructeurs et les fournisseurs automobiles.

1.1.3 AUTOSAR

L'arrivée du logiciel dans le monde automobile a permis d'améliorer les performances et la sécurité des véhicules. Les moteurs sont désormais contrôlés et régulés de manière logicielle, permettant d'améliorer leur rendement et de diminuer leurs rejets polluants. De nombreuses fonctions de sécurité ont pu être intégrées. La sécurité passive est la plus répandue avec par exemple l'anti-blocage des roues (*Antiblockiersystem*, ABS), l'anti-patinage et l'aide au freinage d'urgence (*Emergency Brake Assist*, EBA). La sécurité active, qui était l'apanage des véhicules haut de gamme, tend aussi à se généraliser avec par exemple la correction électronique de trajectoire (*Electronic Stability Program*, ESP) et la régulation de vitesse adaptative (*Adaptive Cruise Control*, ACC).

Il est estimé que les véhicules actuels embarquent environ cent millions de lignes de code [MÏ10]. Les fonctionnalités embarquées sont réparties sur plusieurs dizaines (voire une centaine pour le haut de gamme) de contrôleurs (*Electronic Control Unit*, ECU). Comme les contrôleurs sont des petites unités de calcul dédiées locales, une fonctionnalité peut être répartie sur plusieurs unités. Ces dernières utilisent des réseaux spécifiques pour communiquer (CAN [ISO93], FlexRay [ISO10], LIN [ISO14], MOST [ABD+98]). Chacun des réseaux répond partiellement aux contraintes du domaine automobile (temps de réponse, performance, robustesse, bande passante, etc.), leur usage étant propre à un domaine : CAN et FlexRay pour les applications critiques, le LIN pour les éléments de confort et le MOST pour le multimédia.

La durée de vie d'un véhicule est parfois supérieure à celle des plates-formes matérielles des contrôleurs embarqués, il faut alors assurer le fonctionnement des logiciels sur de nouvelles plates-formes. Les fonctionnalités sont également partagées entre plusieurs véhicules, il ne doit pas être nécessaire de recommencer le développement d'une fonctionnalité déjà existante. Ainsi, la configuration des fonctionnalités logicielles est désormais évolutive.

Dans ce contexte, le consortium AUTOSAR (*AUTomotive Open System ARchitecture*) [AUT] a été fondé en 2003 par les principaux constructeurs automobiles et équipementiers. Son rôle est d'établir un standard industriel global et ouvert pour l'architecture des logiciels automobiles entre les fournisseurs et les constructeurs [AUT03]. Il cherche à prendre en main la complexité croissante des fonctionnalités logiciels embarquées. À cet effet, il propose une méthodologie de conception ainsi qu'une modélisation des fonctionnalités selon l'ingénierie dirigée par les modèles et le paradigme des composants logiciels. Un métamodèle spécifique est proposé pour représenter les fonctionnalités réalisées logiciellement dans un véhicule. La philosophie sous-jacente à AUTOSAR est une standardisation de l'architecture et des interfaces avec une implémentation propre à chaque fournisseur. Un intergiciel est également défini, le RTE (*Runtime Environment*), pour abstraire les fonction-

nalités offertes par une plate-forme. Les fonctionnalités reposent sur les services du RTE et non pas sur des interfaces spécifiques aux contrôleurs. Ce RTE repose lui-même sur les BSW (*Basic Software*s), ce sont en quelque sorte les pilotes des ressources d'une plate-forme. On y retrouve par exemple les pilotes de la mémoire mais aussi la configuration du système d'exploitation.

La méthodologie de conception d'AUTOSAR se base sur les spécifications des fonctionnalités. Elle peut être définie en trois grandes étapes :

Description Cette première étape contient une description à trois niveaux. Elle requiert que les fonctionnalités à embarquer dans le véhicule soient identifiées. Les fonctionnalités sont réalisées par l'interconnexion de composants logiciels. Ceux-ci sont définis en précisant les ressources matérielles requises (contrôleurs, capteurs et actionneurs). Chaque composant s'exécute sur un contrôleur. La topologie du système global du véhicule (l'ensemble des contrôleurs interconnectés) est spécifiée ainsi que ses attributs tels que les types de réseaux disponibles et les protocoles utilisés. Enfin, les ressources (processeur, mémoire, capteurs, actionneurs) des plates-formes disponibles sont définies au travers de leurs caractéristiques.

Configuration La configuration du système est réalisée en premier lieu. Les composants logiciels sont distribués sur les différents contrôleurs. Cette étape doit prendre en considération les ressources disponibles sur un contrôleur et celles requises par un composant. Il faut, de plus, tenir compte de contraintes globales des fonctionnalités comme par exemple des temps de réponse maximaux. Ainsi, en plus du temps de calcul variable d'un contrôleur à un autre, le temps de propagation entre les composants est primordial pour une application temps-réel. Il peut par exemple être nécessaire de regrouper des composants sur un même contrôleur car les réseaux disponibles n'offrent pas (ou ne sont pas en mesure d'offrir) une garantie temporelle suffisante. Les couches basses (RTE et BSW) des contrôleurs sont ensuite configurées une par une. Leur configuration dépend de l'allocation effective des composants. Par exemple, dans le cas d'un système d'exploitation multitâche, l'ordonnanceur est paramétré et les tâches configurées en fonction des besoins effectifs.

Réalisation L'étape de réalisation à proprement parler contient la génération des artefacts et l'implémentation des composants logiciels. À partir du modèle architectural AUTOSAR et des configurations des contrôleurs, le RTE et la configuration des BSW sont générés. Comme le modèle AUTOSAR n'est pas comportemental, seulement les squelettes des composants logiciels peuvent être générés. C'est ensuite aux concepteurs de réaliser (ou réutiliser) l'implémentation des composants logiciels.

1.2 La gestion de l'énergie

L'énergie est un enjeu majeur de notre époque. Avec d'un côté la nécessité d'économiser l'énergie et de l'autre côté l'accroissement d'éléments consommateurs, on retrouve dans tous les domaines de nombreuses normes et standards visant à réguler les dépenses énergétiques (l'étiquette-énergie des appareils électroménagers [Par10], les normes de constructions des bâtiments [Con10]) ou la pollution (norme Euro 6 sur les rejets de polluants par les véhicules [Com12]). Comme les systèmes informatisés ne cessent de se répandre et de se complexifier, il est primordial de gérer leur consommation énergétique au mieux. L'automobile et en particulier le véhicule électrique n'y font pas exception.

Tout d'abord, il est possible de gérer la consommation d'un système informatique du point de vue matériel et du point de vue applicatif. Au niveau matériel, les ressources d'un ordinateur peuvent offrir plusieurs niveaux de fonctionnement entre la performance et l'économie de consommation [LS98]. Le standard ACPI [IMT96] offre ainsi quatre niveaux de veille, du plus rapide à atteindre au plus long, étant entendu que plus la veille est profonde et plus l'économie d'énergie est importante. Les éléments consommateurs prépondérants dans les systèmes personnels, les systèmes embarqués et les systèmes de grande échelle sont le processeur et la mémoire. Ainsi, de nombreuses approches d'optimisation d'efficacité ont été proposées à base de changement de fréquence [HV14, LQW08] ou encore d'ordonnancement [AA05, TdCF13]. Du point de vue applicatif, un service peut être dégradé pour diminuer sa consommation. Il faut alors jouer sur un compromis entre l'énergie dépensée et la qualité du service offert (comme le temps de réponse [Mar12]).

Dans le domaine automobile, la consommation d'un véhicule est au cœur de nombreux travaux de recherche. Le contrôle optimal de la chaîne de traction est étudié depuis de nombreuses années, il a évolué des véhicules équipés de moteur à combustion vers les véhicules hybrides et totalement électriques. Le moteur électrique d'un véhicule peut ainsi être dimensionné spécifiquement selon les trajets que le véhicule aura à accomplir [GUH14]. Aussi, dans [RSR07], le contrôle optimal de l'énergie des véhicules hybrides est étudié à la conception et est dérivé en lois de commandes pour être embarqué dans le véhicule. La dépense énergétique des véhicules électrifiés (hybrides ou tout électriques) est optimisable à la volée [BAC09, SSK03] ou en fonction du trajet effectué [MTB11, PS11]. D'autres systèmes peuvent également améliorer l'utilisation du véhicule comme la régulation de vitesse adaptative [LLLW10] ou encore la recharge d'une flotte de véhicules électrifiés [MVH+10].

Nous présentons tout d'abord une solution de représentation des besoins énergétiques des plates-formes de systèmes embarqués. Nous présentons ensuite deux solutions de gestion de l'énergie des processeurs selon différentes méthodes. En-

fin, nous abordons une solution d'optimisation de la consommation d'un véhicule électrique tenant compte de la destination et du confort utilisateur.

1.2.1 Modélisation et estimation des besoins d'un système multiprocesseurs sur puce

Présentation Open-PEOPLE (*Open Power and Energy Optimization PLatform and Estimator*) [SCZ+12] est une plate-forme d'estimation et d'optimisation des besoins énergétiques des systèmes électroniques complexes tels que les multiprocesseurs sur puce (*MultiProcessor System on Chip*, MPSoC). Différents niveaux architecturaux sont modélisés, du bloc processeur à l'implémentation de tâches effectuées par le système. Un système peut être hétérogène – composés de plusieurs processeurs spécialisés – ou à base de processeur reconfigurable (*Field-Programmable Gate Array*, FPGA). La plate-forme permet aux concepteurs d'évaluer différentes architectures possibles en définissant les tâches du système de manière logicielle ou de manière matérielle à l'aide de blocs dédiés.

La modélisation des éléments d'un système est réalisée avec le langage AADL (*Architecture Analysis & Design Language*) [FG12]. La méthodologie proposée est la suivante. L'architecture matérielle du système est modélisée de manière grossière, les tâches du système sont détaillées et plusieurs implémentations définies. La plate-forme permet aux concepteurs de parcourir et comparer les différentes solutions pour réaliser les tâches allouées au système, sur le plan énergétique, sur le plan architectural et sur le plan temporel. Une fois une solution choisie, les concepteurs peuvent raffiner leur modélisation pour obtenir des estimations de performance plus précises.

Synthèse Cette méthodologie requiert une modélisation complète du système étudié jusqu'à définir les blocs contenus dans les processeurs. De plus, les plates-formes doivent être instrumentées pour obtenir les modèles de consommation les plus détaillés. Ce côté fastidieux est compensé par des résultats précis permettant d'estimer les consommations de différentes architectures du système. Le découpage architectural permet l'étude à différents niveaux, voire à différentes étapes de la modélisation.

1.2.2 Gestion de la fréquence du processeur (monocœur)

Présentation Koala [SLPH09] est un outil prenant place au niveau du noyau d'un système d'exploitation. En utilisant un modèle de caractérisation du matériel déterminé hors-ligne, il est capable d'estimer la consommation énergétique d'un processus logiciel. Un compromis entre les performances et la consommation glo-

bale du système contrôlé est appliqué en ligne grâce à une politique du système d'exploitation et une variation de la fréquence du processeur.

La caractérisation du matériel est réalisée par une analyse exhaustive de résultats provenant de tests d'évaluation. Les compteurs de performance présents dans la plate-forme matérielle sont utilisés comme sources. La combinaison de compteurs permettant d'établir les prédictions les plus précises est sélectionnée comme base d'étude pour cette plate-forme matérielle.

À chaque fois qu'un processus logiciel est préempté ou bloqué, Koala estime sa consommation énergétique selon les statistiques liées aux compteurs de performance. À la prochaine exécution du processus, Koala détermine la nouvelle fréquence du processeur en fonction de la politique courante du système d'exploitation. Le changement de fréquence est pris en compte dans le calcul de l'état optimal, même si son surcoût est d'environ 1,5%, assurant de meilleures estimations au court du temps.

Synthèse La plate-forme apporte deux contributions, la première est un outillage du système d'exploitation et la seconde est une politique d'exécution selon l'axe performance–économie d'énergie.

L'outillage du système d'exploitation est réalisé hors ligne selon une recherche exhaustive. Le fait de baser la caractérisation de la plate-forme matérielle sur un ensemble dynamique de sources permet d'adapter l'outil à différents types de technologies. Par exemple, les auteurs montrent que leur serveur d'exemple est le plus sensible aux événements liés à la mémoire vive alors que l'ordinateur portable grand public testé montre une meilleure modélisation avec des événements du processeur. Mais malgré un grand nombre de paramètres pris en compte, la caractérisation peut tout de même s'avérée peu fiable. Les auteurs précisent ainsi que des modèles plus proche de la représentation physique (p. ex. fournis par les constructeurs) offriraient une plus grande précision. Étant donné la complexité inhérente aux modèles physiques, un usage en ligne impliquerait certainement une approximation qu'il faudrait comparer à l'approche empirique actuellement utilisée.

1.2.3 Gestion de la fréquence, de l'allocation et de la température du processeur (multicœurs)

Présentation Hanumaiah et Vrudhula [HV14] proposent une gestion énergétique d'un processeur multicœur à travers trois axes d'optimisation. Premièrement, la fréquence et la tension d'alimentation de chaque cœur sont définies séparément. Deuxièmement, les tâches logicielles en cours d'exécution sont allouées aux cœurs de manière dynamique, permettant de répartir ou de concentrer la charge. Enfin,

le troisième axe est la réduction de la température par la régulation du ventilateur refroidissant le processeur dans son ensemble. En effet, la température est non seulement une dissipation d'énergie consommée par le processeur mais elle induit également une dégradation des performances. Cette dégradation est compensée par une augmentation de la tension d'alimentation des cœurs concernés, augmentant ainsi leur demande énergétique.

Synthèse La modélisation des caractéristiques d'une plate-forme matérielle simulée est basée sur des résultats empiriques. Une analyse hors-ligne permet de construire un modèle thermique dont est extrait le modèle de consommation. Les autres caractéristiques de la plate-forme découlent de ce modèle grâce à une définition générique de leur comportement. En revanche, lorsque les auteurs présentent une application de leur approche sur une plate-forme matérielle réelle, ils créent leurs modèles grâce à une boucle fermée s'alimentant des valeurs des capteurs de la plate-forme. Cette différence de méthode est due aux manques de spécifications de la plate-forme réelle testée.

Les trois axes d'optimisation sont liés au même modèle mais ils sont contrôlés de manière séparée et décalée. Ces axes ne sont pas au même niveau de granularité, la fréquence est au niveau de la puce (contrôle en 10ms), l'allocation de tâche est au niveau applicatif (contrôle en 100ms), et la ventilation est au niveau du système physique (contrôle en 1s). Ainsi, les auteurs ont choisi la recherche d'optimum locaux (temporellement) pour permettre d'embarquer leur solution. Ils estiment que la performance d'une recherche basée sur l'historique dépend fortement de l'horizon considéré et qu'il est nécessaire d'avoir une connaissance de la demande future.

1.2.4 Guide de conduite non intrusif pour optimiser la consommation énergétique d'un véhicule électrique

Présentation Kachroudi et coll. proposent dans [KGA12] de guider la conduite d'un véhicule électrique en ligne pour optimiser sa dépense énergétique. Le système de décision proposé est basé sur la connaissance à priori du trajet à réaliser. La solution cherche le contrôle optimal du véhicule – temps de trajet et consommation énergétique – grâce à un algorithme d'optimisation par essaims particulaires (*Particle Swarm Optimisation*, PSO) [KE95]. Elle tient également compte du confort utilisateur en assurant la meilleure utilisation possible du chauffage selon la politique de consommation choisie (sportive, équilibrée ou économique). Enfin, cette solution est une aide autonome pour le conducteur qui reste décideur car elle ne fournit que des indications de guidage.

Synthèse Le confort utilisateur est considéré par le biais de la température de l'habitacle et est donc uniquement influençable par le fonctionnement du chauffage. La recherche de la solution optimale est faite simultanément pour les trois politiques de consommation. Le conducteur est informé des trois solutions en même temps, il peut décider de la politique la plus adéquate pour lui. En revanche, la solution proposée n'est pas adéquate si le conducteur possède un objectif particulier (p. ex. faire le trajet le plus rapidement possible) car elle ne cherche pas à satisfaire le conducteur mais à correspondre à trois politiques déterminées à la conception.

La solution proposée est exécutable en ligne, sur un contrôleur, grâce au coût contenu de l'algorithme d'optimisation. Les auteurs proposent une solution autonome hors de toute modélisation existante. Or, même si elle est non intrusive, elle requiert des informations précises du véhicule comme le niveau de la batterie, la position du véhicule ainsi que la modélisation physique du véhicule en lui-même. Ces informations sont accessibles *via* les systèmes embarqués (hormis certaines caractéristiques comme les besoins énergétiques) mais la solution n'y est pas intégrée.

Les solutions de gestion d'énergie sont nombreuses et existent aussi bien à haut niveau logiciel (p. ex. informatique dans le nuage [BAB12], systèmes à grande échelle [Mar12], *maison numérique* [DAP+13]) qu'au niveau du matériel (p. ex. contrôle de la mémoire [FEL01], conception d'un processeur dédié [WC05]). Dans tous les cas, les approches utilisent les différents niveaux de fonctionnement (ou de veille) disponibles pour optimiser la dépense énergétique du système géré. Celles qui prennent en compte la qualité de la solution fournie ne permettent pas explicitement l'optimisation selon cet axe supplémentaire.

La section suivante étudie la deuxième partie liée à nos problématiques, la gestion de la qualité de service et les architectures logicielles pour les intégrer.

1.3 Qualité de service et architecture logicielle

La qualité de service est une propriété décrivant le degré de satisfaction de la réalisation d'un service par un utilisateur. Elle est formellement définie par l'ITU (*International Telecommunication Union*) [ITU88] comme :

> "The collective effect of service performances which determine the degree of satisfaction of a *user* of the *service*."
>
> « Effet global produit par les caractéristiques d'un *service* fourni à un *usager* qui déterminent le degré de satisfaction que cet usager retire du *service*. »

et par l'ISO (*International Organization for Standardization*) [ISO98] comme :

> "A set of qualities related to the collective behaviour of one or more objects."
>
> « Ensemble de qualités se rapportant au comportement collectif d'un ou de plusieurs objets. »

Nous nous intéressons plus particulièrement à l'architecture logicielle et à la gestion de la qualité de service au sein d'un système logiciel. En rapprochant ces deux concepts, nous étudions en fait une architecture de gestion de qualité de service telles que présentées dans, par exemple, [CCG+93, DJP04, LTBP11, MLS+05, PCML09, RS04].

Campbell définit dans [Cam96] une architecture de gestion de qualité de service comme :

> "a set of quality of service configurable interfaces, services and mechanisms that formalise quality of service in the end-system and network, providing a framework for the integration of control, maintenance and management mechanisms to meet application level end-to-end QoS requirements."

Nous présentons maintenant une solution d'architecture de gestion de la qualité de service. Nous présentons ensuite trois solutions plus orientées architectures et configuration d'architectures avec une idée latente d'optimiser la qualité et l'efficience du service offert.

1.3.1 Architecture de gestion de la qualité de service dans les systèmes embarqués ouverts

Présentation Qinna [TBO05] est une architecture pour la gestion de la qualité de service dans les systèmes embarqués ouverts à base de composants. Elle permet la mise en œuvre et la gestion dynamique de contrats de qualité de service entre différents composants d'un système. Les mécanismes de qualité de service sont génériques et sont utilisés pour gérer, par exemple, l'ordonnancement du système d'exploitation et la mémoire vive allouée à un processus.

Les modèles à composants existants pour le domaine de l'embarqué proposent une vue fonctionnelle bien définie mais sans en proposer une pour la qualité de service. Qinna est basée sur la modélisation à composants Fractal qui sépare le contrôle d'un composant de son contenu. Ainsi le modèle Fractal assure la réalisation des trois premiers niveaux de contrats (l'interface, le comportement, et la synchronisation) [BJPW99] et Qinna assure le quatrième niveau (la qualité de service).

Synthèse L'architecture proposée s'adapte à tout système à composants et propose une gestion générique de la qualité de service. Cette généricité permet de rendre Qinna utilisable avec n'importe quel contexte. En raison de son approche abstraite, elle se situe au niveau des principes et non comme un outil final. Elle permet en effet d'avoir une vue systémique de la qualité de service mais nécessite d'avoir une connaissance des différents niveaux de modélisation pour représenter de manière cohérente la gestion de la qualité de service.

1.3.2 Configuration en ligne sur le long terme d'un logiciel embarqué

Présentation MELOADES [MAA13] est une méthodologie pour l'adaptation en ligne d'un logiciel embarqué sur un ensemble hétérogène de plates-formes. Elle est réalisée en deux étapes de configuration, l'une au déploiement et l'autre sur le long terme, en ligne. Plutôt que d'explorer toutes les configurations pour trouver les moins énergivores pour chaque déploiement, les auteurs proposent de reconfigurer le logiciel de manière automatique en ligne. Grâce à une table de mémoïsation, les configurations sont sauvegardées au fur et à mesure de sorte que lors d'une future utilisation du logiciel avec les mêmes contraintes, la configuration adéquate est déjà connue.

La table de mémoïsation est initialisée au déploiement grâce à un plan d'expériences faisant varier simultanément plusieurs paramètres de la configuration. Cette méthode permet d'obtenir une couverture des connaissances dans un large espace de recherche tout en minimisant le nombre d'évaluations. Chaque configuration est évaluée sur la plate-forme cible et les valeurs des contraintes – les métriques – sont enregistrées dans la table.

Ensuite, et à chaque utilisation du logiciel en ligne, la meilleure solution est cherchée dans la table de mémoïsation. Si elle n'y est pas, une recherche basée sur la solution la plus proche est lancée. Cette recherche s'arrête à la satisfaction des contraintes ou si la limite d'itération est atteinte. Les auteurs proposent quatre méthodes de recherche : deux heuristiques (algorithme génétique et *hill climbing*) et deux aléatoires (glouton et solutions uniques).

Synthèse La configuration automatique d'un logiciel assure deux points. Tout d'abord, les solutions retenues sont adaptées à la plate-forme courante et ne sont pas génériques à une famille de plate-forme. Comme ce résultat est atteint en ligne, il n'est pas nécessaire d'évaluer toutes les solutions possibles au déploiement, ce qui facilite et accélère la recherche de solution. Ensuite, la méthodologie proposée tend vers une simple table de correspondance des contraintes envers les solutions. Au final, les meilleures configurations sont connues du logiciel sans surcoût ma-

jeur hors-ligne tout en assurant l'adéquation des configurations à une plate-forme précise. Il faut en revanche noter que, comme toutes les configurations sont éventuellement connues, un espace mémoire conséquent – au regard de nos contraintes – est nécessaire.

1.3.3 Reconfiguration architecturale en ligne d'un système AUTOSAR

Présentation dTool [BGN+09] est un outil de modélisation permettant une adaptation en ligne de l'architecture d'un système AUTOSAR. Il s'intègre au processus de conception en décrivant l'adaptation *via* des connexions entre composants logiciels. Comme un modèle AUTOSAR ne peut pas contenir de variabilité architecturale, l'outil génère des nouveaux composants assurant le routage adéquat des données en fonction de l'évolution de l'environnement. Les auteurs estiment que le surcoût de temps de calcul induit par leur approche est moins d'une milliseconde pour un contrôleur embarqué standard.

Synthèse L'adaptation de l'architecture telle que proposée par les auteurs est explicitement définie à la conception, hors ligne. Elle prend place dans un modèle séparé pour parer aux limitations d'AUTOSAR. En intégrant des nouveaux composants dans l'architecture existante, l'adaptation en ligne de connexions est possible. Les composants non utilisés dans la configuration courante ne sont pas désactivés. Dans le cas où ceux-ci sont consommateurs non négligeables de ressources, il serait judicieux d'offrir un mécanisme d'interruption de leur exécution, par exemple, en modifiant leur signal de contrôle. La reconfiguration serait ainsi plus efficace en terme de ressources utilisées (et potentiellement d'énergie, à condition que le changement d'état des composants inutilisés ne soit pas plus coûteux).

1.3.4 Mobilité des composants logiciels pour l'efficience énergétique d'une *maison numérique*

Présentation L'approche proposée par Druilhe [Dru13] vise à une réduction de la consommation énergétique par la mobilité des composants logiciels au sein d'une *maison numérique*. L'auteur identifie un ensemble d'équipements fournisseurs de services pouvant également en accueillir de nouveaux. Par exemple, les ordinateurs personnels, les téléphones intelligents et les passerelles résidentielles répondent à ces critères. Chaque équipement propose des services réalisés par des composants logiciels.

À l'apparition (ou disparition) d'un équipement et au démarrage (ou arrêt) d'une application, un nouveau plan de déploiement des composants est calculé. Il

permet de réduire l'empreinte énergétique globale des équipements tout en conservant le niveau actuel de qualité de service. Les contraintes de déploiement considérées sont de deux natures : quantitatives (p. ex. 1Mio de RAM) et énumérées (p. ex. présence de l'utilisateur requise). Les composants logiciels sont ensuite migrés selon le plan de déploiement pour 1) maximiser l'utilisation des équipements requis par les contraintes de déploiement et 2) diminuer l'état énergétique (c.-à-d. passer en veille) des autres équipements.

Synthèse L'approche présentée permet la reconfiguration de la répartition des composants sur les équipements disponibles pour minimiser la consommation énergétique globale. En revanche, la qualité de service n'est pas considérée comme un axe d'optimisation mais est assimilée à une contrainte (il ne faut pas la faire baisser). Cette approche repose sur l'hypothèse que l'ensemble des services est réalisé par des composants logiciels. De plus, l'interopérabilité des plates-formes des équipements est supposée totale.

1.4 Conclusion

Le tableau 1.1 résume les différentes approches étudiées dans ce chapitre. Nous avons dans un premier temps abordé la problématique énergétique puis celles de la qualité de service et de l'architecture logicielle. Ces trois problématiques ne sont pas traitées explicitement, et d'une manière simultanée, par une solution unique.

Les modélisations présentées sont réalisées hors-ligne à partir de spécifications et de comportements connus ou bien en ligne à partir de mesures. La définition d'un modèle de consommation directement en ligne, par exemple si les spécifications sont absentes, requiert d'instrumenter chaque élément à modéliser. Dans le cas d'un véhicule électrique, il faudrait que la consommation de chaque organe soit surveillée en même temps que son contexte pour établir un modèle de consommation déterministe en fonction des services utilisés. Chaque modèle d'organe devrait être instrumenté, même ceux résultant d'une évolution déjà connue car ils seraient considérés en tant que boites noires.

Dans les véhicules, les modèles sont complexes (bas niveau) et ne sont pas destinés à être exécuté en ligne (mais utilisés pour les simulations). Quand il est nécessaire d'en embarquer pour un usage en ligne, ce sont des modèles approximatifs qui sont utilisés. Ces modèles permettent la mise en place d'une réaction mais pas de réaliser de prévisions.

Les modèles de caractérisation théorique permettent une définition hors-ligne et à partir des connaissances du domaine déjà existantes (documentation, expertise). Nous choisissons d'utiliser cette approche en supposant que les caractéristiques des organes sont connues ou accessibles aux concepteurs. Comme nous pouvons

TABLE 1.1 – Comparaison des solutions étudiées portant sur l'énergie, la qualité de service et l'architecture logicielle.

	Modélisation					Granularité [a]		Opti. [b]	Gestion
	empirique	théorique	architecure	énergie	qualité				
Open-PEOPLE	✓	✓	✓	✓		micro		✓	
Koala	✓			✓			macro		✓
Hanumaiah et coll.	✓	✓ [c]		✓		micro			✓
Kachroudi et coll.		✓		✓	✓		macro		✓
Qinna		✓	✓		✓		macro		✓
MELOADES	✓	✓ [d]	✓	✓		micro		✓	✓
dTool		✓	✓				macro	✓	✓
Druilhe et coll.		✓	✓	✓	≈ [e]		macro		✓

a. Caractérisation des éléments ou événements au niveau de la couche physique (micro) ou du système (macro).

b. Optimisation (p. ex. architecturale, configuration) hors-ligne.

c. Le modèle de consommation est empirique, les modèles de convection de la ventilation et de l'allocation de tâches sont théoriques.

d. Une partie des solutions est initialisée au déploiement, les solutions inconnues sont sauvegardées au fur et à mesure de leur résolution.

e. La qualité de service doit être conservée mais l'approche ne cherche pas à l'améliorer.

le voir dans le tableau récapitulatif, les approches étudiées ne permettent pas de caractériser explicitement les besoins énergétiques et les niveaux de qualité tout en apportant une solution adaptative. Ce problème est résolu par la définition d'un langage de modélisation propre, permettant la caractérisation énergétique et qualitative avec une adaptation architecturale.

La gestion des propriétés extra-fonctionnelles est, dans l'ensemble des approches présentées, réalisée de manière architecturale par l'ajout de composants d'adaptation. La méthodologie d'AUTOSAR ne permet pas la représentation de propriétés extra-fonctionnelles, et donc de la qualité de service. En revanche, comme AUTOSAR suit le paradigme des composants logiciels, nous pouvons l'enrichir par de nouveaux composants tout en conservant la compatibilité avec les outils existants. Nous proposons une solution tirant partie du savoir-faire existant sans rompre les processus de développement actuels. Le modèle architectural est ainsi enrichi à la fois d'une caractérisation de consommation et d'une gestion de la qualité de service.

Chapitre 2

ORQA

Sommaire

Ce chapitre présente ORQA, un canevas logiciel pour la mise en place d'un gestionnaire d'énergie à destination des véhicules électriques. À la demande du conducteur, le gestionnaire d'énergie propose une solution optimale selon le trajet demandé et ses préférences d'ordre qualitatif. Nous commençons par présenter le canevas logiciel d'une façon générale, ses objectifs et son fonctionnement. Nous détaillons ensuite les différentes parties de la phase de conception qui vont conduire au gestionnaire d'énergie :

- la caractérisation des organes consommateurs du véhicule ciblé ;
- la définition de la prise de décision (effectuée en ligne) de la solution optimale ;
- la configuration du gestionnaire (à la conception) adaptée aux capacités du véhicule.

Enfin nous présentons l'architecture logicielle adoptée pour le gestionnaire d'énergie.

2.1 Présentation

Le canevas logiciel ORQA a pour but d'offrir un service global au conducteur sur un trajet, de la recherche de la meilleure stratégie de conduite à l'utilisation optimale des organes du véhicule. Cette finalité est assurée par un gestionnaire d'énergie qui cherche à améliorer l'utilisation du véhicule – et de ses composantes – tout en offrant la meilleure qualité de service possible au conducteur. Le canevas logiciel suit l'ingénierie dirigée par les modèles (IdM), il repose sur un ensemble de modèles pour matérialiser les connaissances et représenter l'architecture du gestionnaire d'énergie. Tout d'abord, tous les organes consommateurs du véhicule doivent être caractérisés à la conception pour apporter la connaissance sur leurs besoins énergétiques (→ modèle de caractérisation des organes). Les concepteurs réutilisent ensuite les modèles fonctionnels déjà existants des systèmes embarqués.

À l'exécution (dans le véhicule), les solutions accessibles pour rejoindre la destination du conducteur sont évaluées pour ne retenir qu'une solution jugée optimale (→ modèle de décision). La configuration adéquate des paramètres d'exploration permet une exécution sur les systèmes embarqués, contraints par définition. La configuration est affinée aux capacités du véhicule à la conception (→ modèle de configuration). Lors de la dernière étape de conception, le gestionnaire d'énergie est intégré au système logiciel du véhicule au sein des systèmes embarqués.

D'un point de vue global, nous distinguons la phase de conception d'ORQA de sa phase d'exécution dans le véhicule. La conception est scindée en trois parties : 1) la caractérisation des organes consommateurs du véhicule, 2) la définition de la prise de décision de la solution optimale et 3) la configuration du gestionnaire d'énergie par rapport au véhicule cible. L'exécution du gestionnaire d'énergie est la

phase d'utilisation par le conducteur dans le véhicule. Une demande de trajet déclenche le calcul d'une stratégie de conduite optimale appliquée par le gestionnaire d'énergie pendant le trajet.

2.1.1 Objectifs du canevas logiciel

Pour offrir un service de qualité au conducteur, le canevas logiciel requiert une connaissance systémique du véhicule. La connaissance architecturale du véhicule est définie dans les modèles des systèmes embarqués, c'est l'architecture fonctionnelle du véhicule qui y est représentée. Néanmoins, les modèles existants n'intègrent pas d'aspects extra-fonctionnels du type besoin énergétique et qualité de service. Le premier objectif du canevas logiciel est donc de représenter ces propriétés et de les rendre accessible au système logiciel.

La gestion énergétique du véhicule basée sur l'ensemble de ses éléments consommateurs et de la qualité de service globale constitue le deuxième objectif d'ORQA. Par le biais d'un gestionnaire d'énergie logiciel, ORQA propose au conducteur une solution répondant à sa requête (rejoindre une destination) qui lui offre la meilleure qualité de service possible. Tous les éléments du véhicule sont pris en compte par leurs besoins énergétiques et leurs qualités de service. Le conducteur indique quelles sont ses préférences d'utilisation entre les différents organes et peut également spécifier des contraintes sur le trajet telle qu'un niveau minimal d'énergie restante.

De manière résumée, les objectifs du canevas logiciel ORQA sont :

1. modéliser les propriétés extra-fonctionnelles de besoin énergétique et de qualité de service des éléments du véhicule ;
2. proposer une gestion de l'énergie prenant en considération le conducteur et la qualité de service que le véhicule lui offre.

La méthodologie du canevas logiciel pour répondre à ces objectifs est illustrée dans la figure 2.1. Nous nous plaçons dans un premier temps à la conception, lorsque la modélisation du véhicule est en passe d'être complétée. La première étape est l'extraction de la modélisation des organes du véhicule depuis les modèles des systèmes embarqués. Les organes sont ensuite caractérisés par leurs besoins énergétiques et leurs qualités de service. Cette connaissance des organes permet de générer le gestionnaire d'énergie. De plus, il est intégré dans les systèmes embarqués pour pouvoir être exécuter dans le véhicule. La caractérisation des organes permet d'étudier les capacités spécifiques du véhicule et de créer une configuration adéquate. Passons maintenant à la phase d'exécution, lorsque le conducteur souhaite atteindre une destination. Il effectue une requête au gestionnaire d'énergie qui va explorer l'espace de solution. Au final, le gestionnaire d'énergie sélectionne la

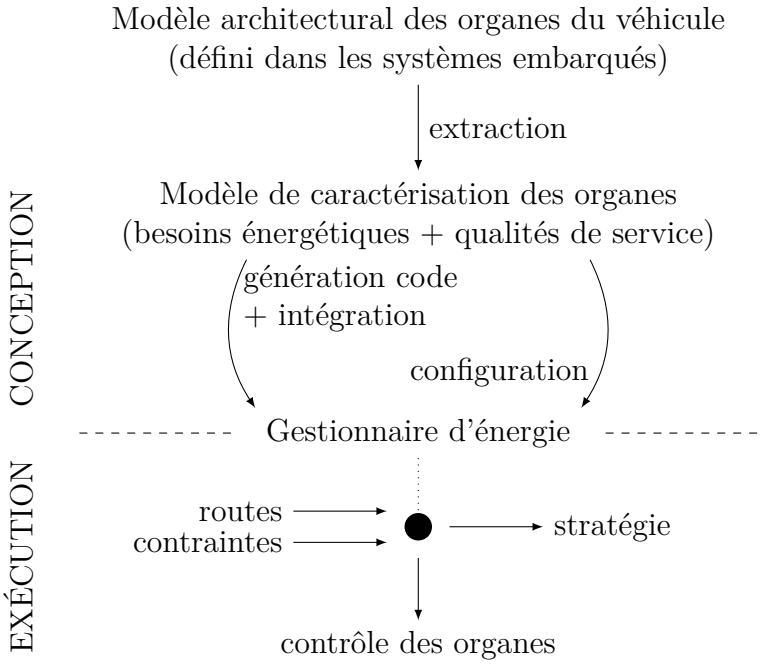


FIGURE 2.1 – Méthodologie du canevas logiciel ORQA.

solution optimale (appelée stratégie de conduite) et débute le contrôle des organes pour assurer la réalisation de cette solution selon les résultats estimés.

2.1.2 Le véhicule comme un système à composants

Un véhicule est une structure regroupant des organes physiques. Nous nous intéressons uniquement aux organes commandés de manière électrique ou électronique dont il est possible de quantifier la demande énergétique. Intrinsèquement, ces types d'organes possèdent une interface logicielle de gestion ou d'information. Ils sont donc nécessairement représentés, à un moment de la conception, dans une architecture logicielle. Nous partons de l'hypothèse que cette architecture suit le paradigme des composants logiciels (comme c'est le cas pour les modèles AUTOSAR, par exemple).

Nous utilisons le terme de *service* pour désigner une dépendance ou une offre d'un composant. Ainsi, un service requis par un composant A pourra être satisfait par le service offert par un composant B si leurs interfaces sont compatibles.

Les composants logiciels qui nous intéressent sont ceux représentant les organes physiques que l'on trouve dans le véhicule. Ils permettent d'établir la relation entre le monde logiciel et le monde matériel du véhicule. Étant donné que, du point de vue logique, les composants logiciels sont les organes du véhicule, nous

utilisons les termes *organes* et *composants* de manière indifférenciée pour désigner les composants logiciels.

Comme nous supposons que les organes physiques d'un véhicule sont représentés par des composants logiciels dans un modèle architectural, nous étudions maintenant la manière d'interpréter les propriétés extra-fonctionnelles de besoin énergétique et de qualité de service de ces composants.

2.2 La modélisation des organes consommateurs

Cette section détaille la modélisation retenue pour les organes consommateurs ainsi que leurs spécificités énergétiques et qualitatives. Nous appelons cette modélisation la *caractérisation des organes consommateurs* du véhicule. Le métamodèle représenté dans la figure 2.2 organise les concepts définis dans ORQA que nous détaillons maintenant.

2.2.1 Les types d'organes

La base du modèle de caractérisation est un système *(System)* représentant le véhicule composé d'organes *(Devices)*. Un organe embarqué dans un véhicule rend un service qui peut être de deux natures différentes : de connaissance ou d'action. Un service de connaissance permet au système de s'informer sur son environnement *via* des capteurs. Par exemple, une photocellule va renseigner le système sur la luminosité ambiante. Un service d'action propose au système, *via* des actionneurs, d'agir sur son environnement. Si la luminosité ambiante n'est pas suffisante pour une bonne visibilité de la chaussée, le système peut décider d'enclencher l'éclairage du véhicule (et de fait, agir sur les ampoules du véhicule).

Les organes liés à la sécurité (par exemple la direction assistée, le correcteur électronique de trajectoire, le système d'éclairage) assurent la sûreté et le bon fonctionnement du véhicule, leur comportement ne doit pas être altéré par un système tiers. Ce sont néanmoins des organes consommateurs d'énergie. Afin d'avoir la représentation systémique de la consommation du véhicule, il est nécessaire pour le canevas logiciel de connaître leurs besoins énergétiques. Ces organes ne sont qu'observables *(ObservableDevices)* par le logiciel, au contraire des organes contrôlables *(ControllableDevices)* qui sont gérés de manière optimale par ORQA à l'exécution.

Nous définissons l'ensemble des organes D du véhicule comme la combinaison des organes observables D^{obs} et des organes contrôlables D^{ctrl} :

$$D = D^{\text{obs}} \cup D^{\text{ctrl}} \tag{2.1}$$

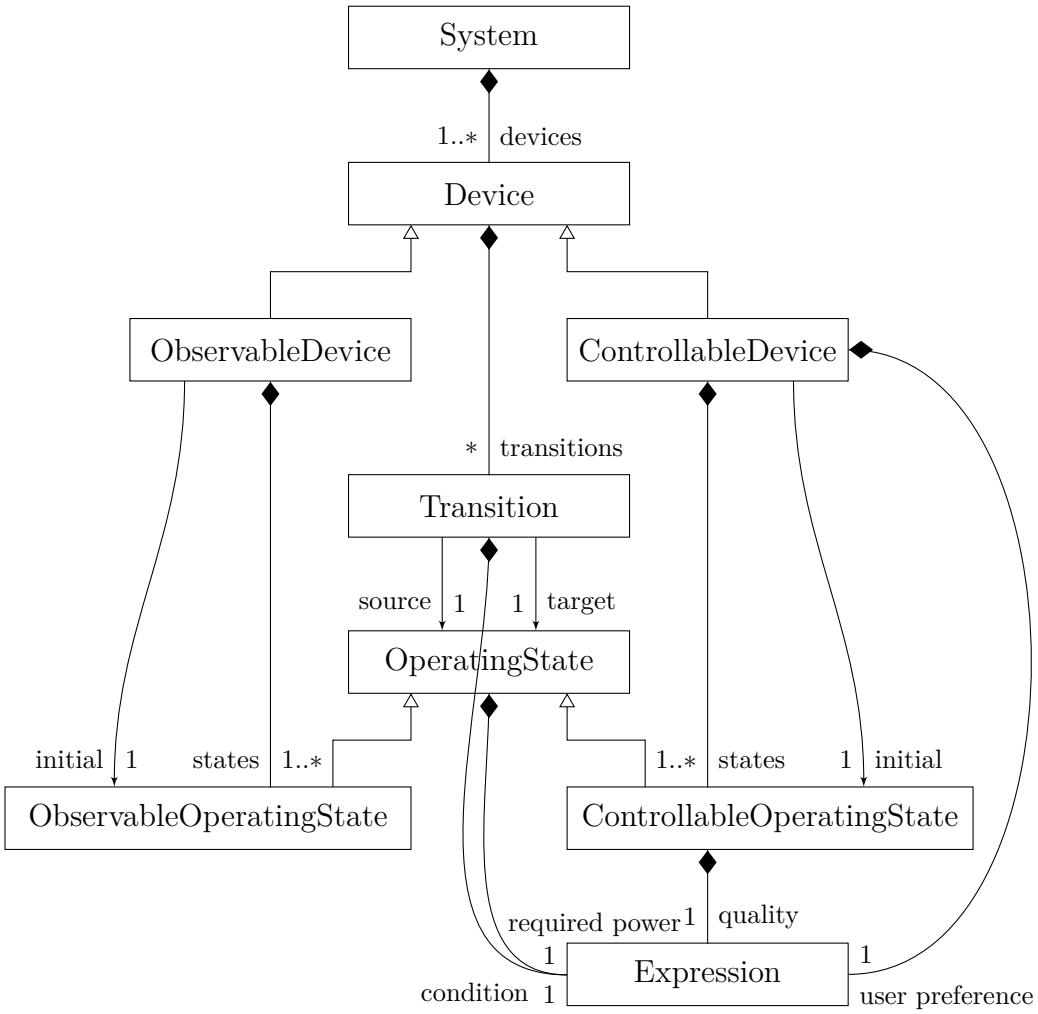


FIGURE 2.2 – Métamodèle de caractérisation des organes consommateurs d'un véhicule.

2.2.2 Le comportement des organes

Le service rendu par un organe est effectué à plusieurs niveaux, appelés modes opératoires. Par exemple, le mode opératoire peut être « en veille » ou « en fonctionnement nominal ». Chaque mode opératoire de l'organe permet de réaliser son service d'une certaine façon, typiquement avec des besoins énergétiques différents et des qualités de service différentes. Nous formalisons dans un premier temps le comportement général d'un organe commun aux organes observables et aux organes contrôlables. Si l'on se réfère au métamodèle de caractérisation (cf. figure 2.2), nous définissons le comportement d'un organe par une machine à état où les transitions *(Transitions)* opèrent sur les modes opératoires appelés *OperatingStates*. Nous définissons également l'usage énergétique de chaque mode opératoire *via* des expressions mathématiques *(Expressions)*.

Le comportement d'un organe est donc défini par un automate fini déterministe :

$$\langle O,C,T,o_0 \rangle \tag{2.2}$$

Ce quadruplet est composé de l'ensemble fini des modes opératoires O (ou états), de l'ensemble fini des conditions C, de l'ensemble fini des transitions T et de l'état initial $o_0 \in O$.

L'ensemble des conditions $C \subset \Omega$ contient les conditions de garde entre deux états. Une condition est une expression à valeur booléenne. De leur côté, les expressions Ω du système peuvent être booléennes, entières ou bien réelles. Une expression du système est une expression mathématique constante ou évaluée à l'exécution. Elle est définie (cf. figure 2.3) par une composition de variables, de valeurs littérales et d'opérateurs mathématiques. Nous utilisons ici une modélisation simple et classique que nous incluons dans notre métamodèle pour éviter toute dépendance.

L'ensemble fini des états O est l'ensemble des modes opératoires de l'organe. L'ensemble des transitions T contient les triplets correspondants aux passages d'un état à un autre :

$$T : O \times C \to O \tag{2.3}$$

Une transition $t \in T$ est définie par une condition $c \in C$ gardant le passage d'un état o_i à un état o_j ($o_i,o_j \in O$) :

$$t = \langle o_i,c,o_j \rangle \tag{2.4}$$

Enfin, l'état initial o_0 est le mode opératoire de départ de l'organe, celui dans lequel il se trouve au démarrage du véhicule.

La définition classique d'un automate fini à état tient compte des états finaux du système. Ainsi, lorsque le système atteint un état final, l'automate termine.

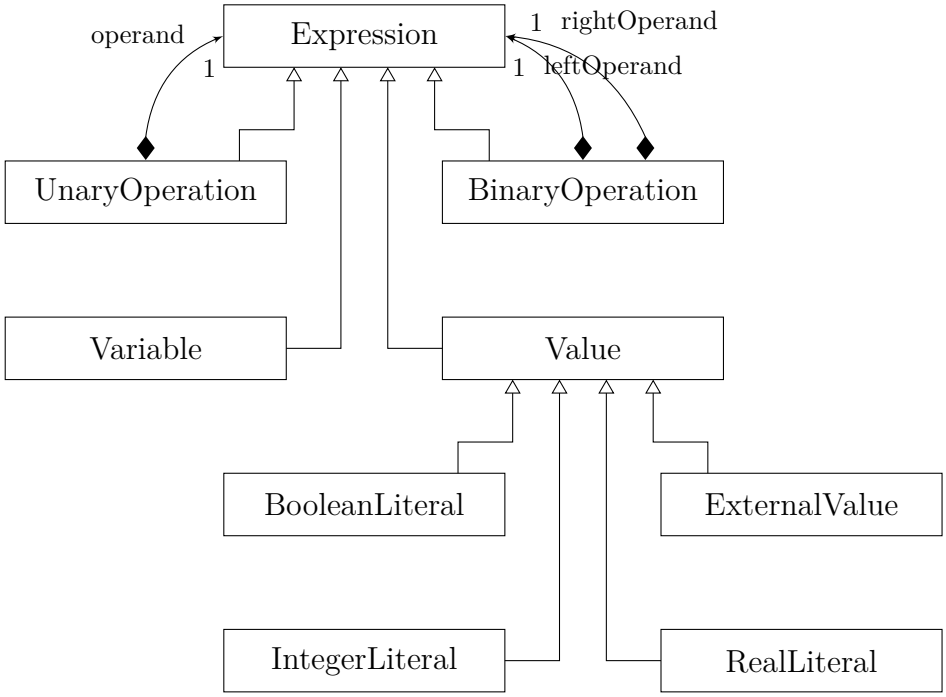


FIGURE 2.3 – Métamodèle des expressions.

ORQA ne tient pas compte des états finaux des organes s'il y en a. Nous considérons en effet que les organes sont opérables tant que le véhicule fonctionne. Cela signifie qu'un organe éteint peut être rallumé car l'automate représente le fonctionnement de l'organe dans le véhicule, et non son fonctionnement autonome.

Chacun des deux types d'organes (observable et contrôlable) possèdent des modes opératoires adaptés à son type, *ObservableOperatingState* pour les organes observables et *ControllableOperatingState* pour les organes contrôlables. Tous les organes étant consommateurs, les deux types de modes opératoires sont caractérisés par leurs besoins énergétiques.

2.2.3 Les besoins énergétiques d'un mode opératoire

Les besoins énergétiques d'un organe sont une notion dépendante du temps et de la façon dont l'organe réalise son service. La dépendance à la réalisation du service renvoie aux modes opératoires de l'organe. La notion d'énergie est la puissance requise sur une certaine durée de temps (une puissance multipliée par une durée). Le modèle énergétique d'un organe correspond en fait à la définition de la puissance requise par ses différents modes opératoires.

La puissance requise d'un mode opératoire est représentée par une expression réelle, évaluée ensuite pour donner la puissance en watts (W) requise par l'organe.

Elle peut être constante et représenter une puissance fixe ou référencer des variables et des fonctions pour représenter une puissance contextuelle, elle est alors évaluée à l'exécution du système. Par exemple, chaque ampoule du système d'éclairage a une puissance d'opération nominale fixe. A contrario, la puissance requise par le moteur pour déplacer le véhicule dépend de paramètres variables tels que la vitesse et l'accélération du véhicule, ou encore la pente de la route. Si l'on définit P comme étant l'ensemble fini des puissances requises d'un organe, le comportement d'un organe dont les besoins énergétiques sont définis devient :

$$\langle O,C,T,P,B,o_0 \rangle \tag{2.5}$$

B est l'ensemble fini des couples définissant les besoins énergétiques des modes opératoires :

$$B : O \rightarrow P \tag{2.6}$$

Nous avons donc un besoin énergétique $b \in B$ d'un état $o_i \in O$ caractérisé par une puissance $p \in P$ tel que :

$$b = \langle o_i,p \rangle \tag{2.7}$$

Notons que comme les puissances requises sont représentées par des expressions mathématiques, nous avons également $P \subset \Omega$.

Les principes pour la définition de la puissance requise par un mode opératoire sont communs aux deux types d'organes. Pour ce qui est de la qualité de service offerte par le véhicule, les organes observables ne sont pas gérés par le canevas logiciel, leur état n'étant pas influencé par ORQA. Leur qualité de service n'est donc pas considérée car elle est subie et ne peut être optimisée. En revanche, les organes contrôlables sont gérés par le canevas logiciel à l'exécution, leur caractérisation englobe également la qualité de service qu'ils offrent. En conséquence, nous définissons le comportement caractérisé d'un organe observable $d \in D^{\text{obs}}$ par :

$$d = \langle O,C,T,P,B,o_0 \rangle \tag{2.8}$$

Nous détaillons maintenant la représentation de la qualité de service d'un organe contrôlable.

2.2.4 La qualité de service d'un organe contrôlable

Les organes contrôlables possèdent intrinsèquement un caractère qualitatif. ORQA utilise cet aspect afin de choisir une solution qui tient compte de la qualité de service rendue à l'utilisateur. La qualité de service d'un organe dépend de la réalisation du service (quel est le mode opératoire courant) et de la préférence du conducteur sur ce service particulier. Nous détaillons maintenant ces deux parties, la première qui a trait aux modes opératoires et la seconde qui se trouve au niveau des organes eux-mêmes.

2.2.4.1 La qualité de service d'un mode opératoire contrôlable

Le mode opératoire d'un organe est une manière de réaliser un service. Il est naturel de classifier ces modes opératoires dans le cas d'un organe contrôlable. En effet, comme le gestionnaire d'énergie gère l'organe à l'exécution, il est nécessaire de définir une classification pour pouvoir sélectionner un mode opératoire. Nous nous appuyons sur le niveau de qualité de service rendu au conducteur.

Cette qualité est modélisée par une expression réelle comprise entre 0 et 1, ce qui revient à utiliser un pourcentage. En général, un mode opératoire ayant un besoin énergétique constant offre une qualité de service fixe (c.-à-d. une expression constante). Et réciproquement pour un mode opératoire dont le besoin énergétique est fluctuant, sa qualité de service est généralement variable. Une qualité de service de 1 (100%) indique un mode opératoire optimal. Une qualité de service de 0 (0%) indique que le service n'est pas réalisé quand l'organe se trouve dans ce mode opératoire. Notons qu'un organe ne rendant pas son service (et donc opérant dans un mode avec une qualité de 0%) peut avoir un besoin énergétique non nul. C'est le cas lorsque l'organe est en veille, le service n'est pas réalisé mais le besoin énergétique n'est pas nul.

2.2.4.2 La préférence utilisateur d'un organe contrôlable

La préférence utilisateur d'un organe représente l'intérêt porté par le conducteur sur l'utilisation de cet organe. À chaque organe contrôlable est affecté une valeur entre 0 (pas d'intérêt) et 1 (intérêt maximal). Nous ajoutons ici une condition supplémentaire, la somme des préférences utilisateur doit être égale à 1, ou encore 100%. Du point de vue de l'utilisateur, c'est un classement pondéré des organes contrôlables qui suit ses attentes. Par exemple, dans le cas d'un système composé de deux organes d_i et d_j classés par des préférences u_i et u_j. Si $u_i = 20\%$ et $u_j = 100\% - u_i = 80\%$, alors l'organe d_i est quatre fois moins intéressant aux yeux du conducteur que l'organe d_j (car $\frac{u_i}{u_j} = \frac{20\%}{80\%} = \frac{1}{4}$). Un classement par défaut est défini par les concepteurs lors de la conception. Le conducteur peut mettre à jour ces valeurs selon ses propres préférences.

Si l'on définit L comme étant l'ensemble fini des expressions des qualités de service des modes opératoires d'un organe contrôlable et $u \in \Omega$ la préférence utilisateur de cet organe, nous pouvons définir le comportement d'un organe contrôlable $d \in D^{\text{ctrl}}$ par :

$$d = \langle O,C,T,P,B,L,A,o_0,u \rangle \tag{2.9}$$

A est l'ensemble fini des couples définissant les qualités de services des modes opératoires contrôlables :

$$A : O \to L \tag{2.10}$$

Nous avons donc la qualité de service du mode opératoire d'un organe $a \in A$ offerte dans l'état $o \in O$ et de valeur $l \in L$ telle que :

$$a = \langle o,l \rangle \tag{2.11}$$

Les expressions mathématiques sont utilisées pour représenter les qualités de service, nous avons donc $L \subset \Omega$.

Nous obtenons finalement la qualité de service d'un organe à un moment donné en multipliant la qualité de service a du mode opératoire courant par la préférence utilisateur de l'organe u :

$$a \times u \tag{2.12}$$

Nous avons défini le comportement caractérisé d'un organe observable par l'équation (2.8), puis celui d'un organe contrôlable par l'équation (2.9). En réutilisant les notations définies précédemment, nous pouvons illustrer la modélisation de deux organes consommateurs, un organe observable d_i et un organe contrôlable d_j. La figure 2.4 illustre ces deux organes en utilisant le formalisme classique des machines à état. De plus, les modes opératoires d'un organe observable possèdent une étiquette représentant leur puissance requise. Les modes opératoires contrôlables possèdent en plus une étiquette pour leur qualité de service offerte. Enfin, les organes contrôlables ont une étiquette supplémentaire pour leur préférence utilisateur. Dans cette illustration, la qualité de service de l'organe contrôlable d_j vaut $l_0 \times u$ quand l'organe opère en o_0 et $l_1 \times u$ quand il opère en o_1.

Grâce au modèle de caractérisation des organes consommateurs, le canevas logiciel possède la connaissance des besoins énergétiques et de la qualité de service offerte par le véhicule. Ces connaissances sont intégrées et utilisées à l'exécution par le gestionnaire d'énergie implémentant ORQA pour répondre à une requête du conducteur. Nous détaillons maintenant comment le gestionnaire d'énergie décide de la meilleure solution à la requête du conducteur à partir des éléments du modèle de caractérisation.

2.3 La prise de décision

Le gestionnaire d'énergie utilise à l'exécution une logique de sélection pour déterminer la meilleure solution selon les souhaits du conducteur. L'évènement déclencheur du processus est la requête du conducteur pour rejoindre une destination. Le conducteur exprime sa demande de qualité de service rendue par les organes *via* des contraintes imposées à la solution. Il définit également une politique de consommation détaillant ses attentes sur le trajet solution. Le gestionnaire d'énergie récupère les routes accessibles pour atteindre la destination. Ces routes forment la base de l'espace de solution explorée par ORQA, elles sont raffinées par des coefficients de vitesse pour évaluer différentes conditions de conduite que le conducteur

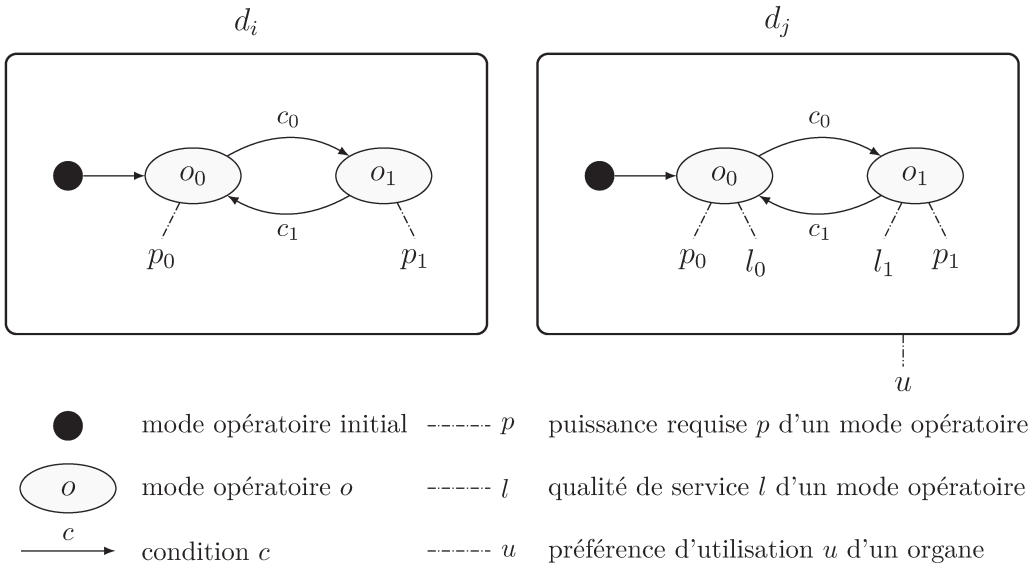


FIGURE 2.4 – Exemple de modélisation de deux organes consommateurs. L'organe d_i est observable et l'organe d_j est contrôlable.

pourrait suivre. Le classement des solutions et la politique de consommation du conducteur permettent à ORQA de sélectionner une solution répondant à ses attentes. Nous détaillons maintenant les éléments intervenants dans cette prise de décision : les coefficients de vitesse appliqués aux routes accessibles, les contraintes fixées par le conducteur et finalement, les fonctions de construction des scores. Enfin, nous présentons les solutions que nous appelons stratégies de conduite.

2.3.1 Les coefficients de vitesse

Les routes potentielles d'un trajet sont récupérées depuis le système logiciel du véhicule (typiquement depuis le GPS). Dans ORQA, une route est modélisée *via* les conditions nominales de conduite d'un voyage, c'est à dire les vitesses maximales autorisées par portion. Une route est composée d'un ensemble fini de portions atomiques qui sont des triplets de la forme suivante :

$$\langle v,d,\alpha \rangle \tag{2.13}$$

où v est la vitesse maximale de la portion, d est sa distance et α sa pente. La vitesse maximale d'une portion est la vitesse de croisière que le véhicule cherche à atteindre sur cette portion. En fonction de son accélération et de la distance de la portion, le véhicule atteint la vitesse de croisière ou s'en approche seulement.

De manière générale, un véhicule dépense le plus d'énergie dans son déplacement. La puissance nécessaire pour déplacer un véhicule est fortement influencée

par sa vitesse[1]. Étant donné que le problème général des véhicules électriques est leur faible capacité énergétique, nous choisissons de proposer des solutions alternatives – requérant moins d'énergie – par une diminution de la vitesse du véhicule.

Nous proposons de raffiner les routes potentielles par l'application d'un coefficient de vitesse. Un coefficient de vitesse $k_v \in [1\%; 100\%]$ est un pourcentage représentant la diminution de vitesse du véhicule par rapport aux conditions nominales de conduite. La nouvelle route raffinée est une variation de la route initiale et représente des conditions de conduite différentes telles que ses portions de routes sont définies par :

$$\langle v \times k_v, d, \alpha \rangle \tag{2.14}$$

Ce raffinement permet à ORQA d'explorer différentes conditions de conduite des routes récupérées et élargit l'espace de solution à explorer. La portion d'une route sur laquelle la vitesse maximale autorisée est de 90 km/h sera par exemple limitée à 63 km/h pour sa variation à 70%. Un coefficient de vitesse ne peut bien sûr pas dépasser 100%. Les vitesses nominales sont en effet celles définies par la législation qui s'applique à la route.

Dans le suite de cette thèse, nous utilisons le terme générique *route* pour désigner l'ensemble des routes raffinées, c'est à dire les routes auxquelles sont appliqués les coefficients de vitesse. Comme un coefficient de vitesse de 100% ne modifie pas les vitesses d'une route, nous appelons les routes raffinées par un coefficient de 100% les routes nominales. Les autres routes raffinées sont simplement désignées par le terme de variations. Les notations correspondantes sont les suivantes pour un ensemble de routes R :

$$R = \{\bar{r} = r_{100}, r_{99}, ..., r_1\} \tag{2.15}$$

où R est l'ensemble fini des routes variations d'une route et r_n est la variation de la route au coefficient $n\%$. Notons que la route nominale a une notation spécifique : $\bar{r}$.

2.3.2 Les contraintes fixées par le conducteur

Le conducteur définit la destination cible ainsi que deux jeux de contraintes dans sa requête : les préférences utilisateur sur les organes contrôlables et la politique de consommation. La politique de consommation d'un trajet est la politique définie par le conducteur à atteindre par ORQA. Elle porte sur les trois résultats (durée, consommation et qualité) des solutions.

Aux yeux du conducteur, l'importance d'un résultat par rapport aux autres peut être différente à chaque utilisation du véhicule. Prenons en exemple le cas

1. Ce point est décrit plus avant au chapitre 3, page 57.

d'une personne se rendant à son lieu de travail avec son véhicule. Elle souhaite effectuer son trajet pour aller rapidement au point d'arrivée, tout en conservant au moins la moitié de l'énergie du véhicule pour pouvoir faire face aux imprévus. En revanche, son trajet retour pourrait se faire de manière plus économique et confortable, sa durée important moins. Cette personne utilisera donc une politique de consommation qui accentue l'importance de la durée du trajet à l'aller et une politique de confort au retour portant plutôt sur la qualité de service. Nous représentons cette notion d'importance par une pondération des trois résultats (durée, consommation et qualité de service). Ces résultats sont utilisés pour classer les solutions comme décrit ci-après.

La pondération s'appuie sur les trois poids w_T, w_E, w_Q. Ces derniers portent respectivement sur la durée d'un trajet, sa consommation et sa qualité de service. De la même manière que pour la répartition de la préférence utilisateur entre les organes, les poids sont compris entre 0 et 1 et leur somme est égale à 1 :

$$w_T + w_E + w_Q = 1 \quad \text{avec } w_T, w_E, w_Q \in [0;1] \tag{2.16}$$

Pour plus de lisibilité, nous notons un poids en terme de pourcentage. Un poids élevé indique la sévérité d'un résultat par rapport aux autres. Si les poids portant sur la durée ou la consommation sont élevés, alors ces résultats sont considérés importants et ils doivent être minimisés. Au contraire, une qualité importante doit être maximisée. En se basant sur l'exemple ci-dessus, nous pouvons avoir les valeurs suivantes pour la première politique portant sur l'économie d'énergie : $w_T = 20\%$, $w_E = 60\%$ et $w_Q = 20\%$. La seconde politique, à vocation qualitative, peut définir les pondérations par : $w_T = 20\%$, $w_E = 30\%$ et $w_Q = 50\%$.

En plus des pondérations, il est possible d'adjoindre de manière optionnelle des contraintes sur la durée et la consommation énergétique d'une solution. Ces contraintes s'appuient sur la solution nominale $\bar{s}$. Cette solution se base sur la version nominale de la route de s et propose une combinaison d'organes dont la qualité est maximale. La contrainte de durée χ_T est exprimée comme un délai maximal par rapport à la solution nominale :

$$\chi_T \geq \frac{T_s}{T_{\bar{s}}} \tag{2.17}$$

où T_s est la durée d'une solution s et $T_{\bar{s}}$ est la durée de la solution nominale $\bar{s}$. Par exemple, $\chi_T = 2$ pour un délai maximal de deux fois la durée de la solution nominale.

De son côté, la contrainte de consommation χ_E est exprimée comme un niveau d'énergie minimum à conserver dans la batterie :

$$\chi_E \leq \frac{E_{\text{disponible}} - E_s}{E_{\text{capacité}}} \tag{2.18}$$

où E_s est l'énergie requise par une solution s, $E_{\text{disponible}}$ est le niveau actuel d'énergie disponible et $E_{\text{capacité}}$ est la capacité énergétique du véhicule (c.-à-d. la quantité maximale d'énergie stockée). Par exemple, $\chi_E = 20\%$ pour un niveau d'énergie à conserver d'au moins 20%.

De la même façon que la préférence d'utilisation des organes est pré-réglée à la conception, les politiques de consommation sont également prédéfinies. L'impact des pondérations est difficilement quantifiable tel quel, l'intention est modélisée par les valeurs mais son influence doit être évaluée.

2.3.3 Les stratégies de conduite

Une solution possible à un trajet est appelée une stratégie de conduite. Elle représente une proposition évaluée par ORQA pour répondre à la requête du conducteur. Une stratégie de conduite λ regroupe une route r (nominale ou variation) et une combinaison d'organes g à des modes opératoires définis :

$$\lambda = \langle r,g \rangle \tag{2.19}$$

Une combinaison d'organes détaille le comportement de chaque organe contrôlable au cours du trajet. C'est à dire qu'elle associe à chaque organe contrôlable un mode opératoire :

$$g : D^{\text{ctrl}} \to O \tag{2.20}$$

L'ensemble des stratégies de conduite Λ est créé en associant chaque route récupérée du système et raffinée à chaque combinaison d'organes :

$$\Lambda = R \times K_v \times G \tag{2.21}$$

où R est l'ensemble fini des routes possibles, K_v l'ensemble fini des coefficients de vitesse et G l'ensemble fini des combinaisons d'organes.

Une stratégie de conduite évaluée grâce au modèle de caractérisation des organes donne trois résultats : i) la durée du trajet, ii) la consommation énergétique globale du véhicule et iii) la qualité de service globale du trajet. Les deux premiers résultats découlent uniquement de l'évolution du véhicule sur la route de la stratégie alors que le troisième dépend également des contraintes et préférences fixées par le conducteur. La durée du trajet T est le temps nécessaire au véhicule pour parcourir les différentes portions de la route selon les vitesses définies (nominales ou réduites par les coefficients de vitesse). La consommation énergétique E du véhicule est la somme des consommations de tous ses organes au long du trajet :

$$E = \sum_{d \in D} \sum_{o \in O} \left(B(o) \times \tau(d,o) \right) \tag{2.22}$$

où l'ensemble O et la fonction $B : O \rightarrow P$ sont relatifs à l'organe d courant et $\tau(d,o)$ est la durée pendant laquelle l'organe d opère dans le mode o.

Finalement, la qualité de service du trajet Q est la somme des qualités de service rendues par les organes pendant le trajet :

$$Q = \sum_{d \in D} \left(u \times \sum_{o \in O} \left(A(o) \times \tau(d,o) \right) \right) \tag{2.23}$$

où la préférence d'utilisation u et la fonction de qualité $A : O \rightarrow L$ sont relatifs à l'organe courant d.

Nous avons précédemment défini que la solution offerte par ORQA au conducteur devait satisfaire certaines contraintes sur ses résultats. Afin de sélectionner la meilleure solution possible y répondant, les stratégies de conduites sont classées par un score reposant sur leurs résultats.

2.3.4 Le score d'une stratégie de conduite

Le score d'une stratégie de conduite permet une comparaison objective des résultats d'une solution par rapport aux autres solutions possibles. Dans un premier temps, les résultats (une durée, une consommation, une qualité de service) sont normalisés. Les valeurs normalisées sont assemblées pour établir les scores des stratégies. La meilleure stratégie est celle possédant un score maximal, elle est sélectionnée comme solution par ORQA.

2.3.4.1 Les fonctions de score

Une fonction de score normalise un résultat (une durée, une consommation ou une qualité de service) par rapport à toutes les stratégies de conduite évaluées, les capacités du véhicule et les contraintes fixées par le conducteur. En fonction du résultat qu'elle évalue, une fonction de score renvoie le meilleur score pour un résultat minimal (durée et consommation) ou maximal (qualité). Un résultat est considéré disqualifiant s'il ne satisfait pas aux contraintes du conducteur ou s'il n'est pas réalisable en l'état (c.-à-d. le véhicule ne dispose pas d'assez d'énergie). Par exemple, si le conducteur a défini une durée maximale $\chi_T = 1{,}5$, toute route r au rapport $\frac{T_r}{T_{\bar{r}}}$ supérieur à 1,5 doit être disqualifiée.

Nous cherchons deux fonctions différentes, une promouvant les valeurs minimales $f_{\min}$ et une autre les valeurs maximales $f_{\max}$. Nous posons les définitions suivantes : $S^{\min}$ et $S^{\max}$ définissent le domaine valide des scores, un score inférieur à $S^{\min}$ est considéré disqualifiant pour la stratégie de conduite concernée. Nous obtenons le domaine X d'un résultat (T le domaine des durées, E le domaine des consommations et Q le domaine des qualités de service) à partir de l'ensemble des

solutions possibles. Les valeurs $X^{\min}$ et $X^{\max}$ sont les valeurs limites autorisées trouvées pour chaque résultat telles que :

$$X^{\min} = \inf_{x \in X_\lambda} x \quad \forall \lambda \in \Lambda \tag{2.24}$$

$$X^{\max} = \sup_{x \in X_\lambda} x \quad \forall \lambda \in \Lambda \tag{2.25}$$

où X_λ est le domaine acceptable d'un résultat pour une stratégie λ. De plus, d'après les contraintes énoncées précédemment par l'équation (2.17) et l'équation (2.18), les sélections des valeurs limites des résultats sont telles que :

$$T^{\max} \leq \chi_T \times T_\lambda^{\text{nominale}} \quad \forall \lambda \in \Lambda \tag{2.26}$$

$$E^{\max} \leq \chi_E \times E_{\text{capacité}} \tag{2.27}$$

$$E^{\max} \leq E_{\text{disponible}} \tag{2.28}$$

Nous appliquons ces contraintes pour la durée et la consommation à $f_{\min}$ car nous cherchons à promouvoir leurs valeurs minimales. Nous énumérons maintenant les conditions que $f_{\min}$ doit satisfaire :

$$f_{\min}(X^{\min}) = S^{\max} \tag{2.29}$$

$$f_{\min}(X^{\max}) = S^{\min} \tag{2.30}$$

$$f_{\min} \text{ est monotone sur } [X^{\min}; X^{\max}] \tag{2.31}$$

$$f_{\min}(x) < S^{\min} \quad \forall x > X^{\max} \tag{2.32}$$

Comme nous cherchons à promouvoir le minimum d'un domaine, il n'y a pas de valeur plus petite que $X^{\min}$ que l'on souhaite promouvoir, elle obtient donc le score maximal $S^{\max}$ (2.29). En contrepartie, la valeur maximale autorisée $X^{\max}$ produit le score minimal $S^{\min}$ (2.30). La fonction de score doit produire un score proportionnel aux valeurs, du moment qu'elles appartiennent au domaine autorisé. C'est à dire que, quand les valeurs augmentent, le score doit diminuer (2.31). Une valeur disqualifiante, c'est à dire supérieure à $X^{\max}$ car ne répondant pas aux contraintes précédentes (2.26), (2.27) et (2.28), doit produire un score disqualifiant (2.32).

De manière équivalente, les conditions que $f_{\max}$ doit satisfaire pour les résultats de qualité de service sont :

$$f_{\max}(X^{\min}) = S^{\min} \tag{2.33}$$

$$f_{\max}(X^{\max}) = S^{\max} \tag{2.34}$$

$$f_{\max} \text{ est monotone sur } [X^{\min}; X^{\max}] \tag{2.35}$$

Nous suivons le même raisonnement que pour $f_{\min}$ en transposant le minimum au maximum. Ce qui signifie que la valeur maximale $X^{\max}$ produit le score maximal

$S^{\max}$ (2.33). La valeur minimale $X^{\min}$ obtient le score minimal $S^{\min}$ (2.34). La fonction $f_{\max}$ est également monotone (2.35). En revanche, comme le domaine des qualités de service est fixé à $[0;1]$ et qu'aucune contrainte ne vient s'y ajouter, il n'y a pas de condition disqualifiante définie pour $f_{\max}$. Ceci implique que le domaine produit par $f_{\max}$ est borné par $f_{\max}(X^{\min})$ et $f_{\max}(X^{\max})$. C'est à dire que le domaine de sortie de $f_{\max}$ est $[S^{\min}; S^{\max}]$.

Nous définissons $f_{\min}$ et $f_{\max}$ de la manière suivante pour répondre aux conditions que nous venons d'énoncer :

$$f_{\min}(x) = S^{\max} + \frac{(S^{\max} - S^{\min}) \times (X^{\min} - x)}{X^{\max} - X^{\min}} \tag{2.36}$$

$$\begin{aligned} f_{\max}(x) &= (S^{\max} + S^{\min}) - f_{\min}(x) \\ &= S^{\min} - \frac{(S^{\max} - S^{\min}) \times (X^{\min} - x)}{X^{\max} - X^{\min}} \end{aligned} \tag{2.37}$$

Les évolutions de $f_{\min}$ et de $f_{\max}$ sont illustrées par la figure 2.5. Enfin, les fonctions de score des résultats découlent directement des fonctions $f_{\min}$ (minimisation de la durée et de la consommation) et $f_{\max}$ (maximisation de la qualité de service). En choisissant un domaine de score de $[1;100]$, les fonctions de score de durée S_T, de consommation S_E et de qualité de service S_Q sont :

$$S_T(t) = 100 + \frac{99 \cdot (T^{\min} - t)}{T^{\max} - T^{\min}} \tag{2.38}$$

$$S_E(e) = 100 + \frac{99 \cdot (E^{\min} - e)}{E^{\max} - E^{\min}} \tag{2.39}$$

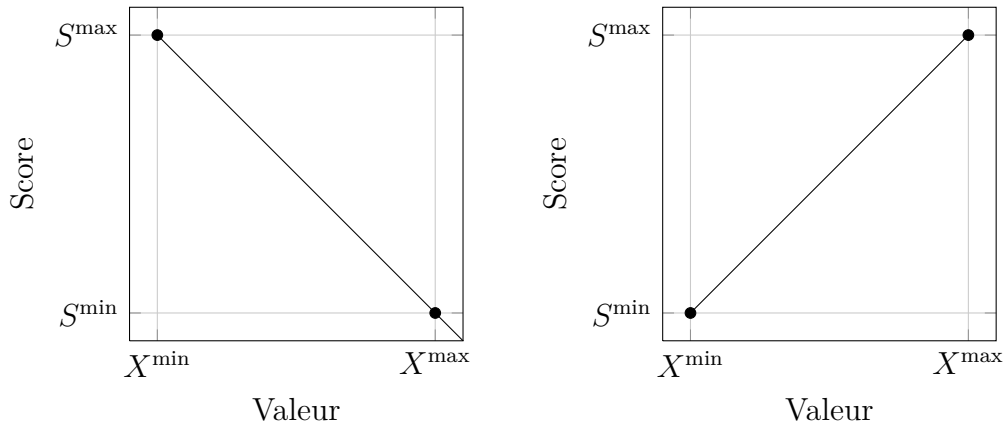
$$S_Q(q) = 1 - \frac{99 \cdot (Q^{\min} - q)}{Q^{\max} - Q^{\min}} \tag{2.40}$$

2.3.4.2 La combinaison des fonctions de score

Les valeurs renvoyées par les fonctions de score sont combinées par une moyenne arithmétique pondérée et forment le score de la stratégie de conduite impliquée. Les poids de la moyenne sont ceux de la politique de consommation définie par le conducteur. Nous définissons donc le score S d'une stratégie par :

$$\begin{cases} S = S_T \times w_T + S_E \times w_E + S_Q \times w_Q \\ S = 0 & \exists S_i \notin [S^{\min}; S^{\max}] \end{cases} \tag{2.41}$$

Un score est nul si au moins un de ses résultats est disqualifiant, c'est-à-dire s'il ne satisfait pas les contraintes. Comme les fonctions de score renvoient des valeurs dans le domaine $[1;100]$ et que les poids sont dans le domaine $[0;1]$, les



(a) $f_{\min}$: promotion de la valeur minimale (b) $f_{\max}$: promotion de la valeur maximale

FIGURE 2.5 – Fonctions de score promouvant (a) la valeur minimale du domaine et (b) la valeur maximale du domaine. Les valeurs disqualifiantes passées à $f_{\min}$ (c.-à-d. celles supérieures à la limite maximale imposée $X^{\max}$) produisent un score inférieur au score minimum.

scores appartiennent au domaine $[0; 100]$. Il est alors possible de comparer les différentes stratégies de conduite par leur score qui reflète la concordance d'une solution avec les souhaits du conducteur. La stratégie dont le score est le plus élevé est sélectionnée par le canevas logiciel comme solution au trajet.

2.4 La configuration du gestionnaire d'énergie à la conception

La recherche de la meilleure solution réalisée par le gestionnaire d'énergie correspond à calculer pour chaque stratégie de conduite les résultats du trajet. Pour prendre en considération les potentielles limites de la plate-forme d'implémentation, le canevas logiciel peut être configuré de sorte que son empreinte soit diminuée dans la durée de calcul et dans la quantité de mémoire à allouer. Il n'est peut être pas nécessaire d'effectuer une exploration exhaustive si les capacités du véhicule tendent toujours vers certaines solutions. De son côté, l'application de la solution pendant le trajet est une tâche simple qui consiste à vérifier que la consommation ne dépasse pas ce qui et prévu par la solution choisie, il faut sinon re-lancer le processus de recherche pour une nouvelle solution à jour. En conséquence, nous décidons de nous focaliser sur l'exploration de l'espace de solution pour diminuer l'empreinte d'ORQA en ligne.

Nous avons deux pistes possibles pour accélérer l'exploration :

1. réduire l'espace de solution ;
2. simplifier le calcul des solutions.

Les solutions possibles sont les évaluations des stratégies de conduites, qui sont elles-mêmes le résultat du produit cartésien des routes (nominales et variations) et des combinaisons d'organes (2.21). Nos deux pistes reviennent donc à 1) diminuer le nombre de stratégies de conduite et à 2) simplifier leur évaluation. Les combinaisons d'organes sont connues à la conception mais leurs qualités de service dépendent du conducteur et de l'ordre qu'il a choisi pour les organes contrôlables (2.23). Il n'est donc pas possible d'influer sur ce paramètre qualitatif à la conception. La simplification de l'exploration porte donc sur les routes, c'est à dire sur la réduction des coefficients de vitesse à investiguer.

La simplification de l'exploration des solutions se base sur l'étude de routes jugées « typiques ». Les concepteurs savent à quoi est destiné le véhicule, s'il est spécialisé (p. ex. parc privé) ou généraliste. En conséquence, il leur est possible de définir des routes types pour lesquelles étudier l'influence des configurations sur l'exploration. Nous pouvons ici faire un parallèle avec la consommation moyenne des véhicules vendus dans le commerce, évaluée sur les routes « types ».

Comme l'influence des combinaisons d'organes ne peut être étudiée sans préférences utilisateur, les stratégies de conduite utilisent une même combinaison d'organes et leurs qualités de service ne sont pas prises en compte. La combinaison influençant le moins les résultats est celle qui propose la consommation la plus basse possible. C'est typiquement la combinaison d'organes où les organes contrôlables sont éteints.

La figure 2.6 illustre les ratios des résultats de durée et de consommation d'un véhicule généraliste sur un ensemble de vingt-cinq routes générées. Les ratios sont calculés par rapport aux résultats nominaux afin d'avoir une comparaison possible entre les différentes routes considérées. Les ratios de durée et de consommation pour un coefficient de vitesse k sont définis par :

$$\forall r \in R \quad \begin{cases} ratio_k^T(r) & = \dfrac{T_r}{T_{\bar{r}}} \\ ratio_k^E(r) & = \dfrac{E_r}{E_{\bar{r}}} \end{cases} \tag{2.42}$$

Nous pouvons constater une similitude certaine des résultats, nous en tirons l'idée de réduction du nombre de solutions par l'hypothèse que plusieurs solutions aux résultats similaires peuvent être représentées par une seule solution (piste 1). Nous constatons également une évolution semblable des résultats indépendante de la route, c'est à dire que l'évolution des résultats dépend principalement des capacités du véhicule. D'où la seconde piste de simplification de l'évaluation par

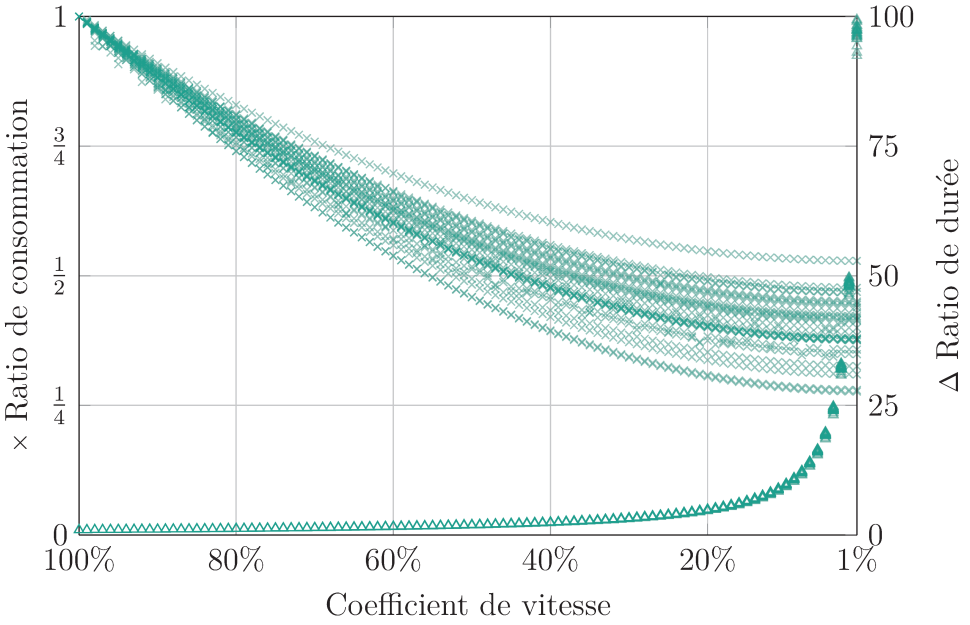


FIGURE 2.6 – Évolution des ratios des résultats de durée (triangles) et de consommation (croix).

un rapprochement de l'évolution des résultats pour des routes différentes. Ces deux pistes ne sont pas antagonistes et peuvent être complémentaires. Il est en effet possible d'évaluer de manière plus simple un nombre réduit de stratégies de conduite, le problème étant d'avoir des solutions de qualité. La clef d'une bonne configuration du gestionnaire d'énergie est de fournir des solutions proches (voire identiques) aux solutions optimales tout en étant contraint par son déploiement sur la plate-forme du véhicule. Nous présentons maintenant chacune de ces deux pistes.

2.4.1 Réduction de l'espace de solution par le partitionnement des coefficients de vitesse

Nous décrivons maintenant la première piste de simplification de la recherche de solution qui propose de regrouper des solutions semblables et ainsi de diminuer l'espace de solution à explorer. Nous avons défini que l'espace de solution était composé des stratégies de conduites évaluées et que ces stratégies étaient le résultat du produit cartésien des routes et des combinaisons d'organes. Les routes sont formées à partir des chemins possibles trouvés par le système auxquels ont été appliqué les coefficients de vitesse. La réduction de l'espace de solution implique donc une réduction des chemins possibles et des coefficients de vitesse. Il n'est

pas possible de déterminer à l'avance si les chemins possibles seront similaires ou pas. En revanche, les coefficients de vitesse sont définis dans l'intervalle [1%; 100%] avec un pas de discrétisation fixé à 1, soit cent valeurs. La réduction de l'espace de solution est la définition d'un nombre de coefficients de vitesse restreint.

Regrouper des résultats revient à former des groupes de valeurs proches, c'est à dire réaliser un *partitionnement* des coefficients de vitesse. La finalité du partitionnement est de représenter un groupe de coefficients de vitesse par un seul dans la configuration. Les groupes de résultats doivent permettre d'agglomérer des coefficients de vitesse qui produisent des résultats similaires. Dans ORQA, nous utilisons un partitionnement de type *k-médoïdes*. Cet algorithme permet de définir les groupes, chacun représenté par un vecteur lui appartenant, minimisant pour chaque groupe l'écart relatif entre le vecteur représentatif et les autres vecteurs du groupe. Chaque vecteur représente un coefficient de vitesse et est constitué de l'ensemble des ratios des routes pour ce coefficient. Les durées et les consommations sont deux résultats distincts qui évoluent également de manières distinctes. Un vecteur V_k représentant un coefficient de vitesse k est défini par :

$$V_k = \langle ratio_k^T(r_1), ratio_k^E(r_1), ..., ratio_k^T(r_n), ratio_k^E(r_n) \rangle \tag{2.43}$$

où $ratio_k^T$ et $ratio_k^E$ sont respectivement les ratios de durée et de consommation d'une route pour un coefficient k. Les n routes évaluées sont celles considérées dans le partitionnement. Un partitionnement résulte en un ensemble de k groupes de vecteurs. k est compris entre un et cent (c.-à-d. le nombre de coefficients), un groupe pour regrouper tous les coefficients et cent groupes pour séparer chaque coefficient. La figure 2.7 applique par exemple l'algorithme de partitionnement aux ratios de durée et de consommation. Les cinq partitions demandées sont identifiées par des couleurs différentes. Chacune d'elle est associée à un coefficient de vitesse ; dans notre cas, ce sont 2%, 15%, 49%, 72% et 92%.

Pour déterminer le meilleur nombre de partitions (c.-à-d. le nombre de coefficients de vitesse), le partitionnement est effectué avec les 100 valeurs possibles de k. La comparaison des 100 partitionnements possibles conduit à la sélection du partitionnement maximisant les similarités. Nous nous basons sur des méthodes classiquement utilisées pour déterminer le nombre adéquat de partitions telles que la *méthode du coude* ou la *méthode de la courbure*. Ces deux méthodes étudient l'évolution de l'erreur entre les vecteurs (leur distances) à l'intérieur des groupes. Naïvement, un partitionnement est considéré digne d'intérêt lorsqu'il n'est pas intéressant d'augmenter le nombre de partitions par rapport à la réduction d'erreur apportée. Cette erreur est la somme des carrés des erreurs (*sum of squared errors*, SSE) telle que définie par :

$$SSE = \sum_i (V_i - \overline{V_i})^2 \tag{2.44}$$

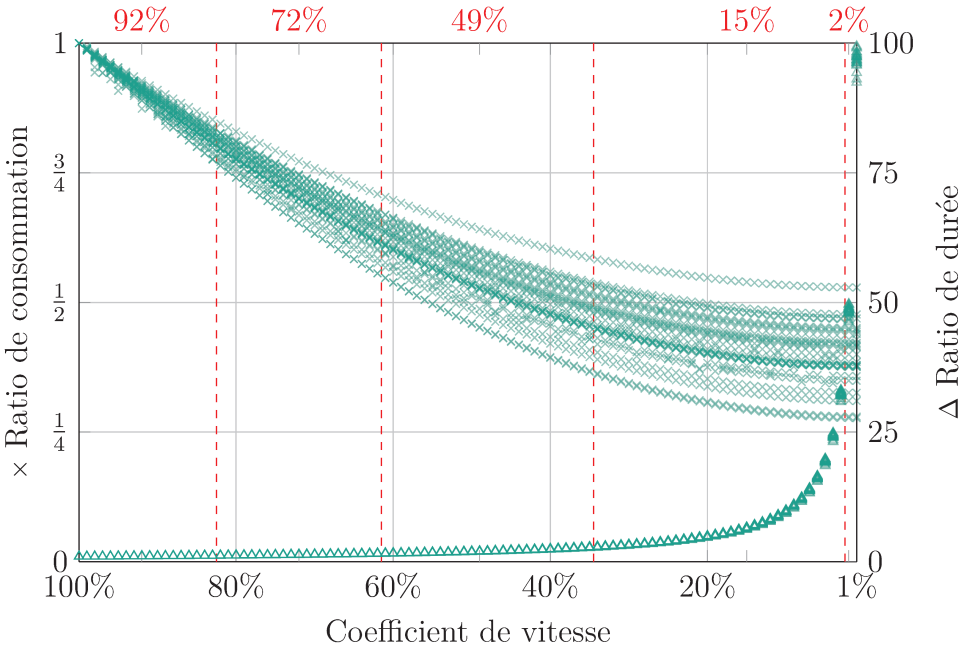


FIGURE 2.7 – Exemple de partitionnement des ratios des résultats de durée et de consommation.

où $\overline{V_i}$ est le vecteur représentatif associé à V_i.

La méthode du coude consiste à dessiner l'évolution de la SSE selon le nombre de partitions. Lorsqu'un coude est visible (c.-à-d. un angle net dans la pente formant un coude), le point à l'extérieur du coude indique un nombre intéressant de partitions. La méthode de la courbure est l'étude de la dérivée seconde de la SSE. Son évolution forme des pics aux points d'intérêts, c'est à dire que les points d'intérêts sont mis en avant.

Ces deux méthodes sont appliquées sur les SSE des partitionnements possibles et permettent de proposer des partitionnements pertinents. C'est à dire des domaines de coefficients de vitesse réduits tels qu'ils représentent le mieux l'ensemble des coefficients de vitesse pour un véhicule donné.

2.4.2 Simplification de l'évaluation des stratégies de conduite par l'approximation des résultats

La deuxième piste pour accélérer l'exploration de l'espace de solution est la simplification de l'évaluation des stratégies de conduite. Elle est basée sur le fait que les ratios des évaluations évoluent de manière semblable (cf. figure 2.6) selon les coefficients de vitesse, et ce pour les deux résultats de durée et de consom-

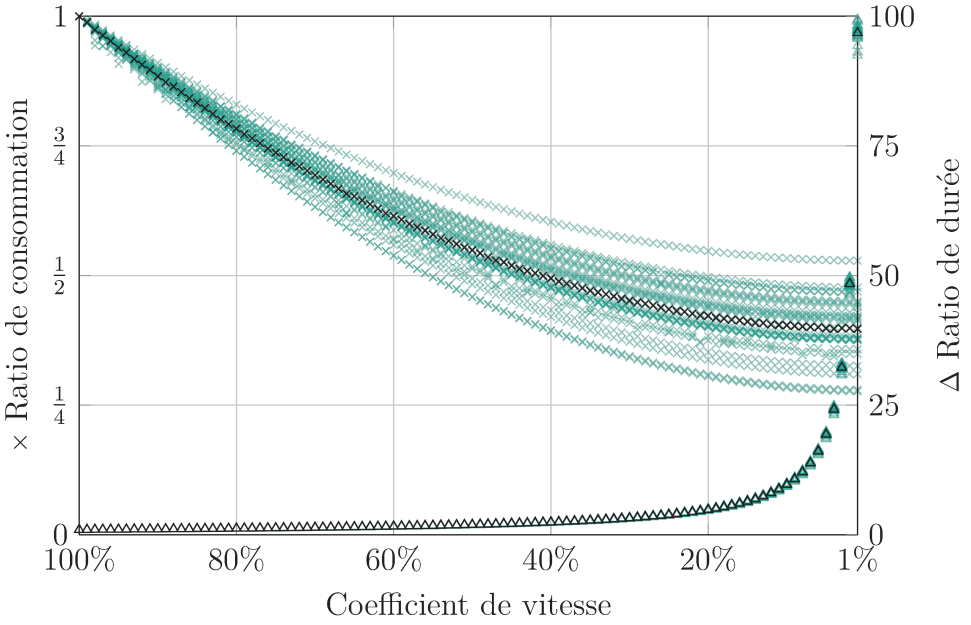


FIGURE 2.8 – Exemple de simplification des ratios des résultats de durée et de consommation ; les ratios approximés sont en noir.

mation. Intuitivement, nous constatons que les courbes des ratios se ressemblent et qu'il serait possible de les approximer avec une erreur faible. Dans le domaine des statistiques, les méthodes de régression permettent d'analyser la relation d'une variable par rapport aux autres. Traduit dans notre contexte, cela correspond à analyser la relation d'un ratio d'une évaluation par rapport aux ratios des autres évaluations. Cette analyse est reconduite pour chaque coefficient de vitesse est mène à une fonction de régression. En conséquence, nous proposons de simplifier l'évaluation de stratégies de conduites en approximant leurs résultats plutôt que de les évaluer systématiquement. Cette approximation est telle que :

$$\overline{ratio_k^X} = \frac{1}{n} \sum_{i=1}^{n} ratio_k^X(r_i) \tag{2.45}$$

où $\overline{ratio_k^X}$ est le ratio approximé de tous les $ratio_k^X(r_i)$ et X peut être T ou E. La projection de ces ratios approximés sur les résultats présentés en exemple précédemment est illustrée par la figure 2.8.

Il est ainsi possible d'approximer les ratios des résultats. Étant donné que tous les ratios sont relatifs à l'évaluation nominale (coefficient de vitesse à 100%), nous supposons que cette dernière est calculée réellement. À partir de ses résultats, les ratios approximés sont appliqués et permettent d'obtenir les résultats correspondant aux autres coefficients de vitesse.

TABLE 2.1 – Extrait des approximations de ratios de durée et de consommation.

Coefficient de vitesse	Durée ratio moyen	écart type	Consommation ratio moyen	écart type
100%	1,00	-	1.00	-
90%	1,10	±0,00 (0%)	0,88	±0,01 (1%)
80%	1,24	±0,01 (1%)	0,78	±0,02 (3%)
60%	1,63	±0,02 (1%)	0,61	±0,04 (7%)
40%	2,43	±0,03 (1%)	0,49	±0,05 (10%)
20%	4,85	±0,08 (2%)	0,42	±0,06 (14%)
1%	96,8	±1,66 (2%)	0,40	±0,06 (15%)

Pour chaque coefficient de vitesse, les ratios des évaluations sont représentés par un ratio approximatif. Les résultats de durée et de consommation évoluant différemment, leurs ratios sont approximés de manières indépendantes ($ratio^T$ et $ratio^E$).

Pour permettre aux concepteurs de déterminer l'intérêt de cette approche, les taux d'erreur induits par les ratios sont calculés. C'est à dire qu'un ratio ayant un taux d'erreur de 20% induira, en moyenne, une erreur de 20% sur les résultats qu'il approxime. Le taux d'erreur d'un ratio approximé introduit la fiabilité d'une approximation. Il est défini par l'écart relatif de l'approximation aux valeurs réelles :

$$erreur_k^X = \sqrt{\sum_{i=1}^{n} \frac{((ratio_k^X(r_i) - \overline{ratio_k^X})/ratio_k^X(r_i))^2}{n}} \tag{2.46}$$

Au final, les concepteurs peuvent constater de la fiabilité d'un ratio et décident en conséquence de l'utiliser ou pas. Le tableau 2.1 présente un extrait des ratios approximés de durée et de consommation illustrés par la figure 2.8 ainsi que leurs écarts type.

2.4.3 Composition des deux approches d'optimisation

Les deux approches d'optimisation peuvent être composées et proposer deux nouvelles configurations, en fonction de l'ordre de composition. Comme l'approche de partitionnement se base sur les similarités des résultats et que l'approche d'approximation les regroupe par coefficients de vitesse, l'ordre de composition devrait peu influencer sur la configuration finale. Néanmoins, cette intuition n'est pas prouvée de manière formelle et ORQA propose donc les deux possibilités.

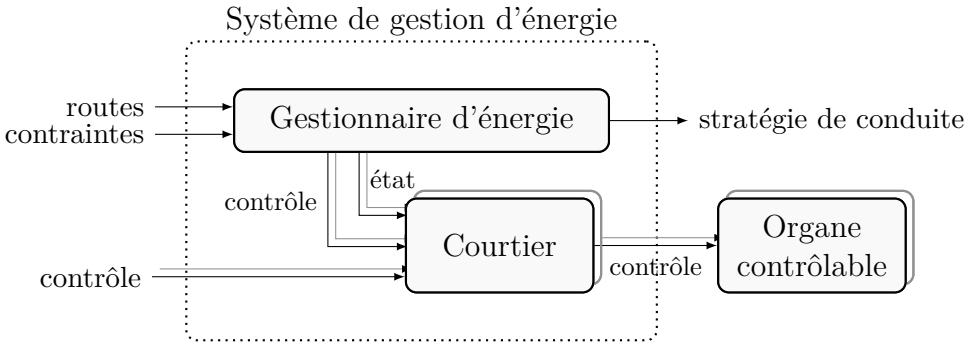


FIGURE 2.9 – Architecture logicielle du système de gestion d'énergie.

Approche de partitionnement puis approche d'approximation. Cet ordre de composition correspond à l'extraction d'une partie des résultats de la base de données, ceux dont le coefficient de vitesse fait partie des coefficients représentatifs. L'approche d'approximation est ensuite appliquée de la même manière que défini précédemment sur ce sous-ensemble de résultats.

Approche d'approximation puis approche de partitionnement. Dans cet ordre de composition, les ratios moyens obtenus par l'approche d'approximation sont directement passés à l'approche de partitionnement. Cette dernière procède de la même manière que précédemment, mais avec un résultat obtenu beaucoup plus rapidement. En effet, le partitionnement est effectué sur $2n$ dimensions en parallèle (n étant le nombre de routes étudiées) car les ratios de durée et de consommation sont étudiés séparément pour toutes les routes étudiées. Ici, n vaut 1 et il n'y a donc plus que l'évolution d'un ratio de durée et d'un ratio de consommation à étudier, ceux issus de l'approximation.

2.5 L'architecture logicielle du système de gestion d'énergie

Nous présentons maintenant l'architecture logicielle du système de gestion d'énergie (cf. figure 2.9). Nous regroupons sous le terme *système de gestion d'énergie* les différents composants logiciels qui permettent effectivement la gestion d'énergie au sein du véhicule. Le rôle du gestionnaire d'énergie est de sélectionner la meilleure stratégie de conduite en fonction des routes possibles jusqu'à destination et des contraintes fixées par le conducteur. Il assure ensuite la réalisation de cette stratégie en contrôlant les organes *via* des courtiers logiciels.

2.5.1 Le gestionnaire d'énergie

Le gestionnaire d'énergie réalise deux opérations : 1) il recherche la meilleure solution selon la requête du conducteur et 2) il assure l'application de cette solution pendant le trajet. La recherche de la meilleure solution est l'exploration de l'ensemble des stratégies de conduite évaluées à partir de la requête du conducteur (les routes à destination ainsi que les contraintes optionnelles, cf. section 2.3). Les coefficients de vitesse et les combinaisons d'organes sont des données définies à la conception, elles font partie des connaissances du gestionnaire d'énergie. L'application de la solution est le contrôle des organes contrôlables pendant le trajet. C'est-à-dire qu'après avoir sélectionné une stratégie de conduite optimale, il débute la gestion du véhicule et de ses organes. Son fonctionnement est décrit par l'algorithme 2.1. Nous y retrouvons la première opération des lignes 2 à 12 et la seconde des lignes 14 à 16. De plus, le gestionnaire d'énergie exporte la stratégie de conduite sélectionnée à la ligne 13.

Algorithme 2.1 Fonctionnement du gestionnaire d'énergie.

```
1: procédure GestionnaireEnergie(routes, contraintes)
2:     pour chaque k_v ∈ coefficients de vitesse
3:         pour chaque r ∈ routes
4:             pour chaque g ∈ combinaisons d'organes
5:                 évaluer la stratégie de conduite formée par ⟨r,k_v,g⟩
6:             fin pour chaque
7:         fin pour chaque
8:     fin pour chaque
9:     pour chaque λ ∈ stratégies de conduite
10:        calculer le score de la stratégie λ en considérant les contraintes
11:    fin pour chaque
12:    sélectionner la stratégie de conduite dont le score est maximal pour définir
   solution
13:    informer le système (et le conducteur) de la stratégie de conduite solution
14:    pour chaque d ∈ organes contrôlables
15:        commander au courtier de d d'appliquer le mode opératoire de d défini
   dans solution
16:    fin pour chaque
17: fin procédure
```

Le gestionnaire d'énergie utilise les courtiers logiciels comme relais pour agir sur le contrôle des organes. La stratégie de conduite comporte une combinaison d'organes. C'est elle que le gestionnaire d'énergie décompose pour commander les courtiers des organes contrôlables.

2.5.2 Les courtiers des organes contrôlables

Un courtier logiciel est créé pour chacun des organes contrôlables. Son rôle est de modifier le contrôle de l'organe auquel il est associé lorsque le gestionnaire d'énergie est actif, c.-à-d. lorsqu'il gère le véhicule. Ainsi, nous distinguons deux états d'un courtier directement lié au comportement du gestionnaire d'énergie : *actif* et *inactif*. Un courtier prend place au côté d'un organe contrôlable, il remplace la commande de contrôle originale adressée à l'organe. Cette commande est celle qui existe dans le système logiciel du véhicule avant la présence du système de gestion d'énergie.

Le comportement d'un courtier est générique, il est décrit par l'algorithme 2.2. En fonction de l'état du courtier défini par le gestionnaire d'énergie, il contrôle son organe associé par une certaine commande. S'il est *actif*, alors la commande du gestionnaire d'énergie est récupérée (ligne 3), sinon c'est la commande originale provenant du système qui est récupérée (ligne 5). La commande ainsi définie est transmise à l'organe associé au courtier.

Algorithme 2.2 Fonctionnement d'un courtier logiciel.

```
1: fonction COURTIER(état, commande originale, commande GE) retourne commande
2:     si état = actif alors
3:         assigner la commande provenant du gestionnaire d'énergie commande GE à commande
4:     sinon
5:         assigner la commande originale commande originale à commande
6:     fin si
7: fin fonction
```

2.6 Conclusion

Ce chapitre a présenté ORQA, un canevas logiciel pour la mise en place d'un gestionnaire d'énergie pour les véhicules électriques. Pour assurer une bonne gestion des organes du véhicule, le gestionnaire d'énergie requiert les connaissances de besoins énergétiques et de qualité liées aux organes. Ces propriétés extra-fonctionnelles n'étant pas représentées dans les modèles existants, le canevas logiciel en propose une modélisation spécifique.

La modélisation des systèmes embarqués comporte les fonctionnalités logicielles du véhicule. Comme ORQA s'intéresse principalement à l'amélioration de l'usage de l'énergie au sein du véhicule, il prend en considération tous les organes à com-

mandes électriques ou électroniques. De fait, ces organes offrent les services permettant la réalisation des fonctionnalités proposées par le véhicule. Le comportement d'un organe est l'évolution de son mode opératoire, c'est à dire de la façon par laquelle il opère. Chaque mode opératoire est complété par son besoin énergétique modélisé sous la forme d'une expression mathématique. Certains organes sont contrôlés à l'exécution pour assurer la réussite du trajet. ORQA considère la qualité de service offerte par ces organes, elle sert à décider lesquels utiliser en priorité. En conséquence, les organes contrôlables ont une caractérisation qualitative au niveau de leurs modes opératoires mais aussi entre eux pour signifier la préférence d'utilisation du conducteur.

Ces connaissances permettent la génération du gestionnaire d'énergie à embarquer dans le véhicule. Ce gestionnaire répond à une requête du conducteur pour atteindre une destination. Le conducteur spécifie une politique de consommation et éventuellement des contraintes de durée et de consommation. Le gestionnaire d'énergie explore l'espace de solution composé des routes et des combinaisons d'organes. Chaque route accessible est raffinée à l'aide de coefficients de vitesse pour représenter différentes conditions de conduite. Les organes pouvant fournir leur services dans plusieurs modes opératoires, ils sont combinés un à un pour correspondre aux différentes combinaisons. Chaque solution potentielle est évaluée par ses résultats de durée, de consommation énergétique et de qualité de service. Grâce à la politique de consommation indiquée par le conducteur, ces résultats sont pondérés et mènent aux scores des solutions. La solution au score le plus élevé est sélectionnée comme la stratégie de conduite à appliquer par ORQA.

En considérant une exécution en ligne dans un véhicule, il est possible que le surcoût induit par le gestionnaire d'énergie soit trop important. Nous cherchons alors à optimiser sa complexité par le biais d'une configuration optionnelle. Grâce à une étude des capacités du véhicule (et de ses organes) sur un ensemble de routes représentatives, ORQA propose une configuration adéquate du gestionnaire d'énergie. La configuration repose sur deux principes : 1) la réduction de l'espace de solution et 2) la simplification de l'évaluation des solutions potentielles. Ces deux principes sont présentés par deux approches cherchant le meilleur compromis entre simplification et erreur induite. Enfin, elles permettent de présenter aux concepteurs plusieurs configurations intéressantes en adéquation avec les capacités du véhicule.

Le système de gestion d'énergie est composé du gestionnaire d'énergie et de ses courtiers logiciels. Le composant gestionnaire d'énergie réalise tout d'abord la recherche de la solution optimale puis assure la réalisation du trajet. Par le biais des courtiers, il applique en premier lieu la combinaison d'organes contenue dans la stratégie de conduite sélectionnée. Les courtiers agissent en tant que relais au niveau des organes contrôlables auxquels ils sont associés. Ensuite, et tout au

long du trajet, le gestionnaire d'énergie assure la concordance entre les résultats attendus pour la solution et ceux effectifs. Si ces derniers ne sont pas conformes, une nouvelle recherche est lancée dont la solution remplace alors la stratégie de conduite en place.

Chapitre 3

Mise en œuvre

Sommaire

Nous avons présenté les concepts du canevas logiciel ORQA, sa méthodologie et son architecture. Nous décrivons maintenant sa mise en œuvre.

En premier lieu, nous revenons sur la modélisation des connaissances énergétiques et qualitatives des organes consommateurs. Le métamodèle de cette caractérisation est réalisé en *Ecore* et, pour faciliter la manipulation de ses modèles, nous introduisons un langage de modélisation dédié textuel réalisé avec *Xtext*. Nous détaillons ensuite la bibliothèque de modèles inclus dans ORQA contenant des exemples de modélisation des principaux organes consommateurs.

Pour mettre en œuvre le processus d'ORQA, l'architecture logicielle du système cible est augmentée du système de gestion d'énergie. Ceci débute par l'extraction des organes du modèle initial pour initialiser le modèle de caractérisation. L'étape suivante est la mise en place du système de gestion d'énergie d'ORQA dans l'architecture cible puis la génération de son implémentation.

Enfin, nous abordons l'implémentation des algorithmes d'optimisation de la configuration du gestionnaire d'énergie. Ils sont réalisés par un ensemble de programmes écrits en Python.

3.1 Caractérisation des organes du véhicule électrique

Les principes généraux du métamodèle de caractérisation des organes consommateurs ont été introduits dans la section 2.2. Dans cette section, nous détaillons tout d'abord l'implémentation du métamodèle. Nous introduisons également un langage de modélisation dédié textuel facilitant la manipulation – création, édition, ré-utilisation – des modèles par les concepteurs. Nous abordons ensuite les modèles des organes proposés par ORQA dans le cadre d'une bibliothèque accessible aux concepteurs.

3.1.1 Métamodèle et langage dédié

La caractérisation énergétique et qualitative des organes consommateurs du véhicule est réalisé dans un modèle dédié. Les concepts principaux du métamodèle correspondant ont été illustrés par la figure 2.2. Le métamodèle complet est lui-même représenté dans un modèle *Ecore* grâce à EMF *(Eclipse Modeling Framework)* [Eclb], le canevas logiciel de (méta)modélisation du projet Eclipse [Ecla]. Nous détaillons dans un premier temps le contenu du métamodèle de caractérisation des organes dans le modèle *Ecore*. Afin de faciliter la manipulation des modèles de caractérisation, nous définissons un langage de modélisation dédié (*Domain Specific Modelling Language*, DSML). Vis à vis des concepteurs, ce langage

est une syntaxe concrète pour le métamodèle. Il est ainsi possible de créer un modèle textuel de caractérisation en utilisant des mots-clefs du domaine. Ce langage est réalisé grâce à *Xtext* [Eclc].

3.1.1.1 Le métamodèle de caractérisation des organes consommateurs en *Ecore*

EMF est un canevas logiciel de modélisation et de génération de code issu du projet Eclipse. Sa base est le métamodèle *Ecore* offrant une modélisation basée sur le paradigme objet, proche du langage de programmation Java dans lequel il est réalisé. La représentation utilisée suit celle des diagrammes de classes UML. D'autres projets reposent sur EMF pour fournir un outillage couplé à un métamodèle défini avec *Ecore*. Citons par exemple l'outil de génération de classes et d'interfaces Java reflétant le métamodèle ou encore la génération d'éditeurs graphiques de modèle. La figure 3.1 représente le métamodèle de caractérisation en *Ecore* tel que vu depuis l'éditeur graphique intégré. Nous détaillons maintenant les spécificités introduites dans ce métamodèle.

Le système (*System*) représente le véhicule cible et contient les organes consommateurs (*Device*). Pour caractériser ces organes, nous nous basons sur des expressions mathématiques (cf. section 2.2). La manipulation et la réutilisation de ces expressions est simplifiée par l'utilisation de variables *(Variable)* et de fonctions *(Function)* définies au sein d'un système. Nous regroupons finalement les éléments appartenant à un système (organes, variables et fonctions) sous un concept commun d'élément nommé (c.-à-d. les classes *Device*, *Variable* et *Function* héritent de *NamedElement*).

Nous avons défini un organe comme un composant, il peut donc être utilisé, par définition, dans différents véhicules. Réciproquement, une gamme de véhicules peut utiliser différents organes pour réaliser un même service. Au niveau du modèle de caractérisation, le paradigme des composants est utilisé en offrant la possibilité d'importer d'autres systèmes dans le système courant. Typiquement, un organe est modélisé séparément et est ensuite importé dans les systèmes des véhicules. Cette notion est modélisée par des objets de la classe *Import* appartenant au système. L'idée sous-jacente est que les nouveaux véhicules réutilisent des organes existants, ce qui correspond ici à importer des modèles déjà établis.

Les variables et les fonctions peuvent être externes ou locales. Elles sont définies en suivant le paradigme fonctionnel classique. Une variable externe (réciproquement une fonction externe) fait référence à une donnée (une opération) accessible dans le système cible. Une variable locale est une constante définie dans un système, elle n'est pas accessible depuis l'environnement logiciel (c.-à-d. non exportée par le gestionnaire d'énergie). Ceci permet la factorisation des expressions constantes

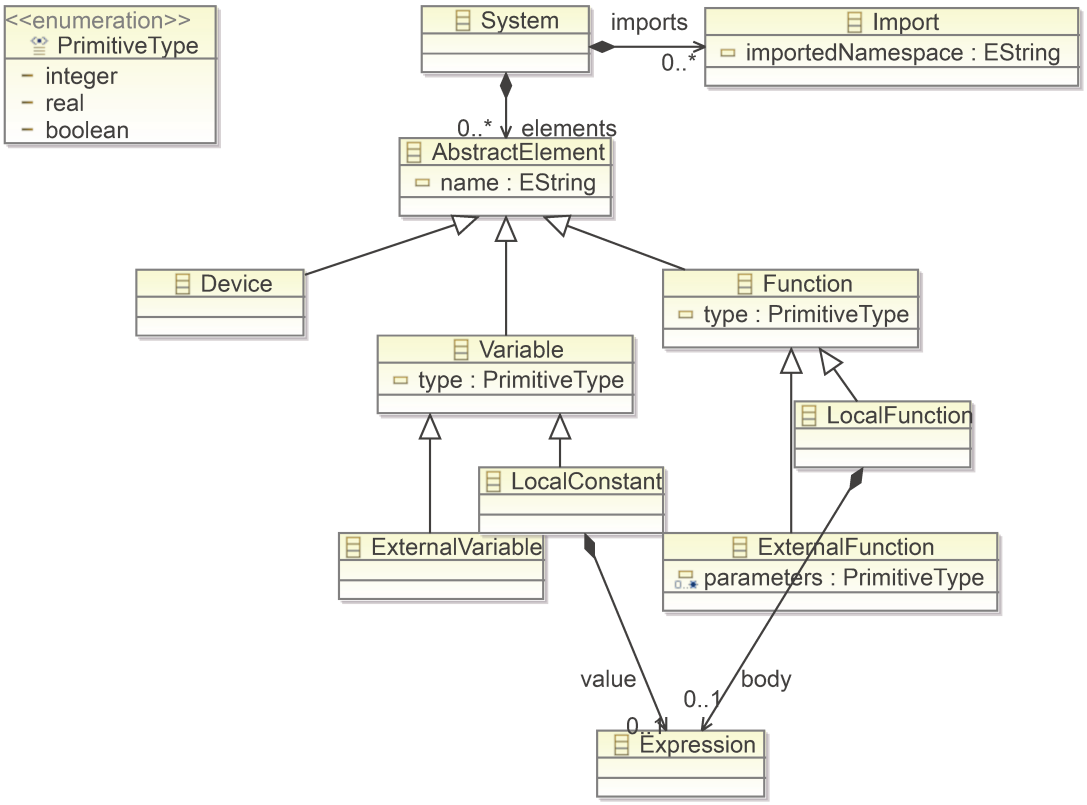


FIGURE 3.1 – Vue graphique du métamodèle *Ecore* de la caractérisation des organes consommateurs d'un véhicule.

et l'abstraction d'un système en séparant sa configuration dans un modèle dédié. De même, une fonction locale est définie dans un système et reste interne au cadre du gestionnaire d'énergie.

Il existe plusieurs initiatives de représentation d'expressions mathématiques. Citons par exemple OpenMath [The00], un effort de standardisation de la sémantique d'objets mathématiques afin de faciliter leur échange à travers le monde numérique. La sémantique des objets y est décrite grâce à la composition d'opérations et de définitions. MathML [W3C98] s'intéresse plus à la représentation d'objets mathématiques, sans forcément en connaître la sémantique exacte. Ce langage est défini pour capturer la structure de formules mathématiques et l'afficher de manière adéquate. Ces langages sont complexes et requièrent une chaîne de traitement propre à un langage (parser, lexer, compilateur, etc.). Comme nous visons une plate-forme embarquée, la disponibilité de librairies mathématiques avancée n'est certainement pas assurée. Ainsi, il faudrait également assurer la transformation des objets définis à l'aide de langages complexes en structures plus simples, compréhensibles par la plate-forme cible. Nous proposons donc un langage simple, proche des structures mathématiques de base disponibles dans les langages couramment embarqués (p. ex. C, C++). Il offre néanmoins la possibilité d'utiliser des structures propres à la plate-forme *via* les fonctions externes.

Nous avons introduit les expressions comme des expressions mathématiques constantes ou à évaluer à l'exécution. Elles sont composées de littéraux typés (booléens, entiers et réels) et d'opérations sur ceux-ci. Ces dernières sont les opérations mathématiques classiques suivantes : $+$, $-$, $\times$, $\div$, ^{x}, $\sqrt{}$, abs, cos, sin, tan, $<$, $\leq$, $=$, $\neq$, $>$, $\geq$ ainsi que la négation d'une expression booléenne. De plus, des fonctions d'interpolations linéaires sont fournies pour $\mathbb{R}^2$ et $\mathbb{R}^3$ (respectivement appelées `interpolate2` et `interpolate3`).

Pour faciliter la manipulation de modèles, deux techniques de représentation sont couramment utilisées. La première est l'utilisation d'artefacts graphiques. L'utilisateur manipule par exemple des boîtes et des flèches représentant des concepts et leurs liens, comme par exemple dans un diagramme de classe UML. La seconde technique est textuelle. L'utilisateur manipule dans ce cas des mots clefs propres au domaine, définis au sein d'une grammaire. C'est le cas des langages de programmation textuels. Nous proposons maintenant un langage textuel de modélisation dédié au métamodèle de caractérisation.

3.1.1.2 La modélisation des organes consommateurs par un langage dédié *Xtext*

Le langage de modélisation dédié d'ORQA permet aux concepteurs de manipuler les modèles de caractérisation plus simplement qu'avec la solution par défaut d'EMF qui repose sur des fichiers XMI. Ce langage est décrit grâce à *Xtext*, qui

permet la définition d'un langage de modélisation dédié à partir d'un modèle *Ecore*. Cette définition est donnée sous la forme d'une notation de Backus–Naur étendue (*Extended Backus–Naur Form*, EBNF). De plus, *Xtext* génère automatiquement les différents outils requis par un langage (arbre de syntaxe abstraite, parser, lexeur, compilateur, etc.). Ce langage dédié reprend tous les concepts décrits par le métamodèle de caractérisation avec une approche textuelle à base de mots clefs tels que `device`, `operating state`, `required power`, `quality`, etc.

La notion d'import est gérée de manière automatique par le canevas logiciel *Xtext*. Elle est totalement transparente pour les concepteurs, ils peuvent alors importer d'autres modèles décrits par le langage. Par défaut, un système importé fusionne son espace de nom avec celui du système courant. L'utilisation des éléments du système se fait alors de manière directe, *via* la référence des éléments. Pour éviter toute conflit de nommage ou pour simplement préciser la provenance d'éléments, il est possible de définir un espace de nom local à un système importé grâce au mot-clef `as`. Par exemple, `import Common.orqa as cmn` réalise l'import du fichier `Common.orqa` dans le modèle courant ; ses éléments sont accessibles en utilisant l'espace de nom `cmn` (p. ex. `cmn.rho` pour l'élément `rho`).

Nous utilisons par la suite quatre variables externes spéciales reconnues par le canevas logiciel : la vitesse du véhicule `vehVelocity`, son accélération `vehAcceleration`, la pente de la route `roadSlope` et le coefficient de vitesse `velocityCoefficient`. Ces données sont requises par le composant gestionnaire d'énergie et sont offertes dans la modélisation *via* le fichier `Common.orqa`.

3.1.2 Bibliothèque de modèles

ORQA intègre une bibliothèque de modèles d'organes génériques afin de simplifier le travail des concepteurs. En effet, de la même manière qu'il est commun d'utiliser des composants « sur étagère », nous proposons des modèles d'organes dans une bibliothèque logicielle pouvant servir de base aux modèles de nouveaux systèmes. Les modèles proposés dans la bibliothèque sont des exemples de caractérisation des principaux organes contrôlables (la chaîne de traction et la climatisation) et observables (le système d'éclairage et un système tiers). Les concepteurs peuvent les utiliser tels quels ou bien les utiliser comme base pour leur modélisation.

Les valeurs constantes (g l'accélération de la pesanteur et ρ (`rho`) la densité de l'air) et les variables d'environnement (la vitesse et l'accélération du véhicule, la pente de la route ainsi que le coefficient de vitesse actuel) sont regroupées dans un modèle commun (cf. listing 3.1) accessibles à tous les systèmes par un import. Nous décrivons maintenant la caractérisation proposée dans la bibliothèque pour chacun des principaux organes.

Listing 3.1 – Définitions de base communes à tous les systèmes : `Common.orqa`.

```
1 system {
2   variables {
3     const real g    := 9.80665
4     const real rho  := 1.204
5     real vehVelocity, vehAcceleration, roadSlope,
          velocityCoefficient
6   }
7 }
```

3.1.2.1 La chaîne de traction

Le véhicule est déplacé grâce à la chaîne de traction qui entraîne ses roues. D'après les lois de la dynamique du véhicule [Rob11, EGE09, GCE99, LL12], les forces qui s'appliquent à ce dernier sont la force motrice qui s'oppose aux forces résistives (la force aérodynamique, la résistance au roulement ainsi que la résistance à la pente). Ces dernières sont exprimées selon la vitesse v et l'accélération a du véhicule ainsi que par la pente α de la route :

$$F_{aero} = 1/2 \times \rho \times c_x \times A \times v^2 \tag{3.1}$$

$$F_{rr} = m \times g \times k_{rr} \tag{3.2}$$

$$F_{cr} = m \times g \times \sin(\alpha) \tag{3.3}$$

où F_{aero} est la force aérodynamique, F_{rr} est la résistance au roulement et F_{cr} est la résistance à la pente. ρ et g sont respectivement, et de manière classique, la densité de l'air et l'accélération de la pesanteur terrestre. m est la masse du véhicule, c_x est son coefficient de pénétration dans l'air, A sa surface au vent et k_{rr} le coefficient de résistance au roulement.

La chaîne de traction du véhicule doit fournir une force de propulsion F_{prop} suffisante pour contrer les forces résistives s'y appliquant. Le moteur électrique meut le véhicule par le biais de la transmission qui transmet son mouvement aux roues motrices. La transmission possède une certaine efficacité η_T, elle peut être directe ou posséder plusieurs vitesses. Le choix du type de transmission dépend des caractéristiques du moteur employé, il est possible d'atteindre des performances équivalentes en terme d'accélération et de démultiplication [LL12]. Dans le cas d'une transmission directe de démultiplication K_T, la puissance délivrée par le moteur P_{deliv} est définie par :

$$P_{deliv} = \frac{F_{prop} \times v}{\eta_T \times K_T} \tag{3.4}$$

La deuxième loi de Newton définit la somme des forces s'appliquant à un corps comme l'accélération de ce dernier, proportionnellement à sa masse. En utilisant les forces définies précédemment, nous obtenons :

$$a \times m = F_{prop} - F_{aero} - F_{rr} - F_{cr} \tag{3.5}$$

ou encore :

$$F_{prop} = F_{aero} + F_{rr} + F_{cr} + a \times m \tag{3.6}$$

Le moteur électrique a une efficacité différente en fonction de son mode opératoire, c'est à dire s'il meut le véhicule (moteur) ou s'il le ralentit (générateur) pour récupérer une partie de l'énergie du freinage. La puissance électrique (consommée ou générée) du moteur est alors définie par :

$$P_{moteur} = \begin{cases} P_{deliv} \div \eta_{moteur} & \text{dans le mode opératoire } \textit{moteur} \\ P_{deliv} \times \eta_{generateur} & \text{dans le mode opératoire } \textit{générateur} \end{cases} \tag{3.7}$$

L'efficacité d'un moteur électrique peut être ramenée à une fonction dépendant de sa vitesse angulaire ω et du couple demandé en sortie [SF12, LL12]. Ces paramètres peuvent être déduits à partir de la vitesse du véhicule et de la puissance que doit délivrer la chaîne de traction de la manière suivante :

$$\omega = k_T \times \frac{v}{r} \tag{3.8}$$

$$couple = \frac{F_{prop} \times r}{k_T} \tag{3.9}$$

où r est le rayon des roues. Un exemple d'une telle fonction est la cartographie illustrée par la figure 3.2.

Les lois et fonctions définies précédemment sont caractérisées par un ensemble d'éléments propres à chaque véhicule (et à sa chaîne de traction). Nous séparons les définitions génériques des caractéristiques du véhicule pour rendre notre modèle de chaîne de traction facilement adaptable à différentes cibles. Ainsi, le listing 3.2 est le modèle textuel générique de la bibliothèque et le listing 3.3 est un exemple d'utilisation de ce modèle générique.

Étudions maintenant ce modèle générique de chaîne de traction. Les définitions communes sont importées (ligne 1) sous l'espace de nom `common`. Les caractéristiques propre à une chaîne de traction spécifique sont définies comme variables externes (ligne 4). Les expressions constantes de la force de propulsion par la chaîne de traction (3.6) ont été extraites et sont définies lignes 5 à 7. Nous retrouvons ensuite une fonction externe pour calculer l'efficacité du moteur (ligne 10). Puis, les fonctions locales comme la puissance finale requise par le moteur (3.7) aux lignes 19 et 23. Les qualités de service offertes dans les deux modes opératoires

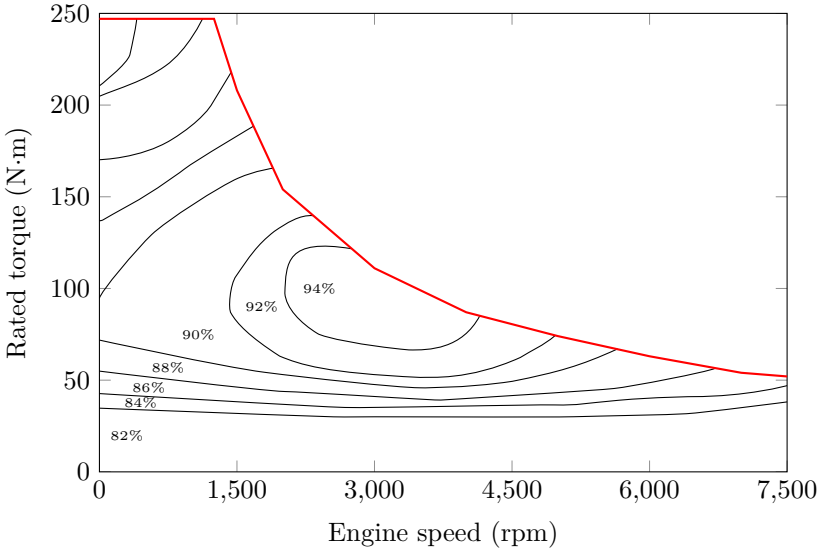


FIGURE 3.2 – Cartographie de l'efficacité d'un moteur électrique de berline en fonction de la vitesse angulaire du moteur et du couple demandé.

sont identiques et prennent la valeur du coefficient de vitesse (ligne 20 et ligne 24). Enfin, les lignes 29 et 30 définissent le changement de mode opératoire du moteur, à quel moment il meut le véhicule (accélération du véhicule nulle ou positive) et à quel moment il est utilisé pour récupérer de l'énergie (accélération négative).

Le listing 3.3 illustre une spécialisation d'un modèle générique de la bibliothèque. Le modèle de la bibliothèque est en premier lieu importé (ligne 1). Les variables et fonctions externes sont ensuite définies. Les caractéristiques du véhicule sont définies par des constantes (lignes 4 à 10). L'efficacité du moteur est définie comme une fonction d'interpolation (ligne 14). Elle est décrite par une cartographie discrète fonction de la vitesse de rotation du moteur et du couple demandé en sortie. Cette cartographie est interpolée linéairement pour fournir l'efficacité d'un couple ⟨couple,vitesse⟩ (⟨torque,speed⟩ en anglais). Dans l'exemple de ce listing, l'ensemble des couples (ligne 15) et l'ensemble des vitesses (ligne 16) indexent l'ensemble des efficacités (ligne 17). Enfin, la ligne 22 annonce l'utilisation de l'organe chaîne de traction.

3.1.2.2 La climatisation

La climatisation est le système de gestion de température de l'habitacle. Elle permet au conducteur de rafraîchir ou de réchauffer ce dernier moyennant une consommation énergétique. La température de l'habitacle est influencée par plusieurs facteurs, nous retenons les trois plus importants. En premier lieu le rayonne-

Listing 3.2 – Modèle théorique de la chaîne de traction : `Drivechain.orqa`.

```
import Common.orqa as common
system {
  variables {
    real m, eta_T, k_T, k_rr, c_x, A, r
    const real C_1 := 0.5 * common.rho * c_x * A
    const real C_2 := m * common.g * k_rr
    const real C_3 := m * common.g
  }
  functions {
    real engine_efficiency(real torque, real speed)
    real Fprop() : (C_1 * vehVelocity**2) + C_2 + (C_3 *
       sin(roadSlope)) + (m * vehAcceleration)
    real Pdeliv() : (Fprop() * vehVelocity) / (eta_T * k_T)
    real engineSpeed() : (k_T * vehVelocity) / r
    real engineTorque() : (Fprop() * r) / k_T
  }
  device Drivechain {
    operating states {
      operating state Motor {
        required power Pdeliv() /
           engineEfficiency(engineSpeed(), engineTorque())
        quality velocityCoefficient
      }
      operating state Generator {
        required power Pdeliv() *
           engineEfficiency(engineSpeed(), engineTorque())
        quality velocityCoefficient
      }
    }
    default operating state Motor
    transitions {
      Generator to Motor when vehAcceleration >= 0
      Motor to Generator when vehAcceleration < 0
    }
  }
}
```

Listing 3.3 – Exemple d'utilisation du modèle générique de la chaîne de traction : `myDrivechain.orqa`.

```
import Drivechain.orqa
system {
  variables {
    const real m       := 1500
    const real eta_T   := 0.95
    const real k_T     := 1.0
    const real k_rr    := 0.008
    const real c_x     := 1.3
    const real A       := 2.75
    const real r       := 0.3
  }
  functions {
    real engine_efficiency(real torque, real speed) :
      interpolate3(
        [0,50,100,150,200,250,300,350,400, ... ,650],
        [-305,-275,-245, ... ,-5,0,5, ... ,245,275,305],
        [0.905,0.905,0.905, ... ,0.76,0.74,0.72],
        torque,
        speed
      )
  }
  device Drivechain
}
```

ment solaire, il chauffe de manière directe ou indirecte (par réflexion) la carrosserie et les surfaces vitrées du véhicule. En second lieu, la vitesse du véhicule modifie la convection de l'extérieur du véhicule. La convection augmente avec une vitesse élevée, ce qui entraîne un transfert thermique plus efficace entre l'extérieur du véhicule et l'air qui l'entoure. C'est d'ailleurs le troisième facteur : la température de l'air environnant le véhicule.

À partir des facteurs énoncés précédemment, nous pouvons résumer la demande énergétique d'un système de climatisation par :

$$P_{clim} = f_1(T_{int},T_{ext}) + f_2(T_{int},T_{obj}) + f_3(v) + P_{soleil} \tag{3.10}$$

avec f_1 à f_4 des fonctions à définir, T_{int} la température intérieure, T_{ext} la température extérieure, T_{obj} la température souhaitée par le conducteur et P_{soleil} la puissance récupérée par le rayonnement solaire.

Nous utilisons une approche empirique d'une climatisation basée sur des données collectées lors d'une campagne visant à évaluer la surconsommation induite par l'usage d'air-conditionné dans un véhicule [Bar98]. Ces données (cf. tableau 3.1) nous permettent d'établir un modèle de régression sur la puissance de fonctionnement moyenne requise par le système de climatisation en fonction de la température extérieure et de la température souhaitée par le conducteur. Le modèle de régression linéaire estimant la puissance moyenne de la climatisation est alors :

$$P_{clim} = 151 \times |T_{ext} - T_{obj}| + 116 \tag{3.11}$$

avec une erreur absolue moyenne de moins de 6% par rapport aux données collectées. Il est à noter que cette fonction n'est définie que pour certaines valeurs de températures estivales et que la température extérieure est systématiquement supérieure à la température souhaitée.

Le système que nous proposons possède trois modes opératoires : éteint, fonctionnement économique et fonctionnement automatique. Le fonctionnement automatique est le fonctionnement nominal du système de climatisation, c'est à dire celui qui cherche à répondre à la demande de l'utilisateur. Nous le définissons par le modèle de régression défini ci-dessus. Le fonctionnement économique consiste à diminuer, par un coefficient de réduction, l'écart de température souhaité par le conducteur pour diminuer la puissance requise.

Le modèle ORQA empirique de climatisation est présenté dans le listing 3.4. Il est à noter que le coefficient de réduction de température est utilisé non seulement pour calculer la puissance requise (ligne 17) mais aussi pour la qualité du mode opératoire économique (ligne 18). Nous proposons en effet de réduire la qualité de service d'autant que l'écart de température souhaité par le conducteur. Nous définissons tout d'abord trois variables entières (ligne 3) qui vont nous permettre

TABLE 3.1 – Puissances moyennes requises par la climatisation sur un cycle de conduite NEDC issues d'une campagne menée par Barbusse et coll. [Bar98]. Les puissances estimées sont obtenues par modèle de régression linéaire.

Températures			Puissances	
extérieure	souhaitée	écart	requise	estimée
25°C	20°C	5°C	843W	871W
30°C	23°C	7°C	1122W	1173W
30°C	20°C	10°C	1463W	1626W
35°C	25°C	10°C	1685W	1626W
35°C	23°C	12°C	2109W	1928W
35°C	20°C	15°C	2492W	2381W
40°C	20°C	20°C	3009W	3136W

d'encapsuler l'état courant de la climatisation (ligne 6). De plus, l'écart de température entre l'extérieur et la température souhaitée par le conducteur est également définie de manière externe (ligne 7). L'état courant de la climatisation et l'écart de température sont typiquement liés à des capteurs et actionneurs physiques, nous déportons donc leur calcul hors du modèle générique. La suite du modèle générique suit le fonctionnement que nous avons décrit ci-dessus.

Le listing 3.5 est un exemple d'utilisation du modèle générique de climatisation. Il contient uniquement la définition du coefficient de réduction (ligne 4) et l'utilisation de l'organe (ligne 6). Les fonctions d'état courant (`operatingState()`) et d'écart de température (`delta_T()`) ne sont pas définies mais uniquement référencées. Le canevas logiciel considère qu'elles sont définies dans l'implémentation (le code ANSI-C) et qu'il peut y faire référence. Les développeurs ont pour charge de s'assurer de leur implémentation.

3.1.2.3 Le système d'éclairage

Le système d'éclairage est un organe observable. Son comportement peut être spécifié de manière automatique par le système du véhicule ou de manière manuelle *via* un interrupteur à disposition du conducteur. Dans ces deux cas, nous proposons un système simple composé d'un mode opératoire par type d'éclairage qui utilise des ampoules aux besoins énergétiques constants. Les types d'éclairage proposés sont : éteint *(off)*, feux de position *(day-time)*, feux de croisement *(low beam)* et feux de route *(high beam)*.

Le listing 3.6 présente le système générique proposé dans la bibliothèque. Comme pour le modèle de la climatisation, nous définissons les modes opératoires adéquats (ligne 3) ainsi qu'une fonction externe `operatingState()` (ligne 7) ren-

Listing 3.4 – Un modèle empirique de climatisation : `ClimateControl.orqa`.

```
system {
  variables {
    int OFF, ECO, AUTO
  }
  functions {
    int operatingState()
    real delta_T()
    real Pclim(real k) : 151 * delta_T() * k + 116
  }
  device ClimateControl {
    operating states {
      operating state Off {
        required power 0
        quality 0
      }
      operating state Eco {
        required power Pclim(k_eco)
        quality 100 * k_eco
      }
      operating state Auto {
        required power Pclim(1)
        quality 100
      }
    }
    default operating state Off
    transitions {
      * to Off when operatingState() = OFF
      * to Eco when operatingState() = ECO
      * to Auto when operatingState() = AUTO
    }
  }
}
```

Listing 3.5 – Exemple d'utilisation du modèle empirique du système de climatisation : `myClimateControl.orqa`.

```
1 import ClimateControl.orqa
2 system {
3   variables {
4     const real k_eco := 0.5
5   }
6   device ClimateControl
7 }
```

voyant le mode opératoire courant. L'idée sous-jacente est la même, nous utilisons une approche d'encapsulation de l'état physique.

Le listing 3.7 est un exemple d'utilisation du système générique d'éclairage. Les puissances de fonctionnement des ampoules sont définies aux lignes 4 à 7, ces valeurs sont des exemples issus d'un manuel automobile Bosch [Rob11].

3.1.2.4 Les autres consommateurs

Enfin, nous regroupons les autres organes consommateurs observables dans un système de puissance requise constante. Ce sont des consommateurs dont les consommations sont approximées à des constantes, soit parce qu'elles sont difficilement quantifiables, soit parce qu'elles ne varient pas. Par exemple, nous y regroupons les systèmes embarqués (requérant 100W au pire des cas) ainsi que le tableau de bord et tous ses indicateurs (environ 50W). Nous y ajoutons également les organes observables utilisés de manière sporadique comme les essuies-glaces. Le système que nous proposons est défini par la pire puissance requise par ses différents organes. Ce système simplifié a un intérêt lors de simulation pour évaluer le pire scénario. Dans le cas d'un véhicule réel, les concepteurs ont un retour des consommations effectives et sont en mesure de définir plus précisément les puissances requises par les différents organes. Le même manuel automobile [Rob11] est une fois de plus utilisé pour estimer les besoins énergétiques de ces autres organes.

Nous retrouvons le modèle de ce système et un exemple d'utilisation dans le listing 3.8 et dans le listing 3.9.

3.1.3 Conclusion

Dans cette section, nous avons tout d'abord détaillé le métamodèle tel qu'il est réalisé en *Ecore* et plus particulièrement les éléments supplémentaires facilitant son utilisation. Puis, nous avons défini un langage de modélisation dédié. Ce langage, utilisable de manière optionnelle, est introduit afin d'offrir aux concepteurs

Listing 3.6 – Un modèle simplifié du système d'éclairage : `Lighting.orqa`.

```
system {
  variables {
    int OFF, DAYTIME, LOWBEAM, HIGHBEAM
    real Poff, Pdaytime, Plowbeam, Phighbeam
  }
  functions {
    int operatingState()
  }
  device ClimateControl {
    operating states {
      operating state Off {
        required power Poff
      }
      operating state DayTime {
        required power Pdaytime
      }
      operating state LowBeam {
        required power Plowbeam
      }
      operating state HighBeam {
        required power Phighbeam
      }
    }
    default operating state Off
    transitions {
      * to Off when operatingState() = OFF
      * to DayTime when operatingState() = DAYTIME
      * to LowBeam when operatingState() = LOWBEAM
      * to HighBeam when operatingState() = HIGHBEAM
    }
  }
}
```

Listing 3.7 – Exemple d'utilisation du modèle simplifié du système générique d'éclairage : `myLighting.orqa`.

```
import Lighting.orqa
system {
  variables {
    const real Poff       := 0
    const real Pdaytime   := 50
    const real Plowbeam   := 200
    const real Phighbeam  := 250
  }
  device Lighting
}
```

Listing 3.8 – Modèle générique d'un système consommateur à puissance constante : `StaticMiscellaneous.orqa`.

```
system {
  variables {
    real P_misc
  }
  device StaticMiscellaneous {
    operating states {
      operating state Nominal {
        required power P_misc
      }
    }
    default operating state Nominal
  }
}
```

Listing 3.9 – Exemple d'utilisation du modèle générique d'un système consommateur à puissance constante : `myStaticMiscellaneous.orqa`.

```
import StaticMiscellaneous.orqa
system {
  variables {
    const real P_misc := 1500
  }
  device StaticMiscellaneous
}
```

une modélisation textuelle reprenant les concepts présents dans le métamodèle. Finalement, nous avons présenté la bibliothèque de modèles fournie avec ORQA. Elle offre des modèles de caractérisation génériques pour les principaux organes consommateurs du véhicule. Ces modèles sont extensibles afin d'être utilisés comme base pour différents systèmes réels.

3.2 Extraction des composants (AUTOSAR)

AUTOSAR est le standard de modélisation utilisé dans les systèmes embarqués des véhicules. Il permet de décrire l'architecture des fonctionnalités logicielles embarquées dans le véhicule. Cette architecture définit les relations entre les différents composants physiques et logiciels du véhicule ainsi que leur fonctionnement. En conséquence, la connaissance des organes est déjà modélisée en AUTOSAR. Nous avons alors à extraire les éléments nécessaires pour constituer la base du modèle de caractérisation des organes consommateurs. Nous supposons l'architecture complète vis à vis des organes consommateurs présents dans le véhicule. Notre prototype extrait les composants des organes consommateurs du modèle AUTOSAR vers un nouveau modèle de caractérisation *via* une transformation dédiée que nous décrivons maintenant.

Dans le modèle AUTOSAR, les organes sont des composants appartenant à une catégorie spéciale, les capteurs et les actionneurs *(Sensor–Actuator*s*)*. Ceci signifie qu'ils sont liés au matériel et qu'ils agissent comme interface entre le monde logiciel et le monde matériel. Ces composants offrent et reçoivent des données qui peuvent être des commandes ou des valeurs discrètes grâce à des ports dédiés. Les composants consommateurs contrôlables par ORQA ont nécessairement un port de commande par lequel le gestionnaire d'énergie pourra contrôler le comportement de l'organe. Les commandes pouvant transiter par ces ports sont définies dans le modèle AUTOSAR. Nous supposons qu'elles sont regroupées au sein d'énumérations composées des différents modes opératoires de chaque organe et qu'un port de commande est nommé `command`.

L'extraction des composants des organes consommateurs est réalisée de la manière suivante (cf. algorithme 3.1). Le modèle initial AUTOSAR est parcouru à la recherche des organes consommateurs et de leur contrôle, c'est à dire de composants capteurs ou actionneurs ayant un port de commande en entrée (ligne 2). Un tel organe est, par défaut, associé à un organe observable (ligne 5). Les concepteurs feront ensuite la distinction entre les organes observables, les organes contrôlables et éventuellement les organes à ignorer. À partir des valeurs prises par le port de commande, les modes opératoires de l'organe sont définis : les valeurs prises par la commande (ligne 7) et les modes opératoires en eux-mêmes (ligne 8).

Le modèle de caractérisation est maintenant initialisé, il contient les définitions

Algorithme 3.1 Règle d'extraction des composants consommateurs du modèle initial AUTOSAR.

Modèle d'entrée : initial **conforme à** AUTOSAR
Modèle de sortie : carac **conforme à** ORQA

```
1: Règle INTERFACES avec source : AUTOSAR→SWC
2:     si non (source.isInstanceOf("SWC_SensorActuator") et source.hasReceiverPort("command")) alors
3:         continue
4:     fin si
5:     d ← carac.getSystem().createInstance("ObservableDevice", source.getName())
6:     pour chaque o ∈ source.getReceiverPort("command").getEnumeration()
7:         d.addVariable("int", o.getName(), o.getValue())
8:         d.addOperatingState(o.getName().toCamelCase())
9:     fin pour chaque
10: fin Règle
```

des organes consommateurs tels qu'ils sont décrits dans le modèle initial. Il est nécessaire de différencier les organes contrôlables des organes observables car cette information n'est pas disponible au niveau du modèle initial. Elle est en effet considérable comme étant une propriété extra-fonctionnelle (). De plus, les transitions entre les modes opératoires ne sont pas extraites car elles relèvent du comportement des organes, concept qui n'est pas non plus présent dans le modèle initial. Nous passons maintenant à la caractérisation des organes consommateurs pour y intégrer les connaissances extra-fonctionnelles requises par le canevas logiciel.

3.3 Mise en place du système de gestion d'énergie

La mise en place du système de gestion d'énergie requiert d'intégrer les connaissances énergétiques et la politique de choix au sein de l'architecture cible. Notre prototype étant basé sur AUTOSAR, l'architecture cible du système de gestion d'énergie est intégrée au modèle AUTOSAR initial que nous appelons alors *modèle amélioré*. Le modèle amélioré est composé d'une partie indépendante du véhicule cible, de ses organes et des connaissances contenues dans le modèle de caractérisation. Ces deux parties sont implémentées en ANSI-C, la première est directement proposée dans le canevas logiciel alors que la seconde est le résultat d'une transformation de modèle appliquée au modèle de caractérisation. Nous décrivons maintenant les deux étapes de la mise en place du système de gestion d'énergie

par deux transformations de modèles.

3.3.1 Intégration de l'architecture du système de gestion d'énergie

L'architecture du système de gestion d'énergie est basée sur un composant gestionnaire couplé à des composants courtiers (cf. section 2.5). La transformation de modèle définit dans un premier temps les interfaces de ces nouveaux composants. Puis, dans un second temps, la transformation produit leurs instances.

3.3.1.1 Création des interfaces des composants

L'algorithme 3.2 présente de manière simplifiée la règle de définition des interfaces des composants courtiers et du gestionnaire d'énergie. Le modèle de caractérisation est le modèle d'entrée (`carac`) à partir duquel on améliore le modèle initial (`initial`) pour créer le modèle amélioré (`enhanced`). Ce dernier a pour base le modèle initial (ligne 2).

Le gestionnaire d'énergie requiert en entrée les éléments de la requête conducteur (ligne 5), à savoir les routes possibles et les contraintes fixées par le conducteur. Il propose en sortie la stratégie de conduite solution (ligne 6).

Un type de composant courtier est défini pour chaque organe contrôlable (ligne 12). Il reprend le nom de l'organe défini dans le modèle de départ en y ajoutant le suffixe `_Broker_Type`. Un courtier agit en tant que relais pour la commande d'un organe. Il requiert deux entrées de commande, l'une pour la commande originale de contrôle (ligne 13) et l'autre pour la commande provenant du gestionnaire d'énergie (ligne 14). Comme le gestionnaire d'énergie définit son état (*actif* ou *inactif*), le courtier possède une troisième entrée pour le contrôle de son état (ligne 15). L'unique sortie du courtier (ligne 16) permet d'envoyer la commande finale à l'organe contrôlé.

Par réflexion, le gestionnaire d'énergie possède deux ports en sortie par composant courtier pour définir leurs états et leurs commandes (ligne 17 et ligne 18).

3.3.1.2 Instanciation des composants

L'algorithme 3.3 présente de manière simplifiée la règle d'instanciation des composants courtiers et du gestionnaire d'énergie. Le gestionnaire d'énergie est instancié une unique fois (ligne 3), de même que les courtiers (ligne 9), à la racine du système (ligne 2). Il est ensuite nécessaire d'établir les connexions requises par les courtiers et par le gestionnaire d'énergie. Chaque courtier est associé à un organe spécifique, il est positionné directement en amont de son port de contrôle afin d'intercepter les commandes de contrôle venant du système.

Algorithme 3.2 Règle de création des interfaces des courtiers et de l'interface du gestionnaire d'énergie dans le modèle amélioré AUTOSAR.

Modèle d'entrée : carac **conforme à** ORQA
Modèle d'entrée : initial **conforme à** AUTOSAR
Modèle de sortie : enhanced **conforme à** AUTOSAR

```
1: Règle INTERFACES avec source : ORQA→System
2:     enhanced ← copie de initial
3:     target ← enhanced.getRootComposition()
4:     emi ← target.createInstance("SWC_Type", "EnergyManager_Type")
5:     emi.addReceiverPort("request", "DriverRequest")
6:     emi.addSenderPort("drivingStrategy", "DrivingStrategy")
7:     pour chaque device ∈ source.devices
8:         si non device instance de ORQA→ControllableDevice alors
9:             continue
10:        fin si
11:        cmdtype ← device.getPort("command").getType()
12:        bi ← target.createInstance("SWC_Type", device.getName() +"_Bro-
    ker_Type")
13:        bi.addReceiverPort("command_original", cmdtype)
14:        bi.addReceiverPort("command_EM", cmdtype)
15:        bi.addReceiverPort("state", "BrokerState")
16:        bi.addSenderPort("command", cmdtype)
17:        emi.addSenderPort("state_"+device.getName(), "BrokerState")
18:        emi.addSenderPort("command_"+device.getName(), cmdtype)
19:    fin pour chaque
20: fin Règle
```

Les connexions existantes vers le port de contrôle de l'organe sont détachées de l'organe pour arriver sur le port de contrôle en entrée du courtier (ligne 11). Deux nouvelles connexions sont créées du gestionnaire d'énergie vers le courtier, une pour la commande (ligne 13) et une pour l'état du courtier (ligne 14). De plus, le courtier est connecté à l'organe qu'il contrôle (ligne 15). Enfin, le service principal du gestionnaire d'énergie – la requête du conducteur de réalisation d'un trajet – est laissé déconnectée. C'est aux concepteurs de l'inter-connecter *via* une interface homme machine. Cette opération est hors du champ d'application de nos travaux.

Algorithme 3.3 Règle d'instanciation des courtiers et du gestionnaire d'énergie dans le modèle amélioré AUTOSAR.

Modèle d'entrée : carac **conforme à** ORQA
Modèle d'entrée : enhanced **conforme à** AUTOSAR
Modèle de sortie : enhanced **conforme à** AUTOSAR

```
1: Règle INSTANCES avec source : ORQA→System
2:     target ← enhanced.getRootComposition()
3:     em ← target.createInstance("EnergyManager_Type", "EnergyManager")
4:     pour chaque device ∈ source.devices
5:         si non device instance de ORQA→ControllableDevice alors
6:             continue
7:         fin si
8:         swc ← target.getInstance(device.getName())
9:         broker ← target.createInstance(device.getName()+"_Broker_Type",
   device.getName()+"_Broker")
10:        pour chaque conn ∈ swc.getReceiverPort("command").getConnections()
11:            conn.bindTo(broker, "command_original")
12:        fin pour chaque
13:        em.getSenderPort("command_"+device.getName()).createConnection()
   .bindTo(broker, "command_EM")
14:        em.getSenderPort("state_"+device.getName()).createConnection()
   .bindTo(broker, "state")
15:        broker.getSenderPort("command").createConnection().bindTo(swc,
   "command")
16:    fin pour chaque
17: fin Règle
```

Le comportement du gestionnaire d'énergie est générique, sa configuration est propre au véhicule qu'il intègre, elle est réalisée séparément. Le gestionnaire d'énergie est par conséquent déjà implémenté dans le canevas logiciel. La liaison entre la plate-forme cible et le code générique du gestionnaire ainsi que l'implémenta-

TABLE 3.2 – Correspondance de représentation d'un organe entre le modèle de caractérisation et l'encapsulation ANSI-C.

Modèle de caractérisation	Implémentation en ANSI-C
O, ensemble des modes opératoires o_0, mode opératoire initial	tableau + compteur littéral
C, ensemble des conditions T, ensemble des transitions	fonction
P, ensemble des puissances requises B, ensemble des besoins énergétiques	fonction
* L, ensemble des qualités de service * Q, ensemble couples de qualité	fonction

* Note : L et Q sont définis pour les organes contrôlables uniquement.

tion des courtiers sont en revanche générés. La liaison correspond à du code de remplissage pour l'appel du code générique depuis le composant AUTOSAR. L'implémentation des courtiers est générée suivant l'algorithme défini précédemment (cf. algorithme 2.2).

3.3.2 Génération du modèle concret des connaissances énergétiques et qualitatives

Le modèle de caractérisation comporte les connaissances énergétiques et qualitatives des organes du véhicule requises pour assurer sa gestion. Nous reprenons ici les caractéristiques définies précédemment (cf. section 2.2) et transposons leur représentation en ANSI-C. La correspondance entre les deux représentations est énumérée dans le tableau 3.2.

L'encapsulation décrite dans le listing 3.10 réalise cette correspondance. Le type `tOS` représente un mode opératoire. Les différentes parties de l'encapsulation sont :

- `name` (ligne 2), le nom de l'organe que l'on retrouve dans le modèle de caractérisation ;
- `operatingStates` (ligne 3), le tableau des modes opératoires de l'organe ;
- `operatingStates_nb` (ligne 4), le nombre de modes opératoires contenus dans le tableau ;
- `defaultOperatingState` (ligne 5), le mode opératoire par défaut de l'organe ;
- `nextState` (ligne 6), la fonction déterminant le prochain état de l'organe en fonction de l'environnement dont la signature est

Listing 3.10 – Définition de l'encapsulation d'un organe consommateur par une structure.

```
1 typedef struct {
2   char * name;
3   tOS operatingStates[];
4   uint8 operatingStates_nb;
5   tOS defaultOperatingState;
6   fConsumingDeviceNextState nextState;
7   fConsumingDeviceRequiredPower requiredPower;
8   fConsumingDeviceQuality quality;
9 } tConsumingDevice;
```

`typedef tOS (*fConsumingDeviceNextState)(tOS);`

- `requiredPower` (ligne 7), la fonction déterminant la puissance requise en fonction d'un mode opératoire dont la signature est `typedef tPower (*fConsumingDeviceRequiredPower)(tOS);`
- `quality` (ligne 8), la fonction déterminant la qualité de service rendue dont la signature est `typedef tQuality (*fConsumingDeviceQuality)(tOS)`.

3.4 Implémentation des optimisations de la configuration du gestionnaire d'énergie

Les deux optimisations de la configuration du gestionnaire d'énergie présentées précédemment (cf. section 2.4) sont toutes deux basées sur l'étude des résultats de routes typiques. Par l'hypothèse que les concepteurs connaissent l'usage auquel est destiné le véhicule, nous supposons qu'ils sont en mesure de définir des routes représentatives. L'organisation des outils proposés par ORQA pour configurer le gestionnaire d'énergie est présentée par la figure 3.3. Chacune des routes typiques est évaluée et ses résultats sont sauvegardés dans une base de données. Nous utilisons les deux approches d'optimisation pour proposer des configurations pertinentes du gestionnaire d'énergie aux concepteurs. Les différentes étapes de l'étude puis de la recherche de configurations sont réalisées par des programme écrits en Python.

L'évaluation des routes est réalisée grâce à l'implémentation du modèle de caractérisation en ANSI-C. Une enveloppe Python permet la manipulation de celle-ci pour évaluer l'ensemble des routes et stocker les ratios des résultats dans la base de données. L'ensemble des résultats manipulés est sauvegardé sous forme de fichiers CSV (*Comma-Separated Values*) [IET95] où les valeurs utilisent les unités

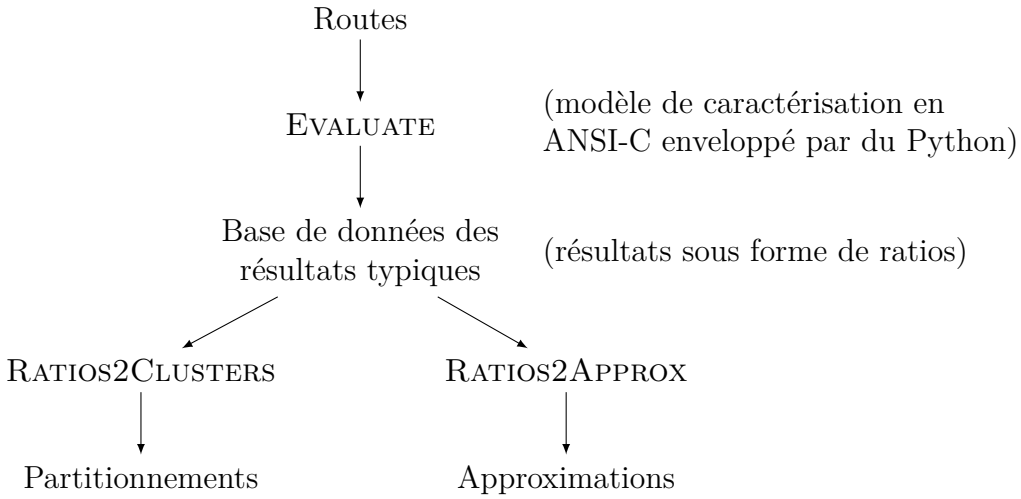


FIGURE 3.3 – Organisation des outils de configuration.

du système international. Ce format est textuel, simple à manipuler et très répandu comme format d'échange de données tabulaires.

La représentation d'une route se fait par portion selon les colonnes de vitesse (`velocity`), de distance (`distance`) et d'angle de la pente (`angle`). Le nom d'une route est le nom du fichier la représentant.

La base de données des résultats typiques est organisée selon les sept colonnes suivantes : 1) `routeIndex` l'indice entier permettant l'indexation des différentes routes ; 2) `route` le nom factuel de la route ; 3) `velocityCoefficient` le coefficient de vitesse ; 4) `statesCombination` la combinaison des organes ; 5) `ratio_T` les ratios de durée ; 6) `ratio_E` les ratios de consommation et 7) `ratio_Q` les ratios de qualité.

3.4.1 Partitionnement des coefficients de vitesse

Le partitionnement des coefficients de vitesse est réalisé en appliquant l'algorithme des *k-médoïdes* sur un ensemble de vecteurs représentant les coefficients de vitesse. L'algorithme des *k-médoïdes* est réalisé par la méthode PAM *(Partitioning Around Medoids)* tel que décrit par l'algorithme 3.4. Nous distinguons deux parties dans cet algorithme, celle d'initialisation (ligne 3) et celle d'échange (lignes 6 à 13) répétée jusqu'à la convergence de l'algorithme. L'algorithme recherche k partitions représentatives d'un ensemble de vecteurs. Une partition est centrée – et est représentée – par un de ces vecteurs, appelé alors médoïde.

Dans PAM, les médoïdes initiaux sont sélectionnés de manière aléatoire (ligne 3). Il est nécessaire de réaliser l'algorithme de partitionnement un grand nombre de

Algorithme 3.4 Partitionnement autour des médoïdes, une implémentation des *k-médoïdes*.

```
1: fonction PAM(vecteurs, k) retourne (médoïdes,associations)
2:     pour i ∈ [1; k] faire
3:         assigner un élément aléatoire de vecteurs à médoïdes[i]
4:     fin pour
5:     pour chaque v ∈ vecteurs
6:         assigner le médoïde le plus proche de v à associations[v]
7:     fin pour chaque
8:     pour chaque m ∈ médoïdes
9:         pour chaque v ∈ vecteurs \ médoïdes
10:            échanger m et v et calculer le coût total des partitions
11:        fin pour chaque
12:    fin pour chaque
13:    sélectionner la configuration de médoïdes la moins coûteuse
14:    répéter les étapes 5 à 13 jusqu'à ce que médoïdes ne change plus
15: fin fonction
```

fois pour s'assurer d'éviter une convergence locale. Grâce à l'initialisation plus réfléchie proposée par la méthode *k-means++* [AV07], il n'est pas nécessaire de rejouer l'algorithme. Cette méthode (cf. algorithme 3.5) définit les médoïdes initiaux tel qu'une couverture maximale des vecteurs est recherchée. Le premier médoïde est sélectionné aléatoirement (ligne 2). Les médoïdes suivants sont aussi sélectionnés de manière aléatoire mais selon une probabilité proportionnelle à d^2 (ligne 7). C'est à dire que la probabilité qu'un vecteur soit sélectionné est égale au carré de sa distance avec le plus proche médoïde divisé par l'ensemble des carrés des distances des vecteurs aux médoïdes. Ainsi, plus un vecteur est éloigné des médoïdes et plus il a de chances d'être sélectionné comme nouveau médoïde. Ceci permet d'augmenter la couverture des partitions initiales en éloignant les médoïdes le plus possible.

La phase d'échange de PAM modifie les partitions en déplaçant les médoïdes un par un (ligne 10). Les médoïdes sont les vecteurs minimisant les coûts des partitions (ligne 13), ils sont au centre de celles qu'ils représentent. Le coût d'une partition C est la somme des distances entre le médoïde m de la partition et les vecteurs qui y appartiennent :

$$coût(C) = \sum_{i \in C} d(i,m) \tag{3.12}$$

où d est une fonction de distance entre deux vecteurs. Nous nous basons sur la distance euclidienne pour un espace à n-dimensions par la suite : $d(x,y) =$

Algorithme 3.5 Sélection des médoïdes initiaux avec la méthode *k-means++*.

1: **fonction** K-MEANS++(*vecteurs*, k) **retourne** *médoïdes*
2: assigner un élément aléatoire de *vecteurs* à *médoïdes*[1]
3: **pour** $i \in [2; k]$ **faire**
4: **pour** $j \in [1; taille(vecteurs)]$ **faire**
5: calculer la distance d[j] du j-ième vecteur *vecteurs*[j] par rapport au médoïde le plus proche
6: **fin pour**
7: assigner un élément aléatoire de *vecteurs* à *médoïdes*[i] selon une probabilité proportionnelle à $\frac{d^2}{\sum d^2}$
8: **fin pour**
9: **fin fonction**

$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$.

Le programme de partitionnement RATIOS2CLUSTERS réalise un partitionnement sur les ratios passés en entrée. Il utilise l'algorithme de partitionnement PAM avec la phase d'initialisation de *k-means++*. Le domaine de k (le nombre de partitions) définit l'ensemble des partitionnements à effectuer. Il peut être défini de manière optionnelle, il vaut par défaut $[1; n]$ avec n le nombre de coefficients de vitesse différents déduits grâce aux ratios. Par exemple, avec des ratios sur $[50; 100]$, le programme calcule le partitionnement avec 1, 2, 3 partitions, jusqu'au cas extrême de 51 partitions où chaque coefficient est dans sa propre partition. Les résultats de ce programme sont organisés par les colonnes `velocityCoefficient`, `error` et `medoids`. La colonne `error` est, pour une ligne, la somme des erreurs des partitions. Ces partitions sont présentées par leurs coefficients de vitesse représentatifs dans la colonne `medoids`.

3.4.2 Approximation des résultats

La simplification de l'évaluation des solutions, ou stratégies de conduite, repose sur les similarités des évolutions des ratios. Sur un graphique, ce sont les courbes des ratios qui évoluent de manière similaire selon les coefficients de vitesse. La méthode proposée est de regrouper différentes évolutions de ratios et d'utiliser l'évolution moyenne pour les approximer (équation (2.45)) à la manière d'une fonction de régression. Ainsi, l'évolution moyenne fournit un ensemble de ratios moyens que l'on peut appliquer à un résultat nominal et obtenir son évolution approximée.

Le programme de simplification RATIOS2APPROX calcule les ratios moyens d'un ensemble de ratios passés en entrée. Il réalise la moyenne d'un type de ratio par coefficients de vitesse et en calcule l'écart type. Les résultats sont orga-

nisés par les cinq colonnes suivantes : `velocityCoefficient`, `ratio_T_approx`, `ratio_T_stddev`, `ratio_E_approx`, `ratio_E_stddev`. Le suffixe `_approx` correspond à la valeur d'approximation dont l'écart type (*standard deviation*) est défini par le suffixe `_stddev`.

3.4.3 Conclusion

Les deux approches d'optimisation de la configuration du gestionnaire d'énergie sont réalisées par une combinaison d'outils. Elles sont toutes les deux basées sur les similarités observables entres les résultats de routes différentes. Nous avons défini dans un premier temps une base de données regroupant les résultats de routes typiques auxquelles sont appliquées les coefficients de vitesse. Nous avons ensuite détaillé la première approche et son outil de partitionnement pour regrouper différents coefficients par les résultats qui y sont liés. Nous réduisons ainsi le domaine des coefficients à un sous-ensemble composé d'un coefficient représentatif par partition. Puis, nous avons abordé l'implémentation de la seconde approche qui établit un ratio moyen par type de résultat pour chaque coefficient de vitesse. Ceci simplifie le calcul des évaluations des stratégies de conduite en permettant d'approcher les valeurs de résultats pour l'ensemble des coefficients de vitesse à partir d'une seule évaluation. Les outils proposés ont été développés en Python.

3.5 Conclusion

Ce chapitre a présenté la mise en œuvre du canevas logiciel ORQA. Dans un premier temps, l'implémentation du métamodèle de caractérisation est présentée avec *Ecore*. Nous y introduisons un concept d'import pour permettre la réutilisation de modèles existants. Afin de simplifier la définition et l'utilisation des modèles de caractérisation par les concepteurs, nous avons introduit un langage de modélisation dédié. Réalisé à l'aide d'*Xtext*, ce langage tire profit du mécanisme d'import qui est géré de manière transparente et automatique par *Xtext*. Ainsi, nous sommes ensuite passés aux modèles prédéfinis et intégrés à ORQA dans une bibliothèque utilisant le langage dédié. Leur rôle est de proposer un exemple de caractérisation pour les principaux organes consommateurs. Grâce au mécanisme d'import, nous pouvons séparer le fonctionnement des organes de leur configuration. C'est à dire que les caractéristiques propres aux organes et dépendants d'un véhicule à un autre sont définis séparément. Les concepteurs peuvent donc importer des modèles pré-définis d'organes et uniquement modifier leurs caractéristiques.

Nous avons ensuite abordé notre prototype compatible AUTOSAR. Son but est d'intégrer les connaissances énergétiques et qualitatives au sein d'une modélisation AUTOSAR grâce à ORQA. Le modèle enrichi résultant de cette intégration reste

compatible AUTOSAR. Le modèle initial AUTOSAR est utilisé comme base pour le modèle de caractérisation. Une transformation de modèle permet l'extraction des informations pertinentes architecturales et comportementales d'AUTOSAR vers ORQA. Les concepteurs complètent ensuite cette modélisation, telle que présentée au chapitre précédent, par les connaissances énergétiques et qualitatives.

Le système de gestion d'énergie est mis en place par deux transformations de modèles. La première permet son intégration dans le modèle AUTOSAR en définissant les composants puis en les instanciant. La seconde transformation permet de générer l'implémentation du système en ANSI-C. Les connaissances énergétiques et qualitatives sont intégrées de manière directe au gestionnaire d'énergie.

Nous avons finalement détaillé l'implémentation des optimisations de la configuration du gestionnaire d'énergie. Les résultats typiques du véhicules sont établis et stockés dans une base de données. À partir de cette base, les deux programmes d'optimisation sont utilisés et permettent de proposer des configurations affinées.

Chapitre 4

Validation

Sommaire

Dans ce chapitre, nous évaluons le canevas logiciel ORQA, sa valeur ajoutée et ses performances. Un canevas logiciel n'est pas formellement évaluable en tant que tel, ni l'architecture à laquelle il est associé. Ce chapitre présente l'utilisation du canevas logiciel et du gestionnaire d'énergie. Nous évaluons la pertinence des optimisations de la configuration du gestionnaire d'énergie et leur influence sur les résultats obtenus. Nous illustrons l'application d'ORQA avec deux exemples de véhicules, à la phase de conception puis à la phase d'utilisation.

Ce chapitre débute par la présentation des exemples étudiés. Nous nous appuyons sur les modèles génériques présents dans la bibliothèque pour caractériser les deux véhicules étudiés. Nous nous basons sur deux scénarios d'utilisation typiques du domaine.

Pour ces exemple, nous étudions différentes configurations possibles pour les gestionnaires d'énergie des deux véhicules. L'application des approches d'optimisation permet de proposer des configurations satisfaisant les contraintes de la plateforme matérielle visée.

Enfin, nous réalisons l'application des deux véhicules avec gestionnaire d'énergie sur les deux scénarios. Nous comparons les solutions fournies par le gestionnaire d'énergie avec les solutions nominales des scénarios, c'est à dire celles sans gestion d'énergie, et concluons finalement sur l'intérêt du gestionnaire d'énergie.

4.1 Présentation de l'exemple

Nous illustrons l'utilisation du canevas logiciel ORQA pour deux véhicules, l'un de petit gabarit à destination majoritairement urbaine et l'autre de type berline, à usage polyvalent. Nous présentons ensuite les scénarios retenus pour illustrer l'utilisation du gestionnaire d'énergie. Nous définissons enfin pour chacun des deux véhicules des préférences d'utilisation afin de pouvoir comparer les solutions obtenues par la suite avec et sans gestionnaire d'énergie.

4.1.1 Les véhicules

Les deux véhicules servant d'exemple sont proches de véhicules actuellement commercialisés en France. Le premier véhicule est un véhicule minimaliste à vocation urbaine, il ne possède pas de module de climatisation, ni d'accessoire de confort consommateur, mais uniquement une chaîne de traction et un système d'éclairage. Sa modélisation – avec le langage dédié d'ORQA – est présentée dans le listing 4.1. Nous utilisons les modèles génériques définis précédemment dans la bibliothèque pour la chaîne de traction, le système d'éclairage et pour le groupement de petits consommateurs (lignes 2 à 4). La configuration de la modélisation de la chaîne de traction est définie par les constantes aux lignes 7 à 13 et par

TABLE 4.1 – Caractéristiques des quatre routes exemples.

Route	Distance	Répartition des environnements urbain	extra-urbain	voie rapide
Sa#1	76km	5%	3%	92%
Sa#2	78km	24%	76%	0%
Sb#1	12km	78%	14%	8%
Sb#2	12km	44%	17%	39%

la fonction d'efficacité ligne 21. La configuration du système d'éclairage reprend l'exemple spécifié précédemment (lignes 14 à 17). La puissance nécessaire aux accessoires consommateurs est supposée moindre que celle présentée précédemment pour une berline, nous la définissons à 750W (ligne 18). La chaîne de traction définie ici correspond à un petit véhicule urbain (dont le moteur est de 16kW) présenté dans un projet régional français [OBA98].

Le second véhicule, plus polyvalent, possède un système de confort supplémentaire : la climatisation. Le listing 4.2 présente la modélisation de ce véhicule avec le langage dédié ORQA. De la même façon que précédemment, nous utilisons les modèles génériques de la bibliothèque et spécifions leurs caractéristiques : chaîne de traction (lignes 8 à 14, ligne 23), système d'éclairage (lignes 15 à 18), climatisation (ligne 19), et accessoires consommateurs (ligne 20). Les caractéristiques de la chaîne de traction proviennent d'une modélisation de la motorisation d'une Toyota Prius™ dans le cadre du projet ADVISOR [MBH+02] soutenu par le ministère de l'énergie et par le laboratoire national des énergies renouvelables des États-Unis d'Amérique.

4.1.2 Les scénarios

Nous étudions deux scénarios pour l'utilisation des véhicules. Le premier scénario *Sa* a pour objectif d'atteindre le campus universitaire de Rennes 1 (Beaulieu) en partant du campus de Laval (cf. figure 4.1). Deux routes principales existent, la première par l'autoroute (route *Sa*#1) et la seconde par une route départementale (route *Sa*#2). Le second scénario *Sb* est la traversée de Rennes du sud au nord (cf. figure 4.2), réalisable en passant par le centre-ville (route *Sb*#1) ou par le périphérique (route *Sb*#2). Les environnements de ces quatre routes sont mixtes avec une prédominance d'un type d'environnement en particulier (*Sa*#1, *Sa*#2 et *Sb*#1) ou une combinaison d'environnements (*Sb*#2). Le tableau 4.1 présente les caractéristiques de ces routes.

Pour étudier le comportement du gestionnaire d'énergie, nous prenons comme hypothèse que l'énergie disponible dans les véhicules n'est pas de 100%. En consi-

Listing 4.1 – Modélisation du véhicule urbain : `UrbanVehicle.orqa`.

```
import Common.orqa
import Drivechain.orqa
import Lighting.orqa
import StaticMiscellaneous.orqa
system {
  variables {
    const real m       := 1200
    const real eta_T   := 0.925
    const real k_T     := 1.0
    const real k_rr    := 0.008
    const real c_x     := 0.28
    const real A       := 2.25
    const real r       := 0.25
    const real Poff        := 0
    const real Pdaytime    := 50
    const real Plowbeam    := 200
    const real Phighbeam   := 250
    const real P_misc := 750
  }
  functions {
    real engine_efficiency(real torque, real speed) :
      interpolate3(
        [0,20,40,60,80,100,120,140,160],
        [-837.76, ... ,-104.72,-52.36,0,52.36,104.72, ...
           ,837.76],
        [0.4,0.46,0.42, ... ,0.9,0.9,0.9],
        torque,
        speed
      )
  }
  device Drivechain
  device Lighting
  device StaticMiscellaneous
}
```

Listing 4.2 – Modélisation du véhicule polyvalent : `PolyvalentVehicle.orqa`.

```
import Common.orqa
import Drivechain.orqa
import Lighting.orqa
import ClimateControl.orqa
import StaticMiscellaneous.orqa
system {
  variables {
    const real m       := 1700
    const real eta_T   := 0.925
    const real k_T     := 1.0
    const real k_rr    := 0.008
    const real c_x     := 0.27
    const real A       := 2.5
    const real r       := 0.3
    const real Poff        := 0
    const real Pdaytime    := 50
    const real Plowbeam    := 200
    const real Phighbeam   := 250
    const real k_eco := 0.5
    const real P_misc := 1500
  }
  functions {
    real engine_efficiency(real torque, real speed) :
      interpolate3(
        [0,50,100,150,200,250,300,350,400, ... ,650],
        [-305,-275,-245, ... ,-5,0,5, ... ,245,275,305],
        [0.905,0.905,0.905, ... ,0.76,0.74,0.72],
        torque,
        speed
      )
  }
  device Drivechain
  device Lighting
  device ClimateControl
  device StaticMiscellaneous
}
```

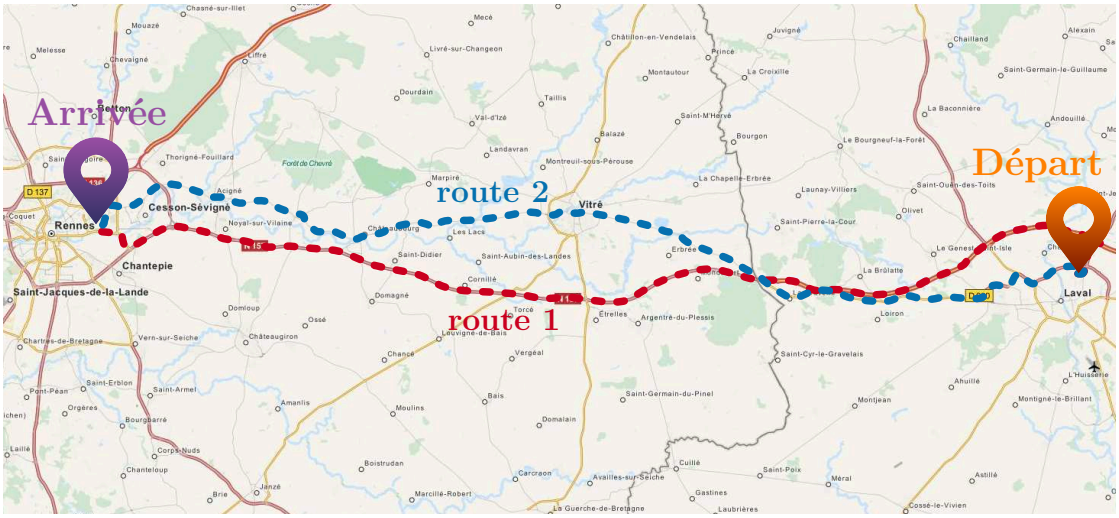


FIGURE 4.1 – Les deux routes du scénario *Sa*, de Laval à Rennes (© OpenStreetMap, MapQuest).

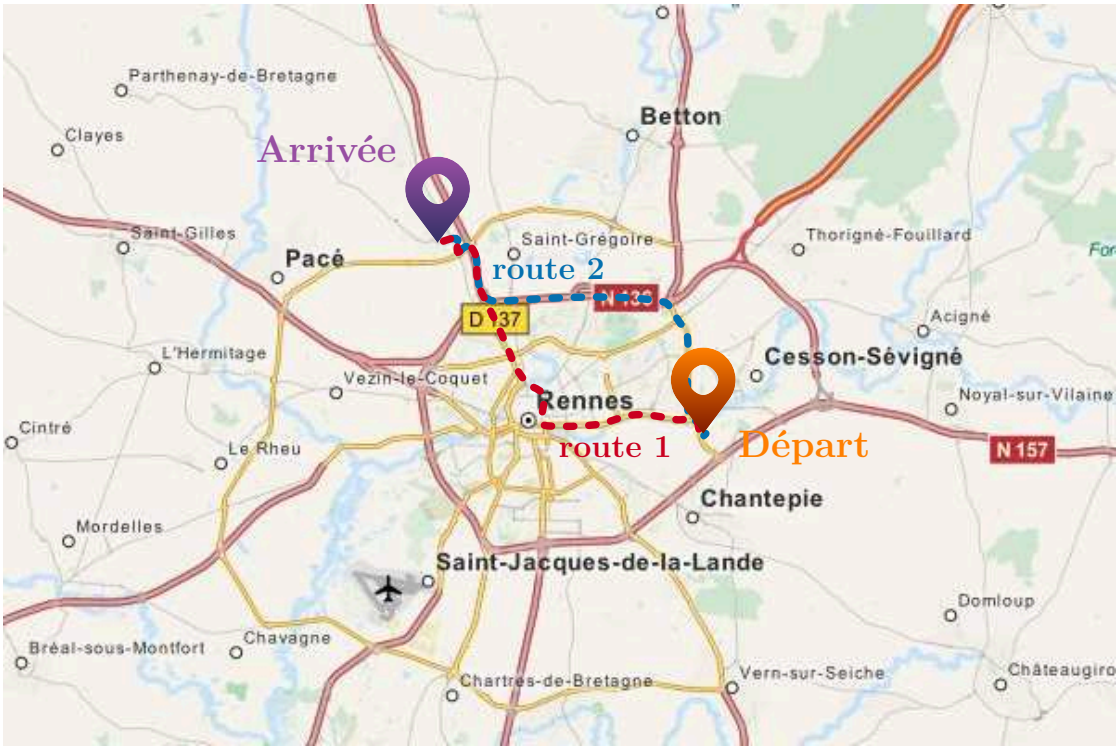


FIGURE 4.2 – Les deux routes du scénario *Sb*, la traversée de Rennes du sud au nord (© OpenStreetMap, MapQuest).

TABLE 4.2 – Politiques de consommation utilisées pour l'évaluation.

Politique	Poids durée (w_T)	énergie (w_E)	qualité (w_Q)
compromis	1/3	1/3	1/3
rapidité	80%	10%	10%
économie	10%	80%	10%
confort	10%	10%	80%

dérant un système de stockage d'énergie contenant 20 kilowattheures, nous proposons que la quantité disponible pour le scénario *Sa* soit de 75% (c.-à-d. 15 kilowattheures) et de 10% (c.-à-d. 2 kilowattheures) pour le scénario *Sb*. Nous verrons par la suite que ces quantités ne sont pas suffisantes pour réaliser les routes du scénario *Sb* de manière nominale. La première route du scénario *Sa* n'est pas non plus réalisable mais la deuxième l'est.

4.1.3 Les contraintes utilisateur

Les contraintes utilisateur sont définies par le conducteur et regroupent ses préférences d'utilisation ainsi que les politiques de consommation. Nous définissons maintenant les préférences du conducteur pour illustrer l'utilisation du gestionnaire d'énergie.

Les préférences d'utilisation L'utilisation du véhicule urbain ne peut être soumis qu'à une seule définition de préférence d'utilisation. En effet, ce véhicule ne possède qu'un seul organe contrôlable, la chaîne de traction, dont la préférence est alors de 100%. Le véhicule polyvalent possède deux organes contrôlables, la chaîne de traction et la climatisation. Nous définissons les préférences utilisateurs de sorte que la chaîne de traction soit autant favorisée que la climatisation : $u_{drivechain} = u_{climate} = \frac{1}{2}$.

Les politiques de consommation Nous définissons quatre politiques de consommation : *compromis*, *rapidité*, *économie* et *confort*. La politique de compromis répartit équitablement les poids entre les trois résultats durée, consommation et qualité. Les trois autres politiques appuient fortement sur un des résultats. Les poids des politiques de consommations sont présentés dans le tableau 4.2.

De plus, nous définissons une contrainte de durée pour l'ensemble des trajets et des scénarios $\chi_T = 2$. Ainsi, les solutions dont le temps de parcours est supérieur à deux fois le temps nominal sont ignorées.

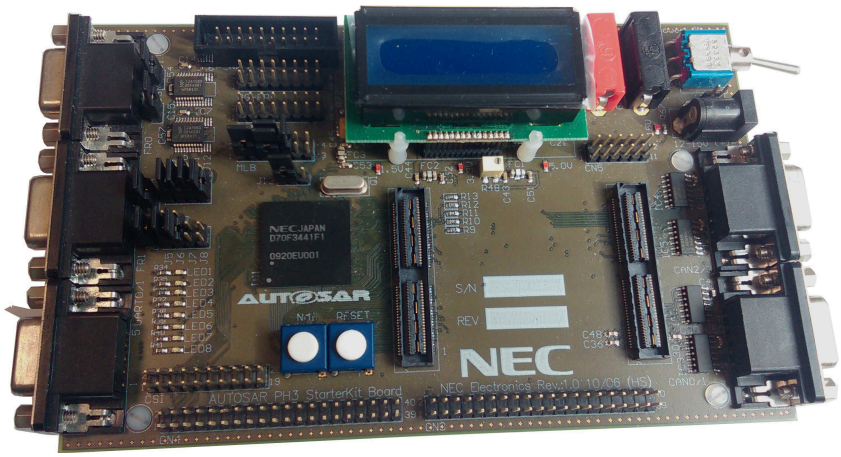


FIGURE 4.3 – La plate-forme d'évaluation NEC V850E/PHO3.

4.1.4 Plate-forme d'exécution

Une plate-forme d'exécution (un ECU) embarquée dans un véhicule propose typiquement entre 512 et 1024 kilooctets de mémoire morte (ROM) et entre 32 et 64 kilooctets de mémoire vive (RAM). Nous utilisons ici la plate-forme NEC™ V850E (cf. figure 4.3). C'est un kit d'évaluation fourni avec un système d'exploitation compatible AUTOSAR. Ses capacités sont recommandées pour assurer une application de direction assistées ou pour un contrôle de motorisation électrique. Elle embarque un processeur RISC 32 bits opérant à 128MHz, 768 kilooctets de mémoire morte et 60 kilooctets de mémoire vive. Le système d'exploitation temps réel exécuté sur cette plate-forme requiert environ 10 kilooctets de mémoire vive, la quantité disponible pour les applications est donc de l'ordre de 50 kilooctets.

4.1.5 Hypothèses et limitations

D'après la modélisation des routes telle que définie dans ORQA, les évènements extérieurs spontanés tels que le trafic ne sont pas considérés. En conséquence, nous ignorons tout évènement extérieur pouvant perturber le déroulement normal de la conduite lors de l'établissement des trajets. Les ralentissements dus aux ronds-points et feux tricolores sont pris en compte. Les ronds-points sont représentés par des points de passage à 20 km/h ou 30 km/h en fonction de leur taille et les feux tricolores par des points de passage à 50% de la vitesse environnante.

La modélisation des véhicules telle que décrite dans ce chapitre est réalisée à la conception. Nous passons maintenant à la configuration des gestionnaires d'énergie générés par ORQA pour les véhicules exemples. La phase de génération correspond à l'application de transformations de modèles sur les modèles des véhicules, nous ne la détaillons pas ici car elle reste générique (cf. section 3.3).

4.2 Configuration du gestionnaire d'énergie

La configuration du gestionnaire d'énergie est réalisée à la conception pour être ensuite embarquée dans le composant gestionnaire d'énergie. À l'exécution d'ORQA dans le véhicule, le gestionnaire d'énergie évalue les solutions accessibles en fonction de sa configuration.

4.2.1 Résultats typiques

Nous générons un ensemble de routes afin de constituer une base de données des résultats typiques pour chaque véhicule. Nous utilisons une majorité de routes urbaines pour le véhicule urbain et des routes différentes pour le véhicule polyvalent. Les répartitions que nous choisissons pour le véhicule urbain sont (respectivement pour le véhicule polyvalent) $\frac{2}{3}$ ($\frac{1}{3}$) de routes urbaines, $\frac{1}{6}$ ($\frac{1}{3}$) de routes rurales, $\frac{1}{6}$ ($\frac{1}{3}$) de voies rapides.

Nous générons deux bases de 60 routes, une par véhicule. La figure 4.4 et la figure 4.5 illustrent ces deux bases par les évolutions des différents ratios. Nous pouvons y voir que le ratio de durée croît de manière exponentielle pour de faibles coefficients de vitesse, et ce, pour les deux véhicules.

L'influence des accessoires consommateurs est clairement visible à faible vitesse où le véhicule urbain réalise le trajet beaucoup plus lentement. À ce moment là, la consommation de la chaîne de traction devient un consommateur moins important sur la durée car sa puissance moyenne requise est plus faible que celle des autres équipements. D'où une pente ascendante du ratio de consommation à faible vitesse du véhicule.

Nous décidons de réduire le domaine de coefficients de vitesse étudié à l'intervalle $[40\%; 100\%]$. Ainsi, nous nous concentrons sur les solutions dont la durée de réalisation sera au maximum d'environ deux fois et demie la durée nominale comme cela est visible sur les parties agrandies des figures 4.4 et 4.5. Dans la suite de cette partie, nous utilisons ce domaine réduit comme base pour appliquer les approches d'optimisation.

4.2.2 Application des optimisations de la configuration du gestionnaire d'énergie

Nous avons produit les résultats typiques pour les deux véhicules que nous étudions pour l'ensemble des routes. Nous nous intéressons désormais à y appliquer les optimisations présentées à la section 2.4.

Nous avons proposé deux approches pour optimiser la configuration du gestionnaire d'énergie et simplifier la recherche de stratégie de conduite. La première

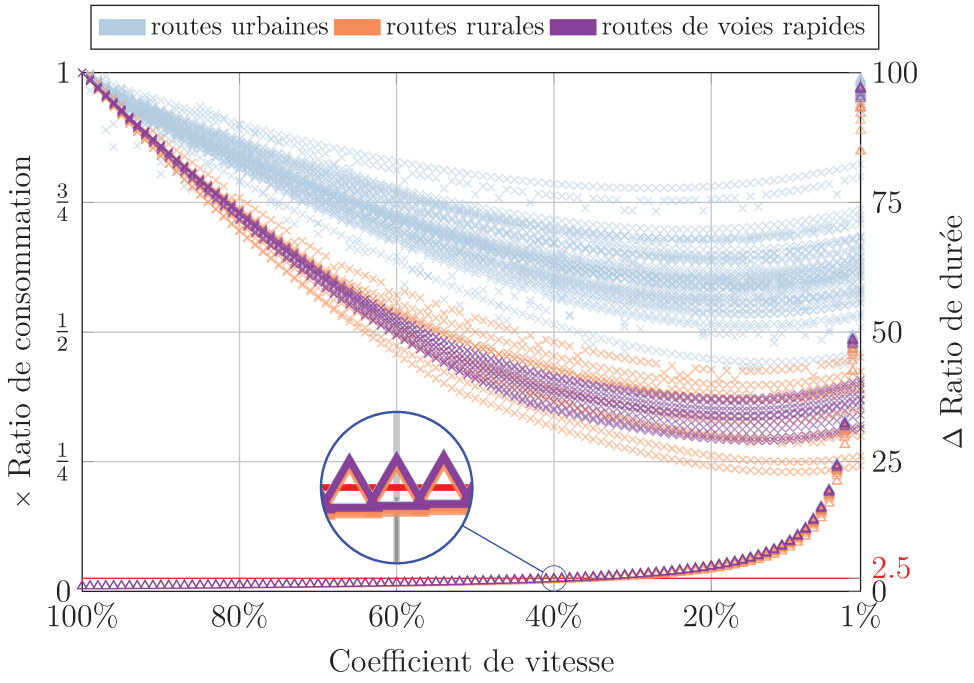


FIGURE 4.4 – Évolution des ratios des résultats de durée et de consommation du véhicule urbain.

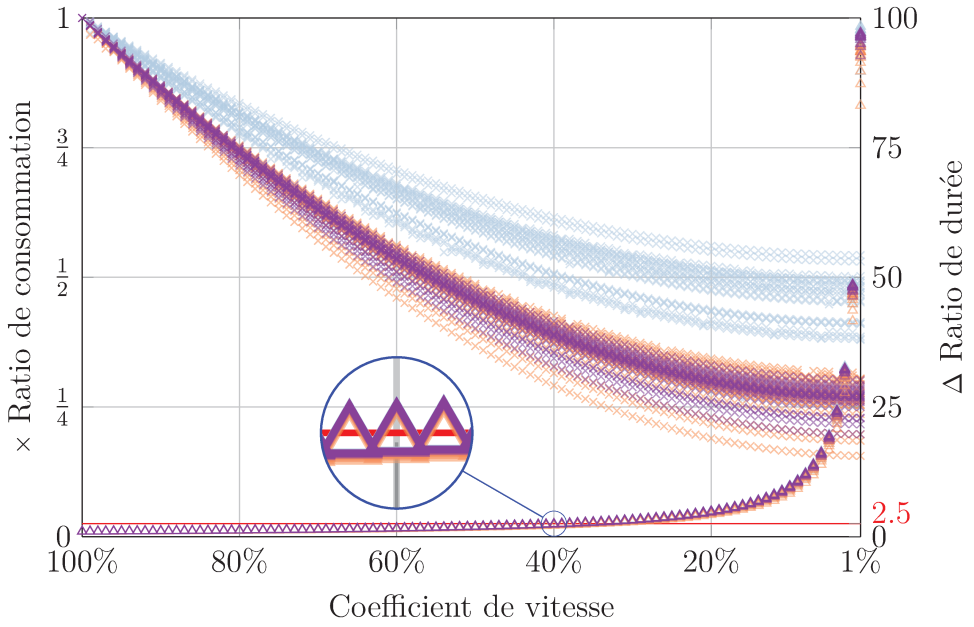


FIGURE 4.5 – Évolution des ratios des résultats de durée et de consommation du véhicule polyvalent.

TABLE 4.3 – Résultats de l'approche de partitionnement pour le véhicule urbain : nombre de coefficients, écarts type moyens et coûts.

Config. (# coef.)	Écart type moy. (durée / conso.)	Coût (θ / ROM / RAM)
routes urbaines (13) routes rurales (17) voies rapides (13)	2% / 1% 1% / 2% 2% / 2%	$15 \pm 2 \times \theta_{eval}$ / 26ko / 139ko
tous env. (12)	2% / 2%	$12 \times \theta_{eval}$ / 26ko / 98ko
tous env. (61)	-	$61 \times \theta_{eval}$ / 26ko / 489ko

(approche de partitionnement) est la réduction de l'espace de solution par un regroupement des coefficients de vitesse, réduisant ainsi leur nombre. La seconde (approche d'approximation) est une approximation de l'évolution des résultats permettant ainsi, à partir des résultats de la route nominale, d'approximer les résultats de ses variations.

Les approches d'optimisation sont définies sur un ensemble de routes, et ces routes sont naturellement groupées selon leur environnement. Ainsi, nous appliquons les approches d'optimisation sur quatre groupes de routes : urbaines, rurales, voies rapides, et toutes les routes (urbaines, rurales et voies rapides). Nous obtenons alors deux configurations possibles, l'une où chaque environnement est indépendant (résultat des trois premiers groupes) et l'autre où tous les environnements sont regroupés (c'est à dire que l'environnement n'est pas caractérisant). À ces deux configurations optimisées s'ajoute la configuration initiale. Nous choisissons ensuite une configuration parmi les trois possibles en nous basant sur les valeurs retournées par les outils d'ORQA, telles que l'erreur induite par une configuration, et sur les ressources limitées de la plate-forme cible.

4.2.2.1 Application de l'approche de partitionnement

L'application de l'approche de partitionnement aux deux véhicules est résumée par les tableaux 4.3 et 4.4. La configuration de base est la même pour les deux véhicules et définit les 61 coefficients de vitesse compris entre 40% et 100%.

La configuration optimisée pour le véhicule urbain définit entre 13 et 17 coefficients de vitesse si les environnements sont considérés séparément, et 12 coefficients en les regroupant : {42, 49, 56, 63, 69, 77, 83, 89, 97}.

La configuration optimisée pour le véhicule polyvalent définit entre 11 et 13 coefficients de vitesse pour les environnements séparés, et 13 coefficients en considérant les environnements ensemble : {42, 46, 50, 54, 59, 65, 69, 74, 81, 86, 91, 94, 99}.

TABLE 4.4 – Résultats de l'approche de partitionnement pour le véhicule polyvalent : nombre de coefficients, écarts type moyens et coûts.

Config. (# coef.)	Écart type moy. (durée / conso.)	Coût (θ / ROM / RAM)
routes urbaines (11) routes rurales (13) voies rapides (12)	2% / 2% 2% / 2% 2% / 2%	$12 \pm 1 \times \theta_{eval}$ / 26ko / 107ko
tous env. (13)	2% / 2%	$13 \times \theta_{eval}$ / 26ko / 107ko
tous env. (61)	-	$61 \times \theta_{eval}$ / 26ko / 502ko

Un partitionnement des résultats par environnement est plus précis qu'un partitionnement général, mais dans cet exemple, la différence d'écart type moyen entre les deux partitionnements est faible : au plus de 1%. Les coûts mémoire des configurations du véhicule urbain varient fortement dû aux différents nombres de coefficients de vitesse. Comme les deux configurations potentielles du véhicule polyvalent ont le même nombre de coefficients maximal (13), les coûts mémoires sont identiques. L'effet de l'approche de partitionnement sur la configuration est également visible sur la complexité de calcul des solutions, qui dépend elle aussi du nombre de coefficients de vitesse.

Toutes les configurations étudiées, malgré la réduction importante du nombre de coefficient de vitesse, ont un coût mémoire trop important pour notre plateforme cible (50ko de RAM disponible). Dans cet exemple, l'approche de partitionnement n'est donc pas utilisable seule.

4.2.2.2 Application de l'approche d'approximation

L'application de l'approche d'approximation aux deux véhicules exemples est résumée par les tableaux 4.5 et 4.6. La configuration de départ est la même pour les deux véhicules et définit les 61 coefficients de vitesse compris entre 40% et 100%.

La deuxième approche permet, au contraire de l'approche de partitionnement, la définition de configurations viables en terme de mémoire requise (ROM et RAM). Les deux configurations optimisées par l'approche d'approximation requièrent, à quelques dizaines d'octets près, la même quantité de mémoire.

La première configuration réalise des approximations sur des ensembles de routes similaires. Il est logique qu'elle produise une erreur moins importante que la seconde configuration plus générale.

Avec cette approche, la configuration optimisée du véhicule urbain tout envi-

TABLE 4.5 – Résultats de l'approche d'approximation pour le véhicule urbain : écarts type moyens et coûts.

Config.	Écart type moy. durée / conso.	Coût (θ / ROM / RAM)
routes urbaines routes rurales voies rapides	1% / 3% 3% / 3% 1% / 1%	$1 \times \theta_{eval} + 60 \times \theta_{approx}$ / 29ko / 9ko
tous env.	2% / 8%	$1 \times \theta_{eval} + 60 \times \theta_{approx}$ / 29ko / 9ko
tous env.	-	$61 \times \theta_{eval}$ / 26ko / 489ko

TABLE 4.6 – Résultats de l'approche d'approximation pour le véhicule polyvalent : écarts type moyens et coûts.

Config.	Écart type moy. durée / conso.	Coût (θ / ROM / RAM)
routes urbaines routes rurales voies rapides	2% / 3% 2% / 2% 1% / 1%	$1 \times \theta_{eval} + 60 \times \theta_{approx}$ / 29ko / 10ko
tous env.	2% / 5%	$1 \times \theta_{eval} + 60 \times \theta_{approx}$ / 29ko / 10ko
tous env.	-	$61 \times \theta_{eval}$ / 26ko / 502ko

ronnement confondu a un écart type des résultats de durée compris entre +1% et +2% plus important que la configuration distinguant les environnements. La différence entre ces deux configurations est plus importante sur les résultats de consommation avec une augmentation entre +5% et +7% des écarts type.

Les différences des deux configurations optimisées du véhicule polyvalent sont du même ordre. Les écarts type des résultats de durée augmentent au maximum de +1% et ceux de consommation augmentent entre +2% et +4%.

Comme ces ratios sont utilisés pour évaluer des routes, nous transposons leur erreur aux résultats bruts. Par exemple, sur un long trajet d'une heure et de 10kWh, un écart type de 5% correspond à une erreur type de 3 minutes sur l'estimation de durée. Au niveau de la consommation, un écart type de 5% (500Wh) n'est pas négligeable. Ainsi, nous sélectionnons la configuration distinguant les environnements pour contenir l'erreur d'approximation.

TABLE 4.7 – Résultats de la composition des deux approches (partitionnement puis approximation) pour le véhicule urbain : nombre de coefficients, écarts type moyens et coûts.

Config.	Écart type moy. (durée / conso.)	Coût (θ / ROM / RAM)
routes urbaines routes rurales voies rapides	1% / 3% 2% / 3% 1% / 1%	$1 \times \theta_{eval} + 15 \pm 2 \times \theta_{approx}$ / 27ko / 8ko
tous env.	2% / 8%	$1 \times \theta_{eval} + 12 \times \theta_{approx}$ / 27ko / 8ko
tous env.	-	$61 \times \theta_{eval}$ / 26ko / 489ko

TABLE 4.8 – Résultats de la composition des deux approches (partitionnement puis approximation) pour le véhicule polyvalent : nombre de coefficients, écarts type moyens et coûts.

Config.	Écart type moy. (durée / conso.)	Coût (θ / ROM / RAM)
routes urbaines routes rurales voies rapides	2% / 3% 3% / 2% 1% / 1%	$1 \times \theta_{eval} + 12 \pm 1 \times \theta_{approx}$ / 27ko / 9ko
tous env.	2% / 5%	$1 \times \theta_{eval} + 13 \times \theta_{approx}$ / 27ko / 9ko
tous env.	-	$61 \times \theta_{eval}$ / 26ko / 502ko

4.2.2.3 Application de la composition des deux approches

La composition des deux approches d'optimisation est maintenant étudiée. Les deux ordres de composition produisent des résultats similaires.

L'application de l'approche par partitionnement puis de l'approche d'approximation aux deux véhicules est résumée par les tableaux 4.7 et 4.8. Cet ordre d'application produit une configuration où les coefficients choisis sont ceux issus de l'approche par partitionnement. Au lieu d'évaluer systématiquement les résultats pour chaque coefficient, l'approche d'approximation est appliquée et permet d'obtenir des ratios approximés. Nous pouvons constater que les erreurs induites par cette composition sont proches de celles induites par l'approche d'approximation, tout en étant globalement légèrement supérieures. Ces deux approches sont moins précises que l'approche par partitionnement mais, dans les deux cas, elles permettent de réduire considérablement les coûts mémoires.

La composition de l'approche d'approximation puis de l'approche par partitionnement produit des résultats équivalent à la première composition. Les erreurs induites sont néanmoins légèrement supérieures, avec une hausse d'erreur moyenne comprise entre 0 et 1%. Les résultats de cette composition sont présentés sous forme de tableaux à l'annexe A.

4.2.2.4 Conclusion

La configuration initiale du gestionnaire d'énergie empêche son utilisation sur la plate-forme d'exécution ciblée en raison de coûts mémoire trop importants. Pour les deux véhicules, la première approche d'optimisation à base de partitionnement de coefficients de vitesse ne permet pas la définition d'une configuration viable vis à vis des contraintes de la plate-forme. En revanche, la deuxième approche fournissant des approximations de ratios permet d'embarquer le gestionnaire d'énergie grâce à une configuration moins gourmande. La condition est, dans notre exemple, d'accepter d'avoir des résultats (les stratégies de conduite) moins précis de l'ordre de 3% pour le véhicule urbain et 2% pour le véhicule polyvalent. Enfin, la composition de ces deux approches offre des configurations équivalentes à celles de la deuxième approche, tant au niveau des ressources requises que des erreurs d'approximation.

Nous choisissons d'utiliser la deuxième approche pour définir les configurations des deux gestionnaires d'énergie. Nous évaluons maintenant les gestionnaires d'énergie des véhicules étudiés pour les deux scénarios.

4.3 Application aux scénarios

Nous appliquons maintenant l'utilisation d'ORQA et du gestionnaire d'énergie aux scénarios présentés. Par le biais des deux scénarios, nous simulons une utilisation du gestionnaire d'énergie par le conducteur.

Nous détaillons d'abord les résultats nominaux obtenus sans gestionnaire d'énergie pour les deux scénarios étudiés. Nous appliquons ensuite les scénarios aux véhicules exemples en utilisant les configurations des gestionnaires d'énergie optimisées pour chacun des véhicules. Quatre politiques de consommation *compromis*, *rapidité*, *économie* et *confort* sont utilisées pour faire varier les préférences utilisateurs.

4.3.1 Résultats nominaux

Nous calculons tout d'abord les résultats nominaux tels qu'obtenus sans gestionnaire d'énergie. À partir des modèles des deux véhicules définis précédemment, nous exécutons le calcul des résultats nominaux de durée et de consommation pour

TABLE 4.9 – Résultats nominaux du véhicule urbain.

Route	Durée	Consommation	Qualité
Sa#1	43min 51s	16 801Wh	100%
Sa#2	1h 09min 30s	14 030Wh	100%
Sb#1	14min 48s	2 573Wh	100%
Sb#2	11min 54s	2 549Wh	100%

TABLE 4.10 – Résultats nominaux du véhicule polyvalent.

Route	Durée	Consommation	Qualité
Sa#1	43min 51s	15 634Wh	100%
Sa#2	1h 09min 30s	14 385Wh	100%
Sb#1	14min 48s	2 821Wh	100%
Sb#2	11min 54s	2 578Wh	100%

chacune des routes des scénarios en considérant une qualité maximale rendue par le véhicule. Ces résultats sont mis en forme dans les tableaux 4.9 et 4.10. Comme ces trajets sont réalisés de manière nominale, la qualité du service rendue est maximale (100%).

Nous constatons que les durées de réalisation d'une route sont identiques pour les deux véhicules. Ceci s'explique simplement par la méthode de réalisation d'un trajet définie dans ORQA (cf. section 2.3). Les étapes d'un trajet sont réalisées selon une accélération du véhicule fixe. Ainsi, les caractéristiques de la chaîne de traction telles que ses limitations de couple et de vitesse ne sont pas prises en compte pour la réalisation du trajet. Au final, la réalisation d'un trajet – et donc la durée nécessaire à chaque étape – est commune à tous les véhicules et est indépendante des limitations des organes réels.

Nous constatons également que les consommations des véhicules sont différentes selon les types d'environnement de manière cohérente avec les véhicules. Ainsi, nous observons que le véhicule urbain consomme plus d'un kilowattheure supplémentaire que le véhicule polyvalent (+7%) sur la route *Sa*#1 qui est majoritairement à haute vitesse. Et réciproquement pour le trajet urbain *Sb*#1, le véhicule urbain requiert moins d'énergie (-9%) que le véhicule polyvalent.

Nous avons défini que le scénario *Sa* devait être réalisé avec une quantité d'énergie de 15 kilowattheures et le scénario *Sb* avec 2 kilowattheures. Sans gestionnaire d'énergie, la première route du scénario *Sa* n'est pas réalisable alors que la deuxième route, plus longue, est réalisable. Le scénario *Sb* n'est pas du tout réalisable, ni par le véhicule urbain ni par le véhicule polyvalent, car les deux véhicules requièrent plus de 2 500 wattheures peu importe la route empruntée.

4.3.2 Évaluation du gestionnaire d'énergie

Nous étudions maintenant les résultats proposés par le gestionnaire d'énergie d'ORQA dans différents cas. Les stratégies de conduite proposées par ORQA sont comparées aux stratégies de conduite nominales.

Les figures 4.6 et 4.7 présentent les résultats des stratégies de conduite nominales et celles proposées par ORQA pour le véhicule polyvalent dans le cas du scénario *Sa* et pour le véhicule urbain dans le cas du scénario *Sb*. Les résultats détaillés de ces scénarios pour les deux véhicules sont regroupés dans l'annexe B. Cette dernière contient également les solutions et résultats proposés par ORQA pour le véhicule urbain dans le scénario *Sa* et pour le véhicule polyvalent dans le scénario *Sb*.

4.3.2.1 Réalisation du scénario *Sa* par le véhicule polyvalent

Il est important de noter que le scénario *Sa* est réalisable sans gestion d'énergie par la deuxième route. Les solutions proposées avec les politiques de compromis et de rapidité sont meilleures au point de vue de la durée et de la consommation mais perdent en qualité. La politique de compromis garde l'utilisation nominale des équipements mais, étant donné qu'elle propose d'utiliser un coefficient de vitesse de 88%, la qualité de la solution est moindre car, dans notre modèle, le coefficient influe sur la qualité de service de la chaîne de traction. Cette politique définit la réalisation du scénario par la première route en 49 minutes et 30 secondes pour une consommation de 13,2 kilowattheures et une qualité de 88%. En somme, sa solution économise 20 minutes et plus d'un kilowattheure par rapport à la solution réalisable de manière nominale. En revanche, la politique de rapidité propose une solution certes plus rapide, mais également plus économique et moins qualitative en désactivant la climatisation. Sa réalisation requiert 44 minutes et 14 secondes pour 12,3 kilowattheures. La qualité estimée est de 50%.

La politique d'économie propose une solution extrême au regard du positionnement des solutions sur la figure 4.6. Cette réalisation lente de la première route est estimée à une durée de 1 heure et 27 minutes pour 5 580 wattheures. La qualité estimée de 24% est due au coefficient de vitesse faible (48%) et à l'extinction de la climatisation. Cette solution permet d'économiser presque les deux tiers de l'énergie requise pour sa réalisation nominale en doublant le temps de trajet et en diminuant drastiquement le confort utilisateur.

Enfin, la politique de confort mène à la réalisation de la seconde route du scénario avec un coefficient de vitesse de 95%. Elle est estimée à une durée de 1 heure et 13 minutes pour 13 298 wattheures et une qualité estimée de 95%. Le confort est conservé avec la combinaison d'organes nominale et une vitesse assez élevée (coefficient de vitesse de 95%). Cette solution est une alternative équivalente

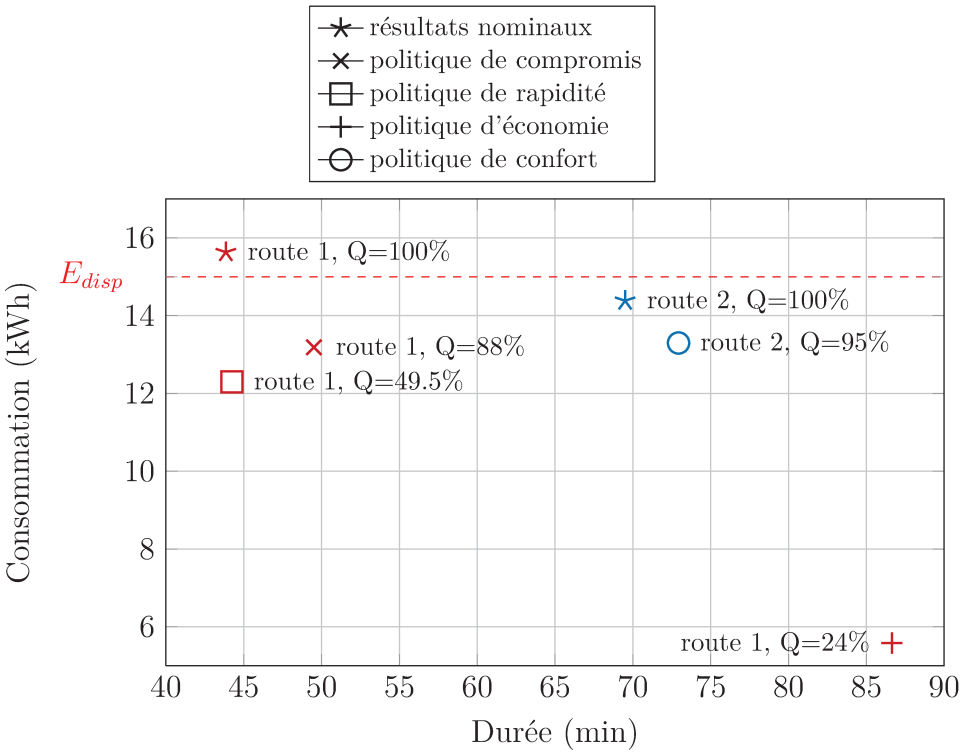


FIGURE 4.6 – Résultats des stratégies de conduite optimales du véhicule polyvalent pour le scénario *Sa*.

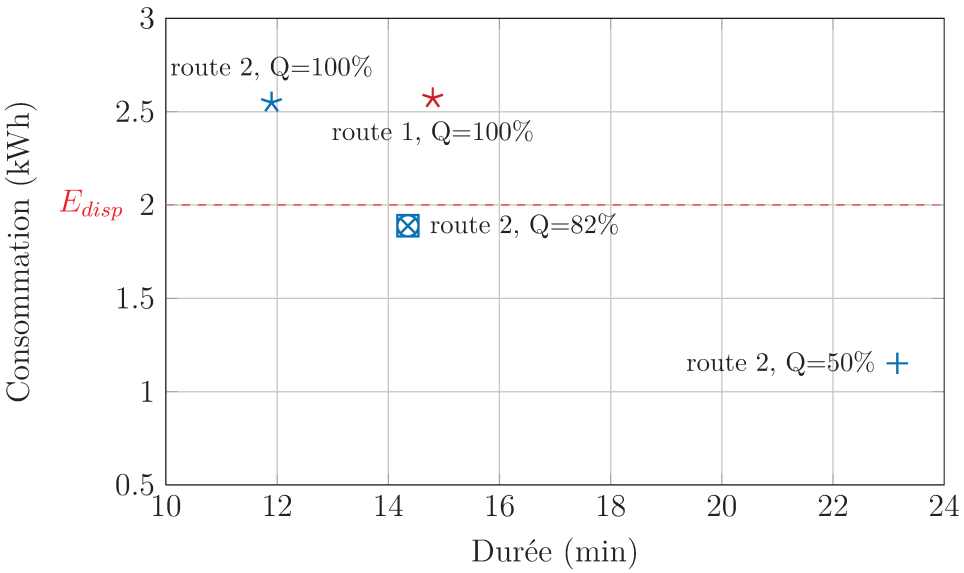


FIGURE 4.7 – Résultats des stratégies de conduite optimales du véhicule urbain pour le scénario *Sb*.

à la réalisation nominale de la deuxième route. Elle nécessite moins d'énergie mais dure plus longtemps.

4.3.2.2 Réalisation du scénario *Sb* par le véhicule urbain

Les politiques de consommation de compromis, de rapidité et de confort mènent à la même solution : l'application du coefficient de vitesse de 82%. La combinaison d'organes des solutions proposées est commune à toutes les politiques de consommation car il n'y a pas d'alternative possible. La politique d'économie mène à l'application du coefficient de vitesse de 50%.

Nous constatons premièrement que le trajet est réalisable à l'aide du gestionnaire d'énergie alors qu'aucune des solutions nominales n'est satisfaisante. De plus, les quatre politiques de consommation utilisent la même route ($Sb\#2$). Dans le cas de la solution commune à trois des politiques, le trajet est estimé à une durée de 14 minutes et 21 secondes pour une consommation énergétique de 1 888 wattheures. C'est à dire qu'en diminuant la vitesse pour la deuxième route, il est possible de réaliser le trajet tout en étant plus rapide que le trajet nominal de la première route (27 secondes de moins).

Tout comme pour le scénario Sa, la solution proposée avec la politique d'économie réalise le trajet de manière lente mais requiert moins d'énergie que par une réalisation nominale. En utilisant au maximum les contraintes utilisateur, la politique d'économie propose un trajet presque deux fois plus lent que de manière nominale mais offre une alternative peu consommatrice d'énergie (moins de 1 200 wattheures, soit moins de la moitié de la réalisation nominale).

4.3.3 Conclusion

Le scénario Sa est réalisable de manière nominale uniquement par sa deuxième route. Le gestionnaire d'énergie du véhicule polyvalent propose deux solutions plus intéressantes sur les plans temporel et énergétique. Les solutions des politiques d'économie et de confort ne sont pas clairement avantageuses par rapport à la réalisation nominale de la route 2 mais se positionnent en alternatives pour des objectifs différents. Les valeurs estimées avec la configuration optimisée du gestionnaire d'énergie sont globalement justes en comparaison des évaluations à l'exception de celles de la politique de rapidité (cf. annexe B). La consommation énergétique évaluée présente un écart de +1 837 wattheures, soit une sous-estimation de 15% de l'énergie requise. En prenant en compte cette erreur d'approximation, la solution proposée est toujours réalisable avec un besoin énergétique de 14 134 wattheures.

Sans gestionnaire d'énergie, le scénario Sb n'est pas réalisable. C'est à dire que les deux routes proposées requièrent plus d'énergie que disponible dans le cas d'une réalisation nominale. Les stratégies de conduite proposées par le gestionnaire

d'énergie d'ORQA permettent la réalisation du scénario. Différentes réalisations sont proposées *via* les politiques de consommation : un compromis entre la durée, la consommation et la qualité ; un trajet rapide ; un trajet économique ; et un trajet confortable. Les valeurs estimées avec la configuration optimisée du gestionnaire d'énergie sont proches de celles calculées systématiquement avec une erreur maximale d'approximation de 1% (cf. annexe B).

4.4 Conclusion

Ce chapitre a présenté l'évaluation du canevas logiciel ORQA. Nous avons tout d'abord illustrer son utilisation à travers deux véhicules exemples, l'un urbain et l'autre polyvalent. Ces véhicules ont été modélisés avec le langage dédié textuel d'ORQA et en utilisant les modèles génériques de la bibliothèque. La génération des gestionnaires d'énergie n'a pas été abordée de nouveau car elle est générique et ne dépend pas des modèles de caractérisation. Nous avons également présenté une plate-forme d'exécution cible afin d'avoir des points de repères en terme de ressources disponibles.

Une base de données par véhicule a été constituée à partir de routes générées. La répartition des environnements des route générées a été décidée en fonction de l'usage du véhicule : plus de routes urbaines pour le véhicule urbain et une répartition équitable pour le véhicule polyvalent. Ainsi, les résultats obtenus pour chaque base sont supposés être plus proches de l'usage final du véhicule (plus « typiques »). L'étude de ces bases de résultats nous a permis d'établir une configuration dont le domaine des coefficients de vitesse est réduit à 61 éléments. Nous avons ensuite appliqué les approches d'optimisations sur les bases de résultats. Finalement, en raison des capacités de la plate-forme d'exécution, nous avons sélectionné les configurations fournies par l'approche d'approximation pour les deux véhicules.

Les deux scénarios étudiés ont permis d'évaluer les résultats proposés pour les véhicules dans différents environnements. La seconde route du scénario Sa permettait une réalisation nominale de ce dernier, mais pas la première route. Sans gestionnaire d'énergie, le scénario Sb n'était pas du tout réalisable de manière nominale. Le gestionnaire d'énergie d'ORQA a identifié plusieurs solutions viables selon des politiques de consommation différentes. Dans le cas du scénario Sa, le gestionnaire d'énergie a proposé deux nouvelles solutions plus rapides et plus économes que la solution nominale de la seconde route. Dans le cas du scénario Sb, il a permis de réaliser un scénario qui ne l'était pas de manière nominale. Enfin, les résultats estimés par le gestionnaire sont globalement proches des valeurs évaluées.

L'application de la méthodologie d'ORQA sur deux véhicules exemples a apporté les connaissances énergétiques et qualitatives à leur systèmes logiciels. Le

gestionnaire d'énergie intégré à l'architecture et aux composants logiciels a permis de proposer des solutions, en ligne, pour deux scénarios. Ces solutions ont permis la réalisation, ou l'amélioration, des scénarios.

Conclusion

Cette thèse s'intéresse à optimiser la gestion de la ressource énergétique au sein des véhicules électriques du point de vue utilisateur. L'approche doit respecter les attentes des utilisateurs – c'est à dire la qualité attendue – tout en assurant une gestion de l'énergie adéquate. Pour répondre à ce besoin, nous avons proposé le canevas logiciel ORQA permettant d'enrichir les modèles existants des systèmes embarqués avec les connaissances énergétiques et qualitatives des organes du véhicule. À partir de ces informations, un gestionnaire d'énergie est généré. Il permet au conducteur de décider de la politique de consommation et de l'utilisation des équipements qu'il souhaite obtenir pour un trajet.

Nous avons dans un premier temps défini un métamodèle de caractérisation dédié aux organes du véhicules. Ce métamodèle concerne les besoins énergétiques et les niveaux de qualité de service de chaque organe consommateur, toutes deux liées au modes de fonctionnement de ces organes. Les concepteurs manipulent les modèles de caractérisation à l'aide d'un langage textuel de modélisation dédié à ORQA. Pour simplifier cette tâche, ORQA fournit en plus une bibliothèque de modèles génériques des principaux organes consommateurs : chaîne de traction, climatisation, système d'éclairage, et autres consommateurs.

Le rôle du gestionnaire d'énergie est de trouver la solution optimale au trajet voulu par le conducteur. À cette fin, il étudie les routes accessibles ainsi que différentes stratégies de conduite en réduisant leurs vitesses. Puis, il propose une solution en fonction de la politique de consommation du conducteur et d'éventuelles contraintes de durée et d'énergie à conserver. Le gestionnaire d'énergie assure la réalisation de la solution choisie en intégrant un contrôle des organes du véhicule définis comme non critiques par les concepteurs pendant la caractérisation. Les préférences d'utilisation sont déterminées par le conducteur et suivies par le gestionnaire d'énergie. Ce dernier permet ainsi d'optimiser l'utilisation du véhicule selon les souhaits du conducteur.

Le gestionnaire d'énergie est exécuté sur une plate-forme (un calculateur) des systèmes embarqués qui possède des ressources, par définition, limitées. La configuration par défaut du gestionnaire d'énergie requiert une grande quantité de mémoire vive de l'ordre d'un demi mégaoctet, environ dix fois plus que celle dis-

ponible sur une plate-forme d'exécution moyenne. Pour répondre à ce problème, nous avons proposé deux approches d'optimisation de la configuration du gestionnaire d'énergie à la conception, toutes les deux sont basées sur une étude de résultats typiques du véhicule. La première approche permet de diminuer l'espace de solution exploré en ligne par le gestionnaire d'énergie. La seconde est la simplification de l'évaluation des solutions explorées. Elle permet également de diminuer la complexité de la recherche de solution.

Enfin, un cas d'étude sur deux véhicules a été présenté. Nous y avons comparé les résultats obtenus avec et sans gestionnaire d'énergie. Plusieurs configurations des gestionnaires d'énergie ont également été comparées et leurs performances illustrées.

Perspectives

Les travaux présentés dans cette thèse ouvrent la voie à de nombreuses perspectives que nous présentons maintenant.

ORQA est focalisé sur les organes consommateurs d'un véhicule électrique afin d'optimiser leur utilisation. La source d'énergie d'un tel véhicule est totalement abstraite dans cette approche. Or, il existe différentes technologies de stockage d'énergie et chacune d'elles a ses propres caractéristiques. Par exemple, les batteries conventionnelles ne permettent pas un échange massif d'énergie (dans un sens ou dans l'autre) dans une courte durée mais les super-capacités le peuvent. De plus, pour optimiser la durée de vie de ces batteries et leur performance sur le long terme, la demande énergétique doit être lissée le plus possible. Des hybridations batteries conventionnelles–super-capacités sont mises au point dans ce but précis [MRBL14]. Le gestionnaire d'énergie d'ORQA s'appuie sur un modèle simplifié de la source d'énergie du véhicule. S'il avait conscience de la ou les technologies de stockage d'énergie et des utilisations les plus appropriées (optimales), alors l'usage des consommateurs du véhicule serait optimisé en même temps que les performances du système de stockage. Cet objectif pourrait être atteint en étendant le métamodèle de caractérisation pour prendre en considération les éléments *producteurs* d'énergie qu'il faudrait également prendre en compte dans l'algorithme de choix de la solution.

La variation des routes potentielles utilise des coefficients de vitesse fixes. Il n'est peut être pas toujours judicieux de diminuer la vitesse de la même manière tout au long d'un trajet. Prenons l'exemple d'un trajet combinant une partie urbaine et une partie sur voie rapide. Nous pouvons imaginer que réduire sa vitesse de 50km/h à 30km/h (coefficient de vitesse de $\frac{3}{5}$) est réalisable aisément mais passer de 130km/h à 78km/h le paraît moins, sauf en cas de nécessité. Une piste serait de pouvoir utiliser différents coefficients de vitesse pour un même trajet. Comment

choisir ces coefficients est une question ouverte : de manière stochastique ? selon des probabilités définies à l'avance ? Nous pourrions également nous baser sur l'historique des trajets et du ressenti du conducteur pour définir de nouvelles méthodes de variation de conduite. ORQA se rapprocherait encore plus d'une utilisation du véhicule orientée utilisateur en se basant sur son expérience même.

Comme nous l'avons abordé dans l'introduction, la communication entre les véhicules et l'infrastructure environnante prend de l'ampleur. La recherche de solution à un trajet pourrait tirer partie d'informations contextuelles ou de conseils d'itinéraires pour, par exemple, désengorger certains parcours. Ainsi les coefficients de vitesse pourraient aussi évoluer selon l'environnement du véhicule. Enfin, d'autres axes abordant le partage d'informations à grande échelle sont la sensibilisation des réseaux électriques (*smart grid*s) [Far10] et des bâtiments (*smart home*s [Ald03] et *smart building*s). Dans cette optique, le véhicule électrique peut agir comme un tampon énergétique chez l'habitant [MVH+10]. Par exemple comme source énergétique d'appoint ou de manière plus régulière en reportant la demande énergétique du foyer aux heures pleines vers les heures creuses. Ceci soulagerait le réseau électrique et permettrait d'assurer son fonctionnement tout en diminuant le coût final de la facture énergétique. En ayant connaissance de tels objectifs, le gestionnaire d'énergie pourrait de lui-même proposer des contraintes sur la consommation d'un trajet au conducteur en prévoyant la future utilisation du véhicule.

Annexe A

Résultats de la composition des approches d'optimisation

Cette annexe présente les résultats de la composition des deux approches d'optimisation pour les deux véhicules exemples. L'approche d'approximation est premièrement appliquée puis suivie de l'approche par partitionnement (cf. chapitre 4, page 87). Le tableau A.1 résume les résultats de cette application pour le véhicule urbain et le tableau A.2 pour le véhicule polyvalent.

TABLE A.1 – Résultats de la composition des deux approches (approximation puis partitionnement) pour le véhicule urbain : nombre de coefficients, écarts type moyens et coûts.

Config. (# coef.)	Écart type moy. (durée / conso.)	Coût (θ / ROM / RAM)
routes urbaines (11) routes rurales (12) voies rapides (13)	1% / 3% 3% / 3% 1% / 1%	$1 \times \theta_{eval} + 12 \pm 1 \times \theta_{approx}$ / 27ko / 8ko
tous env. (11)	3% / 8%	$1 \times \theta_{eval} + 11 \times \theta_{approx}$ / 27ko / 8ko
tous env. (61)	-	$61 \times \theta_{eval}$ / 26ko / 489ko

TABLE A.2 – Résultats de la composition des deux approches (approximation puis partitionnement) pour le véhicule polyvalent : nombre de coefficients, écarts type moyens et coûts.

Config. (# coef.)	Écart type moy. (durée / conso.)	Coût (θ / ROM / RAM)
routes urbaines (11) routes rurales (10) voies rapides (11)	2% / 3% 3% / 2% 1% / 1%	$1 \times \theta_{eval} + 11 \pm 1 \times \theta_{approx}$ / 27ko / 9ko
tous env. (12)	2% / 5%	$1 \times \theta_{eval} + 12 \times \theta_{approx}$ / 27ko / 9ko
tous env. (61)	-	$61 \times \theta_{eval}$ / 26ko / 502ko

Annexe B

Résultats des solutions aux scénarios Sa et Sb

Cette annexe présente les résultats complets des stratégies de conduite proposées par le gestionnaire d'énergie d'ORQA pour les deux scénarios étudiés à la validation (cf. chapitre 4, page 87). Les solutions du véhicule urbain pour le scénario Sa sont illustrées par la figure B.1 et celles du véhicule polyvalent pour le scénario Sb par la figure B.2. Enfin, les résultats complets des routes des deux scénarios sont regroupés dans le tableau B.1 pour le véhicule urbain et dans le tableau B.2 pour le véhicule polyvalent.

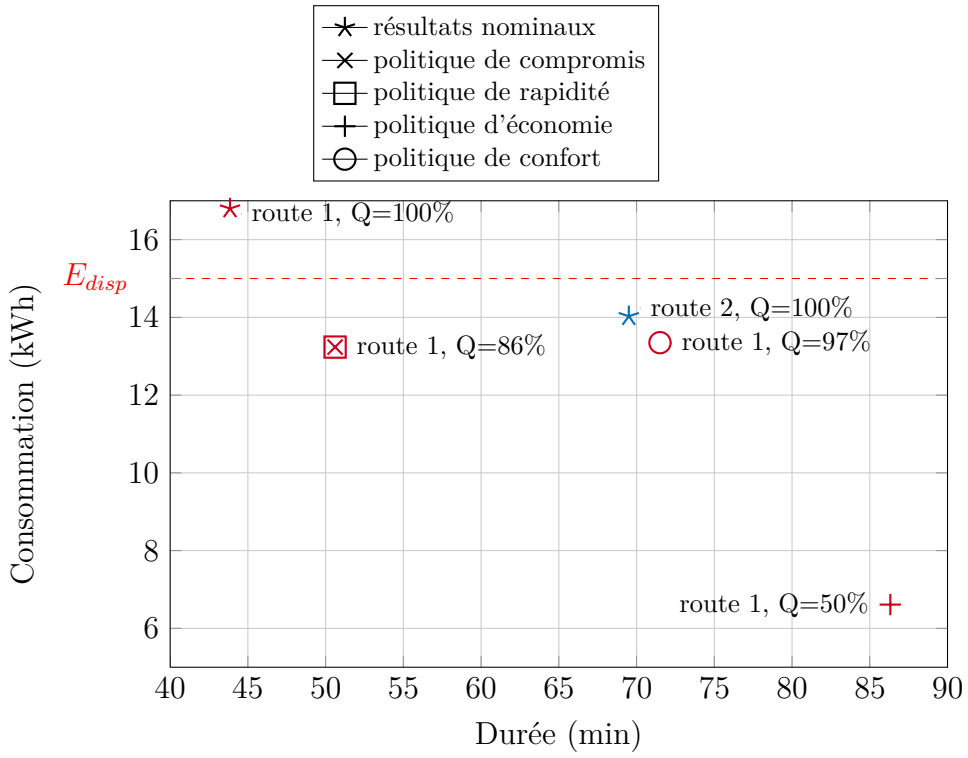


FIGURE B.1 – Résultats des stratégies de conduite optimales du véhicule urbain pour le scénario *Sa*.

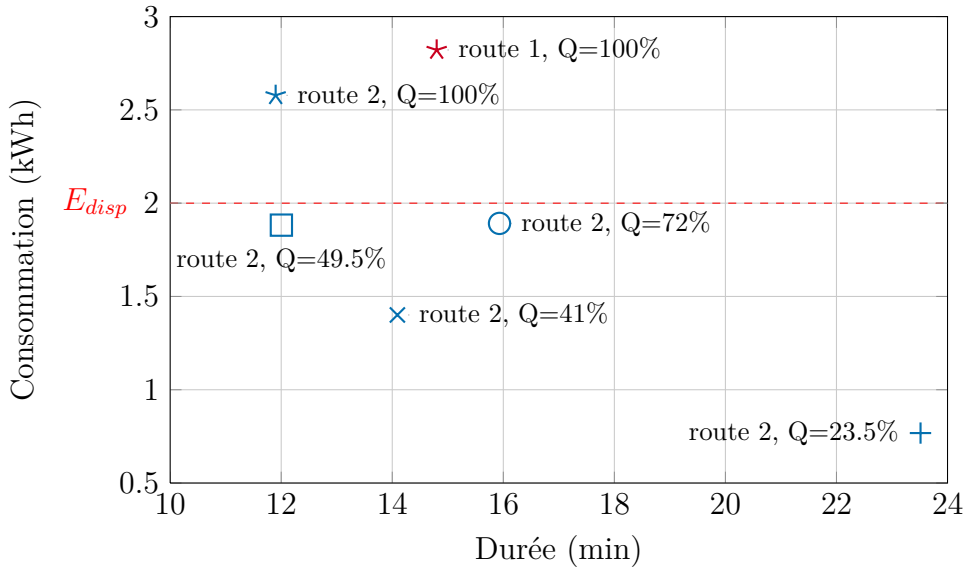


FIGURE B.2 – Résultats des stratégies de conduite optimales du véhicule polyvalent pour le scénario *Sb*.

TABLE B.1 – Résultats des stratégies de conduite proposées par ORQA pour le véhicule urbain. Les résultats approximés proviennent des solutions de la configuration optimisée ; les résultats évalués sont correspondent aux mêmes solutions si elles étaient systématiquement évaluées

		Résultats approximés			Résultats évalués		
Politique	Route	durée	consommation	qualité	durée	consommation	qualité
compromis	$Sa\#1$	51min 37s	13 236Wh	86%	51min 37s (0%)	13 236Wh (0%)	86% (0%)
rapidité	$Sa\#1$	51min 37s	13 236Wh	86%	51min 37s (0%)	13 236Wh (0%)	86% (0%)
économie	$Sa\#1$	1h 26min 19s	6 611Wh	50%	1h 26min 19s (0%)	6 611Wh (0%)	50% (0%)
confort	$Sa\#2$	1h 12min 30s	13 350Wh	97%	1h 11min 27s (0%)	13 444Wh (+1%)	97% (0%)
compromis	$Sb\#2$	14min 21s	1 888Wh	82%	14min 21s (0%)	1 888Wh (0%)	82% (0%)
rapidité	$Sb\#2$	14min 21s	1 888Wh	82%	14min 21s (0%)	1 888Wh (0%)	82% (0%)
économie	$Sb\#2$	23min 09s	1 152Wh	50%	23min 09s (0%)	1 152Wh (0%)	50% (0%)
confort	$Sb\#2$	14min 21s	1 888Wh	82%	14min 21s (0%)	1 888Wh (0%)	82% (0%)

TABLE B.2 – Résultats des stratégies de conduite proposées par ORQA pour le véhicule polyvalent. Les résultats approximés proviennent des solutions de la configuration optimisée ; les résultats évalués sont correspondent aux mêmes solutions si elles étaient systématiquement évaluées.

		Résultats approximés			Résultats évalués			
Politique	Route	durée	consommation	qualité	durée		consommation		qualité	
compromis	*Sa*#1	50min 30s	13 187Wh	88%	50min 30s	(0%)	13 187Wh	(0%)	94%	(+7%)
rapidité	*Sa*#1	44min 14s	12 297Wh	49,5%	44min 16s	(0%)	14 134Wh	(+15%)	49,5%	(0%)
économie	*Sa*#1	1h 27min 38s	5 580Wh	24%	1h 30min 54s	(+4%)	5 859Wh	(+5%)	24%	(0%)
confort	*Sa*#2	1h 13min 56s	13 298Wh	95%	1h 13min 13s	(0%)	13 528Wh	(+2%)	97,5%	(+3%)
compromis	*Sb*#2	14min 06s	1 401Wh	41%	14min 21s	(+2%)	1 646Wh	(+17%)	41%	(0%)
rapidité	*Sb*#2	12min 00s	1 882Wh	49,5%	12min 01s	(0%)	2 199Wh	(+17%)	49,5%	(0%)
économie	*Sb*#2	24min 31s	768Wh	23,5%	25min 35s	(+5%)	930Wh	(+21%)	23,5%	(0%)
confort	*Sb*#2	16min 56s	1 892Wh	72%	16min 24s	(+3%)	1 894Wh	(0%)	86%	(+19%)

Publications

Les travaux présentés dans cette thèse ont donnés lieu aux publications suivantes :

- Borjan Tchakaloff, Sébastien Saudrais et Jean-Philippe Babau : Modèles et transformations de modèles pour l'introduction de la consommation énergétique dans le standard AUTOSAR. *In* Philippe Collet et Philippe Merle, éditeurs : *Actes de la 1ère Conférence en IngénieriE du Logiciel (CIEL'12)*, pages 1–6, Rennes, France, 2012.
- Borjan Tchakaloff, Sébastien Saudrais et Jean-Philippe Babau : ORQA : Modeling Energy and Quality of Service within AUTOSAR Models. *In Proceedings of the 9th international ACM Sigsoft conference on Quality of software architectures (QoSA'13)*, page 3, Vancouver, British Columbia, Canada, 2013. ACM Press.
- Borjan Tchakaloff, Sébastien Saudrais et Jean-Philippe Babau : Efficient models configuration for an electric vehicle energy management software. *In Proceedings of the 40th Euromicro Conference on Software Engineering and Advanced Applications (SEAA'14)*, Verona, Italy, 2014. IEEE.
- Borjan Tchakaloff, Sébastien Saudrais et Jean-Philippe Babau : Integration of physical models in the ORQA framework for electric vehicle energy management. *In Proceedings of the 19th international doctoral symposium on Components and architecture (WCOP'14)*, pages 7–12, Marcq-en-Baroeul, France, 2014. ACM Press.

Bibliographie

[AA05] Tarek A. AlEnawy et Hakan Aydin : Energy-aware task allocation for rate monotonic scheduling. *Proceedings of the 11th IEEE Real Time and Embedded Technology and Applications Symposium (RTAS2005)*, 2005.

[ABD+98] Audi, BMW, Daimler, HARMAN et Microchip Technology : Media Oriented Systems Transport (MOST). Rapport technique, MOST Cooperation, 1998.

[Ald03] Frances K. Aldrich : Smart Homes : Past, Present and Future. *In* Richard Harper, éditeur : *Inside the Smart Home*, pages 17–39. Springer-Verlag, London, 2003.

[AUT] AUTOSAR : AUTomotive Open System ARchitecture (AUTOSAR). http://autosar.org.

[AUT03] AUTOSAR : AUTomotive Open System ARchitecture (AUTOSAR). Specification, AUTOSAR Partnership, 2003.

[AV07] David Arthur et Sergei Vassilvitskii : k-means++ : The advantages of careful seeding. *Proceedings of the 18th annual ACM-SIAM symposium on Discrete algorithms (SODA'07)*, pages 1027–1035, 2007.

[BAB12] Anton Beloglazov, Jemal Abawajy et Rajkumar Buyya : Energy-aware resource allocation heuristics for efficient management of data centers for Cloud computing. *Future Generation Computer Systems*, 28(5):755–768, mai 2012.

[BAC09] Harpreetsingh Banvait, Sohel Anwar et Yaobin Chen : A rule-based energy management strategy for Plug-in Hybrid Electric Vehicle (PHEV). *In 2009 American Control Conference*, pages 3938–3943. IEEE, 2009.

[Bar98] Stéphane Barbusse : Climatisation automobile, énergie et environnement. *Recherche - Transports - Sécurité*, 60:3–18, septembre 1998.

[BCL+06] Éric Bruneton, Thierry Coupaye, Matthieu Leclercq, Vivien Quéma et Jean Bernard Stefani : The FRACTAL component model and its support in Java. *Software - Practice and Experience*, 36:1257–1284, 2006.

[BGN+09] Basil Becker, Holger Giese, Stefan Neumann, Martin Schenck et Arian Treffer : Model-Based Extension of AUTOSAR for Architectural Online Reconfiguration. *In* Stefan Van Baelen, Thomas Weigert, Ileana Ober et Huascar Espinoza, éditeurs : *Proceedings of the 2nd International Workshop on Model Based Architecting and Construction of Embedded Systems (ACES-MB 2009), held as part of the 2009 International Conference on Model Driven Engineering Languages and Systems (MoDELS'09)*, volume 507 de *CEUR Workshop Proceedings*, pages 123–137. CEUR Workshop Proceedings, octobre 2009.

[BJPW99] Antoine Beugnard, Jean-Marc Jézéquel, Noël Plouzeau et Damien Watkins : Making components contract aware. *Computer*, 32(7):38–45, juillet 1999.

[Cam96] Andrew T. Campbell : *A Quality of Service Architecture*. Thèse de doctorat, Lancaster University, 1996.

[CCG+93] Andrew T. Campbell, Geoff Coulson, Francisco Garcia, David Hutchinson et Helmut Leopold : Integrated quality of service for multimedia communications. *In IEEE INFOCOM'93 The Conference on Computer Communications*, pages 732–739. IEEE Computer Society Press, 1993.

[Com12] Commission Européenne : Règlement (UE) n° 459/2012 de la Commission du 29 mai 2012 modifiant le règlement (CE) n° 715/2007 du Parlement européen et du Conseil ainsi que le règlement (CE) n° 692/2008 de la Commission en ce qui concerne les émissions des véhicules particuliers et utilitaires légers (Euro 6), 2012. JOUE L 142, 1.6.2012, p. 16–24.

[Con10] Conseil d'État : Décret n° 2010-1269 du 26 octobre 2010 relatif aux caractéristiques thermiques et à la performance énergétique des constructions. Décret 2010-1269, République Française, 2010. JORF n° 0250 du 27 octobre 2010 page 19250 texte n° 2.

[DAP+13] Rémi Druilhe, Matthieu Anne, Jacques Pulou, Laurence Duchien et Lionel Seinturier : Components mobility for energy efficiency of digital home. *In Proceedings of the 16th International ACM Sigsoft symposium on Component-Based Software Engineering (CBSE'13)*, page 153, New York, USA, 2013. ACM Press.

[DJP04] Olivier Defour, Jean-Marc Jézéquel et Noël Plouzeau : Extra-functional contract support in components. *In Proceedings of the 7th International Symposium on Component-Based Software Engineering (CBSE'04)*, mai 2004.

[Dru13] Rémi Druilhe : *L'Efficience Énergétique des Services dans les Systèmes Répartis Hétérogènes et Dynamiques : Application à la Maison Numérique*. Thèse de doctorat, Université Lille 1 Sciences et Technologies, décembre 2013.

[Ecla] Eclipse Foundation : Eclipse. http://eclipse.org.

[Eclb] Eclipse Foundation : Eclipse Modeling Framework (EMF). http://eclipse.org/modeling/emf/.

[Eclc] Eclipse Foundation : Xtext. http://eclipse.org/Xtext/.

[EGE09] Mehrdad Ehsani, Yimin Gao et Ali Emadi : *Modern electric, hybrid electric, and fuel cell vehicles : fundamentals, theory, and design*. CRC Press, 2e édition, 2009.

[Far10] Hassan Farhangi : The path of the smart grid. *IEEE Power and Energy Magazine*, 8:18–28, 2010.

[FEL01] Xiaobo Fan, Carla Ellis et Alvin Lebeck : Memory controller policies for DRAM power management. *In Proceedings of the 2001 international symposium on Low power electronics and design (ISLPED'01)*, pages 129–134, New York, USA, août 2001. ACM Press.

[FG12] Peter H. Feiler et David P. Gluch : *Model-Based Engineering with AADL : An Introduction to the SAE Architecture Analysis & Design Language*. Addison-Wesley, 2012.

[FSLM02] Jean-Philippe Fassino, Jean-Bernard Stefani, Julia Lawall et Gilles Muller : THINK : A Software Framework for Component-based Operating System Kernels. *In USENIX Annual Technical Conference General Track*, pages 73–86, Monterey (USA), juin 2002. USENIX Association.

[GCE99] Yimin Gao, Liping Chen et Mehrdad Ehsani : Investigation of the Effectiveness of Regenerative Braking for EV and HEV. Rapport technique, SAE International, Warrendale, PA, août 1999.

[GUH14] Stephan Gunther, Stefan Ulbrich et Wilfried Hofmann : Driving cycle-based design optimization of interior permanent magnet synchronous motor drives for electric vehicle application. *In Proceedings of the International Symposium on Power Electronics, Electrical Drives, Automation and Motion (SPEEDAM'14)*, pages 25–30. IEEE, juin 2014.

[HC01] George T. Heineman et William T. Councill : *Component-based software engineering : putting the pieces together*. Addison-Wesley Longman Publishing Co., Inc., juin 2001.

[HV14] Vinay Hanumaiah et Sarma Vrudhula : Energy-Efficient Operation of Multicore Processors by DVFS, Task Migration, and Active Cooling. *IEEE Transactions on Computers*, 63(2):349–360, février 2014.

[IET95] IETF : Common Format and MIME Type for Comma-Separated Values (CSV) Files. RFC 4180, Internet Engineering Task Force, 1995.

[IMT96] Intel Corporation, Microsoft Corporation et Toshiba Corporation : Advanced Configuration and Power Interface (ACPI) specification. Spécification, ACPI Promoters Corporation, 1996.

[ISO93] ISO : Controller Area Network (CAN) for high-speed communication. ISO 11898 :1993, International Organization for Standardization, 1993.

[ISO98] ISO : Quality of service : Framework. ISO/IEC 13236 :1998, International Organization for Standardization, 1998.

[ISO06] ISO : Eiffel : Analysis, Design and Programming Language. ISO/IEC 25436, International Organization for Standardization, 2006.

[ISO10] ISO : Communication on FlexRay. ISO 10681-1 :2010, International Organization for Standardization, 2010.

[ISO12] ISO : Object Management Group Object Constraint Language (OCL). ISO/IEC 19507, International Organization for Standardization, 2012.

[ISO14] ISO : Local Interconnect Network (LIN). ISO/DIS 17987-1 [under development], International Organization for Standardization, 2014.

[ITU88] ITU-T : Quality of service and dependability vocabulary. ITU-T Recommendation E.800, Telecommunication standardization sector of the International Telecommunication Union, 1988.

[KE95] James Kennedy et Russell Eberhart : Particle swarm optimization. *In Proceedings of the International Conference on Neural Networks (ICNN'95)*, volume 4, pages 1942–1948. IEEE, 1995.

[KGA12] Sofiene Kachroudi, Mathieu Grossard et Neil Abroug : Predictive Driving Guidance of Full Electric Vehicles Using Particle Swarm Optimization. *IEEE Transactions on Vehicular Technology*, 61(9):3909–3919, novembre 2012.

[LL12] James Larminie et John Lowry : *Electric Vehicle Technology Explained*. John Wiley & Sons, Ltd, Chichester, UK, 2e édition, août 2012.

[LLLW10] Li-hua Luo, Hong Liu, Ping Li et Hui Wang : Model predictive control for adaptive cruise control with multi-objectives : comfort, fuel-economy, safety and car-following. *Journal of Zhejiang University SCIENCE A (Applied Physics & Engineering)*, 11(3):191–201, février 2010.

[LQW08] Shaobo Liu, Qinru Qiu et Qing Wu : Energy Aware Dynamic Voltage and Frequency Selection for Real-Time Systems with Energy Harvesting. *In Design, Automation and Test in Europe (DATE'08)*, pages 236–241. IEEE, mars 2008.

[LS98] Jacob R. Lorch et Alan Jay Smith : Software strategies for portable computer energy management. *IEEE Personal Communications*, 5(3): 60–73, juin 1998.

[LTBP11] Maxime Louvel, Julien Tous, Jean-Philippe Babau et Alain Plantec : Ensuring QoS of Multimedia Applications in Heterogeneous Home Networks : The CPU Use Case. *In Proceedings of the 9th International Conference on Embedded and Ubiquitous Computing (IFIP'11)*, pages 19–26. IEEE, octobre 2011.

[MÏ0] Jürgen Mössinger : Software in Automotive Systems. *IEEE Software*, 27(2):92–94, mars 2010.

[MAA13] Frank Maker, Rajeevan Amirtharajah et Venkatesh Akella : MELOADES : Methodology for long-term online adaptation of embedded software for heterogeneous devices. *Journal of Systems Architecture*, 59(8):643–655, septembre 2013.

[Mar12] Moreno Marzolla : Optimizing the energy consumption of large-scale applications. *In Proceedings of the 8th international ACM SIGSOFT conference on Quality of Software Architectures (QoSA'12)*, page 123, New York, USA, 2012. ACM Press.

[MBH+02] Tony Markel, A. Brooker, T. Hendricks, V. Johnson, K. Kelly, B. Kramer, M. O'Keefe, S. Sprik et K. Wipke : ADVISOR : A systems analysis tool for advanced vehicle modeling. *In Journal of Power Sources*, volume 110, pages 255–266, 2002.

[MBNR68] M. Douglas McIlroy, J.M. Buxton, Peter Naur et Brian Randell : Mass-produced software components. *In Proceedings of the 1st International Conference on Software Engineering (ICSE'68)*, pages 88–98, Garmisch Pattenkirchen, Germany, 1968.

[Mey97] Bertrand Meyer : *Object-Oriented Software Construction*. Prentice Hall PTR, 2e édition, 1997.

[MH98] Vlada Matena et Mark Hapner : Enterprise JavaBeans (EJB). Spécification, Sun Microsystems Inc., 1998.

[MLS+05] Prakash Manghwani, Joseph Loyall, Praveen Sharma, Matthew Gillen et Jianming Ye : End-to-End Quality of Service Management for Distributed Real-Time Embedded Applications. *In Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (WPDRTS'05)*, page 138a. IEEE, 2005.

[MRBL14] Tedjani Mesbahi, Nassim Rizoug, Patrick Bartholomeüs et Philippe Le Moigne : A New Energy Management Strategy of a Battery/Supercapacitor Hybrid Energy Storage System for Electric Vehicular

Applications. *In Proceedings of the 7th IET International Conference on Power Electronics, Machines and Drives (PEMD 2014)*, page 3.3.01. Institution of Engineering and Technology, janvier 2014.

[MTB11] Felicitas Mensing, Rochdi Trigui et Éric Bideaux : Vehicle trajectory optimization for application in ECO-driving. *In Proceedings of the IEEE Vehicle Power and Propulsion Conference (VPPC'11)*, pages 1–6. IEEE, septembre 2011.

[MVH+10] Kevin Mets, Tom Verschueren, Wouter Haerick, Chris Develder et Filip De Turck : Optimizing smart energy control strategies for plug-in hybrid electric vehicle charging. *In Proceedings of the IEEE/IFIP Network Operations and Management Symposium Workshops (NOMSW'10)*, pages 293–299, 2010.

[NGT92] Oscar Nierstrasz, Simon Gibbs et Dennis Tsichritzis : Component-oriented software development. *Communications of the ACM*, 35:160–165, 1992.

[OBA98] F. Ortiz, R. Buisset et C. Adès : VEDELIC : New Technologies for Electric Vehicles, Results of Tests and Demonstration. *In Proceedings of the 15th Electric Vehicle Symposium*, Brussels, 1998.

[OMG] OMG : Object Management Group (OMG). http://www.omg.org/.

[OMG91] OMG : Common Object Request Broker Architecture (CORBA). Spécification, Object Management Group, Inc., 1991.

[OMG97] OMG : Unified Modeling Language (UML). Spécification, Object Management Group, Inc., 1997.

[OMG02] OMG : CORBA Component Model (CCM). Spécification, Object Management Group, Inc., 2002.

[OW202] OW2 : The Fractal Component Model. Spécification, ObjectWeb Consortium, 2002.

[Par10] Parlement européen : Directive 2010/30/UE du Parlement européen et du Conseil du 19 mai 2010 concernant l'indication, par voie d'étiquetage et d'informations uniformes relatives aux produits, de la consommation en énergie et en autres ressources des produits liés à l'énergie. Directive 2010/30/EU, Conseil de l'Union européenne, 2010. JOUE L 153, 18.6.2010, p. 1–12.

[PCML09] Luigi Palopoli, Tommas Cucinotta, Luca Marzario et Giuseppe Lipari : AQuoSA - adaptive quality of service architecture. *Software : Practice and Experience*, 39(1):1–31, janvier 2009.

[PdE+09] Panos Papadimitratos, Arnaud de La Fortelle, Knut Evenssen, Roberto Brignolo et Stefano Cosenza : Vehicular communication

systems : Enabling technologies, applications, and future outlook on intelligent transportation. *IEEE Communications Magazine*, 47(11):84–95, novembre 2009.

[PS11] Nicolas Petit et Antonio Sciarretta : Optimal Drive of Electric Vehicles Using an Inversion-Based Trajectory Generation Approach. *In* Bittanti Sergio, Angelo Cenedese et Sandro Zampieri, éditeurs : *Proceedings of the 18th IFAC World Congress (IFAC'11)*, pages 14519–14526, août 2011.

[Rob11] Robert Bosch GmbH : *Automotive Handbook*. John Wiley & Sons Inc, 8e édition, mai 2011.

[RS04] Alain Roy et Volker Sander : Gara : A Uniform Quality of Service Architecture. *In Grid Resource Management*, volume 64, pages 377–394. Springer International Publishing, 2004.

[RSR07] Grégory Rousseau, Delphine Sinoquet et Pierre Rouchon : Constrained Optimization of Energy Management for a Mild-Hybrid Vehicle. *Oil & Gas Science and Technology - Revue de l'IFP*, 62(4):623–634, décembre 2007.

[SCZ+12] Éric Senn, Daniel Chillet, Olivier Zendra, Cécile Belleudy, Sébastien Bilavarn, Rabie Ben Atitallah, Christian Samoyeau et Agnès Fritsch : Open-People : Open Power and Energy Optimization PLatform and Estimator. *In 2012 15th Euromicro Conference on Digital System Design*, pages 668–675. IEEE, septembre 2012.

[SF12] Rabia Sehab et Gilles Feld : An Online Estimation of Energy Recovery in an Electric Vehicle Using ARTEMIS Mission Profiles. *In Proceedings of the IEEE Vehicle Power and Propulsion Conference (VPPC'12)*, pages 333–338. IEEE, octobre 2012.

[SLPH09] David C. Snowdon, Étienne Le Sueur, Stefan M. Petters et Gernot Heiser : Koala - A Platform for OS-Level Power Management. *In Proceedings of the 4th ACM european conference on Computer systems (EuroSys'09)*, page 289, New York, USA, 2009. ACM Press.

[Sol00] Richard Soley : Model Driven Architecture. Rapport technique, Object Management Group, Inc., 2000.

[SSK03] Niels J. Schouten, Mutasim A. Salman et Naim A. Kheir : Energy management strategies for parallel hybrid vehicles using fuzzy logic. *Control Engineering Practice*, 11(2):171–177, février 2003.

[Szy02] Clemens Szyperski : *Component Software : Beyond Object-Oriented Programming*. Addison-Wesley Longman Publishing Co., Inc., octobre 2002.

[TBO05] Jean-Charles Tournier, Jean-Philippe Babau et Vincent Olive : Qinna, a component-based qos architecture. *In Proceedings of the International ACM Sigsoft symposium on Component-Based Software Engineering (CBSE'05)*, pages 1–16, 2005.

[TdCF13] Silvana Teodoro, Andriele Busatto do Carmo et Luiz Gustavo Fernandes : Energy efficiency management in computational grids through energy-aware scheduling. *In Proceedings of the 28th Annual ACM Symposium on Applied Computing (SAC'13)*, page 1163, New York, USA, 2013. ACM Press.

[The00] The OpenMath Society : The OpenMath Standard. Spécification, The OpenMath Society, 2000.

[W3C98] W3C : Mathematical Markup Language (MathML). W3C Recommendation REC-MathML-19980407, World Wide Web Consortium, 1998.

[WC05] Alice Wang et Anantha Chandrakasan : A 180-mV subthreshold FFT processor using a minimum energy design methodology. *IEEE Journal of Solid-State Circuits*, 40(1):310–319, janvier 2005.

[WK94] Sara Williams et Charlie Kindel : The component object model (COM) : A technical overview. Rapport technique, Microsoft, Inc., 1994.

[WK98] Jos Warmer et Anneke Kleppe : *The Object Constraint Language : Precise Modeling With UML*. Addison-Wesley Professional, 1998.

Liste des algorithmes

Liste des tableaux

Table des figures

Table des listings

ORQA : un canevas logiciel pour la gestion de l'énergie dans les véhicules électriques

Résumé

Les véhicule électriques présentent désormais une alternative crédible aux véhicules équipés de moteurs à combustion interne. Ils sont plus propres et plus confortables à l'utilisation. La gestion de l'énergie d'un véhicule électrique est actuellement focalisée sur le fonctionnement du moteur, principal consommateur d'énergie, sans tenir compte du confort de l'utilisateur.

Le travail présenté dans cette thèse intègre la prise en compte des préférences utilisateur au sein de la gestion de l'énergie des véhicules électriques. La contribution est réalisée par un canevas logiciel nommé ORQA. Dans un premier temps, les organes du véhicule sont caractérisés par leurs besoins énergétiques et par les niveaux de qualité de service qu'ils proposent. Un gestionnaire d'énergie est ensuite intégré au système logiciel du véhicule. Il se base sur les caractéristiques des organes et propose à l'utilisateur une solution de configuration de trajet prenant en compte ses préférences d'utilisation ainsi que d'éventuelles contraintes de temps et d'énergie. Cette solution définit des contraintes d'utilisation sur le moteur et les organes de confort lors du déroulement du trajet.

Le gestionnaire d'énergie est exécuté au sein des systèmes embarqués du véhicule dont les plates-formes sont fortement contraintes. Pour satisfaire celles-ci, il est primordial de proposer une configuration efficace du gestionnaire d'énergie. L'intérêt et la validité de l'approche employée sont démontrés au travers de la comparaison entre la configuration originelle et une configuration optimisée sur deux véhicules exemples.

Mots-clefs

Gestion d'énergie, qualité de service, véhicule électrique, ingénierie dirigée par les modèles, systèmes embarqués.

ORQA: a framework for the energy management in electric vehicles

Abstract

Electric vehicles are nowadays a viable alternative to vehicles built around an internal combustion engine. They offer a cleaner and more comfortable driving thanks to the electric engine. The energy management of an electric vehicle is usually focused on the engine operation, the biggest energy consumer, thus ignoring the user comfort.

The contribution presented in this thesis allows for the consideration of the user preferences inside the energy management of electric vehicles. It takes shape with a framework named ORQA. First, the vehicle devices are characterised by their energy requirements and the quality levels they offer. An energy manager based on the devices characteristics is then integrated into the software system of the vehicle. The manager offers the user a solution to configure a trip request. The configuration is based on the usage preferences and optional constraints over duration and consumption. The solution defines a set of constraints on the motor and on the comfort-related devices during the trip execution.

The energy manager is executed in the embedded systems of the vehicle which platforms are highly constrained. Thus, the energy manager's implementation is optimised to satisfy the execution platform constraints. The interest and the validity of the chosen approach are attested by the comparison of the original configuration and an optimised configuration on two example vehicles.

Keywords

Energy management, quality of service, electric vehicle, model based engineering, embedded systems.