Un Modèle Hiérarchique pour les Architectures Reconfigurables en Radio Logicielle

Jean-Philippe Delahaye, Jacques Palicot, Pierre Leray

IETR/SUPELEC, Campus de Rennes, Avenue de la Boulais, Cesson-Sévigné, BP 81127, France
Jean-Philippe.Delahaye@supelec.fr,Jacques.Palicot@supelec.fr,Pierre.Leray@supelec.fr

Résumé – Cet article présente un modèle hiérarchique de la gestion de configuration d'architecture reconfigurable dédié aux systèmes multi-standards. A partir d'une étude au niveau algorithmique nous proposons de décrire un modèle fonctionnel de gestion de configuration tenant compte des besoins applicatifs. L'analyse algorithmique porte sur une chaîne d'émission multi-standards UMTS/FDD Uplink, GSM Uplink et 802.11g mode OFDM. Le but du modèle présenté est de minimiser les ressources matérielles à reconfigurer à chaque nouveau changement de contexte applicatif. Par le contrôle de la reconfiguration partielle et dynamique de la chaîne, ce modèle hiérarchique est bien adapté à la gestion des ressources hétérogènes d'un système en correspondance avec les besoins d'adaptabilité de l'application multi-standards.

Abstract – This paper presents a functional model based on a hierarchical architecture template meeting with Software Defined Radio System requirements (SDR System). The concepts and mechanisms required to design future reconfigurable system architectures are addressed in the paper. The definition of the new features requested in such architectures is based on a thorough multi-standards functional analysis of a transmitter (i.e. UMTS/FDD Uplink, GSM Uplink, and 802.11g OFDM mode). In addition to a classical datapath for processing a configuration management path has been integrated. This model aims at helping the design and management of a heterogeneous dynamically reconfigurable hardware architecture for SDR terminals.

1 Introduction

Les techniques radio logicielle introduitent par Mitola [7] visent à offrir un accès à un large choix de standards de radiocommunications sur une architecture matérielle unique. Une grande fléxibilité de l'architecture du système radio logicielle est donc nécessaire afin de répondre à la diversité des traitements à exécuter. Cette fléxibilité est assurée par l'utilisation d' architectures reconfigurables [1]. La diversité des traitements implique l'utilisation d'architecture matérielles hétérogènes [6] afin que l'exécution soit optimisée. Diversité des traitements et hétérogénéité de l'architecture induisent une grande variété de configurations. Il est donc nécessaire de définir une architecture de gestion et de contrôle de configurations adaptée. Les différents besoins en termes de gestion de configuration d'un terminal radio logicielle ont largement été justifiés dans la littérature, comme par exemple par Gultchev et al. dans [2]. De plus, suivant les besoins d'adaptabilité du terminal, les configurations sont de natures différentes, comme le décrit Kountoutis et al. dans [5]. Par exemple, la reconfiguration intervient lors d'un changement de standard, de mode, mais aussi en vue de la correction ou l'amélioration des performances d'algorithmes. Ceci correspond à différents niveaux de granularité. Le problème se pose de gérer cette multi-granularité de contexte en correspondance avec une gestion efficace des ressources matérielles dont la nature est hétérogène, par exemple bit-level configuration des FPGAs et word level configuration des DSPs. Notre modèle hiérarchique permet de faciliter cette gestion de configuration multi-granularité et ainsi minimiser les ressources à reconfigurer à chaque changement de contexte afin d'assurer une reconfiguration dynamique.

Ce papier s'appuie sur des travaux ayant commencés par une étude algorithmique des traitements bande de base effectués dans les chaînes d'émissions des 3 standards de radiocommunications (UMTS, GSM,WLAN 802.11g). Ces standards ont été choisi pour la grande diversité des traitements offerte. Les applications visées nous ont conduit à choisir une architecture orientée flot de données présentée dans le paragraphe suivant. L'étude algorithmique est détaillée dans les parties 2 et 3, elle a mené à la proposition du modèle hiérarchique présenté dans la section 4.

1.1 L'approche Config-Data Path

Le domaine des architectures reconfigurables a déjà largement retenue l'approche flot de données pour les applications télécoms, R. Hartenstein dans [3] en liste les principaux projets. Face à la diversité des applications en radio logicielle et les contraintes de surface, de rapidité de calcul et de basse consommation, chaque type de fonction nécessite une implémentation optimisée appelant des traitements spécialisés. La reprogrammation de ressources matérielles hétérogènes doit donc être pris en charge. Ces raisons nous ont conduit à associer à chaque bloc de traitement, un gestionnaire de configuration dédié, afin de spécialiser la gestion de reconfiguration. D'autres approches de gestion de configuration d'architecture orientée flot de donnée ont été proposées notamment par Srikanteswara et al. [9] dans un modèle en couche. Dans cette proposition, les données de configuration et les données à traiter sont transmises par le même chemin, elles sont donc

encapsulées et augmentées d'un en-tête les différenciant. Afin d'éviter ce surcoût de transmission d'un en-tête et l'ajout d'un protocole nous proposons de séparer les 2 flux de données, qui ont de plus des débits différents. La figure 1 illustre ce modèle basé sur la séparation de la voie de traitement et de la voie de configuration.

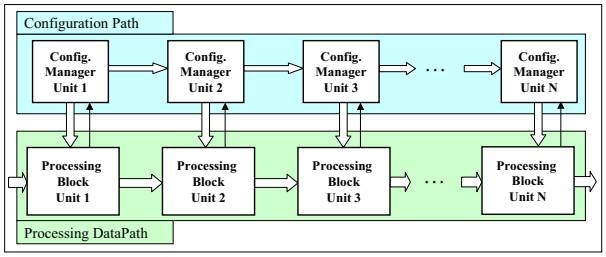

FIG. 1 – Functional Modeling of the Configuration-Processing Datapath

Avantages de l'approche : La séparation des 2 types de flux de données permet de dissocier plus aisément la partie contrôle et le traitement des données. Chaque Configuration Manager Unit (CMU) est associé à un bloc de traitement PBU (Processing Block Unit). Les ressources materielles des PBUs étant spécifiques à un type de traitement, chaque CMU est spécialisé dans la gestion de ces ressources cela permet de simplifier la conception de chaque entité de gestion et d'optimiser leur utilisation. L'architecture distribuée en composants CMUs/PBUs, permet une gestion efficace de la reconfiguration partielle. L'estimation du temps de reconfiguration de chaque PBU par les CMUs et la transmission de ces informations de configurations à travers le chemin de configuration permet de garantir le fonctionnement des reconfigurations dynamiques.

2 Cas de changement de contexte

L'adaptation d'un système radio logicielle à un environnement applicatif multi-standards supportée par une plate-forme d'exécution unique passe par l'utilisation de composants reconfigurables. Bien qu'un changement d'application ait un impact sur l'ensemble des couches protocoles de l'application de transmission, nous restreignons le terme changement de contexte à la couche physique. Afin d'optimiser l'adaptabilité du système reconfigurable, il faut diminuer le coût d'une demande de changement de contexte, en minimisant les ressources à reconfigurer. La première étape d'optimisation de la reconfiguration passe par une analyse au niveau algorithmique des différents traitements communs entre les différentes applications. A ce niveau, nous pouvons noter les travaux de R. Hoshyar et al. [4] proposant une méthode pour réaliser cette analyse des applications multi-modes. Par ailleurs, sur la définition et la proposition d'opérateurs communs efficaces et les études de parametrisation, nous nous reportons aux travaux de J. Palicot et al. [8]. Lors d'une reconfiguration optimisée, seul les blocs de traitements spécifiques au nouveau contexte par rapport au précédent se verront modifiés afin d'éviter un rechargement total du contexte d'exécution. Afin d'augmenter le taux de réutilisation des blocs de traitements (ou opérateurs communs) ceux-ci seront paramétrables. Enfin, face à la diversité de changement de contexte, nous en avons définit différents types.

- **Standard Switching** : Un changement complet de standard implique une refonte profonde de la chaîne de transmission. Les chaînes de transmissions entre les standards étant très différentes, c'est le type de changement de contexte qui demande les plus larges reconfigurations. Il est à noter qu'un standard switching concerne aussi les couches supérieures OSI.
- **Mode Switching** : Certains standards spécifient plusieurs modes de fonctionnement. Un mode switch est donc un changement de contexte intra-standard alors qu'un standard switch est un changement inter-standard. Dans la plupart des cas, le changement de mode ne concerne donc pas les couches hautes. Les paramètres du changement de mode pour la couche physique sont déterminés par la couche MAC. Par exemple le standard 802.11g possède plusieurs mode de fonctionnement DSSS, FHSS, OFDM.
- **Service Switching** : Un service switch sur la couche physique reste un changement de contexte intra-standard contrairement à un service switch au niveau applicatif pouvant entraîner un standard switch. Ce changement peut correspondre par exemple à une demande de changement de débit, la chaîne de traitement est généralement peu modifiée. Ce changement peut-être effectué par paramétrage ou par reconfiguration d'une partie des fonctions. Par exemple en mode OFDM du 802.11g, lors d'une demande d'augmentation de débit de 6Mbit/s à 54Mbit/s, la fonction de mapping change respectivement de BPSK à 64-QAM.
- **Performance enhancement, bug fixing** : La possibilité de changer un motif de traitement en vue de l'amélioration de ses performances ou dans le but de corriger une erreur peut correspondre à divers types de situations. La possibilité de réaliser ce type de petits changements nécessite la mise en place de règles de développement des algorithmes par composants de traitements rendant leurs reconfigurations possible de manière indépendantes. Ceci nécessite en outre la mise en place de mécanismes de contrôle afin de limiter la discontinuité de traitement occasionnée.

L'analyse des différents besoins de changement de contexte, effectuée ci-dessus, a mis en évidence qu'il existe différents niveaux de granularité des changements de contextes. Il est donc nécessaire de pouvoir gérer cette multi-granularité au niveau de la reconfiguration de l'architecture système afin d'en améliorer l'efficacité. Nous proposons dans la section 4, une architecture de gestion de configuration hiérarchique répondant efficacement aux besoins de gestion de la multi-granularité de configuration liées aux applications multi-standards.

3 Classification fonctionnelle

Cette seconde étape d'analyse porte sur les fonctions bande de base des 3 standards à l'émission. L'étude algorithmique nous a mené à constituer des classes suivant les caractéristiques et besoins matériels communs aux fonctions. Chacune des 3 classes fonctionnelles présentées par la suite, regroupe les fonctions sous forme générique. Par exemple la fonction générique codage convolutionnel dont les paramètres sont le taux de codage et les polynômes générateurs sont présentés pour les 3 standards étudiés dans le tableau 1.

TAB. 1 – Parameters for the 'Convolutional coding' Generic Function

Conv. Coding tri-standards	Coding Rate	Generator Polynomial
GSM	1/2	$G_0 = D^4 + D^3 + 1$
TCH/FH		$G_1 = D^4 + D^3 + D + 1$
TCH/HS	1/3	$G_4 = D^6 + D^5 + D^3 + D^2 + 1$
		$G_5 = D^6 + D^4 + D + 1$
		$G_6 = D^6 + D^4 + D^3 + D^2 + D + 1$
UMTS	1/2	$G_0 = D^8 + D^6 + D^5 + D^4 + 1$
BCH, PCH, RACH		$G_1 = D^8 + D^7 + D^6 + D^5 + D^3 + D + 1$
CPCH, DCH ...	1/3	$G_0 = D^8 + D^6 + D^5 + D^3 + D^2 + D + 1$
		$G_1 = D^8 + D^7 + D^5 + D^4 + D + 1$
		$G_2 = D^8 + D^7 + D^6 + D^3 + 1$
802.11g	1/2	$G_a = D^6 + D^4 + D^3 + D + 1$
(OFDM Mode)		$G_b = D^6 + D^5 + D^4 + D^3 + 1$

Une fonction est composée de blocs de traitements paramétrables. Ces blocs permettent de réaliser l'ensemble des fonctions paramétrées des différents standards. L'intérêt de la classification fonctionnelle est de grouper les fonctions nécessitant des ressources de traitement similaires. Les blocs de traitement paramétrables sont donc utilisables comme blocs ou motifs communs à plusieurs fonctions. Le groupement en classe fonctionnelle est proposé en

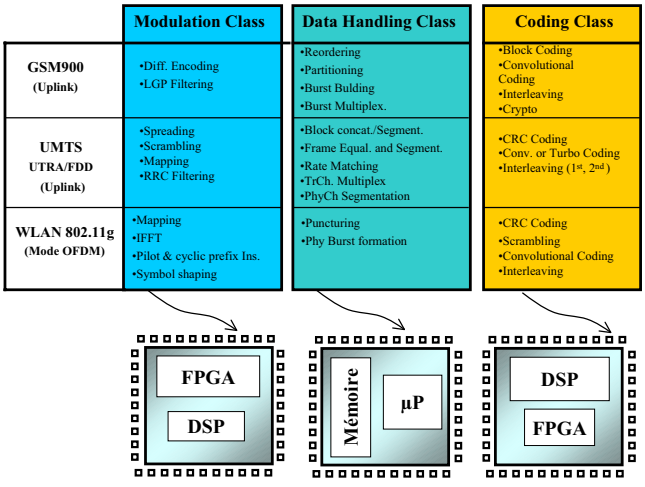

FIG. 2 – Functional Classification and HW/SW Needs

correspondance avec les ressources matérielles nécessaires, comme l'illustre schématiquement la figure 2. Ainsi, les implémentations de motifs d'exécution logiciels/matériels sont adaptées aux capacités de traitements requises par une classe de fonctions. Les 3 classes fonctionnelles sont présentées en détail ci-dessous :

- **Coding Class** : Cette classe regroupe les fonctions de codage canal, cyclic, convolutionnel turbo... . Elle est caractérisée par une grande variété de schémas de codage, ce qui nécessite une architecture d'exécution flexible. Une implémentation logicielle est possible car la capacité de traitement demandée n'est pas élevée sauf dans le cas du 802.11g à 54Mbps. En revanche, l'exécution des architectures de type processeurs 16/32 bits est sous-optimisée car les données à traiter sont de type bit. Une implémentation faible consommation sera basée sur l'utilisation de blocs de traitements matériel paramétrables.
- **Data Handling Class** : Cette classe regroupe les fonctions de manipulation de packets de données. Ces fonctions de transfert des données sont orientées contrôle. Les données sont de taille très différentes en entrée et sortie, dû à la nature des fonctions (concaténation, segmentation, multiplexage, etc.) . Très consommatrices de mémoire elles nécessitent une grande flexibilité de traitement, offerte par des architectures de type processeurs. Afin de réduire la consommation de puissance, il est nécessaire de faire attention à la gestion d'alimentation des blocs mémoires utilisés suivant les fonctions.
- **Modulation Class** : Cette classe rassemble les fonctions effectuées en bande de base juste avant la transposition en fréquence. Les fonctions sont souvent réalisées sur des données sur-échantillonnées demandant de grandes capacités de calcul notamment les fonctions de filtrage. Certaines de ces fonctions tirent avantages d'une implémentation sur des accélérateurs matériels.

4 Approche hiérarchique de gestion de configuration

Suite à l'analyse au niveau algorithmique, nous proposons ici de décrire un modèle fonctionnel de la gestion de configuration tenant compte des besoins de reconfiguration, déduit de cette étude. De plus, la gestion de configuration doit être en même temps adaptée aux capacités de reconfigurabilité des composants. Le but est de minimiser les ressources matérielles à reconfigurer à chaque nouvelle demande de changement de contexte de l'application. L'aspect hiérarchique de cette architecture de gestion de configuration permet de gérer efficacement la multi-granularité de configuration. Le gestionnaire de niveau 1 a un rôle de superviseur initiant une reconfiguration sur la chaîne de traitement. Cette reconfiguration est alors prise en charge par les gestionnaires de niveaux 2 à 3. Nous décrivons plus en détails ci-dessous le rôle de chaque niveau hiérarchique :

- **Niveau hiérarchique 1** : Le gestionnaire de configuration (*L1_CM*) fonctionne au niveau standard. Cette entité est en charge de sélectionner les fonctions gé-

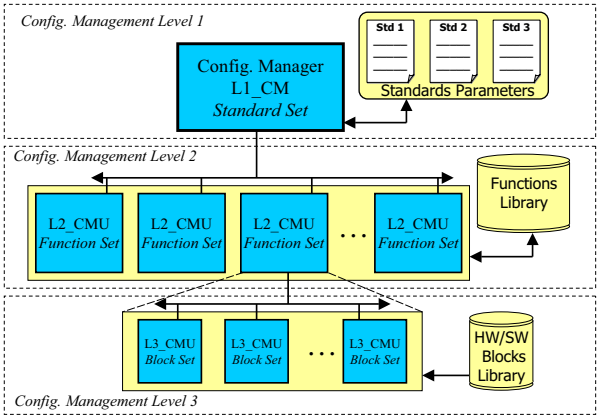

FIG. 3 – Hierarchical model of Configuration Management

nériques afin de constituer une chaîne dont l'ensemble des blocs fonctionnels correspond à un standard donné. A ce niveau, les fonctions sont vues comme des boîtes noires assemblées les unes par rapport aux autres et dotées d'un ensemble de paramères définit suivant les spécifications du standard choisi. Un changement de standard sera donc gérer par ce niveau hiérarchique.

- **Niveau hiérarchique 2** : Chaque fonction paramétrée par le *L1_CM* est prise en charge par une unité de gestion dédiée, appelée *Configuration Manager Unit (L2_CMU)*. Une fonction est constituée d'un ou plusieurs blocs de traitements. Le rôle d'un *L2_CMU* est de configurer cette fonction en sélectionnant les blocs de traitement en spécifiant aussi leurs interconnexions.
- **Niveau hiérarchique 3** : Chaque sous-bloc est paramétré, suivant les critères définies par le *L2_CMU*, puis instancié matériellement par un *L3_CMU*. Les interfaces de communications associées au bloc de traitement sont aussi configurées par le *L3_CMU*. La tâche principale des *L3_CMUs* est donc de configurer les ressources matérielles.

L'approche hiérarchique de gestion de configuration est appropriée pour gérer la multi-granularité de configuration et améliore par une gestion distribuée le contrôle de la reconfiguration partielle de la chaîne de traitement. Le partitionnement des fonctions en sous-bloc de traitement permet de garder une grande flexibilité de configuration. La mise en oeuvre de cette flexibilité est rendue efficace par la gestion hiérarchique. Cette découpe hiérarchique corresponds aussi aux différents besoins de changement de contexte identifiés dans le paragraphe 2. Suivant le type de changement de contexte demandé les différents niveaux de gestions de configurations sont utilisés. Le changement de standard utilisera tout les ressources de configuration du *L1_CM* aux *L3_CMUs*. Lors d'un changement de service où une seule fonction est changée, seul le *L2_CMU* associé est utilisé. Dans le cas d'une correction d'un programme (bug fixing) intervenant sur quelques motifs de traitements seuls les *L3_CMUs* concernés par le changement sont alors sollicités.

L'architecture d'exécution associée : L'architecture d'exécution est hétérogène pour répondre aux besoins des différentes classes de traitements présentées dans le paragraphe 3. Le partitionnement d'une application en composants puis en sous-blocs (PBU) simplifie la configuration. En effet, un PBU est soit matériel soit logiciel, le fait d'associer à chaque bloc de traitement un *L3_CMU* permet de spécialiser la gestion de configuration de ce dernier aux ressources qu'il contrôle.

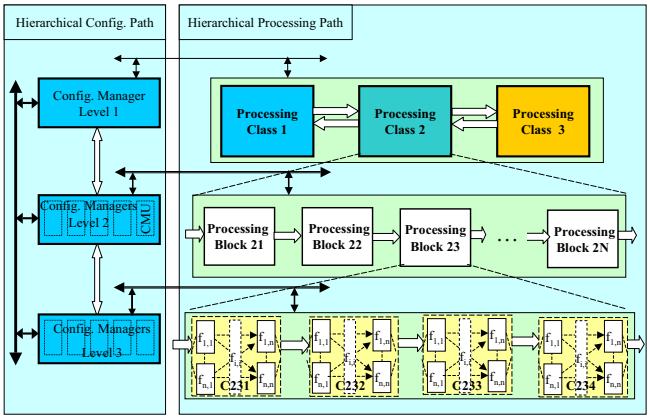

FIG. 4 – Hierarchical model for SDR System Design

5 Conclusions et perspectives

La conception système dans le contexte radio logicielle implique de définir une architecture ouverte et flexible prenant en compte la possibilité d'adapter le système aux futures évolutions des standards ou l'émergence de nouveaux. Nous pensons que notre approche est assez générique pour répondre à la diversité des standards. L'analyse fonctionnelle réalisée à l'émission et le modèle hiérarchique proposé peuvent être étendus côté réception. Le modèle de gestion hiérarchique sera validé sur une plate-forme de prototypage DSP/FPGA supportant les applications de reconfiguration d'un émetteur multi-standards.

Références

[1] M. Cummings and S. Haruyama., "Fpga in software radio," *IEEE Comms. Mag.*, pp. 108–112, 2 1999.

[2] S. Gultchev, K. Moessner, and R. Tafazoli, "Management and control of reconfiguration procedures in software radio terminals," in *Proc. 2^{nd} Workshop on Software Radios*, Karlsruhe, Germany, March 2002, pp. 125–129.

[3] R. Hartenstein, "A decade of reconfigurable computing : a visionary retrospective," in *Proc. Design, Automation, and Test in Europe*, Munich, Germany, 2001, pp. 642–649.

[4] R. Hoshyar, S. Gultchev, K. Seo, and R. Tafazolli, "Software reconfigurability - algorithm level approach," in *Proc. 4^{th} International Conference on 3G Mobile Communication Technologies*, London, UK, June 2003.

[5] A. Kountouris and C. Moy, "Reconfiguration in software radio systems," in *Proc. 2^{nd} Workshop on Software Radios*, Karlsruhe, Germany, March 2002, pp. 119–124.

[6] M. Mehta, N. Drew, and C. Niedermeier, "Reconfigurable terminals : an overview of architectural solutions," *IEEE Comms. Mag.*, pp. 82–88, 8 2001.

[7] J. Mitola, "The software radio architecture," *IEEE Comms. Mag.*, vol. 33, pp. 26–38, 5 1995.

[8] J. Palicot and C. Roland, "Fft a basic function for a reconfigurable receiver," in *Proc. 10th International Conference on Telecommunications*, Papeete, Tahiti, Febuary 2003, pp. 898–902.

[9] S. Srikanteswara, J. Reed, P. Athanas, and R. Boyle, "A soft radio architecture for reconfigurable platforms," *IEEE Comms. Mag.*, vol. 38, pp. 140–147, 2 2000.