

Conception d'un système avancé de réacteur PWR flexible par les apports conjoints de l'ingénierie système et de l'automatique

Lori Lemazurier

▶ **To cite this version:**

Lori Lemazurier. Conception d'un système avancé de réacteur PWR flexible par les apports conjoints de l'ingénierie système et de l'automatique. Automatique. Ecole nationale supérieure Mines-Télécom Atlantique, 2018. Français. NNT : 2018IMTA0069 . tel-01815345

HAL Id: tel-01815345

https://theses.hal.science/tel-01815345

Submitted on 14 Jun 2018

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

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

UNIVERSITE BRETAGNE LOIRE

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Thèse de Doctorat

Lori LEMAZURIER

Mémoire présenté en vue de l'obtention du
grade de Docteur de L'Ecole nationale supérieure Mines-Télécom Atlantique Bretagne-Pays de la Loire - IMT Atlantique
sous le sceau de l'Université Bretagne Loire

École doctorale : Mathématique & Sciences et Technologies de l'Information et de la Communication (MathSTIC)

Discipline : Automatique, Ingénierie-Système, Nucléaire
Unité de recherche : Laboratoire des Science du Numérique de Nantes (LS2N)

Soutenue le : 2 février 2018
Thèse N : 2018IMTA0069

Conception d'un Système Avancé de Réacteur PWR Flexible par les Apports Conjoints de l'Ingénierie Système et de l'Automatique

JURY

Rapporteurs :	**Marija JANKOVIC**, Maitre de Conférence HDR, Centrale-Supelec **Henri BOURLES**, Professeur, CNAM
Examinateurs :	**Nicolas PETIT**, Professeur, Mines ParisTech (Président du jury) **Eric BONJOUR**, Professeur, Université de Lorraine **Mohamed YAGOUBI**, Maitre-Assistant, Co-encadrant de thèse, IMT Atlantique
Invité :	**Alain GROSSETETE**, Senior Expert, Framatome
Directeur de Thèse :	**Philippe CHEVREL**, Professeur, IMT Atlantique
Co-directeur de Thèse :	**Vincent CHAPURLAT**, Professeur, IMT Mines Alès

Remerciements

Je ne saurais trouver les mots justes qui qualifieraient ma gratitude à l'égard de ceux qui ont contribué à ce travail de thèse. Mes premières pensées vont pour les membres de mon encadrement, envers qui j'éprouve un profond respect et à qui j'espère avoir fait honneur par la restitution du savoir si précieux qu'ils m'ont transmis. Je commencerai par Philippe CHEVREL, mon directeur de thèse, pour le calme et la rigueur qu'il m'a apportés et pour sa présence dans les moments difficiles. Vincent CHAPURLAT, mon co-directeur de thèse, pour son soutien sans faille et sa motivation constante. Alain GROSSETETE, mon encadrant industriel, pour sa disponibilité, sa bienveillance et son engagement tout au long de la thèse. Et enfin Mohamed YAGOUBI, mon co-encadrant de thèse, pour son aide précieuse.

J'aimerais ensuite remercier les rapporteurs, Marija JANKOVIC et Henri BOURLES, d'avoir accepté d'évaluer mon travail et pour leurs remarques constructives qui m'ont permis d'approfondir mon travail et d'améliorer la qualité du manuscrit. Je remercie également les examinateurs, Nicolas PETIT et Eric BONJOUR, pour l'indulgence qu'ils ont eu à mon égard compte tenu des difficultés rencontrées pour la transmission du manuscrit, mais aussi pour leurs remarques précieuses qui me permettront de prolonger ce travail de thèse.

Je ne peux oublier l'ensemble des ingénieurs de Framatome qui ont contribué de près ou de loin à la réalisation de ces travaux : Jean-Pierre DANIEL, Pierre BARTHELET et Andrea BARBARINO ainsi que les équipes DTIPCN-F et DTIPCR-F qui m'ont encouragé et ont fait de mon cadre de travail un environnement agréable. Merci aux doctorants et docteurs des écoles IMT Atlantique et IMT Mines Alès pour les moments partagés et les encouragements ô combien précieux. Merci à tous mes amis sans qui le voyage aurait été bien plus pénible. Dans le désordre, Eddy, Antony, Alex, Céline, Valentina, Laetitia, Constance, Romain, Antoine, Nico, Pilou, Albin, J-C, Manu et Léon.

J'aimerais enfin remercier ceux qui ont toujours cru en moi et qui m'ont permis de surmonter les moments d'incertitude. Je pense en particulier à Cecília, qui m'a donné la force et a su me supporter jusqu'au bout mais aussi à ma mère et mon père. Et enfin, à tous les membres de ma famille, en particulier Mina, Tessy, Jade, Karim, ainsi que tous les autres…

Résumé

Devant l'augmentation de la part des énergies renouvelables en France, cette thèse propose d'étudier l'augmentation de la flexibilité des réacteurs à eau pressurisée en croisant deux disciplines pour, chacune, atteindre des objectifs complémentaires : l'Ingénierie Système (IS) et l'Automatique.

Dans le contexte de l'ingénierie de systèmes complexes et du Model Based Systems Engineering, ce travail propose dans un premier temps une méthode de conception se fondant sur les principes normatifs de l'IS et respectant les habitudes et les pratiques courantes en ingénierie de Framatome (ex AREVA NP). Cette méthode a pour vocation de formaliser et assurer le passage des exigences aux architectures et d'améliorer les capacités de vérification des modèles développés lors de la conception. Elle s'organise autour de langages de modélisation interopérables, couvrant l'ensemble des processus promus par l'IS. La méthode proposée est appliquée sur le système dont les performances sont les plus limitantes dans le contexte de l'augmentation de flexibilité : le *Core Control*. Ce composant algorithmique du réacteur assure le contrôle des paramètres de fonctionnement du cœur : la température moyenne, la distribution axiale de puissance et la position des groupes de grappes.

La thèse propose ensuite des contributions techniques relevant du champ de l'Automatique. Il s'agit de concevoir un système de régulation répondant aux exigences issues de la formalisation IS évoquée ci-dessus. La solution proposée repose sur une stratégie de commande hiérarchisée, utilisant la complémentarité des approches dites de commande multi-objectif, de séquencement de gains et enfin de commande prédictive. Un modèle de réacteur nucléaire simplifié innovant est développé à des fins de conception du système de régulation et de simulations intermédiaires. Les résultats obtenus ont montré les capacités d'adaptation de la démarche proposée à des spécifications diverses. Les performances atteintes sont très encourageantes lorsque évaluées en simulation à partir d'un modèle réaliste et comparées à celles obtenues par les modes de pilotages classiques.

Abstract

In the event of increasing renewable energies in France, this thesis proposes to study the flexibility increase of pressurized water reactors (PWR) throughout two different engineering disciplines aiming at complementary objectives: Systems Engineering (SE) and Control theory.

In a first phase, within the frame of complex systems design and Model Based Systems Engineering, this work proposes a SE method based on SE standard principles and compliant with Framatome's (ex AREVA NP) practices and addressing the revealed issues. This SE contribution is twofold: formalize and ensure the path from requirements to system architectures and enhance the capabilities of models verification. The method revolves around interoperable modeling languages, covering the SE processes: from requirement engineering to system architecture design. The method is applied to the system, which performances are the most limiting in the context of flexibility increase: the *Core Control*. This algorithmic reactor component ensures the control of: the average coolant temperature, the axial offset and the rod bank position, three of the core main functioning parameters.

In order to provide a technical contribution relying on some advanced control methodologies. It consists in designing a control system meeting the requirements defined by the SE method application. The proposed solution is in a two-layer control strategy using the synergies of multi-objective control, gain-scheduling and predictive control strategies. A simplified innovative nuclear reactor model is employed to conceive the control algorithm, simulate and verify the developed models. The results obtained from this original approach showed the ability to adapt to various specifications. Compared to conventional core control modes, the simulation results showed very promising performances, while meeting the requirements, when evaluated on a realistic reactor model.

Table des matières

Partie I. Introduction Générale 23
1 Contexte et problématiques 25
2 Etat de l'art IS 29
2.1 Introduction à l'Ingénierie Système (IS) 29
2.2 Les pratiques de Framatome en IS 36
2.3 L'existant en IS 38
2.3.1 L'écriture structurée d'exigences 38
2.3.2 La traçabilité des exigences 40
2.3.3 Ingénierie des architecture (IA) 41
2.3.4 Comportement 41
2.3.5 MBSE 42
2.3.6 La vérification et la validation des modèles 43
2.4 Positionnement des travaux de la partie IS 43
3 Etat de l'art Automatique 49
3.1 Modélisation et contrôle des REP 49
3.1.1 Modélisation des REP 49
3.1.2 Contrôle des REP 50
3.2 Les modes de pilotage de Framatome 52
3.2.1 Objectifs de commande et actionneurs considérés 52
3.2.2 Le Mode A 53
3.2.3 Le mode G 54
3.2.4 Le Mode T 54
3.3 Les stratégies de commande avancées retenues dans la thèse 59
3.3.1 La commande multi-objectif 62
3.3.2 La commande à gains-séquencés 65
3.3.3 La commande prédictive 67
Partie II. Méthode d'Ingénierie Système 73
4 Introduction de la partie IS 75
5 Les Concepts 79
5.1 Le méta modèle 80
5.2 Le système 80
5.2.1 Le composant 81
5.2.2 L'interface 81
5.2.3 Composant logique 82
5.3 L'Exigence 84
5.3.1 Le modèle d'exigences 84

5.3.2 Propriétés des exigences....85
5.3.3 Analyse d'impact des exigences....86
5.3.4 Les types d'exigences....86
5.3.5 Les attributs de l'exigence....87
5.3.6 Les relations inter-exigences....88
5.3.7 Les relations entre exigences et autres éléments de modélisation....90
5.4 La Fonction....90
5.4.1 Le Flux....91
5.4.2 L'Item....92
5.4.3 Le service....92
5.5 Architecture....93
5.6 Bloc Système....94
5.7 Mode Opérationnel et Transition....95
5.8 Le scénario opérationnel....97
5.9 Notion de vue....97
5.10 Les propriétés....99
5.11 La sémantique opérationnelle d'un DSML....101
5.11.1 Le moniteur de DSML....101
5.11.2 Le Blackboard....102
5.11.3 La sémantique opérationnelle....103
6 Les DSML....105
6.1 DSML d'Architecture....106
6.2 DSML de Modélisation d'Exigences....115
6.3 DSML de Séquences....123
6.4 DSML de Dépendance....127
6.5 DSML de Modes & Transitions....132
6.6 DSML Fonctionnel....138
6.7 DSML de Spécification Fonctionnelle (SF)....146
7 La démarche opératoire....151
8 Cas d'utilisation – Le *Core Control*....155
8.1 Les exigences du Core Control....155
8.2 Définition de la vue contextuelle du *Core Control*....157
8.3 Scénarios opérationnels du Core Control....159
8.4 Définition des modes du Core Control....161
8.5 L'architecture fonctionnelle du Core Control....161
8.6 Le diagramme de spécification....163
8.7 Liens entre les diagrammes....165
9 Passage à l'ingénierie de métier : l'automatique....167

Partie III. Contrôle-Commande du Cœur des REP : une Conception Revisitée 169

10 Stratégie de commande 171
10.1 Description du système à contrôler 171
10.2 Cahier des charges de la régulation 173
10.2.1 Limites de conditions d'opérations (LCO) 174
10.2.2 La performance 175
10.2.3 Sollicitations des actionneurs 175
10.2.4 Robustesse 176
10.2.5 Priorité des sorties à contrôler 176
10.2.6 Définitions des références pour le contrôle 176
10.3 Choix de la stratégie de commande employée 178
10.3.1 Commande simplifiée : régulation par les groupes de grappes 178
10.3.2 Commande avec variation de débit 180
10.3.3 Commande retenue dans la thèse : commande hiérarchisée à deux niveaux 181
10.4 Présentation de la démarche de conception et de paramétrage du régulateur 189
11 Modélisation pour la commande 190
11.1 Modélisation thermo-hydraulique 191
11.1.1 Modèle thermo-hydraulique du cœur 191
11.1.2 Modèle thermo-hydraulique du GV 192
11.1.3 Du GV à la turbine 192
11.1.4 Modélisation des branches chaude et froide 192
11.2 Modélisation neutronique du cœur 193
11.2.1 Notion de réactivité 193
11.2.2 Equations de la neutronique 194
11.2.3 Relation puissance et densité de neutrons 194
11.2.4 Effets sur la réactivité 195
11.3 Modélisation de l'Axial Offset 202
11.4 Modélisation du retard de bore 204
11.5 Le modèle non-linéaire du réacteur 206
11.6 Modèle point quasi-LPV 207
11.6.1 Caractérisation des non-linéarités 207
11.6.2 Représentation d'état du modèle point non-linéaire 209
11.6.3 Linéarisation de la représentation d'état non-linéaire 210
11.6.4 Mise sous la forme quasi-LPV du modèle 213
11.7 Conclusions sur la modélisation du réacteur 217
12 Conception et paramétrage du Régulateur Multi-Objectif 218
12.1 Structure du régulateur multi-objectif 218
12.2 Traduction du cahier des charges pour la commande 220

12.2.1 Démarche pour la prise en compte de contraintes temporelles 221
12.2.2 Critère H∞ sur les LCO 224
12.2.3 Critère H∞ sur la robustesse 225
12.2.4 Critères H∞ sur la sollicitation des commandes 226
12.2.5 Critère H2 pour la performance 227
12.3 Formalisation du problème de commande multi-objectif 230
12.3.1 Forme standard de commande 230
12.3.2 Méthode et outils de résolution du problème de commande 233
12.4 Synthèse finale du régulateur à gains-séquencés 233
12.4.1 Synthèse à un point de fonctionnement 233
12.4.2 Séquencement du régulateur et points de fonctionnement considérés 235
12.4.3 Méthode de détermination du régulateur à gains séquencés 237
12.4.4 Simulation et analyse du régulateur à gains séquencés 240
12.5 Conclusions sur la partie régulateur multi-objectif 241
13 Conception et paramétrage du Superviseur 244
13.1 Modèle non-linéaire utilisé pour la commande du superviseur 245
13.2 Formalisation du problème de commande 246
13.2.1 Réglage du superviseur 246
13.2.2 Définition des commandes considérées 247
13.2.3 Définition de la fonction objectif du superviseur 249
13.2.4 Définition des contraintes du superviseur 251
13.3 Algorithme développé 252
13.3.1 Solveur utilisé 252
13.3.2 Principe de fonctionnement de l'algorithme 253
13.4 Simulation et Analyse du superviseur 254
13.4.1 Suivi de rampe à 5%PN/min superviseur seul 254
13.4.2 Rampe et réglage de fréquence superviseur seul 255
13.5 Conclusions sur la partie superviseur 256
14 Simulation et analyse du RMOSC 257
14.1 Rampe de puissance du RMOSC 257
14.2 Step de puissance du RMOSC 261
14.3 Rampe et réglage de fréquence du RMOSC 264
14.4 Conclusions de la partie simulation du RMOSC 268
14.5 Comparaison du RMOSC avec le mode T 269
14.5.1 Scénarios de simulation du mode T 269
14.5.2 Comparaison des résultats de simulation du mode T et du RMOSC 269
14.5.3 Conclusions sur la simulation 271
Partie IV. Conclusions et Perspectives de la Thèse 273
15 Conclusions 275

16 Perspectives 279
16.1 Perspectives dans le domaine de l'Automatique appliquée au contrôle du réacteur 279
16.2 Perspectives IS 280
Annexes 283
Annexe A. Blocs de spécification Framatome 284
Annexe B. Boilerplates considérées 285
Annexe C. Robustesse MIMO 286
Annexe D. Conditionnement du système 288
Annexe E. Du débit de dilution/borication à la commande de bore 289
Annexe F. Modèle multi-mailles 293
Annexe G. Démonstration du développement en série de Taylor de l'équation non-linéaire 296
Annexe H. Commande LQ 297
Annexe I. Résultats de simulation du régulateur à gains séquencés seul 298
Annexe J. Résultats de simulation Superviseur seul. Rampe de puissance 100%PN à 60%PN à 5%PN/min 301
Annexe K. Passage des exigences aux spécification 304
Annexe L. Allocation des exigences 305
Annexe M. Interactions entre les DSML 306
Annexe N. Diagrammes de spécification 307
Annexe O. Diagramme fonctionnel du mode manuel 309
Annexe P. Diagrammes fonctionnels du Core Control 310
Annexe Q. Résultats de simulation mode T – Rampe 100-60-100%PN à 5%PN/min 311
Annexe R. Résultats de simulation mode T – Réglage de fréquence 314

Liste des figures

Figure 1: Décomposition en sous-systèmes 31
Figure 2 : Cycle en V selon [ISO 15288] 32
Figure 3 : Les différentes couches de méta modélisation d'OMG 34
Figure 4 : Exemple de PBS 36
Figure 5 : Cycle RADIV 37
Figure 6: Tableau récapitulatif de l'état de l'art IS 45
Figure 7: Principe de la régulation du mode A 54
Figure 8 : Mouvement typique des groupes lors d'une variation de puissance. Figure tirée de [Grossetête, 2007] 56
Figure 9: Programme de température moyenne 57
Figure 10: Principe du contrôle de l'AO. Figure tirée de [Grossetête, 2007] 57
Figure 11: Principe du contrôle du Pmax. Figure tirée de [Grossetête, 2007] 58
Figure 12: Forme standard de commande 59
Figure 13: Forme standard de commande dans le cas mixte $H2/H\infty$ 63
Figure 14: Base de fonctions pour la commande prédictive 68
Figure 15: Principe de la commande prédictive 69
Figure 16: Eléments de composition d'une méthode 76
Figure 17: Positionnement de la méthode dans le RADIV 77
Figure 18: Méthode proposée vs. PBS et référentiels d'exigences 77
Figure 19: Démarche employée pour développer la méthode 78
Figure 20: Exemple de Méta modèle 80
Figure 21: Le concept Système dans le méta modèle proposé 81
Figure 22: Méta modèle Système et Composant 81
Figure 23: Exemple d'interface – Cœur 82
Figure 24: Exemple d'architecture logique- Figure réalisées par l'outil Capella 83
Figure 25: Exemple de PBS - Figure réalisées par l'outil Capella 84
Figure 26: Schéma de principe du modèle d'exigences 85
Figure 27: Exemple pour le raffinement d'une exigence 89
Figure 28: Méta modèle de l'Exigence et ses relations 89
Figure 29: Méta modèle Fonction Flux Item 91
Figure 30: Méta modèle de la vue contextuelle 92
Figure 31: Représentation des liens entre les architectures 94
Figure 32: Illustration du bloc système et des relations avec les blocs système de différents niveaux 95
Figure 33: Méta modèle des modes et transitions 97
Figure 34 : Moniteur de DSML 102
Figure 35: Illustration DSML / Moniteur / Blackboard 104
Figure 36: Synthèse des différentes vues adressées par le DSML de contexte 108
Figure 37: Méta modèle du diagramme de contexte 111
Figure 38: Exemple de diagramme de contexte 114
Figure 39: Méta modèle pour le DSML d'exigences 117
Figure 40: Capture d'écran de l'outil l'outil développé pour l'Ingénierie des exigences (avec Xtext) 122
Figure 41: illustration de la relation de dépendance 128
Figure 42: Représentation avec le temps 128
Figure 43: Exemple de diagramme de dépendance n°1. Diagramme réalisé avec l'outil Matlab Simulink : Lookup Table 130
Figure 44: Exemple de diagramme de dépendance (temporel) n°2. Diagramme réalisé avec l'outil Matlab Simulink : Signal Builder 131
Figure 45: Méta modèle de la vue Comportementale (M/T) 134
Figure 46: Diagramme de Modes et Transition 137
Figure 47: Exemple de diagramme eFFBD standard 139
Figure 48: Exemple de diagramme eFFDB simplifié 139

Figure 49: Méta modèle du diagramme fonctionnel....142
Figure 50: validation des transformations effectuées par une fonction [Meinadier, 1998] 144
Figure 51: Exemple d'un bloc dans le langage de spécification I&C Framatome (*Limiter*) 146
Figure 52: Exemple du langage de spécification I&C Framatome 149
Figure 53: Liens entre les différents langages utilisés....151
Figure 54: Démarche opératoire....154
Figure 55 : Exigences sur allouées au *Core Control* 156
Figure 56: Référentiel d'exigences et SOI....157
Figure 57: Diagramme de contexte....158
Figure 58: Illustration du diagramme de séquence 159
Figure 59: Illustration du diagramme de dépendance (avec le temps) 160
Figure 60: Illustration du diagramme de dépendance (version sans le temps) 160
Figure 61 : Diagramme de modes et transition du Core Control....161
Figure 62: Architecture fonctionnelle du Core Control dans le mode Automatique....162
Figure 63: Diagramme de spécification fonctionnelle de la fonction F0001 165
Figure 64: Schéma de principe de la régulation....173
Figure 65: Domaine de fonctionnement autorisé 174
Figure 66: Profil de la température moyenne de référence 177
Figure 67: Position de référence pour Pbank (cumulée) 178
Figure 68: Organigramme des travaux 181
Figure 69: Commande hiérarchisée 183
Figure 70: Schéma de principe de la régulation adoptée 184
Figure 71: Illustration de la différence de trajectoire régulateur prédictif et multi-objectif....186
Figure 72: Organisation de la partie automatique 189
Figure 73: Illustration de l'efficacité (Kp) d'un quadruplet de barre en fonction de son insertion (palier TSN – quadruplet PA) – BU150 – E18....196
Figure 74: Illustration de l'effet modérateur 199
Figure 75: Efficacité de l'effet modérateur en fonction de la température....200
Figure 76: Illustration de l'effet doppler....201
Figure 77: Chaine de réaction produisant le xénon 135 (les rendements sont relatives aux fissions de l' $235U$ et les durées sont les périodes radioactives)....201
Figure 78: Effet de la barre PA sur l'AO (Données SMART – Palier TSN – Cycle E18 – BU150 – AO initial -3%) 203
Figure 79: Effet de la température branche chaude sur l'AO à température branche froide constante ($Tf = 295\ °C$) (Données en début de vie) 203
Figure 80 : Effet de la température branche froide sur l'AO à température branche chaude constante ($Tc = 329\ °C$) (Données en début de vie) 204
Figure 81: Réponse indicielle de l'approximant de Laguerre....205
Figure 82: Illustration de l'espace d'état....209
Figure 83: Modèle LPV retardé 216
Figure 84: Schéma standard de commande 219
Figure 85: Forme du régulateur multi-objectif....219
Figure 86: Scénario pire cas de variation de puissance turbine 222
Figure 87: Comparaison signal créneau et décomposition en série de Fourier ordre 3....222
Figure 88: Bode du système considéré....224
Figure 89: Modèle augmenté multi-objectif pour le problème $H2/H\infty$....232
Figure 90: Résultats de simulation d'un régulateur multi-objectif LTI autour d'un point de fonctionnement 234
Figure 91: Position de référence des barres Pbank et Hbank pour le séquencement du modèle quasi-LPV 236
Figure 92: Synthèse multi-modèle par zone....238
Figure 93: Séquences pour la synthèse LPV 239
Figure 94: Schéma de séquencement du régulateur à gains séquencés K....240
Figure 95 : Schéma global de régulation 244

Figure 96: Commandes constantes par morceaux du superviseur 249
Figure 97: Schéma de principe de l'algorithme du superviseur 254
Figure 98: Tracé de la puissance turbine – Rampe de puissance 100%PN à 60%PN et réglage de fréquence 255
Figure 99: Tracé de la température moyenne - Rampe de puissance 100%PN à 60%PN et réglage de fréquence 256
Figure 100: Tracé de la puissance turbine – Rampe de puissance 100%PN à 60%PN 258
Figure 101: Tracé de la température moyenne - Rampe de puissance 100%PN à 60%PN 259
Figure 102: Tracé de l'AO - Rampe de puissance 100%PN à 60%PN 259
Figure 103: Tracé de la position de Pbank - Rampe de puissance 100%PN à 60%PN – Descente de charge 259
Figure 104: Tracé de la position de Pbank - Rampe de puissance 100%PN à 60%PN – Remontée en charge 260
Figure 105: Tracé de la position des groupes - Rampe de puissance 100%PN à 60%PN 260
Figure 106: Tracé des vitesses et débit des actionneurs - Rampe de puissance 100%PN à 60%PN 260
Figure 107: Xénon vs. Concentration en bore - Rampe de puissance 100%PN à 60%PN 261
Figure 108: Tracé de la puissance turbine – Step de puissance 100%PN à 90%PN 262
Figure 109: Tracé de la température moyenne - Step de puissance 100%PN à 90%PN 262
Figure 110: Tracé de l'AO - Step de puissance 100%PN à 90%PN 263
Figure 111: Tracé de la position de Pbank - Step de puissance 100%PN à 90%PN 263
Figure 112: Tracé de la position des groupes - Step de puissance 100%PN à 90%PN 263
Figure 113: Tracé des vitesses et débit des actionneurs - Step de puissance 100%PN à 90%PN 264
Figure 114: Xénon vs. Concentration en bore - Step de puissance 100%PN à 90%PN 264
Figure 115: Tracé de la puissance turbine - Rampe 100%PN à 70%PN & Réglage de fréquence 266
Figure 116: Tracé de la température moyenne - Rampe 100%PN à 70%PN & Réglage de fréquence 266
Figure 117: Tracé de l'AO - Rampe 100%PN à 70%PN & Réglage de fréquence 266
Figure 118: Tracé de la position de Pbank - Rampe 100%PN à 70%PN & Réglage de fréquence 267
Figure 119: Tracé de la position des groupes - Rampe 100%PN à 70%PN & Réglage de fréquence 267
Figure 120: Tracé des vitesses et débit des actionneurs - Rampe 100%PN à 70%PN & Réglage de fréquence 267
Figure 121: Xénon vs. Concentration en bore - Rampe 100%PN à 70%PN & Réglage de fréquence 268
Figure 122: Marge de robustesse MIMO 286
Figure 124: Schématisation du problème de variation de concentration de bore dans le primaire 290
Figure 125: Schéma bloc de la conversion de commande de bore 292
Figure 126: Illustration du principe de la modélisation 293
Figure 127: Illustration du couplage thermo-hydraulique des mailles 295
Figure 128: Tracé de la puissance turbine – Rampe de puissance 100%PN à 60%PN à 5%PN/min 298
Figure 129: Tracé de la température moyenne - Rampe de puissance 100%PN à 60%PN à 5%PN/min 298
Figure 130: Tracé de l'AO - Rampe de puissance 100%PN à 60%PN à 5%PN/min 298
Figure 131: Tracé de la position de Pbank - Rampe de puissance 100%PN à 60%PN à 5%PN/min 299
Figure 132: Tracé de la position des groupes - Rampe de puissance 100%PN à 60%PN à 5%PN/min 299
Figure 133: Tracé des vitesses et débit des actionneurs - Rampe de puissance 100%PN à 60%PN à 5%PN/min 299
Figure 134: Xénon vs. Concentration en bore - Rampe de puissance 100%PN à 60%PN à 5%PN/min 300
Figure 135: Tracé de la puissance turbine – Rampe de puissance 100%PN à 60%PN à 5%PN/min 301
Figure 136: Tracé de la température moyenne - Rampe de puissance 100%PN à 60%PN à 5%PN/min 301
Figure 137: Tracé de l'AO - Rampe de puissance 100%PN à 60%PN à 5%PN/min 301
Figure 138: Tracé de la position de Pbank - Rampe de puissance 100%PN à 60%PN à 5%PN/min 302
Figure 139: Tracé de la position des groupes - Rampe de puissance 100%PN à 60%PN à 5%PN/min 302
Figure 140: Tracé des vitesses et débit des actionneurs - Rampe de puissance 100%PN à 60%PN à 5%PN/min 302
Figure 141: Xénon vs. Concentration en bore - Rampe de puissance 100%PN à 60%PN à 5%PN/min 303
Figure 142: Illustration du passage du système aux spécifications 304
Figure 143 Illustration de l'allocation des exigences 305

Figure 144: Schéma global des interactions entre les DSML 306
Figure 145 : Diagrammes de Spécification fonctionnelle des fonctions F0002 et F0003.....307
Figure 146:Diagrammes de spécification fonctionnelle des fonctions F0004 et F0005 307
Figure 147: Diagramme de spécification fonctionnelle de la fonction F0020 308
Figure 148: Diagramme de spécification fonctionnelle de la fonction F0010 308
Figure 149: Diagramme fonctionnel du mode Manuel du Core Control 309
Figure 150: Les diagrammes fonctionnels du Core Control.....310
Figure 151: Simulation Mode T - Puissance turbine - Rampe 100-60-100%PN à 5%/min 311
Figure 152: Simulation Mode T – Température moyenne - Rampe 100-60-100%PN à 5%/min 311
Figure 153: Simulation Mode T – Axial Offset - Rampe 100-60-100%PN à 5%/min.....312
Figure 154: Simulation Mode T – Groupes de grappes Pbank et Hbank - Rampe 100-60-100%PN à 5%/min 312
Figure 155: Simulation Mode T – Xénon et Concentration en Bore - Rampe 100-60-100%PN à 5%/min...313
Figure 156: Simulation Mode T – Puissance turbine - Réglage de fréquence 314
Figure 157: Simulation Mode T – Température moyenne - Réglage de fréquence.....314
Figure 158: Simulation Mode T – Axial Offset - Réglage de fréquence 315
Figure 159: Simulation Mode T – Groupes de grappes Pbank et Hbank - Réglage de fréquence 315
Figure 160: Simulation Mode T – Xénon et Concentration en Bore - Réglage de fréquence 316

Acronymes, Notations et Nomenclature

Acronymes

ACT	Average Coolant Temperature (température moyenne du cœur)
AFIS	Association Française d'Ingénierie Système
AO	Axial Offset
ASN	Autorité de sûreté nucléaire
DSML	Domain Specific Modeling Language (langage de modélisation spécifique à un domaine)
GV	Générateur de Vapeur
IA	Ingénierie des Architectures
IE	Ingénierie des Exigences
IHM	Interface Homme Machine
INCOSE	International Council on Systems Engineering
IS	Ingénierie Système
I&C	Instrumentation and Control (Contrôle et instrumentation)
LCO	Limiting Condition Operation (limites de conditions d'opération)
LMI	Linear Matrix Inegalities (Inégalité matricielle linéaire)
LPV	Linear Parameter Variant (linéaire à paramètres variants)
LQ	Linear Quadratic (linéaire quadratique)
LQG	Linear Quadratic Gaussian (Linéaire Quadratique Gaussien)
LTI	Linear Time Invariant (linéaire à temps invariant)
MBSE	Model Based Systems Engineering (ingénierie système basée sur des modèles)
MIMO	Multiple Inputs Multiple Outputs (plusieurs entrées plusieurs sorties
PBS	Product Breakdown Structure (décomposition produit)
pcm	Pour Cent Mille
PE	Pas de grappes extraits
PI	Proportional Integral (proportionnel dérivée)
PID	Proportional Integral Derivative (proportionnel intégrale dérivée)
PN	Puissance Nominale
ppm	Partie Par Million
REP	Réacteur à eau pressurisée
SF	Spécification Fonctionnelle
SISO	Single Input Single Output (une entrée et une sortie)
SOI	System of Interest (Système d'intérêt)
STUD	System Used To Design (système pour faire)
V&V	Vérification & Validation (vérification et validation)

Notations

$\mathbb{R}$	Corps des nombres réels
$(.)^T$	Transposée d'une matrice
$\sigma(.)$	Vecteur des valeurs singulières
$\|.\|_2$	Norme H_2
$\|.\|_\infty$	Norme H_∞
$T_{y\to u}$	La fonction de transfert en l'entrée u et la sortie y
s	Variable de Laplace

Nomenclature des signaux

AO	Distribution axiale de puissance
c	Concentration des précurseurs dans le cœur
C_b	Concentration en bore
C_P	Capacité thermique massique de l'eau du circuit primaire

d_P	Signal de perturbation de puissance turbine
D_P	Puissance turbine réelle
D_U	Programme de suivi de charge
H_{bank}	Position du groupe de grappes Hbank
K	Régulateur multi-objectif à gains séquencés
K_n	Coefficient de conversion de puissance
n	Densité de neutrons dans le cœur
N_P	Horizon de prédiction
N_C	Horizon de commande
P_{bank}	Position du groupe de grappes Pbank
P_K	Puissance cœur
Q_P	Débit d'eau du circuit primaire
T_c	Température branche chaude
T_c^{GV}	Température branche chaude au niveau du GV
T_f	Température branche froide
T_f^{GV}	Température branche froide au niveau du GV
T_m	Température moyenne
T_S	Pas d'échantillonnage du superviseur
u_K	Vecteur des commandes du régulateur multi-objectif
u_Q	Vitesse de variation, assimilable au débit, de bore
$\overline{u_Q}$	Commande de bore retardée
U	Vecteur des commandes réelles appliquées au processus
U_S	Vecteur des commandes de références du superviseur
U_S^j	Vecteur des valeurs des commandes fixes sur le pas d'échantillonnage j
vH_{bank}	Vitesse du groupe de grappes Hbank
vP_{bank}	Vitesse du groupe de grappes Pbank
ρ	Réactivité
τ_{AO}	Constante de temps de l'AO
τ_{bc}	Constante de temps de la branche chaude
τ_{bf}	Constante de temps de la branche froide
τ_{co}	Constante de temps du cœur
τ_{GV}	Constante du temps du GV

Publications

[Lemazurier *et al.*, 2016] L. Lemazurier et A. Grossetete. PHD Advanced Control/SE methodology - 2nd annual report. *Framatome Internal Publication*, ref: D02-DTIPC-F-17-0022, 2016.

[Lemazurier *et al.*, 2017a] L. Lemazurier, M. Yagoubi, Ph. Chevrel and A. Grossetête. Multi-Objective H_2/H_∞ Gain-Scheduled Nuclear Core Control Design. *The 20th World Congress of the International Federation of Automatic Control,* 50(1), pages 3256-3262, 2017.

[Lemazurier *et al.*, 2017b] L. Lemazurier, V. Chapurlat et A. Grossetête. An MBSE Approach to Pass from Requirements to Functional Architecture. *The 20th World Congress of the International Federation of Automatic Control,* 50(1), pages 7260-7265, 2017.

Lemazurier *et al.*, 2018a] L. Lemazurier, Ph. Chevrel, M. Yagoubi and A. Grossetête. A Multi-Objective Nuclear Core Control Performing Hot and Cold Leg Temperature Control. *European Control Conference,* 2018.

[Lemazurier *et al.*, 2018b] L. Lemazurier, Ph. Chevrel, M. Yagoubi and A. Grossetête. An Advanced Multi-Objective Nuclear Core Control for PWR Reactors. *Control Engineering Practice,* 2018*, soumis.*

Partie I.
Introduction Générale

1 Contexte et problématiques

Le contexte du nucléaire en France et dans le monde se durcit au fil des années, au rythme de différents évènements marquants tel que l'accident nucléaire de Fukushima en 2011. Les contraintes émises par les entités chargées de la régulation du nucléaire tel que l'ASN (Autorité de Sûreté du Nucléaire française) sont de plus en plus strictes. De plus, compte tenu de la part importante du nucléaire dans le mix énergétique français, l'introduction massive de sources d'énergies renouvelables (éolien et solaire) sur le réseau électrique peut induire un besoin supplémentaire de flexibilité de production d'énergie, se portant sur les centrales nucléaires. La flexibilité traduit la capacité d'une unité de production d'électricité à adapter sa production. Les énergies renouvelables ont la particularité d'être intermittentes en fonction du moment ou même des aléas climatiques. En l'absence de système de stockage de masse d'énergie cette hausse de flexibilité induit une charge accrue de réglage de la puissance mise à disposition par les générateurs traditionnels, y compris les réacteurs nucléaires en France. On peut donc s'attendre à un accroissement des besoins en performance des centrales nucléaires.

Des exigences de sûreté plus contraignantes et des performances accrues sont autant de facteurs augmentant la complexité des installations, qui plus est dans le cadre d'un système complexe comme les réacteurs à eau pressurisée (REP). C'est dans ce contexte que de nombreux industriels se penchent sur des méthodes de conception innovantes de systèmes telles que l'Ingénierie Système (IS). L'IS est une approche pluridisciplinaire, ayant pour but de faire collaborer un ensemble d'acteurs métier différents autour de la conception de systèmes complexes. Cette approche se base sur un ensemble de processus aujourd'hui standardisés et souvent éprouvés dans plusieurs secteurs industriels. Ces processus couvrent l'ensemble des activités nécessaires pour concevoir et réaliser des systèmes en partant de l'élicitation des besoins client à la vérification et validation d'une solution retenue compte tenu de l'ensemble du cycle de vie du système.

Les problématiques majeures rencontrées dans la pratique de l'IS dans un cadre industriel comme le nucléaire auquel Framatome (ex AREVA NP) s'intéresse résident, d'une part, dans l'ambiguïté du contenu sémantique des exigences exprimées par les différentes parties prenantes de la conception. En effet, les exigences sont souvent rédigées dans un langage libre et non structuré. La compréhension et, de fait, l'interprétation et l'utilisation de ces exigences pour guider ensuite la conception sont alors subjectives et dépendent de la lecture de chacun. D'autre part, il apparait souvent difficile de justifier et de tracer les choix de conception, pris pour répondre à ces besoins. En effet, en pratique, les exigences sont allouées à la solution dans son ensemble ce qui rend la compréhension des choix faits par le concepteur difficile. De plus, dans le cas de modifications ultérieures ou de réutilisation d'éléments issus de conceptions précédentes, l'impact de la

modification des exigences est difficilement perceptible voire justifiable. Enfin, il apparait que les erreurs de modélisation, même mineures, peuvent engendrer des coûts importants (davantage encore considérant l'ingénierie nucléaire où les études de sûreté sont particulièrement longues). La vérification des modèles est cependant souvent sous-estimée dans les outils d'IS mis en place pour les industriels.

Dans ce contexte nous proposons une **première contribution, méthodologique,** consistant en une méthode de conception basée sur l'IS, dans le cadre de l'ingénierie de systèmes nucléaires complexes. Cette contribution se concentrera sur :

- Le **passage des besoins et des exigences vers la solution.**
- La **vérification** de la cohérence des différents **modèles.**

Une seconde contribution est abordée dans la thèse. Elle permet à la fois d'apporter une solution **technique** au problème de flexibilité mais aussi d'illustrer la méthode d'IS développée au préalable. La flexibilité des REP est généralement limitée par le pilotage du cœur. Le pilotage consiste à contrôler à minima :

- La température moyenne du cœur.
- La distribution axiale de puissance.

Ce pilotage est réalisé par différents moyens d'action : les grappes de contrôle et la modification de la concentration en bore du circuit primaire. Différents modes de pilotage du cœur existent actuellement, comme par exemple le mode A, le mode G et le mode T (ce dernier étant utilisé dans le cadre de l'EPR, le dernier modèle de réacteur développé par Framatome). Or, en raison de l'augmentation des performances due à l'introduction des énergies renouvelables, il est fortement probable qu'une amélioration des modes de pilotage soit nécessaire.

Les difficultés majeures du contrôle du cœur sont, d'une part, liées au caractère non-linéaire des modèles décrivant le comportement du système. En effet, les lois de commande actuelles, comme celles utilisées dans le mode T, bénéficient de logiques d'adaptation sophistiquées afin de contrôler le cœur sur l'ensemble du domaine de fonctionnement. D'autre part, bien que Framatome dispose d'un code de simulation multi-physique précis et qualifié, il est difficilement utilisable à des fins conceptuelles pour une loi de commande. De plus, les dynamiques du cœur sont disparates (constantes de temps très variées), les variations de puissance ne sont connues que partiellement à l'avance. Enfin, certaines parties de l'existant ne peuvent pas être remises en question, en particulier le groupement des grappes.

L'automatique moderne offre un large panel de méthodologies de commande de systèmes complexes, permettant d'appréhender leur comportement. En particulier,

la formulation des objectifs de commande sous la forme d'un problème d'optimisation a permis un essor considérable de la commande avancée, autant par l'amélioration des performances atteignables que par la prise en compte de contraintes de plus en plus complexes sur les asservissements. En effet, contrairement aux méthodologies classiques de commande, les méthodes récentes de l'automatique, comme les inégalités matricielles linéaires (LMI), les méthodes utilisant l'optimisation non-lisse ou encore la commande prédictive, permettent de garantir certains objectifs ou certaines contraintes de commande précis.

Ainsi, cette deuxième contribution (relative au domaine de l'automatique) consiste à **concevoir un système de régulation du cœur** permettant :

- De prendre en compte un ensemble d'exigences portant sur les performances atteignables du cœur tout en respectant les contraintes qui lui sont imposées.
- De contrôler le cœur sur l'ensemble du domaine de fonctionnement.

Ainsi, la thèse a pour vocation à faire cohabiter les domaines de l'automatique et de l'IS dans le but d'obtenir une solution cohérente pour couvrir les besoins de Framatome dans le cadre de la conception et du pilotage des cœurs des futures installations. L'articulation entre ces deux domaines permettra notamment d'illustrer le passage du cadre structurel et méthodologique posé par l'IS à l'ingénierie métier formalisée par la méthode proposée d'automatique.

Afin de répondre aux problématiques formulées ci-dessus, nous procédons de la manière suivante :

Tout d'abord, dans la Partie I, nous effectuons un état de l'art des deux domaines (IS et automatique) dans lesquels nous présentons les pratiques de Framatome ainsi que l'ensemble des travaux jugés cohérents avec les problématiques annoncées. La Partie II a pour but de présenter les contributions de la partie IS. En particulier, cette partie présente une méthode d'IS, i.e. les concepts et les relations entre ces concepts, les DSML, la démarche opératoire ainsi qu'un cas d'utilisation de la méthode d'IS. Cette partie conclut sur l'interaction entre les contributions IS et automatique. Dans la Partie III nous présentons la contribution automatique. Elle se fonde sur une approche hiérarchique à deux niveaux : un superviseur procédant à base de commande prédictive pour définir des trajectoires de référence et un régulateur multi-objectif et multivariable pour appréhender au plus près des exigences « métier » (formalisation mathématique à l'articulation avec l'IS) tout en tirant le meilleur parti de l'installation. Cette partie présente aussi les résultats de simulation du régulateur développé. Enfin, la Partie IV présente les conclusions et perspectives de la thèse.

2 Etat de l'art IS

Sommaire

2.1 Introduction à l'Ingénierie Système (IS) 29
2.2 Les pratiques Framatome en IS 36
2.3 L'existant en IS 38
2.3.1 L'écriture structurée d'exigences 38
2.3.2 La traçabilité des exigences 40
2.3.3 Ingénierie des architecture (IA) 41
2.3.4 Comportement 41
2.3.5 MBSE 42
2.3.6 La vérification et la validation des modèles 43
2.4 Positionnement des travaux de la partie IS 43

Ce chapitre présente un état de l'art concernant les aspects liés à la problématique de l'IS telle que perçue pour répondre aux attentes de Framatome. Nous commençons par une brève introduction de la discipline, suivie de la présentation des pratiques de Framatome et des différentes avancées de l'industriel dans ce domaine. Nous présentons ensuite les différents travaux que nous avons jugés intéressants de détailler et terminons cette partie par une discussion autour des différents éléments présentés afin de positionner les contributions de ce travail présentées dans la suite du manuscrit.

2.1 Introduction à l'Ingénierie Système (IS)

[AFIS, 2012] définit l'IS en s'inspirant d'une définition de la norme [IEEE Std 1220, 2005] comme : « une démarche méthodologique coopérative et interdisciplinaire qui englobe l'ensemble des activités adéquates pour concevoir, faire évoluer et vérifier un ensemble de produits, processus et compétences humaines apportant une solution aux besoins des parties prenantes ».

L'IS a pour objectif de passer du besoin exprimé par un client ou un utilisateur (souvent appelé maitre d'ouvrage par opposition au maitre d'œuvre chargé de la conception du système) à une solution correspondant au besoin spécifié. Dans le cadre de l'IS, cette solution se matérialise sous la forme d'un système qui peut être défini comme un ensemble d'éléments en interaction, qui, lorsqu'il est l'objet de la conception est appelé SOI. De même la nécessité de définir ce SOI engendre l'existence d'un autre système, second objet d'attention de l'IS, appelé le Système

Pour Faire (STUD en anglais pour *System Used To Design*). En effet, la conception d'un système complexe nécessite de définir et piloter un ensemble de processus regroupant des activités, des tâches et des ressources, tant techniques que humaines. C'est l'objet de ce STUD, autrement dit, le projet.

Le système passe par un ensemble d'étapes ou de stades tout au long de son existence formant le cycle de vie. Ce cycle de vie commence d'ores et déjà à la genèse du SOI, c'est-à-dire lors de sa conceptualisation et s'étend jusqu'au retrait de service du système.

Du point de vue du client, le SOI est une « boite noire » remplissant les exigences spécifiées rassemblées dans le référentiel des exigences du SOI. Cette boite noire est étudiée dans son contexte opérationnel et tout au long de son cycle de vie. La conception du système consiste alors à ouvrir cette boite noire afin de définir comment le système va réaliser les missions qui lui sont allouées dans la finalité fixée. On parle alors de perspective « boite blanche ».

L'IS promeut une approche analytique de décomposition des systèmes en sous-systèmes qui sont mis alors en interaction pour remplir la mission et la finalité du système. La perspective boite blanche correspond alors à l'étude et l'analyse des sous-systèmes du SOI et de leurs interactions. Cette approche « descendante » est présentée sur la Figure 1 où le système S1 est décomposé en sous-systèmes S11, S12 et S12 dont les interactions sont matérialisées par des flèches noires. Les sous-systèmes seront eux-mêmes décomposés de manière itérative en sous-systèmes. On parle alors de **niveau système** pour décrire les différents niveaux hiérarchiques de la décomposition des systèmes en sous-système (e.g. la Figure 1 montre deux niveaux système, celui de S1 et celui de S11, S12 et S13).

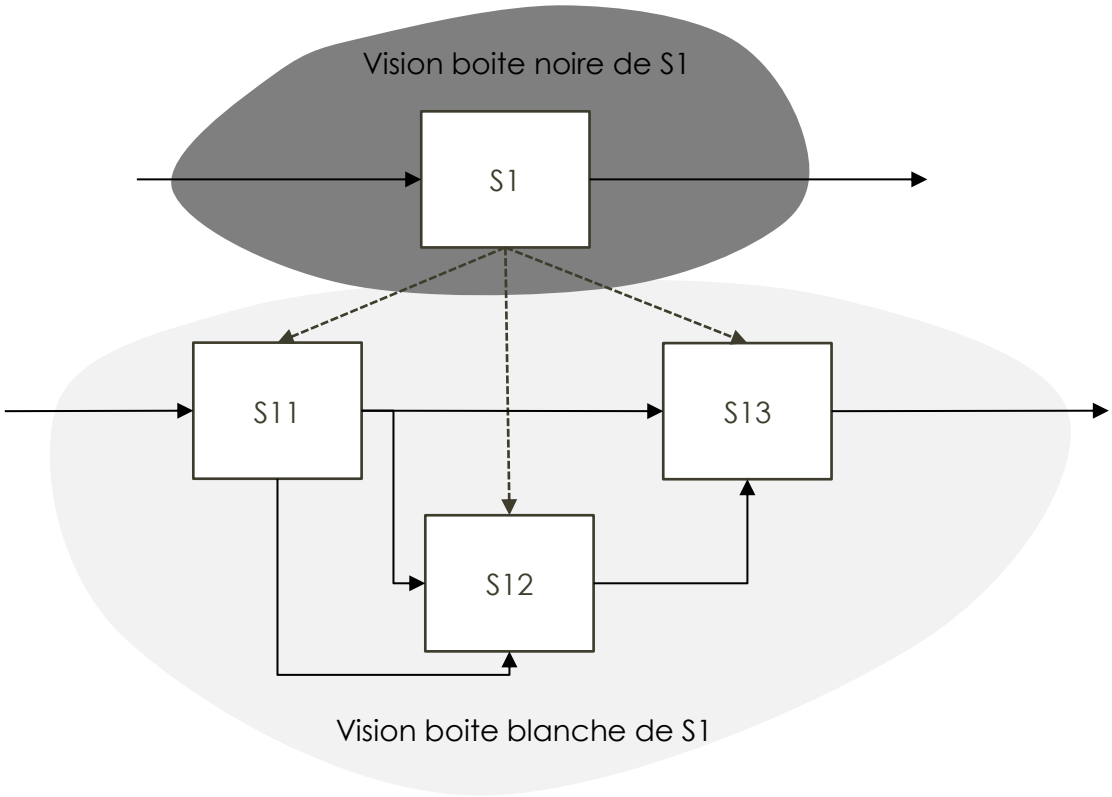


Figure 1: Décomposition en sous-systèmes

Les processus mis en œuvre au cours du projet sont largement documentés dans différentes normes comme [ISO15288, 2015] faisant référence ou encore [SEBoK, 2016].

Selon [ISO15288, 2015] les processus sont décomposés en quatre catégories :

- Les processus contractuels
- Les processus d'entreprise
- Les processus de management
- Les processus techniques

Dans notre étude, nous nous concentrerons sur les processus techniques, traitant majoritairement de la conception proprement dite du SOI. Les processus techniques sont décomposés en sous processus présentés sur la Figure 2. Nous invitons le lecteur à se reporter à [ISO15288, 2015] ou encore à [AFIS, 2012] pour une description précise des différents processus. L'ensemble de ces processus forme le communément appelé Cycle en V (cf. Figure 2).

Les processus qui nous intéressent principalement dans notre étude sont :

- L'Analyse de Mission
- La Définition des Exigences des Parties Prenantes

- La Définition des Exigences Système
- La Conception des Architectures
- L'Analyse du Système

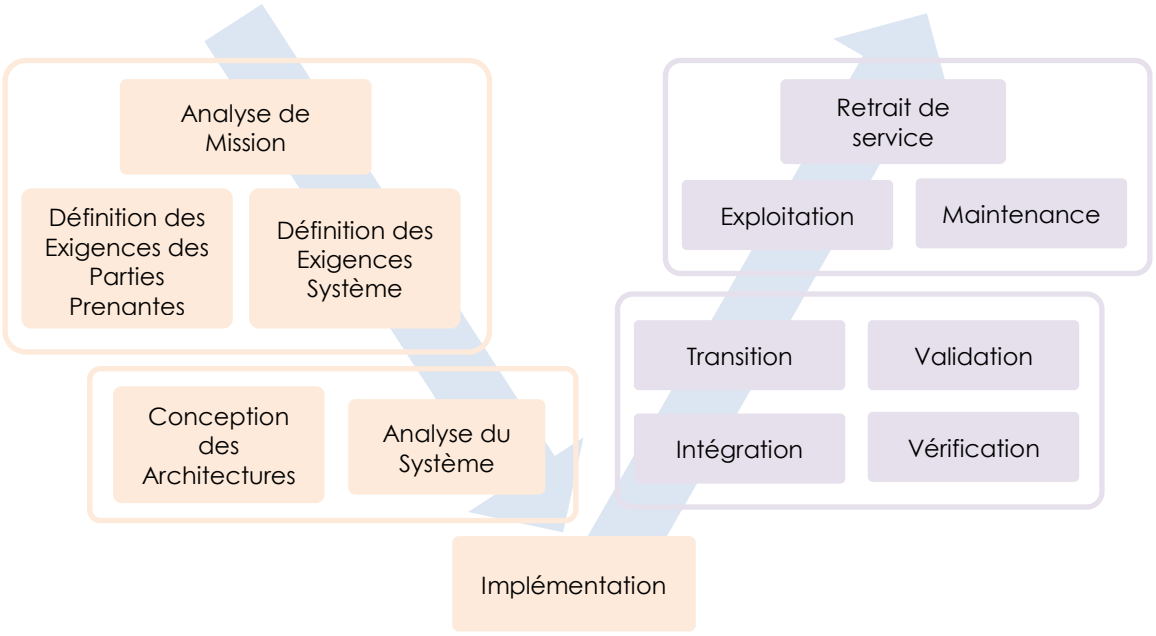


Figure 2 : Cycle en V selon [ISO 15288]

La philosophie IS propose donc, en suivant le principe de décomposition du SOI en sous-systèmes, d'appliquer les processus techniques à chaque niveau système, i.e. sur le SOI, puis sur chacun de ses sous-systèmes considérés alors généralement en parallèle, ensuite leurs sous-systèmes et ainsi de suite. Nous définirons par la suite les pratiques adoptées par Framatome cohérentes avec ces définitions.

Dans les années 2000, le monde de l'ingénierie du logiciel a apporté une vision novatrice de l'ingénierie dirigée par les modèles i.e. qu'il est possible d'abstraire tout élément du monde par la création et la manipulation de modèle de cet élément. C'est ce principe qui a inspiré le MBSE, approche dans laquelle on considère que « tout est modèle » [Greenfield et Short, 2003]. De fait, elle s'oppose à une conception visant des documents, c'est-à-dire dans laquelle le partage d'information ne se fait qu'au travers de documents d'ingénierie.

[SAGACE, 1999] et [Pénalva, 1997] définissent le modèle comme : « ce qui sert ou doit servir d'objet d'imitation pour faire ou reproduire quelque chose ». Un modèle est donc une abstraction possible d'un concept, d'un phénomène ou encore d'un système réel possédant en totalité ou en partie les mêmes caractéristiques et propriétés que l'élément qu'il représente, et pouvant se manifester sous des formes diverses, graphiques ou textuelles, et de nature mathématique ou physique.

L'intention du MBSE est de permettre aux acteurs de décrire des niveaux d'abstraction d'un système par divers types de modélisation afin :

- De mieux partager l'information entre les ingénieurs en ce qui concerne la conception des systèmes
- De formaliser les pratiques de conception au moyen de langages de modélisation
- D'augmenter la qualité et réduire les risques en détectant au plus tôt et en limitant les erreurs de conception

L'abstraction des éléments de modélisation sont décrits par ce que l'on appelle des **concepts**. Par exemple, le concept de Système est une abstraction de l'esprit permettant de définir un ensemble d'éléments en interaction.

Pour modéliser, il faut pouvoir mettre en relation ces concepts (e.g. un système échange des services avec un autres système). Pour cela, on utilise le principe de la méta modélisation telle que définie dans [MDA, 2014], [MOF, 2015] et [Overbeek, 2006] qui aboutit à la production d'un ou plusieurs méta modèles. Un méta modèle a pour but de définir l'ensemble des concepts relatifs à un domaine, voire à un aspect particulier du système ou encore une vue système. Par exemple, dans le cas MBSE, on trouve classiquement et a minima des concepts tels que le système et l'exigence, la fonction ou le composant, leur attributs i.e. les caractéristiques de chaque concept et chaque relation (e.g. le nom de la fonction ou le type d'un composant) ainsi que les relations qu'ils entretiennent (e.g. une exigence de type fonctionnelle doit être allouée à une fonction). On peut aussi structurer ce méta modèle pour faire apparaître les notions de vue système e.g. une fonction appartient à la vue fonctionnelle qui décrit le QUOI du système, ce qu'il doit faire alors que le composant sera un concept de la vue organique qui décrit le COMMENT i.e. comment sont choisis puis agencés les composants pour pouvoir exécuter les fonctions attendues.

Le modèle du système réel n'est ni plus ni moins qu'un ensemble d'instances des concepts et des relations de ce méta modèle, créées et connectées comme indiqué dans la Figure 3.

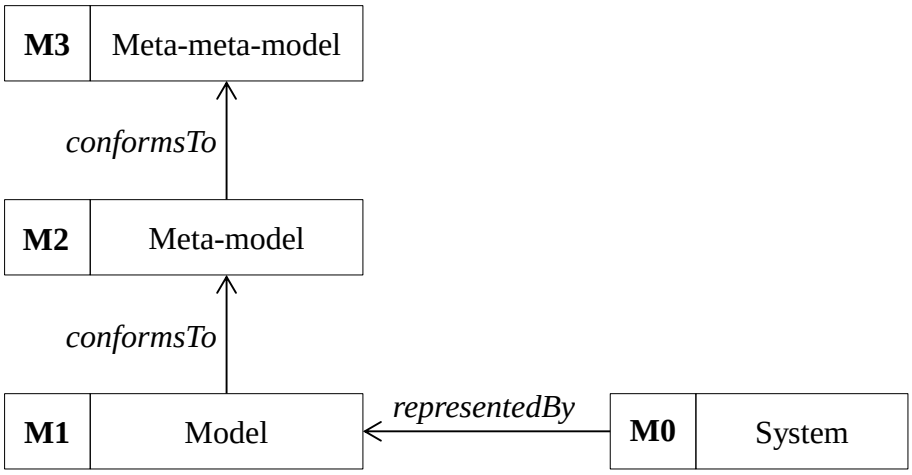


Figure 3 : Les différentes couches de méta modélisation d'OMG

En MBSE, différents langages de modélisation (DSML) sont donc nécessaires pour pouvoir créer des modèles et donc manipuler tout ou partie des concepts et des relations dans une vue particulière. On peut par exemple définir un ou plusieurs DSML fonctionnels pour la seule vue fonctionnelle.

Pour définir un DSML, il nous faut définir trois éléments essentiels :

- Un ensemble de concepts et de relations entre ces concepts qui traduisent l'intention d'un acteur métier de décrire son système pour un objectif donné et sous un point de vue particulier. Ces concepts et leurs relations sont formalisés dans la suite sous forme d'un méta modèle plus restreint mais cohérent avec les autres méta modèles qui définit ce que l'on appelle la **syntaxe abstraite du langage**. Cette syntaxe abstraite décrit aussi les contraintes de manipulation de ces concepts et relations pour ensuite construire des modèles du système (e.g. contraintes de cardinalité, d'exclusion, de totalité, qu'elles soient, ou pas, elles-mêmes conditionnées).
- Le langage doit ensuite être pourvu d'une syntaxe concrète c'est à dire d'une représentation (graphique ou textuelle) qui soit lisible et acceptable pour l'acteur métier. De fait, plusieurs **syntaxes concrètes** peuvent être définies pour la même syntaxe abstraite. Le choix donnera des modèles plus ou moins lisibles et aisément compréhensibles.
- Enfin, il est nécessaires de définir une **sémantique opérationnelle** i.e. un ensemble de règles et de mécanismes qui décrivent comment un modèle établi avec ce langage doit être interprété pour imaginer le comportement du système modélisé, et permettent donc de vérifier puis de valider ce modèle soit au moyen de mécanismes de simulation (exécution directe du modèle) ou de mécanismes de preuve (*theorem prover* ou *model checker*) de propriétés baptisées propriétés modèle (i.e. des propriétés de construction telles que celles qui sont proposées dans la suite) et de propriétés système (i.e.

des propriétés traduisant les exigences auxquelles le système modélisé doit répondre).

Cet aspect vérification et validation est crucial. En effet, selon [ISO15288, 2015] la vérification est le processus fournissant une preuve objective que le système remplit les exigences qui lui sont attribuées. La validation correspond quant à elle au processus fournissant une preuve objective que le système remplit les besoins spécifiés par les parties prenantes du système (la maitrise d'ouvrage). Ces processus sont cependant attribués à la phase **remontante** du cycle en V, dans le cadre des processus dits d'Intégration, de vérification, de Transition et de Validation. Il s'agit ici, dès la conception de s'intéresser aux processus de vérification et de validation qui supportent la conception et pour lesquels ces définitions s'appliquent aux modèles du système [Chapurlat, 2007].

Dans la phase **descendante** du cycle en V, la vérification consiste donc à s'assurer et à justifier que chaque modèle constitué est « bien construit », i.e. qu'il ne comporte pas d'erreurs et/ou de contradictions, est cohérent avec les autres modèles du même SOI (bien qu'établis éventuellement pour des objectifs différents et avec des DSML différents). De la même manière, la validation dans la phase descendante vise à statuer sur le fait que le modèle est « bon » du point de vue des besoins i.e. de s'assurer et de justifier que le modèle décrit un système qui remplit les exigences. On parle donc ici de qualité du modèle [Chapurlat, 2007].

Cette partie de la V&V est souvent négligée dans la phase de conception et est pourtant primordiale car elle est permet de réduire les risques encourus en cas d'erreurs de modélisation. Erreurs qui peuvent être éliminées pourtant assez tôt dans la conception. Il existe pour cela quatre stratégies de vérification de modèles selon [Chapurlat, 2007] :

- La modélisation guidée. Elle consiste à structurer et contraindre la modélisation de façon à réduire, voire empêcher et donc éradiquer les erreurs de modélisation.
- L'expertise. Elle consiste à faire intervenir un expert statuant sur la cohérence puis sur la qualité du modèle. Fréquemment usitée, elle repose uniquement sur les capacités de jugement de l'expert.
- La preuve formelle. Cette stratégie repose sur la modélisation puis l'analyse du comportement du modèle avec des outils formels, basés sur des formalismes mathématiques.
- La simulation. Cette dernière stratégie consiste à s'assurer de la cohérence et de la qualité du modèle au regard des objectifs de l'acteur ayant créé ce modèle en exécutant la sémantique opérationnelle ou en transformant ce modèle dans un modèle tiers lui-même exécutable.

2.2 Les pratiques de Framatome en IS

Cette partie a pour but de présenter les pratiques de Framatome en IS.

Tout d'abord, Framatome propose une approche centrée sur le PBS (Product Breakdown Structure), i.e. la décomposition hiérarchique produit de la centrale nucléaire, dont nous donnons un exemple illustratif sur la Figure 4.

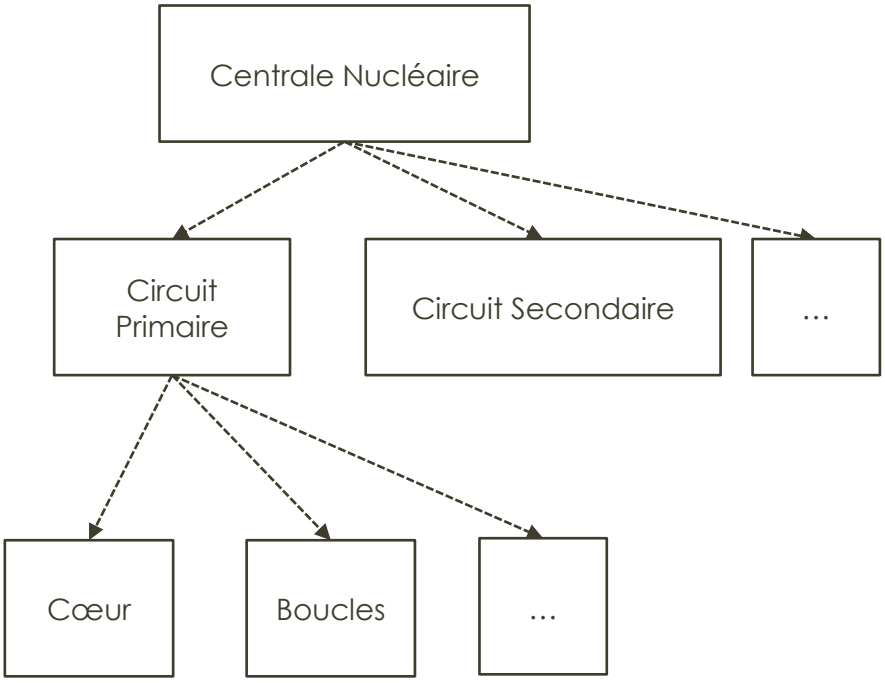


Figure 4 : Exemple de PBS

Ensuite, comme annoncé dans le chapitre 1, Framatome a fait le choix d'introduire les méthodologies IS dans ses activités de conception. Pour cela, l'entreprise a développé ses propres processus d'ingénierie. Ces processus sont cohérents avec les processus de la norme [ISO15288, 2015] et se synthétisent comme montré Figure 5. On définit le cycle RADIV, acronyme défini par :

- Requirement & Capture Analysis. Cette phase rassemble les processus : Analyse de Mission, Définition des Exigences des Parties Prenantes et Définition des Exigences Système de l'[ISO15288, 2015].
- Architecture Design. Equivalent du processus de Conception des Architecture du l'ISO15288.
- Design Implementation. De même, il trouve son équivalent dans le processus d'Implémentation de l'[ISO15288, 2015].
- Integration.
- Verification & Validation. Ces dernières correspondent aux processus d'IVTV (Implémentation, Vérification, Transition et Validation) de l'[ISO15288, 2015].

Le RADIV s'applique donc itérativement à chaque niveau du PBS, i.e. à chaque niveau système. La Figure 5 montre différents liens (éléments jaunes sur la figure) permettant de partager l'information entre les différents processus dans un même niveau système et entre les différents niveaux système. Par exemple, « Requirements From Upper System & Other Stakeholders » représente l'ensemble des exigences qui seront allouées à un système provenant soit du système du dessus, appelé le sur-système ou alors d'autres parties prenantes. Parallèlement, le lien « Requirements on SOI sub-systems » représente les exigences systèmes qui seront allouées aux sous-systèmes du SOI.

Notons les trois liens « SOI requirements Specification », « SOI Design Description » et « V&V plans » représentant respectivement les spécifications du système autant de par ses exigences que par son design ainsi que les plans de V&V ayant pour but la vérification et la validation du SOI.

Enfin, une fois la phase de design (RAD de RADIV) effectuée, la phase IV démarre au cours de laquelle les sous-systèmes sont intégrés, vérifiés et validés pour fournir la solution système attendue et requise par le niveau système supérieur (le sur-système).

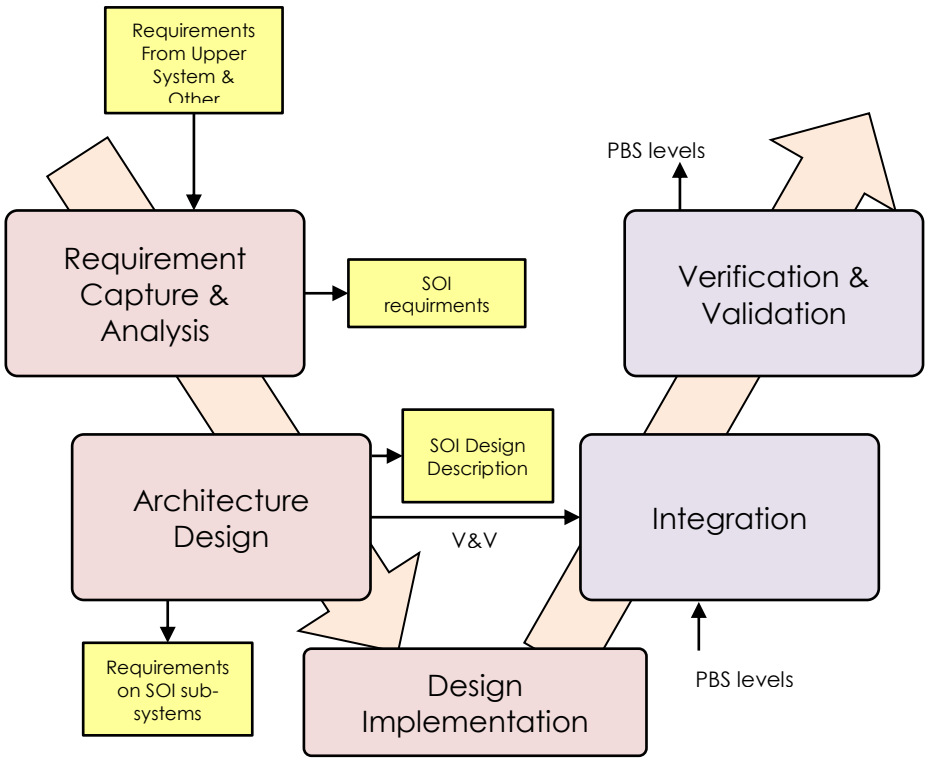


Figure 5 : Cycle RADIV

Ensuite, toujours dans le cadre du RADIV, les exigences sont définies textuellement. Elles possèdent certains attributs comme : un identifiant, un contenu et un contexte. Dans l'entreprise, les bonnes pratiques donnent un certain nombre de mots interdits

(comme dans le cas de la norme [ISO 29148, 2011]) et d'autres dont l'utilisation est fortement encouragée (e.g. utilisation de « shall » au lieu de « must » quand cela est possible).

En ce qui concerne la traçabilité des exigences, Framatome propose une relation inter-exigences permettant de relier une exigence « amont » (*upstream requirement*) à une exigence « aval » (*downstream requirement*).

Enfin, certains outils support au RADIV ont été choisis dans le cadre du déploiement de l'IS chez Framatome. Notamment, l'outil Teamcenter de Siemens. Cet outil permet de gérer l'ensemble des exigences ainsi que les différents éléments de modélisation comme les fonctions ou encore les systèmes définis dans le PBS (lui aussi décrit sur l'outil).

De même, Framatome fait partie du projet Clarity[1], impliquant un grand nombre d'acteurs industriels français autour de Capella (cf. [Roques, 2016]), un logiciel de MBSE développé par Thalès.

2.3 L'existant en IS

Compte tenu du contexte IS de Framatome présenté ci-dessus, différents travaux ou pratiques existantes nous semblent adresser en partie différents verrous que nous avons identifiés dans le chapitre 1.

2.3.1 L'écriture structurée d'exigences

L'écriture des exigences, à la fois dans leur forme et leur sémantique, est l'un des sujets les plus cruciaux lors des activités et des processus d'ingénierie des exigences. Nous avons vu qu'actuellement Framatome propose des règles d'écriture basées sur des mots et expressions interdites ainsi que sur l'utilisation privilégiée de certains termes (e.g. l'utilisation de « shall »). On retrouve des règles similaires dans la norme [ISO 29148, 2011] définissant des critères de langage d'exigence textuelles, conseillant de ne pas utiliser des termes ambigus comme les superlatifs « best » ou bien « most ».

L'utilisation du langage naturel libre pour l'écriture des exigences a l'avantage d'être compréhensible par tous les acteurs de la conception. Cependant, l'écriture est alors assujettie au style et à l'appréciation des auteurs. Les exigences sont en

[1] http://www.clarity-se.org/, dernière visite le 14/11/2017

pratique ambiguës et imprécises dans ce cas. Pour cette raison, un grand nombre de langages de modélisation d'exigences ont vu le jour.

Tout d'abord, certains langages comme RDM (Requirement Definition Model) [Zelm *et al.*, 1995], KAOS [Lamsweerde et al.,1991], UCM (Use Case Map) faisant partie de la norme URN (User Requirements Notation) [ITU-T, 2008], GRL (Goal Oriented Requirement Language) elle aussi faisant partie de URN ou encore URML (Unified Requirements Modeling Languages) [Schneider *et al.*, 2012] définissent des langages de modélisation d'exigences, pour la plupart graphiques et apportant un haut niveau de rigueur pour leur modélisation [Amokrane *et al.,* 2015] et [Amokrane, 2016]. Cependant, ces langages ne sont pas intuitifs et nécessitent une formation préalable des utilisateurs.

D'autres approches plus accessibles utilisant un langage naturel « controlé » (CNL – Controlled Natural Language) ont été introduites de manière à rester facilement exploitables (car compréhensibles de tous), tout en ajoutant un aspect semi-formel pour contraindre le langage naturel à des formulations précises [Williams *et al.*, 2014]. Parmi les CNL existantes on retrouve Attempto Controlled English [Fuchs, 2006], SBVR [SBVR, 2008], Structured English et RuleSpeak [SBVR, 2008].

D'autres travaux se sont penchés sur une écriture des exigences plus structurée que les CNL tout en utilisant un langage naturel. En ce sens, [CESAR, 2012] propose des patterns textuels d'exigences appelés **Boilerplates**. Ces patterns sont des éléments textuels concaténables dont une partie est fixe et l'autre mutable. Autrement dit, des morceaux de texte à trous que l'auteur de l'exigence peut joindre bout à bout et remplir afin de constituer une exigence ayant du sens. Leur but est de :

- Limiter l'ambiguïté grâce à leur structure et l'instanciation unique des « trous »,
- De structurer l'écriture, ce qui a pour effet de la rendre plus concise et précise,
- Tout en restant compréhensible de tous (langage naturel). Un exemple de boilerplate est donné ci-dessous :

The <system> shall <action>

Les éléments mutables, précisés entre les symboles '<'et '>' sont spécifiés par l'auteur. [Farfeleder *et al.*, 2011] en fournit une classification. Le concept de boilerplate a été introduit en premier par [Hull et al., 2002]dont l'idée première était de faciliter la lecture et augmenter la qualité des documents d'ingénierie. Notons que l'on retrouve également dans la norme [ISO 29148, 2011] le principe de boilerplate.

RAT (Requirement Authoring Tool)[2] est un outil permettant d'écrire des exigences grâce au formalisme des boilerplates. Il est doté d'un éditeur dans lequel l'utilisateur peut écrire directement son exigence l'aidant, grâce notamment à des fonctionnalités d'auto complétion et de correction, à écrire et structurer ses exigences. Cet outil permet de *parser* (séparer les éléments sémantiques) les exigences de sorte que les éléments sémantiques contenus dans les exigences soient analysés et comparés à ceux des autres exigences. Il permet ainsi d'analyser la cohérence et l'homogénéité de ces dernières et d'assurer la vérification à travers la modélisation guidée.

Le majeur désavantage de cette approche est qu'elle nécessite un grand nombre de boilerplates pour ne pas limiter l'écriture des exigences.

Enfin, [Micouin, 2008] propose lui aussi une méthode d'écriture structurée d'exigence basée sur le concept de propriétés (Property Based Requirements - PBR). Cette approche diverge cette fois de l'écriture textuelle. Elle propose d'écrire l'exigence sous une forme « mathématisée » dans laquelle les variables de modélisation et les contraintes ou critères de performance sont mis en relation. Par exemple, l'exigence suivante $Req: C \Rightarrow val(VHL.S_{max}) = [165\ km/h, 175\ km/h]$ définit un intervalle pour la vitesse maximum S_{max} du véhicule VHL.

2.3.2 La traçabilité des exigences

La traçabilité des exigences est définie par [EIA632, 1999] comme l'aptitude à identifier les relations entre les différents artéfacts du processus de développement, notamment : la filiation des exigences, les exigences allouées, et l'affectation d'exigences aux caractéristiques de conception. Nous avons vu que Framatome propose une relation inter-exigences permettant de relier une exigence « amont » (*upstream requirement*) à une exigence « aval » (*downstream requirement*). Le langage de modélisation pour les systèmes SysML (cf. [SysML, 2012]), extension d'UML (Unified Modelling Language), propose quant à lui trois types de relations inter-exigences, que sont : le raffinement, la dérivation et la décomposition (ces relations étant utilisées dans la thèse nous les développerons dans la partie 5.3.6), ainsi que des relations reliant les exigences à certains éléments de modélisation, notamment aux modes et scénarios opérationnels (de même ces relations étant utilisées dans la thèse nous les développerons dans la partie 5.3.7). Ces relations ont pour but d'assurer la traçabilité des exigences tout au long de la décomposition système.

[2] https://www.reusecompany.com/requirements-authoring-tool, dernière visite le 14/11/2017

2.3.3 Ingénierie des architecture (IA)

L'IS considère 3 types d'architectures : l'architecture fonctionnelle, l'architecture logique et l'architecture organique (aussi appelée architecture physique). Nous donnons une définition de ces architectures dans la partie 5.5. En ce qui concerne l'ingénierie des architectures fonctionnelles nous citerons tout d'abord l'outil Capella suscitant l'intérêt de nombreux industriels notamment autour du projet Clarity. Cet outil permet de couvrir en grande partie l'IA fonctionnelles grâce en particulier à sa vue System Analysis (Analyse du Système) dans laquelle les fonctions et les flux qu'elles échangent peuvent être décrit.

Un DSML particulièrement intéressant pour la spécification des architectures fonctionnelles et logiques est le DSML eFFBD (enhanced Functional Flows Block Diagram) [Vitech, 2011]. Ce DSML permet de représenter la manière dont les fonctions sont agencées grâce à des constructs de modélisation (choix, séquence, parallélisme ou encore boucle) et donc enchaînées dans le temps proposant une vision dynamique de ces architectures. Il permet de plus de représenter les différents flux échangés entre ces fonctions. En termes d'outillage support, l'outil Core de Vitech implémente ce DSML.

Dans le domaine de l'ingénierie des architectures organiques (physiques), citons tout d'abord le DSML composite qu'est SysML. Ce DSML est en effet composé de plusieurs DSML dont le Block Definition Diagram (BDD) et le Internal Block Diagram (IBD) qui permettent de décrire l'architecture organique, c'est-à-dire l'agencement des éléments d'un système et les liens qui les relient.

Comme pour l'IA fonctionnelle Capella couvre aussi la modélisation des architectures logique et organique. L'outil propose une approche multi-vues avec en particulier le paradigme de composant logique et physique (organique). Il permet alors de définir une architecture logique (composants logiques en interaction) et une architecture physique (composants physique/organiques) en interaction). De plus, Capella permet ensuite d'allouer l'architecture fonctionnelle à des composants logiques et d'allouer des composants logiques à des composants organiques (cf. Figure 31 en 5.5).

2.3.4 Comportement

Un autre aspect de l'IS que nous abordons dans la thèse est celui de la description du comportement des systèmes au travers de la vue comportementale ou opérationnelle selon les auteurs. Il existe un grand nombre de DSML permettant de spécifier un comportement d'un système. SysML propose notamment un diagramme d'activités, un diagramme d'états et un diagramme de séquence. On retrouve en

partie ces diagrammes dans l'outil Capella (diagramme de modes et de séquences).

- Le diagramme de séquence est un DSML utilisé par de nombreux outils supports (Capella et Core cités précédemment notamment). Il a pour vocation à décrire les interactions entre des systèmes, et principalement dans un paradigme boite noire, c'est-à-dire sans considérer le comportement interne du système et donc principalement du point de vue du besoin comportemental du système dans son environnement.
- Le diagramme d'activités permet de représenter le comportement du système par le séquencement des activités à réaliser par le système. Dans SysML, il est complémentaire au diagramme d'état. Le diagramme d'activité ressemble ainsi au diagramme eFFBD par l'enchainement des tâches (fonctions dans le cas eFFBD) à réaliser.
- Enfin, le diagramme de modes dans Capella, est un DSML permettant de définir un état particulier du système dans lequel on lui associe un comportement précis. D'autres outils implémentent ce DSML comme *Stateflow* de *Matlab*.

Enfin, la vue comportementale d'un système doit souvent porter aussi sur les comportements entrées-sorties du système. Dans ce domaine, le DSML SysML propose le diagramme paramétrique permettant de spécifier des comportements entrée-sortie sous la forme de schémas blocs. Simulink de Matab offre de la même manière un langage graphique complet, clair et efficace de spécification du comportement entrée-sortie. On peut aussi citer *Xcos* de *Scilab*[3] alternative open-source de *Simulink*.

Il existe enfin de nombreux outils de modélisation similaires comme Open*Modelica* qui implémente le DSML Modelica et qui est de fait un outil de modélisation multi-métier orienté objet, *Amesim* de *LMSImagine, Catia,* ou encore *Dymola*. Ces outils sont largement utilisés dans différents domaines industriels pour la modélisation multi-physique.

2.3.5 MBSE

En matière de MBSE, nous avons dit précédemment que Framatome a choisi l'outil *Teamcenter* de *Siemens*. Cet outil offre la possibilité de gérer la base de données contenant les différents éléments de modélisation comme les exigences, le PBS ou encore les fonctions. Précédemment, nous avons cité les outils Capella et Core qui permettent eux aussi de manipuler des modèles.

[3] https://www.scilab.org/fr, dernière visite le 14/11/2017

2.3.6 La vérification et la validation des modèles

Plusieurs travaux adressent la problématique de la vérification, quelquefois d'une partie de la validation, des modèles au sens abordé dans la partie 2.1, i.e. dans la phase descendante du cycle en V. Une grande partie des travaux que nous avons présentés précédemment participe à cette vérification. De manière générale, c'est l'intention de tout outil ou DSML de MBSE : favoriser le partage d'informations entre les différents acteurs de la modélisation afin de réduire les erreurs de modélisation.

Dans 2.1 nous avons présenté quatre stratégies de vérification : par la modélisation guidée, par simulation, par preuve formelle et par expertise. Dans le cas de la modélisation guidée appliquée aux exigences, on peut citer les boilerplates de [CESAR, 2012], aussi présentés dans [ISO 29148, 2011] ou encore l'outil RAT, et [Micouin, 2008]. Ils guident et favorisent l'écriture structurée des exigences en guidant l'auteur dans sa rédaction et en limitant donc les erreurs classiques de modélisation d'exigences en maximisant la capacité de l'auteur à écrire des exigences SMART (Simple, Measurable, Achievable, Relevant, Traceable).

En ce qui concerne la vérification par simulation, on peut noter que *Modelica*, *Simulink*, *Amesim* et *Dymola* offrent des solides capacités de simulation. A ce titre [Renier *et al.*, 2011] formalise le passage de SysML à Modelica dans le but de formaliser des modèles de conception dans une phase de design préliminaire. Cet article montre que l'approche MBSE basée de *SysML* facilite la compréhension et la lisibilité du modèle tandis que *Modelica* apporte les capacités de simulation et de vérification par simulation.

2.4 Positionnement des travaux de la partie IS

Dans les parties précédentes, nous avons vu : les verrous que nous cherchons à adresser dans la thèse, les pratiques de Framatome et, enfin, une partie de l'existant que nous considérons comme en lien avec les deux premiers points. Dans la suite, nous justifierons les différents choix qui ont été faits pour répondre aux problématiques exposées.

La Figure 6 présente un tableau récapitulatif comparant les possibilités offertes des différents outils et langages que nous avons précédemment cités. Tout d'abord, prenons le cas de SysML. Ce DSML est très utilisé dans l'industrie car il offre une approche multi-vue intéressante. Cependant, ce langage n'offre pas la possibilité de structurer le langage pour l'écriture des exigences. De plus, SysML est assez limité en ce qui concerne la définition des architectures fonctionnelles et logiques. Enfin, SysML ne possède pas de sémantique opérationnelle reconnue permettant son analyse par preuve ou par simulation. Le seul type de vérification que l'on peut

entreprendre avec ce modèle est la vérification par l'expertise, ce qui est très limité (cf. 2.1).

L'outil Core offre un environnement multi-vues particulièrement intéressant pour une approche MBSE. Il implémente le langage eFFBD (entre autres) et donc bénéficie des avantages de ce langage. Cependant, l'outil n'implémente pas une méthode structurée. Il offre une grande liberté de modélisation ce qui peut perdre l'utilisateur dans son utilisation. Enfin, il ne possède pas de langage structuré pour la rédaction d'exigences.

Capella est un outil très complet qui permet de couvrir un grand nombre phases du cycle de vie. Cependant, cet outil ne permet pas en l'état d'adresser les problématiques qui nous concernent en termes d'ingénierie des exigences. Bien qu'il soit possible de créer des exigences, leur contenu reste libre et non structuré. De plus, le typage des exigences ne permet pas d'assurer la traçabilité inter-exigences (cf. 5.3.6). Bien qu'il possède une démarche opératoire, l'outil offre un grand nombre de langages et de nombreuses vues. Tout comme Core, ceci peut avoir pour effet de perdre l'utilisateur.

Les outils de simulation comme Simulink ou Modelica offrent quant à eux une plateforme performante pour la modélisation et la simulation. Cependant, l'ingénierie des exigences n'est pas vraiment centrale dans ces environnements. Ils ne permettent donc pas de structurer le langage et de guider l'auteur dans sa conception.

Existant		Ingénierie des Exigences			Ingénierie des architectures			Définition du Contexte	Définition du comportement	Vérification des exigences
		Gestion des exigences	Ecriture Structurée des exigences	Traçabilité des exigences	Fonctionnelle	Logique	Physique			
DSML	SysML	-	-	++	--	---	++	+	+++	-
	eFFBD	---	---	---	+++	---	---	--	++	-
	Diagramme de contexte	---	---	---	-	---	+	+++	---	---
	Diagramme de Modes et Transitions	---	---	---	---	---	---	---	+++	---
	Diagramme de séquences	---	---	---	---	---	---	---	+++	-
	Boilerplates	---	+++	---	---	---	---	+	+	++
	Property Based Requirement	---	++	-	---	---	---	---	+	++
	Block Definition Diagram	---	---	---	++	+	+	---	++	---
	Internal Block Diagram	---	---	---	+	+	++	-	++	---
Outils	Core	++	+	++	+	-	++	+	++	-
	Capella	--	---	-	+++	+++	+++	++	+	--
	RAT	+	+++	-	---	---	---	-	+	+++
	Simulink	---	---	---	+	+	+	--	++	--
	Modelica	--	--	--	++	++	++	-	+++	-
	Teamcenter	+++	-	++	--	--	--	---	--	--

Figure 6: Tableau récapitulatif de l'état de l'art IS

Dans la thèse, nous nous concentrerons alors sur l'ensemble de langages existants, jugés pertinents compte tenu du contexte de travail présenté précédemment (cf. Partie I et 2.2). Ils sont énoncés ci-après :

- Les boilerplates (CESAR) : Les boilerplates offrent une bonne capacité de structuration du langage tout en utilisant le langage naturel, et donc compréhensible par tous.
- Le diagramme de séquences (Capella) : le DSML proposé par Capella est efficace pour définir les interactions du SOI avec les éléments de son environnement.
- Le Lookup Tables/Signal Builders (Matlab Simulink) : ces DSML permettent de définir des dépendances entre deux ou plusieurs signaux. Nous nous en servirons pour définir les dépendances entre les entrées et sorties des éléments de modélisation de nos DMSL (fonction et système en particulier).
- Le diagramme de Modes et Transition (Capella / Stateflow / SysML) : Le diagramme des Modes est un incontournable des DMSL utilisés dans la vue comportementale.
- Le DSML fonctionnel (eFFBD de Vitech) : ce DSML permet de définir l'architecture fonctionnelle d'un système en utilisant pleinement le paradigme fonctionnel qui fait défaut à d'autres DSML comme SysML. Il permet de plus de décrire une vision dynamique de l'Architecture fonctionnelle et les échanges de flux nécessaires.
- Le DSML de spécification fonctionnelle de Framatome. Ce DSML spécifie le traitement entrée-sortie des fonctions.
- Le diagramme de Modes et Transition (Capella / Stateflow / SysML) : Le diagramme des Modes est un incontournable des DMSL utilisés dans la vue comportementale.

Au regard de ces éléments, nous proposerons une méthode complète d'IS (voir 0 pour la définition de méthode), utilisant et fédérant les langages de modélisation présentés ci-dessus et qui consiste à :

- Assister l'ingénierie des exigences et tracer le passage des exigences aux architectures. Nous proposerons un ensemble de relations inter-exigences (cf. 5.3.6) et exigences/éléments de modélisation (cf. 5.3.7).
- Guider l'utilisateur dans sa conception et ce, autant grâce aux langages eux-mêmes (e.g. les boilerplates pour structurer le langage de rédaction d'exigences) que par la possibilité qui est offerte à cet utilisateur de vérifier la cohérence de sa modélisation.
- Vérifier les modèles et en faciliter à terme une partie de la validation. Premièrement, nous proposerons des propriétés pour chacun des modèles (cf. 5.10). Deuxièmement, une partie des modèles sera simulable afin de tester (et donc vérifier de manière comportementale ou supporter les activités de validation).

NB : bien que la méthode que nous proposons ait pour vocation à s'appliquer sur des systèmes quels qu'ils soient, notre étude s'intéressera aux systèmes de contrôle commande de Framatome. Nous nous focalisons donc dans les parties suivantes sur les éléments inhérents aux systèmes de contrôle commande et chercherons à démontrer comment notre étude peut s'adapter à des systèmes de natures et de niveaux de complexité différents.

3 Etat de l'art Automatique

Sommaire

3.1 Modélisation et contrôle des REP 49
3.1.1 Modélisation des REP 49
3.1.2 Contrôle des REP 50
3.2 Les modes de pilotage de Framatome 52
3.2.1 Objectifs de commande et actionneurs considérés 52
3.2.2 Le Mode A 53
3.2.3 Le mode G 54
3.2.4 Le Mode T 54
3.3 Les stratégies de commande avancées retenues dans la thèse 59
3.3.1 La commande multi-objectif 62
3.3.2 La commande à gains-séquencés 65
3.3.3 La commande prédictive 67

3.1 Modélisation et contrôle des REP

Dans cette partie nous allons présenter l'existant en matière de contrôle des réacteurs à eau sous pression (REP). Nous verrons tout d'abord les différents modèles qui sont utilisés dans la littérature et ensuite les différentes techniques employées pour manœuvrer les tranches nucléaires.

3.1.1 Modélisation des REP

La modélisation des réacteurs nucléaires fait l'objet de nombreux travaux. [Li, 2016] propose un tour d'horizon des différentes approches de modélisation existantes. Les principales approches citées par l'article sont :

- **La modélisation point.** C'est l'approche la plus classique. Le cœur est considéré comme un point. Les effets dimensionnels ne sont donc pas représentés. Cette approche est utilisée par [Arab-Alibeik, 2003], [Ablay, 2013], [Dong, 2011], [Edwards *et al.*, 1990] et [Na, Shin et Kim, 2003] pour le contrôle des réacteurs nucléaires.
- **La modélisation axiale** (une dimension). Le modèle du cœur résulte de l'agrégation de plusieurs modèles points, associés à des points répartis axialement. Ces points échangent des neutrons par diffusion. Cette approche permet de représenter les effets unidimensionnels comme la distribution axiale

de puissance (AO). Pour le contrôle des réacteurs on trouve souvent le cas « bipoints », i.e. que le cœur est divisé en seulement deux points. Son utilisation est évoquée dans [Zhang *et al.,* 2015], [Li, 2014], [Ukai et Iwazumi, 1994], [Wang *et al.*, 2014] et [Bauer et Poncelet, 1974].

- **La modélisation 3D.** Dans ce cas le cœur est divisé en de multiples mailles dans les trois dimensions. Cette approche permet de représenter les effets radiaux (épuisement dans l'espace du combustible, AO radial et axial, etc.). Ce type de modèle est peu utilisé pour la synthèse de contrôleurs basés sur des modèles. En revanche, pour les stratégies type PID il est souvent utilisé pour régler les gains du régulateur et vérifier les objectifs de commande en simulation car il fournit une très bonne représentativité des phénomènes physiques.

Enfin, [Bendotti *et al.*, 1994] aborde la modélisation du cœur par une approche pragmatique. Le cœur est considéré comme un modèle LTI autour d'un point de fonctionnement. Le système LTI est déterminé par identification entrée-sortie des différents transferts. Le modèle résultant est d'ordre 6. Les entrées considérées sont deux groupes de grappes de contrôle (cf. groupes considérés dans le mode G en partie 3.2.3) et les sorties sont la température moyenne et l'AO.

3.1.2 Contrôle des REP

Un grand nombre de techniques de contrôle ont été appliquées au cas des réacteurs nucléaires [Li, 2016] passe en revue les techniques de contrôle ayant été appliquées aux différentes technologies de réacteurs. On distingue tout d'abord deux types de contrôle des réacteurs :

- Le contrôle du niveau de puissance

Le contrôle du niveau de puissance consiste à contrôler la puissance du cœur par insertion ou extraction des grappes de contrôle dans le cœur [Grossetête, 2007]. Différentes techniques ont été employées dans ce cas. [Edward *et al.*, 1992] propose d'appliquer la théorie de la commande optimale en employant un régulateur par retour d'état observateur sur la base d'un modèle point. Sur la base d'un modèle point similaire [Arab-Alibeik, 2003] propose la synthèse d'un régulateur LQG/LTR.

D'autres méthodes de contrôle ont été employées dans ce cas. Notamment, [Dong, 2011] applique la théorie de la commande non-linéaire, [Na, Shin et Kim, 2003] celle de la commande optimale ou encore [Emara *et al.,* 2013] celle de la commande H_∞.

- La variation de charge

La variation de charge consiste à faire évoluer la puissance fournie par la turbine de manière à suivre un programme de puissance prédéfini. Les variations de charge impactent les conditions d'opération du cœur et fait évoluer deux grandeurs caractéristiques : la **température moyenne** et la **distribution axiale de puissance** (AO) dans le cœur.

Les modes de pilotage classiques

Afin de contrôler le cœur lors des variations de charge quatre modes de pilotage principaux ont été développés successivement depuis les années 70 par Westinghouse Electric Corporation USA, Framatome Corporation France devenue ensuite AREVA NP puis de nouveau Framatome en 2018 : le mode A, le mode G, le mode T et le MSHIM ainsi que le Konvoi.

Les modes A et G permettent le contrôle automatique de la température moyenne mais conservent un contrôle manuel de l'AO. Notons de plus que le mode G permet de remonter à puissance nominale à tout moment par les grappes de contrôle [Mourlevat, 2007] contrairement au mode A. Les autres modes : T, MSHIM et Konvoi permettent tous de contrôler la température moyenne et l'AO en automatique. Tous ces modes utilisent principalement la technologie PID.

Les modes A, G et T seront détaillés dans la partie 3.2 et nous ne détaillerons pas les autres modes cités.

La commande avancée des réacteurs

La commande avancée des réacteurs a été approchée par un grand nombre de techniques : la commande optimale, la commande robuste, la commande adaptative ou encore par retour d'état.

Afin de contrôler l'axial offset, une grande partie des travaux identifiés utilise **un modèle axial bipoint** de cœur. Notamment, [Li, 2014] conçoit un régulateur LQG/LTR dans le but de contrôler la puissance du cœur et l'AO. Afin de contrôler le cœur dans toutes les configurations de groupes de grappes, deux contrôleurs sont proposés. L'un contrôle le cœur pour les grappes dans le haut du cœur, l'autre lorsqu'un groupe de grappes est inséré. Les oscillations xénon sont rejetées et la puissance du cœur et l'AO sont bien maintenus dans des domaines acceptables. [Winokur et Tepper, 1984] s'intéresse à la commande optimale pour étendre les capacités de variations de charge des réacteurs à eau pressurisée en utilisant lui

aussi une modélisation axiale bipoints. Les travaux de [Wang *et al.*, 2014] et [Zhang *et al.,* 2015] s'intéressent quant à eux à la faisabilité de variations de charge sans ajustement de la concentration en bore pour le réacteur CPR1000. Certains groupes de grappes sont pré-insérés dans le cœur (même à 100%PN) permettant de contrôler le cœur en dépit des variations xénon.

[Bendotti *et al.*, 1994] propose une approche utilisant un modèle simplifié du cœur (cf. 3.1.1) pour concevoir un régulateur H_∞. Cet article montre la faisabilité d'un tel régulateur à un niveau de puissance donné. Cependant, le régulateur montre des limites quand il est utilisé à différents niveaux de puissances. Un régulateur à gains séquencés sur la puissance est alors conseillé. C'est ce que propose [Bodenheimer et al., 1995] en synthétisant un régulateur LPV basé une approche LMI [Apkarian et Gahinet, 1995]. Le REP est alors contrôlé conformément aux attentes sur un domaine étendu paramétrisé ici par la puissance de 50%PN à 99%PN.

3.2 Les modes de pilotage de Framatome

Les modes de pilotage sont des algorithmes de contrôle commande permettant de contrôler le cœur. Il existe un grand nombre de modes de pilotage de centrales nucléaires. Nous aborderons ici uniquement les modes utilisés par Framatome, c'est-à-dire les modes A, G et T[4].

3.2.1 Objectifs de commande et actionneurs considérés

Dans les REP, les variations de puissance entrainent la variation de certaines grandeurs qui doivent être contrôlées. Les deux principales grandeurs à contrôler sont :

- **La température moyenne**. Cette grandeur est contrôlée autour d'une température moyenne de référence afin de garantir les conditions opérationnelles du réacteur.
- **Le déséquilibre axial de puissance** (Axial Offset en anglais, nous la noterons AO dans ce qui suit). Cette grandeur est contrôlée afin de maitriser les oscillations de puissances axiales produites par les variations spatiales de xénon [Mourlevat, 2007].

Dans les modes de pilotage les plus avancés, le contrôle du cœur peut aussi considérer une autre grandeur à contrôler :

[4] Il existe aussi le mode X dont nous ne parlerons pas ici. Nous invitons le lecteur à se référer à [Mourlevat, 2007] pour les détails concernant ce mode de pilotage. De même il existe un autre mode développé par Framatome, appelé le *Konvoï* que nous n'aborderons pas ici.

- **Le Pmax**. Cette grandeur représente la puissance pouvant être atteinte par extraction des groupes de contrôle [Grossetête, 2007].

Le contrôle de ces grandeurs se fait par l'intermédiaire de plusieurs actionneurs :

- **Les grappes de contrôle**. Ces organes s'insèrent dans le cœur faisant varier la réactivité du cœur (cf. 11.2.1). Elles sont divisées en plusieurs groupes selon les différents réacteurs et s'insèrent en séquences.
- **Le bore soluble**. Le bore est injecté dans le cœur sous forme soluble afin de faire varier sa concentration (NB : on injecte du bore pour faire augmenter sa concentration et on injecte de l'eau pour la réduire).

3.2.2 Le Mode A

Le mode A est le mode de pilotage le plus simple des quatre cités plus haut. Il est principalement utilisé sur les centrales 900 et N4 en France. Le but de ce mode est de contrôler la température moyenne du cœur ainsi que le déséquilibre axial de puissance.

Dans ce mode, sont considérés quatre groupes de grappes notés A, B, C et D constitués de 8 grappes chacun. Ces groupes s'insèrent progressivement avec un recouvrement fixe [Mourlevat, 2007].

La température moyenne du cœur est contrôlée de manière automatique directement au moyen des groupes de grappes . Elle doit être maintenue à l'intérieur d'une bande de $\pm$ 0.83 °C autour de sa température de référence. Cette dernière est fonction du programme de température et de la puissance première roue de la turbine (cf. Figure 7). La technologie utilisée pour cette régulation est un PID dont le gain varie en fonction de la puissance du cœur pour compenser les variations de réactivité en basse puissance.

L'AO est quant à lui contrôlé manuellement par l'opérateur au moyen de la dilution et de la borication (injection d'eau ou de bore soluble dans le circuit primaire matérialisé par u_{Bore} sur le schéma Figure 7). L'injection de bore ou d'eau dans le circuit primaire modifie la température moyenne. Le contrôle automatique de la température induira un déplacement des groupes de grappes qui lui-même modifie l'AO. Cela demande donc une bonne connaissance des effets physiques du réacteur par l'opérateur.

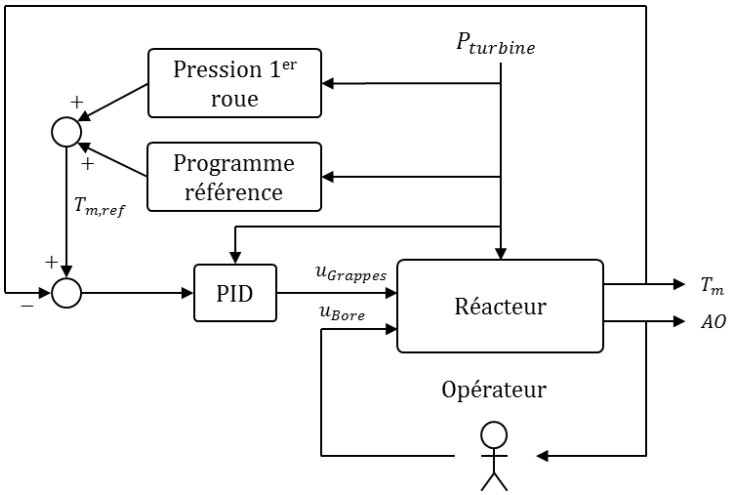


Figure 7: Principe de la régulation du mode A

L'avantage de ce mode de pilotage est qu'il est simple et éprouvé. Cependant, ne peuvent être pratiquées que des variations de charge lentes et ce mode ne permet pas de contrôler le Pmax (retour à puissance désirée instantané).

3.2.3 Le mode G

Le mode G a été conçu afin de pouvoir remonter à puissance nominale « instantanément », i.e. par les grappes de contrôle. Autrement dit, un cas particulier du contrôle de Pmax où la puissance est la puissance nominale.

Pour ce faire, un groupe de grappes est inséré en permanence dans le cœur de manière à compenser le défaut de puissance. Quatre groupes de grappes sont considérés : G1, G2 , N1 et N2. Leurs positions sont asservies à une position de référence donnée directement par la puissance électrique. Cet asservissement est en boucle ouverte et la position est étalonnée périodiquement pour prendre en compte l'épuisement du combustible.

Pour contrôler la température un asservissement en boucle fermée similaire à celui du mode A est réalisé, utilisant un autre groupe appelé groupe R. De la même manière que sur le mode A, l'AO est contrôlé par variation de concentration de bore manuelle (la variation e concentration en bore fait varier la température contrôlée par le groupe R qui, enfin, fait varier l'AO).

3.2.4 Le Mode T

Le Mode T est le mode de pilotage le plus avancé actuellement. Il a été développé pour les besoins des centrales EPR [Grossetête, 2007], [Grossetête, 2008] et [Grossetête, 2014] de Framatome.

Généralités

Comparativement aux autres modes de pilotages présentés précédemment, le mode T a pour objectif de contrôler automatiquement (i.e. sans action nécessaire de l'opérateur) tous les paramètres de contrôles, à savoir :

- La température moyenne.
- L'AO.
- Le Pmax[5]. Concrètement, ce critère de contrôle est traduit en une position de barre Pbank de référence. Le contrôle de Pmax consiste donc à contrôler la position de ce groupe autour de cette référence.

Les moyens de contrôle considérés sont :

- Les grappes de contrôle
- La dilution et la borication (à noter qu'elles s'opèrent de manière automatique).

Configuration des groupes de grappes

Un cœur d'EPR contient 36 grappes de contrôle (PA à PI) regroupés en 5 groupes P1 à P5. L'affectation des groupes PA à PI aux groupes P1 à P5 est modifiée régulièrement au cours du cycle afin d'homogénéiser l'épuisement dans le cœur [Grossetête, 2008].

Les groupes P1 à P5 sont ensuite regroupés en deux groupes à composition variable :

- Pbank. Attribué au contrôle de la température moyenne.
- Hbank. Attribué au contrôle de l'AO.

Comme montrés sur la Figure 8, les groupes P1 à P5 composent les groupes Pbank et Hbank de la manière suivante :

- A 100%PN, Pbank est composé de P1 et Hbank de P2 à P5. Pbank est légèrement plus inséré que Hbank.
- Lorsque la puissance diminue jusqu'à 85%PN, Pbank s'insère de manière à contrôler la température moyenne sur sa référence. La baisse de puissance engendre des variations d'AO contrôlées par Hbank.

[5] Notons que le Pmax dans ce cas caractérise bien la puissance pouvant être atteinte par extraction des groupes de grappes, et non pas uniquement la puissance nominale comme c'est le cas pour le mode G.

- La puissance continue de diminuer. Lorsque La distance entre P1 et P2 devient égale à l'écartement maximal entre deux groupes de grappes (une demi hauteur de cœur), P2 se désolidarise du groupe Hbank et passe dans le groupe Pbank. Il s'insère alors avec P1.
- Lorsque la puissance continue à diminuer, Pbank continue à s'insérer. Quand la distance maximale entre P2 et P3 est atteinte, P3 passe dans Pbank et s'insère, et ainsi de suite avec P4.

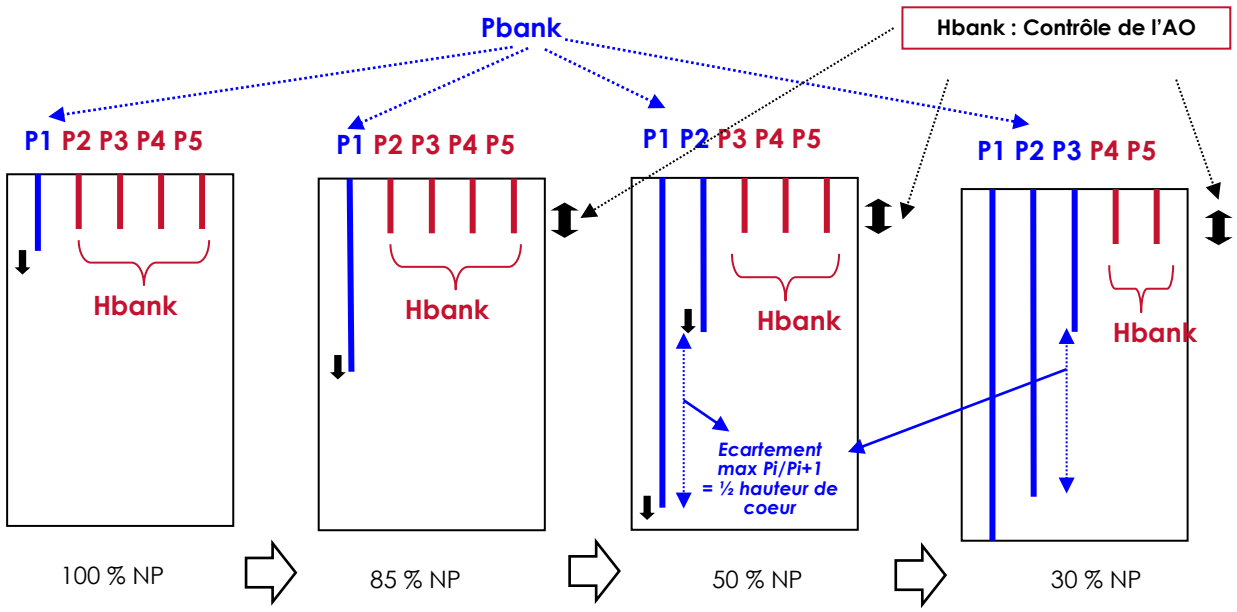


Figure 8 : Mouvement typique des groupes lors d'une variation de puissance. Figure tirée de **[Grossetête, 2007]**

Le Pmax

De façon similaire au mode G, le Pmax est traduit en position de référence du groupe Pbank de compenser le défaut de puissance. Le contrôle du Pmax revient donc à contrôler la position du groupe Pbank selon un profil d'insertion déterminé en fonction de la puissance du cœur et de la puissance à laquelle l'opérateur veut pouvoir retourner. Par exemple un Pmax de 100%PN signifie que la position du groupe Pbank permet de retourner à 100%PN uniquement par son extraction.

Le Pmax est rendu variable de manière à économiser des effluents (bore) lors du fonctionnement à basse puissance après une variation de charge [Grossetête, 2007].

Contrôle de la température moyenne

Comme dit précédemment, c'est le groupe Pbank qui est utilisé en boucle fermée pour le contrôle de la température moyenne de façon similaire aux modes A et G.

La température est contrôlée selon un programme de température présenté sur la Figure 9.

Le température est contrôlée dans une bande morte de ±0.5°C autour de cette référence.

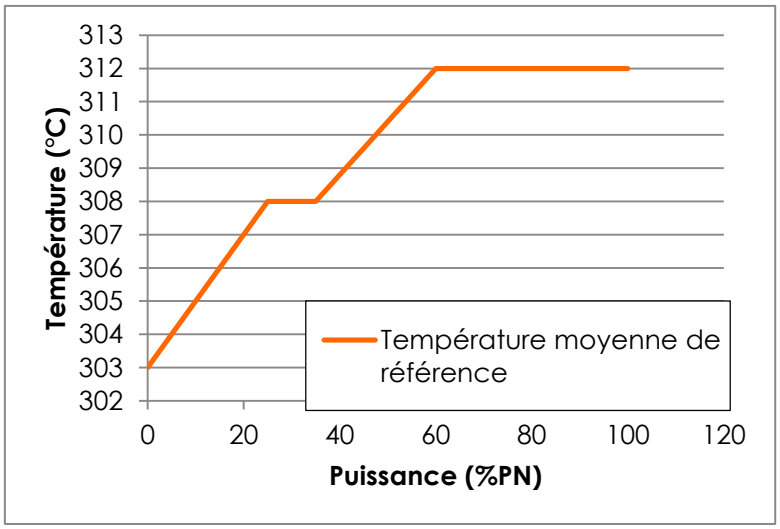


Figure 9: Programme de température moyenne

Contrôle de l'AO

Le principe du contrôle de l'AO est présenté Figure 10. L'idée principalement est de contrôle l'AO par des variations de position de Hbank compensées par des variations de Pbank via la régulation de tempérante (discutée ci-dessus). On appelle ce principe les **mouvements compensés**.

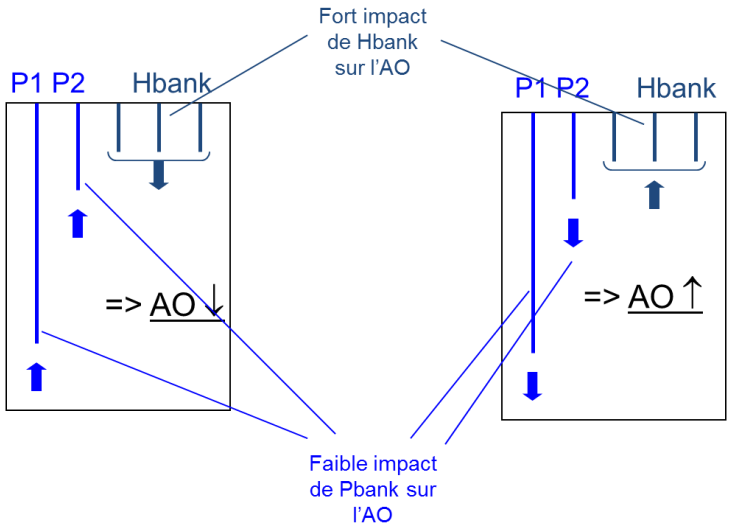


Figure 10: Principe du contrôle de l'AO. Figure tirée de **[Grossetête, 2007]**

Le Contrôle du Pmax

Le principe du contrôle du Pmax est présenté Figure 11. L'idée est d'effectuer des borications (augmentation de la concentration en bore) ou dilutions (diminution de la concentration en bore) de manière à maintenir la position de Pbank dans une bande morte dépendant de la consigne de Pmax et de la puissance thermique du cœur. Les borications (resp. dilutions) provoquent des variations de température moyenne régulée par extraction (reps. Insertion) du groupe Pbank (cf. contrôle de la température moyenne).

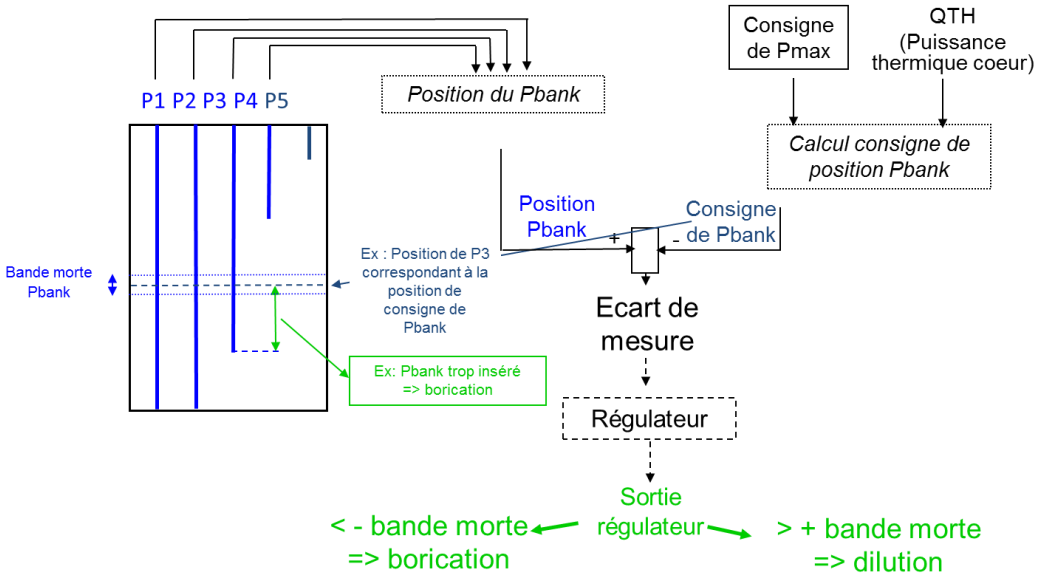


Figure 11: Principe du contrôle du Pmax. Figure tirée de [Grossetête, 2007]

Particularité du pilotage quand les groupes sont dans le haut du cœur

Lorsque les groupes P1 à P5 sont dans le haut du cœur (cf. Figure 8), les variations de position de Pbank (composée uniquement de P1) ont un impact conséquent sur l'AO (contrairement au cas où les groupes sont dans la partie basse du cœur cf. Figure 10). La stratégie adoptée est similaire à celle des modes A et G.

L'idée est d'inverser l'attribution des régulations d'AO et de Pmax précédemment présentées (la boucle de régulation de température reste inchangée) aux actionneurs de la manière suivante :

- Le contrôle du Pmax se fait par mouvements compensés de Hbank induisant des variations de températures régulées par des mouvements de Pbank.
- Le contrôle de l'AO se fait par borication ou dilution, induisant a fortiori des variations de température régulées par la position de Pbank induisant une modification d'AO appropriée.

Pour de plus amples explications sur le fonctionnement du mode T nous invitons le lecteur à se référer aux articles suivants : [Grossetête, 2007], [Grossetête, 2008] et [Grossetête, 2014].

Position cumulée du groupe Pbank

On définit la position cumulée du groupe Pbank notée P_{bank} par la formule suivante :

$$P_{bank} = \min(P_4, 214) + \min(P_3, 214) + \min(P_2, 214) + P_1$$

Où P_1, P_2 , P_3 et P_4 désignent respectivement les positions des groupes P1 à P4. La valeur 214 est choisie pour une position mi-cœur des groupes de grappes.

3.3 Les stratégies de commande avancées retenues dans la thèse

Dans cette partie nous introduisons les méthodologies de commande avancée retenues dans la thèse.

Il est nécessaire dans un premier temps d'introduire le paradigme central en théorie de la commande robuste. Ce paradigme repose sur une structure de commande par rétroaction sur des modèles dits linéaires à temps invariant (LTI) des systèmes dynamiques. Le principe de la commande en boucle fermée (en anglais feedback control) est au cœur des théories de la commande avancée et définit une structure de commande très générale permettant une formulation unifiée des différents problèmes d'analyse et de synthèse des systèmes bouclés proposée par [Doyle, 1983] (cf. Figure 12).

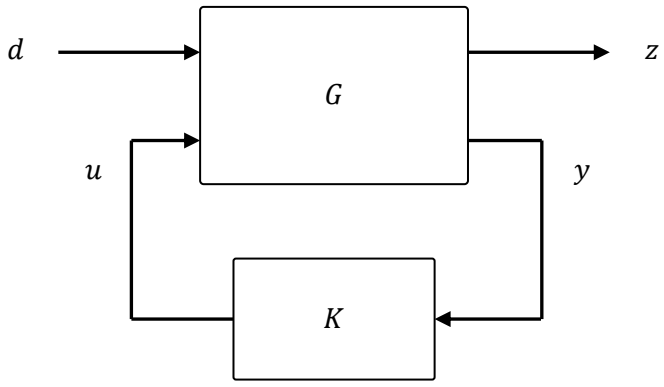


Figure 12: Forme standard de commande

Où G est le modèle LTI et K le correcteur. G modélise les interactions dynamiques entre les deux ensembles de signaux d'entrée $d \in \mathbb{R}^{n_d}$ (vecteur des signaux

exogènes représentant les consignes ou les perturbations), $u \in \mathbb{R}^{n_u}$ (le vecteur des commandes délivrées par le correcteur K) et les deux ensembles de signaux de sortie $z \in \mathbb{R}^{n_z}$ (le vecteur des signaux de sortie contrôlées), $y \in \mathbb{R}^{n_y}$ (le vecteur des sorties accessibles à la mesure et donc disponibles pour élaborer la commande). Notons à ce stade que le modèle LTI G peut être incertain avec notamment une dépendance à l'incertitude de différentes natures. L'objectif principal de la commande est d'assurer la stabilité de la boucle fermée ainsi qu'un certain degré de performance pour l'ensemble des réalisations possibles du modèle incertain. Il s'agit, ainsi, de rendre la structure de commande peu sensible, voire robuste aux variations paramétriques et aux perturbations affectant le modèle du système à commander. Par ailleurs, la notion de performance pour un système dynamique recouvre de très nombreuses caractérisations dans la littérature. Ceci s'explique essentiellement par le développement historique des méthodes modernes de commande (pour un aperçu de cet historique et de sa mise en perspective nous invitions l'auteur à se référer à [Astrom et Kumar, 2014]).

La séparation entre domaines fréquentiel et temporel a conduit à des développements séparés et complémentaires de différents formalismes. Les interrogations concernant les propriétés de robustesse du régulateur LQG (paradigme moderne de la commande optimale en boucle fermée cf. [Tahk et Speyer, 1987]) ont poussé au développement des bases du cadre H_∞. [Hardy, 1915] a, ainsi, introduit la norme H_∞ qui sert d'outil fondamental pour la spécification du niveau d'incertitude admissible et conjointement pour la spécification du niveau de performance entrée-sortie (cf. [Doyle *et al.*, 1989] et [Zames, 1981]).

Le problème H_∞

Etant donné un modèle G, le problème H_∞ consiste à déterminer un correcteur K stabilisant de manière interne le système bouclé défini par la forme standard décrite en Figure 12 et minimisant la norme H_∞ du transfert en boucle fermée entre z et d. Si Ω désigne l'ensemble des correcteurs stabilisant de manière interne G et si $T_{z\to d}$ est la fonction de transfert en boucle fermée entre z et d alors le problème d'optimisation formalisé est le suivant :

$$\min_{K\in\Omega} \|T_{z\to d}(s)\|_\infty$$

- **Définition :** soit $\boldsymbol{X}$ une fonction de transfert stable et propre, on définit sa norme $\boldsymbol{H}_\infty$ par :

$$\|X\|_\infty = \sup_\omega \bar{\sigma}(X(j\omega))$$

La norme H_∞ des systèmes a été au centre de nombreuses études au cours des années 90 (voir [Zhou et al., 1995] et les références incluses). Elle a en effet permis de donner un formalisme unifié pour l'analyse et la synthèse fréquentielle [Duc et Font, 1999]. Elle permet de prendre en compte des objectifs de suivi de référence, de stabilité, de roll-off, de marges de stabilité et de modelage (fréquentiel) des différents transferts. Elle peut être vue comme la norme induite par la norme euclidienne des signaux comme le montre la relation suivante :

$$\|T_{z\to d}(s)\|_\infty = \sup_{\substack{d\in L_2 \\ d\neq 0}} \left(\frac{\|z\|_2}{\|d\|_2}\right)$$

Il existe une caractérisation par inégalité matricielle de la borne supérieure de cette norme (cf. [Boyd, 1994]).

Le problème H_2

De façon similaire au cas H_∞, le problème de commande H_2 revient à résoudre le problème d'optimisation suivant :

$$\min_{K\in\Omega} \|T_{z\to d}(s)\|_2$$

Elle permet d'intégrer des considérations énergétiques sur l'ensemble de la bande de fréquence. La norme H_2 est par exemple un bon outil pour évaluer l'aptitude du système à filtrer, les bruits engendrés par les capteurs, sur les actionneurs ou bien encore issus de perturbations.

- **Définition :** soit $\boldsymbol{X}$ une fonction de transfert strictement propre et stable, on définit sa norme $\boldsymbol{H_2}$ par:

$$\|X(s)\|_2 = \left(\frac{1}{2\pi}\int_{-\infty}^{\infty} Trace\left(X^*(jw)X(jw)\right)dw\right)^{1/2}$$

où X^* désigne la transposée conjuguée de X.

Le calcul algébrique de la norme H_2 utilise les grammiens d'observabilité et de commandabilité du système [Maciejowski, 1989]. De plus, il existe une caractérisation par égalité (Lyapunov) ou inégalités matricielles de la borne supérieure de cette norme.

Au vu des éléments précédents (cf. parties 3.1, 3.2 et 3.3), nous avons constaté qu'il existe un grand nombre de méthodologies de commande candidates pour

l'amélioration des performances des REP. Trois d'entre elles ont particulièrement retenu notre attention :

- La commande multi-objectif et en particulier les méthodes relevant fondamentalement de l'optimisation non-convexe et non-lisse, qui montrent un intérêt croissant. Ces méthodes permettent de traiter des problèmes de commande à partir d'une description fine des attendus (performance et robustesse) en termes de critères et contraintes quantifiées, et d'imposer la structure et l'ordre du régulateur (e.g. via un retour statique de sorties augmentées) [Sadabadi et Peaucelle, 2016].
- La théorie des régulateurs à gains-séquencés permet d'élargir le champ d'application des régulateurs robustes à temps invariant, trop limitants pour conduire à des performances satisfaisantes sur l'ensemble du domaine de fonctionnement du cœur (cf. e.g. [Bendotti *et al.*, 1994]).
- La commande MPC permet de prendre en compte explicitement des contraintes temporelles sur les entrées ou les états. Elle permet en outre d'appréhender efficacement les non-linéarités du système (dans sa version non-linéaire dite NMPC), ainsi que les retards ou signaux (perturbations ou consignes) connus à l'avance [Eliasi *et al.*, 2012] et [Na, Shin et Kim, 2003].

Nous allons dans ce qui suit faire un bref tour d'horizon de ces méthodes.

3.3.1 La commande multi-objectif

La commande multi-objectif est souvent définie, dans la littérature, comme étant un problème de commande mixant plusieurs critères et parfois plusieurs formalismes [Vroemen et Jager, 1997] . Les méthodes mixant un certain nombre de synthèses mono-objectif H_∞, H_2, LQG, l_1, etc, parfois antagonistes, ont pris une place importante dans la recherche en commande robuste comme en témoignent ces nombreuses publications : [Hindi *et al.*, 1998], [Clement, 2001], [Boyd et Barrat, 1991], [Scherer, 1995] et [Scherer, 2000]. L'une des commandes multi-objectif les plus connues est certainement la commande mixte H_2/H_∞ (un problème de commande encore ouvert). Si les travaux présentés dans [Doyle, 1978] ont montré que le régulateur LQG peut présenter des marges de robustesse vis-à-vis d'incertitudes paramétriques arbitrairement faibles, les compensateurs H_∞ ont également montré leur robustesse dans plusieurs applications pratiques. La synthèse H_∞ d'un côté est essentiellement fondée sur l'analyse de la performance dans le pire cas alors que la norme H_2 reflète une sorte de performance moyenne.

Le problème mixte $\mathrm{H}_2/\mathrm{H}_\infty$

La définition du problème mixte H_2/H_∞ se base sur le schéma standard de la Figure 13. Etant donné un niveau de performance H_∞ admissible γ, il consiste à déterminer un retour de sortie admissible ($K \in \Omega$) tel que :

$$\min_{K\in\Omega} \ \left\|T_{z_2\to d_2}\right\|_2$$
$$sous \ \left\|T_{z_\infty\to d_\infty}\right\|_\infty \leq \gamma$$

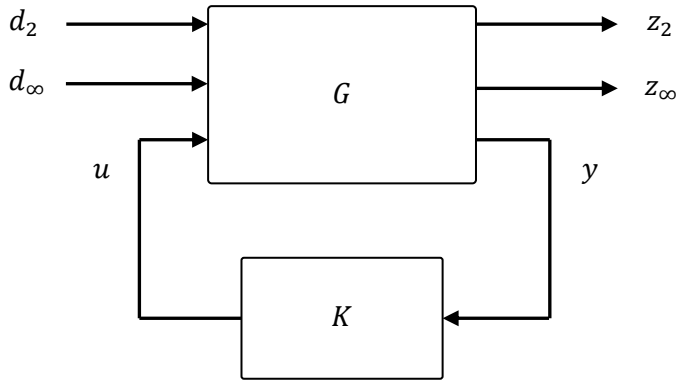


Figure 13: Forme standard de commande dans le cas mixte H_2/H_∞

Dans la même veine il est possible d'ajouter des contraintes au problème de commande en empilant les critères H_∞ par exemple tel que défini précédemment.

Pour résoudre ce type de problème, les approches, dites classiques, basées sur les techniques d'optimisation convexe (essentiellement la programmation semi-définie positive SDP [Boyd, 1994]) ou de résolution d'équations de Riccati [Doyle *et al.*, 1989], semblent, dans le contexte cité ci-dessus, céder la place à de récentes démarches d'optimisation hybride globale/locale. Dans notre étude nous focaliserons notre attention sur le développement d'algorithmes spécifiques d'optimisation non lisse. Ces approches permettent de calculer des solutions locales à la problématique de synthèse de lois de commande dans le contexte multi-objectif.

L'intérêt des approches non lisses, par rapport à des approches basées sur le paradigme de Lyapunov et le formalisme LMI, est qu'elles permettent d'optimiser aisément de multiples contraintes définies dans les domaines temporel ou fréquentiel, ainsi que des contraintes structurelles sur la loi de commande. Elles permettent également d'intégrer des spécifications au plus proche d'un cahier des charges. Les récents développements autour des méthodes d'optimisation non lisse ont donné lieu à la mise en œuvre de solveurs numériques tels que celui implémenté

dans la fonction *Systune*[6] de Matlab (Robust control Toolbox, Mathworks) ou encore HIFOO également implémenté sous Matlab, mais son code est ouvert et libre d'accès (voir les références suivantes : [Burke *et al.*, 2002], [Vanbiervliet *et al.*, 2009], [Bompart *et al.*, 2007], [Apkarian *et al.*, 2007a], [Apkarian, 2013], [Lewis, 2007] et [Gumussoy *et al.*, 2009]).

Par ailleurs, pour introduire au mieux le besoin de recourir à des méthodes d'optimisation non lisse, nous introduisons dans ce qui suit le problème de stabilisation d'un système dynamique par retour de sortie statique à travers la minimisation d'un critère non lisse -voire non Lipschitz- associé directement aux pôles de la boucle fermée.

Considérons le système LTI suivant :

$$G := \begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases} \tag{3.1}$$

Dans le cas d'un retour de sortie statique, on a :

$$u(t) = Ky(t) \tag{3.2}$$

En conservant les mêmes notations que précédemment et avec, A, B, C, et D les matrices de la **représentation d'état** du système et K le régulateur de sortie statique, i.e. une matrice de gains de dimensions appropriées.

Contrainte de stabilité

Le problème de stabilité du régulateur se définit par le théorème suivant :

- **Théorème** [Sadabadi et Peaucelle, 2016] et [Bompart, 2007] : Les assertions suivantes sont équivalentes et prouvent que le régulateur de sortie statique défini dans (3.2) stabilise le système (3.1) :
  - La partie réelle des valeurs propres de $\boldsymbol{A + BKC}$ sont toutes négatives.
  - Il existe une matrice symétrique $\boldsymbol{P}$ satisfaisant l'inégalité de Lyapunov suivante :

$$P > 0, \quad \{P(A + BKC)\} < 0$$

  - L'abscisse spectrale (*spectral abscissa* en anglais) de $\boldsymbol{A + BKC}$ notée $\boldsymbol{\alpha(A + BKC)}$ est négative :

[6] https://fr.mathworks.com/help/slcontrol/ug/systune.html dernière visite le 14/11/2017

$$\alpha(A + BKC) < 0 \quad (3.4)$$

Selon [Sadabadi et Peaucelle, 2016] et [Bompart, 2007] les difficultés de la stabilisation par un régulateur statique de sortie sont liées à la minimisation de l'abscisse spectrale $\alpha(A + BKC)$, de sorte qu'elle soit négative (cf. (3.4)). Cette fonction est continue mais non-Lipschitzienne en K. Elle ne possède donc pas de bonnes propriétés pour l'optimisation car elle est non-convexe et surtout non-lisse. [Sadabadi et Peaucelle, 2016] présente un état de l'art de l'ensemble des méthodes permettant de surmonter le problème du retour de sortie statique. Cet article conclut que dans le cas d'un régulateur de sortie statique les approches les **plus efficaces** sont celles basées sur **l'optimisation non-lisse** [Apkarian et Noll, 2006a], [Apkarian et Noll, 2006b], [Apkarian et Noll, 2006c], [Apkarian et Noll, 2006d] et [Apkarian *et al.*, 2008].

L'approche par optimisation non-lisse présente un certain nombre d'avantages, en particulier :

- Les variables de décision sont les paramètres du régulateur. Elles sont donc plus lisibles que dans le cas des LMI où les variables de décision sont les coefficients de la matrice de Lyapunov. On pourra aussi imposer des contraintes sur le régulateur : coefficients de la matrice de gain du contrôleur structuré.
- La dimension de l'espace de recherche est réduite au nombre de paramètres du régulateur et donc elle est indépendante de l'ordre du système que l'on cherche à commander [Bompart, 2007].
- Un certificat de convergence garantissant un optimum pour la solution trouvée.

En revanche, cette approche présente l'inconvénient de ne trouver qu'un optimum local pour la solution trouvée. Nous invitons le lecteur à se référer au chapitre 2 de la thèse [Bompart, 2007] pour des détails concernant l'optimisation non-lisse.

3.3.2 La commande à gains-séquencés

La commande à gains séquencés, en anglais *Gain-Scheduling*, est une méthode largement utilisée pour la conception d'asservissements de systèmes non-linéaires. L'idée générale de cette méthode est d'utiliser un modèle LPV du système considéré, c'est-à-dire un ensemble de systèmes LTI à différents points de fonctionnement, couvrant l'ensemble du domaine de fonctionnement du système. Un régulateur est synthétisé à chacun des points de fonctionnement, sur la base du modèle LTI correspondant. L'interpolation de ces régulateurs fournit alors un unique régulateur séquencé sur ces points de fonctionnement (*Gain Scheduled Controller*) [Rugh et Shamma, 2000]. Les paramètres de séquencement sont les paramètres

caractérisant la variation du modèle LPV, et par conséquent permettant de séquencer le régulateur.

L'une des méthodes les plus utilisées pour l'obtention d'un modèle LPV est de linéariser le modèle non-linéaire du système (lorsqu'il est connu) par des méthodes bien connues comme le développement en séries de Taylor au premier ordre.

Les avantages de cette approche sont :

- Cette approche est pragmatique et a montré un grand nombre de succès dans différents domaines d'application [Balas, 2002], [Bodenheimer et al., 1995] et [Tan et Packard, 2000] ?
- Elle facilite la compréhension d'un asservissement en gardant la représentation habituelle de la commande des systèmes linéaires (contrairement aux approches de la commande non-linéaire) [Rugh et Shamma, 2000].
- En effet, elle permet d'adapter rapidement le comportement du régulateur à celui du système [Rugh et Shamma, 2000].
- Sous certaines conditions, certaines approches proposées [Becker et Packard, 1991], [Scherer et Kose, 2012] et [Apkarian et Gahinet, 1995] fournissent des garanties de stabilité et de performances.
- Cette approche ne contraint pas la structure du régulateur à priori.

Les inconvénients de cette approche sont :

- Les preuves de stabilité et de performances ne sont garanties que localement, i.e. autour d'un point de fonctionnement [Rugh et Shamma, 2000], qui plus est, sous certaines hypothèses. Une étude ultérieure à la synthèse du régulateur à gains séquencés est requise (Analyse de Lyapunov) [Troffino, 1999], [Ramos et Peres, 2002], [Leite et Peres, 2004] et [Oliveira *et al.*, 2005].
- Le paramètre de séquencement doit varier « lentement » et doit être suffisamment représentatif du comportement non-linéaire du système (i.e. il doit permettre de représenter correctement le comportement du système sur l'ensemble du domaine de fonctionnement par la représentation LPV ou quasi-LPV) [Rugh et Shamma, 2000].
- Ainsi, le seul moyen de s'assurer de la conformité d'une telle régulation est de s'appuyer sur un nombre extensif de simulations.

Les travaux [Becker et Packard, 1991], [Scherer et Kose, 2012] et [Apkarian et Gahinet, 1995] ont permis d'apporter à cette approche des garanties de stabilité et de performance. Cependant, cette méthode impose **un fort conservatisme** ainsi que la **structure du régulateur** [Gahinet et Apkarian, 2013]**.**

3.3.3 La commande prédictive

La commande prédictive voit le jour dans les années 60 [Popoi, 1963] et connait ses premières applications industrielles dans les années 70 avec les travaux de [Richalet *et al.*, 1976]. L'idée générale de la commande prédictive est de calculer à chaque période d'échantillonnage du contrôleur la séquence des commandes à appliquer sur un horizon de prédiction [Dufour, 2000], à partir d'un modèle de connaissance du système. Dans sa version la plus répandue, seule la première commande est appliquée et l'algorithme recommence à la prochaine période d'échantillonnage.

L'algorithme consiste classiquement à résoudre un problème d'optimisation. Les différentes avancées [Grosdidier et al., 1988] ont permis de prendre en compte des contraintes dans le problème d'optimisation, permettant ainsi une meilleure prise en compte des spécifications industrielles. Les domaines faisant massivement appels à ces méthodes sont historiquement l'industrie du gaz, les procédés chimiques et pétroliers [Dufour, 2000].

Principe de la commande prédictive

La commande prédictive tire parti de la connaissance d'un modèle du système à contrôler pour simuler en temps réel son comportement sur un horizon de temps prédéfini et déterminer, en conséquence, les commandes optimales, de manière à :

- Minimiser une **fonction objectif**, formalisant un critère de performance défini selon les besoins de fonctionnement du système. Par exemple, le dépassement maximum de la température moyenne du le cœur.
- **Respecter des contraintes** de fonctionnement. Par exemple, les positions maximale et minimale autorisées des barres.

La fonction objectif et les contraintes sont formulées par le concepteur de la régulation dans le but de traduire les objectifs de commande (spécifications issues du cahier des charges) sous la forme de fonctions de coûts supportant la formalisation d'un problème d'optimisation mathématique.

La fonction objectif traduit généralement le critère de performance. Elle est formalisé par la fonction J qu'on cherche à minimiser (ou maximiser selon les cas). Cette fonction peut concerner :

- Les variables d'état du système représentées par le vecteur x.
- Les variables d'entrée du système représentées par le vecteur u
- Un comportement désiré c

On traduit alors le critère de performance par :

$$\min_{u\in\Omega}\left\{J(u)=\int_{Q} g\big(x(t),u(t),c(t)\big).dt\right\}$$

où Ω est l'espace de validité des contraintes et Q un cylindre dont la base est définie par les variables x, u et c et de longueur le temps de prédiction [Dufour, 2000].

Le problème de commande se résume donc à un problème d'optimisation sous contraintes, les variables de décision étant les entrées de commande du système u.

Afin de mettre en jeu un nombre fini de variables de décision, (nombre fini de d'échantillons de commandes), et obtenir ainsi un problème d'optimisation en dimension finie, on discrétise le plus souvent l'espace de décision défini par la fonction u. Il est procédé le plus souvent à une projection des entrées sur une base de fonctions polynomiales :

$$u(t)=\sum_{i=0}^{n} a_i t^i$$

Ceci est illustré à l'ordre 2 par la Figure 14.

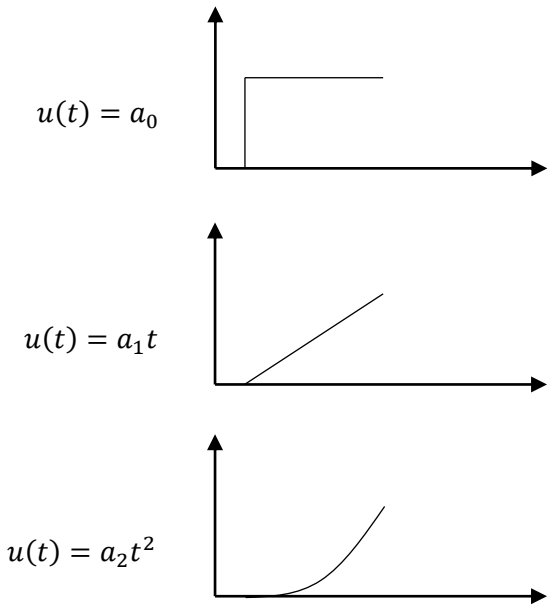


Figure 14: Base de fonctions pour la commande prédictive

Ainsi, les variables de décision pour l'optimisation sont les coefficients a_i des fonctions polynomiales définies ci-dessus.

Trois paramètres clés sont habituellement considérés pour le « réglage » de la commande prédictive (tel que présenté sur la Figure 15) [Dufour, 2000] :

- La période d'échantillonnage de la commande T_e

Ce paramètre définit le temps entre deux mises à jour des entrées de commande. C'est alors aussi le temps maximal pour le calcul de ces dernières. Ce paramètre est particulièrement dimensionnant pour la commande prédictive.

- L'horizon de commande noté N_c

L'horizon de commande définit la longueur des suites d'arguments de la commande. Il représente donc le temps pendant lequel la suite des commandes optimales est calculée.

- L'horizon de prédiction noté N_p

L'horizon de prédiction définit la longueur sur laquelle est évalué le problème de commande.

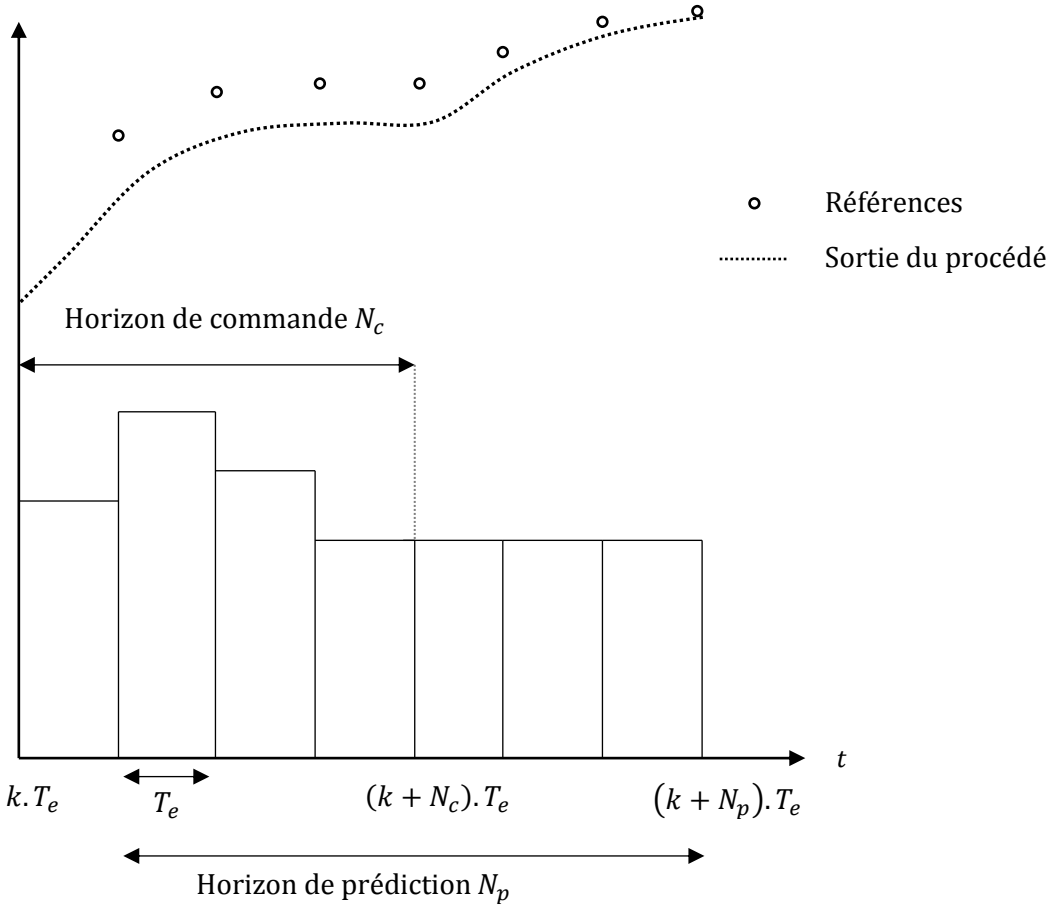


Figure 15: Principe de la commande prédictive

Stabilité du problème

Quelle que soit la stratégie de commande utilisée, une étude de stabilité est toujours nécessaire. Elle consiste à vérifier que la commande, solution de l'algorithme de commande prédictive utilisée, permet d'assurer la stabilité du procédé en boucle fermée [Dufour, 2000]. Il est possible, dans certains cas d'assurer la stabilité d'un problème de commande prédictive. Le cas le plus simple est le cas où l'horizon de prédiction N_p est infini. [Keerthi et Gilbert, 1988] introduit une contrainte terminale égalitaire qui, dans le cas infini, garantit la stabilité pour un modèle linéaire. Selon [Dufour, 2000], dans le cas non-linéaire cette méthode n'est à priori pas applicable. De plus, dans le cas d'un horizon de prédiction fini, la commande prédictive ne garantit pas, dans le cas général, la stabilité.

Formalisation du problème de commande

La formalisation du problème de commande consiste à formuler la fonction objectif (critère de performance) J et les contraintes répondant au cahier des charges. L'algorithme consiste alors à calculer à l'instant $t = k.T_e$ la séquence des N_c commandes optimales à appliquer sur l'horizon de prédiction N_p. La fonction objectif à minimiser fait intervenir le vecteur des commandes du processus u, le vecteur des sorties (estimées ou mesurées) y_p et g une fonction formalisant le critère de performance. L'objectif de minimisation se formule généralement par l'équation suivante :

$$\min_{\tilde{u}} J(\tilde{u}) = \sum_{j=k+1}^{k+N_p} g(u(j-1), y_p(j))$$

Avec $\tilde{u}$ la séquence des N_c commandes calculée par l'algorithme. On la définit de la manière suivante :

$$\tilde{u} = [u(k) \quad \dots \quad u(k+N_c-1)]^T$$

NB : pour tout signal s, $s(j)$ est défini comme la valeur de $s(t)$ à l'instant $t = j \times T_e$, avec T_e la période d'échantillonnage.

Les commandes restantes à appliquer jusqu'à la fin de l'horizon de prédiction sont alors :

$$u(j-1) = u(k+N_c-1) \quad (j \in \{k+N_c+1, k+N_p\})$$

Le problème d'optimisation est soumis aux contraintes portant sur les mêmes variables, tel que :

$$\begin{cases} c_i\left(u(j-1), y_p(j)\right) \leq 0 \\ i \in \text{ג} = \{1, \dots, n_{co}\} \\ j \in \text{ד} = \{k+1, \dots, k+N_p\}) \end{cases}$$

Avec,

$$\begin{cases} n_{co} \text{ le nombre de contraintes} \\ u \in \mathbb{R}^m \\ y_p \in \mathbb{R}^p \\ m \text{ le nombre de commandes} \\ p \text{ le nombre de sorties} \end{cases}$$

Résolution du problème

La résolution de problèmes d'optimisation sous contraintes de cette nature, appliquée aux problèmes de commande prédictive en particulier, est largement documentée. Nous invitons le lecteur à se référer à [Dufour, 2000] faisant un tour d'horizon des méthodes de résolution de commande prédictive sous contrainte.

Partie II.
Méthode d'Ingénierie Système

4 Introduction de la partie IS

Notre étude a pour but de développer une méthode outillée d'ingénierie s'inspirant de l'Ingénierie Système et adaptée au contexte et enjeux de Framatome. Cette méthode permet de :

- Supporter le passage de l'Ingénierie des Exigences à l'Ingénierie des Architectures en privilégiant une approche basée sur des modèles.
- Guider la conception.
- Introduire de la vérification des modèles. Grâce notamment à des propriétés modèles et des capacités de simulation.

On définit une méthode par les 5 composantes présentées sur la Figure 16. A savoir :

- **Les concepts et les relations sémantiques entre ces concepts**, relevant du domaine métier qui nous intéresse. Il s'agit donc d'une part des concepts métier que Framatome utilise pour la spécification des exigences et la description et la vérification des architectures fonctionnelles et organiques. D'autre part, il s'agit de certains des concepts IS jugés pertinents dans le cadre de notre étude car ils sont liés aux besoins et pratiques de Framatome. Cette définition est indispensable afin de couvrir les besoins de description des systèmes dont l'ingénierie est le cœur de métier de Framatome et pour réconcilier, voire améliorer, les pratiques métier de l'entreprise. La liste de ces concepts doit cependant rester ouverte à tout enrichissement futur. Comme nous le verrons dans la suite, l'ensemble des concepts et des relations est formalisé sous forme d'un méta modèle (cf. 2.1) unifié que nous présentons dans la suite. Ce dernier définit la syntaxe abstraite des DSML.
- **Les langages de modélisation (DSML)**. Ils permettent de manipuler sans ambiguïté l'ensemble des concepts pour bâtir des modèles puis analyser et raisonner sur la base de ces modèles des systèmes dont l'ingénierie est l'objectif de Framatome. Il s'agit donc de DSML. Pour les besoins de vérification et de simulation, ces DSML seront dotés de capacités et de règles 1) d'exécution pour promouvoir une simulation directe, et 2) de description et de preuve directe des propriétés (cf. [Nastov, 2016]) attendues du modèle comme du système modélisé. Cette vérification comme cette simulation ne doivent pas nécessiter de transformation des modèles vers des modèles tiers prouvables via des techniques formelles ou exécutables via des simulateurs. Ainsi, nous couvrons ici les besoins d'autonomie des acteurs métier quant à ces transformations, à la facilité et à la rapidité de mise en œuvre et à la qualité des simulations.

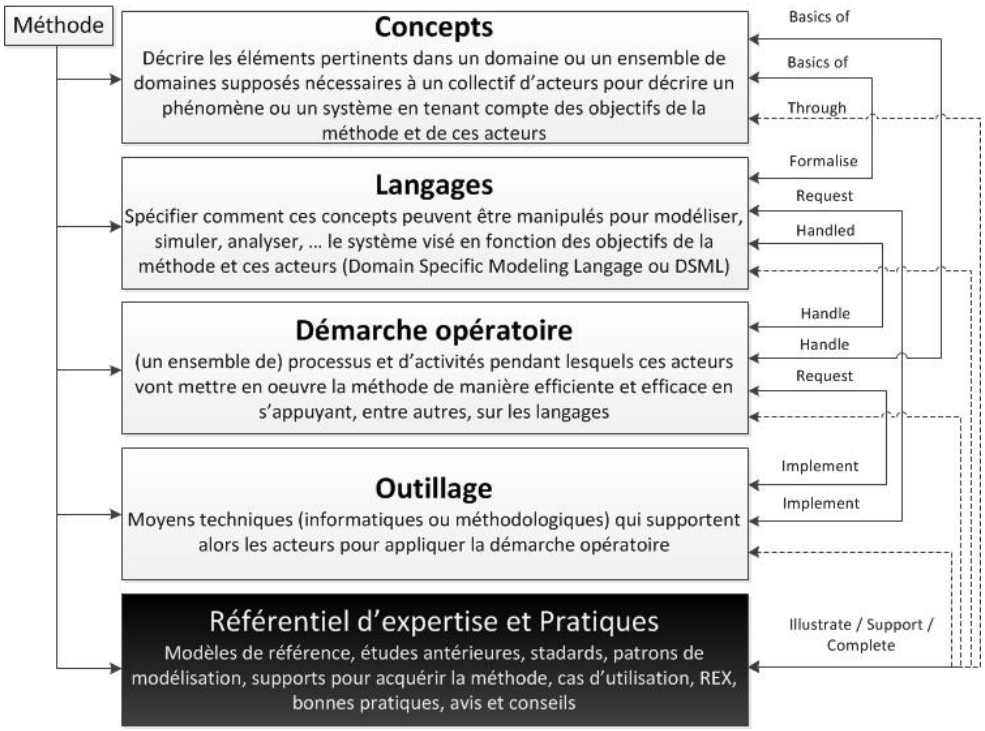


Figure 16: Eléments de composition d'une méthode

- **La démarche opératoire** permet de guider les acteurs métier dans l'utilisation de la méthode et définit donc un processus avec des activités, des rôles et des règles de modélisation, de vérification et de simulation.
- **Les outils** sont ici essentiellement des outils informatiques qui permettent de mettre en œuvre la démarche opératoire, de faciliter la mise en œuvre des langages au cours des différentes activités et d'en contrôler le cheminement.
- **Le référentiel d'expertise** est enfin l'ensemble des bonnes pratiques, des REX et autres modèles de référence, qui est à bâtir et à enrichir au fur et à mesure de l'utilisation de la méthode par différents acteurs selon des contextes et des attentes particulières. En particulier, on considèrera parmi les modèles de référence un ensemble de patrons [Pfister *et al.*, 2012] de conception architecturale, de rédaction d'exigences et de construction de scénarios opérationnels pour guider voire faciliter les activités de modélisation.

L'ingénierie de Framatome possède des spécificités. Premièrement, Framatome utilise le cycle RADIV évoqué partie 2.2 (cycle en V usuel revisité par Framatome pour correspondre aux pratiques de l'entreprise), reprend et organise l'ensemble des processus de conception à réaliser à chaque niveau système (à chaque nœud du PBS). Ensuite, le PBS, i.e. l'arborescence des systèmes de la centrale, est supposé connu à l'avance. Notons qu'un référentiel d'exigences (Requirement Baseline) est associé à chaque nœud du PBS comme présenté sur la Figure 18. Enfin, l'ensemble des éléments de modélisation, notamment les systèmes, les fonctions et les exigences sont rassemblés et gérés dans un outil de gestion du cycle de vie produit appelé Teamcenter. La méthode devra donc s'intégrer dans le processus de Framatome (RADIV) mais aussi se conformer aux outils choisis (Teamcenter) et aux

pratiques de l'industriel (en particulier le choix structurant qu'est l'IS centrée sur un PBS supposé déjà connu).

La méthode d'IS proposée couvre les deux phases de conception que sont l'ingénierie des exigences et l'ingénierie des architectures. Comme le montre la Figure 17 cela correspond aux deux processus proposés dans le RADIV : *Requirements Capture and Analysis* et *Architecture Design*. La Figure 17 montre notamment que les entrées de la méthode seront les référentiels d'exigences associés à chaque système (nœud du PBS). La méthode nourrira l'ensemble des sous-systèmes et de leurs référentiels au moyen des DSML mis en place.

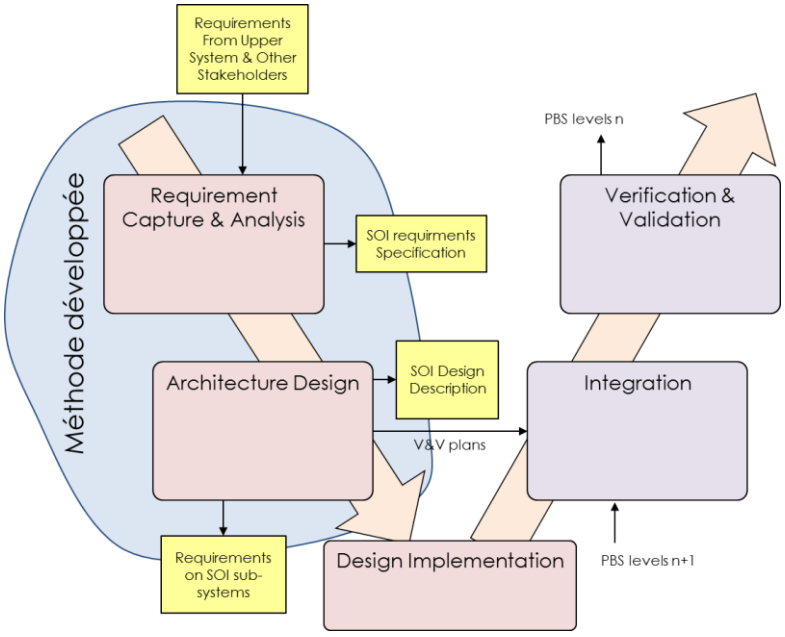


Figure 17: Positionnement de la méthode dans le RADIV

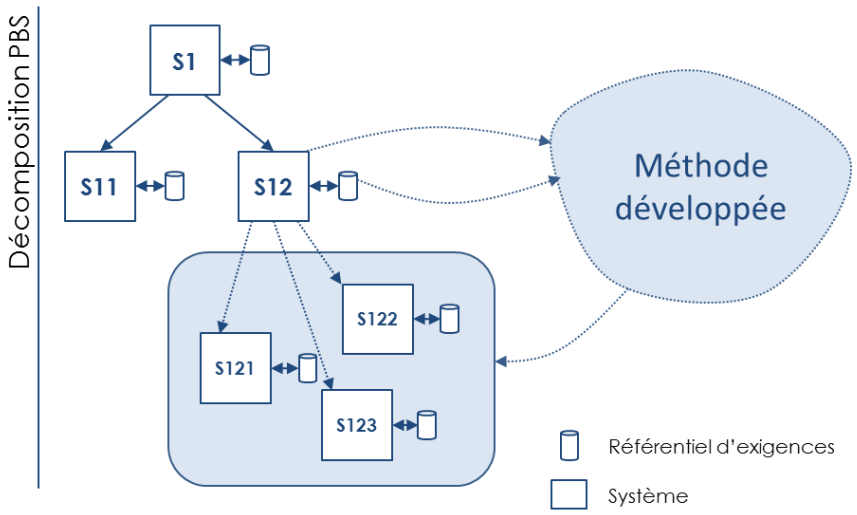


Figure 18: Méthode proposée vs. PBS et référentiels d'exigences

La démarche employée pour développer la méthode est présentée sur la Figure 19. Elle se déroulera de la manière suivante :

- Tout d'abord, nous verrons les différents concepts nécessaires à la description des différents langages.
- Dans un second temps, nous définirons l'ensemble des langages de modélisation (DSML).
- Ensuite, nous étudierons la démarche opératoire qui permet d'organiser et structurer le travail des ingénieurs métier utilisant la méthode, en particulier les activités de modélisation, de vérification et de simulation puis de décision évoquées plus haut.
- Dans une quatrième partie, nous verrons les outils développés. Ces outils supportent ces acteurs métier tout au long de la démarche pour modéliser, vérifier et simuler en vue de fournir des justifications vers les activités de décision.
- Enfin nous présenterons un cas d'utilisation du *Core Control* faisant office de référentiel d'expertise.

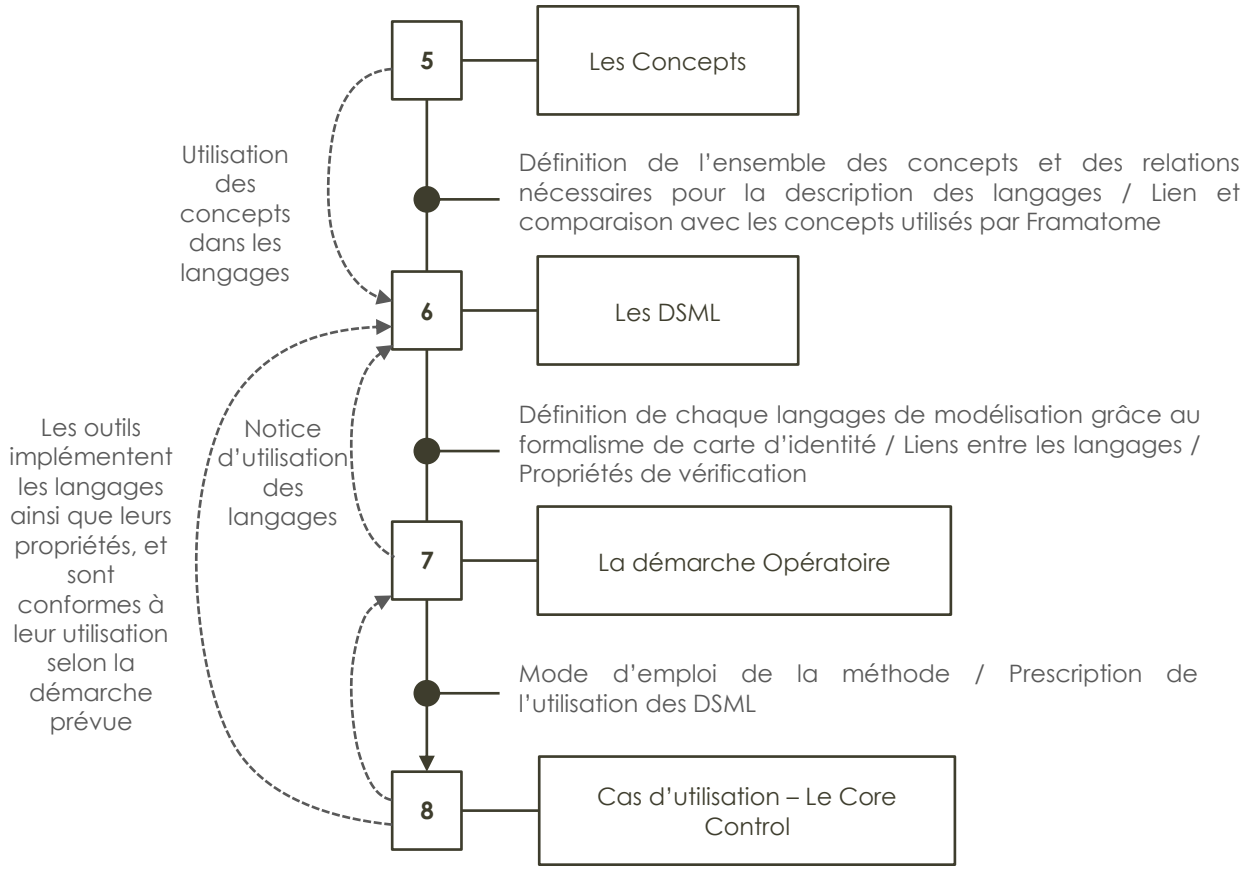


Figure 19: Démarche employée pour développer la méthode

5 Les Concepts

Sommaire

5.1 Le méta modèle 80
5.2 Le système 80
5.2.1 Le composant 81
5.2.2 L'interface 81
5.2.3 Composant logique 82
5.3 L'Exigence 84
5.3.1 Le modèle d'exigences 84
5.3.2 Propriétés des exigences 85
5.3.3 Analyse d'impact des exigences 86
5.3.4 Les types d'exigences 86
5.3.5 Les attributs de l'exigence 87
5.3.6 Les relations inter-exigences 88
5.3.7 Les relations entre exigences et autres éléments de modélisation 90
5.4 La Fonction 90
5.4.1 Le Flux 91
5.4.2 L'Item 92
5.4.3 Le service 92
5.5 Architecture 93
5.6 Bloc Système 94
5.7 Mode Opérationnel et Transition 95
5.8 Le scénario opérationnel 97
5.9 Notion de vue 97
5.10 Les propriétés 99
5.11 La sémantique opérationnelle d'un DSML 101
5.11.1 Le moniteur de DSML 101
5.11.2 Le Blackboard 102
5.11.3 La sémantique opérationnelle 103

Ce chapitre a pour but de présenter l'ensemble des concepts que nous utiliserons par la suite dans les DSML. Un concept est un élément abstrait du langage qui permet de représenter une idée ou un phénomène dans un domaine métier particulier. Ces concepts seront ensuite structurés et adaptés au processus RADIV de Framatome dont ils sont parfois absents.

5.1 Le méta modèle

Un méta modèle modélise l'ensemble des concepts et leurs relations qui sont utilisés pour définir les DSML par la suite. Ainsi, le méta modèle est communément adopté pour représenter la syntaxe abstraite d'un DSML. Nous en donnons un exemple Figure 20. Notre étude a abouti à la réalisation d'un méta modèle unifié adapté à Framatome. Tout au long de la présentation des concepts et des DSML, nous donnerons des parties ciblées du méta modèle.

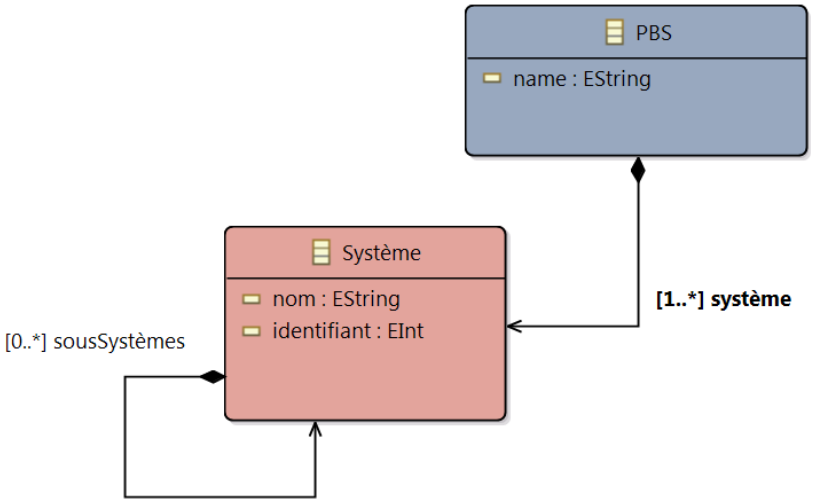


Figure 20: Exemple de Méta modèle

5.2 Le système

Le Système est défini par [ISO15288, 2015] comme « *une combinaison d'éléments en interaction, organisés pour atteindre un ou plusieurs objectifs bien définis* ».

Dans les pratiques Framatome, un système peut être décomposé en sous-systèmes (*subSystems*) et, comme énoncé plus haut, cette décomposition est alors représentée sous forme d'un PBS qui ne peut être donné ici pour des raisons de confidentialité de Framatome.

Un système possède une mission (*missionLitteralDescription*) et une finalité (*purposeLiteralDescription)* ainsi qu'un type (*systemType*).

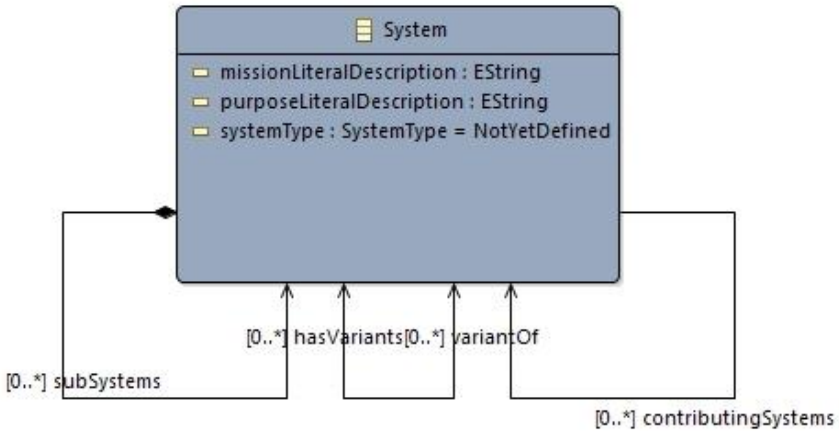


Figure 21: Le concept Système dans le méta modèle proposé

5.2.1 Le composant

Le Système est décomposé en composants qui peuvent soit être : des liens, d'autres systèmes ou des interfaces (cf. Figure 22). Par rapport aux pratiques Framatome on apporte ici un formalisme plus complet pour la définition du PBS.

Ce méta modèle traduit le fait qu'un système est un type de composant. Il possède ainsi au moins une interface. Les éléments du PBS peuvent être des liens. Ces dernières relient les systèmes entre eux.

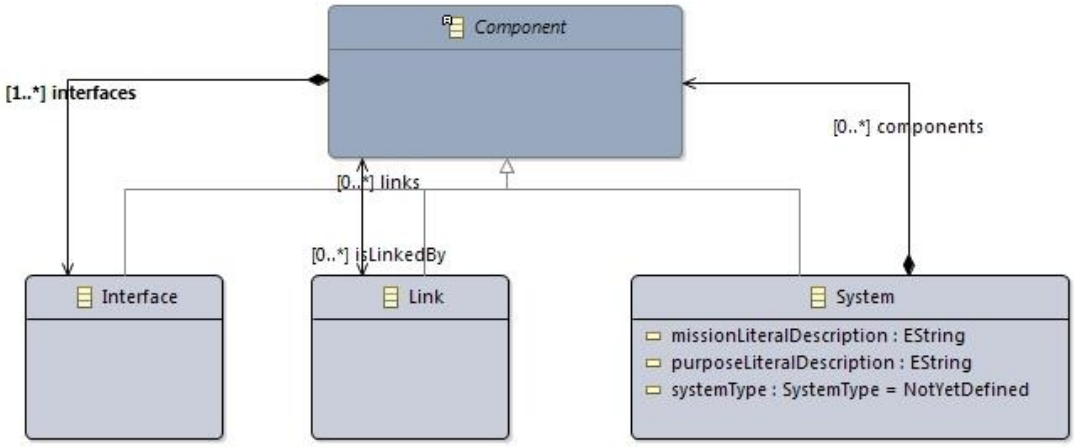


Figure 22: Méta modèle Système et Composant

5.2.2 L'interface

L'interface est définie comme : « la frontière commune à deux ou plusieurs constituants (ou au système et à son environnement) au niveau de laquelle les règles

d'échange, de compatibilité, d'intégrité et de non régression sont à respecter au cours de la vie du système ».

La Figure 23 montre l'interface entre le cœur et les boucles. L'interface supporte les échanges entre le SOI et les systèmes dits à l'interface.

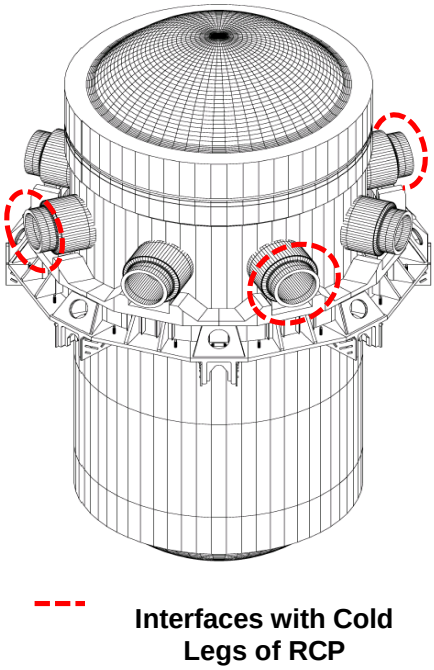


Figure 23: Exemple d'interface – Cœur

5.2.3 Composant logique

Le composant logique est une représentation « physique » idéale d'un système, observé au plus près de son comportement fonctionnel. Il permet de faire abstraction de toute représentation organique et de rapprocher des systèmes qui ont des interactions opérationnelles et fonctionnelles fortes. Par exemple, le cœur et l'algorithme de contrôle qui le commande, le *Core Control,* possèdent des interactions fonctionnelles importantes, bien qu'ils soient éloignés dans le PBS (le cœur est dans le réacteur et le *Core Control* dans les armoires de contrôle-commande). Ainsi, le composant logique est à mi-chemin entre une représentation fonctionnelle et organique.

Dans le cas d'une IS basée sur un PBS déjà connu, certains systèmes sensés collaborer sont éloignés les uns des autres dans le PBS car ils ne font pas partie de la décomposition du même sur-système. Afin de tenir compte plus précisément des interactions fonctionnelles entre ces sous-systèmes nous proposons de privilégier le **paradigme logique**. Il consiste à changer le point de vue de la représentation du

système donnée par le PBS en une représentation mettant en évidence les éléments qui, fonctionnellement parlant, interagissent.

L'architecture logique consiste alors à définir les liens entre les différents composants logiques (cf. 2.3.3).

Les Figure 24 et Figure 25 montrent respectivement un exemple d'architecture logique et un exemple de PBS. Sur la Figure 25 on peut voir que le système *Core Control* est dans une branche différente du PBS que le système *Core (Cœur)*. Or, la Figure 24 montre que ces deux systèmes ont des interactions qu'il parait important de représenter. C'est le système *reactor* qui reçoit l'ensemble du besoin de fonctionnement du cœur, dont certaines exigences de contrôle commande. Il créé alors le composant logique *Core* et le composant logique *Core Control* afin de faire l'ingénierie de ces derniers.

Ainsi, la vue logique permet bien de mettre en interaction ces deux composants, sans pour autant remettre en question la décomposition PBS.

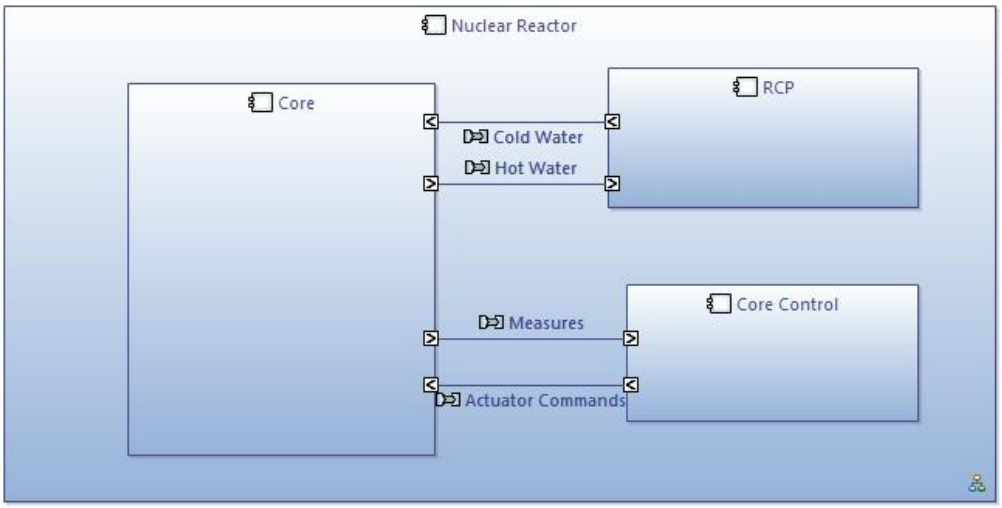


Figure 24: Exemple d'architecture logique- Figure réalisées par l'outil Capella

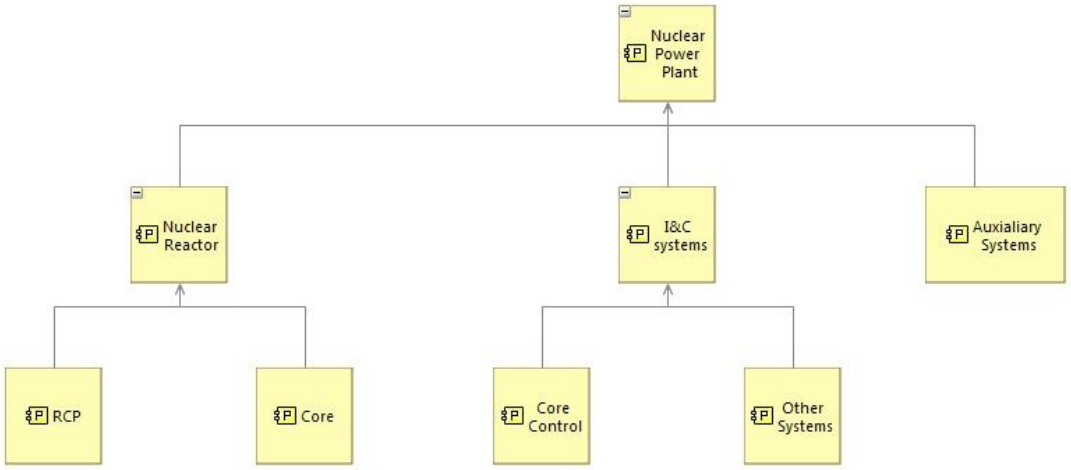


Figure 25: Exemple de PBS - Figure réalisées par l'outil Capella

5.3 L'Exigence

Selon [AFIS, 2012], une exigence est un « *énoncé qui spécifie une aptitude, une caractéristique ou une limitation d'un système, d'un produit ou d'un processus, pour contribuer, dans des conditions d'environnement données, à la satisfaction d'un but donné* ».

Une exigence peut se représenter et se manipuler sous une forme textuelle (la plus souvent privilégiée) ou graphique. L'exigence est un concept central dans une approche MBSE. En effet, une exigence est un engagement pris par le concepteur du système vis-à-vis de ses parties prenantes et traduit donc des besoins avérés de ces dernières. Les exigences guident donc le processus de conception et doivent être spécifiées et raffinées, puis tracées à des fins de justification, pour chaque niveau système (et donc à chaque nœud du PBS). Les exigences sont regroupées dans un référentiel d'exigences attaché à ce nœud et précisant donc les engagements pris par Framatome auprès de ses clients. L'approche que nous proposons est que le partage de l'information entre les différents niveaux système et l'interopérabilité entre les acteurs métier pour la conception se fasse par le partage des exigences à chaque niveau système. Nous formaliserons cette vision grâce notamment au concept de Bloc Système décrit en partie 5.6.

5.3.1 Le modèle d'exigences

Les exigences créées dans le but de spécifier un système (exigences allouées au SOI, à ses fonctions ou tout autre élément de modélisation qui le concerne) et les relations qu'elles ont avec les éléments de modélisation (autres exigences, fonctions, sous-systèmes, documents de justification, etc.) constituent ce que l'on appelle le **modèle d'exigences** illustré sur la Figure 26. Les exigences peuvent être liées soit :

- A d'autres exigences. Nous définirons les relations inter-exigences en partie 5.3.6.
- A des éléments de modélisation : un système, une fonction, un mode ou encore un scénario opérationnel.

On peut voir le modèle d'exigence comme une toile dont les intersections sont les exigences et les extrémités sont des éléments de modélisation. Ce modèle d'exigence est particulièrement important dans notre démarche car il comporte l'ensemble des interconnections entre tous les éléments de modèle. C'est le support de la vérification des exigences que nous présenterons dans la partie suivante.

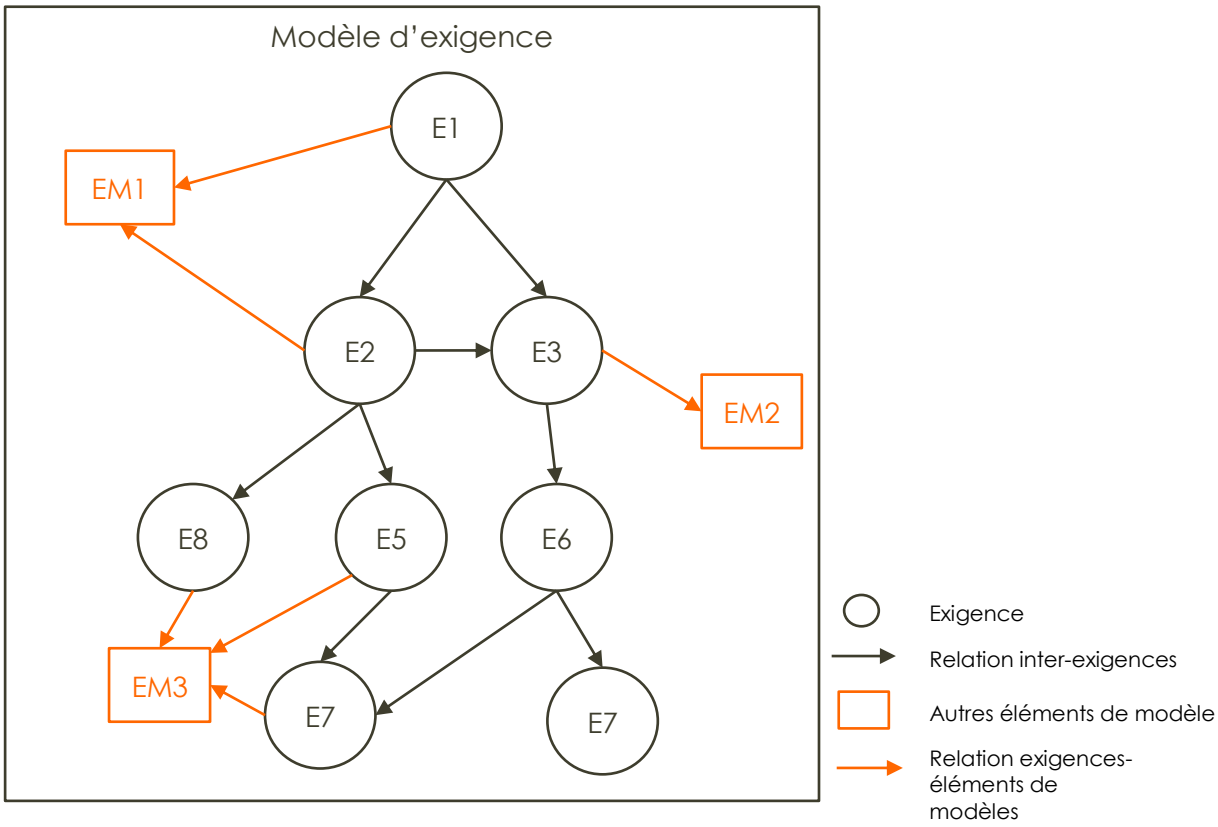


Figure 26: Schéma de principe du modèle d'exigences

5.3.2 Propriétés des exigences

La problématique principale est que, bien souvent écrite dans un langage textuel non-structuré et naturel, la formulation de l'exigence peut être ambigüe rendant son interprétation plus difficile. Certains travaux [SEBoK, 2016] ont mené à l'élaboration de caractéristiques qu'une exigence doit posséder :

- **Nécessaire**. L'exigence définit une capacité essentielle.
- **Non-ambigüe**. Elle ne peut être interprétée que d'une manière.
- **Cohérente**. L'exigence ne contient pas d'éléments contradictoires.

- **Unique**. L'exigence, telle que formulée, ne nécessite pas de plus de détails ou de raffinement pour être comprise.
- **Réalisable**. Elle est techniquement réalisable.
- **Traçable**. Il est possible de tracer l'exigence à partir d'exigences amonts ou d'autres sources et ce, jusqu'aux composants ou aux fonctions auxquelles elle est allouée.
- **Vérifiable**. Il existe des éléments pour la vérification de l'exigence et la justification de son respect par le SOI ou par le modèle global du SOI lui-même composé de l'ensemble des modèles du SOI, vue par vue.

5.3.3 Analyse d'impact des exigences

L'analyse d'impact consiste à analyser l'impact de la modification d'une exigence sur le reste du modèle du SOI en se référant aux différentes relations existantes entre les instances des concepts de modélisation (e.g. relations inter-exigences, exigence-mode, exigence-fonction ou exigence-composant). Par exemple, si l'exigence E1 est liée à une autre exigence E2 par une relation alors, toute modification de E1 peut impacter la définition de E2 selon la nature de la relation qui peut exemple être une relation de raffinement (cf. Figure 26). La réciproque peut de fait être aussi vraie. C'est aussi valable pour d'autres relations (e.g. exigence fonctionnelle allouée à une fonction). Puisque le référentiel des exigences contient l'ensemble des exigences et les relations qu'elles possèdent, il est l'objet d'analyses d'impacts de la modification d'une exigence sur l'ensemble des éléments de modèle (dont les exigences) « connectés » à cet exigence.

5.3.4 Les types d'exigences

L'exigence peut être de 3 types différents : fonctionnelle, non-fonctionnelle ou de performance. Il arrive parfois qu'on classe la performance dans le non-fonctionnel. Ces types permettent d'allouer les exigences à différents éléments du modèle comme des systèmes, des fonctions (cf. 5.4) ou encore des processus (nous n'étudierons pas ce cas dans notre étude). D'après notre recherche bibliographique nous avons rassemblé et concilié les différents types d'exigences existants dans le tableau suivant :

Type de l'Exigence		Description	Allocation
Fonctionnelle		Prescription du comportement d'une fonction	Fonction
Performance		Prescrit la performance attend d'un système (e.g. le débit maximum du REA de 36 m^3/h)	Fonction
Non-fonctionnelle	Design	Prescrit tout ce qui est relatif à la conception physique d'un élément de modèle (e.g. dimensions, couleur, etc.)	Système

	Processus	Prescrit les lois, les pratiques, les règles devant être respectées (e.g. respect de l'ISO 15288)	Système
	Contrainte	Prescrit des contraintes (e.g. la température maximale en entrée du GV de 330°C)	Système
	Interface	Prescrit tout ce qui attrait à une interface entre deux systèmes	Système
	V&V	Définit les plans de vérification et de validation selon les processus du cycle en V	Système
	-ilités	Exigences support (e.g. disponibilité, maintenabilité, vulnérabilité, sûreté, etc.)	Système

Tableau 1: Les types d'exigences et leur allocation

Les différents types d'exigences sont des éléments permettant de mettre en place de la vérification, notamment par l'analyse de l'allocation des exigences selon leur type à des éléments de modélisation. Par exemple, une exigence de dimensionnement (design) allouée à une fonction serait une erreur.

5.3.5 Les attributs de l'exigence

Une exigence possède les attributs décrits dans le tableau suivant :

Nom de l'attribut	Type de l'attribut	Description
Racine (*root*)	Booléen	Est vraie si l'exigence est une exigence origine, i.e. qu'elle n'est pas dérivée, raffinée ou décomposée d'une autre exigence, faux sinon.
Priorité (*priority*)	Entier	Indique le niveau de priorité de l'exigence (0 étant la priorité la plus faible et 3 la plus forte ???).
Criticité (*criticity*)	Entier	Variable indiquant le niveau de criticité de l'exigence. Plus sa valeur est grande plus la criticité est importante.
Flexibilité (Flexibility)	Entier	Variable indiquant la flexibilité, i.e. la possibilité de faire varier l'exigence (à 0 elle ne peut pas être changée et à 100 elle peut être changée si nécessaire).
Satisfaction (isSatisfied)	Booléen	Variable indiquant si l'exigence est satisfaite.
Justification (isJustified)	Booléen	Variable indiquant si l'exigence est justifiée (au sens d'une preuve de justification type document/étude justifiant l'existence de l'exigence).
Type (type)	Catégorie	Variable indiquant le type d'exigence (cf. ???).
Contenu (*content*)	Chaine de caractères	Enoncé textuel de l'exigence.

Tableau 2: Attributs d'une exigence

La plupart de ces attributs ont pour but d'aider à la gestion et la vérification des exigences. Par exemple, pour la gestion, en classant les exigences par ordre de priorité et/ou de criticité.

5.3.6 Les relations inter-exigences

Comme nous l'avons montré dans la partie 5.3.1, les exigences sont reliées entre-elles. Notre recherche bibliographique nous a mené à de nombreux travaux sur les relations inter-exigences. Par exemple, SysML définit 3 relations entre les exigences : la décomposition, la dérivation et le raffinement. En plus de ces trois relations nous proposons d'en rajouter une quatrième nommée Reformulation (*Rewording*). La Figure 28 formalise l'ensemble de ces relations dans une partie du méta modèle développé. Ainsi, les différentes relations que nous proposons d'utiliser sont :

- La décomposition

Elle consiste à décomposer une exigence en plusieurs exigences dans le but d'en réduire la complexité. Par exemple, l'exigence « The Core Control shall enable manual and automatic average temperature control » peut être décomposée en « The Core Control shall enable manual average temperature control » et « The Core Control shall enable automatic average temperature control ».

- La dérivation

Cette relation permet de déduire d'une collection d'exigence une nouvelle collection d'exigence afin de spécifier le comportement ou l'état d'un système lorsqu'il se trouve dans une configuration particulière. Cette relation permet de changer de niveau des exigences. Elle est particulièrement utile lorsque l'on souhaite tracer des exigences du SOI vers celles de ses sous-systèmes.

Par exemple, une exigence portant sur l'accélération d'un véhicule peut être dérivée en une exigence sur la puissance moteur, une autre sur la taille des roues et une dernière sur le poids du châssis.

- Le raffinement

Le raffinement d'une exigence a pour but d'apporter du détail sur une exigence, bien souvent dans le cas où l'abstraction d'une exigence est trop forte. Le raffinement peut définir et préciser l'ensemble des conditions d'opération nécessaire à la réalisation d'une exigence. Par exemple, l'exigence E1 : « le Core Control doit effectuer des rampes à 5% par minute sans atteindre les LCO » prise seule, n'a aucun sens. Il peut être nécessaire de spécifier les conditions d'opération dans lesquels elle est valable par exemple, « en fonctionnement normal » ou encore « en

fonctionnement dégradé ». Autre exemple, l'exigence peut nécessiter des précisions par exemple : « la LCO AO peut être dépassée pendant un maximum de 100 secondes ».

- La reformulation

La relation de reformulation permet relier deux exigences écrites dans des langages différents mais ayant une équivalence complète. Par exemple, l'exigence E1 : « y1 = e1 + e2 » est équivalente à l'exigence graphique Figure 27.

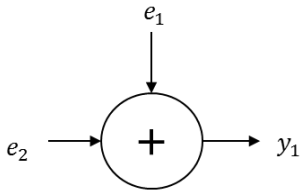


Figure 27: Exemple pour le raffinement d'une exigence

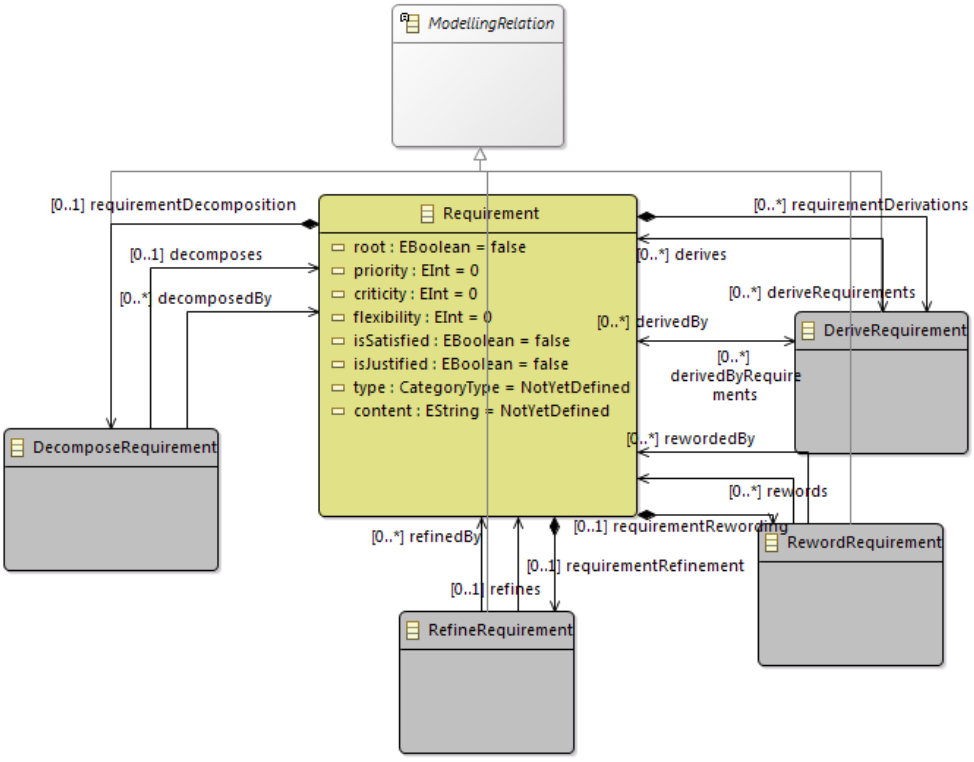


Figure 28: Méta modèle de l'Exigence et ses relations

5.3.7 Les relations entre exigences et autres éléments de modélisation

L'exigence possède aussi des relations avec d'autres concepts. Par exemple un système, une fonction ou un composant (de type rôle alloué à un acteur humain, matériel ou logiciel). Nous considérerons qu'une exigence peut être :

- **Allouée à** : un composant i.e. à un système, une interface, un lien, un rôle, etc. Nous considérerons que seules les exigences non-fonctionnelles peuvent être allouées aux composants. Par exemple, une exigence de dimensionnement (dimension d'un système) est une exigence non-fonctionnelle qui ne peut être allouée qu'à un système.
- **Réalisée par** : une fonction. Nous considérerons que seules les exigences fonctionnelles et de performance sont réalisées par les fonctions.
- **Evaluée au moyen :** de variables de modélisation et de modèles. Cette relation est notamment utile pour la vérification par simulation. Pour cela, on associe à une variable de modélisation des critères d'évaluation à respecter.
- **Exprimée par** : une (ou plusieurs) partie(s) prenante(s).
- **Raffinée par** : un (ou plusieurs) scénario(s) opérationnel(s). Le raffinement a ici le même sens que la relation du même nom dans le cas inter-exigences. Elle permet d'ajouter du détail à l'exigence en se servant de scénarios opérationnels.
- **Induite par** : un mode, un composant, d'autres exigences, une fonction, ou un scénario opérationnel i.e. par un choix de conception.
- **Valable dans** : un (ou plusieurs) mode(s) de fonctionnement et/ou pour un (ou plusieurs) scénario(s) opérationnel(s).

5.4 La Fonction

Selon [AFIS, 2012] une Fonction « *transforme des flux entrants en flux sortants* ». Framatome suit la définition de l'INCOSE [SEBoK, 2016] qui définit une fonction comme « *une tâche caractéristique, une action ou une activité qui doit être réalisée pour obtenir un résultat attendu. Une fonction peut être réalisée par un ou plusieurs systèmes, y compris des équipements matériel, logiciel, humain ou autre activité* ».

On distingue les fonctions internes des fonctions externes. Les fonctions externes sont définies comme les fonctions allouées au SOI, i.e. les fonctions qu'il est sensé réaliser d'un point de vue boîte noire. Autrement dit, ce qui est attendu du système. Les fonctions internes caractérisent le comportement boite blanche du système, i.e. les fonctions réalisées par les sous-systèmes du SOI.

5.4.1 Le Flux

Le flux représente le déplacement d'un ensemble d'éléments ou Items, pouvant être de différentes natures. Le flux peut être de trois natures : d'énergie (e.g. un flux de chaleur), (ii) de matière (e.g. flux de marchandises) et (iii) d'information (e.g. flux de données).

Un flux possède des caractéristiques. Par exemple, un débit, une température et/ou une pression. Ces caractéristiques sont variables dans **l'espace** et dans **le temps**. Par exemple, lorsque le flux d'eau traverse le cœur, un échange de chaleur intervient (modification des caractéristiques du flux), provoquant une élévation de la température du fluide. Ainsi, dans la partie haute et la partie basse du cœur le flux possède des caractéristiques différentes. De même, si la température d'entrée varie dans le temps, chaque point où l'on évalue la caractéristique du flux (température dans le haut du cœur) varie bien dans le temps.

Le flux transporte un ou plusieurs Items. La nature des Items correspond à celle du flux. Par exemple, le flux d'eau du circuit primaire, est un flux transportant différents Items de type matière : de l'eau, du bore et des produits de fission.

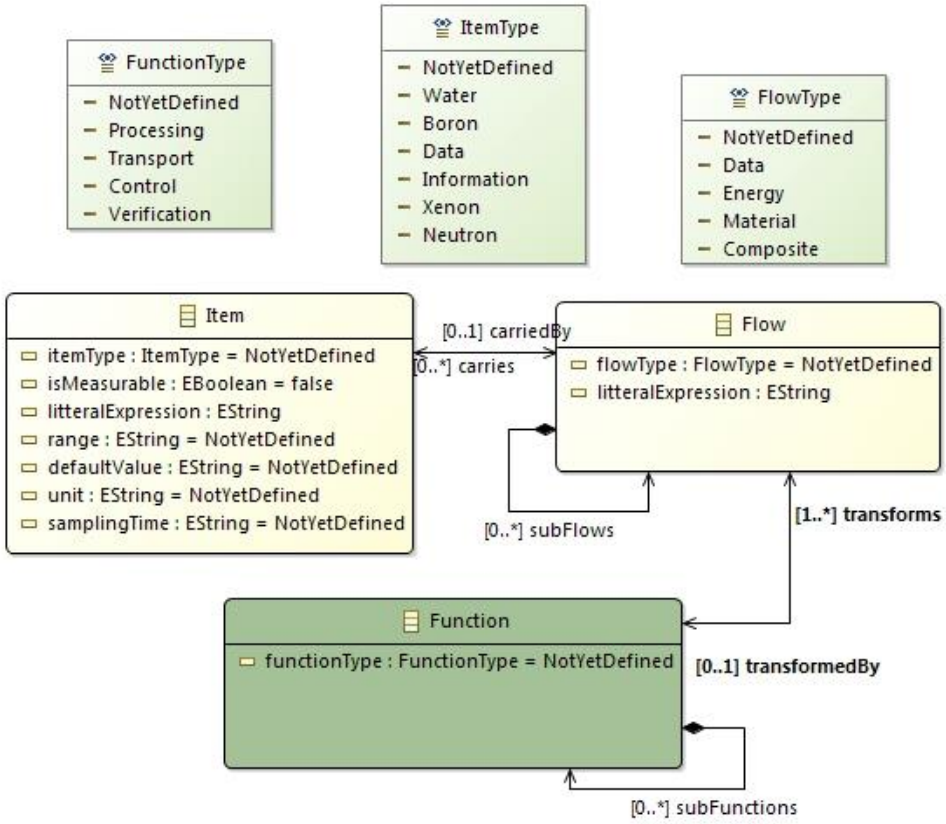


Figure 29: Méta modèle Fonction Flux Item

5.4.2 L'Item

L'Item caractérise les éléments transportés par un flux. Comme montré sur le méta modèle Figure 29, l'item possède un certain nombre d'attributs : (i) l'unité (*Unit*), (ii) le domaine de validité (*Range*), (iii) une valeur par défaut (*DefaultValue*), (iv) un temps d'échantillonnage. Ces attributs et (v) un type (*ItemType*).

5.4.3 Le service

Un service est défini comme : « la fonction, capacité ou comportement fourni par un producteur à un consommateur » [AFIS, 2012]. Usuellement, les services sont utilisés pour exprimer les échanges (plutôt d'un point de vue fonctionnel) entre différents systèmes.

L'extrait du méta modèle Figure 30 montre les relations entre les concepts suivants : le service, le lien et le flux. Les services échangés sont concrétisés par des échanges de flux transportant des items. Enfin, les flux transitent via des liens physiques (e.g. fil, transmissions radio ou tube) qui relient les composants sur lesquels sont allouées les fonctions. Tous ces concepts sont particulièrement utiles pour la définition des interfaces (cf. 5.2.2).

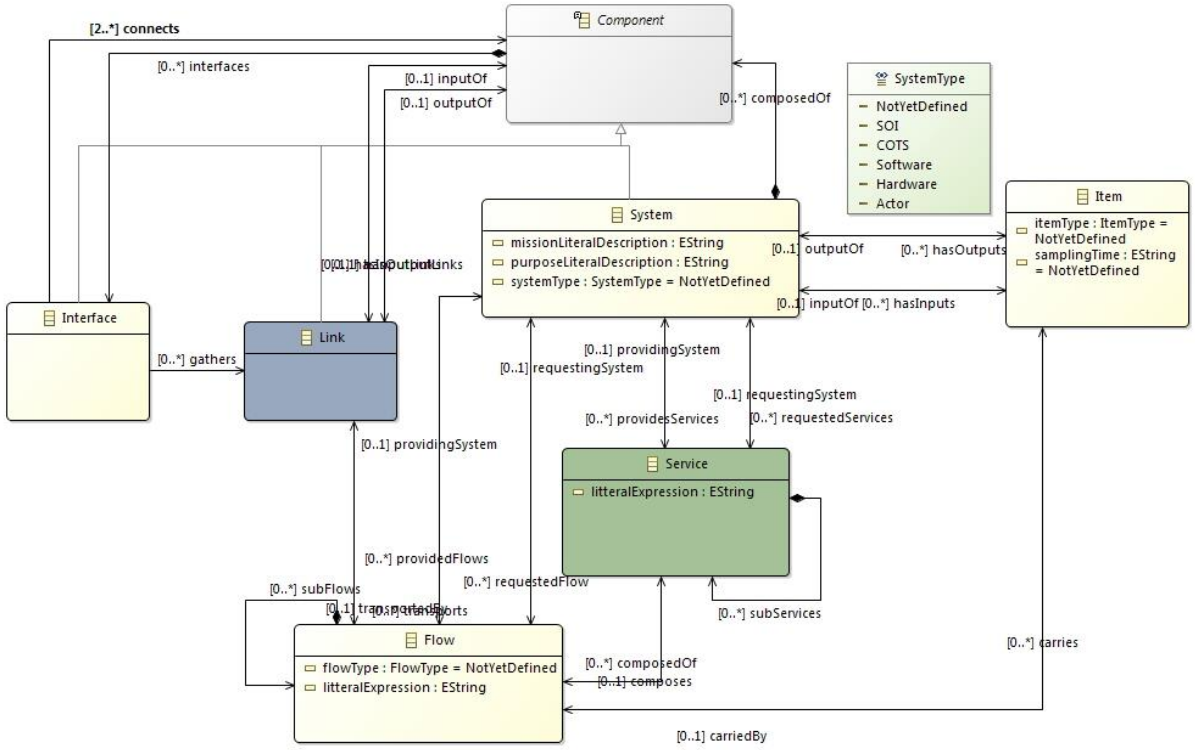


Figure 30: Méta modèle de la vue contextuelle

5.5 Architecture

L'[AFIS, 2012] définit l'architecture d'un système comme : « la représentation, à un niveau de granularité donné, d'un système sous forme d'une structure identifiant les éléments constitutifs du système ». Les éléments constitutifs pouvant être des fonctions ou des éléments physiques. On distingue alors trois types d'architectures : l'architecture fonctionnelle, l'architecture logique et l'architecture physique (aussi appelée ici architecture organique).

- L'architecture fonctionnelle

L'architecture fonctionnelle est une représentation du système par laquelle on le perçoit à travers de ses fonctions mises en interaction sans aucune connotation d'implémentation matérielle (que fait le SOI et non pas comment il le fait). La définition de l'architecture fonctionnelle nécessite, de fait, la décomposition de fonctions en « sous-fonctions » ainsi que la spécification des flux qu'elles échangent.

- L'architecture logique

L'architecture logique met en interaction des composants logiques (cf. 5.2.3). Elle est souvent proche de l'architecture fonctionnelle. En effet, le composant logique est utilisé pour regrouper des fonctions selon une architecture « idéale », des points de vue fonctionnel et opérationnel. Elle se distingue de l'architecture organique (physique) qui elle met en jeu des composants physiques (e.g. matériel), qui est donc à un niveau d'abstraction différent.

Autre différence, les composants logiques sont liés par des liens logiques. Ces liens représentent le regroupement des flux échangés par les fonctions allouées aux composants.

- L'architecture organique (physique)

L'architecture organique (ou physique) fait intervenir des composants organiques (ou physique) et les liens qui les lient. Les composants organiques sont les éléments qui sont réellement conçus. Un tuyau, un disque dur, un calculateur ou un moteur par exemple.

Les architectures fonctionnelle, logique et organique permettent de décrire le système dans des paradigmes différents. Le premier du point de vue de ses fonctions, le deuxième du point de vue des composants logiques. Et enfin du point de vue organique, i.e. les composants physiques. Ces architectures sont complémentaires car elles permettent d'observer le système sous des vues différentes, apportant chacune des détails importants selon les points de vue.

La Figure 31 montre la cohérence entre les paradigmes fonctionnel, logique et physique. Ce diagramme est fait à partir de l'outil Capella et montre l'allocation des différents éléments fonctionnels et logiques du système *Core Control* sur les composants physiques.

Les fonctions sont en vert, les composants logiques en bleu et les composants organiques en jaune (systèmes).

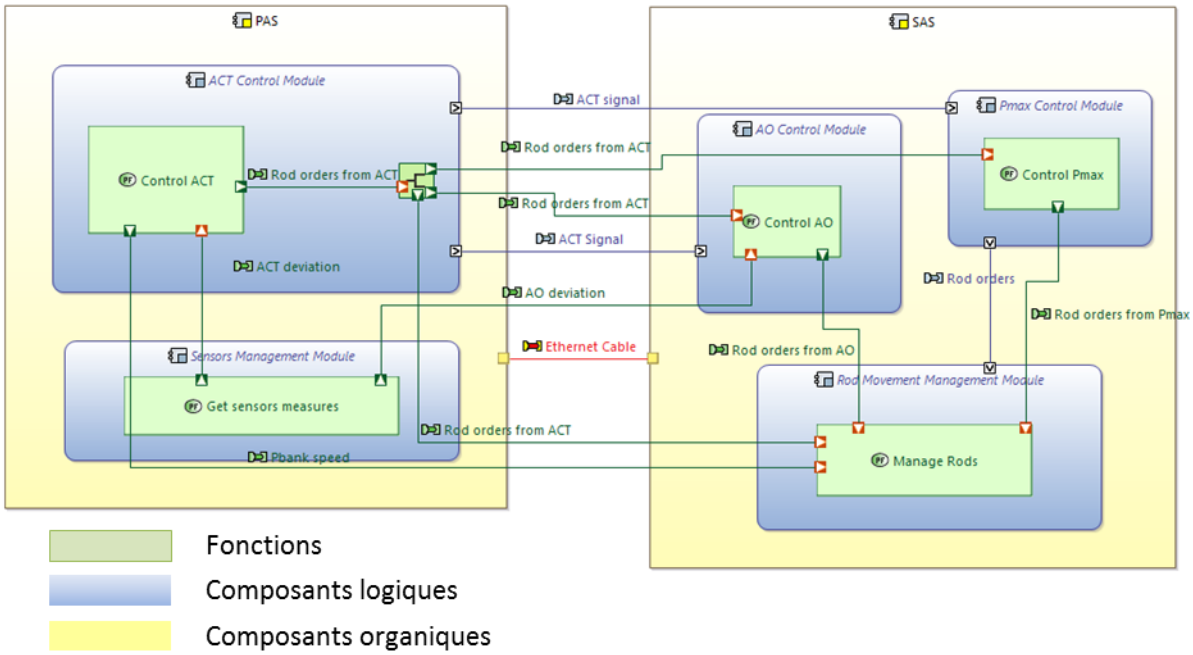


Figure 31: Représentation des liens entre les architectures

5.6 Bloc Système

Le concept de Bloc Système a pour but de réconcilier les différents processus de l'IS avec la décomposition hiérarchique produit PBS adoptée par Framatome, tout en formalisant les différents liens entre chaque niveau système (i.e. nœud du PBS). Un Bloc Système est constitué :

- Du SOI i.e. du système étudié au niveau considéré (e.g. au plus haut niveau du PBS considéré, le SOI est la centrale nucléaire).
- Des différents processus RADIV à appliquer au SOI pour en mener à bien la conception et la réalisation (cf. 2.2).
- Des modèles du SOI issus des activités de modélisation, d'analyse et de choix que les acteurs des phases descendantes et remontantes du RADIV ont pris au cours des activités de ces processus.

Enfin, chaque bloc système est caractérisé par ses entrées et sorties. Le bloc système du SOI prend en entrée les exigences du référentiel d'exigences qui lui est associé. Ce référentiel est un résultat du bloc système de niveau supérieur. Ainsi, chaque référentiel des sous-systèmes est une sortie du bloc système. Le référentiel guide voire contraint la conception du SOI. Le bloc système peut cependant être amené à modifier ce référentiel e.g. pour prendre en compte une contrainte métier particulière pouvant remettre en cause une exigence de plus haut niveau, exigence qui doit alors être répercutée vers le bloc système de niveau supérieur).

Le bloc système du SOI reçoit aussi l'ensemble des sous-systèmes, intégrés, vérifiés et validés qui rentrent dans sa composition, et fournis par les blocs système de niveau inférieurs. Ce bloc système intègre à son tour ses sous-systèmes et met à disposition son système pour le bloc de niveau supérieur.

Comme on peut le voir sur la Figure 32, chaque niveau système communique avec les niveaux du dessous par le partage de référentiels spécifiques d'exigences et de composants et sous-systèmes dûment intégrés, vérifiés et validés.

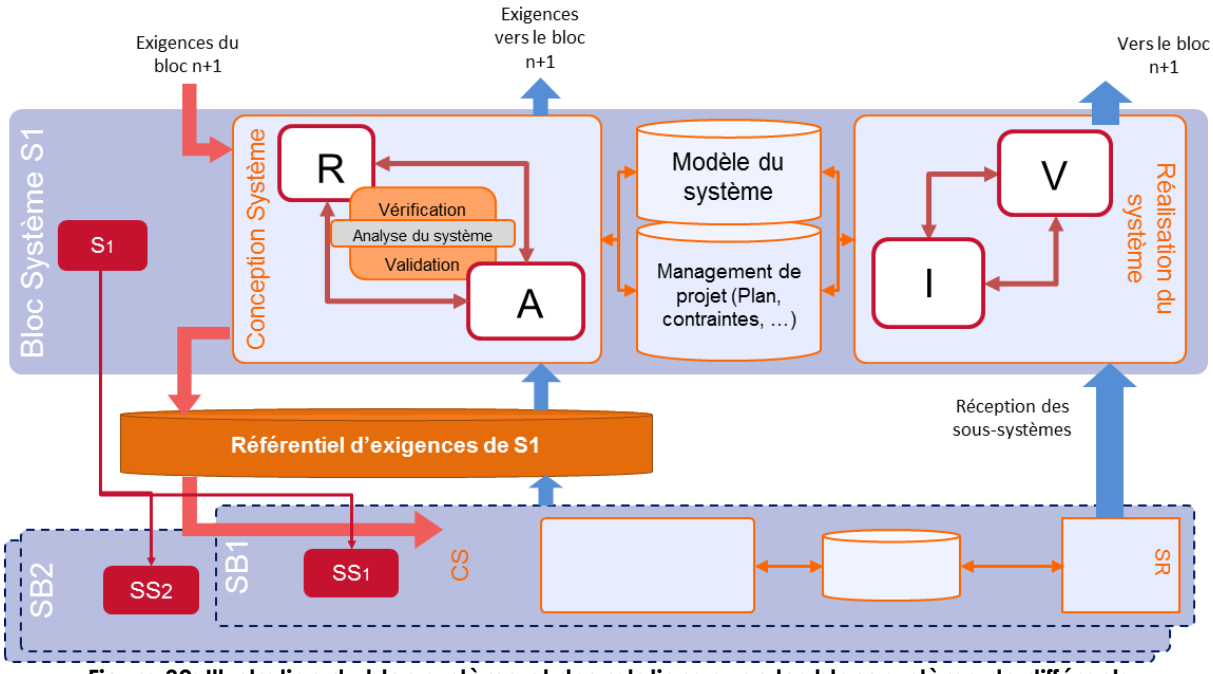


Figure 32: Illustration du bloc système et des relations avec les blocs système de différents niveaux

5.7 Mode Opérationnel et Transition

Un mode opérationnel ou mode de fonctionnement, encore parfois appelé état selon les langages. Il définit une situation, stable ou instable (car éventuellement fugitive) du système, dans laquelle ce dernier possède un comportement bien spécifique.

Les modes opérationnels qu'on retrouve classiquement sont : le mode nominal, le mode automatique et le mode dégradé [Chapurlat et Daclin, 2013].

Certains langages comme *Stateflow* (cf. 2.3.4) autorisent les modes à être actifs en parallèle (i.e. plusieurs modes d'un système peuvent être activés en même temps). Dans notre étude nous ne permettrons pas l'activation en parallèle de mode. En effet, le système est par nature dans un mode de fonctionnement unique et déterministe défini en cours de conception pour des raisons évidentes de sûreté et de sécurité entre autres.

Le mode de fonctionnement est à contraster avec le contexte d'un système. Le contexte représente un état de l'environnement du SOI dans lequel des exigences s'appliquent au SOI. Le contexte intervient donc majoritairement dans le cas de la description boite noire du système. Le mode quant à lui permet de décrire le comportement du SOI en boite blanche.

Le mode possède des relations avec d'autres concepts définis dans la thèse : les **exigences** et les **fonctions** sont **valides dans un ou plusieurs** modes opérationnels.

La transition décrit, quant à elle, la manière dont le système se comporte lors d'un changement de mode. Bien que les comportements dans chaque mode puissent être différents, le comportement global du système doit être vu comme un continuum maitrisé : il ne peut être question d'arrêter un SOI qui doit donc pouvoir transiter de manière cohérente et harmonieuse d'un mode au suivant en tenant compte par exemple des événements ou des décisions de pilotage. Par exemple, le passage du mode automatique au mode manuel pour une voiture autonome pourrait se faire par le contrôle partagé temporaire entre le conducteur et l'algorithme.

La transition est franchie (i.e. on passe du mode source au mode cible de la transition, relations *to* et *from* du méta modèle Figure 33) si les conditions et l'événement de contrôle qui lui sont associés sont vérifiés. Par exemple, si le conducteur a ses mains sur le volant et son attention portée sur la route, le pilote automatique peut passer en mode manuel directement.

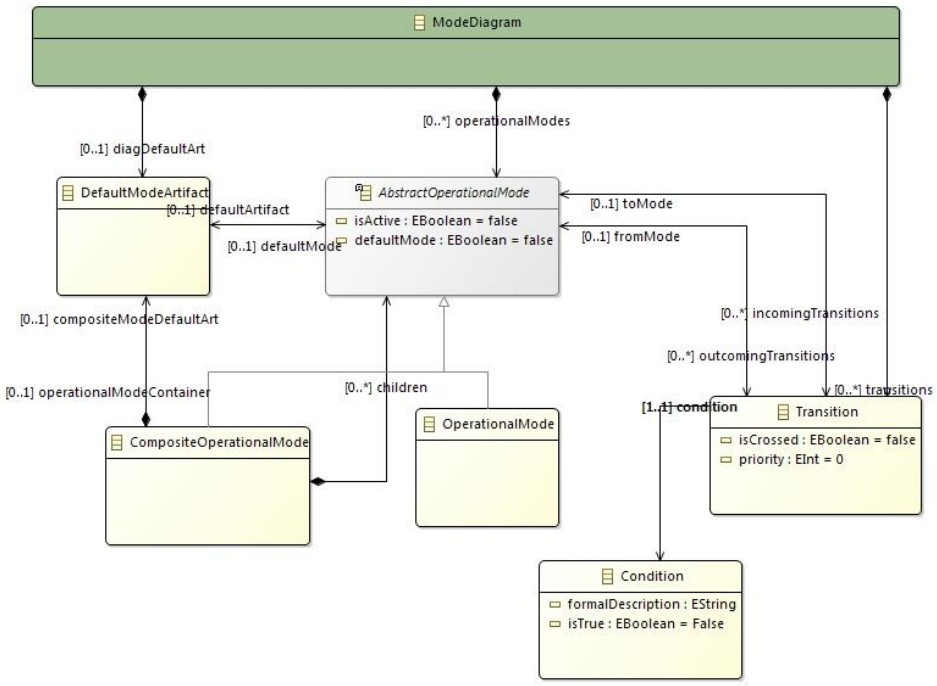


Figure 33: Méta modèle des modes et transitions

5.8 Le scénario opérationnel

Le scénario opérationnel est la description d'une séquence d'événements, correspondant à, ou illustrant une ou plusieurs exigences simultanément (fonctionnelles et/ou non-fonctionnelles) du SOI. Il permet de mettre le SOI en situation et de définir ses interactions avec son environnement et ses utilisateurs. Il permet de même d'irriguer les processus d'IVTV (partie droite du RADIV) en proposant des scénarios de test et d'essai pour assurer la vérification et la validation de l'intégration du SOI.

Le scénario décrit le comportement du SOI. Il aide l'acteur métier à réfléchir et raffiner des exigences du niveau SOI en exigences visant ses sous-systèmes et composants. En particulier par la spécification des interactions et des différents comportements attendus (suite des actions à réaliser).

5.9 Notion de vue

Une vue est une perspective au travers de laquelle on perçoit le système. L'idée générale des vues est de filtrer les éléments de conception du système (fonctions, exigences, composants, etc.) pour faire ressortir uniquement les éléments essentiels (dans cette vue). Par exemple, on peut observer le système au travers de ses fonctions, de ses composants ou encore au travers de ses exigences.

La conception du système consiste à modéliser selon un ensemble de vues, permettant aux différents acteurs métier de définir le système selon leur perspective propre. Dans ce qui suit, nous présenterons les vues que nous avons jugées nécessaires pour l'ingénierie des systèmes de Framatome. On utilise à minima un DSML pour représenter le système dans chacune des vues suivantes :

- La vue Contextuelle.

La vue contextuelle a pour but de décrire la façon dont le SOI interagit avec son environnement et donc d'en préciser les frontières. Le DSML proposé dans la suite est appelé DSML d'architecture. Il possèdera deux syntaxes concrètes dont une permettant de représenter la vue contextuelle que l'on formalisera par le diagramme de contexte. Ce diagramme offre une vision statique (qui ne dépend pas du temps) des services échangés entre le SOI et les différents systèmes qui résultent de la décomposition de son sur-système, ou d'un système de plus haut niveau (voir la structure arborescente du PBS) interagissant avec ce sur-système.

- La vue des Exigences.

Dans cette vue, le système sera observé par le spectre de ses exigences uniquement. Les exigences sont définies et structurées tout en restant conformes au modèle d'exigences, et regroupées dans le référentiel d'exigences associé au SOI (i.e. le système étudié et positionné à un niveau donné du PBS). Cette vue nécessite un DSML de modélisation d'exigences basé sur des Boilerplates (cf. 2.3.1) compte tenu des problématiques de conception de Framatome. Enfin, le référentiel d'exigences est supposé fourni par Framatome qui utilise l'outil Teamcenter (cf. 2.2) pour la gestion de ses exigences.

- La vue Comportementale.

Dans la vue comportementale nous proposons d'utiliser les DSML suivants:

  - Le DSML de séquence (similaire à celui de SysML). Il permet de représenter temporellement les différents services (cf. 5.4.3) entre le SOI en vision boite noire.
  - Le DSML de dépendance. Il définit le comportement entrée/sortie attendu ou encore la variation des entrées pour la simulation. Il permet à la fois de représenter le comportement boite noire et boite blanche du système.
  - Un DSML de Modes et Transitions (M/T). Il permet de spécifier le comportement interne (i.e. boite blanche) du système à travers les modes et les transitions.

- La vue Fonctionnelle.

La vue fonctionnelle permet de spécifier l'architecture fonctionnelle. Cette architecture offre une vision dynamique du SOI puisqu'elles précisent les fonctions et leurs enchaînements dans le temps au moyen de structures de contrôle. Elles évoluent au fur et à mesure de l'avancée du processus de conception du système avec des décisions et choix de composants techniques. Pour cette vue nous proposons un DSML nommé DSML fonctionnel permettant de définir cette architecture. Il est inspiré des diagrammes eFFBD (cf. 6.6).

- La vue logique

La vue logique permet de spécifier l'architecture logique du système. Dans notre démarche nous proposons d'adopter cette vue pour les raisons évoquées dans 5.5. Pour cette vue nous proposons d'utiliser le DSML d'architecture et de créer une syntaxe concrète spécifique à la vue logique (cf. Tableau 4).

- La vue organique

La vue organique est la vue dans laquelle on définit l'architecture organique (physique) du système. Le PBS étant connu à l'avance, nous proposons d'augmenter le diagramme de contexte de la vue contextuelle en y incorporant les sous-systèmes du SOI. L'utilisateur pourra alors définir l'architecture organique du SOI directement sur le même diagramme (cf. Tableau 3).

5.10 Les propriétés

Selon [Chapurlat et Daclin, 2013] « *Une propriété est une caractéristique intrinsèque (comportementale, fonctionnelle, structurelle ou organique, dépendante ou non du temps) d'une entité (un système, un modèle, une entité de modélisation, un phénomène, etc.). Toute propriété traduit une attente, une exigence, une finalité à laquelle l'entité doit répondre* ».

Les propriétés sont par la suite utilisées pour procéder à la vérification, et dans une certaine mesure la validation partielle des modèles établis. L'idée générale est de vérifier à tout instant que l'ensemble des propriétés du modèle, défini par le concepteur, sont bien respectées. Nous distinguons trois types de propriétés :

- **Propriétés système**. Ce sont les propriétés qui traduisent tout ou partie des exigences fonctionnelles, non-fonctionnelles ou des contraintes que l'acteur métier souhaite manipuler, prouver ou évaluer en cours de modélisation. Les modèles du SOI doivent s'y conformer.

- **Propriétés modèle**. Ce sont des propriétés qui expriment les règles de construction à respecter pour chaque modèle (propriétés alors statiques ou atemporelles), les règles de sémantique opérationnelle et les règles de cohérence inter-modèles (statiques et dynamiques car devant être vérifiées tout au long de la vie du système modélisé).
  Les règles de construction sont les règles régissant la structuration des modèles. Par exemple, la propriété suivante est une propriété modèle : « une fonction possède au moins une entrée et une sortie ». Ainsi, si une fonction ne possède pas d'entrée alors le modèle est mal construit.
  Les règles de sémantique opérationnelle peuvent être de deux niveaux. Les règles liées au DSML lui-même et les règles liées à un concept considéré alors isolément. Elles permettent d'interpréter et donc de simuler le comportement d'un modèle conforme à un DSML (e.g. un diagramme eFFBD) ou d'un concept (e.g. une fonction apparaissant éventuellement dans le diagramme eFFBD précédent).
  Les règles inter-modèles sont les règles qui permettent de lier les différents éléments définis par les langages spécifiques, autrement dit les règles exprimées sur la base des relations existantes entre les différents concepts de deux (ou plus) DSML différents (e.g. « un référentiel d'exigences est nécessairement associé à un et un seul SOI » ou encore « les référentiels d'exigences d'un SOI et de l'un de ses sous-systèmes partagent des exigences qui sont le résultat de la mise en œuvre d'une relation de raffinement ou de décomposition »).
  Notons que certaines propriétés modèle de construction comme les contraintes de cardinalités précisées dans la syntaxe abstraite (e.g. 0..N, 0..1, 1..N ou N..M) et les contraintes de totalité et d'exclusion exprimées dans la syntaxe concrète sont ordinairement prises en compte et vérifiées lors de la modélisation. D'autres propriétés modèles nécessitent d'atteindre un certain niveau de maturité des modèles avant que leurs preuves soient possibles comme par exemple les règles de cohérence entre modèles.
  Les propriétés modèle guident donc l'acteur métier pour vérifier ses modèles, le contraindre si besoin (afin d'éviter des erreurs de modélisation) ou encore de l'avertir des conflits ou incohérences qui pourraient survenir (e.g. la fonction F1 n'a pas d'entrée).
  Pour définir de manière formelle et donc pouvoir prouver ce genre de propriété, le langage LUSP (Langage Unifiée de Spécification de Propriétés [Chapurlat 2012] est utilisé. Le lecteur se reportera à cette référence pour connaitre la grammaire de ce langage qui réutilise les concepts et les relations du méta modèle qui est présenté pas à pas dans la suite. Par exemple, la propriété «une fonction possède au moins une entrée et une sortie » se formalisera comme suit :
  $\forall f \in$ Fonction={em = élément de modélisation / classe(em)=Fonction}, [(f.inputParameters() $\neq \varnothing$) and [(f.outputParameters() $\neq \varnothing$)]

- **Les propriétés axiomatiques**. Elles expriment des lois mathématiques, physiques ou d'autres natures caractérisant des concepts ou liant des éléments de modèle. Elles sont considérées comme nécessairement vraies et admises dans un domaine donné. Par exemple, la deuxième loi de newton ($m\vec{a} = \sum \vec{F_i}$) est une propriété axiomatique liant l'accélération d'un système aux forces qui lui sont appliquées.
  Les propriétés axiomatiques sont particulièrement intéressantes pour la vérification des modèles. Premièrement, elles permettent de lier formellement, par l'axiome lui-même, des instances du modèle. Par exemple, l'influence de l'insertion d'une barre sur la réactivité du cœur peut être donnée par une fonction F_1 telle que : $\rho_{coeur} = F_1(P_{rod})$. Ainsi, la fluctuation d'une des variables de modélisation donne des informations de variabilité sur l'autre (par les fonctions F_1 ou F_1^{-1}). Il est alors possible de faire de l'analyse de sensibilité, c'est-à-dire de quantifier la sensibilité paramétrique d'un modèle en connaissant l'ensemble des propriétés axiomatiques de ce dernier. Deuxièmement, si tant est que les sémantiques opérationnelles permettent de les prendre en compte pour, par exemple, détailler le comportement d'un concept, la simulation de ces modèles peut alors devenir plus réaliste et plus fidèle.
  En conclusion, dans la suite, chaque DSML proposé dans ces travaux sera accompagné de la description des propriétés modèles propres à ce DSML. Chaque modèle réalisé au moyen d'un DSML donné devra donc respecter chacune de ces propriétés afin d'être vérifié et, quand la propriété s'y prête, partiellement validé. Les mécanismes de preuve proposés dans (Nastov 2016) seront alors utilisés pour prouver ces propriétés et donc justifier le fait que chaque modèle est a minima 'bien construit', voir est le 'bon modèle'.

5.11 La sémantique opérationnelle d'un DSML

La sémantique opérationnelle d'un langage de modélisation définit comment ce langage doit être interprété, i.e. le comportement des modèles établis et conformes à ce DSML dans le temps. Elle donne vie à ces modèles et est, de fait, une étape nécessaire pour permettre la simulation directe des modèles sans passer par une transformation d'un modèle dans un autre formalisme pour lequel la simulation est possible. Nous proposons dans notre étude d'utiliser un environnement de simulation proposé par [Nastov *et al.*, 2015], [Nastov et al., 2016a] et [Nastov *et al.*, 2016b]. Il est constitué d'un interpréteur de modèles appelé moniteur de DSML et du Blackboard, présentés ci-après.

5.11.1 Le moniteur de DSML

Le moniteur de DSML, encore appelé interpréteur de modèle est un cycle de simulation piloté par l'utilisateur permettant de définir les règles de simulation d'un

langage de modélisation. Il est directement tiré des travaux de [Nastov, 2016]. Il suit un cycle et exécute quatre règles spécifiées pour le DSML à chaque pas de temps précisé par le simulateur. Il se présente de la manière suivante (cf. Figure 34) :

- Initialisation du modèle (IM) : elle permet d'initialiser tous les paramètres du modèle nécessaires à la simulation de celui-ci. Cette initialisation est faite une et seule fois en début de simulation.
- Lecture des entrées (RI) : cette règle permet de lire les entrées du modèle. Elle consiste à récupérer l'ensemble des données dans le blackboard (cf. 5.11.2) pour la simulation au pas de temps actuel.
- Calcul des états futurs (CNS) : cette règle contient l'ensemble des actions à réaliser afin de calculer l'état futur, i.e. l'état interne et les sorties du modèle comme nous le verrons plus loin. C'est durant cette étape du cycle que la sémantique opérationnelle propre au DSML est implémentée.
- Ecriture des sorties (WO) : cette règle a pour objectif d' écrire les sorties du modèle dans le blackboard.

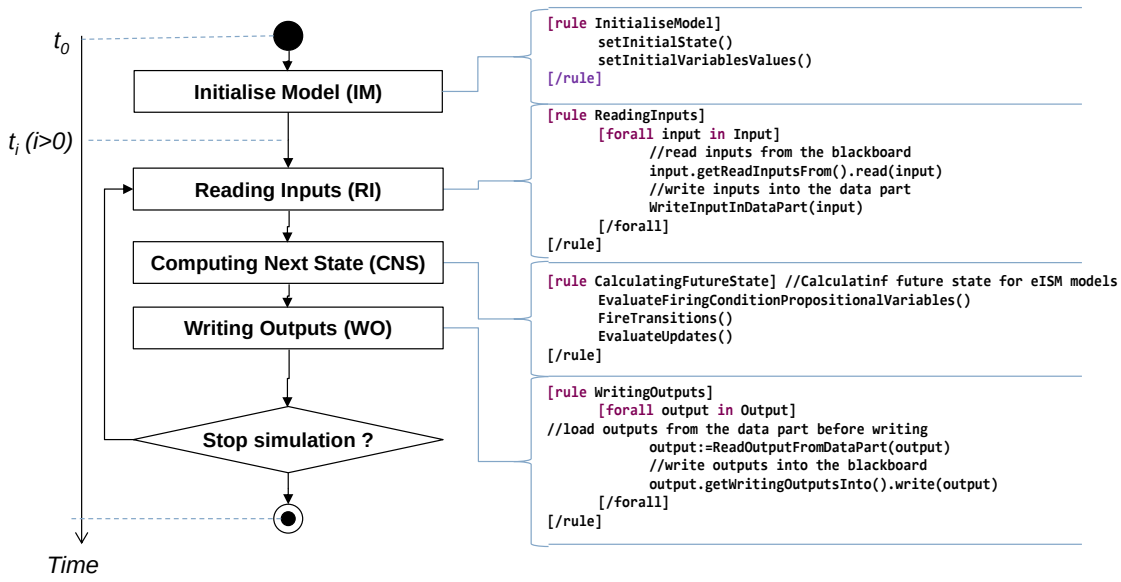


Figure 34 : Moniteur de DSML

5.11.2 Le Blackboard

Le blackboard est un objet informatique utile à la simulation des DSML. Il permet d'écrire ou de lire des données destinées ou provenant d'une instance programmatique i.e. chaque moniteur écrit ses sorties ou lit ses entrées dans le Blackboard qui permet ainsi de distribuer les informations que contiennent ces entrées et sorties entre tous les moniteurs comme présenté sur la Figure 35.

L'utilisation des principes de blackboard et de moniteur de DSML offre un Framework particulièrement intéressant pour la simulation de langages de modélisation. L'idée est la suivante :

- Chaque modèle est interprété par un moniteur qui exécute la sémantique opérationnelle du DSML auquel le modèle est conforme. Le moniteur lit (RI) et écrit (WO) des données sur le blackboard.
- En fonction des entrées, le moniteur applique les règles (CNS) de calcul des états futurs du modèle pour déterminer les sorties.
- Le blackboard contient, gère et met à disposition (en tenant compte d'hypothèses temporelles qui ne seront pas détaillées ici) la totalité des données partagées entre les différents modèles.
- L'ensemble permet finalement une simulation massivement parallèle de plusieurs modèles hétérogènes ne communiquant que via le black board qui assure de fait leur interopérabilité en termes d'entrée/sortie et garantit leur indépendance temporelle par le biais des règles de réconciliation et de synchronisation proposées dans [Nastov, 2016].

5.11.3 La sémantique opérationnelle

Dans notre cas, la sémantique opérationnelle consiste à spécifier, pour chaque DSML ayant vocation à être simulé, i.e. le DSML de Modes et Transitions, le DSML Fonctionnel et le DSML de Spécifications Fonctionnelles, quelles sont :

- Les règles d'initialisation d'un modèle conforme à ce DSML
- Les règles de calcul des états futurs.

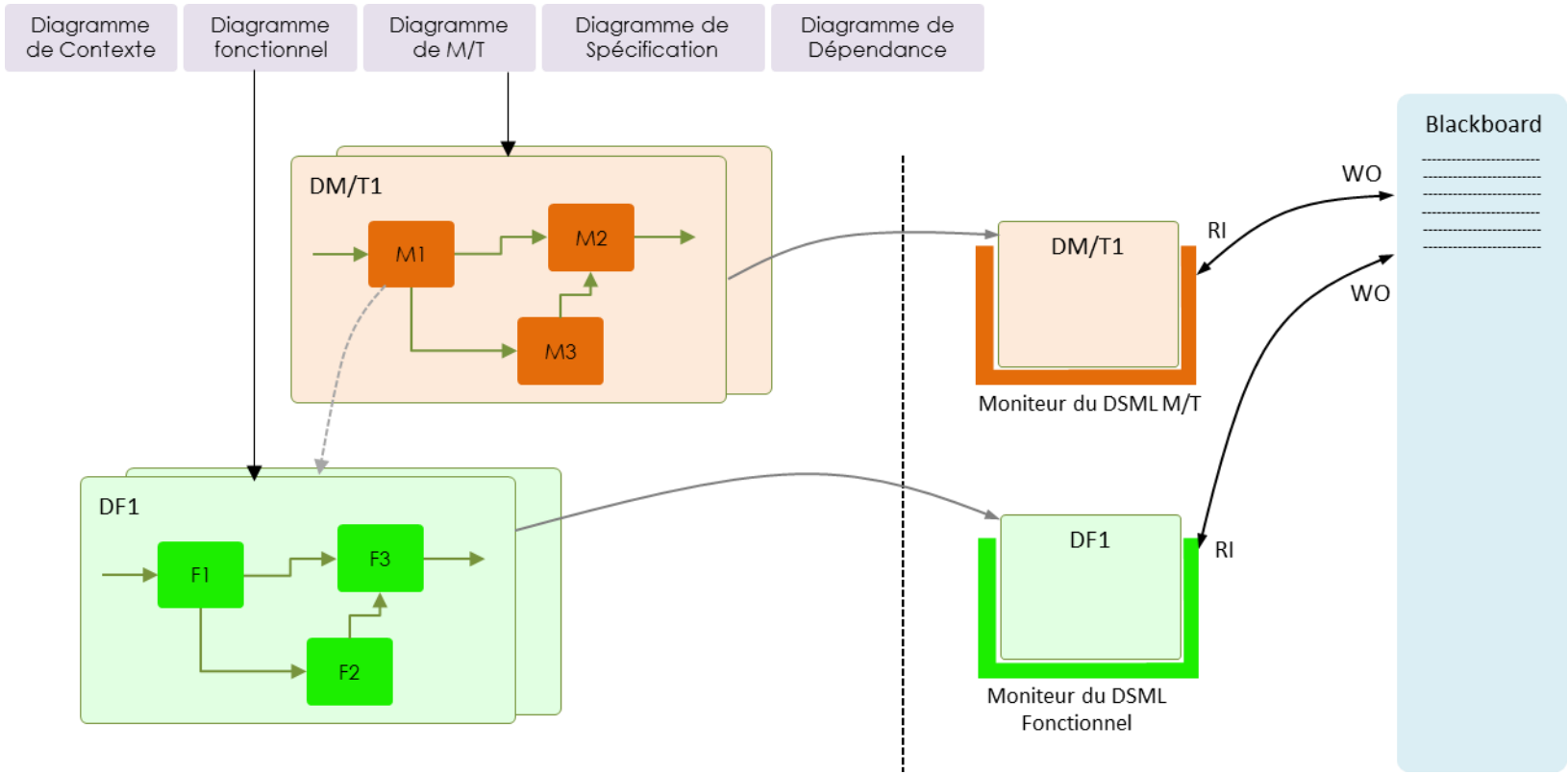


Figure 35: Illustration DSML / Moniteur / Blackboard

6 Les DSML

Sommaire

6.1 DSML d'Architecture 106
6.2 DSML de Modélisation d'Exigences 115
6.3 DSML de Séquences 123
6.4 DSML de Dépendance 127
6.5 DSML de Modes & Transitions 132
6.6 DSML Fonctionnel 138
6.7 DSML de Spécification Fonctionnelle (SF) 146

Ce chapitre a pour but de définir les langages de modélisation (DSML) que nous avons développés ou repris d'autres pratiques existantes puis intégrés dans la méthode (cf. 2.4). Chacun des DSML est présenté sous forme d'une carte d'identité définie comme suit :

- **Nom du DSML** : le nom que nous donnons au DSML.
- **Vue(s) de rattachement** : la (ou les) vue(s) dans laquelle (lesquelles) le DSML est utilisé pour modéliser le SOI pour un objectif propre à chaque acteur métier.
- **Diagramme(s) support du DSML**: le (ou les) diagramme(s) développé(s), supportant le DSML dans la vue de rattachement.
- **Objectif(s) de** :
  - **Modélisation** : Les raisons pour lesquelles ce DSML existe et comment il permet de modéliser.
  - **Vérification** : En quoi le DSML participe à la vérification.
- **Rôle(s) et objectif(s) propre(s) à chaque acteur métier souhaitant utiliser le DSML** : ce que l'utilisateur du DSML doit faire avec le DSML.
- **Place et cohérence dans le cycle en V** : comment il s'insère dans le processus RADIV.
- **Syntaxe abstraite** : les concepts et les relations nécessaires à la définition des DSML et devant de fait être inclus dans le méta modèle unifié sur lequel la méthode repose.
- **Interactions avec les autres DSML** : comment le DSML interagit avec les autres DSML (concepts communs, relations entre deux concepts propres ou attributs pointant sur des concepts propres à des vues différentes).
- **Syntaxe(s) Concrète(s)** : définition, avec d'éventuelle extensions, des objets graphiques (syntaxe graphique) ou textuels (syntaxe textuelle) représentant

les concepts et relations pour permettre leur manipulation lors de la modélisation.

- **Sémantique Opérationnelle** (optionnelle) : ensemble des règles d'interprétation associées au DSML permettant l'exécution (la simulation) des modèles conformes à ce DSML..
- **Propriétés modèle** : les propriétés statiques et dynamiques permettant de s'assurer de la qualité de la construction des modèles (vérification voire validation partielle des modèles conformes au DSML).
- **Modèles de référence** (optionnel) : modèles établis avec ce DSML et qui résultat d'une étude antérieure réussie par exemple, ou qui est vu comme un Patron de modélisation (e.g. un parton de conception architecturale fonctionnelle ou functional design pattern). Ces modèles de référence seront à terme mis à disposition dans le Référentiel des Expertises et Pratiques schématisé Figure 16.
- **Exemple de modèle** : exemple de modèle conforme à ce DSML.
- **Interopérabilité avec le reste des DSML** (optionnel) : pour les DSML qui sont externalisés, i.e. non développés dans notre étude, nous définissons les modalités d'interopérabilité avec les DSML proposés internalisés, i.e. développés dans notre étude.

6.1 DSML d'Architecture

Carte d'identité du DSML

Nom du DSML :	**DSML d'Architecture**
Vues de Rattachement :	**Vue Contextuelle/Logique/Organique**
Diagramme(s) support du DSML Logique	**Diagramme de contexte / Diagramme**

Objectifs

Modélisation

Ce DSML sera utilisé pour les besoins des trois vues contextuelle, logique (au sens de l'architecture logique (cf. 5.5) et organique (physique).

Dans la vue contextuelle, ce DSMLS est utilisé pour décrire les relations que le SOI entretient avec son environnement, c'est à dire ses frontières en termes d'entrées-sorties. L'environnement est composé des systèmes à son (ses) interface(s) dont certains sont de plus contributeurs, des acteurs externes ou de tout autre élément externe qui ne participe pas

directement à son fonctionnement mais qui peut altérer son fonctionnement ou son état. Dans cette vue, l'objectif est donc de faire apparaitre : (i) le SOI, (ii) les éléments de son environnement, et (iii) les interactions entre ces éléments et le SOI. Ces interactions sont modélisées à l'aide du concept de Service (cf. 5.4.3). Nous permettrons de même la définition des flux, des items et des liens qui concrétisent chaque service afin de permettre à l'utilisateur de spécifier complètement les interfaces du SOI.

Dans cette même vue contextuelle, il est proposé ensuite, par décomposition du SOI seul, d'utiliser ce même DSML pour superposer les vues contextuelle et organique (cf. Figure 38). L'idée est de pouvoir représenter les sous-systèmes du SOI et leurs interactions directes directement sur un même diagramme. Les services, flux, items et liens entre les sous-systèmes et/ou entre les sous-systèmes et les systèmes à l'interface du SOI peuvent ainsi être décrits (cf. Figure 38).

De cette manière, ce même DSML est utilisé dans les vues contextuelle et organique. De fait, une seule syntaxe concrète sera nécessaire pour ces deux vues. Nous appellerons ce DSML Diagramme de Contexte (Context Diagram) pour mieux caractériser le rôle de ce DSML d'architecture pour ces deux vues (cf. méta modèle Figure 38 et Tableau 3).

Pour la vue logique, comme dit dans 5.2.3, le paradigme logique est indispensable pour rapprocher des systèmes fonctionnellement proches et pourtant éloignés dans l'architecture PBS (nous invitons le lecteur à se référer à l'exemple fourni dans illustrant bien l'utilité de la vue logique). Pour cette raison, nous proposons d'utiliser ce même DSML d'architecture dans la vue logique afin de décrire l'architecture logique du système. Une syntaxe concrète dédiée à la vue logique sera proposée. Nous appellerons Diagramme Logique (Logical Diagram) cette deuxième variante du DSML d'architecture pour la vue logique (cf. méta modèle Figure 38).

En résumé, nous proposons un seul DSML pouvant représenter les trois vues : contextuelle, organique et logique (cf. Figure 36). Les deux premières bénéficient d'une seule et unique syntaxe concrète décrite dans le Tableau 3 représentée par le Diagramme de Contexte. La troisième bénéficiera de sa propre syntaxe concrète décrite dans le Tableau 4 représentée par le Diagramme Logique.

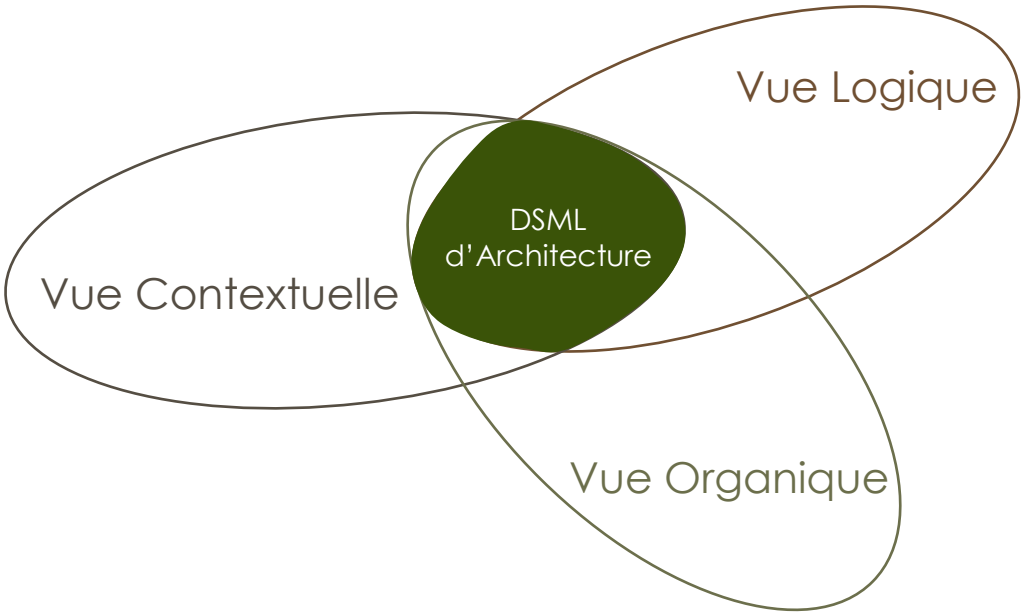


Figure 36: Synthèse des différentes vues adressées par le DSML de contexte

Vérification

Ce DSML fait l'objet d'activité de vérification des modèles par preuve des différentes propriétés modèle définies plus loin qui précisent par exemple des règles de construction de tout modèle élaboré avec ce DSML.

Rôle(s) et objectif(s) propre(s) à chaque acteur métier souhaitant utiliser le DSML

L'utilisateur a pour rôle et objectif de définir l'ensemble des acteurs et systèmes qui peuvent avoir une influence sur le fonctionnement ou l'état du SOI. Certaines interactions sont difficilement prévisibles. Elles peuvent donc être rajoutées à tout moment de la modélisation (ce qui est cohérent avec la notion de conception itérative). Il faudra prendre en compte toutes les phases du cycle de vie du SOI et ne pas oublier de compléter les différentes vues (contextuelle, organique et logique) en conséquence puisqu'elles font respectivement apparaitre la nécessité de certaines interfaces.

Pour cela, l'utilisateur créé un unique diagramme de contexte pour le SOI sur lequel il spécifie chaque service, flux, item et lien qu'il désire. Pour chaque service au moins un flux transporté, contenant au moins un item. Aussi, il fera attention a bien allouer le service à un lien physique provenant du PBS.

L'utilisateur définit l'architecture organique du système sur le même diagramme (recouvrement des vues contextuelle et organique sur le même diagramme). Il définit les sous-systèmes du SOI ainsi que les

services/flux/items/liens entre eux, mais aussi avec les systèmes à l'interface du SOI (cf. Figure 38).

Enfin, comme nous l'avons dit plus haut, l'utilisateur peut passer dans la vue logique en utilisant ce même DSML. Pour cela, il créé un diagramme logique sur lequel il ajoute les composants logiques. Il créé l'architecture en spécifiant de la même manière que pour l'architecture organique, les liens que les composants partagent.

Place et cohérence dans le cycle en V

Le diagramme de contexte touche principalement la phase d'ingénierie des exigences (phase R de RADIV). On considère encore à ce niveau le système comme une boite noire, caractérisé par les entrées et sorties. Les interfaces sont spécifiées compte tenu des exigences d'interface, tout en conservant le point de vue boite noire.

La particularité du diagramme est qu'il permet de définir les sous-systèmes du SOI. On se retrouve alors dans la phase A de RADIV et donc une vision boite blanche du SOI. Nous avons fait ce choix car nous sommes dans une IS basée PBS et les sous-systèmes du SOI sont déjà connus.

Syntaxe abstraite (cf. Figure 37)

Le Diagramme de Contexte.
Concept représentant le DSML dans les vues organique et contextuelle.

Le Diagramme Logique.
Concept représentant le DSML dans la vue logique.

Le Composant.
Concept utilisé pour définir les différents systèmes à l'interface, le SOI, mais aussi les composants physiques, i.e. les sous-systèmes (pour la vue organique) ou encore les composants logiques (pour la vue logique).

Le Service/Flux/Item/Lien.
Le DSML permet de définir les services, les flux et les items échangés entre les différents composants du diagramme mais aussi et les liens entre eux.

La Figure 37 montre que les concepts de diagramme de contexte (*Context Diagram*) ainsi que de diagramme logique (*Logical Diagram*).

Interaction avec les autres DSML

PBS

Premièrement, ce DSML prend en entrée les données du PBS et donc partage le concept commun de Système avec lui (permettant de connaitre le SOI, les systèmes à l'interface et les sous-systèmes pour la spécification des liens).

Référentiel d'exigences

Deuxièmement, les exigences (du DSML de modélisation d'exigences cf. 6.2), contenues dans le référentiel d'exigences associé au SOI (mis au centre du DSML de contexte) provenant du sûr-système ou de parties prenantes (appelées *Upstream Requirements*) sont allouées au SOI. Certaines d'entre-elles spécifient les interfaces et donc seront liées aux instances de Service, Flux, Item et Lien modélisés sur ce diagramme.

Diagramme de modes et transitions

Troisièmement, à partie de ce DSML on associe un et un seul diagramme de mode et transitions (DSML modes). C'est ici un lien fort entre les concepts, qui interviendra par la même occasion dans la démarche opératoire. Enfin, il est en relation avec les flux partagés par les fonctions du DSML fonctionnel (cf. 6.6) et des items partages par le diagramme de spécification (cf. 6.7).

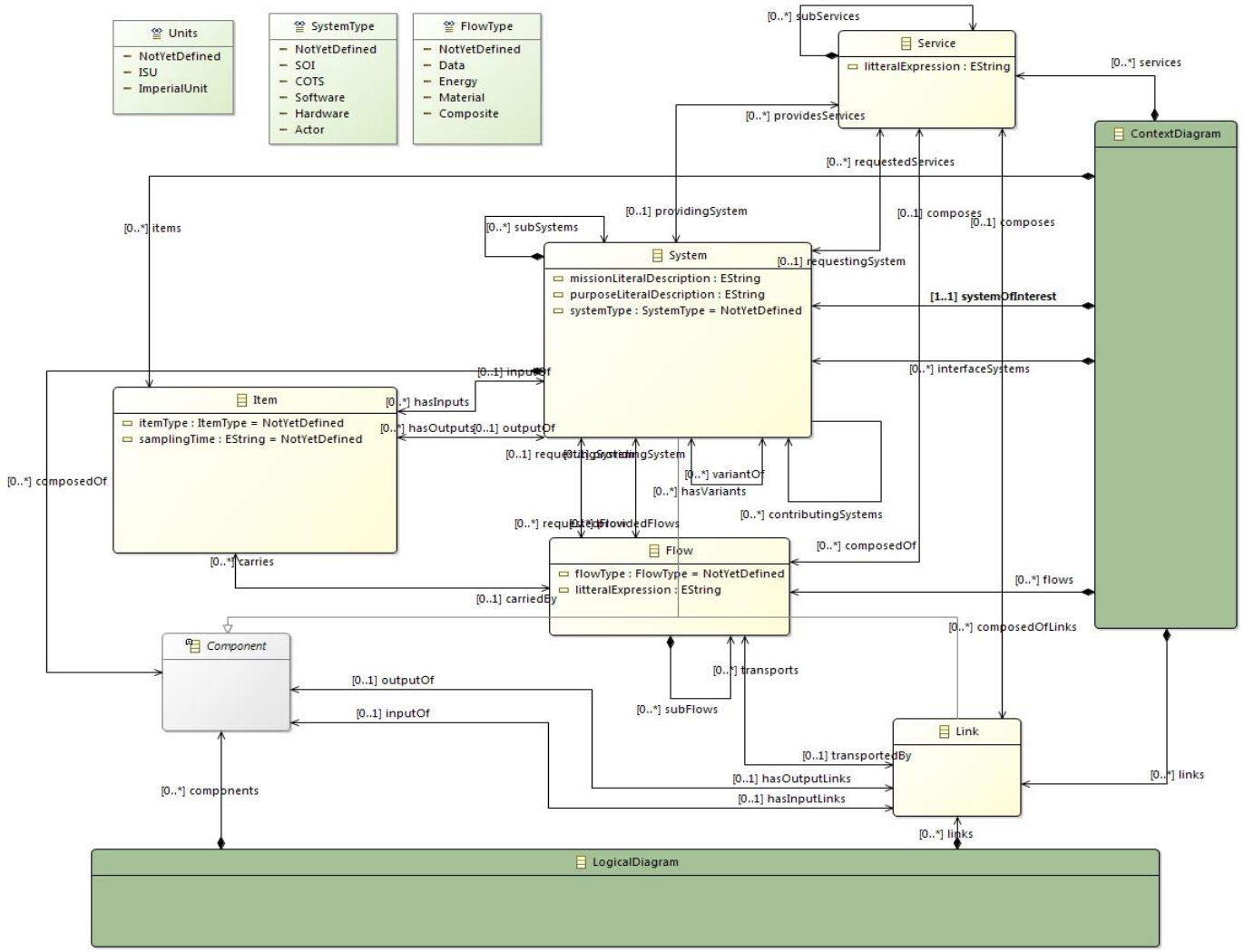


Figure 37: Méta modèle du diagramme de contexte

Syntaxe Concrète du DSML d'Architecture dans les vues contextuelle et organique – Le diagramme de Contexte (Context Diagram)

Eléments graphiques	Description de l'élément graphique
	Liens (composants physique)
	Services échangés entre les systèmes à l'interface et le SOI
	Flux échangés entre les systèmes à l'interface et le SOI
	Items échangés entre les systèmes à l'interface et le SOI
	Liens de composition permettant de représenter : • Les flux qui composent les services • Les items transportés par les flux • Les liens transportent les flux
	SOI
	Sous-systèmes du SOI
	Système à l'interface, contributeurs ou acteurs

Tableau 3: Syntaxe Concrète du DSML d'Architecture dans les vues contextuelle et organique

Une autre fonctionnalité de ce DSML est de fournir la possibilité de spécifier tout au moins en partie les caractéristiques des flux et des items. Si on reprend la définition du concept d'Item (cf. 5.4.2), il est possible de définir un domaine de validité, une unité, un pas d'échantillonnage (tout ceci dans le cas d'Items de type donnée).

Syntaxe Concrète du DSML d'Architecture dans la vue logique – Le diagramme Logique (Logical Diagram)

Eléments graphiques	Description de l'élément graphique
	Liens (composants logique)
	SOI (logique)
	Système à l'interface, contributeurs ou acteurs (logique)

Tableau 4: Syntaxe concrète du DSML d'Architecture dans la vue logique

On peut voir sur le tableau précédent que la syntaxe concrète dans le cas du diagramme logique diffère de celui de contexte.

Sémantique Opérationnelle

Le diagramme de contexte n'est pas voué à faire apparaitre un comportement dynamique du système. Il constitue une perspective statique de ce dernier. Il ne sera donc pas simulé. Une sémantique opérationnelle n'est donc pas nécessaire.

Propriétés modèle

En l'absence d'une sémantique opérationnelle décrivant comment interpréter un modèle conforme à ce DSML, seules quelques-unes des propriétés modèles (cf. 5.10) statiques (i.e. atemporelles) sont précisées dans la suite comme :

- Le SOI possède au moins un service entrant et un service sortant

  $P1 ::= \forall s \in System, [s.type='SOI'] \Rightarrow [(s.providedServices() \neq \emptyset) \text{ and } [(s.requestedServices() \neq \emptyset)]$

  avec $System = \{em = \text{élément de modélisation} / em.class()=System\}$

- Un flux contient au moins un item

  $P2 ::= \forall f \in Flow, [(f.carries() \neq \emptyset)]$

  avec $Flow = \{em = \text{élément de modélisation} / em.class()=Flow\}$

- Un service contient au moins un flux

  $P3 ::= \forall s \in Service, [(s.composedOf() \neq \emptyset)]$

  avec $Service = \{em = \text{élément de modélisation} / em.class()=Service\}$

- Les services/flux/item sont échangés par le SOI et un de ses systèmes à l'interface(SI). i.e. ne peut pas être du SOI vers le SOI ou d'un SI vers un SI. Cette propriété P4 est décomposée en trois sous-propriétés comme suit :

  $P4 ::= \forall s \in System, [s.type='SOI'] \Rightarrow [P4.1(s) \text{ and } P4.2(s) \text{ and } P4.3(s)]$

Avec :

$P4.1(s) ::= (\forall a \in s.providedServices()/a.providingSystem() \neq s) \text{ and } (\forall b \in s.requestedServices()/b.requestingSystem() \neq s)$

$P4.2(s) ::= (\forall a \in s.providedFlows()/a.providingSystem() \neq s) \text{ and } (\forall b \in s.requestedFlows()/b.requestingSystem() \neq s)$

$P4.3 ::= (\forall a \in s.hasInputs()/a.inputOf() \neq s) \text{ and } (\forall b \in s.hasOutputs()/b.outputOf() \neq s)$

Exemple de modèle

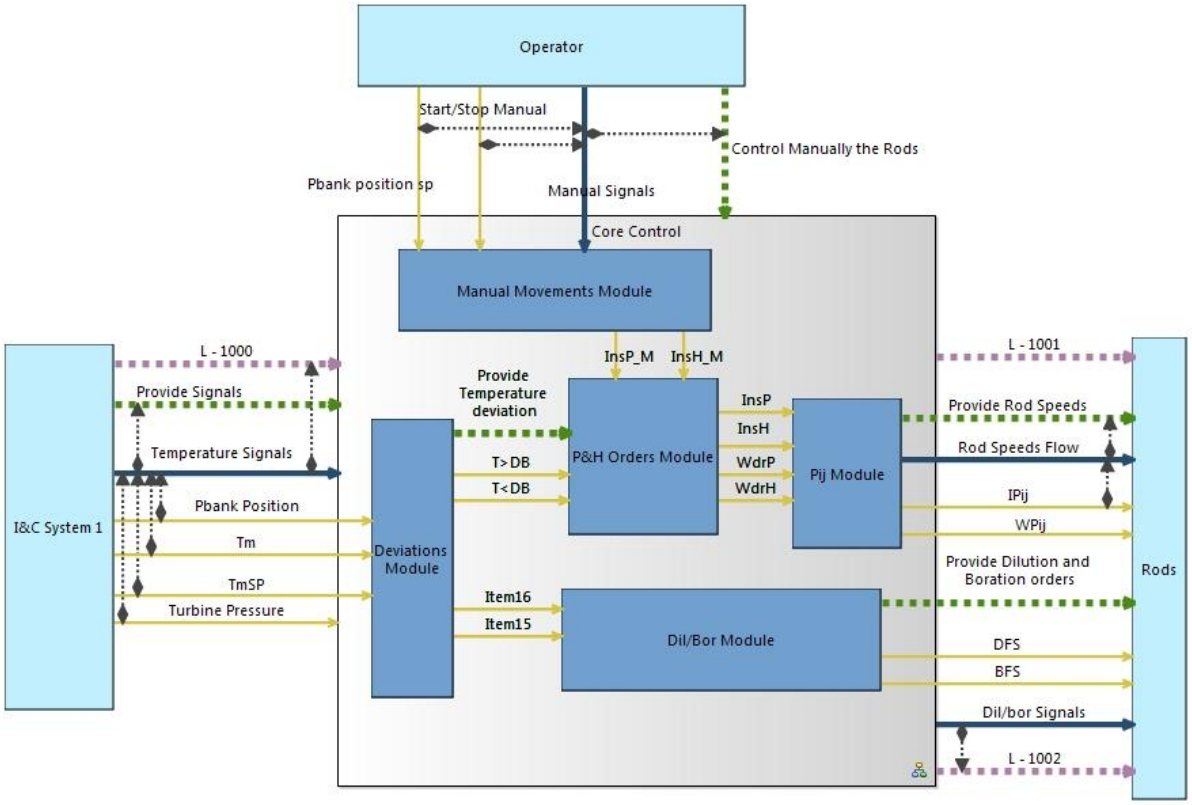


Figure 38: Exemple de diagramme de contexte

6.2 DSML de Modélisation d'Exigences

Carte d'identité du DSML

Nom du DSML : **DSML de Modélisation d'Exigences**

Vue de Rattachement : **Vue des exigences**

Diagramme(s) support du DSML **Pas de diagramme associé**

Objectifs

Modélisation

Nous avons pour but de définir puis, de vérifier le référentiel des exigences de chaque niveau système (nœud du PBS). Comme annoncé dans le chapitre 1, Framatome utilise Teamcenter pour la gestion des exigences. Notre proposition ici est double :

- Premièrement, nous proposons d'utiliser le modèle d'exigence présenté en 5.3.1 contenant les attributs et les relations des exigences et permettant de compléter la vision actuelle de Framatome en particulier en matière de traçabilité. En effet, comme nous l'avons dit dans 5.3.6 et 5.3.7, les exigences sont liées entre elles pas des relations inter-exigences (raffinement, dérivation, reformulation et décomposition) ainsi qu'à des éléments de modélisation (modes / scénarios / fonctions/système, cf. 5.3.7). L'utilisation de ces relations permet d'accroitre les liens de dépendance entre les éléments de modélisation, permettant de faire de l'analyse d'impact (cf. 5.3.3). Cependant, nous ne pouvons que proposer cette contribution (analyse d'impact) car nous n'avons pas la maitrise de l'ensemble des exigences gérées dans Teamcenter. Elle ne sera donc pas plus développée dans la thèse.
- Deuxièmement, nous proposons un langage d'écriture des exigences contraint et structuré, basé sur le langage naturel et s'appuyant sur le formalisme des boilerplates défini dans 2.3.1. Ce langage aide l'auteur de l'exigence dans sa rédaction en lui proposant des patrons textuels pré-remplis que nous décrirons plus loin dans cette partie.

Vérification

Ce DSML permet de structurer, guider et contraindre l'écriture des exigences. Le but est d'utiliser un langage naturel, donc compréhensible pour tous les acteurs métier de la conception, tout en restreignant le langage, ne laissant à l'auteur que le juste nécessaire pour concevoir.

C'est l'idée principale de la **modélisation guidée**, une des quatre méthodes de vérification des modèles (appliquée ici aux exigences) définies dans (cf. 2.1).

Ensuite, ce DSML fait lui aussi l'objet de vérifications de propriétés modèles décrites plus loin pour en attester la qualité de la construction. Il participe enfin à la vérification, voire la validation partielle de la cohérence et de la pertinence des choix des acteurs métiers pour définir le SOI via la traçabilité des exigences vers tous les éléments de modélisation utilisés pour décrire ces choix et le SOI résultant.

Rôle(s) et objectif(s) propre(s) à chaque acteur métier souhaitant utiliser le DSML

L'utilisateur créé ou modifie les exigences ensuite référencées et gérées dans le référentiel d'exigences tout au long du processus descendant de RADIV. L'utilisateur raffine/décompose/dérive ou reformule les exigences en de nouvelles exigences ou bien édite les liens entre des exigences existantes non reliées. Enfin, l'utilisateur édite les relations permettant la traçabilité des exigences (e.g. relations d'allocation aux fonctions ou aux composants et sous-systèmes du SOI).

Place et cohérence dans le cycle en V

Ce DSML couvre principalement la phase d'ingénierie des exigences. Selon le cycle RADIV, à chaque niveau du PBS, les exigences venant du sur-système sont regroupées dans le référentiel d'exigences du SOI. Durant cette phase, l'utilisateur considère le SOI avec une vision boite noire et travaille donc à définir le problème et non la solution fonctionnelle ou organique (technologique) que représente le SOI.

Ces exigences sont raffinées, dérivées, décomposées ou reformulées en d'autres exigences. Certaines d'entre elles concernent et sont allouées aux sous-systèmes du SOI. Ainsi, ce DSML est aussi utilisé pour définir le système en boite blanche.

Syntaxe abstraite (cf. Figure 39)

Les concepts et les relations nécessaires pour la définition de ce DSML sont :

Le référentiel d'exigences
A chaque SOI est associé un unique référentiel d'exigence. Le référentiel d'exigences est composé de l'ensemble des exigences associées au SOI.

L'exigence

Outre le modèle d'exigence présenté plus haut, les relations inter-exigences sont nécessaires (décomposition, reformulation, raffinement et dérivation). Les autres relations existantes entre les exigences et d'autres éléments de modélisation (modes, systèmes, etc.) seront introduites dans la suite au cas par cas. Chaque exigence pourra être reliée à un élément du modèle par ces relations.

Système/Mode/Fonction/Scénario opérationnel

Les exigences sont liées aux différents éléments de modélisation suivants : Système/Mode/Fonction/Scénario opérationnel tel que décrit dans 5.3.7.

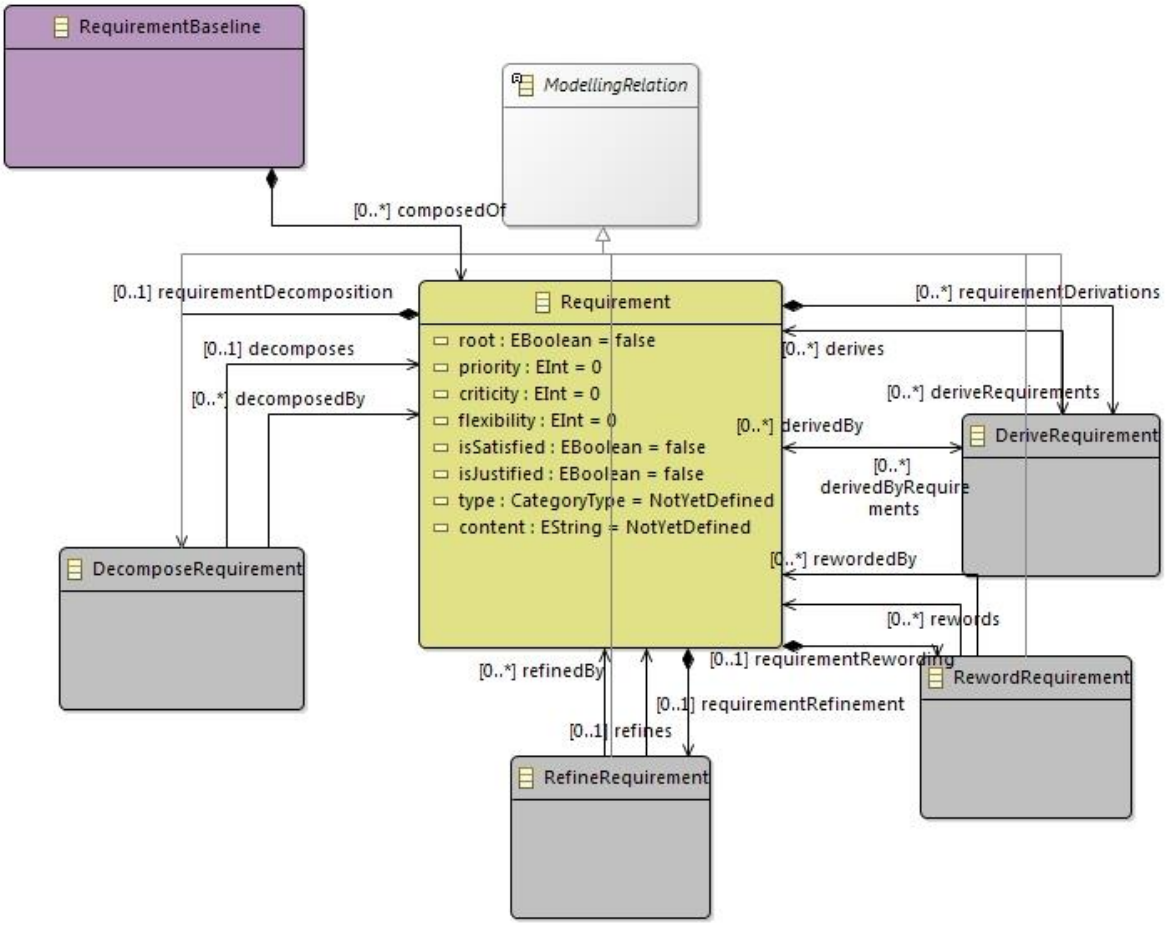


Figure 39: Méta modèle pour le DSML d'exigences

Interaction avec les autres DSML

PBS

Un référentiel d'exigences est associé à chaque système du PBS. Les exigences contenues dans ce référentiel sont éditées par ce DSML.

Diagramme de Contexte
Les exigences (non-fonctionnelles, de V&V et contraintes) sont allouées aux composants et sous-système qui sont définis dans le diagramme de contexte.

Diagramme de Modes et Transitions
Les exigences doivent être respectées dans un ou plusieurs modes.

Diagramme de scénarios opérationnels
Les exigences sont raffinées ou caractérisent les scénarios (cf. relations exigences éléments de modèle 5.3.7).

Diagramme Fonctionnel
Les exigences (fonctionnelles et de performance) sont allouées aux fonctions qui sont définies dans le diagramme fonctionnel.

Diagramme de Spécification fonctionnelle
Certaines exigences peuvent spécifier le comportement des fonctions (exigences fonctionnelles et contraintes), définissant les spécifications fonctionnelles.

Syntaxe Concrète

Dans cette partie, nous allons définir la syntaxe concrète textuelle des boilerplates. Ils sont constitués du *Main* (partie centrale de l'exigence), du *Prefix* (partie introductive de l'exigence) et enfin du *Suffix* (partie terminale de l'exigence).

Le *Main*
Le *main* est la partie principale de l'exigence. Il a pour but de préciser ce qu'un système ou une variable de modélisation doit faire ou être. Par exemple, « The car shall be blue » ou encore « The temperature T shall be strictly below 20 °C ».

Selon les attentes de Framatome l'exigence doit contenir l'élément *shall* dans sa description. La seule partie fixe du *main* sera alors l'élément *shall*. Nous donnons quatre boilerplates à titre d'exemple ici (l'ensemble des *main* considérés sont précisés dans l'Annexe B) :

…<system> shall <action>… (7.1)

…<system> shall <state>… (7.2)

…variable <variable> shall <action>… (7.3)

… variable <variable> shall <state>… (7.4)

L'action (*action*) peut soit n'avoir aucune structure particulière, afin de ne pas trop contraindre l'écriture d'une exigence, soit être structurée par des boilerplates particuliers. Par exemple pour définir une interface, l'action peut être définie comme dans le boilerplate en (5). Ainsi, on peut écrire l'exigence suivante : « S1 shall provide S2 with ACT signal ».

Action : provide <system> with <variable> (7.5)

L'état (*state*) cependant peut être structuré. Dans le cas du boilerplate (2), i.e. pour un système, l'état peut décrire un mode (e.g. « S1 shall be in Automatic mode ») ou un état physique (« the screw shall be blue »). Dans le cas d'une variable, i.e. pour le boilerplate (4), l'état définit des valeurs (limites, domaine, etc.) que la variable doit respecter (e.g. « the temperature T shall belong to [20,30]°C ».

Le *prefix*

Le *prefix* sert à fixer le contexte dans lequel l'exigence va être valable. Il se place avant le *main*. Les boilerplates qui le composent sont tels que décrits dans les boilerplates (8.6) à (8.8) (l'ensemble des *prefix* considérés sont précisés dans l'Annexe B). Ils utilisent des parties fixes (*if* ou *while*) permettant de décrire les conditions d'application de l'exigence. Par exemple, si un événement se produit comme « If the wheels lock » ou encore si un système ou une variable est dans un état particulier, e.g. « If S1 is in Manual mode ».

If <event>, … (7.6)

If <state>, … (7.7)

While <state>, … (7.8)

Les boilerplates qui le composent sont présentés dans l'Annexe B.

Notons qu'un *prefix* peut être constitué de plusieurs *suffix*, i.e. que l'on peut les concaténer pour former un *suffix* donnant plus de détail. Par exemple :

If <event>, If <event>, while <event>, … (7.9)

Ceci permet d'ajouter du détail sur les conditions opérationnelles de l'exigence. Par exemple : « *If <event>, If <event>, While <event>* …».

Le *suffix*

Enfin le suffixe permet d'apporter des détails sur l'exigence. Deux cas sont à distinguer :

- Pour décrire que l'exigence est réalisée après ou avant un événement quelconque. Il est défini par les parties fixes *before*, ou *everytime*. Par exemple, « The car shall stop **after an accident happens** » ou encore « The waiter shall set the table everytime clients leave the table ». Ainsi, le *suffix* « after an accident happens » permet bien d'apporter du détail sur l'exigence.

- Pour d'apporter des précisions quantitatives sur l'application de l'exigence. Ces boilerplates sont définis par les parties fixes : *every*, *for a period of at least*, *for at least*, for, *within, during* et *of*. Toutes ces parties fixes sont suivies d'une partie mutable de la forme : <quantity> <unit>, définissant une quantité et une unité associée. Notons que l'ensemble des *suffix* considérés sont précisés dans l'Annexe B.

… within <quantity> <unit> (7.10)

… after <event> (7.11)

… from <component> (7.12)

On peut alors écrire par exemple « the protection system shall trigger an emergency stop **within 10 seconds** ». Il est possible, comme pour le *prefix* de concaténer les *suffix*. Par exemple :

… within <quantity> <unit>, after <event> (7.13)

Donnons un exemple d'exigence rédigée grâce aux boilerplates :

« The protection system **shall** trigger an emergency stop **within** 10 seconds **after** DNB crisis »

Dans les trois parties de l'exigence, on remarque que les parties mutables sont des éléments sémantiques. On distingue l'événement, l'état, la quantité, l'unité, la variable de modélisation ou encore le composant.

Ces concepts sont définis en Annexe B. Ils permettent d'indiquer à l'utilisateur le type de l'élément à préciser dans la partie mutable. Par exemple si l'utilisateur utilise le *suffix* : « **after** <event> ». L'élément mutable « <event> » indique à l'utilisateur qu'il devra préciser un évènement.

Sémantique Opérationnelle

Le DSML de modélisation d'exigence n'est pas voué à être simulé car il constitue une vision statique, une photo prise à l'instant où on l'observe, du système à travers la description de ses exigences. Nous ne définirons donc pas de sémantique opérationnelle.

Propriétés modèle

Comme pour le DSML précédent, les propriétés modèles statiques données à titre d'exemple ci-dessous sont des propriétés statiques de construction :

- Une exigence fonctionnelle ne peut pas être décomposée en exigence non-fonctionnelle ou de performance

  $P1 ::= \forall r \in Requirement, [r.type='Functional'] \Rightarrow [(r.refinedBy() = \emptyset)$ or $(\forall f \in r.refinedBy(), f.type='Functional')]$

  avec $Requirement = \{em = élément\ de\ modélisation / em.class() = Requirement\}$

- Une exigence de performance ne peut pas être décomposée en exigence non fonctionnelle ni en exigence fonctionnelle

  $P2 ::= \forall r \in Requirement, [r.type='Performance'] \Rightarrow [(r.refinedBy() = \emptyset)$ or $(\forall f \in r.refinedBy(), f.type='Performance')]$

- Les exigences feuilles doivent nécessairement être soit allouées :

  $P3 ::= \forall r \in Requirement, [P3.1$ or $P3.2$ or $P3.3]$

  - Aux fonctions dans le cas d'une exigence fonctionnelle ou de performance

  $P3.1(r) ::= [(r.refinedBy() = \emptyset] \Rightarrow [\ r.allocatedToFunctions() \neq \emptyset)]$

  - Aux composants dans le cas d'une exigence non-fonctionnelle

  $P3.2(r) ::= [(r.refinedBy() = \emptyset] \Rightarrow [\ r.allocatedToFunctions() \neq \emptyset)]$

  - Soit liée (dérivation, raffinement, rewording ou décomposition) à d'autres exigences.

  $P3.3(r) ::= [(r.refinedBy() \neq \emptyset] \Rightarrow [\forall f \in r.refinedBy(), P2.3(f)]$

– Une exigence non-fonctionnelle ne peut pas être décomposée en exigence fonctionnelle

$$P4 ::= \forall r \in \text{Requirement}, [r.type='NonFunctional'] \Rightarrow [(r.refinedBy() = \varnothing) \text{ or } (\forall f \in r.refinedBy(), f.type \neq 'Functional')]$$

Exemple de modèle

Les exigences données ci-après ont été rédigées à partir de l'outil Xtext développé lors de la thèse dont une capture d'écran est donnée en Figure 40.

```
The Core Control shall perform turbine ramp variations of 5 %NP/min;
The Core Control shall perform turbine power step variations of 10 %NP;
The variable ACT_Deviation shall remain within [-1.5 , 1.5]°C;
The variable AO_Deviation shall remain within [-5 , 5] %AO;
The variable Pbank_Deviation shall remain within [-30 , 30] steps;
```

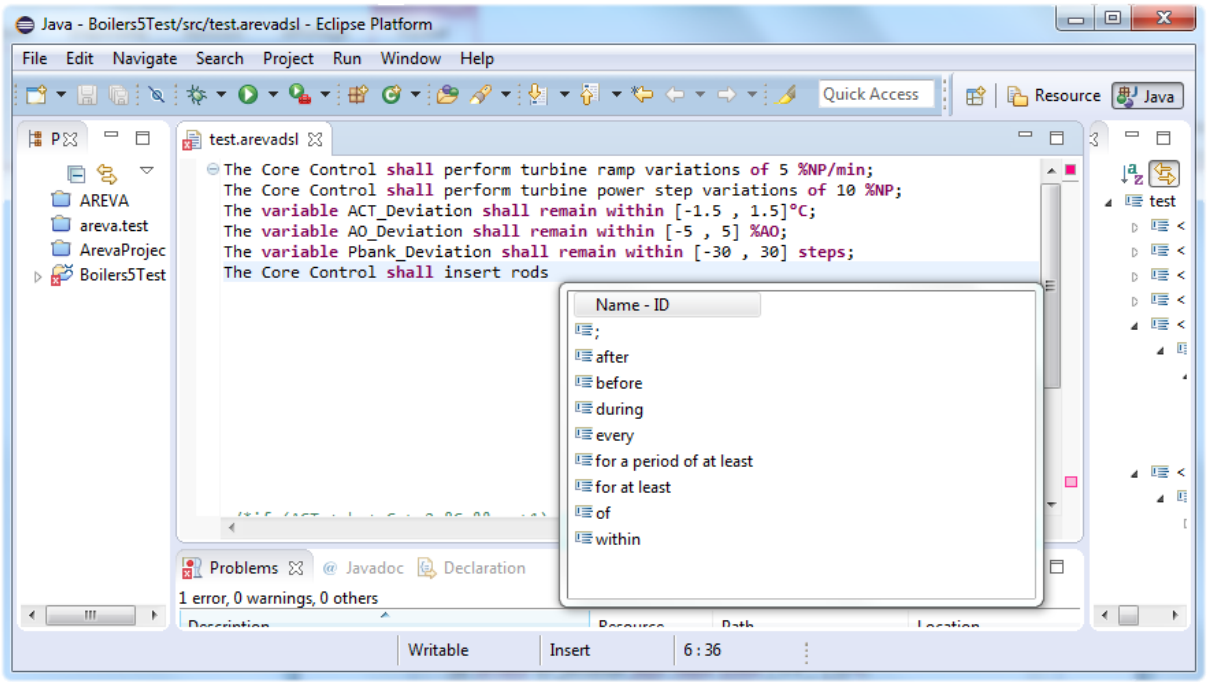


Figure 40: Capture d'écran de l'outil l'outil développé pour l'Ingénierie des exigences (avec Xtext)

6.3 DSML de Séquences

Carte d'identité du DSML

Nom du DSML : **DSML de séquences**

Vue de Rattachement : **Vue comportementale**

Diagramme(s) support du DSML **Diagramme de séquence**

Objectifs

Modélisation
Le DSML de séquences que nous proposons a pour objectifs de définir les scénarios opérationnels du SOI en « boite noire », i.e. comment le SOI se comporte dans le temps avec son environnement. On le dit « boite noire » car il fait abstraction du comportement interne du SOI. Ce DSML permet de définir les interactions, i.e. les échanges de flux correspondant aux services que ce soit pour un comportement attendu (un comportement prévisible) ou pour un comportement redouté (comportements potentiels non-souhaités, souvent relatifs à une situation accidentelle ou incidentelle).

Ce DSML a pour objectif de représenter comment sont orchestrés dans le temps les différents services rendus (sortant) ou requis (entrants) par le SOI, eux-mêmes définis dans le diagramme de contexte.

Vérification
Comme pour le DSML des exigences, nous préconisons ici une démarche de **modélisation guidée** avec quelques-unes des propriétés modèles données en exemple dans la suite.

Rôle(s) et objectif(s) propre(s) à chaque acteur métier souhaitant utiliser le DSML

L'utilisateur de ce DSML a pour objectif de décrire les scénarios i.e. comment s'enchaînent les interactions entre le SOI et son environnement dans le temps (ses systèmes à l'interface). Il peut de fait être l'utiliser pour détailler certaines exigences (relation de raffinement) pour éviter d'avoir alors à fournir une représentation chronologique de scénario parfois difficilement compréhensible dans sa forme textuelle. De fait, l'acteur métier peut représenter autant de scénarios qu'il le désire, la cohérence

et l'absence d'ambiguïtés comportementales de l'ensemble de ces scénarios devant ensuite être vérifiées.

Place et cohérence dans le cycle en V

Ce DSML interviendra quasiment dans la première phase de définition du système (Requirement & Capture Analysis dans RADIV). Il raffine des exigences et permet de préciser le contexte opérationnel du SOI.

Syntaxe abstraite

NB : nous ne donnerons pas ici la syntaxe abstraite (méta modèle) de ce DSML puisque nous reprendrons directement la représentation donnée par Capella du diagramme de séquences.

Système
Le SOI (instance du concept de système) doit à minima être présent dans ce DMSL, ainsi que les systèmes à l'interface (les scénarios décrivant leurs interactions).

Service
Les systèmes échangent des services.

Flux
Les systèmes échangent des flux.

Exigence
Les scénarios opérationnels qui résultent de l'utilisation du DSML peuvent raffiner des exigences.

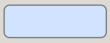
Diagramme de séquences
Le diagramme de séquences est le concept support du DSML de séquence. Il est en relation avec les autres concepts constituant le DSML (Système, Service, Flux, etc.) et permet de créer des instances du DSML lors de la modélisation.

Interaction avec les autres DSML

DSML de modélisation d'exigences
Un diagramme de séquences peut raffiner une exigence

Syntaxe Concrète

La syntaxe concrète que nous utiliserons est celle définie par Capella [Roques, 2016].

Eléments graphiques	Description de l'élément graphique
	Fonction / Service
	SOI
	Système à l'interface
	Ligne de temps

Sémantique Opérationnelle

Dans notre étude, le DSML de séquence ne sera pas simulé. Tout comme les DSML précédemment présentés, il constitue une vision statique du système. Pour cette raison nous ne définissons pas de sémantique opérationnelle.

Propriétés modèle

Comme pour les précédents DSML, il s'agit ici de propriétés modèles statiques contraignant la construction du diagramme de séquence. Ce sont par exemple des propriétés comme celles qui suivent :

- Les services et fonctions définies dans le diagramme de séquence sont les fonctions du SOI ou des systèmes à l'interface, ou encore les services échangés par le SOI et les systèmes à l'interface

  $P1 ::= \forall ds \in DSequence, [P3.1(ds) \text{ and } P1.2(ds)]$

  Avec,

  $P1.2(ds) ::= [P3.2 \text{ or } (\forall s \in ds.systems(), \forall f \in ds.functions() [(s.fonction() \supset f]]$

  avec $DSequence = \{ds = \text{élément de modélisation} / ds.class() = SequenceDiagram\}$

- L'ensemble des composants du diagramme de séquence est inclus dans l'ensemble des éléments du diagramme de contexte

  $P2 ::= \forall ds \in DSequence, [\forall dc \in DContexte, (dc.soi() = ds.soi()] \Rightarrow [\forall c \in ds.components(), dc.components() \supset c]$

  Avec $DContexte = \{dc = \text{élément de modélisation} / dc.class() = ContextDiagram\}$

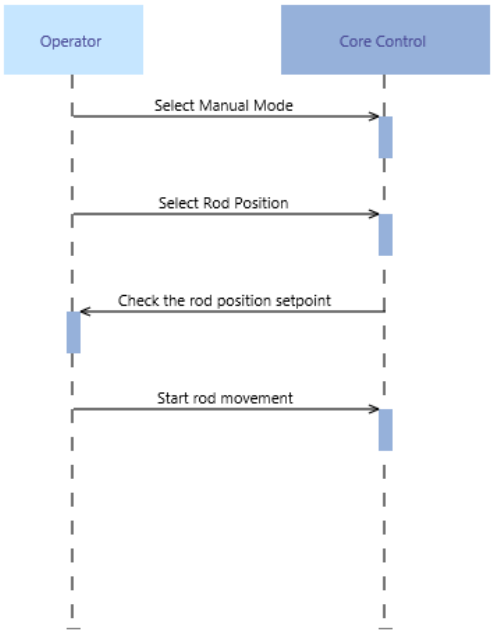
- L'ensemble des fonctions et services du diagramme de séquence est inclus dans l'ensemble des services du diagramme de contexte et des fonctions du SOI

  $P3 ::= \forall ds \in DSequence, [P3.1(ds) \text{ and } P3.2(ds)]$

  $P3.1(ds) ::= \forall\ dc \in DContexte, [(dc.soi()=ds.soi()] \Rightarrow [\forall s \in ds.services(), dc.services() \supset s]$

  $P3.2(ds) ::= \forall\ f \in (ds.soi.fonction(), ds.functions() \supset f$

Exemple de modèle



Interopérabilité avec le reste des DSML

Ce diagramme étant externalisé, nous proposons d'implémenter la syntaxe abstraite et concrète fournie dans CAPELLA sous l'environnement Sirius pour l'intégrer à terme dans notre environnement de modélisation via cette plateforme d'Eclipse.

6.4 DSML de Dépendance

Le DSML de dépendance est un DSML permettant de spécifier soit le comportement entrée/sortie d'une fonction ou alors les données d'entrée, e.g. les conditions limites ou signal d'entrée d'excitation.

Carte d'identité du DSML

Nom du DSML : **DSML de dépendance**

Vue de Rattachement : **Vue comportementale**

Diagramme(s) support du DSML **Diagramme de dépendance**

Objectifs

Modélisation

Le DSML de dépendance a pour but de spécifier les scénarios de fonctionnement entrée-sortie, i.e. le comportement entrée-sortie attendu d'un élément de modélisation (systèmes et les fonctions) en boite noire. Ces scénarios peuvent être dépendants du temps ou non.

En temps normal, la spécification d'un scénario entrée-sortie est réalisée via une exigence soit allouée à un système ou à une fonction (selon sa nature) pour en spécifier le comportement. Ici, on propose de passer par ce DSML pour décrire le comportement. Autrement, le DSML peut aussi bien raffiner une exigence portant sur le comportement de cet élément de modélisation, aidant à caractériser de façon plus précise le comportement.

Lorsqu'il ne dépend pas du temps, le DSML de dépendance caractérise une relation entrée-sortie pouvant s'exprimer de manière mathématique ou sous forme d'un signal comme montré sur la Figure 41. On peut voir que $s = f(e)$.

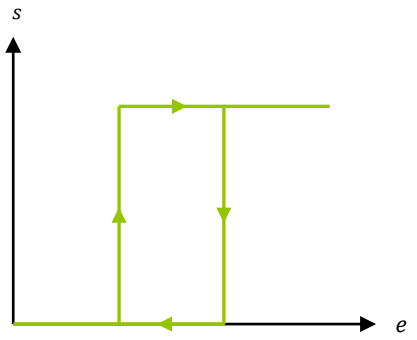


Figure 41: illustration de la relation de dépendance

Lorsqu'il est dépendant du temps (comme montré sur la Figure 42), ce DSML permet de spécifier un scénario type entrée-sortie. Dans ce cas, un scénario spécifié par ce DSML peut servir : de condition limite pour le signal en question (sur la figure c'est le signal de puissance *Power*). Ce signal peut être utilisé pour simuler (vérification par simulation) ou encore comme scénario de test de la fonction ou du système.

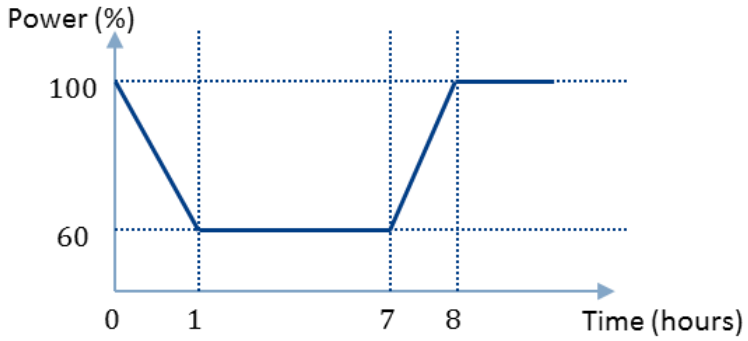


Figure 42: Représentation avec le temps

Vérification

Ce DSML participe à la vérification pour plusieurs raisons :

- Il contribue à la vérification dans la phase remontante de l'IVTV. Les scénarios qu'il spécifie peuvent être utilisés pour des cas test (*use case scenarios*).

- Il sert dans le cadre de la vérification des modèles dans le sens descendant (cf. 2.1). Ici, la vérification par **simulation** mais aussi par **modélisation guidée**. En ce qui concerne la simulation, le DSML de dépendance sera utilisé pour spécifier le comportement attendu entrée-sortie des fonctions dans le DSML Fonctionnel (cf. 6.6). Il consiste (dans la version dépendant du temps) en une spécification des conditions limites d'un signal d'entrée ou de sortie. Dans la version non dépendant de temps, il permet de spécifier le comportement attendu d'une fonction. Dans notre

étude, ce comportement est décrit par le DSML de Spécification fonctionnelle (cf. 6.7). Dans ce cas, le DSML de dépendance sert à vérifier que le DSML de spécification fonctionnelle est conforme aux attentes, en comparant en simulation les entrées-sorties des deux DSML.

- En ce qui concerne la modélisation guidée, nous donnons plus loin dans cette partie les propriétés modèles définissant les règles de construction du DSML devant être respectées. Elles permettent donc de vérifier que le modèle est **bien construit**, i.e. qu'il ne possède pas d'incohérences du moins dans sa construction.

- Il fait enfin l'objet, malgré le choix fait, d'intégrer directement un DSML existant (*Matlab Simulink*) de possibilités de preuve de construction par des propriétés modèles donnés plus loin.

Rôle(s) et objectif(s) propre(s) à chaque acteur métier souhaitant utiliser le DSML

L'utilisateur utilise ce DMSL pour raffiner des exigences opérationnelles (e.g. variation en rampe) ou spécifier le comportement entrée/sortie d'un système ou d'une fonction par leurs items.

Place et cohérence dans le cycle en V

Ce DSML peut intervenir à tout moment de la conception. Il suffit de disposer d'items ou de fonctions dont on veut spécifier le comportement.

Syntaxe abstraite

NB : les syntaxes abstraite et concrète ainsi que la sémantique opérationnelle de ce DSML sont directement inspirées par le formalisme de Matlab Simulink.

Item
L'item est le principal concept dont nous avons besoin pour décrire ce DSML. En effet, les entrées/sorties des systèmes et fonctions, dont nous souhaitons définir le comportement, sont représentées par ce concept.

Diagramme de dépendance
Le diagramme de dépendance est le concept support du DSML de dépendance. Il est en relation avec les autres concepts constituant le DSML et permet de créer des instances du DSML lors de la modélisation.

Interaction avec les autres DSML

Modeleur d'exigences→ Diagramme de dépendance

Les exigences opérationnelles définissent les scénarios opérationnels entrée/sortie des systèmes qui peuvent être repris par ce diagramme.

Diagramme de contexte

Le diagramme de contexte fournit les items partagés par le SOI et les systèmes à l'interface. On peut alors éditer un diagramme de dépendance à partir de ce dernier.

Propriétés modèle

Dans notre étude, nous proposons dans la suite une unique propriété modèle (statique) consistant à s'assurer a minima que les entrée/sorties du diagramme de dépendance sont des items déjà existants dans le modèle courant du SOI :

$$P1 ::= \forall dd \in DDependance, [\forall i \in dd.inputItems(),dd.outputItems () <> null]$$

avec DDependance = {dd = élément de modélisation / dd.class()=DependanceDiagram}

Exemple de modèle

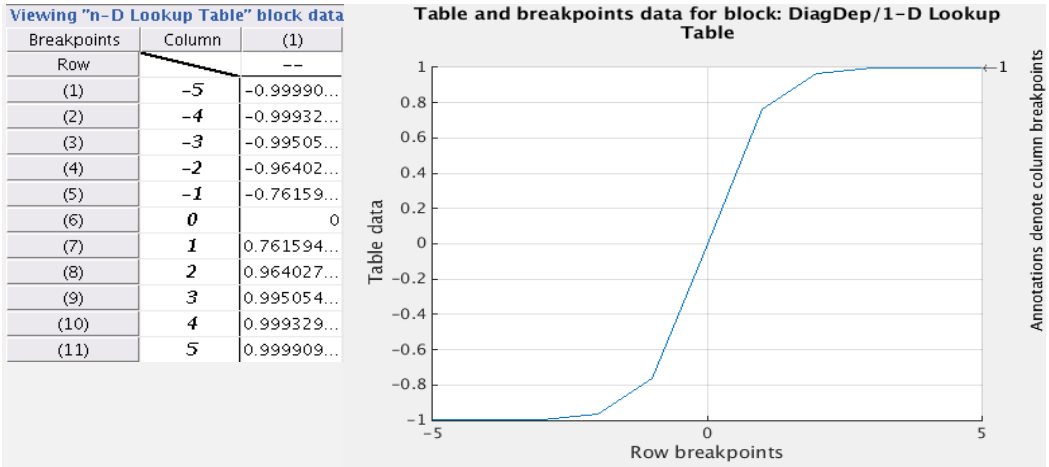


Figure 43: Exemple de diagramme de dépendance n°1. Diagramme réalisé avec l'outil Matlab Simulink : Lookup Table

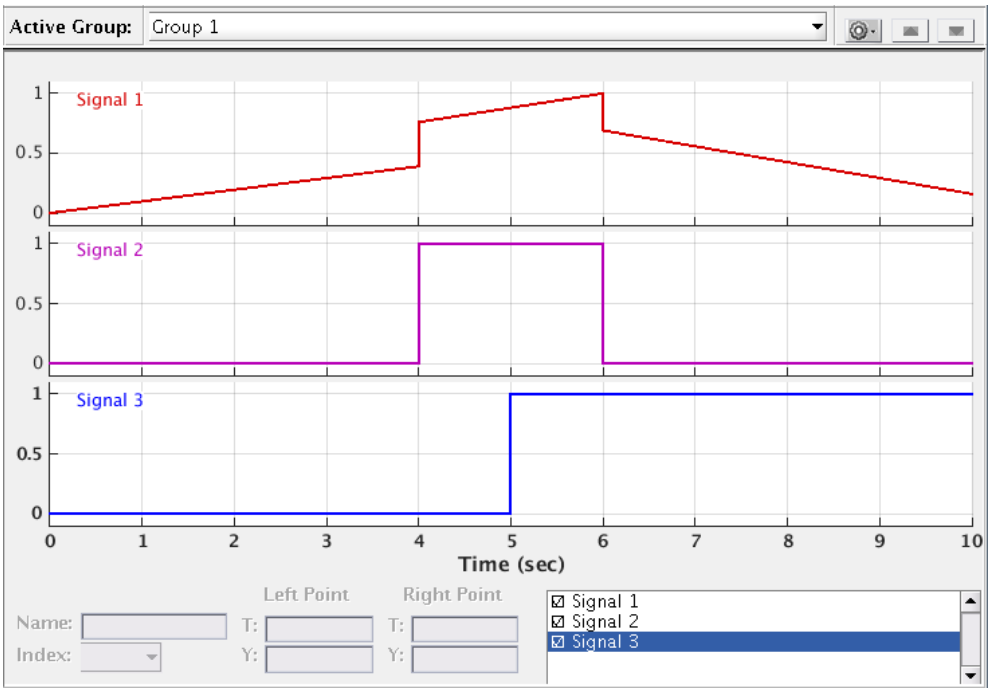


Figure 44: Exemple de diagramme de dépendance (temporel) n°2. Diagramme réalisé avec l'outil Matlab Simulink : Signal Builder

Interopérabilité avec le reste des DSML

Ce DSML est externalisé et implémenté sous Matlab qui est un outil lui-même interfaçable avec des outils Java. Sirius étant implémenté sous Eclipse et Java, il faudra à terme développer une interface Java afin de lancer l'édition des DSML de Matlab depuis nos DSML (par exemple depuis une fonction pour un diagramme de dépendance sans le temps, ou d'une entrée ou une sortie pour un diagramme de dépendance avec le temps).

6.5 DSML de Modes & Transitions

Carte d'identité du DSML

Nom du DSML :	**DSML de Modes et Transitions**
Vue de Rattachement :	**Vue comportementale**
Diagramme(s) support du DSML	**Diagramme de Modes et Transitions**

Objectifs

Modélisation
Un des moyens de spécifier le comportement boite blanche du SOI se fait par la définition de ses modes et transitions. Les **modes** permettent de dégager et **regrouper** des comportements spécifiques du SOI. Par exemple, le mode automatique, le mode manuel ou encore le mode dégradé.

Nous proposons alors que le comportement entrée-sortie du système dans chaque mode soit décrit **uniquement** par les **fonctions valides dans ce mode**.

De cette manière, la conception est organisée à un haut niveau d'abstraction (e.g. le SOI possède un mode manuel, un mode automatique et un mode hybride). La définition des modes oblige le concepteur à penser à tous les cas d'utilisation possibles du SOI.

Vérification
Tout comme les DSML précédents, ce DSML permet de faire de la vérification au sens descendant (cf. 2.1). Deux méthodes de vérification sont utilisées par ce DSML :

- La modélisation guidée : nous donnons plus loin dans cette partie les propriétés modèles définissant les règles de construction du DSML devant être respectées. Elles permettent donc de vérifier que le modèle est **bien construit**, i.e. qu'il ne possède pas d'incohérences du moins dans sa construction.

- La simulation : ce DSML est simulable. Les modes et transitions sont représentés et liés entre eux par ce DSML. Lorsqu'il est simulé, l'utilisateur peut voir l'enchainement des modes à travers leur activation et le franchissement des transitions. Il peut alors vérifier

que le modèle est bien construit, c'est-à-dire, qu'il correspond à l'idée qu'il en avait lorsqu'il l'a conçu.

Rôle(s) et objectif(s) propre(s) à chaque acteur métier souhaitant utiliser le DSML

L'utilisateur de ce DSML a pour objectifs de :

- Définir les modes et les transitions du SOI à partir des exigences fonctionnelles et opérationnelles de son référentiel d'exigences.
- A partir de chaque mode il pourra créer un diagramme fonctionnel pour décrire le comportement entrée/sortie du système

Place et cohérence dans le cycle en V

Ce DSML intervient à la frontière entre la phase d'ingénierie des exigences (R de RADIV) et la phase d'ingénierie des architectures (A de RADIV). Il permet de définir le comportement du SOI et est déduit des exigences opérationnelles et fonctionnelles du système. Il mène vers la définition de l'architecture fonctionnelle du SOI (les fonctions étant définies dans chaque diagramme fonctionnel attaché à chaque mode).

Syntaxe abstraite (cf. Figure 45)

Le Mode
Les modes opérationnels du SOI (cf. 5.7).

La transition
Les transitions entre les modes.

Condition
Les conditions sont associées aux transitions. Ces dernières sont franchies quand la condition est vraie.

Le diagramme de modes et transitions (*Mode Diagram*)
Le diagramme de modes et transitions est le concept support du DSML de Modes et Transitions. Il est en relation avec les autres concepts constituant le DSML et permet de créer des instances du DSML lors de la modélisation.

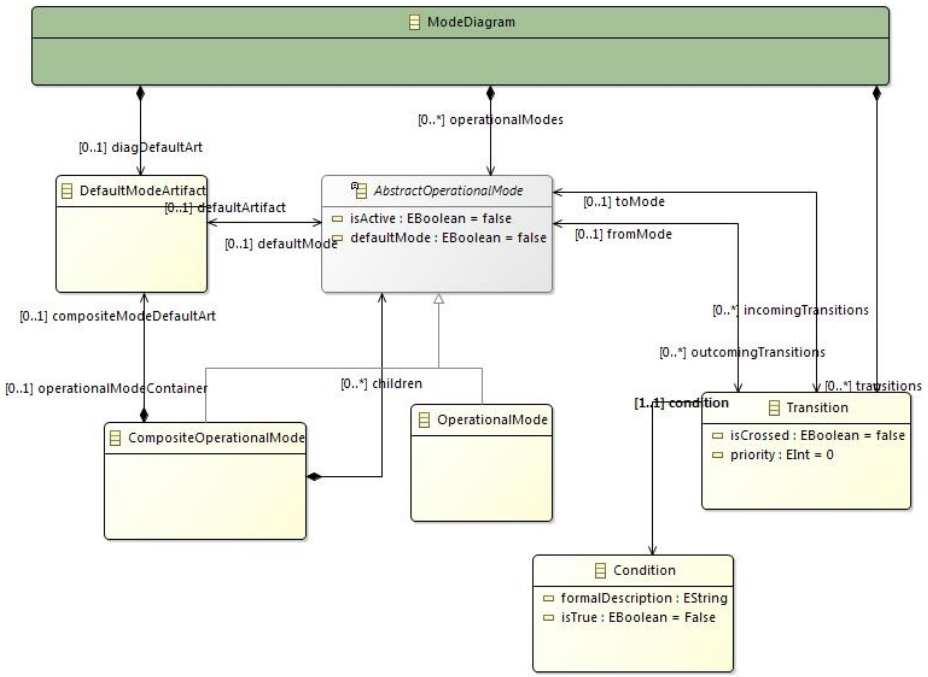


Figure 45: Méta modèle de la vue Comportementale (M/T)

Interaction avec les autres DSML

DSML d'architecture

Un unique diagramme de mode est associé au SOI, lui-même décrit dans le diagramme de contexte (vue contextuelle du DSML d'architecture). L'idée est de considérer un unique groupement de modes en interaction pour décrire le comportement du système. A noter que le comportement entrée-sortie du système sera décrit par les fonctions valides dans les modes.

DSML fonctionnel

Le DSML fonctionnel contient une partie de l'architecture fonctionnelle du SOI, montrant les fonctions, les flux qu'elles échangent. Les fonctions décrites dans ce DSML sont les fonctions valides dans un mode particulier. Ainsi, un unique diagramme fonctionnel est associé à un mode.

Syntaxe Concrète

Eléments graphiques	Description de l'élément graphique
	Transition d'un mode à un autre
	Indicateur du mode par défaut

	Modes opérationnels
	Condition
	Association d'une Condition à une Transition

Sémantique Opérationnelle

La définition de la sémantique opérationnelle consiste à donner les règles d'exécution du langage. Nous donnerons donc les règles d'initialisation du modèle (IM), de lecture des entrée (RI), de calcul des états futurs (CNS) et d'écriture des sorties (WO) telles que définies dans 0.

IM - Initialisation des entrées du modèle

- Activer l'état initial M_0

  $M_{actif} = M_0$

RI – Lectures des entrées

- Récupérer les valeurs d'entrée dans le blackboard puis,
- Figer les valeurs des variables de modélisation employées dans le diagramme des modes.

CNS – Calcul des états futurs

$x =: \emptyset$

$\forall T_j \in \{M_0.\text{outcomingTransition}()\}$ do {

if $(T_j.\text{condition}())$, then

$x := j$;

exit/

if $(x = \emptyset)$, then exit

else,

$M_{actif}.\text{isActive}() := \text{false}$;

$M_{actif} = T_x.\text{toMode}()$;

$\text{executeFunctionalDiagram}(M_{actif})$;

exit}

WO – Ecriture des sorties

- Il n'y a pas de sorties modifiées.

Propriétés modèle

Les propriétés données ci-dessous sont des exemples de propriétés statiques (de construction) ou de propriétés dynamiques devant alors être vérifiées pour chaque pas de simulation au moyen d'un « *model checker* » :

- (statique) Tout mode possède au moins une transition de sortie

$$P_3^M ::= \forall\, M_i \in Mode, M_i.outcomingTranistions() \neq \emptyset$$

- (statique) Tout mode possède au moins une transition d'entrée

$$P_4^M ::= \forall\, M_i \in Mode, M_i.incomingTranistions() \neq \emptyset$$

- (dynamique) Il existe une et une seule transition franchissable à tout instant

$$P_1^M ::= \forall t \in T, \forall\, M_i \in Mode, \forall\, T_j \in M_i.outcomingTransitions() \;\Rightarrow XOR(T_j.condition)$$

- (dynamique) Il existe au moins une transition franchissable pour tout mode quel que soit l'instant

$$P_2^M ::= \forall t \in T, \forall\, M_i \in Mode, \exists T_j \in M_i.outcomingTransitions() \;/\; T_i.condition$$

- (dynamique) A tout instant, tout mode est atteignable à partir de n'importe quel mode (graphe connexe)

$$P_5^M(M_1, M_2) ::= \forall t \in T, \exists T \in M_1.outcomingTransitions(\,) \;/\; (T.toMode = M_2)\; AND\big(T.condition(t)\big)\; OR\; (P(T.toMode, ED))$$

- (dynamique) A l'instant t_0 il existe un mode actif

$$P_6^M ::= en\; t_0, \exists M_j \;/\; M_j.isActive = true$$

Modèles de référence

Le modèle de référence que nous conseillons est le modèle GEMOS développé par [V. Chapurlat 2013 ???]. Il définit un ensemble de modes standards pouvant être pris par défaut.

Exemple de modèle

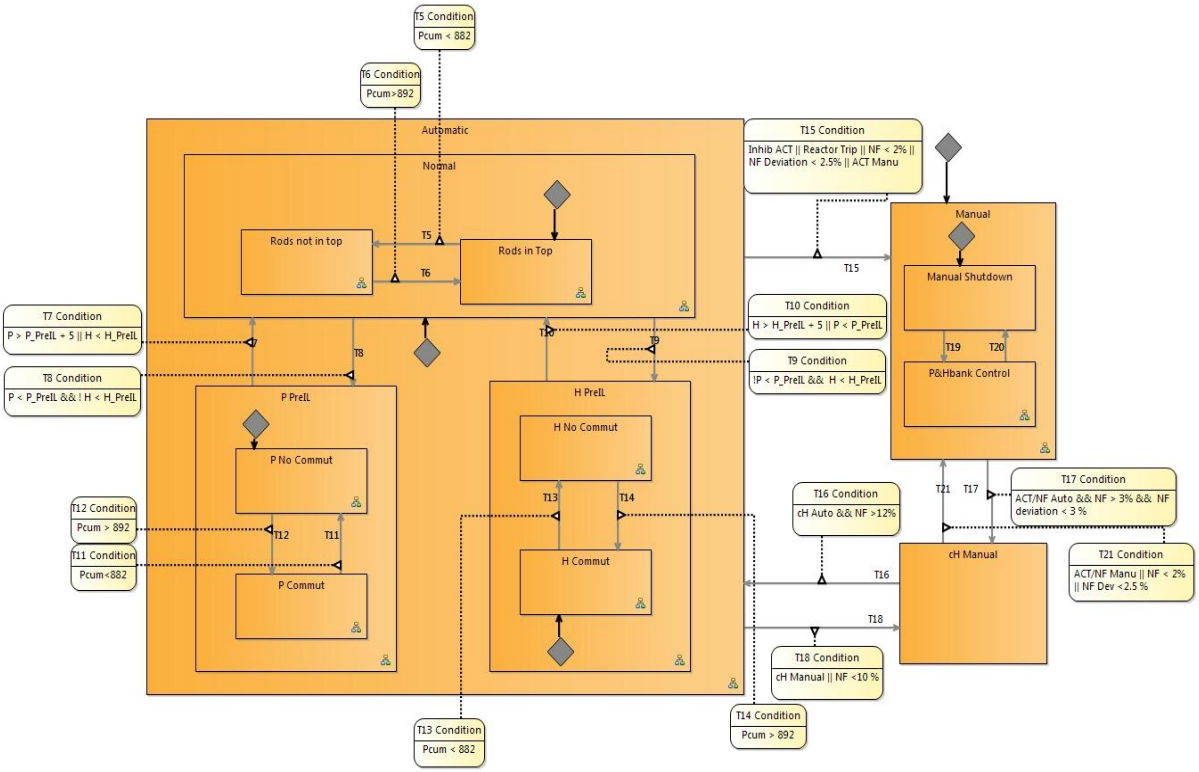


Figure 46: Diagramme de Modes et Transition

6.6 DSML Fonctionnel

Carte d'identité du DSML

Nom du DSML : **DSML fonctionnel**

Vue de Rattachement : **Vue fonctionnelle**

Diagramme(s) support du DSML **Diagramme fonctionnel**

Objectifs

Modélisation

Le DSML fonctionnel représente tout ou partie de l'architecture fonctionnelle du système. Le diagramme fonctionnel est le support de ce DSML. A chaque mode de fonctionnement on associe un unique diagramme fonctionnel, représentant l'ensemble des fonctions valides dans ce mode ainsi que leurs interactions.

Ce langage est inspiré du langage eFFBD (enhanced Functional Flow Block Diagram [Vitech, 2011]. Cette voie est privilégiée car elle permet la dynamique nécessaire dans toute architecture fonctionnelle qui ne peut être décrite par une simple hiérarchie de fonctions. A la différence de SysML qui dispose dans sa vue comportementale à la fois d'un diagramme d'états, d'un diagramme d'activités et d'un diagramme de séquences, notre étude réduit à deux langages dans la vue comportementale : le langage des Modes et Transitions et le langage eFFBD. L'utilisation d'un diagramme eFFBD peut paraître complexe. Son formalisme, de par l'enchevêtrement des fonctions dans des '*constructs*' et qui échangent des flux, peut rendre réticents les utilisateurs inexpérimentés. Dans l'organisation d'un grand groupe comme Framatome, une politique du changement aussi conséquente pourrait rencontrer un fort rejet. C'est pourquoi, nous proposons aussi d'utiliser un formalisme plus simple en alternative pour les utilisateurs qui le désirent.

Comme nous l'avons dit précédemment, nous nous intéressons principalement aux systèmes de contrôle commande. Ces systèmes ont la particularité d'exécuter leurs fonctions avec des pas de temps de calcul spécifiés. C'est une classe de système que l'on retrouve très souvent dans les systèmes I&C Framatome.

Dans ce cas particulier, nous considèrerons que les fonctions associées à un système de type I&C **s'exécutent** de manière massivement **parallèle** à

chaque **pas de temps de calcul**. Dans le formalisme eFFBD on représente cette configuration comme sur la Figure 47.

L'alternative simplifiée de l'eFFDB dans ce cas se représente par la Figure 48. On peut voir que les '*constructs*' sont retirés. On allège donc la représentation, qui se rapproche par la même occasion de celle utilisée par Framatome. On garde ainsi les habitudes métier de l'industriel.

En conclusion, puisque nous traitons **uniquement** le cas des systèmes **I&C** de Framatome, nous utiliserons uniquement la **version simplifiée d'eFFBD**. Dans le cas d'une application étendue de la méthode, l'utilisation du DSML eFFBD complet serait nécessaire.

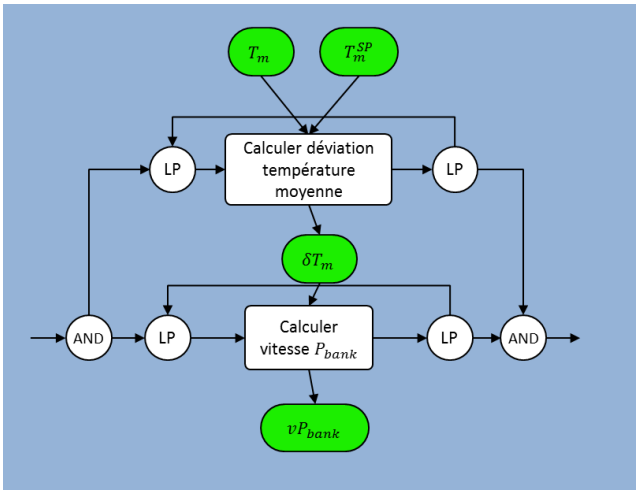


Figure 47: Exemple de diagramme eFFBD standard

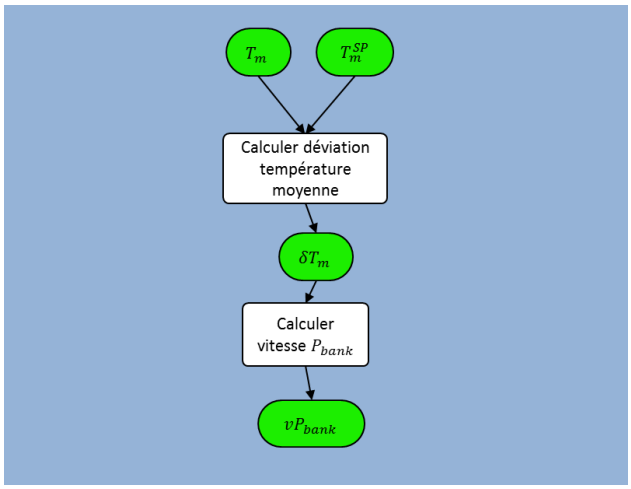


Figure 48: Exemple de diagramme eFFDB simplifié

Vérification

Tout comme les DSML précédents, ce DSML permet de faire de la vérification au sens descendant (cf. 2.1). Deux méthodes de vérification sont utilisées par ce DSML :

- La modélisation guidée : nous donnons plus loin dans cette partie les propriétés modèles définissant les règles de construction du DSML devant être respectées (e.g. les fonctions possèdent au moins une entrée et une sortie). Elles permettent donc de vérifier que le modèle est **bien construit**, i.e. qu'il ne possède pas d'incohérences du moins dans sa construction.

- La simulation : ce DSML est simulable. Le comportement « attendu » entrée-sortie des fonctions est décrit par un ou plusieurs DSML de dépendance. Le comportement « réel », i.e. le traitement entrée-sortie, de la fonction est décrit par le DSML de spécification fonctionnelle (cf. 6.7). L'analyse du comportement attendu et réel en simulation se fait au niveau de la fonction par comparaison des entrées-sorties. On vérifie alors que le comportement de la fonction est bien conforme à ce qui était attendu.

Rôle(s) et objectif(s) propre(s) à chaque acteur métier souhaitant utiliser le DSML

L'utilisateur se servira de ce DSML pour définir l'architecture fonctionnelle. Si on se trouve dans le cas eFFBD, il définira l'enchainement des fonctions et les flux. Dans le cas du diagramme simplifié, il définira uniquement les flux entre les fonctions.

Place et cohérence dans le cycle en V

Ce diagramme intervient dans la phase de définition des architectures, i.e. dans la phase A de RADIV.

Syntaxe abstraite (cf. Figure 49)

Fonctions (*Function*).
Le concept de fonction est défini dans 5.4. Les fonctions sont tirées de la phase d'IE [ISO15288, 2015] i.e. déduites des exigences fonctionnelles contenues dans le référentiel d'exigences. Les fonctions utilisées sont valides dans le mode associé au DSML fonctionnel. Par exemple, dans le cas d'un véhicule autonome, la fonction de pilotage automatique du véhicule est activée uniquement dans son mode automatique. Dans le cas où le véhicule est en mode manuel, cette fonction n'est pas active. Une fonction peut aussi être activable dans plusieurs modes, e.g. le freinage d'urgence est valide en mode automatique et manuel.

Item
Le concept d'Item permet de représenter les entrées et sorties des fonctions.

Diagramme fonctionnel

Le diagramme fonctionnel est le concept support du DSML fonctionnel. A chaque mode est associé un diagramme fonctionnel qui modélise les fonctions valides et leur enchainement lorsque le SOI se trouve dans ce mode ainsi que les flux qu'elles partagent.

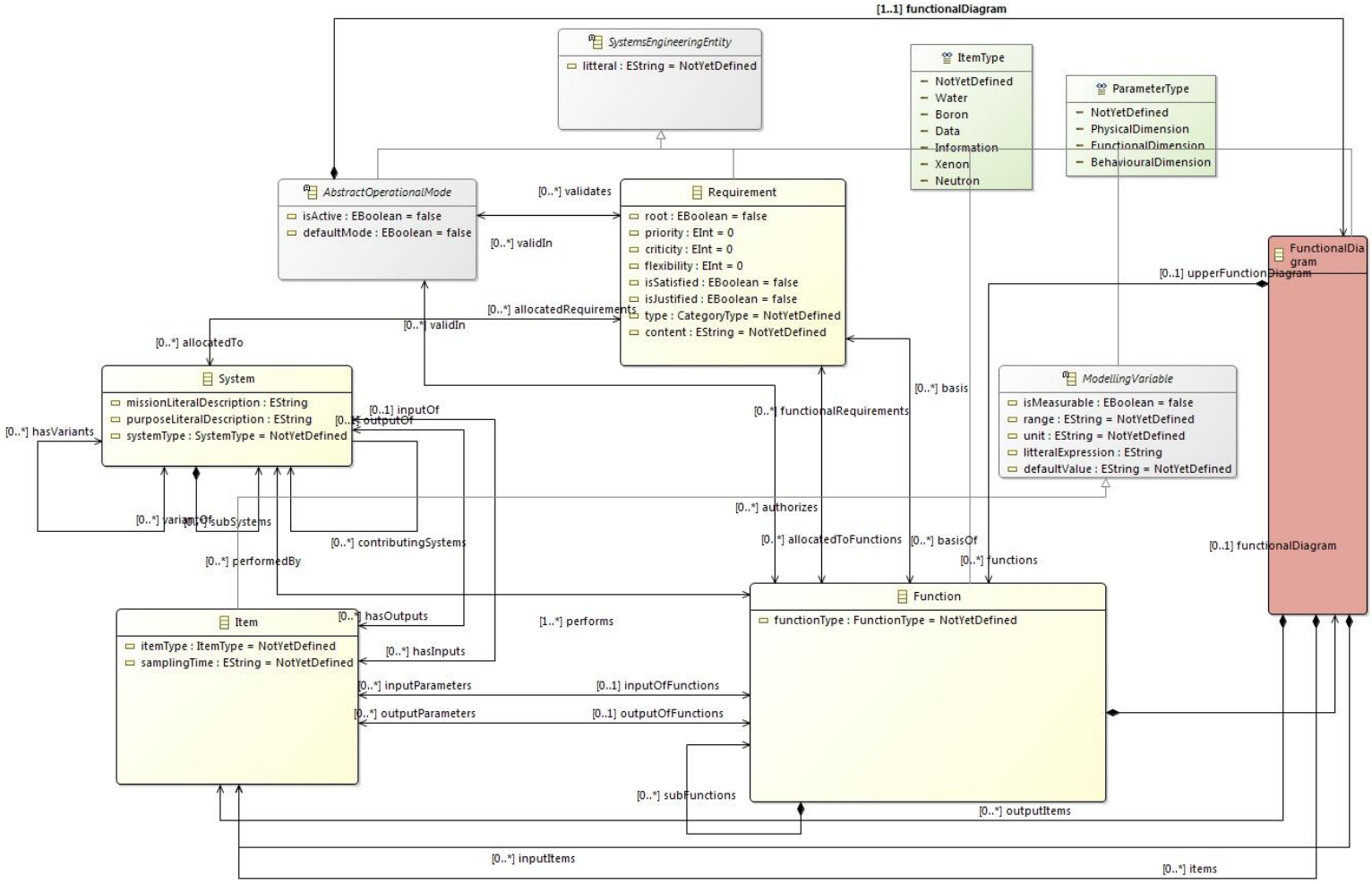


Figure 49: Méta modèle du diagramme fonctionnel

Interaction avec les autres DSML

Diagramme de modes

Un diagramme fonctionnel est défini par mode. L'ensemble des fonctions valides dans ce mode sont représentées dans le diagramme fonctionnel. On assure par cette relation que dans chaque mode l'ensemble du comportement entrée-sortie du système est couvert.

Syntaxe Concrète

Eléments graphiques	Description de l'élément graphique
	Items échangés par les fonctions et les entrées/sorties
	Fonctions
	Entrées
	Sorties

Sémantique Opérationnelle

La définition de la sémantique opérationnelle consiste à donner les règles d'exécution du langage. Nous donnerons donc les règles d'initialisation du modèle (IM), de lecture des entrée (RI), de calcul des états futurs (CNS) et d'écriture des sorties (WO) telles que définies dans 0.

IM - Initialisation des entrées du modèle

- Rien à faire

RI – Lectures des entrées

- Figer les valeurs des variables de modélisation du modèle complet
- Récupérer les valeurs d'entrée dans le blackboard

CNS – Calcul des états futurs

Soit n

for (i = 1 to n)[do {executeSpecificationDiagram(F_i);

exit}]

WO – Ecriture des sorties

- Ecrire les valeurs des sorties dans le blackboard

Propriétés modèle

- (statique) cette propriété notée P_1^F précise que toute fonction F_i possède au moins un item d'entrée et un item de sortie qui doivent 1) être différents l'un de l'autre (pas de bouclage des sorties sur les entrées ; cette première propriété est formalisé par la propriété modèle statique $P_1^F(F_i)$), et 2) de nature homogène selon la matrice ci-dessous :

Entrée \ Sortie	M matière	E énergie	I information
M matière	Transformer	Transformer	Mesurer, Evaluer, …
E énergie		Transformer	Mesurer, Evaluer, …
I information			traiter

Figure 50: validation des transformations effectuées par une fonction [Meinadier, 1998]

Cette matrice indique la nature de la transformation E/S (vision donc boite noire) opérée par une fonction en tenant compte de la nature des items d'entrée et de sortie. Les cellules rouges indiquent par exemple une impossibilité fonctionnelle (e.g. il n'est pas possible de transformer une information en matière) que nous allons donc traduire dans la propriété modèle statique $P_2^F(F_i)$ donnée ci-dessous.

$$P_1^F ::= \forall F_i \in Fonction, [P_1^F(F_i)\ AND\ P_2^F(F_i)\ ]$$

Avec :

$$P_1^F(F_i) :: = [(F_i.inputParameters() \neq \emptyset)\ AND\ (F_i.outputParameters() \neq \emptyset) AND\ (\forall\ p \in F_i.inputParameters(),\ p \notin F_i.outputParameters())\ AND\ (\forall\ p \in F_i.outputParameters(),\ p \notin F_i.inputParameters())]$$

$$P_2^F(F_i) ::= [\ P_{2.1}^F(F_i)\ AND\ P_{2.2}^F(F_i)\ AND\ P_{2.3}^F(F_i)\ ]$$

Avec :

$$P_{2.1}^F(F_i) ::= [\ \forall\ p \in F_i.outputParameters(), p.nature() = 'Material'] \Rightarrow [\exists q \in F_i.inputParameters(), q.nature() = 'Material']$$

$$P_{2.2}^F(F_i) ::= [\ \forall\ p \in F_i.outputParameters(), p.nature() = 'Energy'] \Rightarrow [\exists q \in F_i.inputParameters(), (q.nature() = 'Material'\ \text{OR}\ q.nature() = 'Energy')]$$

$$P_{2.3}^{F}(F_i) ::= [\,\forall\, p \in F_i.outputParameters(), p.nature() = 'Data'] \Rightarrow [\exists q \in F_i.inputParameters(),, (q.nature() = 'Material' \quad \text{OR} \quad q.nature() = 'Energy' OR q.nature() = 'Data'\,]$$

- (statique) Les entrées/sorties de la fonction sont identiques à celles de sa décomposition (préservation des entrées / sorties) ou à celles du SOI dont elle décrit la mission.

$$P_3^F ::= \forall F_i \in Fonction, [P_1^F(F_i)\ AND\ P_{3.1}^F\ AND\ P_{3.2}^F]$$

$$P_{3.1}^{F}(F_i) ::= [(F_i.functionalDiagram.inputItems() = F_i.inputParameters())\ OR\ (F_i.mission().inputItems() = F_i.inputParameters())]$$

$$P_{3.2}^{F}(F_i) ::= [(F_i.functionalDiagram.outputItems() = F_i.outputParameters())\ OR\ (F_i.mission().outputItems() = F_i.outputParameters())]$$

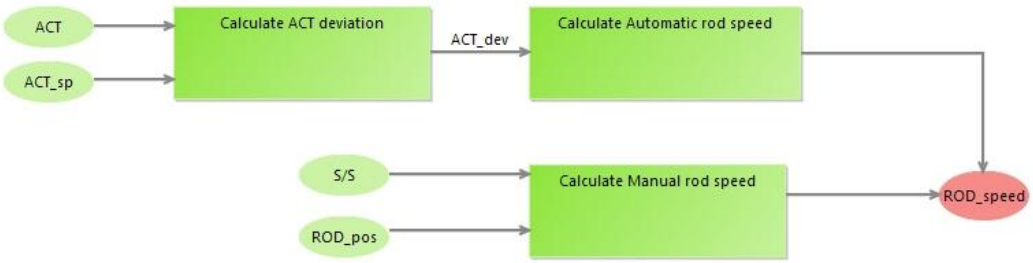
- Toute fonction doit être 1) décomposée au moyen d'un diagramme fonctionnel, ou 2) décrite par un diagramme de spécification, ou 3) il existe au moins un diagramme de dépendance définissant son comportement entrée/sortie selon une vision boite noire. Le connecteur logique OU utilisé ici est bien inclusif. Ce sera à la simulation de permettre de vérifier par exemple que les relations de dépendance entre entrées et sorties de la fonction et le comportement décrit dans le diagramme de spécification sont bien cohérents, aidant alors l'acteur métier à améliorer son modèle ou lever des ambiguïtés

$$P_4^F ::=::= \forall F_i \in Fonction, [(F_i.functionalDiagram() \neq \emptyset)\ OR\ (F_i.specificationDiagram \neq \emptyset) OR\ (F_i.dependanceDiagram \neq \emptyset)]$$

- Tout item défini doit être une entrée ou une sortie d'une fonction

$$P_5^F ::= \forall\, i \in Item, \exists F_i \in Fonction\ (F_i.inputParameters = i\ )\ OR\ (F_i.outputParameters = i\ )$$

Exemple de modèle



6.7 DSML de Spécification Fonctionnelle (SF)

Carte d'identité du DSML

Nom du DSML : **DSML de spécification fonctionnelle**

Vue de Rattachement : **Vue comportementale**

Diagramme(s) support du DSML **Diagramme de spécification fonctionnelle**

Objectifs

Modélisation

Le dernier langage que nous proposons est un langage permettant de définir le traitement entrée-sortie d'une fonction dans le paradigme boite blanche. On l'appelle diagramme de Spécification Fonctionnelle (SF).

Ce DSML permet :

- De garder un formalisme connu et maitrisé et donc de ne pas altérer les méthodes, usages, pratiques et habitudes de travail des acteurs métiers de Framatome, ce dans le but de faciliter le déploiement et l'acceptation à terme de la méthode proposée
- De fédérer l'ensemble des langages de type I&C autour d'un langage unique car différents formalismes utilisés persistent chez Framatome, ce qui a pour effet d'augmenter le nombre de langages.
- De disposer donc d'un DSMLS simple et efficace.

Ce langage est constitué de blocs (boites de différentes formes) dont la sémantique est adaptée au besoin des utilisateurs. Par exemple, la Figure 51 montre un bloc *Limiter* permettant de saturer le signal d'entrée, limitant ses valeurs maximales et minimales à $\pm D$.

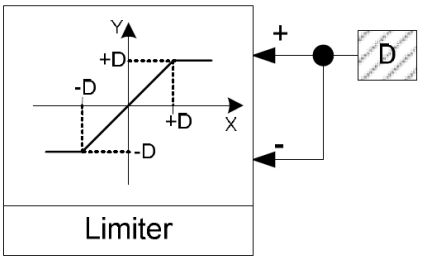


Figure 51: Exemple d'un bloc dans le langage de spécification I&C Framatome (*Limiter*)

Il est aussi constitué d'entrées et de sorties symbolisées par des opérateurs carrés (e.g. Figure 52 *Pcum counter* et *Commut*). Enfin, les blocs, les entrées et les sorties sont connectés par des liens indiquant un signal transitant des uns vers les autres (cf. Figure 52). L'ensemble de ces blocs ont été répertoriés en Annexe A.

Vérification

Tout comme les DSML précédents, ce DSML permet de faire de la vérification au sens descendant (cf. 2.1). Deux méthodes de vérification sont utilisées par ce DSML :

- La modélisation guidée : nous donnons plus loin dans cette partie les propriétés modèles définissant les règles de construction du DSML devant être respectées (e.g. les fonctions possèdent au moins une entrée et une sortie). Elles permettent donc de vérifier que le modèle est **bien construit**, i.e. qu'il ne possède pas d'incohérences du moins dans sa construction.

- La simulation : ce DSML est simulable. Le comportement « réel », i.e. le traitement entrée-sortie des fonctions est décrit par un unique DSML de spécification fonctionnelle, rattaché à la fonction. La simulation montrera l'évolution des entrées-sorties définies dans le DSML. Par leur observation l'utilisateur déduira la conformité du modèle par rapport à ses expectatives.

Rôle(s) et objectif(s) propre(s) à chaque acteur métier souhaitant utiliser le DSML

L'utilisateur de ce DSML a pour mission de définir le comportement entrée/sortie de la fonction associée au DMSL.

Place et cohérence dans le cycle en V

Ce diagramme s'insère dans la partie de définition de l'architecture fonctionnelle du SOI, i.e. le A de RADIV.

Syntaxe abstraite

Pour des raisons de confidentialité, nous ne donnerons pas de détails sur les syntaxes abstraite et concrète complètes ni sur la sémantique opérationnelle du DSML de Framatome. Pour l'exemple, nous reprendrons la syntaxe concrète de Matlab Simulink, qui est relativement proche.

Items
Les entrées/sorties définies sur ce diagramme sont tirées du diagramme fonctionnel associé. Sur ces deux diagrammes elles sont représentées par le concept d'item.

Bloc (Framatome)
Le concept de Bloc est le support pour l'ensemble des blocs utilisés par Framatome pour la définition du traitement entrée-sortie. Une partie de ces blocs est donnée en Annexe A.

. La syntaxe concrète des différents blocs n'est cependant pas donnée dans notre étude pour des raisons de confidentialité.

Le diagramme de Spécification fonctionnelle (*SpecificationDiagram*)
Le diagramme de Spécification fonctionnelle est le concept support du DSML de spécification fonctionnelle. Il est en relation avec les autres concepts constituant le DSML et permet de créer des instances du DSML lors de la modélisation.

Interaction avec les autres DSML

Diagramme fonctionnel
Un unique diagramme de spécification fonctionnelle est associé à chaque fonction définie sur le diagramme fonctionnel (cf. 6.6). Il tire ainsi ses entrées/sorties directement de celle associées à la fonction correspondante.

Propriétés modèle

De même, il n'est possible ici que de donner des exemples de propriétés modèles statiques pour en vérifier la construction. Les propriétés dynamiques ne peuvent, de fait, être présentées ici pour les raisons de confidentialité évoquées plus haut.

- (statique) Les items d'entrée/sortie du diagramme de spécification sont les items d'entrée/sortie de la fonction associée

$$P_{21}^{S} ::= \forall sd \in SpecificationDiagram, \left[(sd.\text{function}().\text{inputItems}() = \text{sd.inputItems}())\text{AND}\left[(sd.\text{function}().\text{outputIe ms}() = \text{sd.outputItems}())\right]\right.$$

- (statique) Chaque fonction est décrite dans un unique diagramme de spécification fonctionnelle (propriété P_2^S) et chaque diagramme de spécification fonctionnelle est associée à une unique fonction (propriété P_3^S).

$P_2^S ::= \forall sd \in SpecificationDiagram, [sd.\text{function}().\text{class}() = \text{Function}]$

Avec $SpecificationDiagram = \{em = élément\ de\ modélisation\ / em.class() = SpecificationDiagram\}$

$P_3^S ::= \forall f \in Fonction, [f.specificationDiagram().class() = SpecificationDiagram]$

Exemple de modèle

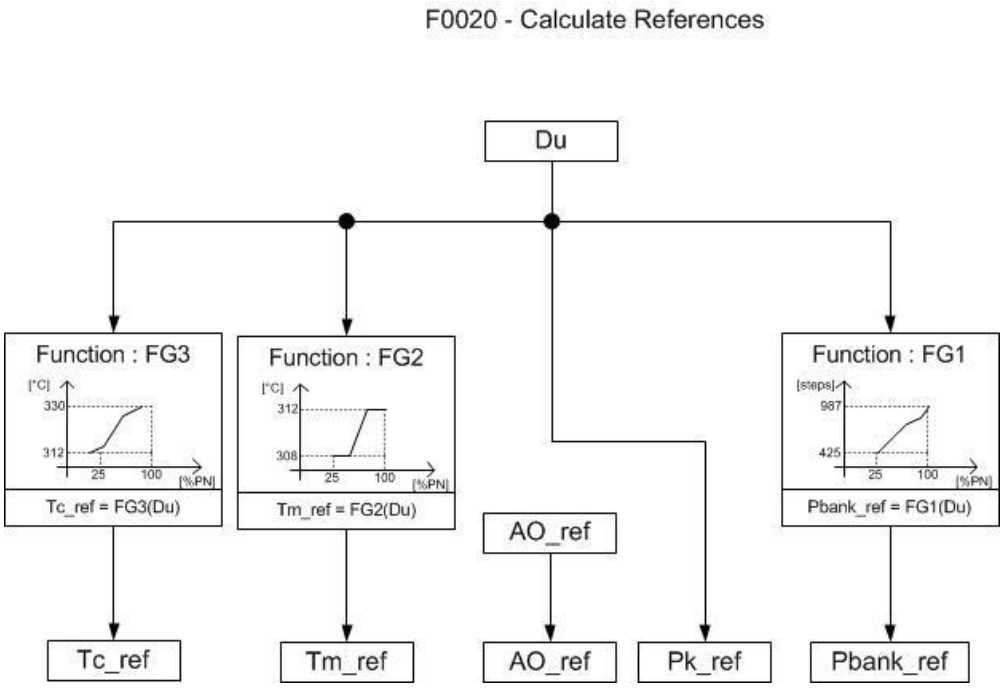


Figure 52: Exemple du langage de spécification I&C Framatome

Interopérabilité avec le reste des DSML

Comme le DSML de dépendance, ce DSML est externalisé. Actuellement en cours de développement par les équipes de Framatome, nous ne pouvons pas proposer d'outils support à ce DSML et donc de moyen d'interopérabilité. Le plus simple pour l'interopérabilité avec les DSML proposés serait qu'ils soient implémentés sur Sirius ou à minima sur Matlab (dans ce cas, de la même manière que le diagramme de dépendance 0, une interface Java est indispensable). En effet, l'environnement de simulation et de vérification des modèles serait alors facilité.

7 La démarche opératoire

La démarche opératoire présente la séquence des activités que chaque acteur métier doit réaliser pour appliquer la méthode.

Avant d'introduire la démarche opératoire, nous présentons les différents liens que possèdent les différents diagrammes développés, supports de la démarche opératoire et plus globalement de la démarche de conception imaginée dans la thèse. Ces liens sont représentés sur la Figure 53. Les blocs gris (e.g. Diagramme de contexte) montrent les diagrammes qui sont adressés dans le cadre de la thèse. Les blocs violets correspondent aux outils utilisés par Framatome. En l'occurrence, Teamcenter et le référentiel d'exigences associé au SOI.

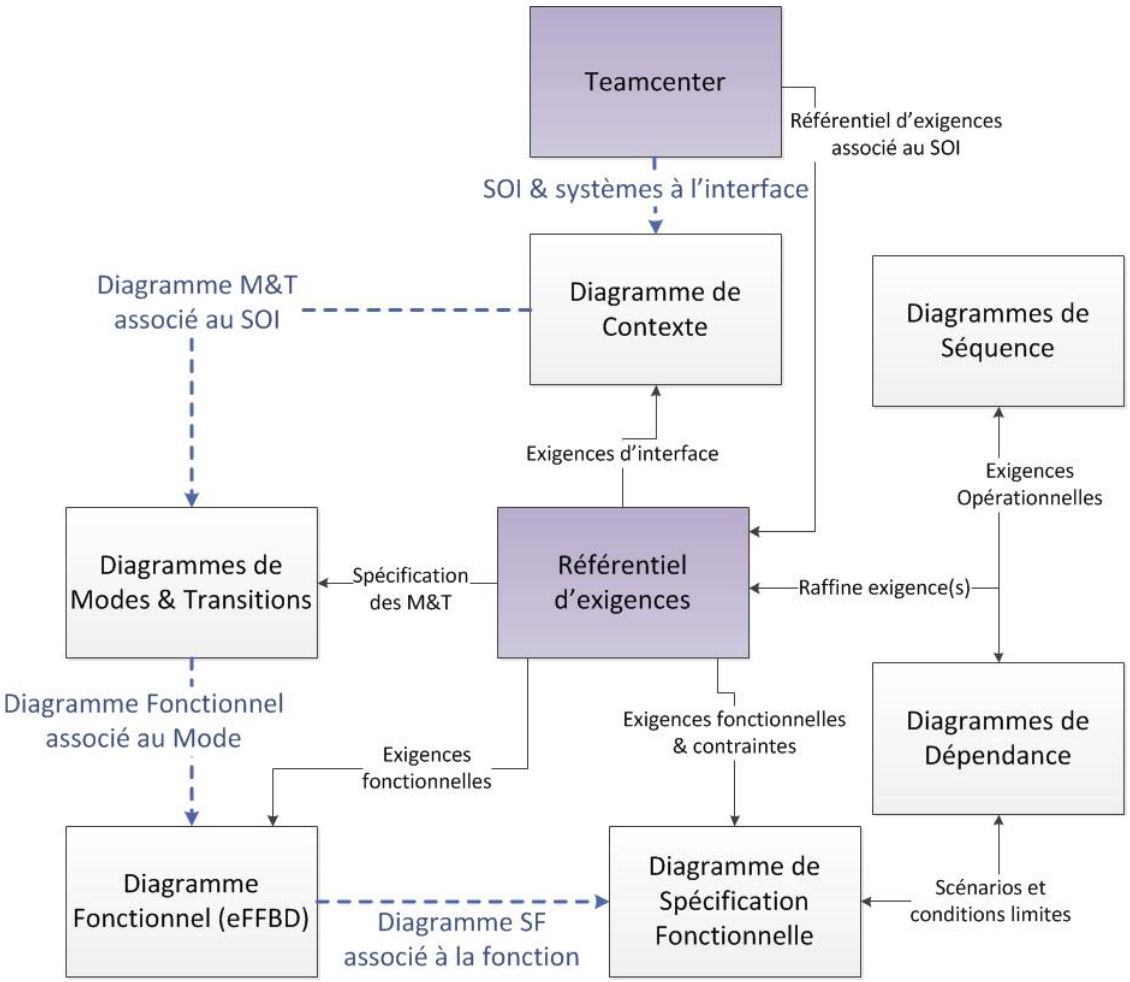


Figure 53: Liens entre les différents langages utilisés

Rappelons que l'une des problématiques que nous adressons est le passage des exigences aux architectures. Comme on peut le voir sur la figure, le référentiel d'exigences joue un rôle central et pivot (au centre de la figure), contenant la totalité des exigences du SOI, modélisées en utilisant le DSML d'exigences. Les

différentes vues gravitent autour de ce référentiel, via les relations introduites plus haut, représentées par les flèches noires. Les exigences sont alors reliées entre-elles ainsi qu'aux éléments de modélisation (fonction/système/scénarios/mode), éléments constitutifs des architectures.

De plus, la méthode proposée s'inscrit dans une démarche progressive allant de l'exigence à l'architecture. Sur la Figure 53, les flèches bleues soulignent le déroulement logique de la méthode. L'utilisateur conçoit le SOI au fil de l'utilisation des différents DSML, dans un ordre logique (partant d'une vision boite noire pour aller vers une vision boite blanche) et apportant de plus en plus de détails sur la solution (jusqu'à la spécification du traitement entrée-sortie du système). Le fil conducteur de la modélisation se résume comme suit : les exigences sont liées entre elles ainsi qu'aux différents éléments de modélisation. En particulier le SOI placé dans son contexte par le diagramme de contexte. Un unique diagramme de modes est associé au SOI. Ces modes servent à regrouper des comportements spécifiques au travers des fonctions valides dans ces modes. Un diagramme fonctionnel est associé à chaque mode, contenant l'ensemble des fonctions valides dans ce mode et formalisant l'architecture fonctionnelle du SOI dans ce mode. Enfin, le traitement entrée-sortie de chaque fonction est décrit par le diagramme de spécification fonctionnelle.

La démarche opératoire est présentée sur la Figure 54 sous la forme d'un diagramme BPMN (Business Process Modelling Notation [BPMN, 2011]). Deux acteurs clés sont identifiés :

- L'architecte : l'acteur responsable de la conception du SOI, aussi bien des processus de management que des processus techniques. Les tâches réalisées par l'architecte sont représentées par les éléments bleus et rouges sur la Figure 54.
- Concepteur : l'acteur maitrisant un domaine métier, intervenant une partie technique de la conception du SOI. Les tâches réalisées par le concepteur sont représentées par les éléments rouges uniquement sur la Figure 54.

La démarche opératoire se déroule comme suit :

- Initialisation de l'étude : l'architecte identifie le SOI, son niveau et sa nature dans l'arborescence produit de Teamcenter (PBS). Il importe le référentiel des exigences du SOI considéré.
- Modélisation du contexte : à l'aide du diagramme de contexte, l'architecte place le SOI dans son contexte, i.e. le SOI et les systèmes à l'interface. Il définit les services, les flux, les items et les liens qui les connectent.
- En parallèle, l'architecte et le concepteur effectuent les tâches suivantes :
  - Modes / Fonctions / Spécifications : l'architecte (uniquement) définit les modes de fonctionnement à l'aide du diagramme de modes et

transitions. Ensuite, l'architecte et le concepteur modélisent les fonctions du système dans chaque mode modélisé précédemment. Ils vérifient les propriétés modèles de ces fonctions. Enfin, ils définissent le comportement entrée-sortie de chaque fonction à l'aide du diagramme de spécification fonctionnelle, tout en vérifiant les propriétés modèles.

Ces tâches se répètent autant de fois que nécessaire en parallèle avec les autres tâches, jusqu'à ce que la solution trouvée réponde aux exigences exprimées.

- Scénarios : l'architecte et le concepteur définissent les scénarios d'évolution du SOI vus en boite noire à l'aide du diagramme de séquences tout en vérifiant les propriétés modèles du DSML.
- Evolution des variables de modélisation : L'architecte et le concepteur définissent les propriétés d'évolution paramétriques ou temporelles des variables de modélisation du SOI à l'aide du diagramme de dépendance. De la même manière que pour les tâches précédentes, ils vérifient les propriétés modèles.
- Exigences : ils définissent les exigences systèmes en utilisant impérativement les relations de raffinement, décomposition, dérivation et reformulation. Ils vérifient de même que les propriétés modèles des exigences sont bien vérifiées. Ensuite, il alloue ces exigences aux éléments de modélisation (fonctions, composants). Enfin, il vérifie et valide le référentiel d'exigences (propriétés modèles et simulation). Si le résultat de la conception ne correspond pas aux attentes, il recommence de la phase de modélisation des exigences.

  Notons que ces tâches s'appliquent itérativement jusqu'à trouver une solution pour laquelle le référentiel d'exigence est vérifié et validé et que tous les modèles sont vérifiés.
- Architecture des sous-systèmes : modéliser et vérifier les propriétés de l'architecture des sous-systèmes du SOI. Ces sous-systèmes sont tirés du PBS quand ils existent (la majorité des cas) et peuvent être créés si nécessaire.

- Une fois ces tâches réalisées, l'architecte et le concepteur valident les modèles et génèrent les justifications nécessaires. Si elles ne sont pas satisfaisantes le cycle de conception redémarre à partir du contexte.
- Enfin, l'architecte génère la documentation et met à jour la base de données Teamcenter avec les exigences et éléments de conception retenus

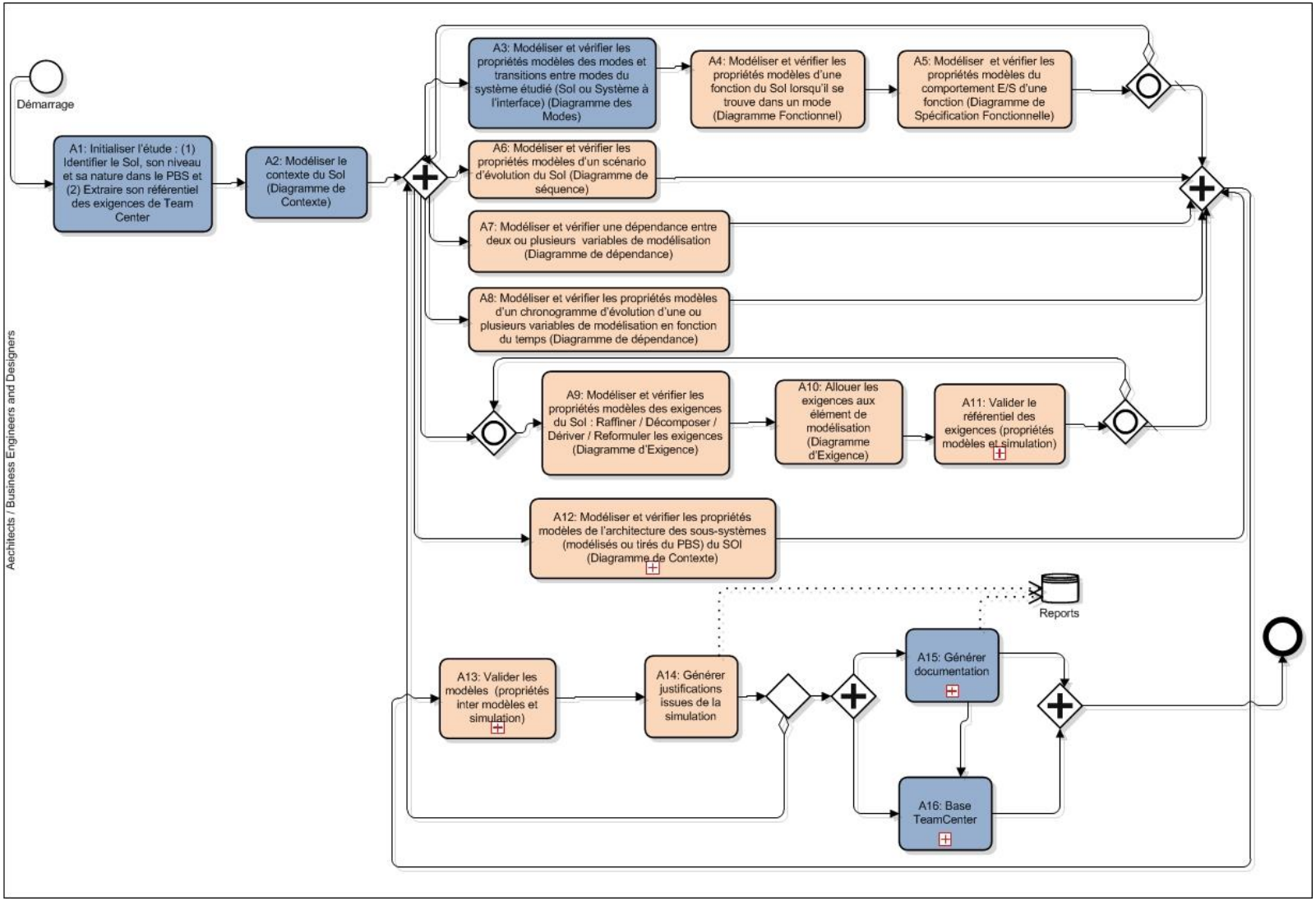


Figure 54: Démarche opératoire

8 Cas d'utilisation – Le *Core Control*

Sommaire

8.1 Les exigences du Core Control 155
8.2 Définition de la vue contextuelle du *Core Control* 157
8.3 Scénarios opérationnels du Core Control 159
8.4 Définition des modes du Core Control 161
8.5 L'architecture fonctionnelle du Core Control 161
8.6 Le diagramme de spécification 163
8.7 Liens entre les diagrammes 165

Comme nous l'avons évoqué dans le chapitre 1, afin d'illustrer la méthode d'IS développée et dans l'optique de répondre à la problématique d'augmentation de flexibilité, nous utilisons le cas d'utilisation du *Core Control*. Nous présentons ici les résultats de l'application de la démarche opératoire présentée en 0.

8.1 Les exigences du Core Control

Les exigences du *Core Control* sont présentées sur la Figure 55. Bien qu'elles aient été rédigées selon les règles établies dans la partie 6.2, nous considérons qu'elles représentent une donnée d'entrée pour notre étude. Par souci d'illustration et de simplification, l'ensemble des exigences allouées au *Core Control* dans le cas industriel ne sont pas présentées ici. La Figure 55 montre les exigences ainsi que les liens qu'elles possèdent entre-elles (cf. 5.3.6) : décomposition, dérivation et raffinement.

Nous invitons le lecteur à se référer à l'Annexe M dans laquelle il trouvera des liens que nous avons formalisés entre les exigences et d'autres éléments de modélisation (e.g. système, mode, scénario, fonction).

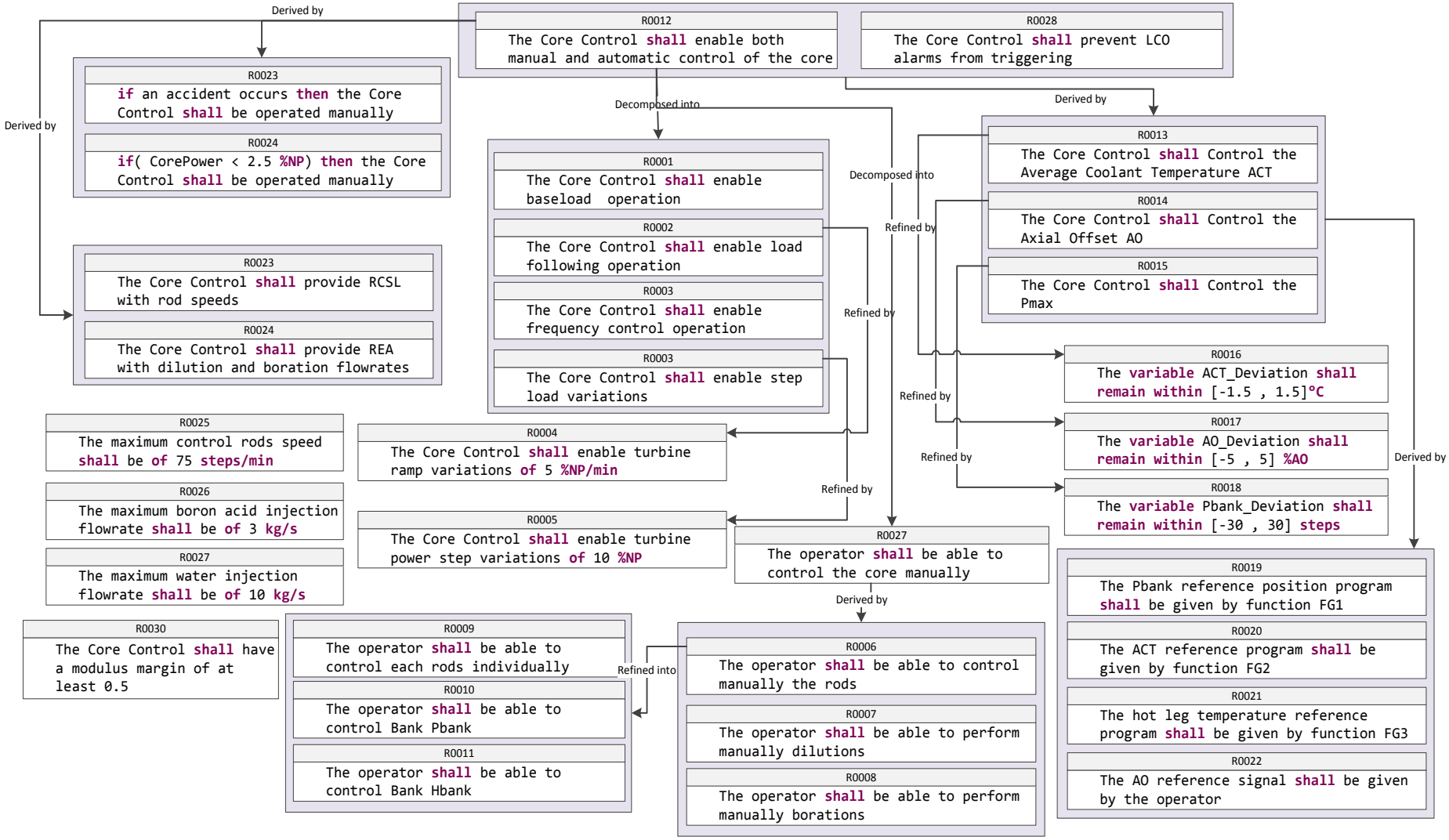


Figure 55 : Exigences sur allouées au *Core Control*

8.2 Définition de la vue contextuelle du *Core Control*

Par l'application de l'activité A2 de la démarche opératoire le diagramme de contexte du SOI est réalisé. Il est présenté sur la Figure 57. Sur cette figure on peut voir :

- Le SOI : le *Core Control*.
- Les systèmes à l'interface, e.g. « I&C other systems ».
- Les services échangés, e.g. « Provide Core measures ».
- Les flux associés aux services, e.g. « Core Measures ».
- Les liens associés aux flux , e.g. « L0002 ».
- Enfin, les items transportés par les flux, e.g. « Tm » la température moyenne.

Etant dans le cas d'une ingénierie centrée sur le PBS, nous connaissons d'ores et déjà les sous-systèmes du *Core Control* qui sont représenté sur le même diagramme. Il est alors aussi possible d'associer certains éléments directement aux sous-systèmes (e.g. le service « Provide Core measures » est directement alloué au sous-système « Banks &CB Control Module »).

Enfin, les exigences présentées dans la partie précédente 0 constituent le référentiel des exigences qui est alloué au *Core Control*, tel qu'illustré sur la Figure 56.

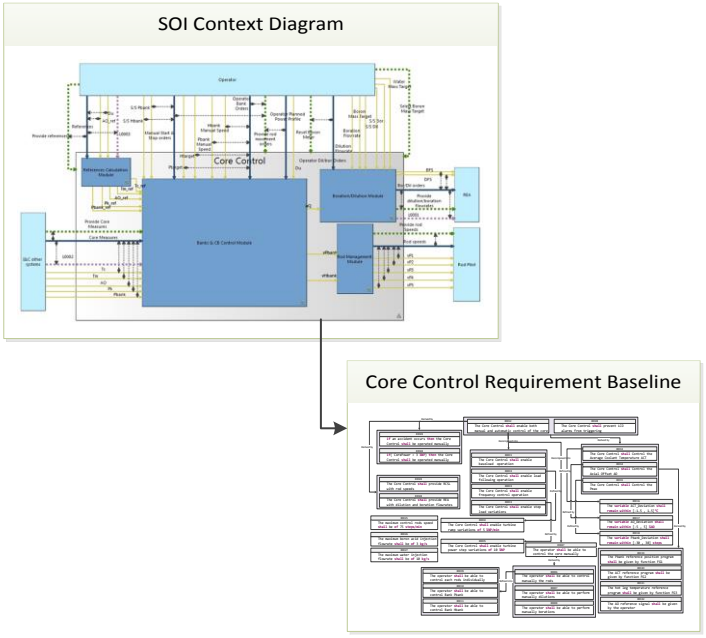


Figure 56: Référentiel d'exigences et SOI

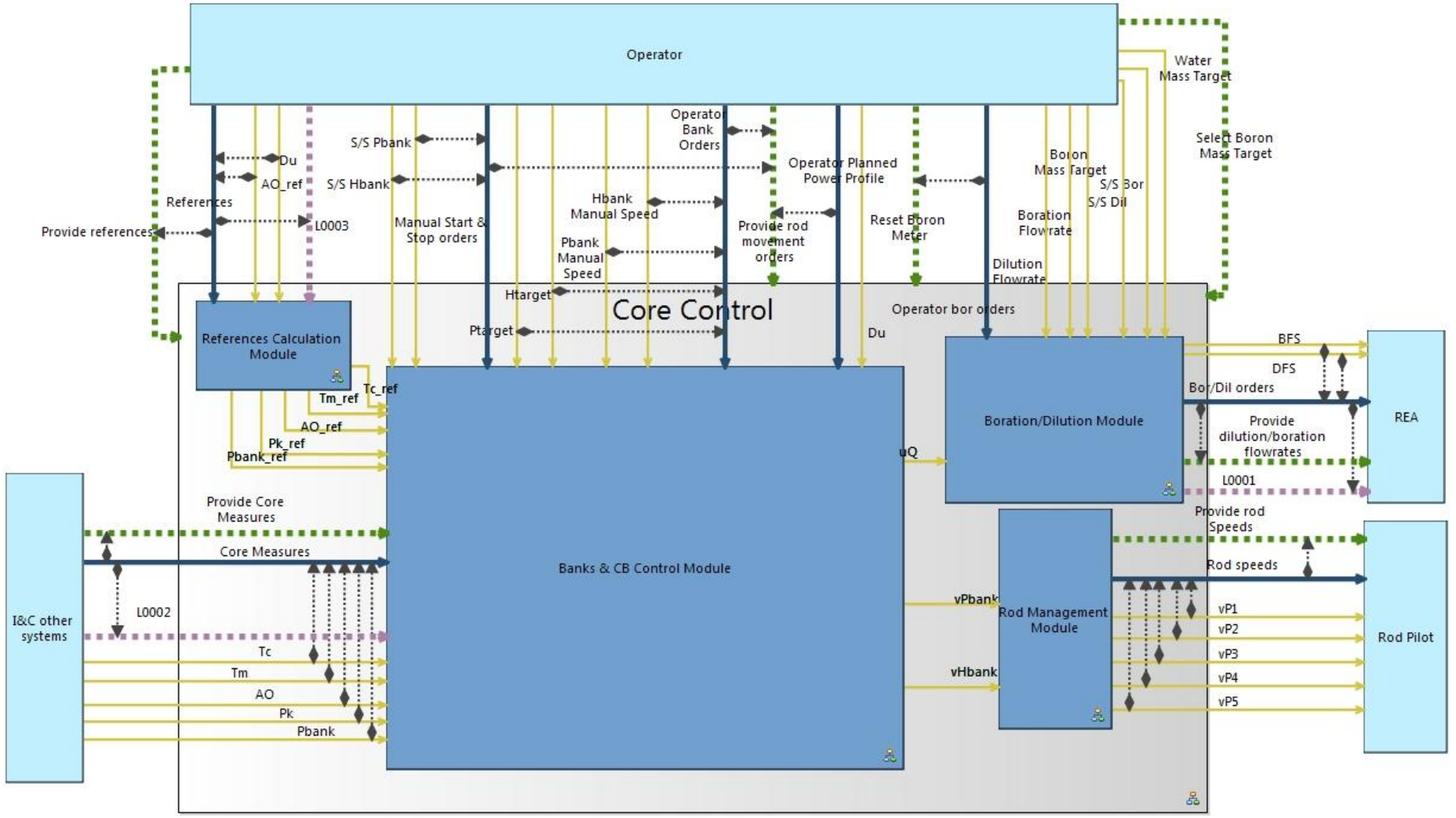


Figure 57: Diagramme de contexte

8.3 Scénarios opérationnels du Core Control

Les activités A6 et A7 mènent à l'établissement de trois scénarios opérationnels définis par les DSML de dépendance et de séquence que nous décrivons ci-dessous.

Le diagramme de séquence défini en partie 6.3 est d'abord utilisé comme illustré en Figure 58. Cet exemple montre les interactions du Core Control avec ses systèmes à l'interface. L'exemple montre un scénario dans lequel l'opérateur doit contrôler le cœur en injectant du bore dans ce dernier. Le diagramme montre que l'opérateur doit d'abord :

- Sélectionner une masse de bore à injecter.
- Ensuite, il met à zéro la mesure de bore injectée (cf. Annexe N).
- Il peut ensuite lancer l'injection.
- Le Core Control envoie alors une commande d'injection au REA.
- Cette commande continue tant que l'opérateur ne l'arrête pas ou alors que la masse injectée est atteinte.

Notons que ce scénario opérationnel raffine l'exigence **R0008** du référentiel d'exigences Figure 55, tel que montré dans l'Annexe L. Le diagramme de dépendance (avec le temps) est ensuite utilisé comme illustré Figure 59. Cet exemple montre la variation de puissance en fonction du temps. Ce signal correspond à un scénario opérationnel de variation de l'entrée ***Du*** précisée sur la figure. Ce diagramme permet de raffiner l'exigence **R0004** donnée dans Figure 55 tel qu'illustré dans l'Annexe L.

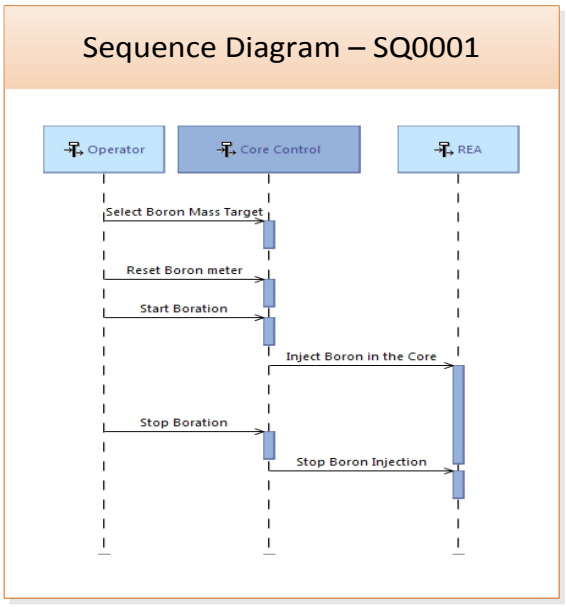


Figure 58: Illustration du diagramme de séquence

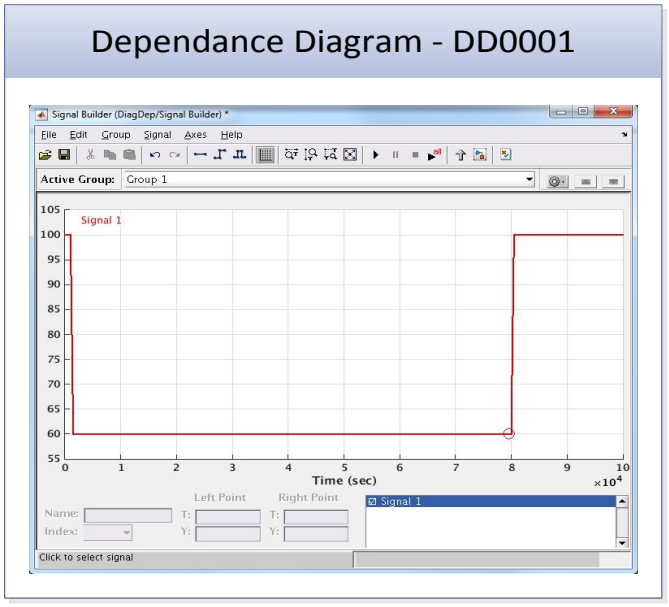


Figure 59: Illustration du diagramme de dépendance (avec le temps)

Un diagramme de dépendance (version sans le temps) est enfin utilisé comme illustré Figure 60. Cet exemple montre la variation de la température moyenne en fonction de la puissance. En se référant à la Figure 57, ces signaux correspondent à l'expression du comportement entrée-sortie entre les signaux ***Du*** (la puissance) et ***Tm_ref*** (la température moyenne de référence) donnés sur la figure. Ce diagramme permet de raffiner l'exigence **R0020** donnée dans Figure 55 tel qu'illustré dans l'Annexe L.

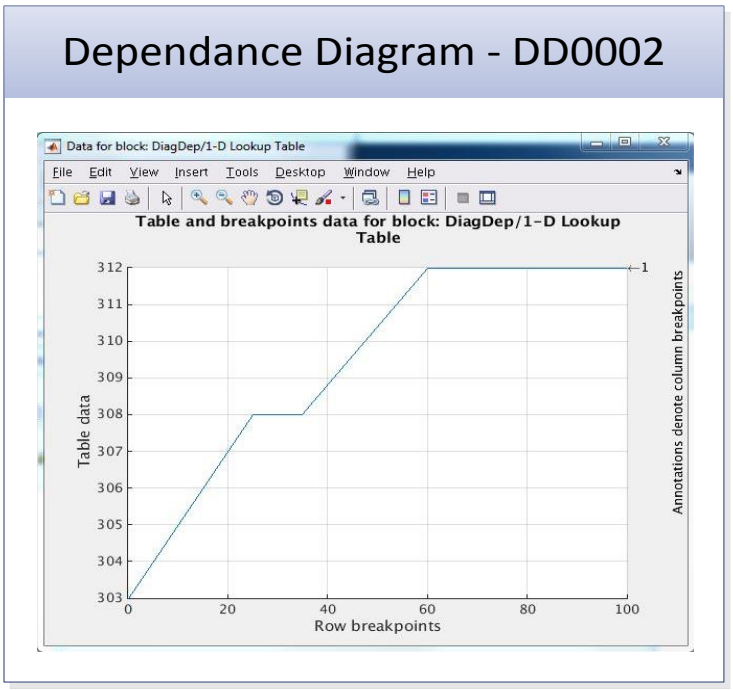


Figure 60: Illustration du diagramme de dépendance (version sans le temps)

8.4 Définition des modes du Core Control

Une partie du diagramme de modes et transition, associé au *Core Control*, est obtenu après application de l'activité A3. Il est donnée Figure 61. On distingue en particulier ici deux modes de fonctionnement :

- Le mode automatique. Dans ce mode le Core Control doit calculer automatiquement (i.e. sans action de l'opérateur) les ordres de commande des actionneurs. Notons dès maintenant que c'est sur ce mode en particulier que nous centrerons notre étude dans la partie Automatique présentée plus loin.
- Le mode manuel. Dans ce mode, c'est l'opérateur qui contrôle le cœur.

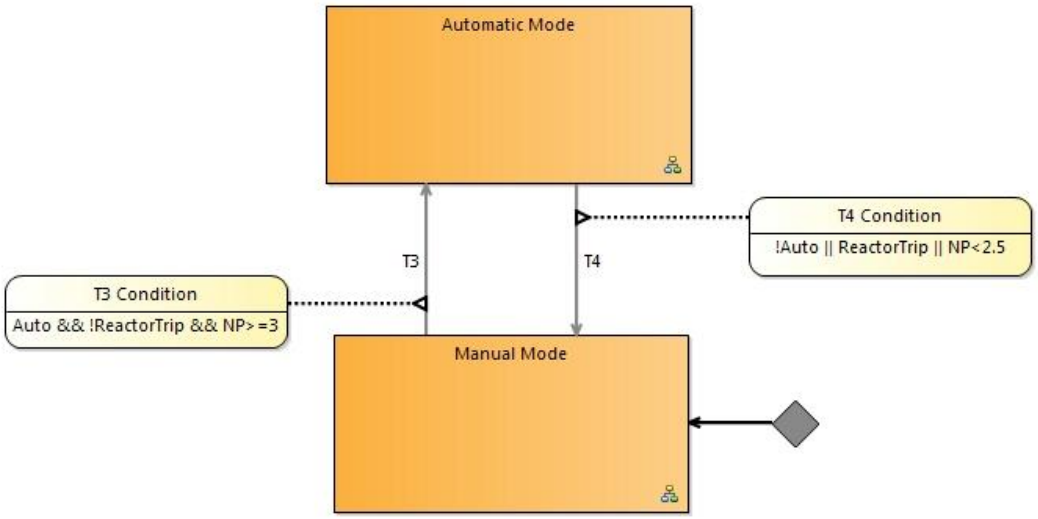


Figure 61 : Diagramme de modes et transition du Core Control

8.5 L'architecture fonctionnelle du Core Control

Par applications de l'activité A4, dans chaque mode, un diagramme fonctionnel est défini, pour décrire alors l'architecture fonctionnelle du SOI dans ce mode. On trouvera en Annexe P l'illustration de l'association des différentes architectures fonctionnelles et des modes.

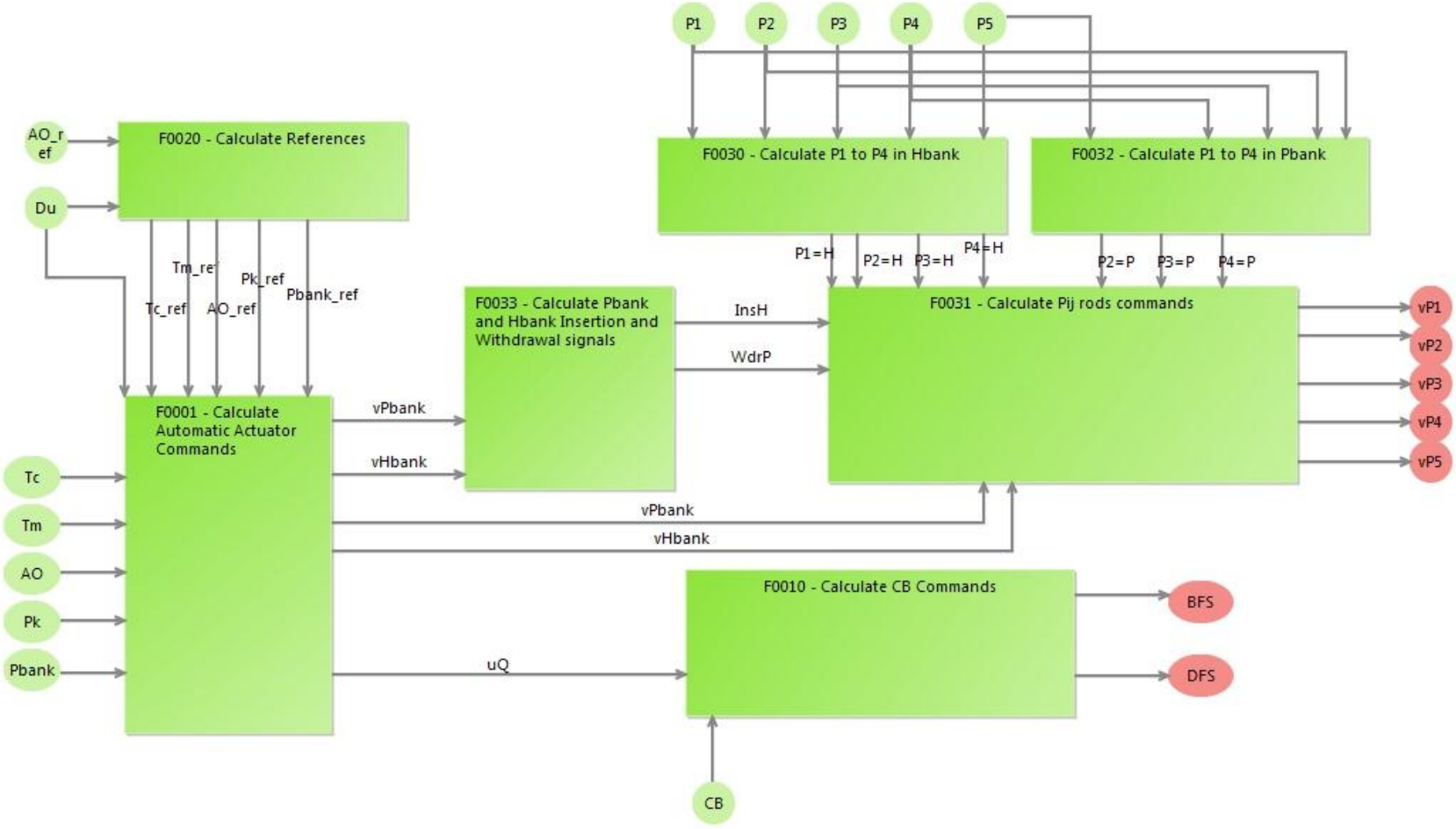


Figure 62: Architecture fonctionnelle du Core Control dans le mode Automatique

La Figure 62 montre le diagramme fonctionnel associé au mode automatique du Core Control (défini dans 8.4). On peut y voir les flux naviguant entre les différentes fonctions. Prenons par exemple la fonction F0001 (« Calculate Automatic Actuator Commands »). Cette fonction prend en entrées les mesures du processus et calcule trois sorties :

- La vitesse de déplacement du groupe Pbank de grappes notée vP_{bank}
- La vitesse de déplacement du groupe Hbank de grappes notée vH_{bank}
- Le débit de dilution/borication noté u_Q

Les autres fonctions sont définies comme suit :

- F0010 : calcule les commandes à envoyer aux actionneurs. Elle traduit la commande de borication/dilution (en ppm/s) en un débit d'acide borique (kg/s) et en débit d'eau (kg/s). Le traitement de cette fonction est donné dans le diagramme de spécification fonctionnelle en Annexe P.
- F0020 : cette fonction calcule les références pour le contrôle.
- Les fonctions F0030, F0031, F0032 et F0033 : traduisent les vitesses des groupes Pbank et Hbank en vitesses des groupes P1 à P5.

Le diagramme fonctionnel associé au mode manuel est donné en Annexe O. On y voit que certaines fonctions utilisées dans le mode automatique sont également utilisées dans le mode manuel (e.g. fonctions F0030, F0031, F0032 et F0033) et que d'autres ne le sont pas (e.g. F0001).

8.6 Le diagramme de spécification

Par application de l'activité A5, le traitement entrée-sortie de chaque fonction est donné dans un diagramme de spécification fonctionnelle associé à cette fonction. Le langage utilisé est celui utilisé par Framatome et donné en Annexe A.

Prenons la fonction F0001 définie dans la partie précédente 8.5. La Figure 63 présente le diagramme de spécification fonctionnelle associé à cette fonction dont elle définit le traitement entrée-sortie.

On peut voir deux blocs principaux :

- Multi-objective Gain-Scheduled Controller (LPV System). Ce bloc est un système LPV prenant en paramètre de séquencement la position des barres Pbank. Il est constitué d'un ensemble de systèmes dynamiques décrits par des équations d'état telle que :

$$\begin{cases} \dot{x}(t) = A(\pi)x(t) + B(\pi)u(t) \\ y(t) = C(\pi)x(t) + D(\pi)u(t) \end{cases}$$

Où π est le paramètre de séquencement, c'est-à-dire le paramètre faisant varier les matrices A, B, C et D. Ce bloc n'est actuellement pas défini dans le langage de spécification fonctionnelle de Framatome.

La partie Automatique aura pour but de faire l'ingénierie complète de ce bloc. De plus, nous montrerons comment certaines exigences peuvent être tracées et allouées à ce bloc.

- Le MPC Controller (Model Predictive Controller). Ce bloc constitue un algorithme de commande prédictive. Pour faire simple, cet algorithme a la capacité de calculer les ordres optimaux sur la base d'un modèle du système. Ce bloc n'est également pas défini dans le langage de spécification fonctionnelle de Framatome (car innovant dans ce contexte).

De la même manière que pour le Multi-objective Gain-Scheduled Controller la partie Automatique aura pour objectif de faire l'ingénierie de ce bloc, i.e. la définition de l'algorithme développé et le modèle utilisé.

En ce qui concerne les autres fonctions (e.g. F0002, F0003, F0004 et F005 et certaines autres), nous avons fourni dans les Annexe N les diagrammes de spécification fonctionnelle associées.

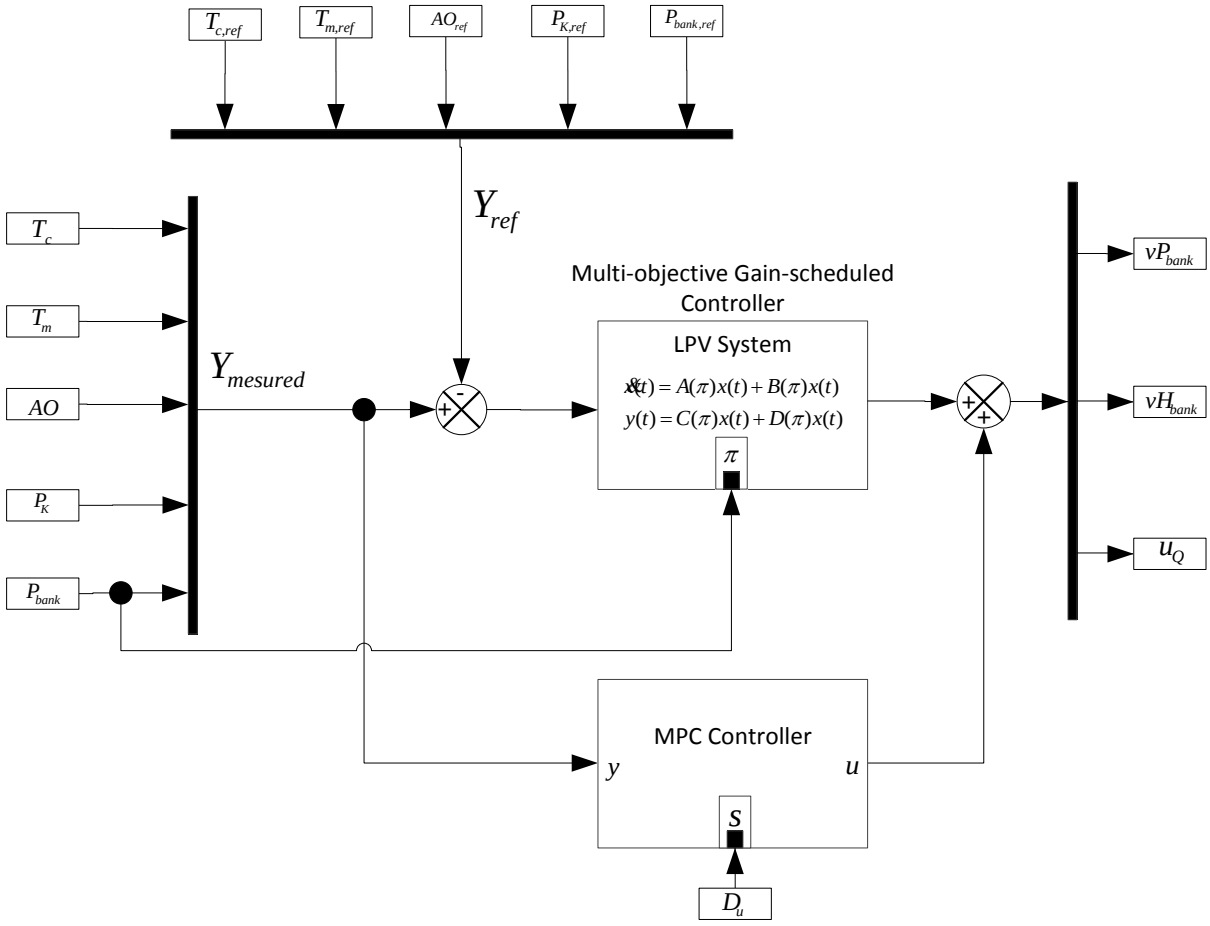


Figure 63: Diagramme de spécification fonctionnelle de la fonction F0001

8.7 Liens entre les diagrammes

Nous présentons ici les liens entre les modèles. Ces liens démontrent la traçabilité des exigences sur les architectures :

- L' Annexe K montre les liens menant du SOI aux traitements entrée-sortie de ses fonctions. La figure synthétise les liens entre le SOI, le diagramme des modes associé, les diagrammes fonctionnels associés à chacun des modes et les diagrammes de spécification fonctionnelle associés à chaque fonction.
- L'Annexe L montre des liens formalisés entre les exigences et différents éléments de modélisation : les scénarios opérationnels (diagramme de séquence et de dépendance), le SOI, les modes ou encore les fonctions.
- L'Annexe M met le référentiel d'exigences au centre des différents diagrammes définis précédemment, synthétise et illustre les liens formalisé dans 5.3.6.

9 Passage à l'ingénierie de métier : l'automatique

Après avoir introduit les différents constituants de la méthode, on précise ici, en guise de conclusion, comment les contributions des travaux présentés précédemment nous amènent maintenant à promouvoir des contributions en automatique de manière cohérente et complémentaire pour couvrir les besoins de Framatome.

Dans le chapitre 8 nous avons présenté le cas d'application de la méthode d'IS du *Core Control*. Au cours de ce cas d'utilisation, nous avons présenté en 8.6 le diagramme de spécification fonctionnelle (cf. Figure 63) de la fonction F0001. Ce diagramme montre deux blocs : le régulateur multi-objectif et le superviseur, deux blocs n'étant pas définis dans le DSML de spécification de Framatome (cf. Annexe A). Nous allons alors dans Partie III, de spécifier chacun de ces blocs, i.e. en quoi ils consistent et leurs rôles respectifs dans l'asservissement du cœur. Nous répondons aux questions suivantes :

- Quelles exigences leurs sont allouées ? Nous allouerons les différentes exigences du cas d'utilisation définies dans 8.1 aux deux blocs. Ces exigences sont formulées dans le cahier des charges de l'asservissement.
- Comment sont-elles traduites dans le langage de l'automatique, c'est-à-dire l'illustration de la reformulation des exigences ? Or, le problème de commande étant formalisé sous la forme d'un problème d'optimisation sous-contraintes, il possède, de fait, des propriétés axiomatiques (cf. 5.10) **garantissant** que les contraintes sont **vérifiées** (si il existe une solution au problème d'optimisation). Les exigences étant traduites sous forme de contraintes, on les vérifie formellement.
- Comment les blocs sont-ils paramétrés ? Nous montrerons comment ces blocs peuvent être paramétrés et réglés.
- Comment les exigences sont-elles vérifiées en simulation ? En effet, les blocs proposés sont simulables. La Partie III propose un modèle simplifié du réacteur permettant de simuler son comportement. Le régulateur résultant de l'étude sera donc simulé sur la base de ce modèle et les exigences seront vérifiées.

Le but est de fournir à Framatome un environnement complet et paramétrable, constitué des deux blocs sur lesquels les exigences sont tracées. De plus nous fournirons les éléments de justification de la conception par l'ensemble de la formulation du problème de commande.

Partie III.
Contrôle-Commande du Cœur des REP : une Conception Revisitée

10 Stratégie de commande

Sommaire

10.1 Description du système à contrôler 171
10.2 Cahier des charges de la régulation 173
10.2.1 Limites de conditions d'opérations (LCO) 174
10.2.2 La performance 175
10.2.3 Sollicitations des actionneurs 175
10.2.4 Robustesse 176
10.2.5 Priorité des sorties à contrôler 176
10.2.6 Définitions des références pour le contrôle 176
10.3 Choix de la stratégie de commande employée 178
10.3.1 Commande simplifiée : régulation par les groupes de grappes 178
10.3.2 Commande avec variation de débit 180
10.3.3 Commande retenue dans la thèse : commande hiérarchisée à deux niveaux 181
10.4 Présentation de la démarche de conception et de paramétrage du régulateur 189

Ce chapitre introduit l'ensemble des travaux de la Partie III et définit la stratégie de commande employée dans la thèse. La première partie 10.1 présente la description du système que l'on cherche à contrôler, contenant les entrées de commande et sorties à réguler considérées. Ensuite nous présentons le cahier des charges contenant les différentes exigences que le système de contrôle commande doit respecter 10.2. Dans un troisième temps nous présentons la méthodologie employée dans la thèse pour répondre à ces exigences.

10.1 Description du système à contrôler

Le système à contrôler est composé du cœur, des branches chaudes et froides ainsi que du générateur de vapeur (GV). On appelle ce système le **réacteur**. On appelle **Circuit Primaire** le système constitué du GV et des branches chaude et froide. Ces éléments sont représentés sur la Figure 64.

Le réacteur possède trois actionneurs (cf. Figure 64) :

- Le groupe de barres Pbank (bloc P sur la Figure 64 → signal de sortie du bloc).
  - Commandé en vitesse notée vP_{bank}
- Le groupe de barres Hbank (bloc H).
  - Commandé en vitesse notée vH_{bank}

- Le débit de bore (bloc Qb).
  - Commandé en débit noté u_Q

Les mesures accessibles sur le système sont :

- T_c la température branche chaude à la sortie du cœur (bloc Tc).
- T_f la température branche froide à l'entrée du cœur (bloc Tf).
- AO la distribution axiale de puissance (bloc AO).
- P_K la puissance du cœur (bloc Pk).
- P_{bank} la position des barres Pbank (bloc P → signal de sortie du bloc).

Une entrée de perturbation correspondant à la puissance extraite par la turbine :

- P_{turb} la puissance turbine (bloc Pturb).

Les sorties à contrôler :

- T_m la température moyenne du cœur (moyenne entre T_c et T_f).
- AO la distribution axiale de puissance.
- P_{bank} la position du groupe Pbank (cf. contrôle de Pmax 3.2.4).

Le contrôleur montré sur la Figure 64 représente le système de régulation sensé contrôler le système décrit précédemment.

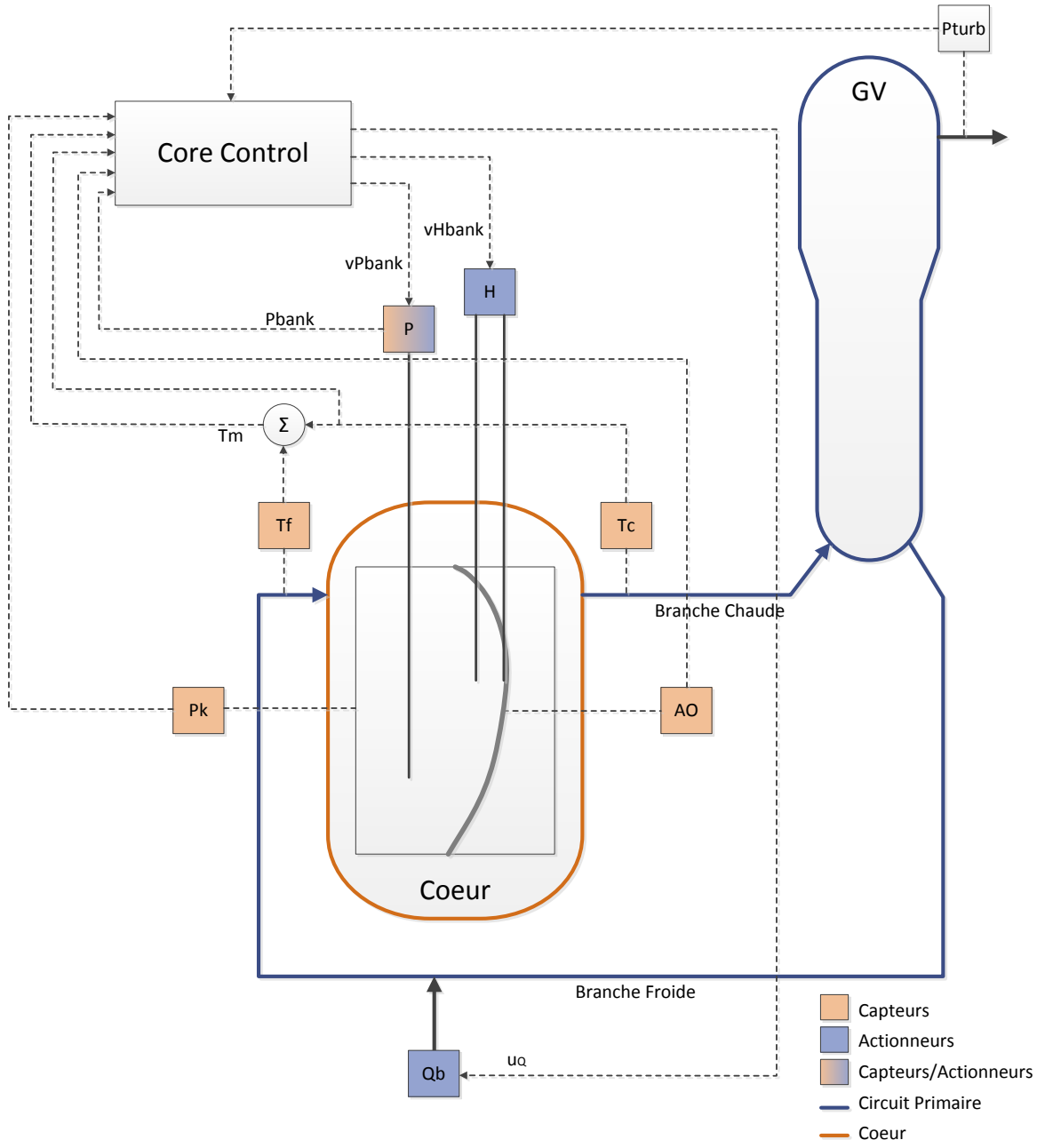


Figure 64: Schéma de principe de la régulation

10.2 Cahier des charges de la régulation

Dans cette partie nous présentons le cahier des charges du *Core Control*. Ce cahier des charges provient des exigences définies lors du cas d'application au *Core Control* de la méthode d'IS en partie 8.1.

10.2.1 Limites de conditions d'opérations (LCO)

Le *Core Control* est contraint par un domaine d'opération autorisé dans lequel les déviations des sorties à contrôler par rapport à leurs références doivent être maintenues. Ce domaine est défini par les limites opérationnelles de fonctionnement (Limiting Condition Operation – LCO) que schématise la Figure 65, c'est à dire les limites hautes et basses pour chaque paramètre de contrôle (à savoir T_m, AO et P_{bank}) par rapport à leurs références.

On définit ces références par :

- $T_{m,ref}$ la référence de température moyenne
- AO_{ref} la référence d'AO
- $P_{bank,ref}$ la référence de position du groupe Pbank

L'écart des paramètres à contrôler par rapport à leurs références est donné par :

- ΔT_m l'écart de température moyenne à sa référence.
- ΔAO l'écart d'AO à sa référence.
- ΔP_{bank} l'écart de position de Pbank à sa référence.

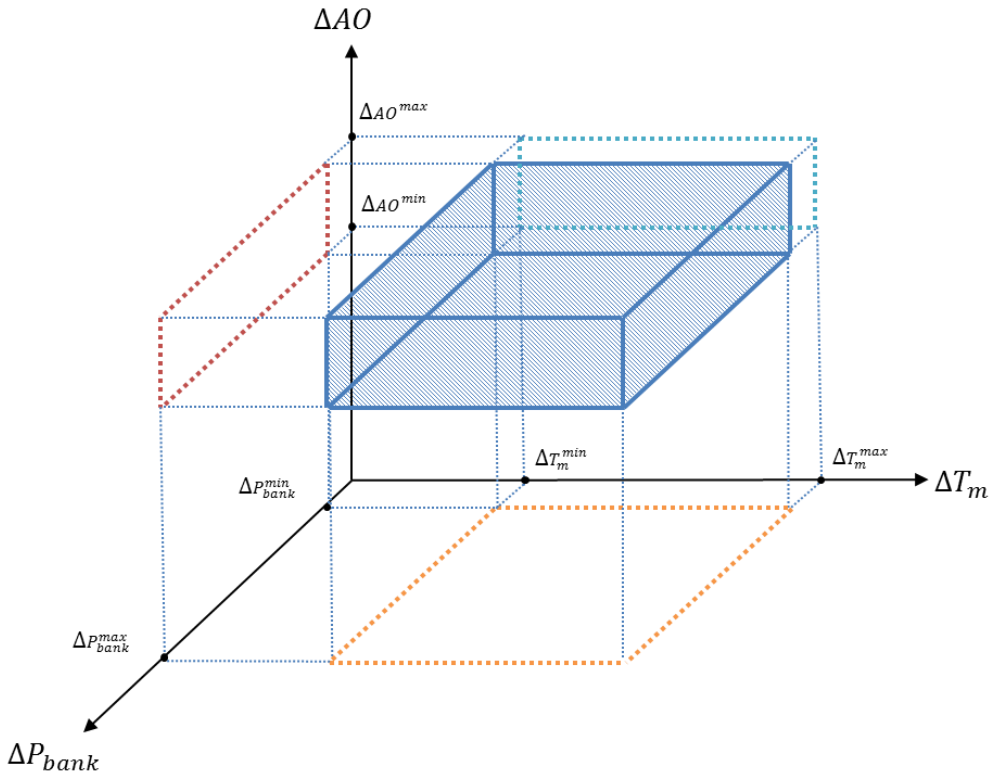


Figure 65: Domaine de fonctionnement autorisé

On définit alors les limites du domaine d'opération telles que :

- $\Delta T_m \quad \in [-\Delta T_m^{max} \ , \ \Delta T_m^{max}]$
- $\Delta AO \quad \in [-\Delta AO^{max} \ , \ \Delta AO^{max}]$
- $\Delta P_{bank} \quad \in [-\Delta P_{bank}^{max} \ , \ \Delta P_{bank}^{max}]$

En pratique, les valeurs données pour ces paramètres sont :

$$\Delta T_m^{max} = 1.5\ °C$$

$$\Delta AO^{max} = 5\ \%AO$$

$$\Delta P_{bank}^{max} = 30\ pas$$

En résumé, ces données sont les déviations maximales que peuvent atteindre la température moyenne, l'axial offset et la position du groupe Pbank.

10.2.2 La performance

Le *Core Control* doit permettre de réaliser les exigences de flexibilité définies ci-dessous :

- Adaptation à la variation de demande journalière (suivi de charge)
  - Rampes de charge : $\pm 5\%/min$ PN entre 25% et 100% de PN
- Adaptation à la variation de demande en temps réel (réglage de fréquence)
  - Primaire : $\pm 5\%$ PN à $1\%/s$
  - Secondaire : $\pm 5\%$ PN à $1\%/min$
- Adaptation aux perturbations réseau (réserve tournante)
  - Echelon : $\pm\ 10\%$ PN entre 30% et 100% PN

Le but du *Core Control* est de maintenir les sorties à contrôler dans le domaine d'opération autorisé qui est défini dans la partie 3.2.4 précédente, quelles que soient les variations de puissance spécifiées ci-dessus.

10.2.3 Sollicitations des actionneurs

Les sollicitations maximales autorisées sont définies de la manière suivante :

- **Saturation des grappes de contrôles en positions :**
  - Position minimale : $P_{barres}^{min} =$ 0 pas extraits
  - Position maximale : $P_{barres}^{max} =$ 410 pas extraits
  - Vitesse minimale : $V_{barres}^{min} =$ 4 pas/min

- Vitesse maximale : $V_{barres}^{max} =$ 75 pas/min

- **Débit maximums et minimums d'injection de borication/dilution :**
  - Débit min de bore : $Q_{bor}^{min} =$ 1 kg/s
  - Débit max de bore : $Q_{bor}^{max} =$ 3 kg/s
  - Débit min d'eau : $Q_{dil}^{min} =$ 1 kg/s
  - Débit max d'eau : $Q_{dil}^{max} =$ 10 kg/s
  - Concentration min : $C_b^{min} =$ 0 ppm

10.2.4 Robustesse

Le régulateur doit posséder des garanties de robustesse :

- Marge de module minimale : $M_m = 0.5$

10.2.5 Priorité des sorties à contrôler

La température moyenne étant un paramètre primordial pour le fonctionnement il est priorisé comparativement à l'AO et la position du groupe Pbank.

10.2.6 Définitions des références pour le contrôle

Les références des sorties à contrôler dépendent du fonctionnement du cœur. Elles sont définies de la manière suivante :

- **La température de référence $T_{m,ref}$**

Le profil de température moyenne est défini par le diagramme de fonctionnement, défini dans 3.2.4. Il est rappelé Figure 66.

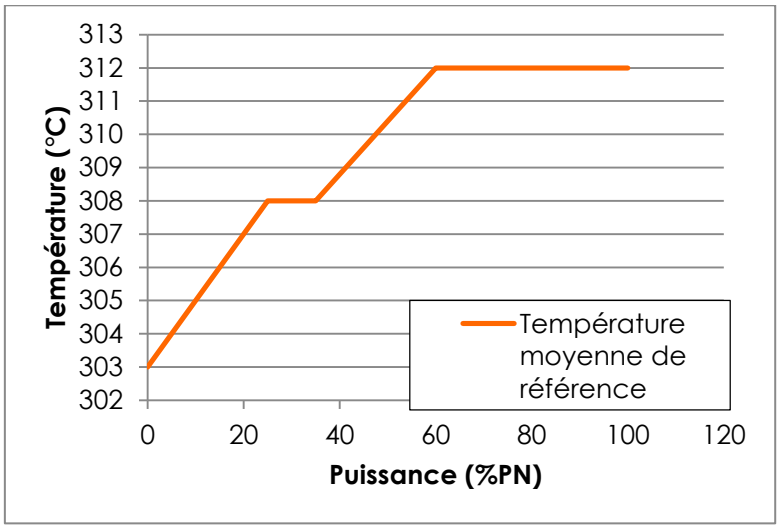


Figure 66: Profil de la température moyenne de référence

- **La distribution axiale de puissance : AO_{ref}**

L'AO de référence est remis à jour tous les mois pour prendre en compte l'épuisement du cœur. Il est directement fourni par l'utilisateur et est considéré constant dans notre étude.

- **La position de référence du groupe Pbank : $P_{bank,ref}$**

La position de référence du groupe Pbank est donnée par un profil d'insertion du groupe fonction de la puissance. Elle est représentée sur la Figure 66. Cette position de référence est donnée en position cumulée (cf. 3.2.4).

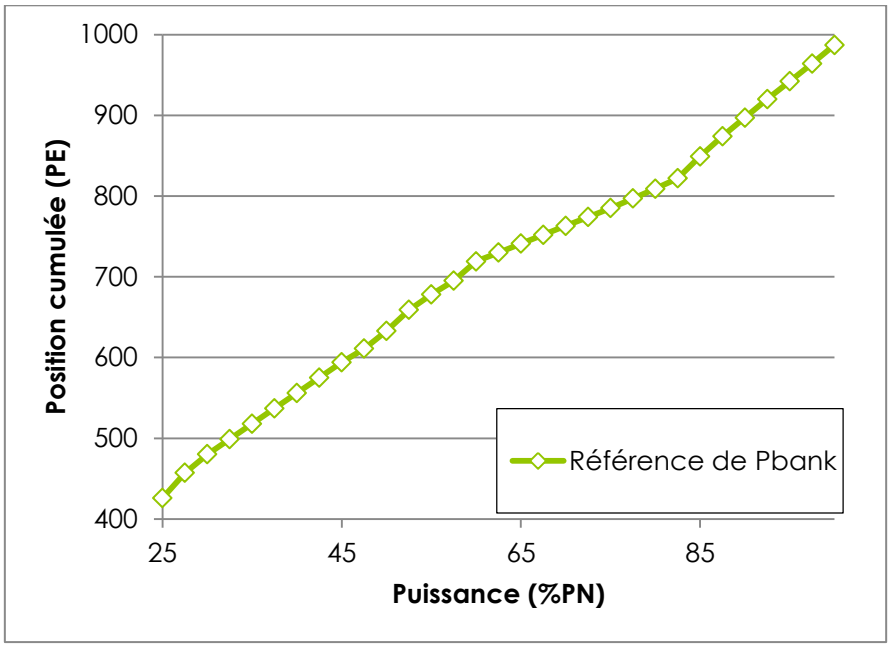


Figure 67: Position de référence pour Pbank (cumulée)

10.3 Choix de la stratégie de commande employée

Le but de cette partie est de présenter le cheminement logique des travaux, partant des articles jusqu'à la stratégie de commande entreprise dans le présent manuscrit. La Figure 68 résume les différentes versions du contrôle que nous avons envisagées dans la thèse.

10.3.1 Commande simplifiée : régulation par les groupes de grappes

Dans un premier temps nous nous sommes inspirés des travaux de [Bendotti *et al.*, 1994] qui définit un régulateur H_∞ pour contrôler la température moyenne et l'AO avec deux groupes de barres. Cette étude, proposant l'utilisation d'un régulateur statique, a montré qu'un régulateur à gains séquencés sur la puissance du cœur était nécessaire pour contrôler le processus sur tout le domaine de fonctionnement considéré.

Par une approche similaire et par souci de simplicité, la première phase de la thèse consista à concevoir un régulateur H_2/H_∞ permettant de contrôler la température moyenne et l'AO considérant seulement les groupes de grappes Pbank et Hbank comme actionneurs.

Les hypothèses que nous considérions à ce moment étaient les suivantes :

- L'effet du Xénon est supposé suffisamment lent pour être compensé par le bore.
- Le contrôle de la position de Pbank n'est pas considéré.

Utilisant un modèle non-linéaire du réacteur (que nous présenterons dans 11.4), nous avons adopté une approche **multi-modèles** consistant à linéariser ce modèle autour de points de fonctionnement **paramétrisés par la puissance du cœur** pour synthétiser un régulateur autour de chaque point de fonctionnement. Ces régulateurs sont ensuite interpolés pour former un régulateur à **gains séquencés**.

Nous avons opté pour une approche par optimisation non-lisse pour la synthèse d'un régulateur multi-objectif H_2 sous contraintes H_∞, nous permettant de considérer un régulateur structuré.

La structure du régulateur choisie consiste en un régulateur constitué d'un retour statique de sorties et d'intégrales sur les grandeurs à contrôler. Par la suite nous appellerons cette structure un **PI multivariables** définit comme suit :

$$u_K = K_P y + K_I \int z$$

Avec u_K le vecteur des commandes, y le vecteur des sorties mesurées, z le vecteur des grandeurs à contrôler, K_P le retour statique filtré et K_I le gain sur l'intégrale des grandeurs à contrôler. L'utilisation d'un PI est appropriée, dans notre cas, car le système ne possède pas de dynamiques oscillatoires. L'amortissement le plus faible relevé, parmi l'ensemble des modèles linéarisés, est de $\xi_{min} = 0.781$.

Ces travaux nous ont mené à la publication de l'article [Lemazurier *et al.*, 2017a]. Cet article montre la faisabilité du contrôle du cœur par un régulateur tel que défini précédemment, sur un domaine de fonctionnement relativement étendu : 60%PN à 90%PN. L'ajout de l'actionneur bore étant indispensable dans le domaine 90%PN à 100%PN, ce domaine n'a pas été abordé.

De plus cet article montre qu'un régulateur LTI (i.e. qui ne possède pas de gains variables) fournit des performances certes moins bonnes que celui à gains séquencés, mais tout à fait acceptables sur le même domaine de fonctionnement. C'est ce qui a motivé l'utilisation d'un régulateur PI multivariables LTI pour le second article [LEMAZURIER 2018] que nous présenterons dans la partie suivante.

10.3.2 Commande avec variation de débit

Une problématique adressée en premier lieu à la thèse était d'étudier l'utilisation du débit primaire pour le contrôle du cœur. Ces travaux ont abouti à la publication d'un second article [Lemazurier *et al.*, 2018a].

Cet article aborde la problématique du contrôle des températures branches chaudes (T_c) et froide (T_f), tout en conservant le contrôle de l'AO par les actionneurs : groupe de barres Pbank et Hbank ainsi que la **variation de débit primaire** (Q_p cf. 10.1).

Utilisant les conclusions de l'article précédent nous avons conçu un régulateur PI multivariable LTI robuste sur la plage de fonctionnement considérée. De la même manière que l'article précédent, le problème de commande est un problème multi-objectif résolu par une méthode d'optimisation non-lisse.

L'étude conclût sur la faisabilité d'un tel régulateur à gains fixes (PI multivariables LTI) robuste sur un domaine de fonctionnement relativement grand et testé sur le code de simulation réaliste de Framatome.

Une fois de plus, le contrôle sur un domaine complet (100%PN à 25%PN) implique l'utilisation d'un actionneur additionnel : le bore.

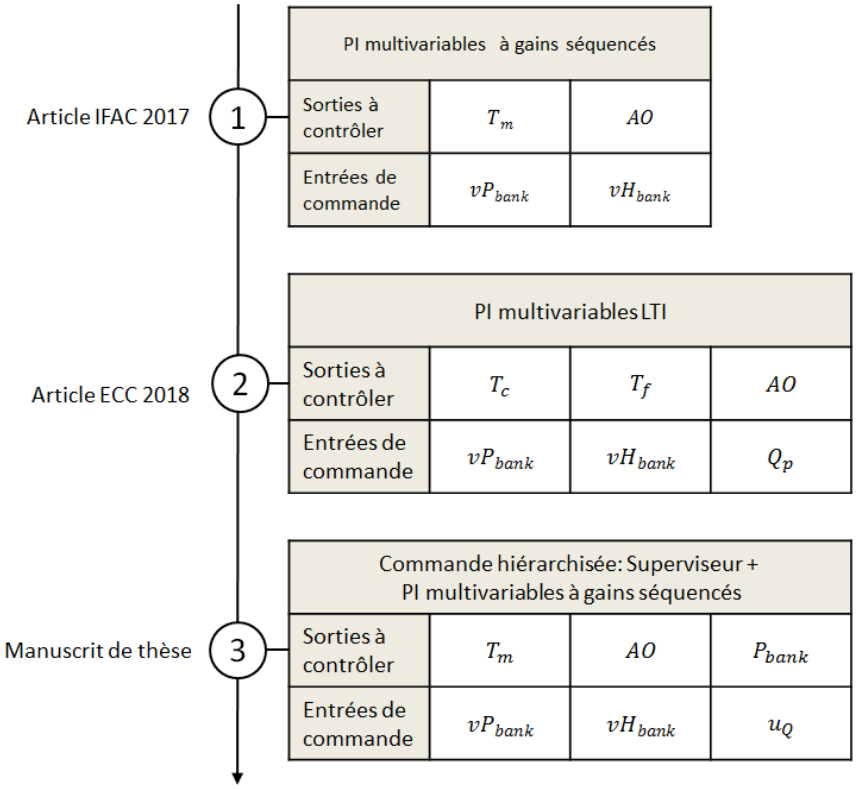


Figure 68: Organigramme des travaux

10.3.3 Commande retenue dans la thèse : commande hiérarchisée à deux niveaux

Dans cette partie nous abordons la stratégie de commande adoptée dans le manuscrit de thèse. Nous commençons par formuler les différentes problématiques liées au problème de commande considéré. Dans un deuxième temps nous décrivons et justifions les choix pour la stratégie de commande. Enfin, nous précisons les éléments qui la composent.

10.3.3.1 Spécificités du problème de commande

Afin d'être en mesure de définir la stratégie de commande, les principales spécificités du problème de commande sont présentées ci-dessous.

Les dynamiques du cœur

Le réacteur possède des dynamiques disparates, i.e. à la fois **lentes et rapides**. La dynamique liée au xénon est très lente (de l'ordre de l'heure) alors que celles liées à puissance et la température sont plutôt rapides (de l'ordre de la dizaine de secondes).

Un comportement non-linéaire important sur le domaine de fonctionnement complet

Sur l'ensemble du domaine de fonctionnement le comportement du cœur est hautement non-linéaire, principalement en raison de l'insertion des groupes de barres dans le cœur. Entre la puissance maximal (100%PN) et une puissance intermédiaire (e.g. 60%PN) l'effet des groupes de grappes sur les différentes sorties à contrôler changent fortement. On observe même une inversion des effets des actionneurs dans certaines zones du cœur.

Retard d'injection de bore

L'actionneur de bore considéré à présent dans le problème de commande possède un retard important : 300 secondes. Bien que le système que l'on cherche à contrôler soit relativement lent (ordre de grandeur global : 10 secondes), un retard de cet ampleur est dimensionnant pour la régulation.

Contraintes de commande

Le problème de commande tel que présenté dans 10.2 contient un grand nombre de contraintes, dont des contraintes temporelles qu'il est souvent difficile de prendre en compte par des techniques de commande traditionnelles.

Utilisation du programme de suivi de charge

Actuellement, chaque unité de production d'électricité reçoit un programme de variation de charge journalier. Ce programme est donc connu à l'avance. Or, on ne profite actuellement pas de ce signal pour prédire des commandes futures. Dans la thèse, nous supposons qu'il est autorisé de considérer les **variations futures** de ce signal pour **l'élaboration de la commande**.

10.3.3.2 Structure de la commande retenue

Pour répondre aux difficultés précisées précédemment (cf. 10.3.3.1), nous adoptons une stratégie de commande hiérarchisée [Claveau, 2005]. Cette stratégie consiste à diviser la complexité d'une loi de commande en la décomposant en plusieurs niveaux hiérarchiques. Un schéma classique de décomposition hiérarchique tiré de la thèse [Claveau, 2005] est donné Figure 69.

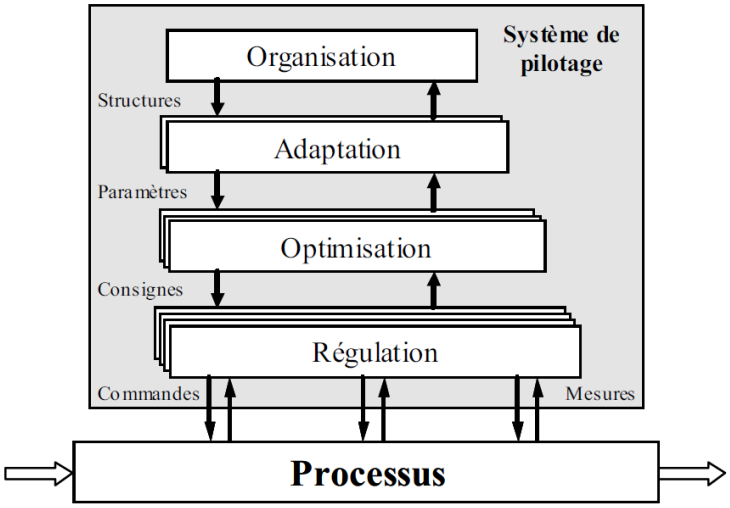


Figure 69: Commande hiérarchisée

Dans notre cas, nous décidons d'adopter les deux derniers niveaux hiérarchiques, soit :

- **La régulation (Régulateur multi-objectif)** : ce niveau agit directement sur le processus. Son rôle est de générer les commandes pour que les sorties à contrôler du processus suivent les trajectoires. Il correspond au régulateur multi-objectif à gains séquencés présenté ci-dessous en partie 10.3.3.3). On le nommera dans la suite le Régulateur Multi-Objectif.
- **L'optimisation (Superviseur)** : ce niveau précise les trajectoires de référence (y compris les commandes de référence) au niveau de la régulation. Ces points sont en général déterminés de façon à optimiser un certain critère sur la base d'un modèle dynamique ou statique. Nous utiliserons pour cet étage la théorie de la commande prédictive. On le nommera dans la suite le Superviseur.

Pour plus de détails sur les autres niveaux hiérarchiques nous invitons le lecteur à se référer à la thèse [Claveau, 2005].

Ces deux niveaux hiérarchiques sont représentés sur la Figure 70 et nous les décrirons plus précisément dans les parties suivantes (10.3.3.4 et 10.3.3.3).

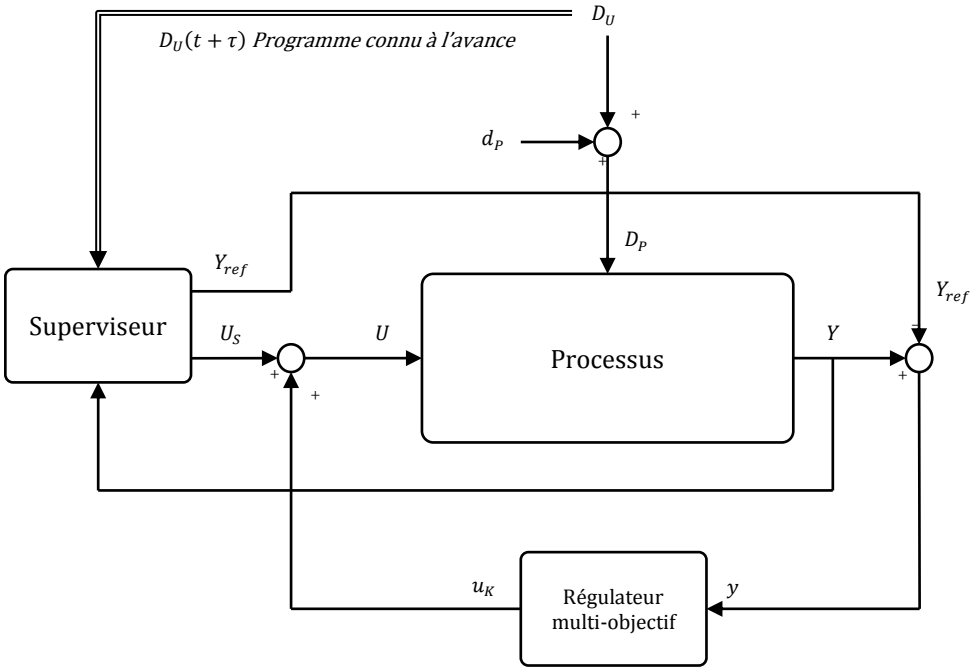


Figure 70: Schéma de principe de la régulation adoptée

Où :

- Processus, le système réel cœur + circuit primaire
- D_u le programme de suivi de charge (connu à l'avance).
- d_P, le signal de perturbation turbine. Ce signal correspond à toutes les perturbations turbine : réglage primaire, secondaire, échelon de réserve de charge.
- D_P désigne la puissance réelle de la turbine

Selon les deux derniers points, nous considérons donc que la puissance turbine peut se décomposer en une composante déterministe connue à l'avance, associée au profil de charge utilisateur et une composante stochastique associée à toutes les perturbations non prévisibles tel que :

$$D_P = D_u + d_P \tag{10.1}$$

- F, la fonction définissant les références pour les régulateurs.

Nous faisons un choix structurant ici, celui de déterminer les références pour le contrôle (cf. 10.2.5) à partir de la puissance de consigne D_u, c'est-à-dire celle donnée par l'opérateur, tel que :

$$Y_{ref} = F(D_u) \tag{10.2}$$

Les **références ne seront donc pas modifiées par les variations de puissance** considérées aléatoires, i.e. données par d_P.

- Y_{ref}, le vecteur des sorties de référence tel que :

$$Y_{ref} = \begin{pmatrix} T_{c,ref} & T_{m,ref} & AO_{ref} & P_{K_{ref}} & P_{bank,ref} \end{pmatrix}^T \quad (10.3)$$

Où les références prises à l'équilibre par les paramètres de contrôle de la tranche variant en fonction du point de fonctionnement du système (cf. 10.2.5) sont :

  - $T_{c,ref}$, la température branche chaude de référence. Elle suit le diagramme de fonctionnement.
  - $T_{m,ref}$, la température moyenne de référence. Elle suit aussi le diagramme de fonctionnement.
  - AO_{ref}, l'Axial Offset de référence . Elle est donnée par l'opérateur.
  - $P_{K_{ref}}$, la puissance cœur de référence . Elle est considérée égale à la puissance turbine. En réalité, à l'équilibre, la puissance cœur est légèrement différente de la puissance turbine pour des raisons de rendement. Pour notre étude on fait donc l'hypothèse $n_{ref} = D_u$.
  - $P_{bank,ref}$, la position du groupe Pbank de référence. Elle est donnée par le profil d'insertion en fonction de la puissance.
- U_S, les commandes fournies par le superviseur. Ce sont les commandes de références pour les actionneurs.
- u_K, les commandes données par le régulateur multi-objectif.
- U, les commandes données aux actionneurs telles que :

$$U = U_S + u_K \quad (10.4)$$

- Y, les sorties du système réel.

$$Y = (T_c \quad T_m \quad AO \quad P_K \quad P_{bank})^T \quad (10.5)$$

- y, l'écart des sorties réelles par rapport aux références telles que :

$$y = Y - Y_{ref} \quad (10.6)$$

avec
$$y = (\delta T_c \quad \delta T_m \quad \delta AO \quad \delta P_K \quad \delta P_{bank})^T \quad (10.7)$$

10.3.3.3 Le régulateur multi-objectif

A l'instar de [Lemazurier *et al.*, 2017a], nous utiliserons un régulateur multi-objectif à gains séquencés. Il répond aux problématiques suivantes :

- *Assurer le contrôle rapproché du système en assurant de bonnes performances à priori autour de chaque point de fonctionnement. En particulier pour le rejet des perturbations liées au réglage de fréquence.*
- *Contrôler le réacteur sur l'ensemble du domaine de fonctionnement grâce à l'adaptation des gains qui le composent au fur et à mesure du fonctionnement.*
- *Garantir la robustesse (approche multi-objectif), localement, autour des points de fonctionnement.*
- *Prendre en compte un grand nombre de contraintes de commande, imposées dans le cahier des charges 10.2.*

Cependant, pour des systèmes présentant d'importantes non-linéarités comme c'est le cas ici, un régulateur multi-objectif à gains séquencés peut montrer des performances médiocres. En effet, il est synthétisé à chaque point de fonctionnement sur la base d'un modèle linéarisé. Or l'utilisation d'un modèle linéarisé peut manquer de représentativité du modèle non-linéaire global. L'inconvénient est que les trajectoires empruntées par ce régulateur peuvent alors être loin de la trajectoire optimale du comportement global. La Figure 71 compare la trajectoire empruntée par un régulateur multi-objectif (trait pointillés) à celle empruntée par un algorithme de commande prédictive (trait plein). Elle illustre le fait que le régulateur multi-objectif ne prend pas en compte de contraintes globales pour le suivi des trajectoires comme le ferait la commande prédictive.

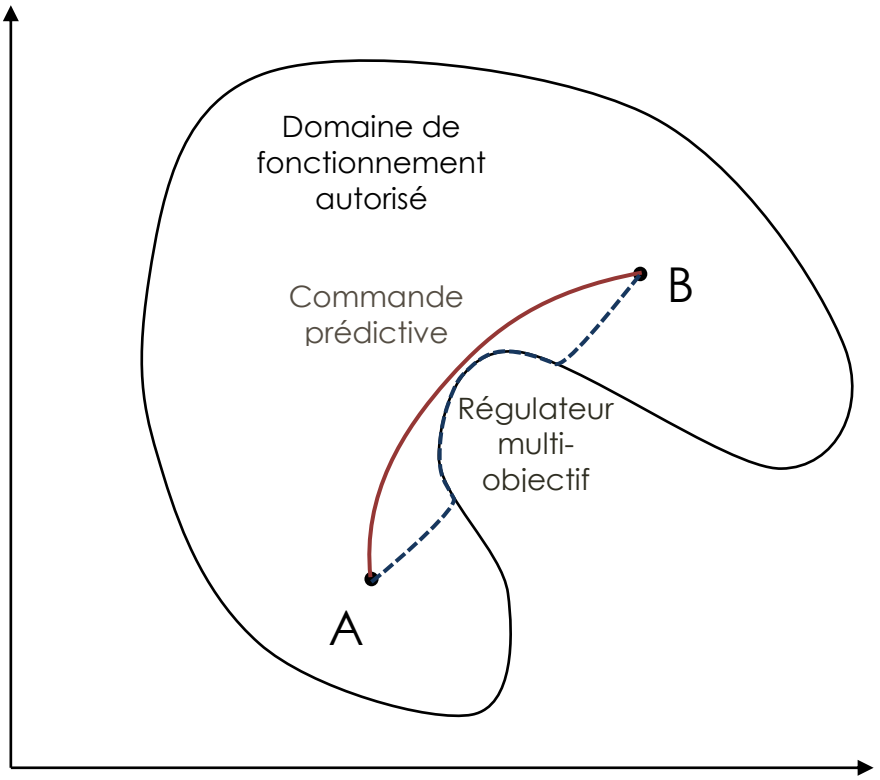


Figure 71: Illustration de la différence de trajectoire régulateur prédictif et multi-objectif

Il consiste en un régulateur PI multivariables à gains séquencés (cf. 10.3.1). Le séquencement du régulateur ne sera plus sur la puissance comparé à [Lemazurier *et al.*, 2017a] mais sur la position du groupe Pbank (cf. 12.4.2).

Le modèle que nous utilisons pour synthétiser les régulateurs LTI multi-objectif est le modèle point linéarisé (cf. 11.6.4) autour des points de fonctionnement considérés (cf. 12.4.2). Remarquons que le modèle utilisé pour ce régulateur ne modélise pas le xénon (cf. 11.6.4). En effet, le xénon étant très lent devant les variations des autres états, le régulateur multi-objectif n'aura pas pour mission de l'anticiper (cette tâche est dédiée au superviseur).

De plus, il est synthétisé sur un ensemble de régulateurs LTI locaux. Il ne bénéficie donc pas d'une bonne représentativité du modèle non-linéaire pour des variations de grande amplitude de puissance (e.g. variations de charge).

10.3.3.4 Le superviseur

Le Superviseur consiste en un algorithme de commande prédictive utilisant le modèle non-linéaire du réacteur présenté dans 11 et sera décrit dans la partie 0. Cette technique de commande répond à un grand nombre de nos problématiques :

- *Elle allie à la fois la capacité de contrôler le système quelles que soient ses dynamiques (lentes : le xénon, et/ou rapides : la température) en utilisant un modèle potentiellement non-linéaire du système.*
- *Elle autorise la prise en compte des retards, même importants. Dans notre cas, l'injection de bore.*
- *De plus, connaissant le programme de suivi de charge par avance* $\boldsymbol{D_U}$*, elle anticipe le comportement en calculant les trajectoires optimales compte tenu du programme.*
- *Enfin, les trajectoires du régulateur seront optimisées sur le comportement global du système et non local comme c'est le cas pour un régulateur à gains séquencés (cf. Figure 71).*

Cependant, comme tout algorithme de commande prédictive :

- Il ne possède pas de garanties de robustesse.
- Il calcule des commandes fixes sur un horizon de temps défini appelé '*pas d'échantillonnage*'. Selon les capacités de calcul disponibles, l'horizon de prédiction et la complexité du modèle utilisé (cf. 13.2), ce pas d'échantillonnage peut être grand. Dans ce cas, le superviseur n'a pas la capacité d'adapter ses commandes pour rejeter d'éventuelles perturbations non prévues. En effet, si ces perturbations sont plus rapides que le pas d'échantillonnage, le superviseur, ayant des commandes fixes sur ce pas, ne

pourra pas adapter ses commandes suffisamment rapidement pour rejeter les perturbations. En l'occurrence, ce sont les variations de puissance dues au réglage de fréquence (cf. 10.2.2 qui risquent de poser problème. En effet, ces dernières sont aléatoires et rapides.

Remarque

Dans cette étude nous utilisons le modèle point non-linéaire (défini dans la suite cf. 11.4). Nous avons en parallèle, développé un modèle axial multi-mailles défini dans l'Annexe F permettant une modélisation plus fine du comportement du cœur. En effet, ce modèle permet de représenter naturellement l'AO (contrairement au modèle point nécessitant un modèle d'AO en complément). Par soucis de simplicité nous avons tout de même choisi de garder le modèle point non-linéaire qui montre de bons résultats.

10.3.3.5 Conclusion sur la stratégie de commande adoptée

En conclusion, l'idée centrale de notre approche est d'associer un algorithme de commande prédictive et un régulateur multi-objectif à gains séquencés. Les avantages de l'un permettent de compenser au moins en partie les faiblesses de l'autre, tel que nous l'illustrons sur le Tableau 5.

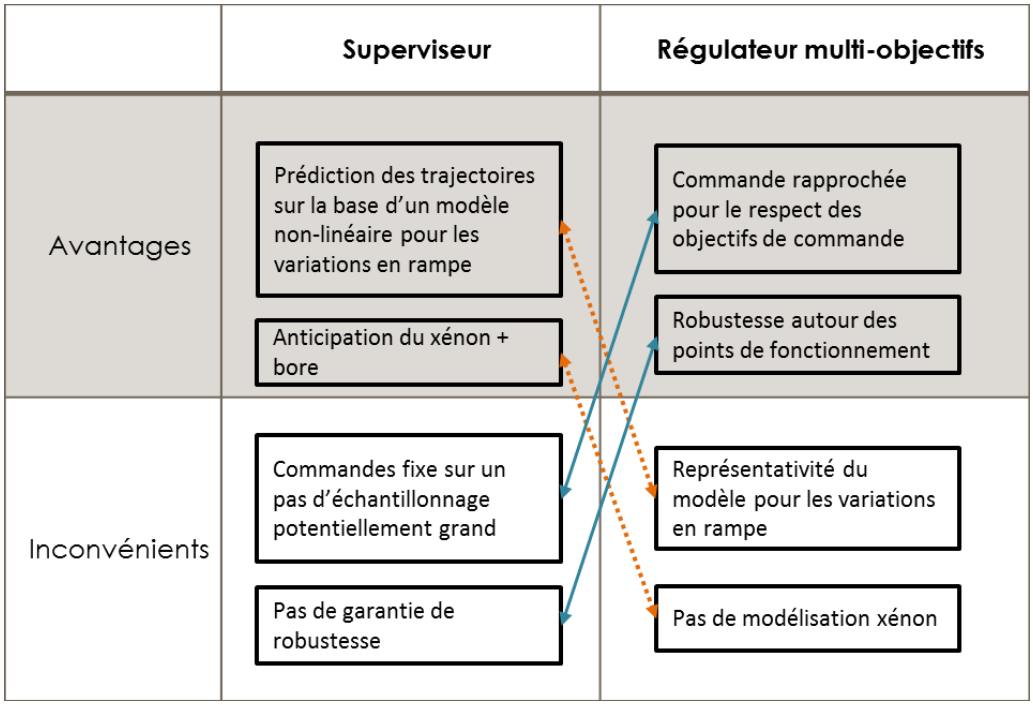


Tableau 5: Avantages et inconvénients des régulations

10.4 Présentation de la démarche de conception et de paramétrage du régulateur

L'organisation de la partie automatique se déroulera de la manière décrite par la Figure 72. Dans une partie Modélisation pour la commande nous définissons deux modèles : un modèle non-linéaire point utilisé par le superviseur ainsi que sa forme linéarisée autour de points de fonctionnement, et mis sous une représentation quasi-LPV pour le régulateur multi-objectif à gains séquencés. Nous verrons ensuite les deux régulateurs : superviseur et multi-objectif présentés et simulés séparément. Enfin, dans la partie Simulation et analyse du régulateur complet nous simulerons le régulateur dans son ensemble tel que défini dans 14.

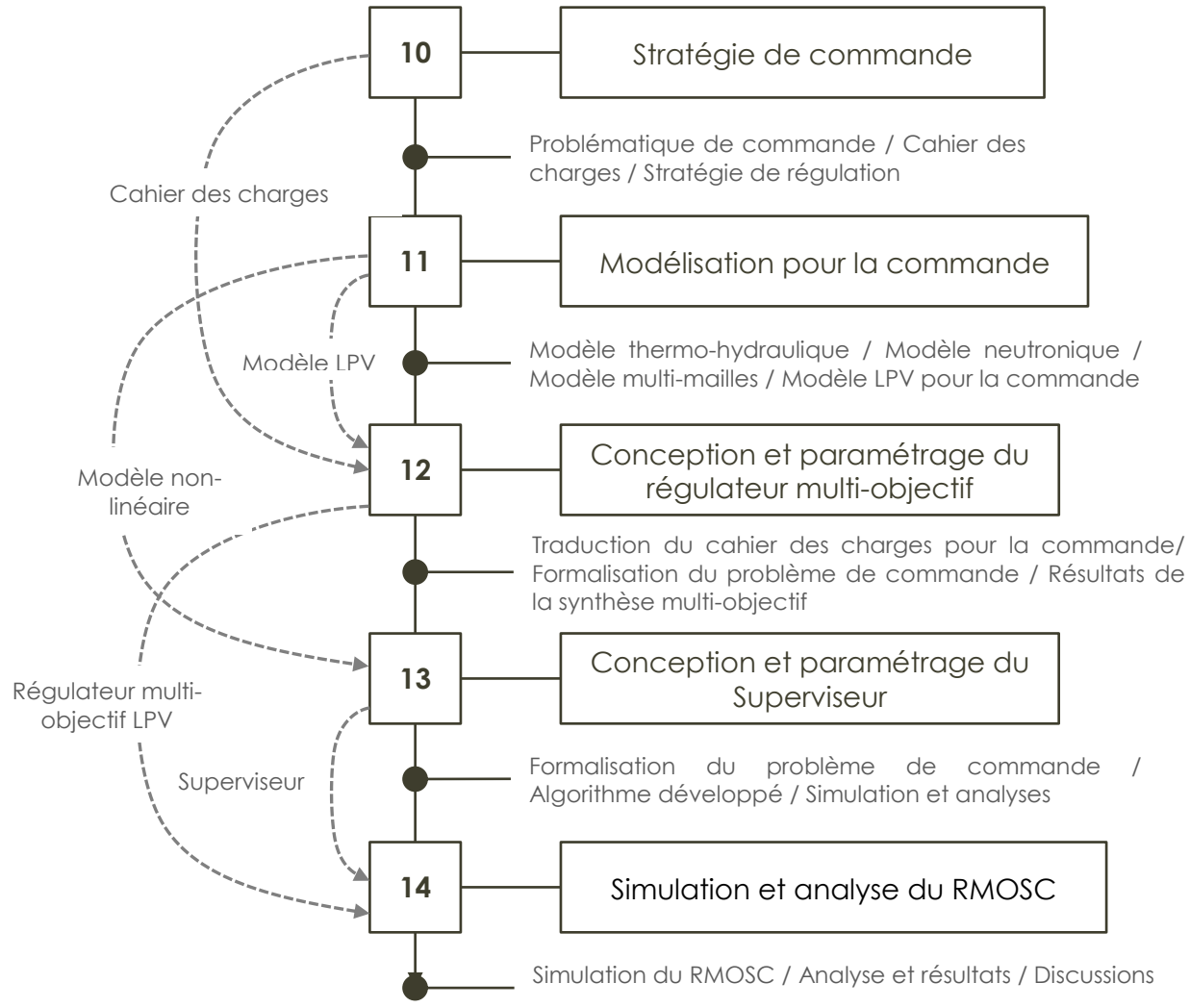


Figure 72: Organisation de la partie automatique

11 Modélisation pour la commande

Sommaire

11.1 Modélisation thermo-hydraulique 191
11.1.1 Modèle thermo-hydraulique du cœur 191
11.1.2 Modèle thermo-hydraulique du GV 192
11.1.3 Du GV à la turbine 192
11.1.4 Modélisation des branches chaude et froide 192
11.2 Modélisation neutronique du cœur 193
11.2.1 Notion de réactivité 193
11.2.2 Equations de la neutronique 194
11.2.3 Relation puissance et densité de neutrons 194
11.2.4 Effets sur la réactivité 195
11.3 Modélisation de l'Axial Offset 202
11.4 Modélisation du retard de bore 204
11.5 Le modèle non-linéaire du réacteur 206
11.6 Modèle point quasi-LPV 207
11.6.1 Caractérisation des non-linéarités 207
11.6.2 Représentation d'état du modèle point non-linéaire 209
11.6.3 Linéarisation de la représentation d'état non-linéaire 210
11.6.4 Mise sous la forme quasi-LPV du modèle 213
11.7 Conclusions sur la modélisation du réacteur 217

Comme nous l'avons vu précédemment, la stratégie de régulation consiste en un superviseur et un régulateur multi-objectif. Chacun de ces régulateur nécessite un modèle. Premièrement, le superviseur utilise un modèle pour calculer en ligne (et donc de manière interne) le comportement du réacteur sur un horizon de temps donné. Deuxièmement, le problème de commande qui sera posé pour le régulateur multi-objectif utilise un modèle quasi-LPV du réacteur pour l'optimisation. Nous allons donc procéder de la manière suivante :

- Tout d'abord, nous définirons un modèle du réacteur. Ce modèle est constitué d'un modèle thermo-hydraulique du réacteur et du modèle neutronique du cœur. Il est non-linéaire.
- Ce modèle sera nommé le **modèle non-linéaire du réacteur** et sera utilisé par le superviseur.

- Ensuite, nous linéariserons ce modèle non-linéaire autour de différents points de fonctionnement pour former le modèle **point quasi-LPV**. Ce modèle sera utilisé pour la synthèse des régulateurs multi-objectif par la suite.

Dans cette partie, nous commencerons par présenter la modélisation thermo-hydraulique du circuit primaire et du cœur. Dans un deuxième temps nous présenterons la modélisation neutronique du cœur vu comme un point. Un modèle d'AO est proposé. Enfin, nous formaliserons le modèle quasi-LPV du réacteur considéré.

11.1 Modélisation thermo-hydraulique

Cette partie a pour but de présenter le modèle thermo-hydraulique considéré. Ce modèle caractérise les échanges thermiques dans le circuit primaire.

11.1.1 Modèle thermo-hydraulique du cœur

Le cœur est considéré comme un échangeur simple de chaleur. La puissance dégagée par le cœur (notée P_K) élève la température du fluide caloporteur de la température d'entrée (branche froide) notée T_f à la température de sortie (branche chaude) notée T_c.

L'expression du flux thermique Φ_K échangé s'obtient en écrivant le bilan de la chaleur perdu par le fluide chaud et gagnée par le fluide froid telle que :

$$\Phi_K = Q_p C_p (T_c - T_f) \tag{11.1}$$

avec Q_p le débit massique primaire (en $kg.s^{-1}$) et C_p la capacité thermique massique du fluide caloporteur (en $J.kg^{-1}.K^{-1}$).

Cependant, le cœur étant de grande dimension, l'établissement de la chaleur dans le cœur ne se fait pas de manière instantanée. Nous choisissons alors un filtre du premier ordre pour représenter le comportement dynamique de l'échange thermique dans le cœur tel que :

$$P_K(s) = \frac{1}{1+\tau_{co}s}\Phi_K(s) \tag{11.2}$$

On peut alors exprimer la température branche chaude en fonction de la puissance cœur et de la température branche froide de la manière suivante :

$$\frac{dT_c}{dt}(t) = -\frac{1}{\tau_{co}}T_c(t) + \frac{1}{\tau_{co}}T_f(t) + \frac{1}{\tau_{co}Q_pC_p}P_K(t)$$

11.1.2 Modèle thermo-hydraulique du GV

Le générateur de vapeur, comme son nom l'indique, est un système permettant de créer de la vapeur d'eau sous pression, utilisée par la suite par la turbine pour produire de l'électricité. Il extrait la chaleur du circuit primaire par échange thermique. De la même manière que pour la modélisation faite du cœur (cf. équation (11.1)), on peut exprimer le flux thermique extrait par le GV de la manière suivante :

$$\Phi_{GV} = Q_pC_p(T_f^{GV} - T_c^{GV})$$

avec Q_p le débit massique primaire (en $kg.s^{-1}$) et C_p la capacité thermique massique du fluide caloporteur (en $J.kg^{-1}.K^{-1}$), T_c^{GV} la température entrant dans le GV (branche chaude provenant du cœur), T_f^{GV} la température sortant du GV (branche froide retournant vers le cœur).

Comme pour le cœur (équation (11.2)), nous choisissons alors un filtre du premier ordre pour représenter le comportement dynamique de l'échange thermique dans le cœur tel que :

$$P_{GV}(s) = \frac{1}{1+\tau_{GV}s}\Phi_{GV}(s) \quad (11.3)$$

Avec τ_{GV} la constant de temps du filtre passe bas.

11.1.3 Du GV à la turbine

Pour simplifier la modélisation, nous considérerons que la puissance extraite par le GV est égale à la puissance turbine. En réalité, la puissance appelée à la turbine est légèrement différente de celle extraite par le GV, mais nous n'aborderons pas ce problème dans notre étude. Ainsi :

$$P_{turb}(t) = P_{GV}(t) \quad (11.4)$$

11.1.4 Modélisation des branches chaude et froide

La branche chaude achemine l'eau du cœur au GV et la branche froide ferme le circuit en ramenant l'eau en sortie du GV vers le cœur. En réalité, il s'agit donc d'un transport d'eau du GV vers le cœur et inversement. Ce transport peut être modélisé comme un retard, mais dans ce cas le système serait de dimension infinie. Pour

simplifier la modélisation nous considérerons un simple filtre du premier ordre. On a donc, avec les mêmes notations que précédemment :

$$T_c^{GV}(s) = \frac{1}{1+\tau_{bc}s} T_c(s) \quad (11.5)$$

$$T_f(s) = \frac{1}{1+\tau_{bf}s} T_f^{GV}(s) \quad (11.6)$$

Avec τ_{bc} et τ_{bf} les constante de temps des filtre passe bas modélisant le retard des branches.

11.2 Modélisation neutronique du cœur

Pour le modèle neutronique du cœur nous adoptons une modélisation point [Kerlin *et al.*, 1976]. Ce modèle consiste à représenter le cœur comme un point chaud, élevant la température du fluide primaire par génération de puissance thermique issue de la réaction nucléaire, i.e. de la fission. Nous commencerons par un bref rappel de la physique des réacteurs. Ensuite nous définirons les équations régissant l'évolution des neutrons dans le cœur.

11.2.1 Notion de réactivité

La neutronique correspond à l'étude de phénomènes physiques produits par le cheminement des neutrons dans la matière. Cette discipline a permis de comprendre les différentes interactions entre les neutrons et la matière constitutive du cœur. Dans un réacteur nucléaire, la majorité des neutrons émis sont issus de la fission des atomes d'uranium, et principalement l'U_{235}. Par capture d'un neutron, cet atome lourd se fissionne en plusieurs produits de fissions et libère de nouveaux neutrons disponibles pour de futures réactions nucléaires. C'est le phénomène de réaction en chaine.

Pour caractériser l'évolution des neutrons dans le cœur on définit la notion de réactivité, matérialisée par la variable ρ. La réactivité représente la variation de la population neutronique et s'exprime en pcm (pour cent-mille). Elle est donc sans unité. Par exemple, un réacteur sur-critique de réactivité $\rho = 1\ pcm$ verra sa population neutronique augmenter de 1 pour cent-mille individus à chaque nouvelle génération.

On distingue donc trois cas :

- $\rho < 0$. La réaction s'étouffe et le nombre de neutrons dans le cœur diminue. Le cœur est sous-critique.

- $\rho = 0$. La réaction dans le cœur est maintenue constante. Chaque neutron thermique émis donne naissance à 1 seul neutron thermique en moyenne. On dit que le cœur est critique.
- $\rho > 0$. La réaction s'emballe, le nombre de neutrons dans le cœur est croissant et suit la loi exponentielle. On dit que le cœur est sur-critique.

11.2.2 Equations de la neutronique

Une partie des neutrons émis, permettant de générer de l'énergie, sont des neutrons libérés par des produits de fission dans un délai bien plus long (de l'ordre de 11,3 secondes après la fission) que celui d'atomes d'uranium. Ils sont appelés les neutrons retardés. On appelle '*précurseurs*' les éléments qui émettent ces neutrons.

Soient n la densité de neutrons et c_i la densité de précurseurs i. On définit alors les équations de la neutronique, régissant l'évolution des neutrons et des précurseurs par les relations suivantes :

$$\frac{dn}{dt} = \frac{\rho-\beta}{l^*} n + \sum_{i=1}^{n_r} \lambda_i c_i \tag{11.7}$$

$$\frac{dc_i}{dt} = \frac{\beta}{l^*} n - \lambda_i c_i \tag{11.8}$$

Où, l^* le temps génération moyen que met un neutron pour en produire un autre, λ_i la constante de désintégration radioactive du précurseur i et β la proportion de neutrons retardés.

Par soucis de simplification, nous considèrerons dans notre étude qu'un seul groupe de neutrons retardés, on déduit des équations (11.7) et (11.8) le système d'équations suivant :

$$\begin{cases} \frac{dn}{dt}(t) = \frac{\rho-\beta}{l^*} n(t) + \lambda c(t) \\ \frac{dc}{dt}(t) = \frac{\beta}{l^*} n(t) - \lambda c(t) \end{cases} \tag{11.9}$$

11.2.3 Relation puissance et densité de neutrons

La puissance du cœur est considérée proportionnelle à la concentration de neutrons dans le cœur. Soit K_n le coefficient de conversion de puissance, on a alors :

$$P_K = K_n . n \tag{11.10}$$

Notons que cette équation permet de relier le modèle thermo-hydraulique défini dans 11.1.1 et le modèle point.

11.2.4 Effets sur la réactivité

La réactivité ρ est fonction des phénomènes physiques qui se produisent dans le cœur. Ces phénomènes seront décrits dans les parties suivantes :

- Les grappes de contrôle
- La concentration de bore
- L'effet modérateur
- L'effet doppler
- La concentration en xénon.

La réactivité globale est alors définie comme la somme des effets dus à ces phénomènes et est définie de la manière suivante :

$$\rho = \rho_{barres} + \rho_{bore} + \rho_{mod} + \rho_{dop} + \rho_{Xe} \quad (11.11)$$

Où ρ_{barres} est la réactivité liée à la position des barres, ρ_{bore} celle de la concentration en bore, ρ_{mod} la réactivité liée à l'effet modérateur, ρ_{dop} la réactivité de l'effet Doppler et ρ_{Xe} celle du Xénon.

11.2.4.1 L'effet des groupes de grappes

Les grappes de contrôles sont composées d'absorbants neutroniques s'insérant dans le cœur dans le but de contrôler la réactivité et la distribution axiale de puissance.

Les grappes de contrôle sont au nombre de 36 regroupées en 9 quadruplés (cf. 3.2.4) : PA, PB, PC, PD, PE, PF, PG, PH, PJ. Notons qu'il n'y a pas de groupe PI pour des raisons de notation. Ces groupes sont ensuite regroupés en 5 groupes P1 à P5 selon plusieurs séquences. Le palier FA3 dispose ainsi de deux séquences définies dans le Tableau 6 ci-dessus.

Groupes	Séquence 1	Séquence 2
P1	PA	PC
P2	PB	PB
P3	PC + PD	PA + PD
P4	PE + PF	PE + PF
P5	PG + PH + PJ	PG + PH + PJ

Tableau 6: Exemple de séquences des groupes de contrôle

Enfin, les groupes P1 à P5 sont regroupés dans deux groupes à composition variable (cf. 3.2.4) Pbank et Hbank.

On pose alors P_i avec $i \in \{1,..,5\}$ la position du groupe de barres i (P1 à P5) et K_{P_i} l'efficacité de ce groupe. L'efficacité d'un groupe traduit la réactivité induite par son déplacement. Elle varie en fonction du flux. Or, puisque le flux n'est pas homogène dans le cœur, l'efficacité du groupe dépend de sa position dans le cœur. On pose :

$$\rho_{P_i}(t) = K_{P_i}\big(P_i(t)\big).(P_i(t) - P_i^0) \qquad (11.12)$$

On identifie sur code de simulation Framatome l'efficacité d'un groupe, comme définie par (11.12). La courbe est reportée Figure 73. On peut alors déduire de l'équation (11.12) la réactivité induite par les cinq groupes P1 à P5. On a alors :

$$\rho_{barres}(t) = \sum_{i=1}^{5} K_{P_i}(P_i(t))(P_i(t) - P_i^0)$$

Notons que les coefficients K_{P_i} ont des ordres de grandeurs différents dûs aux compositions différentes des groupes P1 à P5 (cf. Tableau 6). P1 est composé d'un seul quadruplet de barres alors que P5 est composé de 3 quadruplets. P5 a donc 3 fois plus d'efficacité que P5.

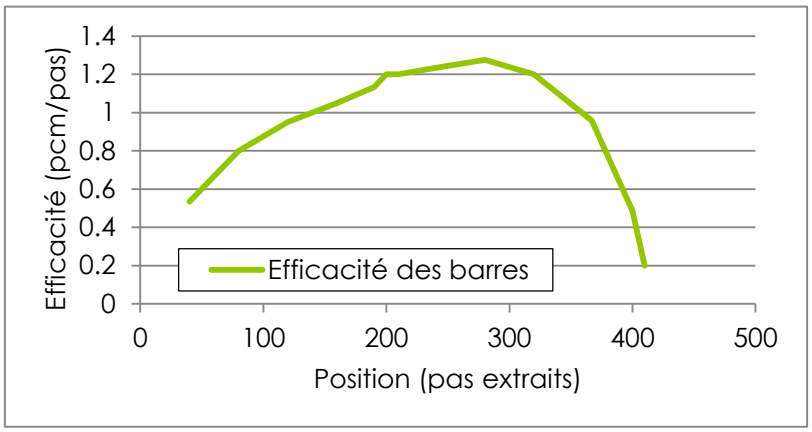


Figure 73: Illustration de l'efficacité (K_p) d'un quadruplet de barre en fonction de son insertion (palier TSN – quadruplet PA) – BU150 – E18[7]

[7] Données extraites de SMART, code de simulation Framatome sur le palier TSN - Cycle E18 - épuisement BU150.

La réactivité insérée par les groupes Pbank et de Hbank séparément dépend de la composition de chacun des groupes et on a alors :

$$\rho_{P_{bank}}(t) = \sum_{i=1}^{n_P} K_{P_i}\big(P_i(t)\big).\,(P_i(t) - P_i^0) \tag{11.13}$$

$$\rho_{H_{bank}}(t) = \sum_{i=n_P+1}^{5} K_{P_i}\big(P_i(t)\big).\,(P_i(t) - P_i^0) \tag{11.14}$$

avec n_P le nombre de groupe i dans Pbank et $\rho_{barres} = \rho_{P_{bank}} + \rho_{H_{bank}}$.

En considérant de petites variations autour d'un point d'équilibre on peut considérer que la position des groupes varie peu et donc que le coefficient associé est constant. On déduit de (11.13) et (11.14) que :

$$\rho_{P_{bank}}(t) = \sum_{i=1}^{n_P} K_{P_i}.\,(P_i(t) - P_i^0)$$

$$\rho_{H_{bank}}(t) = \sum_{i=n_P+1}^{5} K_{P_i}.\,(P_i(t) - P_i^0)$$

Puisqu'on se place dans le cas d'un recouvrement fixe (cf. 3.2.4) la variation de position du groupe Pbank (resp. Hbank) est la même pour les groupes qui le composent. En posant $K_P = \sum_{i=1}^{n_P} K_{P_i}$ et $K_H = \sum_{i=n_P+1}^{5} K_{P_i}$ on déduit des équations précédentes que :

$$\rho_{P_{bank}}(t) = K_{P_b}(P_{bank}(t) - P_{bank}^0)$$

$$\rho_{H_{bank}}(t) = K_{H_b}(H_{bank}(t) - H_{bank}^0)$$

Enfin, on a :

$$\rho_{barres}(t) = K_{P_b}\big(P_{bank}(t) - P_{bank}^0\big) + K_{H_{bank}}(H_b(t) - H_{bank}^0) \tag{11.15}$$

Enfin, la commande de barres appliquée sur le processus et donc calculée par le futur régulateur multi-objectif est la vitesse des barres Pbank et Hbank notées respectivement vP_{bank} et vH_{bank}. On a alors :

$$\frac{dP_{bank}}{dt}(t) = vP_{bank} \tag{11.16}$$

$$\frac{dH_{bank}}{dt}(t) = vH_{bank} \tag{11.17}$$

11.2.4.2 L'effet du Bore

Le bore est un absorbant neutronique ayant pour objectif de réguler la réactivité du cœur. Plus sa concentration est élevée plus il absorbe de neutrons et donc plus il réduit la réactivité du cœur. Pour augmenter la concentration en bore, du bore soluble est injecté par l'intermédiaire du RCV dans le circuit primaire. Inversement, pour en diminuer la concentration, de l'eau claire (concentration en bore nulle) est injectée dans le circuit primaire.

Effet du bore sur la réactivité

L'impact de la dilution / borication sur la concentration en bore dans le cœur est fonction de la concentration. On définit le coefficient d'efficacité du bore noté K_{CB}[8] sur la réactivité. Supposons le cœur à l'équilibre, on définit par C_b^0 la concentration de bore à l'équilibre. Soit, C_b la concentration en bore du cœur et ρ_{bore} la réactivité induite par les variations de concentration de bore. On a alors :

$$\rho_{bore} = K_{CB}(C_b - C_b^0) \tag{11.18}$$

On note u_Q la commande de débit de bore, ou plus exactement de variation de concentration de bore telle que :

$$u_Q(t) = \frac{dC_b}{dt}(t) \tag{11.19}$$

En toute rigueur la modulation de concentration de bore se fait par injection de bore soluble ou d'eau distillée dans le circuit primaire. Nous invitons le lecteur à se référer à l'Annexe E pour les détails concernant la relation entre la commande u_Q et les débits d'injection de bore et d'eau dans le circuit primaire.

11.2.4.3 L'effet modérateur

Le modérateur d'un réacteur ralentit les neutrons afin qu'ils atteignent une vitesse (et donc une énergie) suffisamment faible pour être capturés par les atomes fissiles. Le ralentissement des neutrons est provoqué par les chocs successifs avec les atomes du modérateur. Dans le cas des réacteurs à eau légère pressurisée, ce rôle est joué par les atomes d'hydrogène.

[8] En toute rigueur le coefficient K_{CB} varie au fur et à mesure de la vie de la tranche. Principalement selon les cycles et en particulier pour un cœur neuf (cycle N1). Il varie cependant lentement et est bien connu. Dans toute notre étude nous le considèrerons constant et de l'ordre de $6\ \mathrm{pcm/ppm}$.

L'augmentation de puissance du réacteur a pour effet l'augmentation de la température de l'eau circulant dans celui-ci. Or, l'augmentation de température engendre une diminution de la densité du modérateur et par conséquent l'augmentation de la distance entre les molécules d'eau. Ainsi, la probabilité, pour un neutron naviguant dans le modérateur, de heurter un atome d'hydrogène d'une molécule d'eau décroit. Par conséquent la distance et le temps nécessaires pour atteindre l'état thermique (i.e. une vitesse suffisamment faible pour être capturé par un atome fissile) se trouvent rallongés et ainsi la réactivité et la puissance s'en trouve réduites. C'est ce que l'on appelle, par abus de langage l'**effet modérateur** (rigoureusement, on devrait parler d'effet de variation de modérateur). La Figure 74 montre que le temps de parcours d'un neutron (en vert) est plus long quand les atomes d'hydrogène (en rouge) sont plus éloignés sous l'effet de la dilation du modérateur.

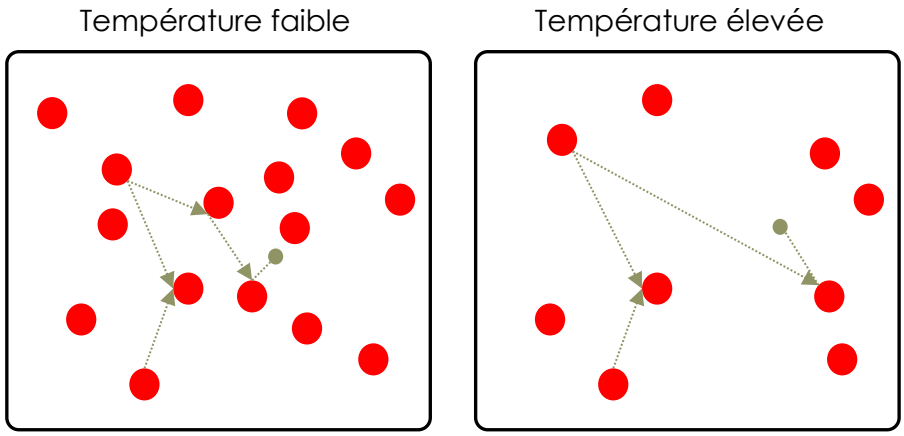


Figure 74: Illustration de l'effet modérateur

Comme l'illustre la Figure 75, l'effet modérateur dépend de la température. Or, entre le haut et le bas du cœur, la température varie fortement (environ 30°C d'écart). Notons, les températures branches chaude et froide respectivement T_c et T_f. On définit par K_{mod} le coefficient modérateur. Il traduit la réactivité induite par l'effet modérateur. Alors, la réactivité induite par la variation de température dans le haut et le bas du cœur se définit par l'équation suivante :

$$\rho_{mod}(t) = K_{mod}\left(T_f(t)\right).\left(T_f(t) - T_f^0\right) + K_{mod}\left(T_c(t)\right).\left(T_c(t) - T_c^0\right) \qquad (11.22)$$

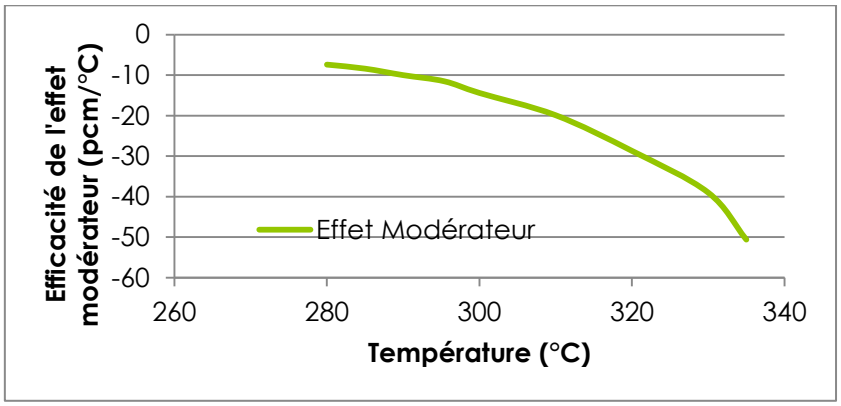


Figure 75: Efficacité de l'effet modérateur en fonction de la température

11.2.4.4 L'effet Doppler

Tout comme l'effet modérateur, l'effet Doppler est un phénomène de contre-réaction, i.e. qui s'oppose à l'évolution de la puissance dans le cœur (autrement dit, il tend à stabiliser la réaction). Ce phénomène provient de la capture stérile des neutrons par l'U_{238}. Cet atome, en grand nombre dans le cœur capte des neutrons qui ne sont plus disponibles pour la réaction (qui s'« étouffe » alors). L'effet doppler est proportionnel à la température moyenne du combustible. Puisque la température moyenne est quasiment proportionnelle à la puissance (cf. 10.2.6), on considère généralement que l'effet doppler est proportionnel à la puissance. De plus, comme illustré sur la Figure 76, l'effet doppler varie en fonction de la puissance.

Il s'exprime alors de la manière suivante :

$$\rho_{dop}(t) = K_{dop}\big(P_k(t)\big).(P_K(t) - P_k^0) \qquad (11.23)$$

Où, P_K est la puissance du cœur, P_k^0 est la puissance à l'équilibre du cœur, K_{dop} le coefficient doppler et ρ_{dop} la réactivité induite par l'effet doppler. Comme le montre la Figure 76, le coefficient doppler dépend de la puissance du cœur.

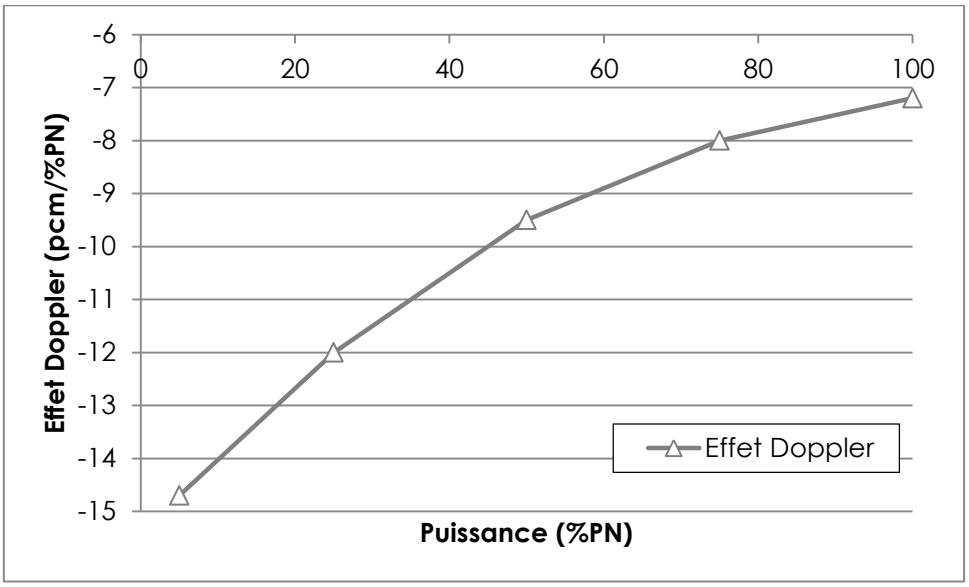


Figure 76: Illustration de l'effet doppler

11.2.4.5 L'effet du xénon

Le xénon est un produit de fission ayant un fort pouvoir absorbant. Il est considéré comme un poison neutronique. Il est produit par la chaine de réaction nucléaire suivante :

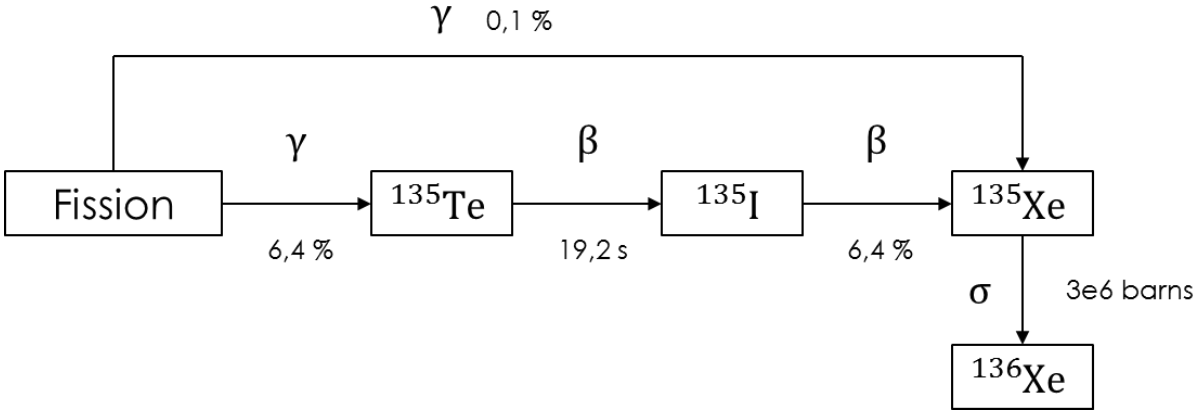


Figure 77: Chaine de réaction produisant le xénon 135 (les rendements sont relatives aux fissions de l'$^{235}\mathbf{U}$ et les durées sont les périodes radioactives)

Ainsi, la variation de puissance engendre des variations de concentration d'Iode puis de Xénon ayant un impact sur la réactivité du cœur. Les équations régissant la dynamique de l'iode et du xénon sont alors définies par :

$$\frac{dI}{dt}(t) = \gamma_I n(t) - \lambda_I I(t) \qquad (11.25)$$

$$\frac{dXe}{dt}(t) = \Gamma_{Xe} n(t) + \lambda_I I(t) - (\lambda_{Xe} + \sigma_{\mathrm{Xe}} n(t)))Xe(t) \quad (11.26)$$

Où Xe est concentration des noyaux de Xénon 135, I est concentration des noyaux d'Iode 135, Φ est le flux de neutrons dans le cœur, Σ_f la section efficace macroscopique de fission, λ_{Xe} est la constante radioactive du xénon, λ_I est la constante radioactive de l'iode, γ_{Xe} est le rendement de fission du xénon, γ_I est le rendement de fission de l'iode (en fait celui du tellure), σ_{Xe}^0 est la section efficace de capture du Xénon.

Enfin, l'effet de la concentration xénon sur la réactivité notée ρ_{Xe} est quasiment linéaire. On l'approxime par la constance K_{Xe} tel que :

$$\rho_{Xe} = K_{Xe}.Xe \quad (11.27)$$

11.3 Modélisation de l'Axial Offset

L'AO est une image de la répartition de puissance dans le cœur et est donnée par l'équation :

$$AO = \frac{P_{haut} - P_{bas}}{P_{haut} + P_{bas}}$$

Les phénomènes qui provoquent la majeure partie des variations d'AO sont la position des groupes de grappes et la température.

- Impact de la position des groupes sur l'AO

La modification de la position des groupes de grappes dans le cœur provoque localement des variations de puissance, modifiant ainsi l'AO. L'extraction (resp. insertion) d'un groupe dans le haut du cœur induit une augmentation (resp. diminution) de l'AO comme l'illustre la Figure 78 pour le groupe de barres PA.

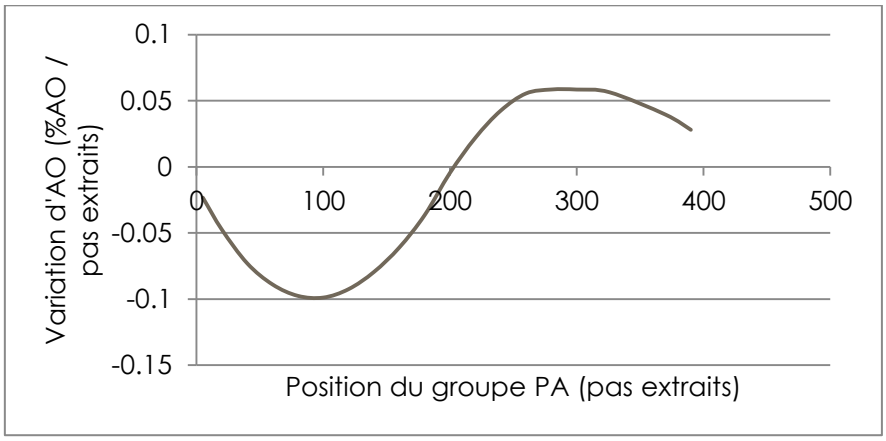


Figure 78: Effet de la barre PA sur l'AO (Données SMART – Palier TSN – Cycle E18 – BU150 – AO initial -3%)

- Impact de la température sur l'AO

Comme nous l'avons présenté dans 11.2.4.3, l'apport de réactivité induit par la variation de la température du modérateur varie selon le haut et le bas du cœur. Ainsi, l'effet de la vairation de chaque branche (chaude et froide) a un impact sur l'AO, comme illustré sur les Figure 79 et Figure 80.

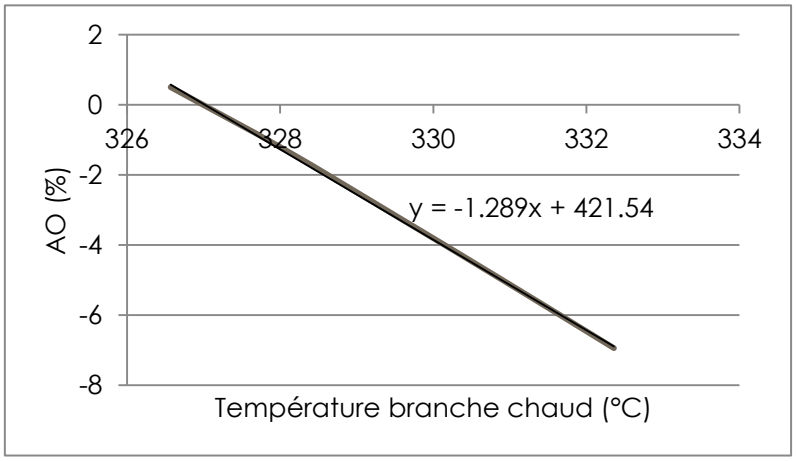


Figure 79: Effet de la température branche chaude sur l'AO à température branche froide constante ($T_f = 295\ °C$) (Données en début de vie)

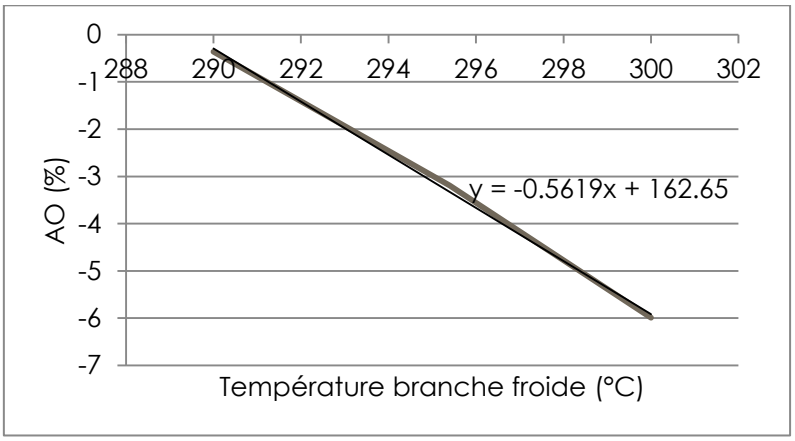


Figure 80 : Effet de la température branche froide sur l'AO à température branche chaude constante ($T_c = 329\,°C$) (Données en début de vie)

Supposons que le cœur est à l'équilibre. Soient T_f la température branche froide, T_c la température branche chaude, P_{bank} la position de Pbank, H_{bank} la position de Hbank, K_{AO}^c et K_{AO}^f les coefficients de l'effet des températures et K_{AO}^P et K_{AO}^H les coefficients de l'effet des barres sur l'AO. Le modèle d'AO est défini par l'équation :

$$AO = \frac{1}{1+\tau_{AO}s}.\left(K_{AO}^c.T_c + K_{AO}^f.T_f + K_{AO}^P.P_{bank} + K_{AO}^H.H_{bank}\right) \qquad (11.28)$$

Notons que nous avons rajouté un filtre de constante de temps τ_{AO}.

11.4 Modélisation du retard de bore

Contrairement à une grande partie des réacteurs du parc français, l'architecture du réacteur considérée dans cette étude (similaire aux EPR de Framatome) n'induit pas de formation de « bouchons » de bore ou d'eau distillée dans les circuits d'injection grâce à un piquage direct dans le circuit primaire. L'architecture induit en revanche un retard pur d'injection de bore et d'eau particulièrement élevé, de l'ordre de 300 secondes. Bien que ce retard soit élevé, les dynamiques du système considéré sont suffisamment lentes pour que le système soit commandable.

Il existe dans la littérature plusieurs possibilités pour traiter un retard pur sur un actionneur. [Chevrel et Berrini, 2010] compare la stratégie avec prédicteur de Smith et l'oppose à un estimateur d'état. Ces approches permettent de traiter le problème de commande sans prendre en compte le retard et d'établir une structure de prédiction (ou d'observation) du retard pour compenser son effet. On notera par ailleurs les travaux de [Bresch-Pietri *et al.*, 2014] qui proposent une approche basée sur la commande prédictive pour contrôler une classe de systèmes linéaires caractérisée par un retard en entrée dépendant des valeurs prises par l'entrée. Cas fréquemment rencontré dans les systèmes industriels.

D'autres approches consistent à introduire un modèle de retard directement dans le modèle de système considéré et de synthétiser le régulateur sur cette base. Le modèle de retard le plus connu est celui de l'approximant de Laguerre [Partington, 1991]. C'est une méthode efficace d'approximation du retard par un système de dimension finie. On le définit généralement par le transfert R_L d'ordre *n* donné dans l'équation suivante :

$$R_B(s) = \frac{\left(1 - \frac{h}{2n}s\right)^n}{\left(1 + \frac{h}{2n}s\right)^n}$$

avec n l'ordre de l'approximant de Laguerre, h le retard approximé.

La Figure 81 compare la réponse temporelle de trois approximants de Laguerre d'ordre 3, 5 et 10 de retard 300 secondes, à la réponse temporelle d'un retard pur de 300 secondes.

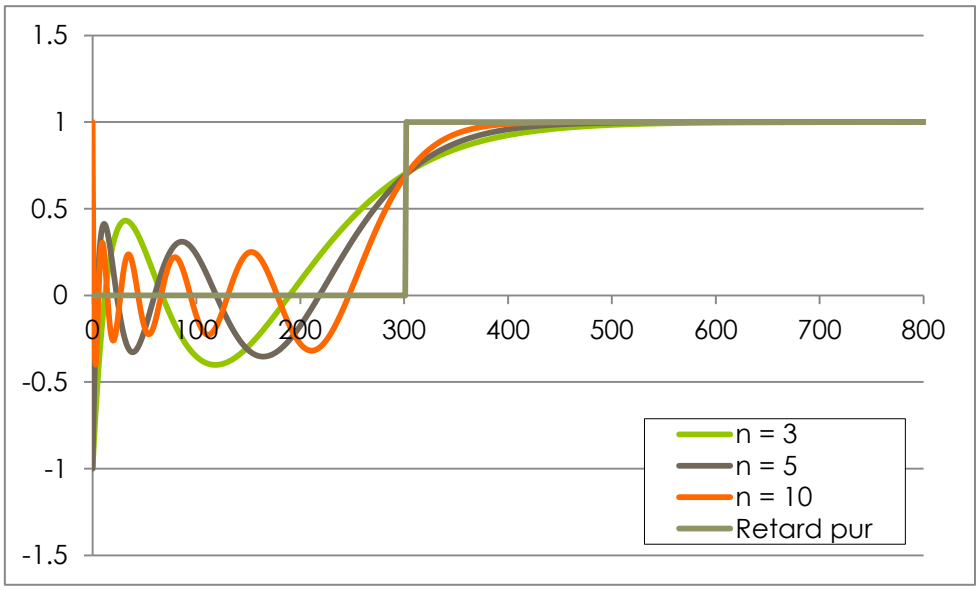


Figure 81: Réponse indicielle de l'approximant de Laguerre

On définit alors la commande de bore retardée $\overline{u_Q}$ telle que :

$$\overline{u_Q} = R_B(s).u_Q \qquad (11.20)$$

Où, R_B est un approximant de Laguerre d'ordre $n = 9$ et de retard $h = 300$ secondes.

Notons que cette modélisation sera utilisée uniquement dans le cadre du régulateur multi-objectif. Pour le superviseur, le modèle considéré sera le retard réel de bore.

11.5 Le modèle non-linéaire du réacteur

Cette partie formalise le modèle non-linéaire. Les équations du modèle thermo-hydraulique données par les équations (11.2, 11.3, 11., 11.5, 11.6) et celles du modèle neutronique et d'AO (11.9, 11.10, 11.11, 11.15, 11.16, 11.17, 11.18, 11.19, 11.22, 11.23, 11.25, 11.26, 11.27, 11.28) données plus haut, forment le modèle non-linéaire du réacteur dont les équations sont présentées ci-dessous :

$$\frac{dT_c}{dt}(t) = -\frac{1}{\tau_{co}}T_c(t) + \frac{1}{\tau_{co}}T_f(t) + \frac{1}{\tau_{co}}\frac{K_n}{Q_p C_p}n(t) \quad (11.a)$$

$$\frac{dT_f}{dt}(t) = -\frac{1}{\tau_{bf}}T_f(t) + \frac{1}{\tau_{bf}}T_f^{GV}(t) \quad (11.b)$$

$$\frac{dAO}{dt}(t) = -\frac{1}{\tau_{AO}}AO(t) + \frac{1}{\tau_{AO}}\left(\boldsymbol{K_{AO}^{c}}T_c(t) + \boldsymbol{K_{AO}^{f}}T_f(t) + \boldsymbol{K_{AO}^{P}}P_b(t) + \boldsymbol{K_{AO}^{H}}H_b(t)\right) \quad (11.c)$$

$$\frac{d\rho_{dop}}{dt}(t) = -\frac{1}{\tau_{dop}}\rho_{dop}(t) + \frac{\boldsymbol{K_{dop}}}{\tau_{dop}}n(t) \quad (11.d)$$

$$\frac{dn}{dt}(t) = \frac{\rho(t)-\beta}{l^*}n(t) + \lambda c(t) \quad (11.e)$$

$$\frac{dc}{dt}(t) = \frac{\beta}{l^*}n(t) - \lambda c(t) \quad (11.f)$$

$$\frac{dT_c^{GV}}{dt}(t) = -\frac{1}{\tau_{bc}}T_c^{GV}(t) + \frac{1}{\tau_{bc}}T_c(t) \quad (11.g)$$

$$\frac{dT_c^{GV}}{dt}(t) = -\frac{1}{\tau_{GV}}T_c^{GV}(t) + \frac{1}{\tau_{GV}}T_f^{GV}(t) + \frac{1}{\tau_{GV}}\frac{P_{turb}(t)}{Q_p C_p} \quad (11.h)$$

$$\frac{dI}{dt}(t) = \gamma_I n(t) - \lambda_I I(t) \quad (11.i)$$

$$\frac{dXe}{dt}(t) = \Gamma_{Xe}n(t) + \lambda_I I(t) - (\lambda_{Xe} + \sigma_{Xe}n(t))Xe(t) \quad (11.j)$$

$$\rho(t) = \rho_{dop}(t) + \rho_{barres}(t) + \rho_{mod}(t) + \rho_{bore}(t) + \rho_{Xe}(t) \quad (11.k)$$

$$\rho_{mod}(t) = \boldsymbol{K_c}\left(T_c(t) - T_c^i\right) + \boldsymbol{K_f}(T_f(t) - T_f^i) \quad (11.l)$$

$$\rho_{barres}(t) = \boldsymbol{K_P}\left(P_b(t) - P_b^i\right) + \boldsymbol{K_H}\left(H_b(t) - H_b^i\right) \quad (11.m)$$

$$\rho_{bore}(t) = \boldsymbol{K_{CB}}\left(C_b(t) - C_b^i\right) \quad (11.n)$$

$$\frac{dP_b}{dt}(t) = vP_{bank} \quad (11.o)$$

$$\frac{dH_b}{dt}(t) = vH_{bank} \quad (11.p)$$

$$\frac{dC_b}{dt}(t) = \overline{u_Q} \quad (11.q)$$

où, $$x = \begin{pmatrix} T_h & T_c & AO & \rho_{dop} & n & c & T_c^{GV} & T_f^{GV} & P_{GV} & P_{bank} & H_{bank} & C_b \end{pmatrix}^T \quad (11.r)$$

$$u = (vP_{bank} \quad vH_{bank} \quad \overline{u_Q})^T \quad (11.s)$$

et, $$d = P_{turb} \quad (11.t)$$

Par souci de lisibilité de la représentation, nous n'avons pas inclus le modèle du retard du bore ici. Rappelons que le retard est modélisé par l'approximation de Laguerre décrite dans la partie 11.2.4.2 telle que :

$$\overline{u_Q}(t) = u_Q(t - h_{bore}) \quad (11.u)$$

Ce modèle sera utilisé sous cette forme pour le superviseur.

11.6 Modèle point quasi-LPV

Cette partie a pour but de formaliser le modèle quasi-LPV que nous utilisons pour la synthèse des régulateurs multi-objectif à gains séquencés. Nous commençons par caractériser les non-linéarités du modèle non-linéaire du réacteur. Ensuite, nous donnons les clés de la linéarisation du système non-linéaire puis nous linéarisons le modèle point non-linéaire donné dans 11.4. Enfin, nous donnons la représentation quasi-LPV.

11.6.1 Caractérisation des non-linéarités

Non-linéarité état-état

Plusieurs non-linéarités se présentent dans les équations présentées ci-dessus. Tout d'abord, observons l'équation (11.e). n étant un état du système et ρ étant la somme de plusieurs états du système, le produit $\rho \times n$ est donc une non-linéarité état-état. Nous linéariserons cette équation dans la partie 11.6.3.

Ensuite, une deuxième non-linéarité est présente dans l'équation (11.j). Elle provient du produit $n \times X_e$. Le xénon ne sera pas modélisé dans le modèle linéarisé pour la

commande multi-objectif, cette équation ne sera donc pas utilisée. Pour cette raison nous ne la linéariserons pas.

Non-linéarités implicites : des coefficients variants

D'autres non-linéarités, implicites cette fois, se présentent dans les équations du modèle. Elles se caractérisent par les coefficients en gras. Ces coefficients varient en fonction de l'état du cœur. La dépendance de chacun de ces coefficients est donnée dans les parties : 11.2.4.1, 11.2.4.2, 11.2.4.3, 11.2.4.4 et 11.2.4.5.

On définit π le vecteur de paramètres variants tel que :

$$\pi = \left(\boldsymbol{K_{AO}^{c}} \quad \boldsymbol{K_{AO}^{f}} \quad \boldsymbol{K_{AO}^{P}} \quad \boldsymbol{K_{AO}^{H}} \quad \boldsymbol{K_{dop}} \quad \boldsymbol{K_{c}} \quad \boldsymbol{K_{f}} \quad \boldsymbol{K_{P}} \quad \boldsymbol{K_{H}} \quad \boldsymbol{K_{CB}}\right)^{T} \qquad (11.29)$$

Le Tableau 7 présente la dépendance de chacun des coefficients de π aux variables de modélisation.

	$\boldsymbol{K_{AO}^{c}}$	$\boldsymbol{K_{AO}^{f}}$	$\boldsymbol{K_{AO}^{P}}$	$\boldsymbol{K_{AO}^{H}}$	$\boldsymbol{K_{dop}}$	$\boldsymbol{K_{c}}$	$\boldsymbol{K_{f}}$	$\boldsymbol{K_{P}}$	$\boldsymbol{K_{H}}$	$\boldsymbol{K_{CB}}$
$\boldsymbol{P_K}$					✓					
$\boldsymbol{T_c}$	✓					✓				
$\boldsymbol{T_f}$		✓					✓			
$\boldsymbol{P_{bank}}$			✓						✓	
$\boldsymbol{H_{bank}}$				✓						✓

Tableau 7: Dépendance des paramètres de π au variables de fonctionnement

Plusieurs possibilités existent pour prendre en compte la variation de ces coefficients. Il est possible par exemple de définir une relation explicite de la dépendance de chacun des coefficients en fonction de l'état du cœur. On déduit alors un modèle linéarisé autour de chaque point de fonctionnement en linéarisant l'équation de dépendance ainsi formalisée.

L'autre possibilité, celle choisie dans la thèse, est d'identifier directement chacun des coefficients autour des points de fonctionnement. En effet, hormis la non-linéarité état-état citée précédemment, les autres équations du système sont déjà sous une forme linéaire.

11.6.2 Représentation d'état du modèle point non-linéaire

Considérons M l'espace d'état du modèle point non-linéaire défini dans 11.4. Il faut voir M comme une surface sur laquelle on peut définir un point de coordonnée x, l'état du système. On définit u un vecteur d'entrée du système, d l'entrée de perturbation du système et f un champ de vecteur paramétré par u et d [Fossard *et al.*, 1993]. On peut alors associer à chaque trio (x, d, u) une direction $f(x, d, u)$ tangente à la surface définie par l'espace d'état M au point x, dans laquelle le système se déplace lors de la sollicitation de u et d.

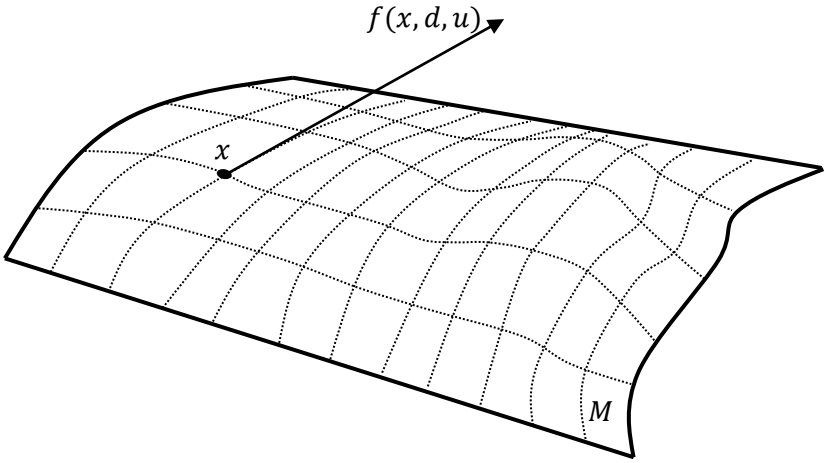


Figure 82: Illustration de l'espace d'état

Soit $\dot{x}$ la dérivée de l'état. On définit alors la représentation d'état du modèle point non-linéaire par la relation suivante :

$$\dot{x} = f(x, d, u) \tag{11.30}$$

L'état x est défini par :

$$x = \begin{pmatrix} T_c & T_f & AO & \rho_{dop} & n & c & T_c^{GV} & T_f^{GV} & P_{bank} & H_{bank} & C_b \end{pmatrix}$$

Le vecteur d'entrées u est :

$$u = (vP_{bank} \quad vH_{bank} \quad u_Q)^T$$

Le vecteur des perturbations d est :

$$d = P_{turb}$$

f représente l'ensemble des équations non-linéaires données dans 11.4.

11.6.3 Linéarisation de la représentation d'état non-linéaire

Dans cette partie nous linéarisons la représentation d'état non-linéaire telle que définie dans les paragraphes précédents 11.6.2.

11.6.3.1 Définition du modèle linéaire tangent

L'établissement d'un modèle linéaire tangent consiste à déterminer un ensemble de systèmes linéaires locaux, aux points d'équilibre du système, lui-même représenté par la fonction f définie dans l'équation (11.30).

Le système non-linéaire défini par la fonction f est dit à l'équilibre si f s'annule, i.e. si $\exists\ (x_0, d_0, u_0) \in M \times \mathbb{R}^m$ tels que :

$$\dot{x} = f((x_0, d_0, u_0) = 0$$

On dit alors que le couple (x_0, u_0) définit l'espace d'équilibre ξ défini tel que :

$$\xi = \{(x_0, d_0, u_0) \in M \times \mathbb{R}^m / f(x_0, d_0, u_0) = 0\}$$

Où m est la dimension du vecteur d'entrées et M l'ensemble de définition de l'état du système.

Selon [D'Andrea-Novel *et al.*, 1994], un point d'équilibre représente un point stationnaire ou un point de fonctionnement du système. Nous choisirons désormais de parler de point de fonctionnement pour décrire tout couple $(x_0, d_0, u_0) \in \xi$.

On définit le point de fonctionnement du système par :

$$x^0 = \begin{pmatrix} T_c^0 & T_f^0 & AO^0 & \rho_{dop}^0 & n^0 & c^0 & T_c^{GV,0} & T_f^{GV,0} & P_{bank}^0 & H_{bank}^0 & C_b^0 \end{pmatrix}$$

et

$$u^0 = \begin{pmatrix} vP_{bank}^0 & vH_{bank}^0 & \overline{u_Q}^0 \end{pmatrix}^T$$

et

$$d^0 = P_{turb}^0$$

Au voisinage de ce point, on peut représenter le comportement du système non-linéaire au premier ordre par le développement de Taylor de l'équation (11.30). On a alors :

$$f(x,d,u) = f(x_0,d_0,u_0) + \frac{\partial f}{\partial x}(x_0,d_0,u_0)(x-x_0) + \frac{\partial f}{\partial u}(x_0,d_0,u_0)(u-u_0) + \frac{\partial f}{\partial d}(x_0,d_0,u_0)(d-d_0) + \sigma(x-x_0, d-d_0, u-u_0)$$

On néglige alors les termes d'ordre supérieurs à 1. Posons δ l'opérateur définissant l'écart d'une variable d'état à son point équilibre. On définit alors :

$$\delta x = x - x_0$$

$$\delta u = u - u_0$$

$$\delta d = d - d_0$$

les variables d'écart de l'état x et du vecteur d'entrée u par rapport au point de fonctionnement considéré et $\delta\dot{x} = f(\boldsymbol{x,d,u}) - f(x_0,d_0,u_0) = \dot{x} - \dot{x}^0$. On peut alors écrire 11.30 sous la forme suivante :

$$\delta\dot{x} = \frac{\partial f}{\partial x}(x_0,d_0,u_0)\delta x + \frac{\partial f}{\partial u}(x_0,d_0,u_0)\delta u + \frac{\partial f}{\partial d}(x_0,d_0,u_0)\delta d \qquad (11.31)$$

Il est important de noter qu'un modèle linéaire tangent n'est pertinent que dans le cas où la linéarisation est effectuée à un point de fonctionnement [Fossard *et al.*, 1993].

Enfin, le modèle linéaire tangent est défini par F, G_d et G_u des fonctions telles que :

$$\delta\dot{x} = F(x_0,d_0,u_0)\delta x + G_d(x_0,d_0,u_0)\delta d + G(x_0,d_0,u_0)\delta u \qquad (11.32)$$

Avec les notations de l'équation (11.31) :

$$F(x_0,d_0,u_0) = \frac{\partial f}{\partial x}(x_0,d_0,u_0)$$

$$G_d(x_0,d_0,u_0) = \frac{\partial f}{\partial d}(x_0,d_0,u_0)$$

$$G_u(x_0,d_0,u_0) = \frac{\partial f}{\partial u}(x_0,d_0,u_0)$$

11.6.3.2 Forme linéarisée du modèle point non-linéaire

L'équation non-linéaire état-état définie dans 11.6.1 se linéarise par série de Taylor au premier ordre, comme nous l'avons montré par l'équation (11.31). La démonstration est reportée en Annexe G.

On déduit alors la forme linéarisée du modèle point non-linéaire autour d'un point de fonctionnement $(x_0, d_0, u_0) \in \xi$, définie comme suit :

$$\frac{d\delta T_c}{dt}(t) = -\frac{1}{\tau_{co}}\delta T_c(t) + \frac{1}{\tau_{co}}\delta T_f(t) + \frac{1}{\tau_{co}}\frac{K_n}{Q_p C_p}\delta n(t) \quad (11.a')$$

$$\frac{d\delta T_f}{dt}(t) = -\frac{1}{\tau_{bf}}\delta T_f(t) + \frac{1}{\tau_{bf}}\delta T_f^{GV}(t) \quad (11.b')$$

$$\frac{d\delta AO}{dt}(t) = -\frac{1}{\tau_{AO}}\delta AO(t) + \frac{1}{\tau_{AO}}\left(\boldsymbol{K_{AO}^{c}}\delta T_c(t) + \boldsymbol{K_{AO}^{f}}\delta T_f(t) + \boldsymbol{K_{AO}^{P}}\delta P_b(t) + \boldsymbol{K_{AO}^{H}}\delta H_b(t)\right) \quad (11.c')$$

$$\frac{d\delta\rho_{dop}}{dt}(t) = -\frac{1}{\tau_{dop}}\delta\rho_{dop}(t) + \frac{\boldsymbol{K_{dop}}}{\tau_{dop}}\delta n(t) \quad (11.d')$$

$$\frac{d\delta n}{dt}(t) = \frac{n^0}{l^*}\rho(t) - \frac{\beta}{l^*}\delta n(t) + \lambda\delta c(t) \quad (11.e')$$

$$\frac{d\delta c}{dt}(t) = \frac{\beta}{l^*}\delta n(t) - \lambda\delta c(t) \quad (11.f')$$

$$\frac{d\delta T_c^{GV}}{dt}(t) = -\frac{1}{\tau_{bc}}\delta T_c^{GV}(t) + \frac{1}{\tau_{bc}}\delta T_c(t) \quad (11.g')$$

$$\frac{d\delta T_c^{GV}}{dt}(t) = -\frac{1}{\tau_{GV}}\delta T_c^{GV}(t) + \frac{1}{\tau_{GV}}\delta T_f^{GV}(t) + \frac{1}{\tau_{GV}}\frac{\delta P_{turb}(t)}{Q_p C_p} \quad (11.h')$$

$$\rho(t) = \rho_{dop}(t) + \rho_{barres}(t) + \rho_{mod}(t) + \rho_{bore}(t) \quad (11.i')$$

$$\rho_{mod}(t) = \boldsymbol{K_c}\delta T_c(t) + \boldsymbol{K_f}\delta T_f(t) \quad (11.j')$$

$$\rho_{barres}(t) = \boldsymbol{K_P}\delta P_b(t) + \boldsymbol{K_H}\delta H_b(t) \quad (11.k')$$

$$\rho_{bore}(t) = \boldsymbol{K_{CB}}\delta C_b(t) \quad (11.l')$$

$$\delta P_K(t) = K_n.\delta n(t) \quad (11.m')$$

$$\frac{d\delta P_b}{dt}(t) = vP_{bank} \quad (11.n')$$

$$\frac{d\delta H_b}{dt}(t) = vH_{bank} \tag{11.o'}$$

$$\frac{d\delta C_b}{dt}(t) = \overline{u_Q} \tag{11.p'}$$

Avec les coefficients en gras (cf. 11.6.1) **identifiés à chaque point de fonctionnement**.

Notons, comme l'avions décrit dans la partie 11.4, par soucis de lisibilité, les équations du retard de bore issues de la modélisation par l'approximation de Laguerre ne sont pas représentées. Rappelons la relation suivante (cf. 11.2.4.2) :

$$\overline{u_Q} = R_B(s).u_Q \tag{11.q'}$$

11.6.3.3 Relation avec le modèle linéaire tangent

A partir des équations (11.a') à (11.q') et de l'équation (11.32), on peut alors exprimer les fonctions F G_d et G_u sous une forme matricielle telle que :

$$F(x_0,d_0,u_0) = \begin{pmatrix} -\frac{1}{\tau_{co}} & \frac{1}{\tau_{co}} & 0 & 0 & \frac{K_n}{C_p Q_n \tau_{co}} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{\tau_{bf}} & 0 & 0 & 0 & 0 & 0 & \frac{1}{\tau_{bf}} & 0 & 0 & 0 \\ \frac{\boldsymbol{K_{AO}^c}}{\tau_{AO}} & \frac{\boldsymbol{K_{AO}^f}}{\tau_{AO}} & -\frac{1}{\tau_{AO}} & 0 & 0 & 0 & 0 & 0 & \frac{\boldsymbol{K_{AO}^P}}{\tau_{AO}} & \frac{\boldsymbol{K_{AO}^H}}{\tau_{AO}} & 0 \\ 0 & 0 & 0 & -\frac{1}{\tau_{dop}} & \frac{\boldsymbol{K_{dop}}}{\tau_{dop} n^0} & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{\boldsymbol{K_c} n^0}{l^*} & \frac{\boldsymbol{K_f} n^0}{l^*} & 0 & \frac{n^0}{l^*} & -\frac{\beta}{l^*} & \lambda & 0 & 0 & \boldsymbol{K_P}\frac{n^0}{l^*} & \boldsymbol{K_H}\frac{n^0}{l^*} & \boldsymbol{K_{CB}}\frac{n^0}{l^*} \\ 0 & 0 & 0 & 0 & \frac{\beta}{l^*} & -\lambda & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{\tau_{bc}} & 0 & 0 & 0 & 0 & 0 & -\frac{1}{\tau_{bc}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{\tau_{GV}} & -\frac{1}{\tau_{GV}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{11.33}$$

$$G_d(x_0,d_0,u_0) = 1 \tag{11.34}$$

$$G_u(x_0,d_0,u_0) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{11.35}$$

11.6.4 Mise sous la forme quasi-LPV du modèle

Définissons tout d'abord ce qu'est une représentation LPV. Considérons le système d'équations non-linéaires G_{NL} suivant :

$$G_{NL} \coloneqq \begin{cases} \dot{x}(t) = f_x(x(t), d(t), u(t)) \\ z(t) = f_z(x(t), d(t), u(t)) \\ y(t) = f_y(x(t), d(t), u(t)) \end{cases}$$

où f_x, f_y et f_z sont des fonctions continues et dérivables, x l'état, u le vecteur d'entrées et d le vecteur des perturbations du système tels que définis en 11.6.2. y les sorties mesurées et z les sorties à contrôler tels que :

$$z = (\delta T_m \quad \delta AO \quad \delta P_{bank})^T$$

$$y = (\delta T_c \quad \delta T_m \quad \delta AO \quad \delta P_K \quad \delta P_{bank})^T$$

Pour mettre sous la forme LPV un tel système non-linéaire il faut trouver une transformation telle qu'il existe un ensemble d'équations différentielles permettant d'écrire le système G_{NL} sous la forme suivante :

$$G_{LPV} \coloneqq \begin{cases} \dot{x}(t) = A(\pi(t))x(t) + B_1(\pi(t))d(t) + B_2(\pi(t))u(t) \\ z(t) = C_1(\pi(t))x(t) + D_{11}(\pi(t))d(t) + D_{12}(\pi(t))u(t) \\ y(t) = C_2(\pi(t))x(t) + D_{21}(\pi(t))d(t) + D_{22}(\pi(t))u(t) \end{cases} \quad (11.36)$$

avec A, B_1, B_2, C_1, C_2, D_{11}, D_{12}, D_{21} et D_{22} les matrices de la représentation d'état, x l'état, z les sorties à contrôler, y les sorties mesurées et ultérieurement utilisées par régulateur, u les commandes, d les perturbations et π la variable séquencement défini par l'équation (11.29).

On déduit cette transformation du modèle linéaire tangent défini par les équations (11.33), (11.34) et (11.35). On a alors :

$$A(\pi) = F(x^0) \quad (11.37)$$

$$B_1(\pi) = B_1 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \qquad B_2(\pi) = B_2 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (11.38)$$

$$C_1(\pi) = C_1 = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \quad (11.39)$$

$$C_2(\pi) = C_2 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & K_n & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \quad (11.40)$$

$$D_{11}(\pi) = D_{11} = \vartheta_{3\times1} \qquad D_{21}(\pi) = D_{21} = \vartheta_{3\times1} \quad (11.41)$$

$$D_{12}(\pi) = D_{12} = \vartheta_{5\times3} \qquad D_{22}(\pi) = D_{22} = \vartheta_{5\times3} \quad (11.42)$$

Seule la matrice A de la représentation LPV (11.36) contient des coefficients variants. G_{LPV} peut donc se représenter comme suit :

$$G_{LPV} \coloneqq \begin{cases} \dot{x}(t) = A(\pi)x(t) + B_1 d(t) + B_2 u(t) \\ z(t) = C_1 x(t) + D_{11} d(t) + D_{12} u(t) \\ y(t) = C_2 x(t) + D_{21} d(t) + D_{22} u(t) \end{cases}$$

Comme précisé dans l'introduction de ce chapitre, nous avons opté pour une paramétrisation par la position des groupes de contrôle Pbank. Ainsi, les coefficients de $A(\pi)$ varient avec la position de cette barre, notée P_{bank}, dans le cœur. P_{bank} étant un état du cœur et paramètre de séquencement on peut parler de **représentation quasi-LPV**. On déduit des équations (11.36)à (11.42) alors les relations :

$$A(\pi) = A(P_{bank})$$

et

$$G_{LPV} \coloneqq \begin{cases} \dot{x}(t) = A(P_{bank})x(t) + B_1 d(t) + B_2 u(t) \\ z(t) = C_1 x(t) + D_{11} d(t) + D_{12} u(t) \\ y(t) = C_2 x(t) + D_{21} d(t) + D_{22} u(t) \end{cases} \quad (11.43)$$

Enfin, rappelons que le modèle de retard de bore R_B, défini dans 11.2.4.2 est inclus séparément de la manière suivante :

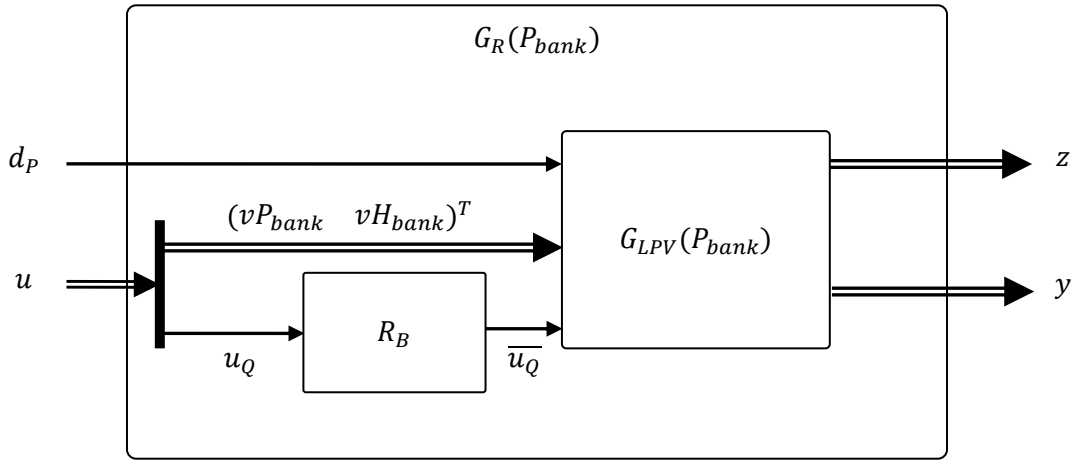


Figure 83: Modèle LPV retardé

11.7 Conclusions sur la modélisation du réacteur

Framatome dispose d'un code de calcul multi-physique du réacteur précis et fiable mais trop complexe pour être utilisé à des fins conceptuelles. La thèse s'est alors intéressé à la conception d'un modèle juste nécessaire pour la commande, comparable à certains modèles existants dans la littérature. Il consiste en un modèle simplifié du circuit primaire, d'un modèle point du cœur et d'un modèle d'AO. L'innovation majeure de ce modèle est l'association d'un modèle d'AO avec un modèle point. L'avantage de ce modèle par rapport aux modèles existants est sa simplicité et sa facilité d'utilisation, en particulier dans le cas de la synthèse de lois commande utilisant un modèle linéaire. En effet, les modèles représentant l'AO les plus simples sont les modèles axiaux et en particulier le modèle bipoints utilisé pour la commande par [Li, 2014], [Bauer et Poncelet, 1974] et [Ukai et Iwazumi, 1994]. Premièrement ce modèle est de dimension plus élevée que le modèle présenté dans la thèse. De plus, le cœur est séparé en deux cœurs distincts échangeant des neutrons des difficultés de modélisation et d'identification se produisent à la frontière entre les deux cœurs lors de l'insertion des grappes dans ces derniers. Problème qui n'est pas rencontré dans le modèle proposé.

12 Conception et paramétrage du Régulateur Multi-Objectif

Sommaire

12.1 Structure du régulateur multi-objectif 218
12.2 Traduction du cahier des charges pour la commande 220
12.2.1 Démarche pour la prise en compte de contraintes temporelles 221
12.2.2 Critère H∞ sur les LCO 224
12.2.3 Critère H∞ sur la robustesse 225
12.2.4 Critères H∞ sur la sollicitation des commandes 226
12.2.5 Critère H2 pour la performance 227
12.3 Formalisation du problème de commande multi-objectif 230
12.3.1 Forme standard de commande 230
12.3.2 Méthode et outils de résolution du problème de commande 233
12.4 Synthèse finale du régulateur à gains-séquencés 233
12.4.1 Synthèse à un point de fonctionnement 233
12.4.2 Séquencement du régulateur et points de fonctionnement considérés 235
12.4.3 Méthode de détermination du régulateur à gains séquencés 237
12.4.4 Simulation et analyse du régulateur à gains séquencés 240
12.5 Conclusions sur la partie régulateur multi-objectif 241

Cette chapitre présente la conception et la simulation du régulateur multi-objectif à gains séquencés. Nous commençons par définir la structure du régulateur utilisée. Ensuite, nous traduisons les exigences du cahier des charges en contraintes et objectifs de commande. Compte tenu de ces éléments, nous formalisons ensuite le problème de commande. Dans un quatrième temps nous synthétisons le régulateur à gains séquencés et présentons les résultats de sa simulation.

12.1 Structure du régulateur multi-objectif

Cette partie montre la structure choisie pour le régulateur multi-objectif ainsi que la paramétrisation de cette structure. Présentons le schéma standard de commande considéré. Il est présenté Figure 84.

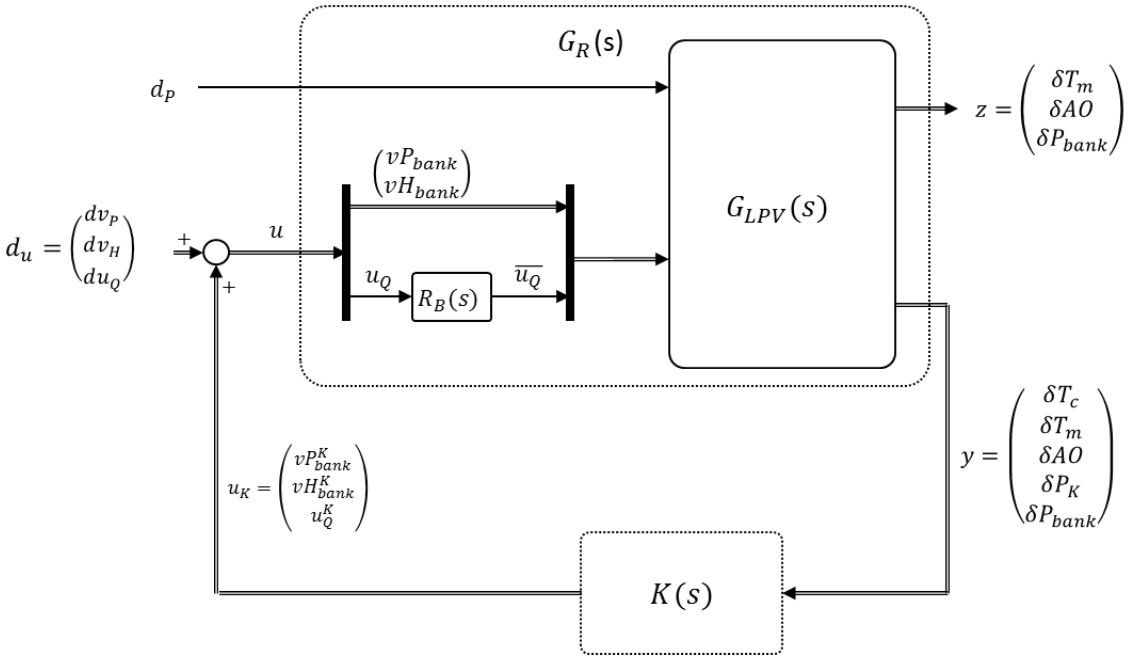


Figure 84: Schéma standard de commande

Nous reprenons ici la forme du régulateur que nous avions introduit dans les travaux [Lemazurier *et al.*, 2017a] et [Lemazurier *et al.*, 2018a]. C'est un régulateur **PI multivariables** (cf. 10.3.1). Il est noté $K(s)$ et se présente sous la forme décrite par la Figure 85. Les raisons qui ont motivé ce choix sont les suivantes. D'une part, l'utilisation d'intégrateurs permet d'assurer une erreur statique nulle. D'autre part, tous points de fonctionnement considérés, le système ne possède que deux paires de pôles complexes conjugués dont l'amortissement le plus faible relevé est de $\xi_{min} = 0.781$. Le système ne présente donc pas de modes oscillants. L'utilisation d'un opérateur dérivé, c'est-à-dire le cas PID, n'est alors pas nécessaire.

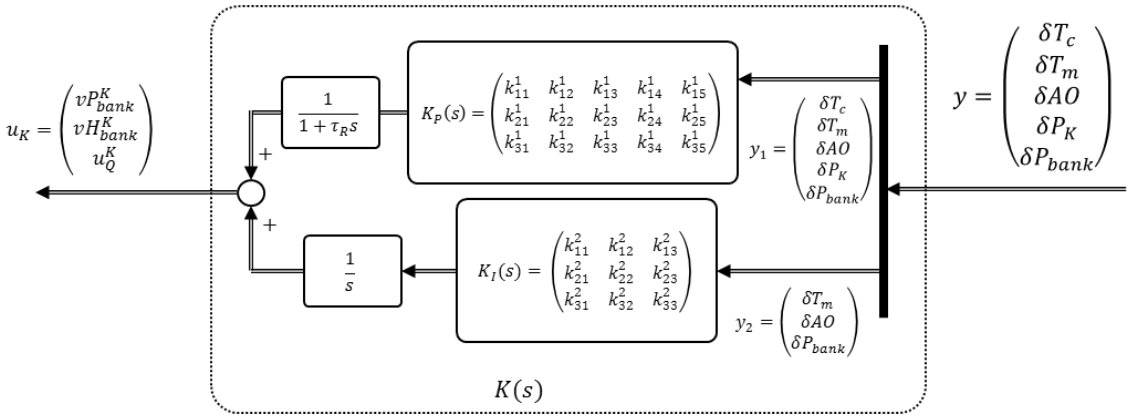


Figure 85: Forme du régulateur multi-objectif

On a alors, $$u_K = K(s).y = \frac{K_P}{1+\tau_R s} y_1 + \frac{K_I}{s} y_2 \quad (12.1)$$

avec $y_1 = y$ $y_2 = z$

et K_P et K_I des matrices de gains, τ_R la constance de filtrage et s la variable de Laplace. Pour rappel, y est le vecteur des sorties mesurées et z les sorties à contrôler (cf. 11.6.4). Cette représentation a l'avantage d'être parcimonieuse, i.e. qu'elle présente un nombre réduit de coefficients tout en étant suffisamment riche pour ne pas obérer la qualité du compromis performance robustesse atteignable. Un nombre réduit de coefficients est aussi significatif d'un nombre réduit de variables de décision pour l'optimisation.

Cette forme a aussi l'avantage d'être lisible de sorte que la lecture des coefficients K_P et K_I permette immédiatement de déterminer le sens physique des actions prises par le régulateur. C'est un point particulièrement important pour l'industriel qui souhaite surveiller que la solution trouvée ne présente pas de comportements indésirables (ordre d'extraction de barres alors qu'une insertion serait de rigueur).

Reprenons la structure de commande choisie. On a y le vecteur des entrées et u_K le vecteur des sorties du régulateur K telles que :

$$y = \begin{pmatrix} \delta T_c \\ \delta T_m \\ \delta AO \\ \delta P_K \\ \delta P_{bank} \end{pmatrix} \qquad u_K = \begin{pmatrix} vP_{bank}^K \\ vH_{bank}^K \\ u_Q^K \end{pmatrix} \qquad z = \begin{pmatrix} \delta T_m \\ \delta AO \\ \delta P_{bank} \end{pmatrix} \tag{12.2}$$

On définit donc les matrices K_P et K_I de la manière suivante :

$$K_P = \begin{pmatrix} k_{11}^1 & k_{12}^1 & k_{13}^1 & k_{14}^1 & k_{15}^1 \\ k_{21}^1 & k_{22}^1 & k_{23}^1 & k_{24}^1 & k_{25}^1 \\ k_{31}^1 & k_{32}^1 & k_{33}^1 & k_{34}^1 & k_{35}^1 \end{pmatrix} \tag{12.3}$$

$$K_I = \begin{pmatrix} k_{11}^2 & k_{12}^2 & k_{13}^2 \\ k_{21}^2 & k_{22}^2 & k_{23}^2 \\ k_{31}^2 & k_{32}^2 & k_{33}^2 \end{pmatrix} \tag{12.4}$$

12.2 Traduction du cahier des charges pour la commande

Dans cette partie nous formalisons les contraintes pour le problème de commande multi-objectif en les reliant aux exigences formulées dans le cahier des charges de la régulation défini en partie 10.2.

Nous commencerons, en 12.2.1 à définir la démarche considérée pour la prise en compte de contraintes temporelles dans le problème de commande, formalisés par

des contraintes H_∞ et présentées dans 12.2.2 et 12.2.4. En 12.2.3 nous présentons le critère de robustesse. Ensuite, en 12.2.5 nous montrerons le critère de performance (fonction objectif) formalisé par une norme H_2.

12.2.1 Démarche pour la prise en compte de contraintes temporelles

La démarche consiste à traduire une contrainte temporelle en une contrainte fréquentielle. Pour ce faire, on cherche un approximant de l'amplitude maximale d'un signal de sortie y d'une fonction de transfert $T_{y\to d}(s) = \frac{y(s)}{d(s)}$ en réponse à une sollicitation connue. On utilise ensuite la norme H_∞ du transfert $T_{y\to d}$, caractérisant le gain pire cas de celui-ci. A partir de ce gain on caractériser l'amplitude maximale du signal de sortie en fonction de l'amplitude du signal créneau pour un état initial stabilisé défini. En effet, si l'état stabilisé nul. En effet, si l'état stabilisé n'est pas nul, le dépassement maximum ne peut être garanti.

Habituellement, la prise en compte de critères temporels est une problématique difficile pour les problèmes de commande [Puyou, 2005]. Dans notre cas, c'est une contrainte majeure du problème de commande. Le cahier des charges spécifie certaines contraintes temporelles devant être respectées. Notamment les LCO (cf. 10.2.1). Par exemple, sur tout scénario, la température moyenne ne doit pas dépasser 1.5°C, i.e. $\forall t,\ |\delta T_m| < 1.5°C$.

Nous cherchons ici un moyen de **caractériser le dépassement maximal de certains paramètres sous une sollicitation connue, considérant un état initial stabilisé**. Par exemple, le dépassement de la température moyenne sous l'effet d'un step de puissance de $\pm$ 10%PN.

Les sollicitations possibles sont données par les scénarii considérés : rampes à 5%PN/min, réglage de fréquence ou encore step de puissance jusqu'à 10%PN. D'après l'expertise de Framatome, le scénario le plus pénalisant (celui qui contraint le plus la régulation) est le **step de 10%PN**.

Nous émettons alors l'hypothèse suivante :

- **Hypothèse :** Si le système en boucle fermée respecte les objectifs de commande sur le scénario : step 10%PN, il respecte les objectifs de commande dans tous les cas considérés.

Considérons un signal d'excitation $d_{10\%}$ le signal représentant le cas le plus pénalisant (step de 10%). On approxime ce signal par un signal créneau de période

T =100 secondes et d'amplitude ΔP_{max} = 10%PN, représenté sur la Figure 86, supposé suffisamment représentatif du créneau.

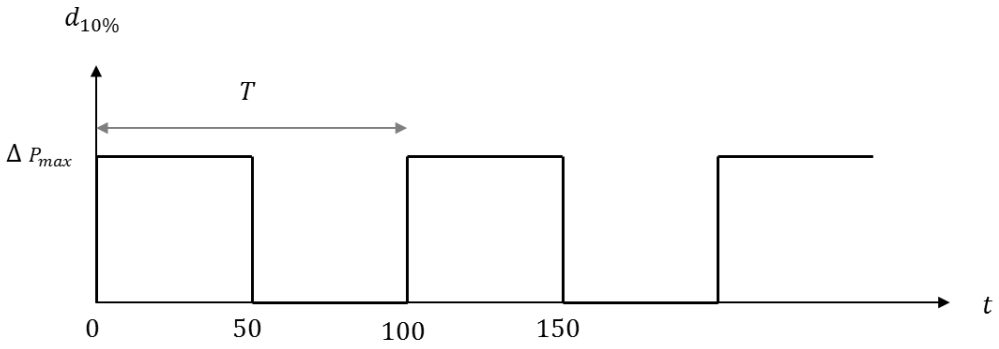


Figure 86: Scénario pire cas de variation de puissance turbine

Décomposons le signal $d_{10\%}$ en série de Fourier, noté s. On obtient le résultat connu pour un créneau suivant :

$$s(t) = \frac{2\Delta P_{max}}{\pi} . \sum_{n=0}^{\infty} \frac{\sin((2n+1)wt)}{2n+1} \qquad (12.5)$$

avec $w = \frac{2\pi}{T}$ et ΔP_{max} l'amplitude du signal $d_{10\%}$ tel que défini sur la Figure 86. Afin de simplifier, nous limiterons l'étude aux trois premières harmoniques du signal supposées suffisamment représentatives. Soit s la décomposition de Fourier au troisième harmonique déduit de (12.5). On a alors :

$$s(t) = \Delta P_{max} \left(\frac{1}{2} + \frac{2}{\pi} \sin(wt) + \frac{2}{3\pi} \sin(3wt) + \frac{2}{5\pi} \sin(5wt) \right) \qquad (12.6)$$

La Figure 87 compare le signal créneau $d_{10\%}$ et le signal s.

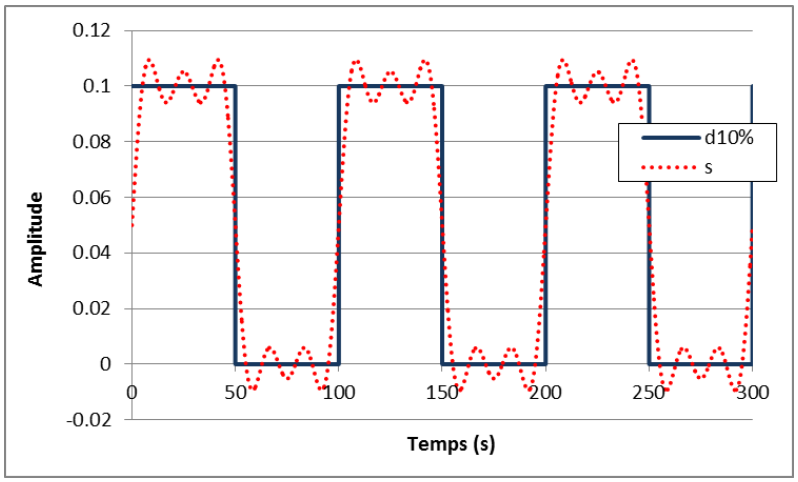


Figure 87: Comparaison signal créneau et décomposition en série de Fourier ordre 3

Par le théorème de superposition, en supposant un état initial nul ($y(t=0)=0$), l'amplitude du signal y de sortie est égale à la somme des amplitudes des harmoniques multipliées par le gain du transfert évalué à la pulsation de chaque harmonique.

Or, si le déphasage induit par la fonction de transfert, entre les trois harmoniques n'est pas trop important, alors l'amplification maximale du signal d'entrée s par la fonction de transfert dans le pire cas, fournit un bon approximant de l'amplitude maximale y_{max} du signal de sortie y. On a alors :

$$\left\|T_{y\to d}\right\|_{\infty} \times s_{max} \sim y_{max}$$

avec s_{max} l'amplitude maximale de s. Or, puisque s est la décomposition en série de Fourier du signal $d_{10\%}$, on a $s \sim d_{10\%}$ et il vient que : $s_{max} \sim \Delta P_{max}$. Puisque l'on cherche D_{max} tel que $y_{max} \leq D_{max}$ il vient que :

$$\left\|T_{y\to d}\right\|_{\infty} \times \Delta P_{max} \leq D_{max} \qquad (12.8)$$

L'équation (12.8) est utilisée par la suite pour l'expression des différents critères H_∞ traduisant les contraintes sur la commande. C'est grâce à celle-ci que nous reformulons les exigences en critères mathématiques.

Vérification de l'hypothèse de faible déphasage

Prenons la fonction de transfert du système bouclé (processus et régulateur), notée $T_{vP_{bank}\to d}$. Son tracé dans le lieu de Bode est reporté Figure 88. Les trois points f1, f2 et f3 correspondent aux gains et phases (notées ϕ_1, ϕ_2 et ϕ_3) associés aux pulsations des trois harmoniques de s ($w_1 = \frac{2\pi}{T}$, $w_2 = \frac{6\pi}{T}$ et $w_3 = \frac{10\pi}{T}$). On observe que les phases des trois harmoniques par $T_{vP_{bank}\to d}$ sont relativement proches (entre $\phi_1 - \phi_2 = 45°$ et $\phi_2 - \phi_3 = 25°$). Le déphasage entre les harmoniques est suffisamment faible pour que l'amplitude du signal de sortie soit dans le pire des cas proche de la norme $\left\|T_{vP_{bank}\to d}\right\|_{\infty} \times vP_{bank}^{max}$ avec vP_{bank}^{max} l'amplitude max de vitesse du groupe Pbank.

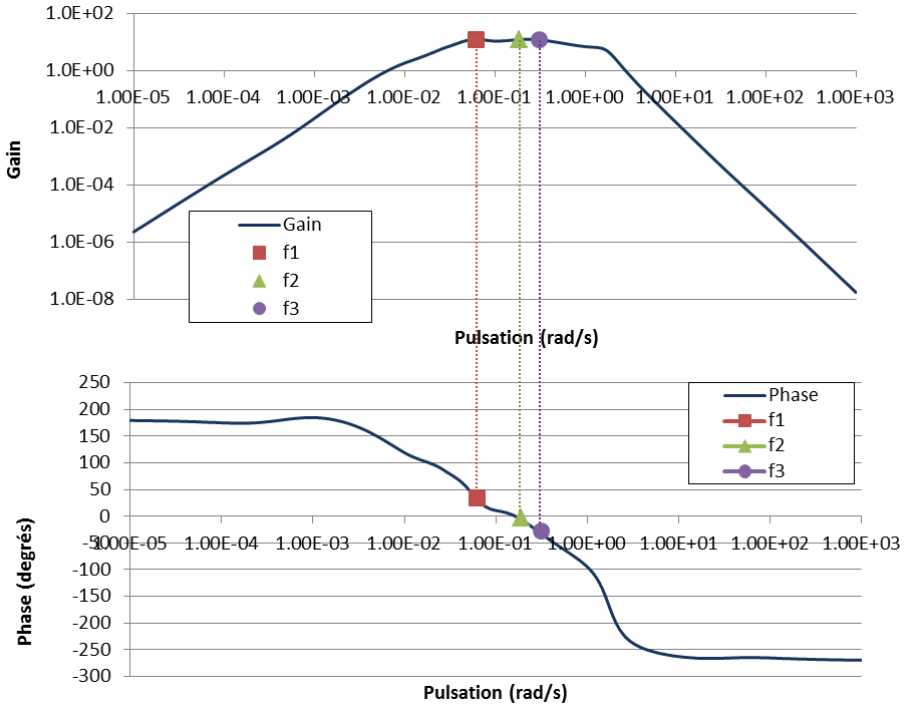


Figure 88: Bode du système considéré

On vérifie que c'est le cas pour tous les signaux auxquels nous appliquerons la démarche par la suite (vP_{bank}, vH_{bank}, u_Q, δT_m, δAO et δP_{bank}). Ainsi, il est possible pour chacun de ces signaux de reformuler les exigences qui les concernent en contraintes formelles mathématiques.

12.2.2 Critère H_∞ sur les LCO

Dans le cas de la température moyenne, on veut garantir que le dépassement de celle-ci soit inférieur à $\Delta T_m^{max} = 1.5\ °C$.

Considérant l'équation (12.8), on pose alors la contrainte :

$$\left\| T_{\delta T_m \to d_P} \right\|_\infty . \Delta P_{max} \leq \Delta T_m^{max}$$

Où d_P est l'entrée de perturbation de puissance turbine (cf. 10.1). On pose :

$$\left\| T_{\delta T_m \to d_P} \right\|_\infty \leq \frac{\Delta T_m^{max}}{\Delta P_{max}}$$

$$\boxed{\left\| W_{T_m} . T_{\delta T_m \to d_P} \right\|_\infty \leq 1}$$

Avec $$W_{T_m} = \frac{\Delta T_m^{max}}{\Delta P_{max}}$$

Dans le cas de l'AO et de la même manière que précédemment, le critère est $\Delta \mathrm{AO}^{\max} = 5\ \%AO$. On pose :

$$\left\| W_{AO}.T_{\delta AO \to d_P} \right\|_\infty \leq 1$$

Avec $$W_{AO} = \frac{\Delta \mathrm{AO}^{\max}}{\Delta P_{max}}.$$

Enfin, pour le dépassement de Pbank et de la même manière que précédemment, on a $\Delta \mathrm{P}_{\mathrm{bank}}^{max} = 30\ pas$. On pose :

$$\left\| W_P.T_{\delta P_{bank} \to d_P} \right\|_\infty \leq 1$$

Avec $$W_P = \frac{\Delta \mathrm{P}_{\mathrm{bank}}^{max}}{\Delta P_{max}}.$$

Ces trois contraintes garantissent chacune le dépassement maximal des sorties à contrôler. Notons cependant que le respect des LCO ne dépend pas uniquement du régulateur multi-objectif, mais du régulateur complet multi-objectif/superviseur.

12.2.3 Critère H_∞ sur la robustesse

La robustesse d'un asservissement traduit la capacité d'un système bouclé à rester stable malgré les erreurs de modèle. Dans le cas multivariable, c'est la marge de module qui est généralement utilisée. Le lecteur trouvera en Annexe C la définition de la marge de module multivariable. Tout d'abord, il nous faut choisir entre la marge de module en entrée ou en sortie du système.

Premièrement, pour tenir compte des non-linéarités d'entrée, liées notamment aux changements de configuration des groupes Pbank et Hbank (cf. 11.2.4.1), nous avons considéré que la marge de module en entrée du système devait impérativement être prise en compte durant la synthèse.

Deuxièmement, selon le conditionnement du système la marge de module en entrée du système peut être équivalente à la marge de module en sortie. Le conditionnement est donné en Annexe D. Dans notre cas, dans les fréquences qui

nous intéressent (entre 0.001 rad/s et 0.1 rad/s, correspondant à des temps de réponse dont l'ordre de grandeur est compris entre 10 secondes et 1000 secondes selon que l'on considère les objectifs de régulations de la température moyenne, l'AO et la position de Pbank) le conditionnement est d'environ 4. Ayant considéré que le conditionnement du système n'était pas critique, nous nous sommes focalisé, durant cette étude de faisabilité, sur un nombre restreint de contraintes, écartant de fait la marge de module en sortie (au bénéfice justement de la marge de module en entrée).

On peut alors définir le critère sur la marge de robustesse en entrée de notre système bouclé par la contrainte suivante :

$$M_m = \frac{1}{\|S\|_\infty} = \frac{1}{\|T_{u \to d_u}\|_\infty} \geq 0.5 = W_2$$

Avec $d_u = (dv_P \quad dv_H \quad du_Q)^T$ le vecteur des bruits sur la commande et $u = (vP_{bank} \quad vH_{bank} \quad u_Q)^T$ le vecteur des commandes réelles du processus. Ce qui donne :

$$\left\| W_R T_{u \to d_u} \right\|_\infty \leq 1$$

Avec $W_R = 0.5$

Cette contrainte garantit les marges de gain et de phase multivariables suivantes :

$$M_g =]m_{gd}\,, \quad m_{ga}[\supset \left]\frac{1}{M_m + 1}\,, \quad \frac{1}{1 - M_m}\right[=]0.67\,, \quad 2[$$

$$M_\phi =]m_{\phi d}\,, \quad m_{\phi a}[\supset]-29°\,, 29°[$$

12.2.4 Critères H_∞ sur la sollicitation des commandes

On limite la vitesse des groupes de grappes Pbank et Hbank à 75 pas/min pour les grappes de contrôle. De la même manière que précédemment on déduit le critère suivant :

- Critère sur la sollicitation des groupes de contrôle Pbank et Hbank

$$\left\| W_K . T_{\binom{vP_{bank}}{vH_{bank}} \to d_P} \right\|_\infty \leq 1$$

Avec $$W_K = \frac{V_{barres}^{max}}{\Delta P_{max}}$$

En ce qui concerne le débit de borication/dilution, en se plaçant à une concentration en bore C_b donnée, on peut alors déduire la commande maximale possible en bore. La relation est donnée dans l'Annexe E. Supposons être à une concentration de $C_b = 1000$ ppm. On a alors :

- Cas dilution : $u_Q^{max} = -Q_{dil}^{max} . \left(\frac{C_b}{M_T}\right) = -3.13.10^{-3}$ ppm/s
- Cas borication : $u_Q^{max} = Q_{bor}^{max} . \left(\frac{C_{b}max - C_b}{M_T}\right) = 8.54.10^{-2}$ ppm/s

La borication étant plus efficace que la dilution, on prend le cas le plus pénalisant, on a alors :

$$u_Q^{max} = 3.13.10^{-3} \text{ ppm/s.}$$

On en déduit le critère sur l'actionneur de bore suivant:

- Critère sur la sollicitation de la commande de bore/eau

$$\left\| W_Q . T_{u_Q \to d_P} \right\|_\infty \leq 1$$

Avec $$W_Q = \frac{u_Q^{max}}{\Delta P_{max}}$$

12.2.5 Critère H_2 pour la performance

Nous entendons par critère performance l'objectif d'optimisation du problème de commande. Dans notre étude, l'idée est de minimiser l'impact de la perturbation d_P sur les sorties à contrôler, données dans z. Pour caractériser la performance nous avons choisi un objectif H_2. Les raisons sont les suivantes :

- Le régulateur multi-objectif a pour but de compenser les variations de puissance dues au réglage de fréquence. Le réglage de fréquence, composé du réglage primaire et secondaire, est un signal aléatoire de puissance, non déterministe, composé d'un large spectre de fréquences.

Dans notre étude, il sera assimilé à un bruit blanc. La norme H_2 se prête donc bien pour ce type d'entrée exogène.

- Pour les variations en rampe de puissance, l'utilisation de cette norme convient bien car elle permet de prendre en compte une large gamme de fréquences.

Les pondérations choisies pour chaque voie sont définies par :

- W_{z_1} La pondération fréquentielle pour la sortie de température moyenne
- W_{z_2} La pondération fréquentielle pour la sortie de l'AO
- W_{z_3} La pondération fréquentielle pour la sortie de position de Pbank

Elles sont définies comme suit :

$$W_{z_1}(s) = \frac{K_1}{s}.\frac{\tau_1 s+1}{\tau_2 s+1}.\frac{1}{\tau_3 s+1} \quad (12.9)$$

$$W_{z_2}(s) = \frac{K_2}{s} \quad (12.10)$$

$$W_{z_3}(s) = \frac{K_3}{s} \quad (12.11)$$

Le Tableau 8 contient les valeurs prises pour les pondérations fréquentielles.

Coefficient	Valeur
K_1	50
K_2	5
K_3	1
τ_1	10^5
τ_2	10^4
τ_3	1

Tableau 8: Valeurs des coefficients des pondérations fréquentielles

Trois détails sont à remarquer dans le choix des filtres de pondération :

Le choix des gains K_1, K_2 et K_3. Ils sont choisis en accord avec les dépassements maxima autorisés (cf. 10.2.1). Au départ ils sont normalisés sur le dépassement maximal autorisé. Le dépassement maximal de δT_m est de 1.5°C, de δAO de 5%AO et de δP_{bank} de 30 pas. On normalise sur δP_{bank}, on pose alors $K_3 = 1$. On pose alors $K_2 = \frac{30}{5} = 6$ le coefficient de δAO et $K_1 = \frac{30}{1.5} = 20$ le coefficient de δT_m. Ces valeurs sont des valeurs d'initialisation. Ils sont réglés ensuite en simulation pour régler le contrôleur.

- Les constantes de temps τ_1 et τ_2 sont choisies grandes devant le retard de bore de manière à pénaliser les gains basse fréquence de la température moyenne. Et $\tau_1 > \tau_2$ pour augmenter le gain et donc pénaliser le transfert liant la perturbation à la température. Principalement en raison de la priorisation de la température moyenne (cf. 10.2.5).
- Le nombre de pôles du transfert $T_{\delta T_m \to d_P}$ est plus faible que ceux de $T_{\delta AO \to d_P}$ et $T_{\delta P_{bank} \to d_P}$. Afin de pénaliser de la même manière en haute fréquence les trois transferts, on rajoute un filtre passe-bas de constante $\tau_3 = 1$.

On se sert alors de la norme H_2 pour définir le critère à minimiser pour le problème de commande comme suit :

$$\min_{K \in \Omega} \left\| W_z . T_{z \to d_P} \right\|_2$$

Où, $z = (\delta T_m \quad \delta AO \quad \delta P_{bank})^T$ est le vecteur des sorties à contrôler, d_P est la perturbation (puissance turbine) et W_z la pondération fréquentielle définie par :

$$W_z(s) = \begin{pmatrix} W_{z_1}(s) & 0 & 0 \\ 0 & W_{z_2}(s) & 0 \\ 0 & 0 & W_{z_3}(s) \end{pmatrix} \qquad (12.12)$$

12.3 Formalisation du problème de commande multi-objectif

Compte tenu des contraintes et objectifs de commande précédemment exposés, nous présentons, dans cette partie, le problème de commande multi-objectif.

12.3.1 Forme standard de commande

L'ensemble des objectifs de commande définis précédemment sont formalisés sur la Figure 89, représentant la forme standard de commande. On admettra les notations suivantes :

u_K	Entrée de commande provenant du régulateur K
d_P	Entrée de perturbation – puissance turbine
d_u	Entrée de perturbation sur les entrées de commande
z_i	Sorties pour le problème de commande ($i \in [1,7]$)
y	Sorties mesurées

Le Tableau 9 récapitule la reformulation des exigences du *Core Control* en contraintes mathématiques formelles sur le problème de commande considéré.

Numéro de l'exigence	Enoncé	Critère associé
E0005	Minimisation de l'impact de la puissance turbine sur les sorties T_m, AO et P_{max}	$\min_{K\in\Omega} \left\| W_z . T_{z\to d_P} \right\|_2$
E0030	Garantir une marge de module d'au moins 0.5.	$\left\| W_R T_{u\to d_u} \right\|_\infty \leq 1$
E0025	La sollicitation de P_{bank} et H_{bank} ne peut excéder 75 pas/min	$\left\| W_K . T_{\binom{vP_{bank}}{vH_{bank}}\to d_P} \right\|_\infty \leq 1$
E0027	La sollicitation du débit de Bore ne peut excéder $4.7.10^{-3}$ ppm/s.	$\left\| W_Q . T_{u_Q\to d_P} \right\|_\infty \leq 1$
E0016	Le dépassement de la température moyenne ne doit pas excéder 1.5°C en valeur absolue	$\left\| W_{T_m} . T_{z_{T_m}\to d_P} \right\|_\infty \leq 1$
E0017	Le dépassement d'AO ne doit pas excéder 5% d'AO en valeur absolue	$\left\| W_{AO} . T_{z_{AO}\to d_P} \right\|_\infty \leq 1$

E0018	Le dépassement de Pbank ne doit pas excéder 30 pas en valeur absolue	$\left\|W_P.T_{z_{Pbank}\to d_P}\right\|_\infty \leq 1$

Tableau 9 : Récapitulatif des reformulations des exigences sur le *Core Control*

Finalement, on pose le problème de commande sous forme du problème d'optimisation suivant :

$$\wp(K \in \Omega) \coloneqq \begin{cases} \min_{K\in\Omega}\left\|W_z.T_{z\to d_P}\right\|_2 \\ \left\|W_R T_{u\to d_u}\right\|_\infty \leq 1 \\ \left\|W_K.T_{u_{bank}\to d_P}\right\|_\infty \leq 1 \\ \left\|W_Q.T_{u_Q\to d_P}\right\|_\infty \leq 1 \\ \left\|W_{T_m}.T_{z_{T_m}\to d_P}\right\|_\infty \leq 1 \\ \left\|W_{AO}.T_{z_{AO}\to d_P}\right\|_\infty \leq 1 \\ \left\|W_P.T_{z_{Pbank}\to d_P}\right\|_\infty \leq 1 \end{cases} \quad (12.13)$$

Notons que $K \in \Omega$, Ω étant l'ensemble des régulateurs stabilisants, respectant les contraintes du problème de commande et de forme décrites par l'équation (12.1). D'après 3.3.1 ce problème de commande est **non-lisse** et **non-convexe**. Le prochain chapitre fait l'objet de la résolution du problème d'optimisation ainsi formulé.

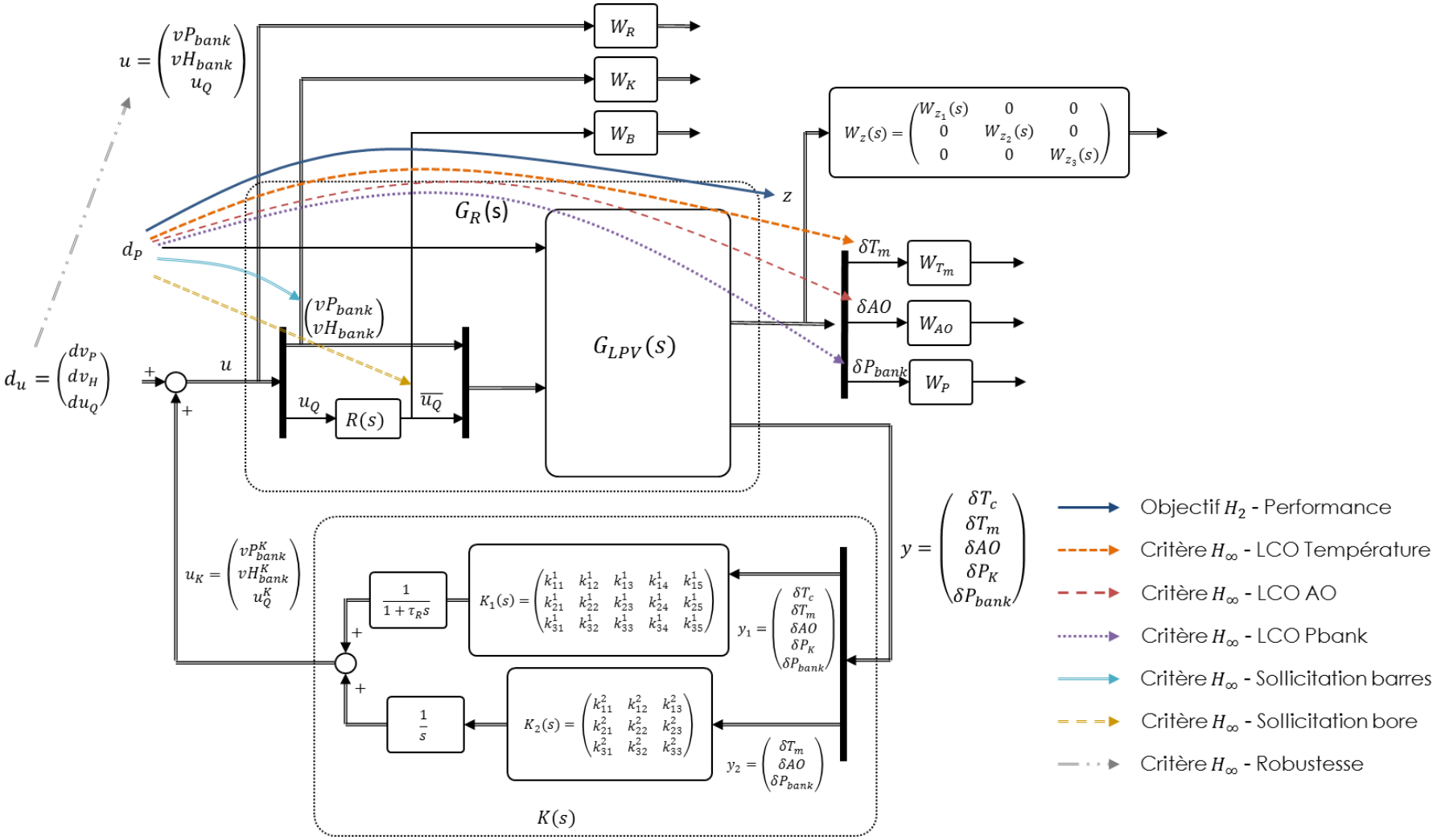


Figure 89: Modèle augmenté multi-objectif pour le problème H_2/H_∞

12.3.2 Méthode et outils de résolution du problème de commande

Afin de résoudre ce problème nous avons utilisé les travaux de [Apkarian et Noll, 2006a], [Apkarian *et al.*, 2007a], [Apkarian *et al.*, 2008], [Apkarian, 2013], [Apkarian *et al.*, 2007b] et [Bruisma et Steinbuch, 1990] utilisant l'optimisation non-lisse pour résoudre des problèmes de commande. Les avantages de cette approche sont donnés dans 3.3.1. L'outil que nous avons utilisé, implémentant la méthode de résolution cité plus haut est *Systune* de *Matlab*. Cet outil possède un environnement complet intégré permettant d'exprimer les contraintes H_2 et H_∞ tels que décris dans le problème de commande (12.13).

12.4 Synthèse finale du régulateur à gains-séquencés

Cette partie a pour but d'appliquer la méthode de synthèse multi-objectif décrite dans la partie précédente (cf. 12.3) sur un maillage des points de fonctionnement possibles, ceci pour ensuite interpoler les coefficients des matrices de gains constitutifs des régulateurs et former un régulateur à gains séquencés valable sur l'ensemble du domaine de fonctionnement.

Dans un premier temps nous décrivons la synthèse à un point de fonctionnement donné et analysons les résultats. Ensuite, nous présentons le principe de la méthode de séquencement mise en œuvre. Ensuite, nous appliquons la méthode de synthèse des régulateurs multi-objectif sur l'ensemble du domaine de fonctionnement. Enfin, nous présentons la simulation du régulateur interpolé.

12.4.1 Synthèse à un point de fonctionnement

Dans cette partie nous donnons les résultats d'une synthèse à un point de fonctionnement. Ce point correspond aux caractéristiques suivantes :

- Puissance : 80%PN
- Position de Pbank initiale : 809 PE
- Position de Hbank : 385 PE
- Température moyenne : 312°C
- Axial Offset : 3%AO

L'algorithme *Systune* trouve avec succès une solution au problème de commande multi-objectif. Toutes les contraintes sont vérifiées et l'objectif de commande minimisé.

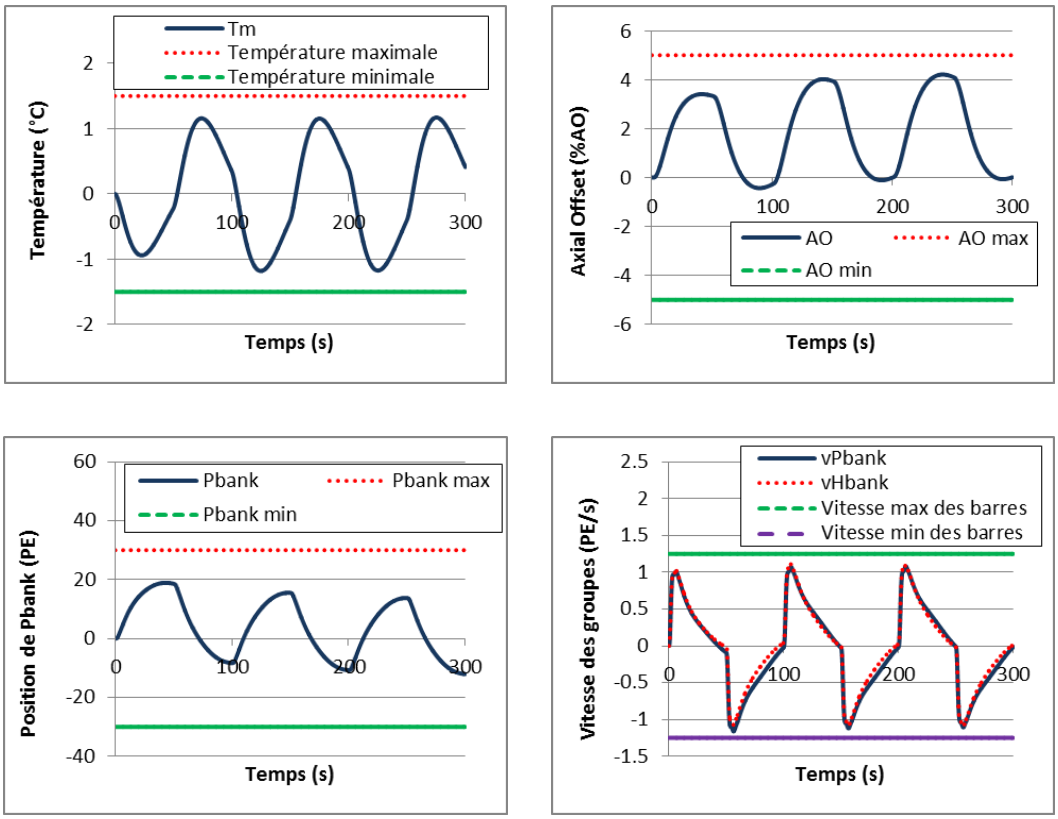


Figure 90: Résultats de simulation d'un régulateur multi-objectif LTI autour d'un point de fonctionnement

Les courbes de simulation données ci-dessus montrent les résultats de simulation d'un régulateur LTI multi-objectif synthétisé autour du point de fonctionnement dont les caractéristiques sont précisées plus haut. Le signal d'excitation est le signal le plus pénalisant considéré dans 12.2.1 : un signal créneau d'amplitude 10%PN et de période 100 secondes.

Ces courbes montrent que :

- La température moyenne (δT_m) est correctement contrôlée car cette grandeur ne dépasse pas son domaine de fonctionnement autorisé défini dans 12.2.2, sur le scénario le plus pénalisant.
- De même pour l'AO (δAO).
- De même pour la position du groupe Pbank (δP_{bank}).
- De plus, la vitesse de variation des groupes Pbank et Hbank (vP_{bank} et vH_{bank}) est bien inférieure aux bornes de dépassement maximales définies dans 12.2.4 pour les deux actionneurs.
- Enfin, les courbes de simulation ne montrent pas le débit de borication/dilution (u_Q). Cependant, le débit respecte bien les exigences spécifiées dans 12.2.4.

En conclusion, la simulation du système bouclé montre que les contraintes qui ont été formulées dans le problème de commande sont bien prises en compte autour d'un point de fonctionnement donné.

12.4.2 Séquencement du régulateur et points de fonctionnement considérés

Cette partie vise à expliquer les choix retenus pour le séquencement du régulateur interpolé. Comme annoncé dans 10.3.3, et à rebours du choix opéré dans [Bendotti *et al.*, 1994] qui préconise un séquencement du régulateur par la puissance, nous considérons que la puissance seule rend insuffisamment compte des variations comportementales liées aux non-linéarités du modèle et qu'il convient alors de considérer en sus les positions des groupes de grappes Pbank et Hbank.

Pbank seul paramètre de séquencement

Pour des raisons de simplification du problème de commande, nous cherchons cependant à réduire le plus possible le nombre de paramètres de séquencement.

- Premièrement, lors des variations de puissance la position de Hbank varie lentement comparée à celle de Pbank. Les effets de la variation de Pbank seront donc prépondérants vis-à-vis de ceux de Hbank.
- Ensuite, comme indiqué précédemment (cf. 11.2.4.1) la position du groupe Pbank conditionne davantage certains comportements non-linéaires que la puissance.

Parmi les trois paramètres candidats (position de Pbank, la puissance du cœur et la position de Hbank), il conviendrait alors de privilégier Pbank pour le séquencement. L'idée est d'identifier les modèles LTI au fur et à mesure de l'insertion du groupe Pbank selon son profil d'insertion (cf. 10.2.5). A chaque position de Pbank correspond une position de Hbank et une puissance. Ainsi, pour un état stabilisé, les positions de Pbank et Hbank ainsi que la puissance du cœur sont liés, comme montré sur la Figure 91.

Pour conclure, le séquencement sera opéré à partir de la seule position de Pbank notée P_{bank}, considérant que les deux autres paramètres (puissance et Hbank) évoluent selon l'équilibre du cœur.

Détermination des points de fonctionnement

Nous avons procédé de la manière suivante : à partir du code de simulation réaliste de Framatome, nous avons identifié en régime statique, à différents niveaux de puissance la position des groupes Pbank (selon le profil de position de référence

donné en 10.2.5) et Hbank (en gardant l'AO sur sa référence pour déterminer une position fidèle de Hbank). La Figure 91 montre l'ensemble des points de fonctionnement que nous avons considérés.

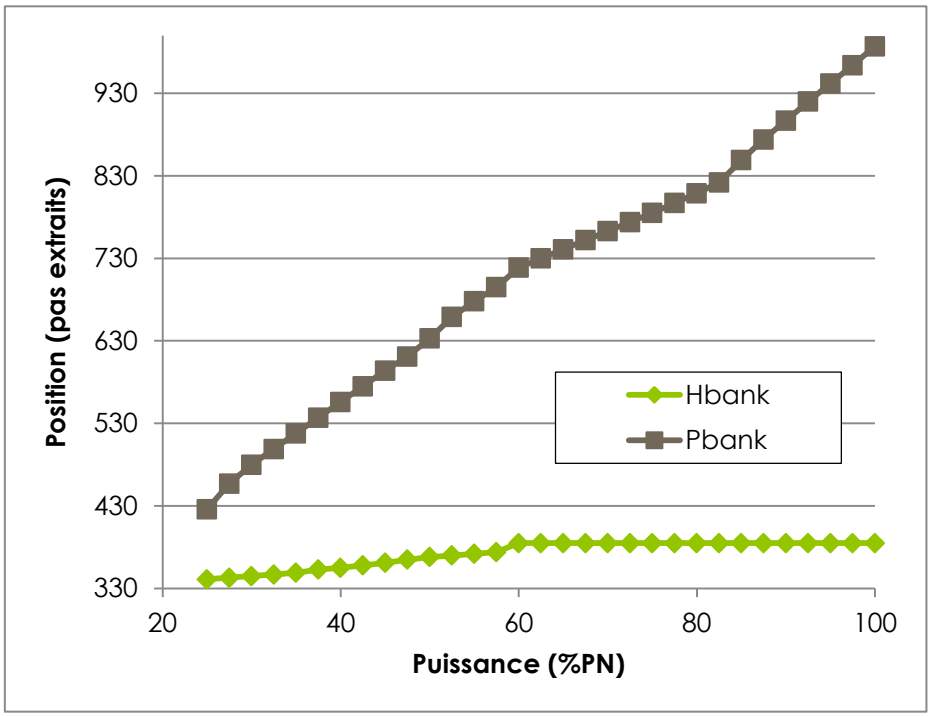


Figure 91: Position de référence des barres Pbank et Hbank pour le séquencement du modèle quasi-LPV

Pour des raisons de simplicité nous nous sommes principalement intéressés au domaine de fonctionnement **60%PN à 100%PN pour toutes les simulations** que nous verrons plus loin. En effet, la zone la plus critique s'avère être entre 90%PN et 100%PN. Entre 60%PN et 25%PN le comportement non-linéaire est moins marqué. On part donc du principe que si le domaine 60%PN à 100%PN est couvert, le reste doit être réalisable en appliquant la même démarche.

Limitations de l'approche

Considérer la position de Pbank comme variable de séquencement est une approche limitative. En effet, pour une position de Pbank donnée, plusieurs positions de Hbank peuvent en réalité être rencontrées pour les raisons suivantes :

- Lors des variations rapides de puissance la position de Pbank varie. Transitoirement, la position de Hbank ne correspond alors pas à l'équilibre sur lequel il a été identifié.
- L'AO de référence est changé par l'utilisateur.
- Les variations spatiales de xénon ont fait évolué l'AO.
- L'épuisement a fait varier la répartition de puissance dans le cœur.

Pour contrer ces effets sans considérer formellement les positions de Pbank et de Hbank et de la puissance, nous avons adopté une approche de *synthèse robuste par zone* que nous développons dans la partie 12.4.3. Cette approche permettra de synthétiser un régulateur vérifiant les contraintes spécifiées dans le problème de commande (cf. 12.3) pour plusieurs modèles de la zone considérée simultanément. De cette manière différentes configurations de Pbank et Hbank seront prises en compte pour chacun des régulateurs LTI support du séquencement. Nous comptons aussi sur la robustesse pour assurer la stabilité en dépit de ces erreurs de modélisation.

12.4.3 Méthode de détermination du régulateur à gains séquencés

Cette partie a pour but de présenter la méthode de détermination des régulateurs LTI sur l'ensemble du domaine de fonctionnement.

En pratique, l'interpolation de régulateurs LTI synthétisés à différents points de fonctionnement peut poser certaines difficultés. Nous nous sommes affranchis de certaines d'entre elles, en choisissant une structure de régulateur parcimonieuse. La structure retenue limite les risques de sur-paramétrisation, qui conduirait à la possibilité d'obtenir un régulateur identique du point de vue comportemental avec des jeux de paramètres différents. Même ainsi, des difficultés peuvent subsister, selon l'espacement entre les modèles LTI du maillage considéré, les critères d'optimisation choisis, voire le choix réalisé pour l'étape d'initialisation. Le problème survient notamment lorsque la synthèse de deux régulateurs à deux points de fonctionnement voisins ne fournit pas des régulateurs suffisamment proches l'un de l'autre. Dans ce cas, le régulateur interpolé entre ces deux points de fonctionnement ne constitue nécessairement une solution viable, soit qu'elle puisse conduire à une instabilité, ou encore qu'elle induise des transitoires trop brutaux lors du changement de coefficients.

Pour pallier ce problème nous adoptons la stratégie suivante :

- La synthèse à un point de fonctionnement consistera en une approche multi-modèle par *zone*. Le régulateur résultant vérifiera les contraintes et sera optimal pour l'ensemble des modèles considérés.
- Le régulateur optimisé à un point de fonctionnement sera utilisé pour initialiser l'optimisation au point de fonctionnement voisin.
- Enfin, nous contraindrons la variation des coefficients des matrices du régulateur de manière à garder ses coefficients suffisamment proches de ceux du régulateur initial (i.e. le régulateur voisin).

Les sections suivantes présenteront la méthode de synthèse multi-modèle par zone ainsi que la méthode d'initialisation des régulateurs.

12.4.3.1 Synthèse multi-modèles par zone

Une *zone* telle que définie ici est constituée de plusieurs modèles associés à un point de fonctionnement et ses voisins. La Figure 92 illustre une synthèse par zone limitées à 3 modèles. En pratique, nous avons considéré 9 modèles par zone. Pour l'exemple illustratif, chaque régulateur K_i est synthétisé au point de fonctionnement i pour l'ensemble des modèles $\mathcal{M} = \{M_{i-1}, M_i, M_{i+1}\}$.

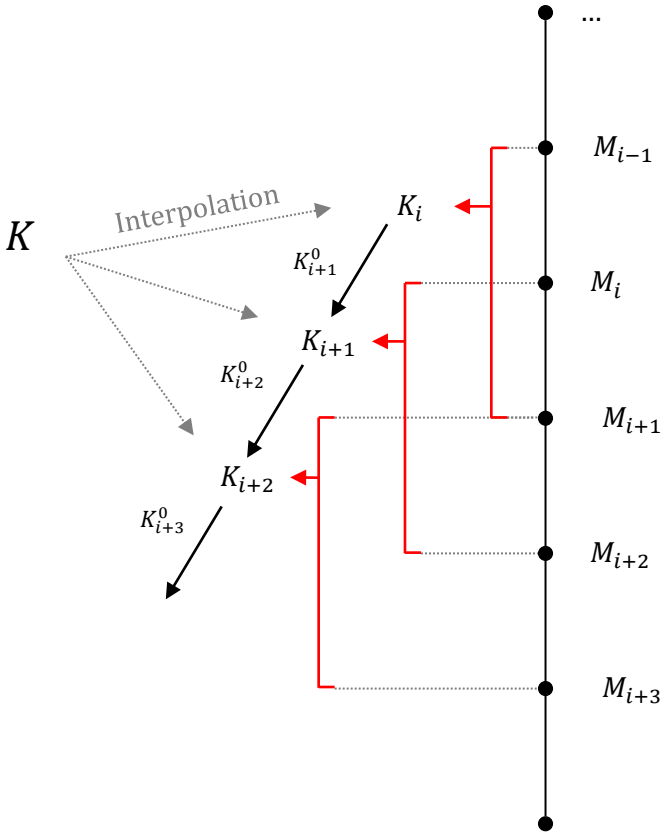


Figure 92: Synthèse multi-modèle par zone

12.4.3.2 Initialisation des régulateur

Nous avons décidé d'initialiser chaque synthèse multi-modèle à partir du régulateur obtenu à la synthèse précédente, i.e. au point de fonctionnement précédent, comme illustré sur la Figure 93. . Chaque régulateur K_i constitue le régulateur initial pour la synthèse du régulateur K_{i+1}. Autrement dit :

$$\forall i \geq 1, K_{i+1}^0 = K_i$$

Reste le choix du régulateur initial K_1^0. Ce régulateur sera calculé par la résolution d'un problème LQ autour du premier point de fonctionnement choisi. Ce problème LQ ne considère pas l'ensemble des contraintes du problème complet. On l'obtient par résolution directe d'une équation de Riccati. L'initialisation est dès lors déterministe.

La Figure 93 récapitule la démarche de synthèse du régulateur LPV.

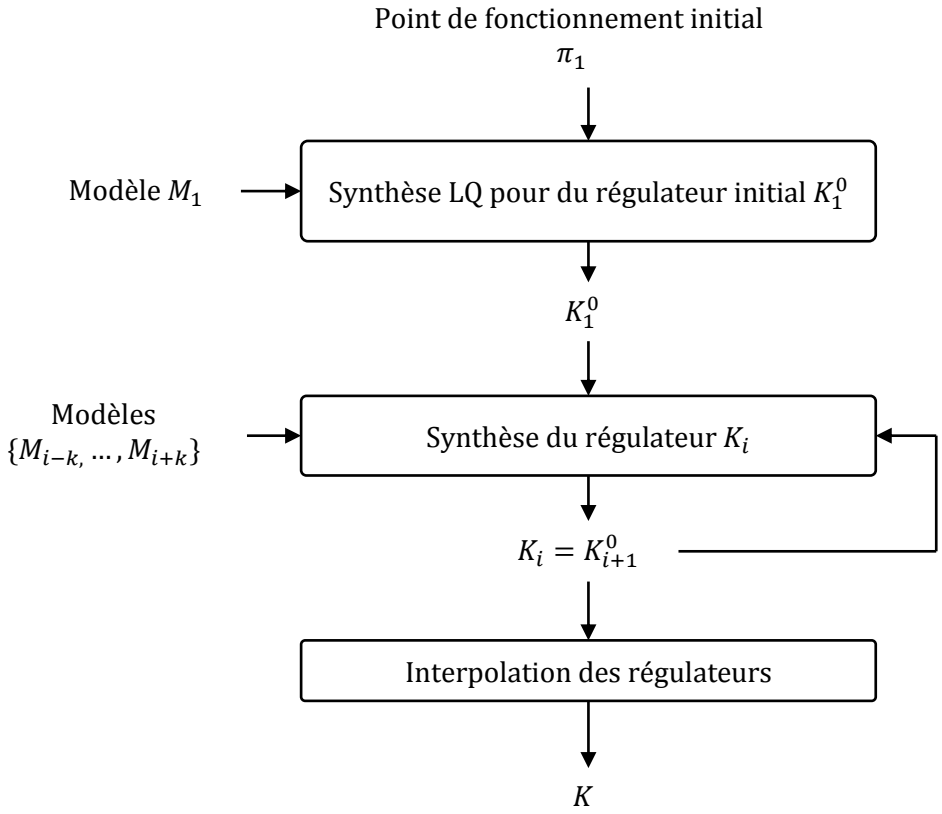


Figure 93: Séquences pour la synthèse LPV

12.4.3.3 Filtrage de la position de Pbank pour le séquencement du régulateur

De manière à éviter une variation rapide des coefficients du régulateur, la position de Pbank servant au séquencement du régulateur est filtrée. On définit $F_{P_{bank}}(s)$ le filtre du premier ordre de la forme suivante :

$$F_{P_{bank}}(s) = \frac{1}{1+\tau_P s}$$

La valeur de la constante de temps τ_P est choisie de manière à pouvoir filtrer suffisamment les variations rapides de barres sans pour autant obérer les

performances du régulateur par une adaptation trop lente des coefficients. On choisit $\tau_P = 50$ secondes montrant un bon compromis en simulation.

Ainsi, le régulateur K depend de la position de Pbank filtrée par $F_{P_{bank}}$ notée P_{bank}^F. La Figure 94: illustre le principe de séquencement du régulateur K à gains séquencés.

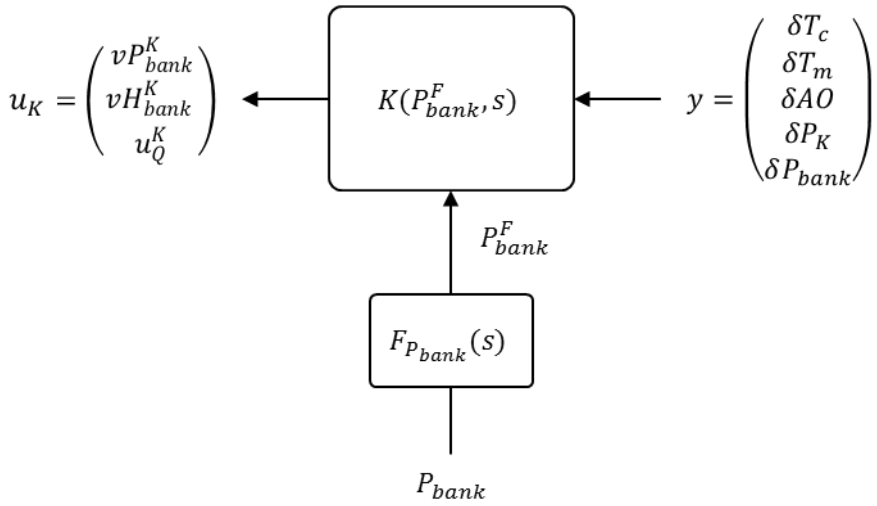


Figure 94: Schéma de séquencement du régulateur à gains séquencés K

12.4.4 Simulation et analyse du régulateur à gains séquencés

Dans cette partie nous présentons les résultats de la simulation du régulateur à gains séquencés. Pour la simulation, nous avons utilisé le modèle non-linéaire point du cœur développé dans la thèse et présenté en partie 11.4.

Le scénario de référence suivant est considéré pour la simulation du régulateur à gains séquencés : rampe de 5%PN/min entre 100%PN et 60%PN superposée à un signal de réglage de fréquence. Nous testons le régulateur sur ce scénario car c'est le scénario de référence. Nous pourrons ainsi observer le comportement du régulateur sur une rampe et sur du réglage de fréquence. Les courbes de simulation sont présentées en Annexe I.

La simulation montre que le comportement du régulateur multi-objectif est conforme aux attentes :

- La température moyenne est maintenue dans le domaine autorisé défini par les LCO.
- De même pour l'AO.
- De même pour la position de Pbank.

Le Tableau 10 résume les écarts maximaux des sorties à contrôler par rapport à leurs références en valeur absolue.

Critères		Observations	Ecart max par rapport à la référence
LCO	T_m	Contrôle correct de la température moyenne	$\lvert\Delta T_m^{max}\rvert = 0.56\ °C \leq 1.5\ °C$
	AO	Contrôle correct de l'AO mais à la limite du domaine autorisé	$\lvert\Delta AO^{max}\rvert = 5.06\ \%AO \sim 5\ \%AO$
	P_{max}	Contrôle correct de la position de Pbank	$\lvert\Delta P_{bank}^{max}\rvert = 14.3\ PE \leq 30\ PE$

Tableau 10: Résultats superviseur seul - Rampe de puissance 100%PN à 60%PN à 5%PN/min

Remarques

- Puisque nous utilisons le modèle non-linéaire point, le xénon est simulé. On remarque sur les simulations que, bien que le modèle utilisé pour synthétiser le régulateur multi-objectif ne contienne pas de modèle de xénon, le régulateur parvient tout de même à rejeter les perturbations en provenant.
- Notons que le faible dépassement de l'AO est considéré maitrisé. Si on veut respecter scrupuleusement ce critère il faut alors prendre des marges sur les critères pour tenir compte des approximations faites, notamment dues à la linéarisation et la prise en compte des contraintes temporelles (cf. 12.2.1).

12.5 Conclusions sur la partie régulateur multi-objectif

Dans cette partie nous avons développé un régulateur multi-objectif à gains séquencés établi sur un modèle quasi-LPV du réacteur. Ce régulateur a été simulé sur le modèle non-linéaire du réacteur présenté en 11.5. En simulation, le régulateur multi-objectif montre des performances correctes vis-à-vis du cahier des charges. Il montre aussi la capacité de rejeter l'effet non-modélisé du xénon.

Cependant, en l'absence d'anticipation de l'évolution du Xénon, le contrôle de l'AO, est fortement impacté. En pratique, l'utilisation d'un modèle d'évolution de la concentration en Xenon à des fins de pré-compensation permettrait de solutionner le problème, et nous la préconisons. Enfin, deux versions simplifiées de ce régulateur, utilisant pour l'une uniquement les groupes de grappes (cf. 10.3.1) et pour l'autre la variation de débit primaire (cf. 10.3.2) ont été testées sur le code de simulation précis et qualifié de Framatome. La première a donné lieu à une publication interne à Framatome [Lemazurier *et al.*, 2016]. Quant à la version utilisant le débit la

publication [Lemazurier *et al.*, 2018a] contient les résultats directement sur code de simulation de Framatome. Dans les deux cas les simulations corroborent les résultats sur modèle simplifié non-linéaire. L'ensemble des contraintes sont respectées et la performance du régulateur est considérée très intéressante.

13 Conception et paramétrage du Superviseur

Sommaire

13.1 Modèle non-linéaire utilisé pour la commande du superviseur 245
13.2 Formalisation du problème de commande 246
13.2.1 Réglage du superviseur 246
13.2.2 Définition des commandes considérées 247
13.2.3 Définition de la fonction objectif du superviseur 249
13.2.4 Définition des contraintes du superviseur 251
13.3 Algorithme développé 252
13.3.1 Solveur utilisé 252
13.3.2 Principe de fonctionnement de l'algorithme 253
13.4 Simulation et Analyse du superviseur 254
13.4.1 Suivi de rampe à 5%PN/min superviseur seul 254
13.4.2 Rampe et réglage de fréquence superviseur seul 255
13.5 Conclusions sur la partie superviseur 256

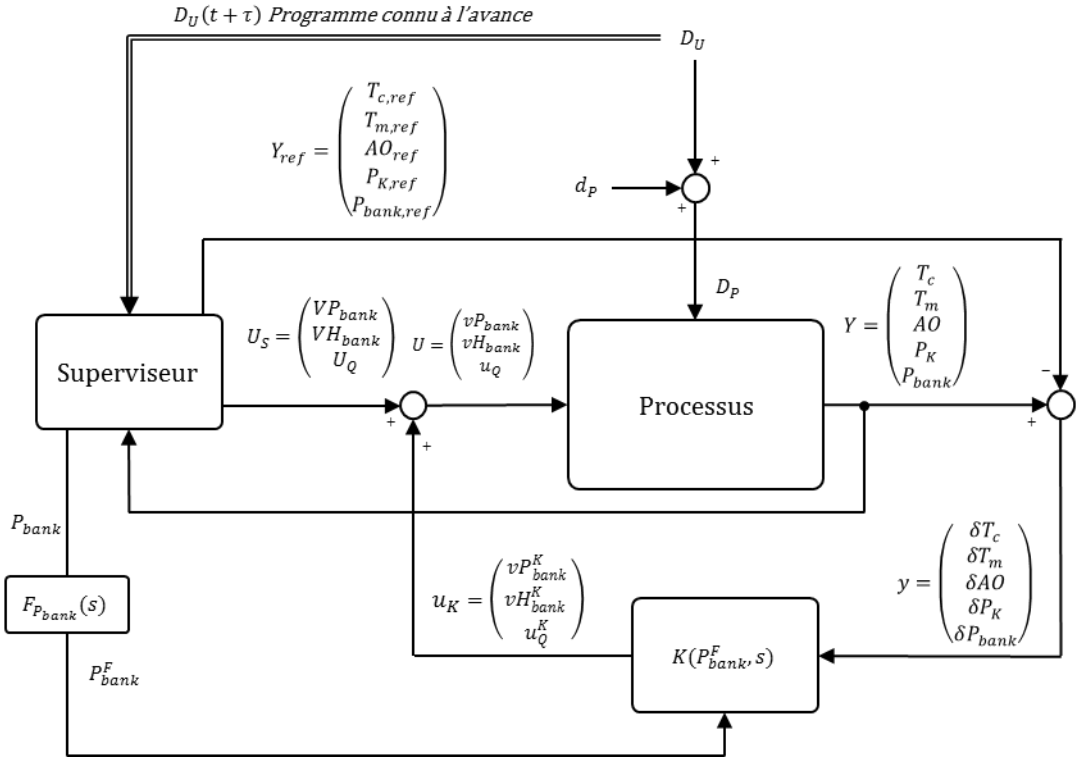


Figure 95 : Schéma global de régulation

Ce chapitre a pour but de présenter le superviseur. Nous commençons par le modèle utilisé pour la commande prédictive. Puis nous formalisons le problème de commande. Dans un troisième temps nous exposons l'algorithme développé. Enfin, nous simulons le superviseur seul et analysons ses performances.

Le schéma de régulation global est présenté sur la Figure 95.

13.1 Modèle non-linéaire utilisé pour la commande du superviseur

Le modèle que nous avons utilisé pour le superviseur est le modèle **non-linéaire point** que nous avons présenté dans la partie 11.4. Les intérêts de ce modèle sont que :

- L'effet du xénon est modélisé.
- Le retard réel du bore est modélisé.
- Plus de précision que le modèle quasi-LPV (car il n'est pas linéarisé autour des points de fonctionnement).

Les équations du modèle sont données dans 11.4 (équations (11.a) à (11.u)). On définit alors le modèle utilisé par le superviseur de la manière suivante :

$$\dot{x_S} = F_S(x_S, U_S) \tag{13.1}$$

$$y_S = G_S(x_S, U_S) \tag{13.2}$$

Avec x_S l'état, U_S le signal des commandes de référence calculées par le superviseur, y_S les sorties, F_S et G_S les fonctions définissant l'évolution de l'état et de la sortie en fonction des entrées et de l'état. En reprenant les mêmes notations qu'auparavant, on a :

$$x_S = \begin{pmatrix} T_h & T_c & AO & \rho_{dop} & n & c & T_c^{GV} & T_f^{GV} & P_{bank} & H_{bank} & C_b \end{pmatrix}^T$$

$$y_S = (T_c \quad T_m \quad AO \quad P_K \quad P_{bank})^T$$

et

$$U_S = (VP_{bank} \quad VH_{bank} \quad U_Q)^T$$

avec VP_{bank}, VH_{bank} et U_Q les commandes de référence de chaque actionneur calculées par le superviseur.

A la différence du modèle linéarisé qui utilise une approximation de Laguerre pour représenter le retard de bore, ce dernier est cette fois considéré comme suit :

$$\forall\, t,\ \overline{U_Q}(t) = U_Q(t - h_{bore})$$

Où, U_Q désigne la commande de bore et h_{bore} le retard de bore. Dans notre étude le retard est de **300 secondes**.

13.2 Formalisation du problème de commande

Dans cette partie nous présentons la formulation du problème de commande, i.e. l'expression de la *fonction objectif* et des *contraintes,* et introduisons l'ensemble des paramètres de réglage de la commande prédictive : pas d'échantillonnage choisi, l'horizon de commande et de prédiction.

13.2.1 Réglage du superviseur

Comme précisé dans la partie 3.3.3, le choix des paramètres : horizon de commande, de prédiction et le pas d'échantillonnage pour le régulateur prédictif sont primordiaux. Dans cette partie nous présentons les choix réalisés pour ces différents paramètres.

Pas d'échantillonnage

Le choix du pas d'échantillonnage doit prendre en compte plusieurs facteurs. Tout d'abord, il doit tenir compte du temps de calcul nécessaire pour résoudre le problème d'optimisation. Dans notre cas, le problème d'optimisation est résolu dans un temps relativement long[9] : de l'ordre d'une dizaine de secondes. Nous le prendrons donc supérieur à ce temps.

De plus, le superviseur a pour fonction de compenser les variations en rampe et l'évolution du xénon qui sont relativement lentes. Quant à lui, le régulateur multi-objectif prendra en charge les variations rapides de puissance, i.e. celles dues au réglage de fréquence. Ainsi, un pas d'échantillonnage court n'est pas nécessaire.

Dans notre étude nous choisirons un pas d'échantillonnage de 100 secondes, que l'on notera :

$$T_S = 100\ (secondes)$$

[9] En utilisant un MacBook Pro, processeur Core 2 Duo 2.53 GHz.

Horizon de prédiction

Comme précisé dans la partie 3.3.3, le choix de l'horizon de prédiction est à mettre en relation avec le temps de calcul, le pas d'échantillonnage, les objectifs de commande et l'horizon de commande (que nous présenterons plus loin). Il doit être suffisamment grand pour anticiper sur une plage suffisamment grande le calcul des trajectoires optimales. Dans notre cas, le phénomène le plus dimensionnant pour l'horizon de prédiction est le retard de bore (300 secondes). L'horizon de prédiction est donc pris supérieur ou égal à 300 secondes. En regard du pas d'échantillonnage relativement long retenu le temps de calcul n'est pas limitant.

Enfin, il faut considérer la longueur des transitoires (variations en rampes de 100%PN à 25%PN à une vitesse de 5%PN/min). Un transitoire de cette ampleur prend environ 900 secondes. Nous choisissons alors un temps de prédiction de 1000 secondes, ce qui correspond, pour un pas d'échantillonnage $T_S = 100$ (secondes) à un horizon de prédiction N_p de 10 pas.

$$N_p = 10$$

Horizon de commande

Considérant le pas d'échantillonnage et l'horizon de prédiction retenus, il reste à déterminer l'horizon de commande. Celui-ci doit être inférieur ou égal à l'horizon de prédiction. Dans cette étude, le temps de calcul n'étant pas un facteur limitant (en regard du pas d'échantillonnage choisi), l'horizon de commande peut être choisi arbitrairement grand, au bénéfice de la qualité de la commande (performance/robustesse). Nous avons fait le choix d'un horizon de commande N_c complet sur l'horizon de prédiction (cf. 3.3.3), soit :

$$N_c = 9$$

13.2.2 Définition des commandes considérées

Dans cette partie nous allons définir les commandes du superviseur.

On définit par U_S^j le vecteur des commandes sur le pas j (i.e. au temps $j.T_S$) calculées par l'algorithme d'optimisation comme illustré sur la Figure 96 tel que :

$$\forall j \geq 1, \qquad U_S^j = \left(VP_{bank}^j \quad VH_{bank}^j \quad U_Q^j\right)^T \tag{13.3}$$

La base de fonctions utilisée pour les commandes est la base des fonctions constantes par morceaux. Cela veut dire que les commandes seront constantes sur la période d'échantillonnage, et discontinues d'un pas d'échantillonnage au suivant (cf. Figure 96).

A chaque pas d'échantillonnage le superviseur optimise N_c valeurs, définissant les commandes sur chaque voie (Pbank, Hbank et débit de bore). Ce qui fait en totalité $3 \times N_c$ commandes (les variables de décision de l'optimisation, cf. 3.3.3). On pose $\widetilde{U_S}$ la matrice des commandes calculées au pas d'échantillonnage k tel que :

$$\widetilde{U_S} = \left[U_S^j \quad \dots \quad U_S^{j+N_c-1}\right]^T \tag{13.4}$$

Comme nous l'avons dit dans 13.2.1, l'horizon de commande choisi reprend la totalité de l'horizon de prédiction défini ci-dessus, i.e. que le superviseur calcule les commandes sur l'ensemble de cet horizon de prédiction. On déduit de (13.4) :

$$\widetilde{U_S} = \left[U_S^j \quad \dots \quad U_S^{j+N_p}\right]^T$$

A chaque pas d'échantillonnage on aura donc :

$$\widetilde{U_S} = \begin{bmatrix} VP_{bank}^j & \dots & VP_{bank}^{j+N_p} \\ VH_{bank}^j & \dots & VH_{bank}^{j+N_p} \\ U_Q^j & \dots & U_Q^{j+N_p} \end{bmatrix}^T \tag{13.5}$$

L'ensemble des coefficients de la matrice $\widetilde{U_S}$ correspondent donc aux variables de décision de l'optimisation. La Figure 96 illustre une séquence de commandes possibles élaborées au niveau du superviseur.

Enfin, à chaque pas d'échantillonnage j on applique alors la première commande calculée notée U_S^j. Alors, le signal des commandes appliqué au processus, noté U_S, évolue continument selon la relation suivante :

$$\forall t \in [j.T_S, (j+1).T_S[\,, \qquad U_S(t) = U_S^k$$

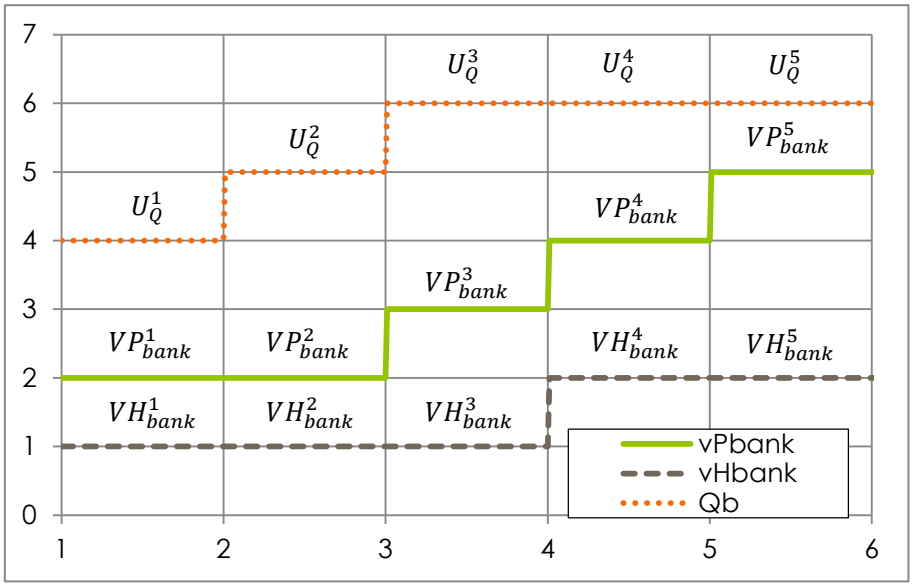


Figure 96: Commandes constantes par morceaux du superviseur

13.2.3 Définition de la fonction objectif du superviseur

La *fonction objectif* que nous considèrerons s'appuie sur les objectifs de performance définis dans la partie 10.2.2. Le superviseur aura pour but de calculer les commandes de référence pour les actionneurs Pbank, Hbank et bore minimisant l'écart des paramètres à contrôler par rapport à leurs références respectives, et ce compte tenu d'un profil de puissance turbine supposé connu par avance.

Les sortie à contrôler sont les mêmes que le régulateur multi-objectif :

- La température moyenne notée T_m.
- La distribution axiale de puissance notée AO.
- La position du groupe de barre Pbank notée P_{bank}

Les références de ces sorties sont données par $T_{m,ref}$, AO_{ref} et $P_{bank,ref}$ (cf. 10.2.1). On pose alors δT_m, δAO et δP_{bank} les écarts par rapport aux références respectives des sorties à contrôler telles que :

$$\delta T_m = T_m - T_{m,ref}$$

$$\delta AO = AO - AO_{ref}$$

$$\delta P_{bank} = P_{bank} - P_{bank,ref}$$

La fonction objectif sera construite de la manière suivant :

$$J = J_U + J_Z \tag{13.6}$$

- **Définition :** Rappelant ici que l'espace L2 des signaux à énergie finie (ou de carré sommable) sur un intervalle $[\boldsymbol{a}, \boldsymbol{b}]$ forme un espace de Hilbert dont la norme est définie par **:**

$$\|f\|_{2,[a,b]} = \sqrt{\int_a^b |f(t)|^2 dt}$$

Par la suite on notera pour simplifier l'écriture,

$$\|f\|_2 = \|f\|_{2,[0,N_P.T_S]}$$

On définit alors le critère pour le contrôle J_Z de la manière suivante :

$$J_Z = K_{T_m}\|\delta T_m\|_2 + K_{AO}\|\delta AO\|_2 + K_{P_{bank}}\|\delta P_{bank}\|_2 \tag{13.7}$$

Et le critère J_U est défini ainsi :

$$J_U = K_{P_{bank}}.\left\|VP_{bank}^f\right\|_2 + K_{H_{bank}}.\left\|VH_{bank}^f\right\|_2 + K_Q.\left\|\overline{U_Q}^f\right\|_2 \tag{13.8}$$

Avec $K_{P_{bank}}$, $K_{H_{bank}}$ et K_Q les pondérations et VP_{bank}^f, VH_{bank}^f et $\overline{U_Q}^f$ les entrées de commande filtrées. Ces dernières sont filtrées de manière à pénaliser les hautes fréquences. On définit alors H_U le filtre passe haut sur les commandes, et les signaux filtrés :

$$VP_{bank}^f = H_U(s).VP_{bank}$$

$$VH_{bank}^f = H_U(s).VH_{bank}$$

$$\overline{U_Q}^f = H_U(s).\overline{U_Q}$$

Avec :

$$H_U(s) = \frac{K_U s}{1+\tau_U s}$$

Traduisons à présent ce critère défini par (13.6), (13.7) et (13.8) dans le domaine discret, avec $\widetilde{U_S}$ la matrice des commandes définie par l'équation (13.5) :

$$J(\widetilde{U_S}) = \sum_{i=j+1}^{j+N_p} \left[\delta T_m(i.T_S)^2 + \delta AO(i.T_S)^2 + \delta P_{bank}(i.T_S)^2 + VP_{bank}^f(i.T_S)^2 + VH_{bank}^f(i.T_S)^2 + {U_Q}^f(i.T_S)^2\right]$$

Par ailleurs, nous noterons z_S^j le vecteur des écarts des sorties à contrôler, évaluées au pas d'échantillonnage j (i.e. au temps $j.T_S$), dans le domaine discret, tel que :

$$z_S^j = [\delta T_m(j.T_S) \quad \delta AO(j.T_S) \quad \delta P_{bank}(j.T_S)]^T \tag{13.9}$$

13.2.4 Définition des contraintes du superviseur

Les contraintes que nous imposerons au superviseur, par l'intermédiaire du problème de commande, sont issues des exigences formulées dans le cahier des charges concernant les sollicitations sur la commande (cf. 10.2.3). Rappelons que les actionneurs sont limités en vitesse et en position. L'intérêt du superviseur, par rapport au régulateur multi-objectif est qu'il a la capacité de prendre explicitement en compte des contraintes temporelles formelles, qu'elles portent sur les entrées ou sur les variables d'état : ici les contraintes de position et de vitesse maximales des groupes de grappes ainsi que les débits maximaux de dilution/borication.

On formule alors les contraintes suivantes, déduites du cahier des charges (cf. 10.2) :

$$\begin{cases} VP_{bank} < 1.25\ (pas/s) \\ VP_{bank} > -1.25\ (pas/s) \\ VH_{bank} < 1.25\ (pas/s) \\ VH_{bank} > -1.25\ (pas/s) \\ U_Q < 2.7.10^{-2}\ (ppm/s) \\ U_Q > -4.7.10^{-3}\ (ppm/s) \\ P_{bank} < 1053\ (pas) \\ P_{bank} > 0\ (pas) \\ H_{bank} < 410\ (pas) \\ H_{bank} > 9\ (pas) \\ C_b > 0\ (ppm) \end{cases} \tag{13.10}$$

Notons que les limites de débit de bore (u_Q) sont déterminées pour une concentration en bore de 1500 ppm (cf. 11.2.4.2 pour la méthode de calcul).

Les contraintes définies dans (13.10) sont reformulées, de manière compacte comme suit (cf. notations des équations (13.3) et (13.9)) :

$$\forall i \in \{j, N_p + j\} \begin{cases} U_S^i < (1.25 \quad 1.25 \quad 2.7.10^{-2}) \\ U_S^i > (-1.25 \quad -1.25 \quad -4.7.10^{-3}) \\ z_S^i < (1053 \quad 410 \quad +\infty) \\ z_S^i > (0 \quad 9 \quad 0) \end{cases}$$

13.3 Algorithme développé

Cette partie présente l'algorithme de commande prédictive développé dans la thèse.

13.3.1 Solveur utilisé

Comme précisé dans 13.1, le modèle utilisé pour calculer les sorties du modèle en fonction des commandes (les variables de décision), est un modèle non-linéaire. Compte tenu des contraintes et de la fonction objectif définis précédemment, un algorithme d'optimisation non-linéaire sous-contraintes est requis pour la résolution d'un tel problème. Notons que la recherche d'un code performant pour l'optimisation n'étant pas un enjeu de la thèse nous avons choisi d'utiliser un algorithme d'optimisation non-linéaire sous-contraintes, basé sur la méthode du point intérieur, similaire à ceux présentés par [Powell, 1978], [Byrd *et al.*, 1999] et [Coleman et Li, 1996], et tel qu'implémenté par la fonction *fmincon*[10] de Matlab. Cette dernière permet la résolution du problème générique :

$$\min_x f(x), \text{ tel que} \begin{cases} c(x) \leq 0 \\ c_{eq}(x) = 0 \\ Fx \leq b \\ F_{eq}x = b_{eq} \\ lb \leq x \leq ub \end{cases}$$

où f est la fonction de coût à minimiser, c et c_{eq} deux fonctions retournant des vecteur, F, b, F_{eq}, b_{eq} des matrices et lb et ub deux vecteurs.

[10] https://fr.mathworks.com/help/optim/ug/fmincon.html dernière visite le 11/11/2017

Nous nous servirons des contraintes formalisées par c, c_{eq}, F, b, F_{eq}, b_{eq} lb et ub pour les contraintes que nous considérons dans le problème d'optimisation posé dans 13.2.4.

13.3.2 Principe de fonctionnement de l'algorithme

Schéma de principe de l'algorithme

L'algorithme du superviseur que nous avons développé dans le cadre de la thèse est présenté sous la forme d'un schéma de principe sur la Figure 97. Cette figure rend compte des différentes fonctions qui sont mises en œuvre ainsi que les flux d'information qu'elles échangent à chaque pas d'échantillonnage.

A chaque pas d'échantillonnage, le superviseur reçoit les mesures du système Y, les références des sorties à contrôler Z_{ref} ainsi que le profil de puissance turbine utilisateur D_U. Pendant ce pas d'échantillonnage :

- Il met à jour la valeur du vecteur d'état initial du système au pas k noté X_0^k (en supposant ici que l'état complet est reconstruit),
- à partir des données d'entrée, il calcule la séquence $\widehat{U_S}$ des commandes optimales par la fonction $fmincon$,
- il applique de t_j à $t_j + T_S$ le premier élément de la séquence de commande calculée U_S^j sur la période d'échantillonnage au système réel,
- il mémorise le reste de la séquence des commandes $U_0^{j+1} = \left(U_S^{j+1} \quad \dots \quad U_S^{j+N_p}\right)^T$ pour initialiser l'optimisation au prochain pas d'échantillonnage,
- et enfin il mémorise la commande de bore appliquée (au pas j) et met à jour la commande de bore retardée $\widehat{u_Q}$.

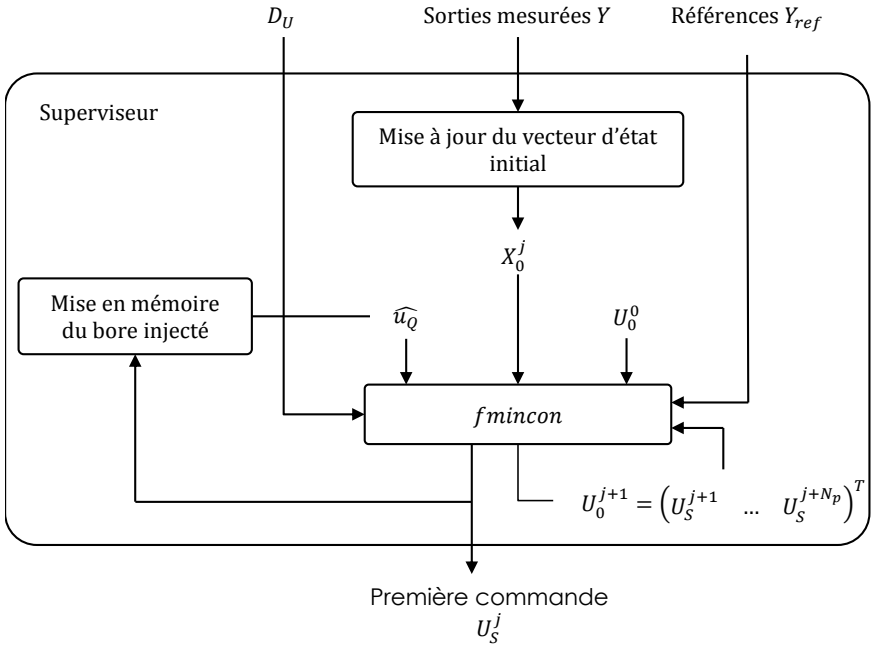


Figure 97: Schéma de principe de l'algorithme du superviseur

13.4 Simulation et Analyse du superviseur

Nous présentons ici les résultats de la simulation du superviseur. Pour la simulation, nous avons utilisé le modèle non-linéaire point du cœur développé dans la thèse et présenté dans 11.4.

13.4.1 Suivi de rampe à 5%PN/min superviseur seul

Le scénario considéré pour la simulation est une rampe de puissance de 5%PN/min entre 100%PN et 60%PN (cf. Figure 134) car c'est le scénario de référence pour le test du *Core Control*. Les courbes de simulation sont présentées en Annexe J.

La simulation montre que le comportement du superviseur est conforme aux attentes :

- La température moyenne est maintenue dans le domaine autorisé défini par les LCO.
- De même pour l'AO.
- De même pour la position de Pbank.
- Le xénon est bien compensé par la concentration en bore.

Le Tableau 11 résume les écarts maximaux des sorties à contrôler par rapport à leurs références en valeur absolue.

Critères		Observations	Ecart max par rapport à la référence
LCO	T_m	Contrôle correct de la température moyenne	$\lvert\Delta T_m^{max}\rvert = 0.82\ °C \leq 1.5\ °C$
	AO	Contrôle correct de l'AO	$\lvert\Delta AO^{max}\rvert = 4.62\ \%AO \leq 5\ \%AO$
	P_{max}	Contrôle correct de la position de Pbank	$\lvert\Delta P_{bank}^{max}\rvert = 22.97\ PE \leq 30\ PE$

Tableau 11: Résultats superviseur seul - Rampe de puissance 100%PN à 60%PN à 5%PN/min

13.4.2 Rampe et réglage de fréquence superviseur seul

Dans cette partie, nous testons le superviseur dans le cas du réglage de fréquence comme montré sur la Figure 98.

On observe sur la Figure 99, comme nous l'avions annoncé dans 10.3.3.4, que le superviseur seul ne peut pas rejeter les variations de puissance dues au réglage de fréquence. La température dépasse les bornes hautes et basses définies par les LCO de température (cf. 10.2.1).

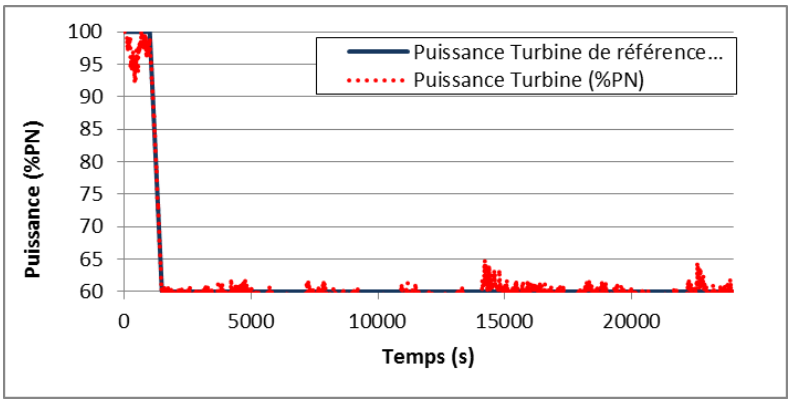


Figure 98: Tracé de la puissance turbine – Rampe de puissance 100%PN à 60%PN et réglage de fréquence

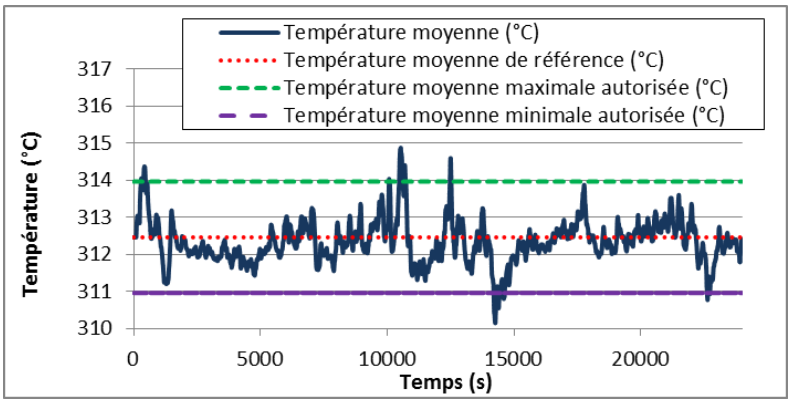


Figure 99: Tracé de la température moyenne - Rampe de puissance 100%PN à 60%PN et réglage de fréquence

13.5 Conclusions sur la partie superviseur

Nous avons proposé, au niveau du superviseur, un algorithme de commande prédictive permettant d'anticiper les variations de charge (e.g. rampes de puissance) prévues en avance. L'algorithme montre des performances correctes vis-à-vis des objectifs de commande sur les scénarios prévus. Cependant, et comme nous l'avions anticipé, le superviseur n'a pas la capacité d'assurer le contrôle des sorties lors des variations de fréquence. La Figure 99 montre le dépassement de la température moyenne du domaine de fonctionnement autorisé.

14 Simulation et analyse du RMOSC

Sommaire

14.1 Rampe de puissance du RMOSC 257
14.2 Step de puissance du RMOSC 261
14.3 Rampe et réglage de fréquence du RMOSC 264
14.4 Conclusions de la partie simulation du RMOSC 268
14.5 Comparaison du RMOSC avec le mode T 269
14.5.1 Scénarios de simulation du mode T 269
14.5.2 Comparaison des résultats de simulation du mode T et du RMOSC 269
14.5.3 Conclusions sur la simulation 271

Dans ce chapitre nous rapportons les résultats de simulation du régulateur complet, constitué de la commande hiérarchisée du **superviseur et du régulateur multi-objectif à gains séquencés** que l'on nommera **RMOSC** pour Régulateur Multi-Objectif Supervisé du Cœur.

Le RMOSC est simulé sur le modèle non-linéaire du réacteur présenté en 11.5. De manière à respecter les exigences de sollicitation des actionneurs des saturations seront introduites sur ces derniers en cohérence avec les sollicitations maximales définies dans 10.2.3.

Nous commençons par tester ce régulateur sur un transitoire de charge à 5%PN/min de 100%PN à 60%PN suivi d'une remontée de charge de 60%PN à 100%PN à la même vitesse. Dans un deuxième temps nous observons la réponse à un step de puissance de 10%PN. Enfin, nous étudions le cas du réglage de fréquence superposé à un transitoire de charge.

14.1 Rampe de puissance du RMOSC

Les courbes de simulation du RMOSC pour une rampe de puissance 100-60-100%PN à 5%PN/min sont présentées sur les figures ci-dessous (cf. Figure 101 à Figure 107). L'organisation des courbes se fait comme suit :

- Figure 101, la variation de puissance.
- Figure 102, Figure 103, Figure 104 et Figure 105, les sorties à réguler ainsi que leur domaine de fonctionnement autorisé.
- Figure 106, la vitesse des actionneurs.
- Figure 107, les concentrations de xénon et de bore.

Sur ce scénario en rampe, le RMOSC fournit des résultats corrects vis-à-vis des critères de commande. Ces résultats sont résumés dans le Tableau 12. On peut voir que toutes les sorties à réguler sont maintenues dans le domaine autorisé, défini par les LCO, i.e. qu'aucune des sorties ne dépasse les limites maximales et minimales qui lui sont associées.

On note, premièrement, un écart de température moyenne, d'AO et de position de Pbank de respectivement 0.19°C, 2.9%AO et 6.7PE, soit respectivement 12%, 58% et 22% des écarts maximums autorisés pour ces variables. On note que l'écart (en %) de la température moyenne est plus faible que pour les autres variables en particulier grâce à la pondération fréquentielle W_{z_1} définie en 12.2.5, en raison de la priorisation de la température moyenne face aux autres sorties. Ensuite, les simulations montrent une erreur statique nulle.

De plus, le xénon est complètement compensé et même anticipé par le régulateur par des variations de bore (cf. Figure 107). Notons, de même, que les vitesses des actionneurs sont saturées lors de la simulation. Elles ne peuvent donc pas excéder les limites physiques des actionneurs réels.

Comparons à présent les résultats du RMOSC à ceux du superviseur seul (cf. 13.4.1). On note que tous les écarts des sorties à réguler du superviseur seul, sur le même scénario, sont plus élevés que dans le cas du RMOSC (cf. Tableau 11 et Tableau 12) ; notamment car le RMOSC bénéficie du contrôle rapproché, contrairement au superviseur seul. On perçoit donc ici l'intérêt de l'architecture hiérarchisée proposée.

Compte tenu des éléments précédents, on peut donc dire que le RMOSC présente de bonnes performances sur ce scénario.

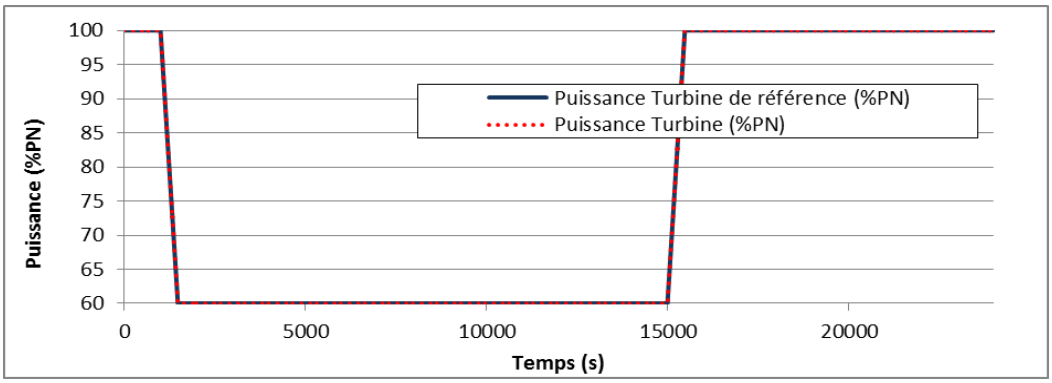


Figure 100: Tracé de la puissance turbine – Rampe de puissance 100%PN à 60%PN

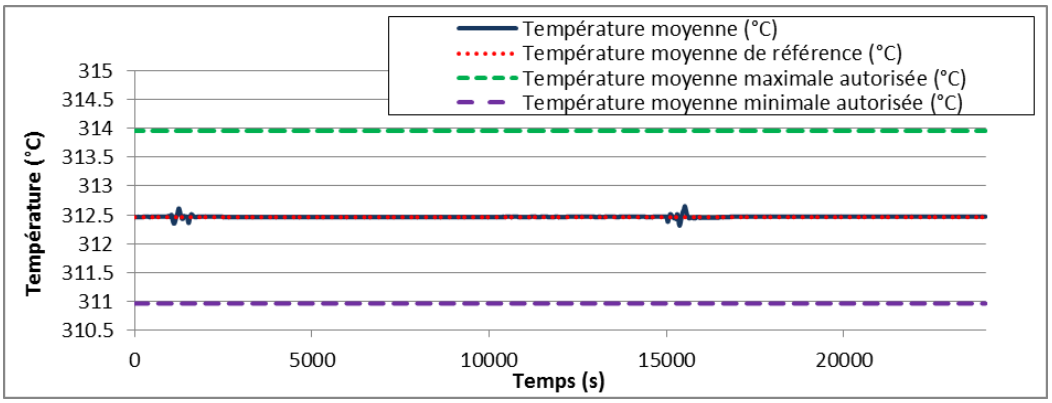


Figure 101: Tracé de la température moyenne - Rampe de puissance 100%PN à 60%PN

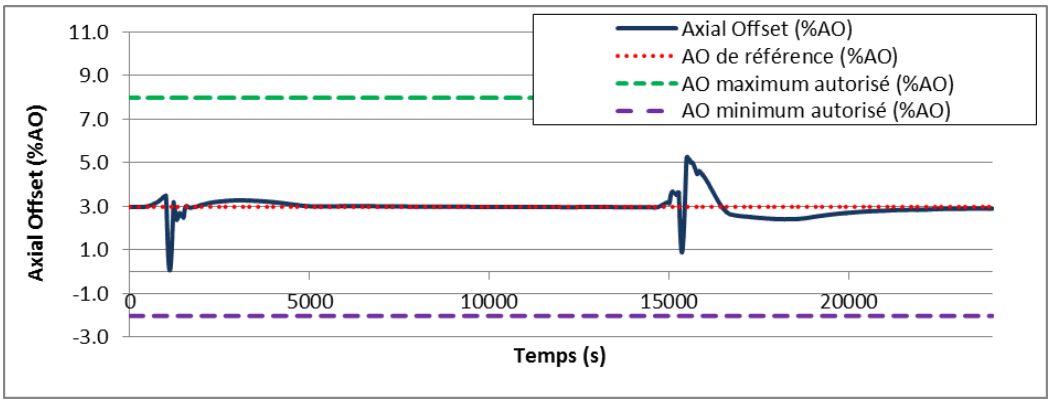


Figure 102: Tracé de l'AO - Rampe de puissance 100%PN à 60%PN

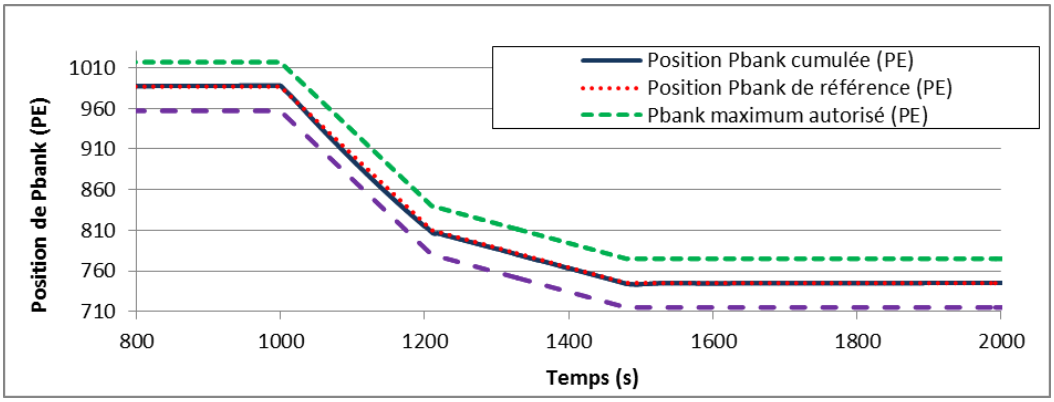


Figure 103: Tracé de la position de Pbank - Rampe de puissance 100%PN à 60%PN – Descente de charge

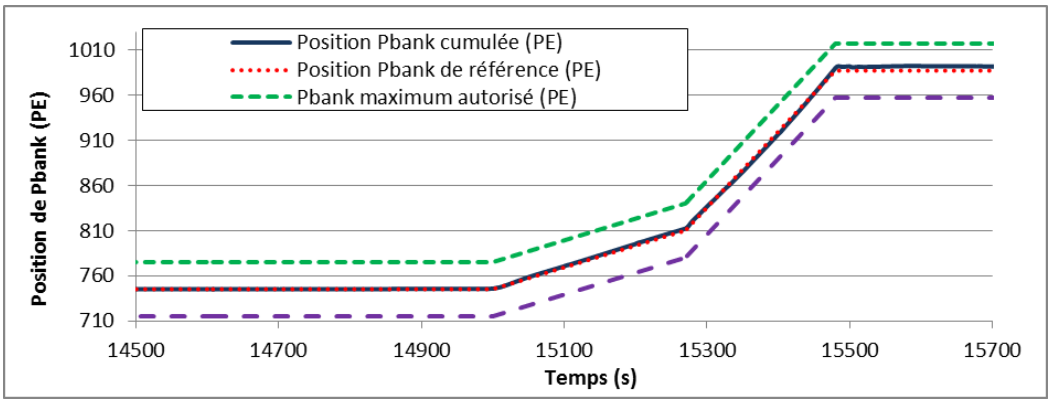


Figure 104: Tracé de la position de Pbank - Rampe de puissance 100%PN à 60%PN – Remontée en charge

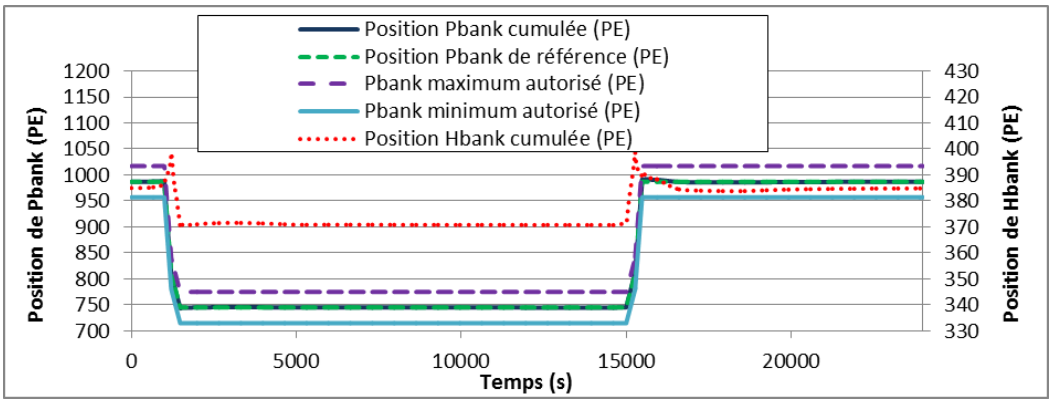


Figure 105: Tracé de la position des groupes - Rampe de puissance 100%PN à 60%PN

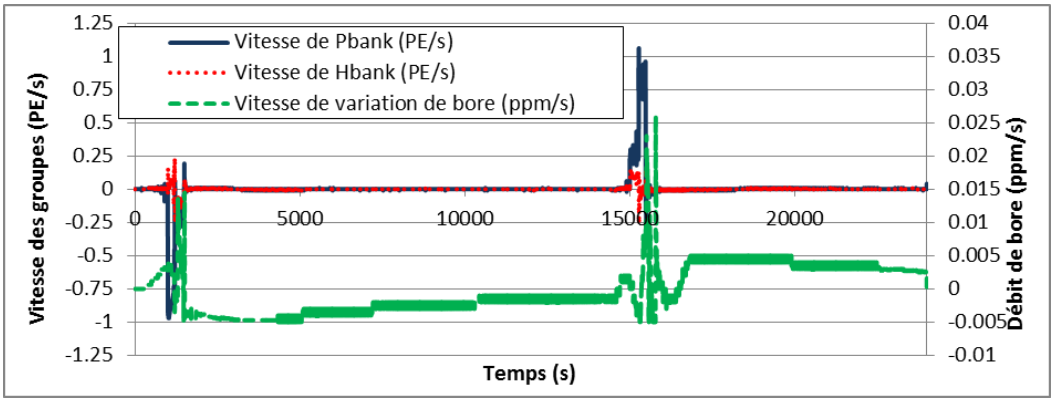


Figure 106: Tracé des vitesses et débit des actionneurs - Rampe de puissance 100%PN à 60%PN

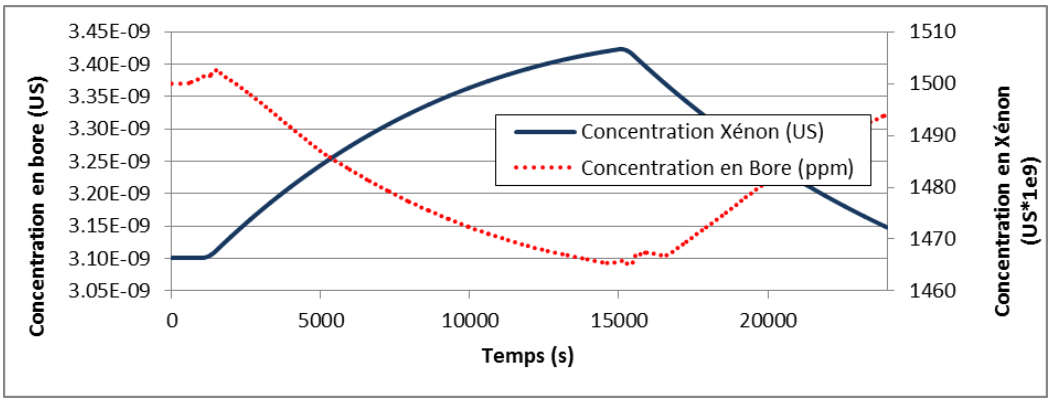


Figure 107: Xénon vs. Concentration en bore - Rampe de puissance 100%PN à 60%PN

Critères		Observations	Ecart max par rapport à la référence
LCO	T_m	Contrôle correct de la température moyenne	$\lvert\Delta T_m^{max}\rvert = 0.19\ °C \leq 1.5\ °C$
	AO	Contrôle correct de l'AO	$\lvert\Delta AO^{max}\rvert = 2.9\ \%AO \leq 5\ \%AO$
	P_{max}	Contrôle correct de la position de Pbank	$\lvert\Delta P_{bank}^{max}\rvert = 6.7\ PE \leq 30\ PE$

Tableau 12: Résultats RMOSC - Rampe de puissance 100-60-100%PN à 5%PN/min

14.2 Step de puissance du RMOSC

Les courbes de simulation du RMOSC pour un step de puissance 100-90%PN sont présentées sur les figures ci-dessous (cf. Figure 108 à Figure 114). L'organisation des courbes est la même que pour la section précédente 14.1, à savoir :

- Figure 108, la variation de puissance.
- Figure 109, Figure 110, Figure 111 et Figure 112 les sorties à réguler ainsi que leur domaine de fonctionnement autorisé.
- Figure 113, la vitesse des actionneurs.
- Figure 114, les concentrations de xénon et de bore.

Les conclusions de ces résultats de simulation (résumés dans le Tableau 13) sont similaires à celles de la section précédente (cf. 14.1), à savoir :

- Les sorties à réguler sont toutes maintenues dans le domaine de fonctionnement autorisé considéré.
- L'erreur statique est nulle.
- Les vitesses et débit des actionneurs sont saturés et donc respectent les exigences associées.

- Le bore compense bien la variation de concentration de xénon.

Un comportement appréciable du RMOSC est l'insertion de Hbank lors de la variation de charge pour aider au contrôle de la température moyenne, suivi d'une extraction pour assurer le contrôle de l'AO.

Cependant, on remarque que, sur ce scénario, l'AO est plus proche des limites autorisées que dans le cas de la rampe de charge (cf. 14.1). Ceci s'explique car le scénario étudié ici est considéré comme le plus dimensionnant dans le problème considéré.

Finalement, compte tenu des éléments précédents, le RMOSC fournit des bons résultats, conformes aux spécifications.

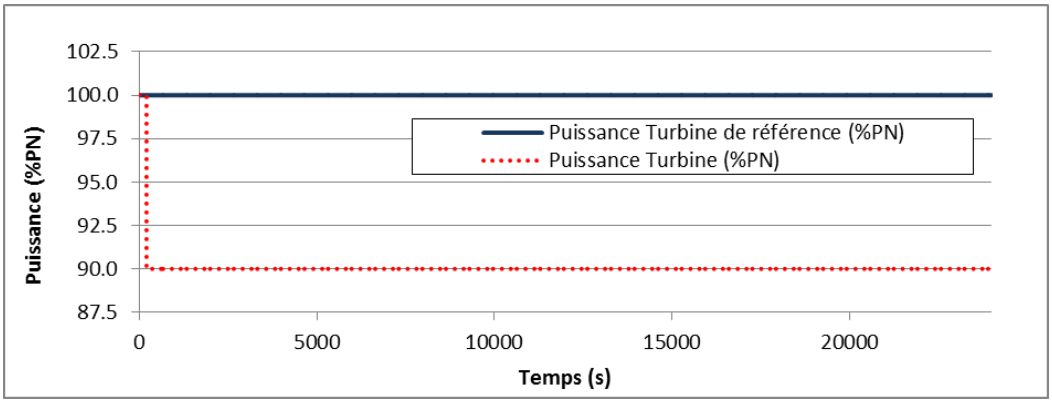


Figure 108: Tracé de la puissance turbine – Step de puissance 100%PN à 90%PN

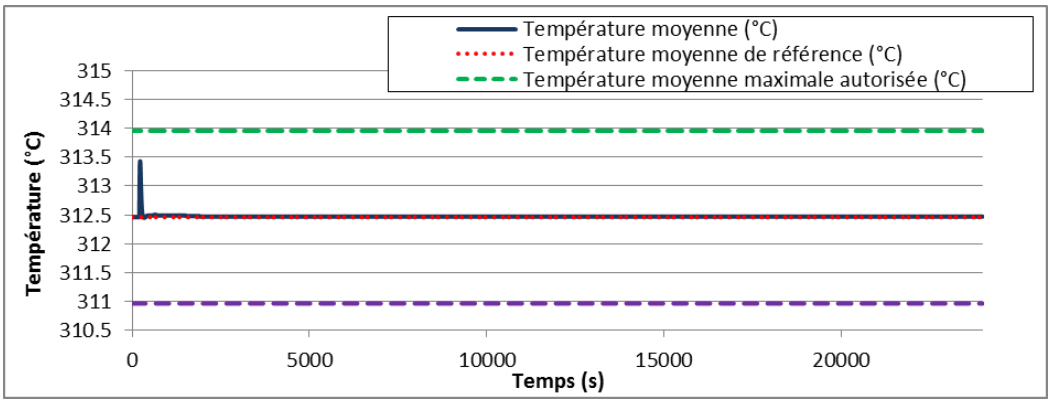


Figure 109: Tracé de la température moyenne - Step de puissance 100%PN à 90%PN

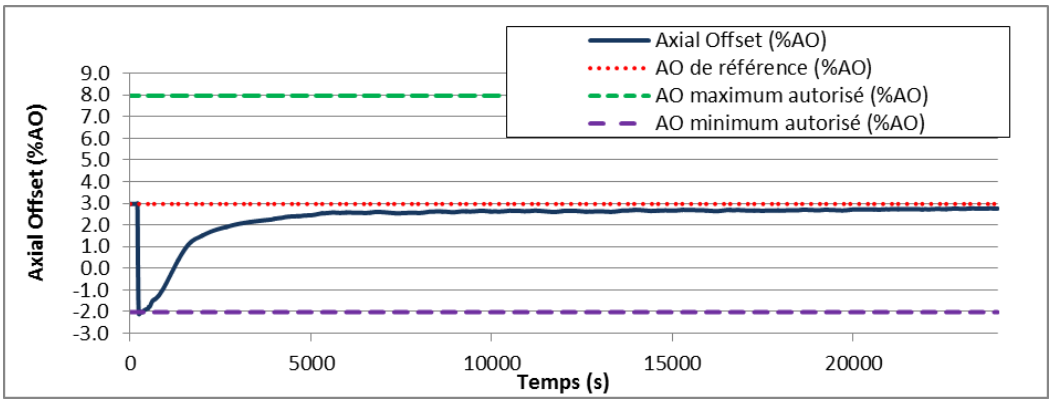


Figure 110: Tracé de l'AO - Step de puissance 100%PN à 90%PN

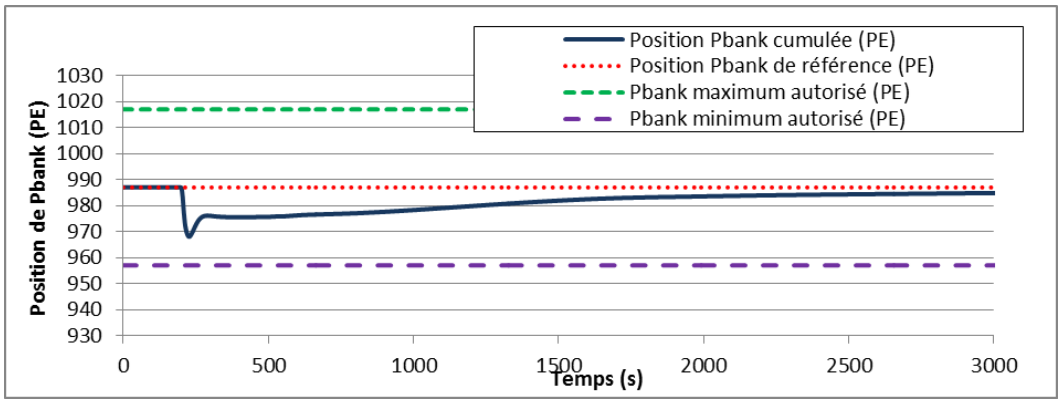


Figure 111: Tracé de la position de Pbank - Step de puissance 100%PN à 90%PN

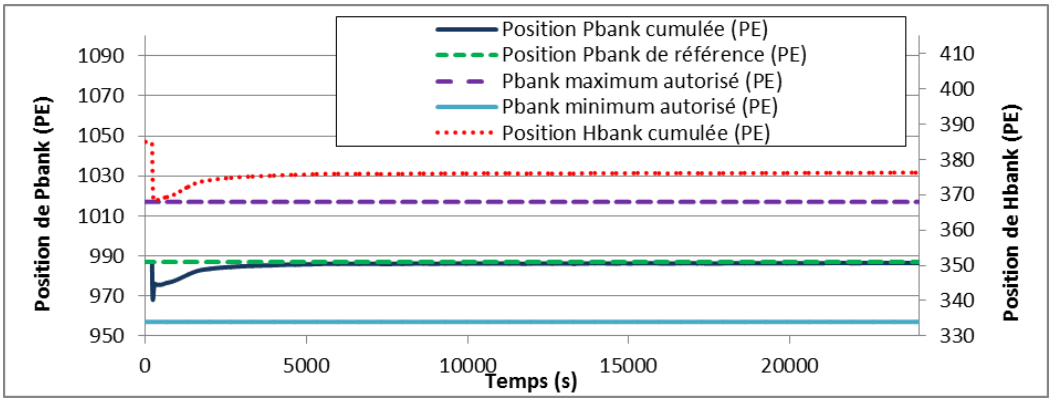


Figure 112: Tracé de la position des groupes - Step de puissance 100%PN à 90%PN

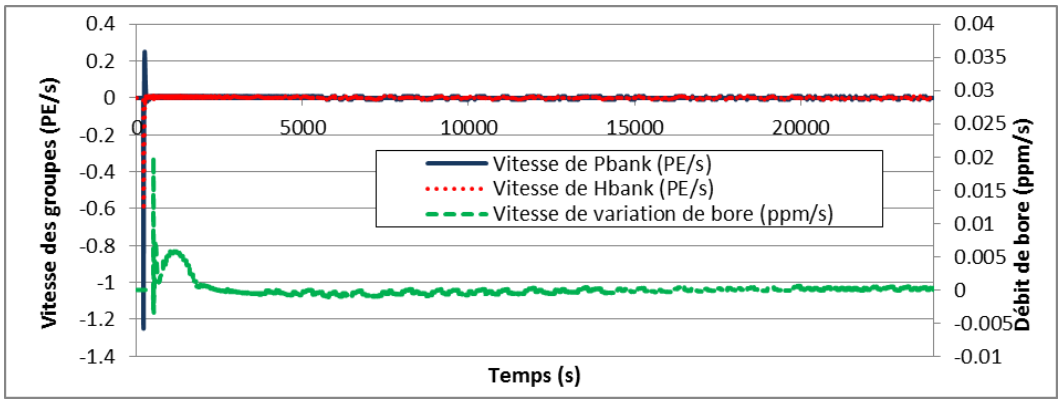


Figure 113: Tracé des vitesses et débit des actionneurs - Step de puissance 100%PN à 90%PN

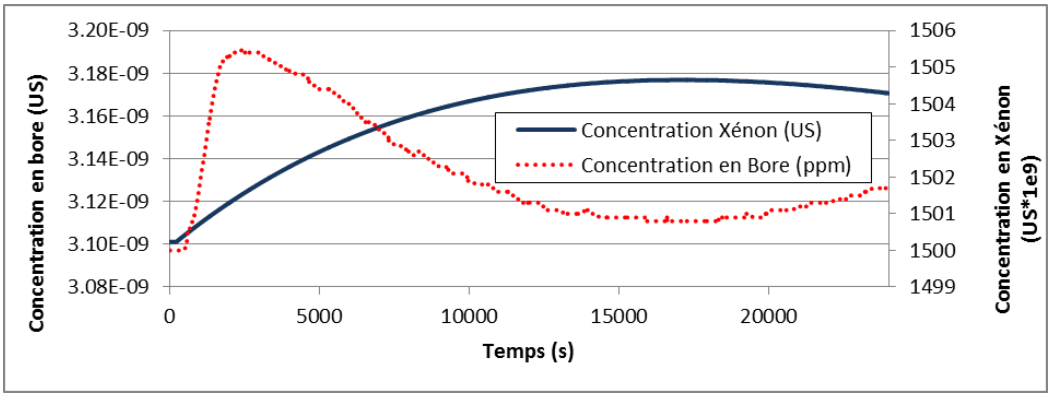


Figure 114: Xénon vs. Concentration en bore - Step de puissance 100%PN à 90%PN

Critères		Observations	Ecart max par rapport à la référence
LCO	T_m	Contrôle correct de la température moyenne	$\lvert\Delta T_m^{max}\rvert = 1.0\ °C \leq 1.5\ °C$
	AO	Contrôle correct de l'AO, léger dépassement	$\lvert\Delta AO^{max}\rvert = 5.09\ \%AO \sim 5\ \%AO$
	P_{max}	Contrôle correct de la position de Pbank	$\lvert\Delta P_{bank}^{max}\rvert = 19\ PE \leq 30\ PE$

Tableau 13: Résultats RMOSC – Step de puissance de 100%PN à 90%PN

14.3 Rampe et réglage de fréquence du RMOSC

Les courbes de simulation du RMOSC pour une rampe 100-70%PN à 5%PN/min avec réglage de fréquence sont présentées sur les figures ci-dessous (cf. Figure 115 à Figure 121). L'organisation des courbes est la même que pour la section 14.1, à savoir :

- Figure 115, la variation de puissance.

- Figure 116, Figure 117, Figure 118 et Figure 119 les sorties à réguler ainsi que leur domaine de fonctionnement autorisé.
- Figure 120, la vitesse des actionneurs.
- Figure 121, les concentrations de xénon et de bore.

Les conclusions de ces résultats de simulation (résumés dans le Tableau 14) sont similaires à celles des sections précédentes (cf. 14.1 et 14.2), à savoir :

- Les sorties à réguler sont toutes maintenues dans le domaine de fonctionnement autorisé considéré.
- L'erreur statique est nulle.
- Les vitesses et débit des actionneurs sont saturés et donc respectent les exigences associées.
- Le bore compense bien la variation de concentration de xénon.

Comparons à présent les résultats du RMOSC avec ceux du régulateur multi-objectif seul. On constate tout d'abord que l'écart par rapport à la référence de la température moyenne et de la position de Pbank sont similaires dans les deux cas (cf. Tableau 10 et Tableau 14). A noter un léger avantage pour le RMOSC. En revanche, l'écart d'AO à sa référence est nettement plus élevé dans le cas du multi-objectif seul. Ceci s'explique par le fait que le régulateur multi-objectif ne bénéficie pas de capacité de prédiction de la concentration xénon, contrairement au RMOSC. Ce dernier possède un algorithme prédictif (le superviseur), doté d'un modèle xénon. Ainsi, le RMOSC anticipe les variations de xénon en injectant de bore en quantité juste et assure un meilleur contrôle. On voit alors l'intérêt de la structure hiérarchisée, en particulier du superviseur.

Comparons à présent les résultats du RMOSC avec ceux du superviseur seul. Comme précisé dans 13.4.2, le superviseur seul n'a pas la capacité de maintenir les sorties à réguler dans leur domaine autorisé dans le cas de variations imprévues (dues au réglage de fréquence). Or, on constate sur les courbes de Figure 116, Figure 117, Figure 118 et Figure 119, que les sorties à réguler sont bien maintenues dans le domaine autorisé dans le cas du RMOSC. En effet, l'utilisation de la structure hiérarchisée proposée (en particulier le régulateur multi-objectif) permet d'assurer le contrôle rapproché.

Cependant, remarquons, dans le cas du réglage de fréquence, que les sollicitations des actionneurs sont élevées. En particulier sur la Figure 119 on peut voir que le groupe Hbank effectue un groupe nombre de mouvement. Dans le cas du Mode T [Grossetête, 2007] l'utilisation de bandes mortes permet de limiter la sollicitation des actionneurs.

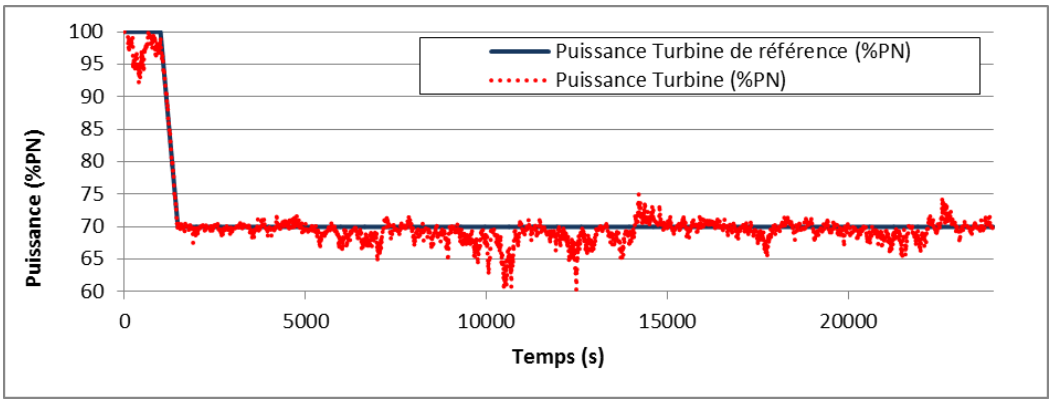


Figure 115: Tracé de la puissance turbine - Rampe 100%PN à 70%PN & Réglage de fréquence

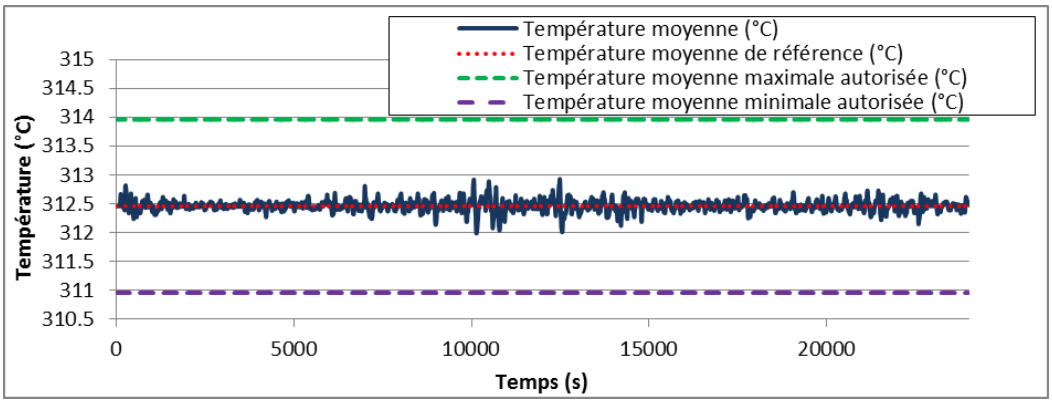


Figure 116: Tracé de la température moyenne - Rampe 100%PN à 70%PN & Réglage de fréquence

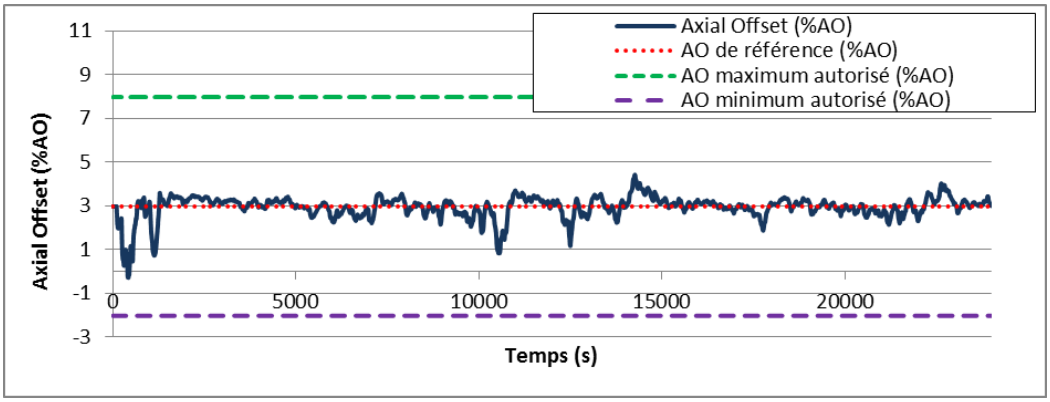


Figure 117: Tracé de l'AO - Rampe 100%PN à 70%PN & Réglage de fréquence

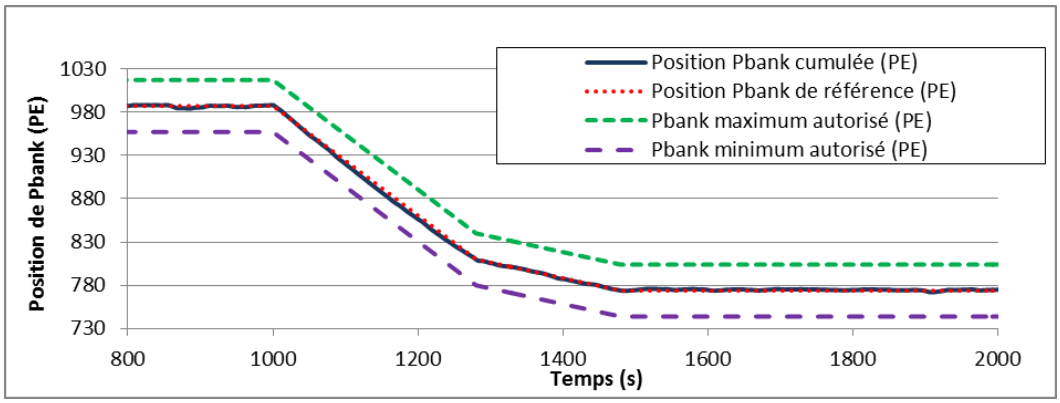


Figure 118: Tracé de la position de Pbank - Rampe 100%PN à 70%PN & Réglage de fréquence

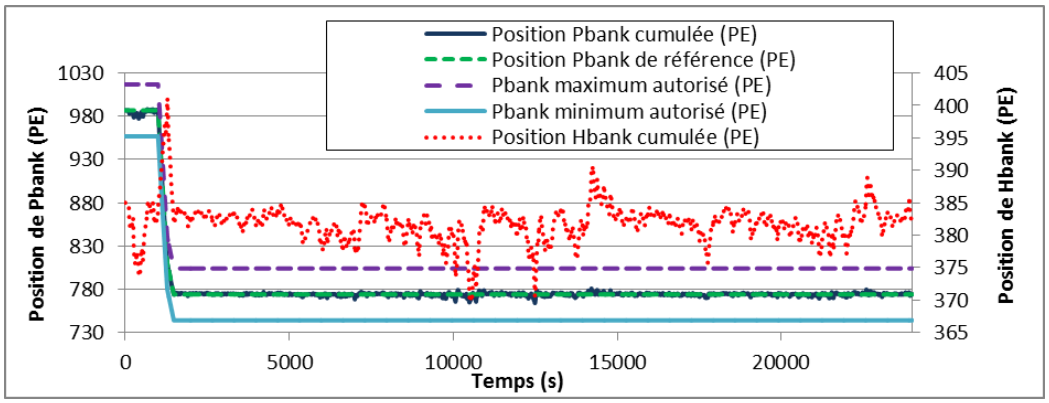


Figure 119: Tracé de la position des groupes - Rampe 100%PN à 70%PN & Réglage de fréquence

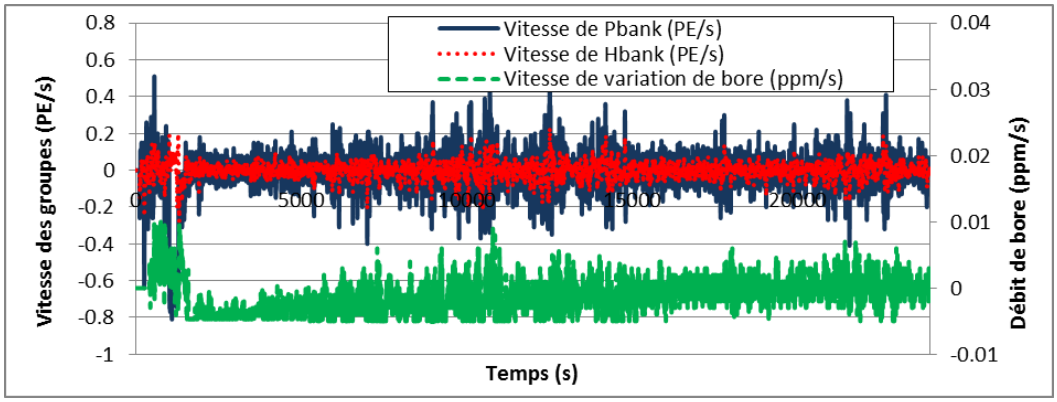


Figure 120: Tracé des vitesses et débit des actionneurs - Rampe 100%PN à 70%PN & Réglage de fréquence

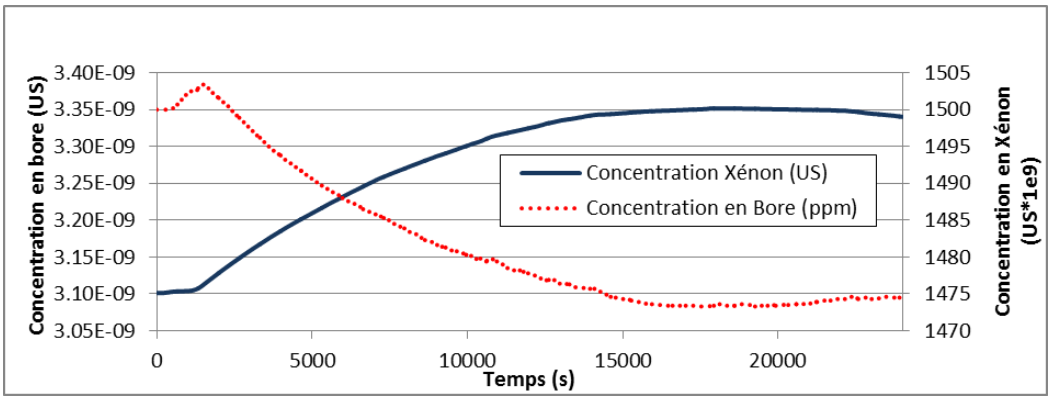


Figure 121: Xénon vs. Concentration en bore - Rampe 100%PN à 70%PN & Réglage de fréquence

Critères		Observations	Ecart max par rapport à la référence
LCO	T_m	Contrôle correct de la température moyenne	$\lvert\Delta T_m^{max}\rvert = 0.47\ °C \leq 1.5\ °C$
	AO	Contrôle correct de l'AO	$\lvert\Delta AO^{max}\rvert = 3.3\ \%AO \leq 5\ \%AO$
	P_{max}	Contrôle correct de la position de Pbank	$\lvert\Delta P_{bank}^{max}\rvert = 10.1\ PE \leq 30\ PE$

Tableau 14: Résultats RMOSC - Rampe et réglage de fréquence

14.4 Conclusions de la partie simulation du RMOSC

Dans cette partie nous avons simulé le RMOSC constitué de la commande hiérarchisée : superviseur + multi-objectif. Les résultats de simulation montrent que les objectifs de performance et les contraintes sont respectés, à savoir en particulier que les sorties à réguler sont maintenues dans le domaine de fonctionnement autorisé sur tous les scénarios de simulation.

Comparé au superviseur seul, le RMOSC permet le contrôle rapproché du processus en particulier pour les scénarios : réglage de fréquence ou step de puissance. Comparé au régulateur multi-objectif seul, les trajectoires empruntées par le RMOSC permettent d'anticiper (grâce au superviseur) les variations xénon et de garder les sorties à contrôler au plus proche de leurs références. Ainsi, nous avons montré l'intérêt de la structure hiérarchisée par rapport aux deux architectures seules.

14.5 Comparaison du RMOSC avec le mode T

Dans cette partie nous présentons les différences avec le mode T présenté dans 3.2.4.

14.5.1 Scénarios de simulation du mode T

Dans cette partie nous présentons les résultats de simulation du mode T. Deux scénarios sont considérés :

- Une rampe à 5%PN/min entre 100%PN et 60%PN (suivi d'une remontée en puissance de 60%PN à 100%PN). Les résultats de cette simulation sont reportés en Annexe Q.
- Réglage de fréquence primaire et secondaire autour de 90%PN. Les résultats de cette simulation sont reportés en Annexe R.

Remarque

Les domaines de fonctionnement autorisés des sorties à contrôler, montrés sur les figures de ces annexes sont ceux considérés dans la thèse et non pas les domaines considérés pour le mode T. Il peut par conséquent dépasser les limites fixées par ce domaine.

14.5.2 Comparaison des résultats de simulation du mode T et du RMOSC

Tout d'abord, il est important de noter que notre intention ici n'est pas de comparer de manière exacte le mode T au régulateur développé, pour les raisons suivantes :

- Les objectifs de commande et contraintes sont plus exigeants dans notre étude que pour le mode T.
- Contrairement au RMOSC, le mode T possède des bandes mortes sur les sorties à contrôler.
- Le mode T est simulé sur simulateur réel tandis que le RMOSC est simulé sur modèle simplifié.
- Les scénarios de simulation ne sont pas exactement les mêmes.

Nous allons donc commenter et comparer de manière qualitative le mode T et le régulateur développé dans la thèse, en nous appuyant sur leurs courbes de simulation.

Nous comparerons les performances globales, ensuite la fatigue des actionneurs.

Comparaison des performances

Nous entendons par performances la capacité du régulateur à minimiser l'impact de la puissance turbine sur les sorties à contrôler : la température moyenne, l'AO et la position du groupe Pbank.

En ce sens, les performances du RMOSC dans la thèse sont meilleures que celles obtenues sur le mode T, sur les deux scénarios. Comparons tout d'abords dans le cas de la rampe de puissance. La Figure 101 et la Figure 151 montrent que l'écart de la température moyenne à sa référence est plus faible dans le cas du RMOSC. De même pour l'AO en comparant les Figure 102 et Figure 152 et de même pour la position de Pbank en comparant la Figure 103 et la Figure 153. Les valeurs des écarts maximums des sorties à contrôler sur ce scénario sont reportés sur le Tableau 15.

Ecarts considérés		**Ecart max par rapport à la référence du mode T**	**Ecart max par rapport à la référence du RMOSC**
LCO	$\lvert\Delta T_m^{max}\rvert$	1.5 °C	0.19 °C
	$\lvert\Delta AO^{max}\rvert$	5.8%AO	2.9 %AO
	$\lvert\Delta P_{bank}^{max}\rvert$	70 PE	6.7 PE

Tableau 15: Comparaison RMOSC / Mode T - Rampe de puissance 100-60-100%PN à 5%PN/min

En ce qui concerne le scénario de réglage de fréquence on constate de même de meilleures performances pour le RMOSC. En effet en comparant la Figure 116 à la Figure 156, la Figure 117 à la Figure 157 et la Figure 118 à la Figure 158 on peut voir que l'écart par rapport à la référence des sorties est globalement plus faible pour le RMOSC. Les écarts maximums des sorties par rapport à leurs références sont relevés en simulation montrés sur le Tableau 16. Notons que contrairement aux chiffres donnés dans le Tableau 14, nous avons occulté la partie en rampe de la simulation de manière à observer uniquement l'effet du réglage de fréquence.

Ecarts considérés		**Ecart max par rapport à la référence du mode T**	**Ecart max par rapport à la référence du RMOSC**
LCO	$\lvert\Delta T_m^{max}\rvert$	1 °C	0.47 °C
	$\lvert\Delta AO^{max}\rvert$	2.5%AO	2.1 %AO

	$\lvert\Delta P_{bank}^{max}\rvert$	26 PE	10 PE

Tableau 16: Comparaison RMOSC / Mode T - Rampe de puissance 100-60-100%PN à 5%PN/min

Anticipation du xénon

Observons désormais le comportement du groupe de grappes Pbank dans le cas du mode T après une rampe de puissance sur Figure 153. On observe une extraction du groupe Pbank. Elle correspond à l'augmentation de la concentration de xénon, due au changement de puissance, présentée sur la Figure 154. La Figure 105 montre le même scénario dans le cas du RMOSC. On constate que la position des groupes de grappe n'est pas affectée par la variation de xénon (cf. Figure 107), grâce notamment à la participation du superviseur qui anticipe la variation du xénon en injectant du bore ou de l'eau en quantité juste nécessaire pour compenser l'effet.

Fatigue des actionneurs

Comme dit dans 3.2.4, le mode T est doté de bandes mortes sur les sorties à contrôler de manière à réduire la fatigue sur les actionneurs. On constate en effet en simulation que les sollicitations sur les actionneurs du mode T sont moins élevées que sur le RMOSC proposé dans la thèse. Comparons la Figure 119 et la Figure 158. Ces figures montrent le groupe Hbank est moins sollicité dans le cas du mode T, réduisant la fatigue de ce actionneur.

14.5.3 Conclusions sur la simulation

En simulation et comparativement au mode T, le RMOSC conçu dans la thèse montre des résultats encourageants. On constate à priori une amélioration des performances et une anticipation appréciable de l'effet du xénon. Cependant, comme précisé en 14.5.2, ces résultats sont à contraster avec :

- La non considération de la fatigue des actionneurs dans la thèse. On observe en effet des sollicitation importantes des actionneurs par rapport au mode T.
- Les modèles utilisés pour la simulation du mode T et du RMOSC différents.
- L'utilisation de bandes mortes dans le cas du mode T. La présence de bandes mortes impacte nécessairement les performances atteignables pour le mode T.
- Les scénarios de simulation similaires mais pas tout à fait identiques.

Partie IV.
Conclusions et Perspectives de la Thèse

15 Conclusions

Après avoir présenté le contexte et les problématiques de la thèse la Partie I présente un état de l'art relatif à l'IS et à l'automatique.

La Partie II présente une méthode d'IS reposant sur des concepts et les relations entre ces concepts (présentés dans le chapitre 5), jugés nécessaires pour l'IS de Framatome et formalisés dans un méta modèle unifié. Différents DSML reposant sur ces concepts et relations sont proposés dans 6. L'ensemble de ces DSML permettent de modéliser les systèmes de Framatome et de vérifier les modèles conçus au travers de différentes vues. Un premier DSML, utilisé dans la vue exigences, consiste en un langage d'écriture structurée basé sur le principe des boilerplates. Les patrons textuels restreignent les libertés de l'auteur des exigences au juste nécessaire pour concevoir selon les pratiques de Framatome. Ce DSML participe ainsi à la vérification des modèles dans la partie descendante de RADIV (i.e. les modèles sont-ils bien construits ?). Le reste consiste en une série de DSML s'attachant aux vues contextuelle, logique, organique et comportementale. Dans leur ensemble, ces DSML supportent le passage entre le référentiel d'exigences d'un SOI et ses architectures. En effet, le référentiel d'exigences est placé de manière centrale dans la méthode, de manière à tisser les liens entre les exigences et tous les éléments de modélisation (en particulier les exigences, les modes, les fonctions, les scénarios opérationnels et les systèmes). Ensuite, l'ensemble des relations sémantiques entre ces différents éléments de modélisation, les vues et les liens qu'elles partagent ainsi que la démarche opératoire guident les utilisateurs, à partir du référentiel d'exigences, à concevoir des modes, puis des fonctions valides dans ces modes, jusqu'aux traitements entrée-sortie.

D'autre part, un ensemble de propriétés est proposé pour chacun des DSML. Ces propriétés ont pour ambition de vérifier la construction de chaque modèle. Elles expriment les règles de construction à respecter pour chaque modèle, les règles de sémantique opérationnelle et les règles de cohérence inter-modèles. Tout comme les boilerplates, elles participent à la vérification. De plus, la sémantique opérationnelle de certains DSML est fournie dans cette étude afin de les rendre simulables et participer ainsi à la vérification des modèles (par simulation).

La démarche opératoire est appliquée au cas *Core Control* dans 8. Elle aboutit notamment à la définition de la fonction nommée F0001, ayant pour mission de calculer les commandes du *Core Control* dans le mode automatique. Cette fonction correspond alors à la loi de régulation dont les spécifications sont définies dans la Partie III. Un ensemble d'exigences lui sont allouées, constituant le cahier des charges pour cette loi de régulation.

Après analyse du cahier des charges, la loi de régulation retenue consiste en un système de régulation hiérarchisé pour lequel le superviseur (MPC Controller) calcule les trajectoires de référence à partir du programme de puissance préétabli. L'autre constituant, le régulateur multi-objectif à gains séquencés (Multi-Objective Gain-Scheduled Controller) assure le contrôle rapproché autour des consignes des sorties à contrôler. Ce régulateur est un PI multivariable à gains séquencés. Le système de régulation hiérarchisé global est alors dénommé Régulateur Multi-Objectif Supervisé du Cœur (RMOSC). Un diagramme de spécification fonctionnelle est alors associé à la fonction F0001 définissant le RMOSC.

Bien que Framatome possède un code de simulation précis et qualifié, ce dernier est trop complexe pour être utilisé dans la démarche de conception employée dans la thèse. Ainsi, un modèle innovant simplifié du réacteur est proposé au chapitre 11. Il est constitué d'un modèle du circuit primaire, d'un modèle point du cœur et d'un modèle d'AO. Ce modèle possède une complexité réduite à l'essentiel pour : (i) faciliter l'élaboration de la commande du superviseur, (ii) la synthèse des régulateurs multi-objectifs en différents points de fonctionnement et enfin (iii) pour la simulation du RMOSC.

Le régulateur multi-objectif à gains séquencés est synthétisé et simulé seul sur le modèle non-linéaire du réacteur développé. Les exigences sont traduites en contraintes et objectifs de commande. Le problème de commande qui en résulte est un problème H_2 sous contraintes H_∞. Les résultats de simulation montrent des performances très encourageantes sur l'ensemble du domaine de fonctionnement considéré (i.e. 60%PN à 100%PN) respectant globalement les exigences. En particulier, les contraintes temporelles de sollicitation des actionneurs et du domaine de fonctionnement sont bien respectées en simulation, et ce grâce à la formalisation originale de contraintes fréquentielles (reformulation des exigences dans le « métier » de l'automaticien). Cependant, les trajectoires suivies par le régulateur sont fortement perturbées par l'absence de prise en compte de l'effet xénon par le modèle quasi-LPV utilisé. [Lemazurier *et al.*, 2016] et [Lemazurier *et al.*, 2018a] rendent compte des résultats de simulation obtenus sur le code de simulation qualifié de Framatome, pour les régulateurs obtenus durant la thèse, dans des cadres d'hypothèses spécifiques.

La conception et le paramétrage du superviseur sont présentés dans le chapitre 0. Sa conception s'appuie sur un algorithme de commande prédictive non linéaire et sur le modèle point non-linéaire proposé préalablement. Cet algorithme procède, en ligne, à une optimisation non-linéaire des variables de décision (méthode du point intérieur). En simulation, le superviseur montre des performances conformes à l'attendu.

Finalement, le RMOSC, combine les actions et les avantages des deux sous-systèmes conçus préalablement (le superviseur et le régulateur multi-objectif à gains séquencés). Son évaluation au chapitre 14 montre de bonnes performances ainsi que des exigences (cf. partie IS) respectées. Les rampes et l'effet xénon sont bien anticipés par l'action du superviseur tandis que le régulateur multi-objectif gère de manière robuste le contrôle des dynamiques plus rapides et le maintien des sorties à réguler dans leur domaine de fonctionnement autorisé. Il est comparé, en simulation, au mode T sur des scénarios de variation de puissance similaires. Bien qu'il soit difficile de comparer précisément ces deux solutions (utilisation de simulateurs différents, bandes mortes sur les sorties à contrôler pour le mode T, etc.), les résultats obtenus du RMOSC sont très prometteurs.

Avant d'aborder la question des perspectives, dressons un court bilan des **contributions de cette thèse**. Au-delà de la qualité des résultats obtenus, la contribution centrale réside dans la méthodologie globale mise en œuvre. Nous avons vu :

1. Une démarche outillée de formalisation du problème à l'intersection entre IS et Automatique. Elle fournit un environnement complet de modélisation des systèmes et de vérification des modèles en s'appuyant sur un ensemble de DSML interopérables. De plus, elle s'appuie sur la traduction formelle des exigences en contraintes mathématiques (dont la prise en compte garantie sous réserve de faisabilité du problème de commande). Enfin, elle utilise un modèle simplifié du *Core Control* à des fins de synthèse de la loi de commande, mais aussi de simulation et *in fine* de vérification des exigences dans les phases de conception.
2. Une stratégie de commande multivariable et multi-objectif visant une gestion optimisée des compromis, à partir d'une architecture de commande lisible et de complexité juste suffisante, et quoiqu'il en soit flexible : elle résulte d'un choix *a priori*. Cette stratégie permet une conception accélérée de nouvelles solutions. En effet, nous l'avons vu quand il s'est agi de considérer que le débit primaire pouvait être modulé (cf. [Lemazurier *et al.*, 2018a]), ce qui n'avait jamais été fait auparavant. Au contraire, les solutions de pilotage classiques procèdent de manière plus ou moins empirique par réglage de chaines de régulation découplées, source de limitation quant aux performances atteignables, tandis que les notions de robustesse sont rarement explicitées. Le réglage est effectué directement sur le code de simulation qualifié de Framatome, d'où un temps de mise au point possiblement allongé.
3. Une solution méthodologique globale proposée offrant à Framatome une voie conceptuelle de lois de commande alternative. Elle permet d'ores et déjà une évaluation rapide quant à l'intérêt, par exemple, d'introduire un nouveau moyen d'action pour gagner en flexibilité de pilotage du réacteur [Lemazurier *et al.*, 2018a].

Hormis le mémoire et les rapports internes, ce travail a donné lieu à la rédaction de plusieurs **articles** :

- [Lemazurier *et al.*, 2017a] propose une commande de la température moyenne et de l'AO uniquement par les barres en supposant la gestion du bore externalisée. Dans cette version on ne considère que deux actionneurs et deux sorties à contrôler.
- [Lemazurier *et al.*, 2017b] propose une approche de conception basée sur les principes normatifs de l'IS pour passer des exigences aux architectures.
- [Lemazurier *et al.*, 2018a] propose l'ajout de l'actionneur débit primaire. Cette version de régulateur montre la capacité de contrôler les températures branches chaude et froide séparément ainsi que l'AO par l'utilisation de trois actionneurs, les groupes de grappes et le débit primaire.
- [Lemazurier *et al.*, 2018b] propose l'utilisation du régulateur multi-objectif à gains séquencés (tel que présenté dans 12.4) pour le contrôle complet d'un cœur de REP, i.e. le contrôle de la température moyenne, de l'AO et de la position du groupe de grappes Pbank.

16 Perspectives

Dans ce chapitre nous exposerons les perspectives possibles des travaux initiés durant la thèse autour de l'IS puis sur les aspects automatique.

16.1 Perspectives dans le domaine de l'Automatique appliquée au contrôle du réacteur

L'analyse de stabilité d'un système linéaire à paramètres variants dans le temps peut être formalisé au travers d'un problème d'optimisation sous contraintes LMI de dimension infinie. Plusieurs travaux proposent des conditions LMI de dimension finie, qui approximent le problème d'analyse de stabilité dans ce contexte. On peut citer les travaux de [Troffino et Souza, 2001], [Ramos et Peres, 2002], [Leite et Peres, 2004], [Oliveira *et al.*, 2005], [Oliveira et al., 2007], [Oliveira et Peres, 2005], [Oliveira et Peres, 2007], [Troffino, 1999] et [Peaucelle, 2000]. L'utilisation de fonctions de Lyapunov dépendant des paramètres dans ce domaine permet d'obtenir des conditions peu pessimistes et de fait utilisables pour une large gamme de problèmes d'analyse. Aussi, nous estimons qu'il serait intéressant d'apporter par ce biais une preuve (numérique) de stabilité du régulateur multi-objectif à gains séquencés.

Dans notre étude nous avons considéré un régulateur PI multivariable. Or différentes réalisations existent pour de tels régulateurs. [Rugh et Shamma, 2000] propose une réalisation différente, $u_K(t) = \int (K_P \dot{x} + K_I e) dt$, équivalente dans le cas LTI, mais pas dans le cas LPV. D'après les auteurs cette écriture donnerait de meilleurs résultats en présence de variations rapides des paramètres. Il conviendrait d'analyser ici l'impact de ce choix d'implémentation.

Par ailleurs, il conviendrait d'approfondir la question de la modélisation des capteurs (incertitude quant à la dynamique et bruits), et de prendre en compte plus finement la robustesse vis à vis des incertitudes de sortie et la sensibilité aux bruits de mesure. Ceci peut être réalisé sans difficultés au travers de la méthodologie proposée, en procédant par empilement de contraintes additionnelles. Il convient dès lors de ne pas s'en priver, même si le conditionnement du système semble plutôt favorable dans la bande de fréquence considérée.

En ce qui concerne le superviseur, la thèse n'avait pas pour ambition d'optimiser le paramétrage de l'algorithme. Elle constitue davantage une preuve de concept. Cette étude peut être approfondie. Par ailleurs, l'incidence de l'utilisation d'un observateur reconstruisant les variables d'état du réacteur non mesurées pourrait être étudiée. Par ailleurs, l'usage d'un modèle axial du réacteur tel que proposé en Annexe F constitue une voie intéressante pour rendre compte plus précisément des oscillations de xénon dans le cœur au sein du superviseur. On pourrait, au-delà,

envisager un modèle axial possédant un minimum de 3 mailles, et évaluer l'impact d'un tel choix (complexifiant mais pas rédhibitoire) sur les performances atteignables.

Si les performances du RMOSC sont encourageantes, les sollicitations des actionneurs sont élevées, comparées au mode T. On pourrait alors envisager d'utiliser des bandes mortes sur les sorties à contrôler, à l'instar du mode T, afin d'en réduire les sollicitations. Or, la version proposée contient des intégrateurs (PI multivariable cf. 12.1) ce qui rend l'utilisation de bandes mortes plus complexe. Il est nécessaire, dans ce cas, d'utiliser des filtres anti-emballement. Une alternative possible aux bandes-mortes serait d'envisager une solution hybride, utilisant les capacités d'anticipation du superviseur pour activer ou désactiver le régulateur multi-objectif lorsque les sorties sortent de leurs domaines autorisés.

16.2 Perspectives IS

Tout d'abord, en ce qui concerne l'outil proposé de formalisation des exigences basé sur les boilerplates, de nouveaux boilerplates spécifiques à certaines pratiques de Framatome peuvent être spécifiés pour enrichir davantage les capacités et mieux structurer le langage. De plus, la sémantique des éléments mutables de ces patrons peut être utilisée afin d'analyser et comparer les éléments modélisés et vérifier plus finement la cohérences des exigences dans leur référentiel (ce qui est similaire à ce qui est proposé par l'outil RAT présenté dans 2.3.1).

Ensuite, la thèse n'a pas eu pour ambition de développer un environnement complet de simulation des DSML bien que les sémantiques opérationnelles aient été fournies pour certains d'entre eux. Une perspective intéressante serait donc de continuer à développer les capacités de simulation des outils. De plus, il peut être envisagé de tirer parti des travaux de [Nastov, 2016] pour implémenter des capacités de preuve de propriétés statiques, sous la forme d'un analyseur OCL (Acceleo de Obeo[11]) et de propriété dynamiques en développant un *model checker*.

En outre, il serait intéressant d'implémenter, en cohérence avec les outils proposés dans la thèse, l'analyse d'impact de la modification d'exigences évoquée dans 5.3.3.

Bien que la méthode d'IS proposée soit centrée sur les systèmes de type I&C, elle est fondée sur un méta modèle unifié et unique. En ce sens, elle est ouverte à l'intégration plus générale de tout type de système. Par exemple, les systèmes

[11] http://www.eclipse.org/acceleo/ dernière visite 15/11/2017

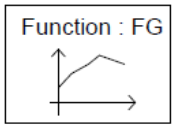
hydrauliques seraient un bon sujet d'étude car ils sont largement utilisés dans le cadre des REP. Pour cela, il faut enrichir le méta modèle de manière à prendre en compte les besoins du métier de l'hydraulique.

L'environnement de modélisation et de simulation pour le *Core Control* proposé dans la thèse pourrait être étendu à d'autres systèmes de contrôle-commande. Pour cela, il suffit de disposer d'un modèle simplifié de comportement dynamique du système devant être contrôlé.

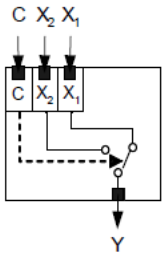
Enfin, il serait intéressant de développer des modèles simplifiés pour l'ensemble des systèmes de la centrale nucléaire qui, à l'instar de celui développé dans la thèse pour le réacteur, permettraient de vérifier rapidement en simulation la faisabilité des exigences dans les phases amont de la conception (phase descendante du cycle RADIV).

Annexes

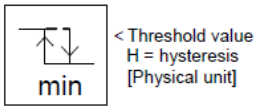
Annexe A. Blocs de spécification Framatome



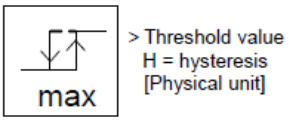
Générateur de fonction – Le signal de sortie correspond à l'interpolation linéaire de la fonction $Y = FG(X)$ évaluée en le signal d'entrée. Les valeurs du signal de sortie sont saturées aux maxima décrits dans la fonction FG



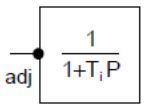
Sélecteur – X_1 et X_2 sont des signaux analogiques et C est logique. Si $C = 1$ alors $Y = X_2$, sinon $Y = X_1$



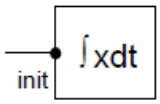
Hystérésis inférieure – La sortie binaire du bloc passe de 0 à 1 quand l'entrée est inférieure à *Threshold*. Elle passe de 1 à 0 quand l'entrée devient supérieure à *Threshold* + *Hysteresis*.



Hystérésis supérieure – La sortie binaire du bloc passe de 0 à 1 quand l'entrée est supérieure à *Threshold*. Elle passe de 1 à 0 quand l'entrée devient inférieure à *Threshold* + *Hysteresis*.



Filtre du premier ordre de constante de temps T_i - quand adj = 0. Si adj = 1 alors la sortie est identiquement égale à l'entrée.



Intégrateur – La sortie est égale à l'intégrale du signal d'entrée par rapport au temps. Le signal booléen d'entrée *init* initialise la sortie à 0 (quand *init* = *1*)



Max – La sortie est égale au max des entrées



Min – La sortie est égale au min des entrées

Annexe B. Boilerplates considérées

TYPE	*NAME*	*CONTENT*
Suffix	C-BP0	… after <event>
	C-BP2	… at least <quantity> times per <unit>
	C-BP3	… before <event>
	C-BP4	… during <state>
	C-BP5	… every <quantity> <unit>
	C-BP7	… for a period of at least <quantity><unit>
	C-BP12	… within <quantity><unit>
Main	C-BP14	<system> shall <action>
	C-BP18	<system> shall be <entity>
	C-BP27	<system> shall be <state>
Prefix	C-BP27	If <event>, …
	C-BP28	If <state>, …
	C-BP31	While <state>, …

Tableau 17: Boilerplates de CESAR considérées

TYPE	*NAME*	*CONTENT*
	C-BP1	… of <entity>
	C-BP12	… within <definition domain><unit>
Main	C-BP14	variable <variable> shall <action>
	C-BP15	variable <variable> shall be <state>
	C-BP16	variable <variable> shall <action>
	C-BP17	<system> shall provide <system> with <service>
	C-BP18	<system> shall provide <system> with <flow>
	C-BP19	<system> shall provide <system> with <item>
Prefix	C-BP27	If <(condition)>, …
	C-BP28	While <(condition)>, …

Tableau 18: Proposition d'ajout de boilerplates pour notre étude

Annexe C. Robustesse MIMO

La robustesse d'un asservissement traduit la capacité d'un système bouclé à rester stable malgré les erreurs de modèle.

Dans le cas monovariable, la marge de module notée M_m représente la distance du lieu de Nyquist au point -1. Il est possible d'utiliser les marges monovariables pour analyser un système multivariable en ouvrant fictivement la boucle d'asservissement pour ensuite analyser transfert par transfert les différentes marges (par exemple marge de gain, phase ou retard). L'information qui s'en dégage peut s'avérer précieuse pour l'évaluation de l'asservissement car elles représentent la robustesse de l'asservissement sur chaque voie séparée l'une de l'autre. Cependant, ces marges ne garantissent pas une robustesse multivariable, i.e. pour des variations simultanées sur chaque voie.

La marge de module M_m se généralise pour les systèmes multivariables. Elle est définie à partir de la fonction de sensibilité S du système bouclé par l'équation :

$$M_m = \min_\Omega \underline{\sigma}(1 + L(jw)) = \frac{1}{\underline{\sigma}(S(jw))} = \frac{1}{\|S\|_\infty}$$

La fonction de sensibilité peut être prise en entrée ou en sortie du système bouclé.

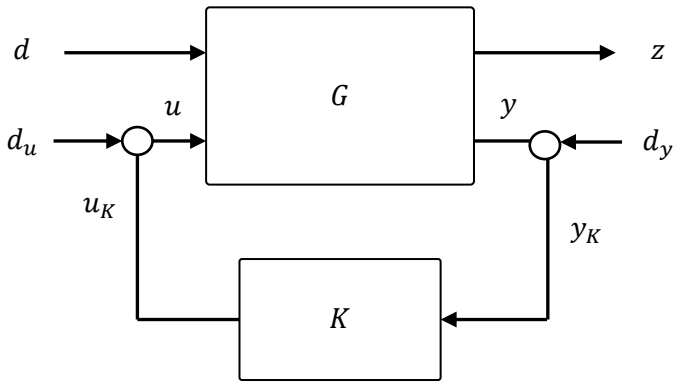


Figure 122: Marge de robustesse MIMO

- Fonction de sensibilité en entrée :

Elle indique la sensibilité de l'entrée u du système aux bruits sur les entrée d_u. Elle est donnée par l'équation suivante :

$$S_e(s) = (I + K(s)G_r(s))^{-1}$$

- Fonction de sensibilité en sortie :

Elle indique la sensibilité de la sortie y_K du système aux bruits sur les sorties d_y. Elle est donnée par l'équation suivante :

$$S_s(s) = (I + G_r(s)K(s))^{-1}$$

Dans le cadre de notre étude, pour un compromis stabilité – robustesse convenable, nous avons choisi une marge de module au moins égale à 0,5. Ainsi on a : $M_m = \frac{1}{\|S\|_\infty} \geq 0{,}5$ et donc $\|S\|_\infty \leq 2$, soit $6\ dB$ environ. Ceci garantit les marges de gain et de phase multivariables suivante :

$$M_g =]m_{gd}\ ,\ m_{ga}[\supset]\tfrac{1}{M_m+1}\ ,\ \tfrac{1}{1-M_m}[$$

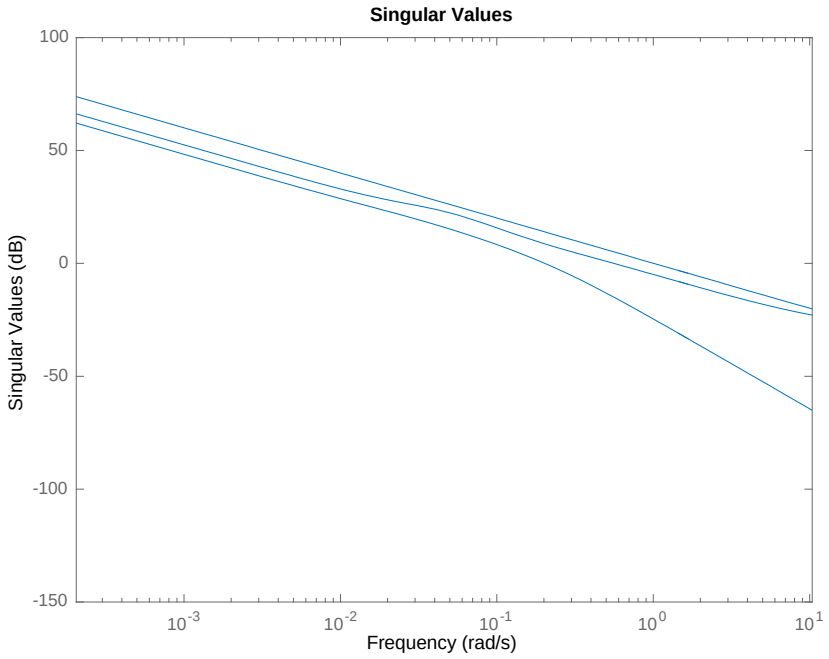
$$M_\phi =]m_{\phi d}\ ,\ m_{\phi a}[\supset]-2\arcsin\left(\tfrac{M_m}{2}\right)\ ,\ 2\arcsin(\tfrac{M_m}{2})[$$

Annexe D. Conditionnement du système

Selon [Maciejowski, 1989] le conditionnement d'un système est une caractéristique qui mesure la difficulté à inverser le système. Il est exprimé comme suit :

$$cond(G) = \frac{\bar{\sigma}(G)}{\underline{\sigma}(G)}$$

Où G est le système, $\bar{\sigma}(G)$ est la plus grande valeur singulière de G et $\underline{\sigma}(G)$ la valeur singulière de G. La figure suivante trace les valeurs singulières du système linéarisé du cœur en fonction de la fréquence (en radians par seconde). En général, on dit qu'un système est bien conditionné si son conditionnement est proche de 1 et mal conditionné si sont conditionnement est très supérieur à 1.



Le conditionnement, calculé à partir des données précédentes du système a été reporté dans le tableau suivant en fonction de la fréquence :

Pulsation (rad/s)	**0.0001**	**0.001**	**0.01**	**0.1**	**1**	**10**
$\boldsymbol{cond = \frac{\bar{\sigma}}{\underline{\sigma}}}$	3.80	3.80	3.70	4.12	17.97	176.1

Annexe E. Du débit de dilution/borication à la commande de bore

Ce qui suit présente la méthode de calcul de l'impact des dilutions et borications sur la concentration en bore.

Le schéma de principe de l'injection de bore dans le primaire est présenté Figure 64. Nous procédons par un bilan de masse dans le circuit primaire. Supposons que la masse totale du primaire (eau + bore) est M_T de concentration en bore C et qu'à tout instant une masse dm d'eau borée est injectée à concentration C_e dans le cœur et qu'une masse dm d'eau borée à la concentration du primaire C est extraite du primaire.

La concentration du circuit primaire est donnée par l'équation $C = M_{bore}/M_T$. La masse totale dans le cœur étant constante on écrit la variation de concentration de bore dans le cœur par :

$$dC = \frac{dM_{bore}}{M_T}$$

La variation de masse de bore dM_{bore} est égale à la masse entrante de bore moins la masse sortante de bore. On pose M_s la masse sortante de bore, et M_e la masse entrante de bore telles que :

$$M_e = dm.C_e \qquad M_s = dm.C \qquad \text{(D.1)}$$

On a donc $dM_{bore} = M_e - M_s$. On déduit donc des équation (D.1) précédentes que :

$$dm = M_T.\frac{dC}{C_e - C}$$

et donc,

$$\frac{dm}{dt} = M_T.\frac{1}{C_e - C}.\frac{dC}{dt} \qquad \text{(D.2)}$$

Supposons qu'à l'état initial la masse injectée (NB : en même temps retirée) dans le cœur est nulle et à l'état finale elle vaut M_f. La concentration initial dans le cœur vaut C_i, la concentration finale (après injection) sera notée C_f. On intègre alors (D.2) entre l'état initial et final.

$$\int_0^{M_f} dm = -M_T \int_{C_i}^{C_f} \frac{1}{C - C_e} dC$$

On a alors :

$$M_f = M_T.ln(\frac{C_e - C_f}{C_e - C_i})$$

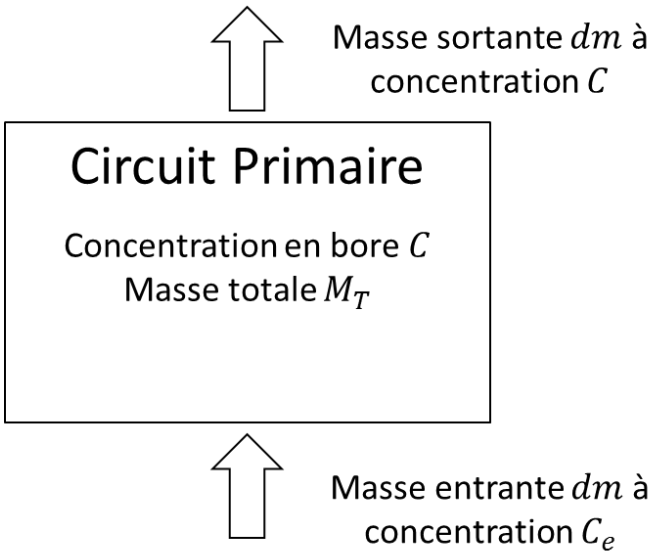


Figure 123: Schématisation du problème de variation de concentration de bore dans le primaire

Alors, les commandes données aux actionneurs sont :

- Dans le cas d'une borication : $\frac{dm}{dt} = Q_{bore}$
- Dans le cas d'une dilution : $\frac{dm}{dt} = Q_{dil}$

Si on reprend l'équation (D.2) on voit que dans le cas d'un borication elle devient :

$$Q_{bore} = \frac{dm}{dt} = M_T.\frac{1}{C_b max - C}.\frac{dC}{dt} \quad \text{(D.3)}$$

où C_b^{max} est la concentration du mélange boré injecté lors de la borication. Et dans le cas d'une dilution, (D.2) devient :

$$Q_{dil} = \frac{dm}{dt} = -\frac{M_T}{C}.\frac{dC}{dt} \quad \text{(D.4)}$$

Application numérique

On suppose que la masse totale du primaire est de 320 tonnes ($\mathrm{M_T} = 3{,}2.10^5\ \mathrm{kg}$) et on se place dans le cas d'un débit constant d'injection dans le cœur (bore ou eau selon borication ou dilution) de $\mathrm{Q_{dil}} = \mathrm{Q_{bore}} = 1\ \mathrm{kg/s}$.

- Cas de la dilution ($\mathrm{C_e} = 0$)

A concentration de bore (C) de 1000 ppm on déduit de 12'' :

$$\frac{\mathrm{dC}}{\mathrm{dt}} = -\frac{1000}{3{,}2.10^5}.1 = -3{,}125.10^{-3}\ \mathrm{ppm/s}$$

A concentration de bore de 100 ppm a :

$$\frac{\mathrm{dC}}{\mathrm{dt}} = -\frac{100}{3{,}2.10^5}.1 = -3{,}125.10^{-4}\ \mathrm{ppm/s}$$

On voit donc que pour un même débit d'injection la modification de concentration de bore varie fortement en fonction de la concentration initiale du bore dans le primaire (facteur 10 de 1000 ppm à 100 ppm). Moins il y a de bore dans le cœur moins le débit est efficace (de moins en moins de bore dans un circuit primaire de masse constante). NB : à concentration nulle la dilution n'a plus d'effet, ce qui est logique puisqu'il n'y a plus de bore à diluer.

- Cas de la borication ($\mathrm{C_e} = \mathrm{C_b}^{\max} = 10105\ \mathrm{ppm}$)

A concentration de bore (C) de 1000 ppm on déduit de 12' :

$$\frac{\mathrm{dC}}{\mathrm{dt}} = \frac{(10105-1000)}{3{,}2.10^5}.1 = 2{,}84.10^{-2}\ \mathrm{ppm/s}$$

A concentration de bore (C) de 100 ppm on a :

$$\frac{\mathrm{dC}}{\mathrm{dt}} = \frac{(10105-100)}{3{,}2.10^5}.1 = 3{,}13.10^{-2}\ \mathrm{ppm/s}$$

On note que l'impact d'une borication sur la concentration en bore est plus important que celui d'une dilution. Pour prendre en compte cette différence nous considérons la variation de concentration de bore en ppm/s notée u_Q définie telle que :

$$\frac{dC}{dt}(t) = u_Q$$

Ainsi, la relation entre la commande et la réactivité est linéaire. Pour retrouver la commande de dilution ou de borication (Q_{bore} et Q_{dil}) on définit la fonction F_{CB} de $\mathbb{R} \to \mathbb{R}^2$, permettant de convertir la commande de bore en commande de dilution-borication équivalente, telle que :

$$F_{CB}: u_Q \to F(u_Q) = \begin{pmatrix} Q_{bore} \\ Q_{dil} \end{pmatrix}$$

où,

$$Q_{bore} = \begin{cases} \frac{M_T}{(C_b max - C)}.u_Q & si\ u_Q \geq 0 \\ 0 & si\ u_Q < 0 \end{cases}$$

$$Q_{dil} = \begin{cases} 0 & si\ u_Q > 0\ \&\ C > 0 \\ -\frac{M_T}{C}.u_Q & si\ u_Q \leq 0\ \&\ C > 0 \\ 0 & si\ C = 0 \end{cases}$$

La Figure 124 montre le schéma bloc équivalent de la conversion de la commande de bore en dilution-borication.

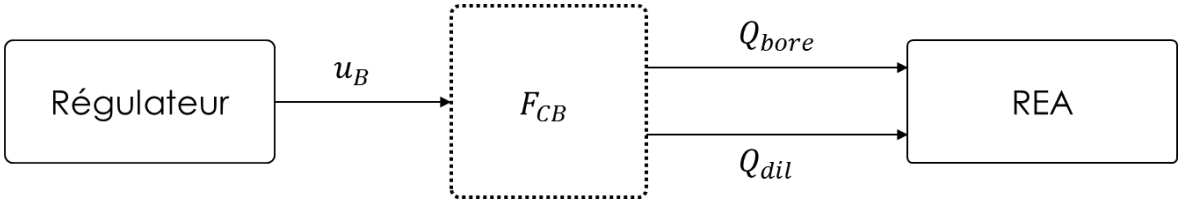


Figure 124: Schéma bloc de la conversion de commande de bore

Annexe F. Modèle multi-mailles

Le modèle multi-mailles modélise plus finement les variations axiales des paramètres du cœur que le modèle point. Les équations que nous considèrerons dans cette partie sont sensiblement les mêmes que pour la modélisation du cœur point. L'idée est que le modèle axial est divisé en plusieurs points (plusieurs modèles points) échangeant des neutrons par diffusion. On calcule alors la distribution axiale de puissance par le calcul des puissances de chaque cœur point indépendamment.

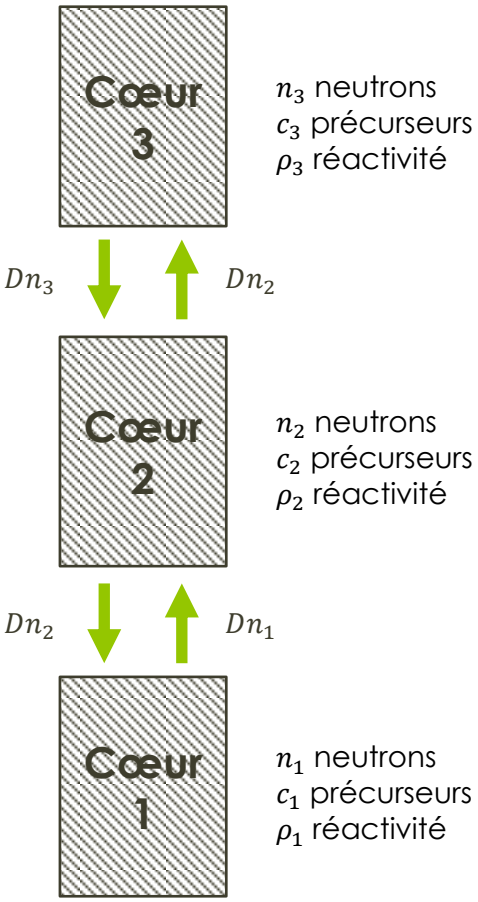


Figure 125: Illustration du principe de la modélisation

- Principe de la modélisation

Le principe de la modélisation d'un cœur axial est illustré sur la Figure 125. Cette figure montre un cœur divisé en 3 mailles. Chaque maille possède un nombre de neutrons rapides n_i, un nombre de précurseurs c_i et une réactivité ρ_i. Chaque maille, ou partie scindée du cœur envoie et reçoit des neutrons des cœur voisins par diffusion. Le nombre de neutrons reçus par le cœur i est proportionnel au nombre de neutrons de ces cœurs voisins et le nombre de nombre émis, par ce même cœur vers ses voisins, est proportionnel au nombre de neutrons qu'il possède. La loi

d'évolution des neutrons dans le cœur i suit alors une loi de diffusion que l'on définit par le coefficient de diffusion D.

Les équations de la neutronique du cœur point (partie précédente) sont conservées à chaque maille (i.e. partie du cœur). Prenons le cas du cœur 1 sur la Figure 125. On a alors :

$$\begin{gathered}\frac{dn_1}{dt} = \frac{\rho_1 - \beta}{l^*}.n_1 + \lambda c_1 + D(n_2 - n_1) \\ \frac{dc_1}{dt} = \frac{\beta}{l^*}.n_1 - \lambda c_1\end{gathered} \qquad \text{(E.1)}$$

L'équation (E.1) montre le terme de diffusion $D(n_2 - n_1)$ ajouté à l'équation de la dynamique des neutrons dans le cœur. Notons que le cœur 1 étant à l'extrémité (tout comme le cœur 3) n'échange des neutrons qu'avec le cœur 2. Le cœur 2 quant à lui échange à la fois avec 1 et 3. On a donc les équations suivantes :

$$\begin{gathered}\frac{dn_2}{dt} = \frac{\rho_2 - \beta}{l^*}.n_2 + \lambda c_2 + D(n_1 + n_3 - 2n_2) \\ \frac{dc_2}{dt} = \frac{\beta}{l^*}.n_2 - \lambda c_2\end{gathered}$$

Notons enfin que le cœur étant divisé également en trois parties les coefficients du modèle neutronique peuvent être supposés égaux[12].

- Couplage avec le modèle thermo-hydraulique du cœur

Dans cette partie nous allons montrer la relation entre le modèle neutronique du cœur axial et le modèle thermo hydraulique considéré.

Le modèle choisi pour chaque partie du cœur est un modèle thermo-hydraulique simple. Le flux à chaque maille i noté Φ_k^i pour s'écrire en fonction du débit massique Q_p, de la capacité thermique massique du fluide caloporteur C_p et des températures d'entrée-sortie de la maille. La Figure 126: Illustration du couplage thermo-hydraulique des mailles montre le principe de couplage entre les cœurs. Le cœur i, possède n_i neutrons rapides, c_i précurseurs, a une température d'entrée T_i et de sortie T_{i+1}. Les équations de la neutronique sont données sur la figure[13]. On peut alors écrire :

[12] En toute rigueur l'épuisement étant différents dans les différentes parties les coefficients sont en réalité légèrement différents.

[13] Attention, les équations de la neutronique données sur la figure sont valable pour $i \neq \{1, n\}$ avec n le nombre de mailles du cœur. En effet, comme nous l'avons vu dans la partie précédente, l'expression du terme de diffusion est différent pour les mailles situées aux extrémités.

$$\Phi_K^i = Q_p C_p (T_{i+1} - T_i)$$

De la même manière que 11.1.1 la puissance échangée Φ_K^i est donnée par un modèle s'simplifié de l'établissement de chaleur dans le cœur. On a alors dans le domaine de Laplace :

$$P_K^i(s) = \frac{1}{1+\tau_m s} \Phi_K^i(s)$$

avec τ_m la constante de temps d'établissement de la chaleur dans le cœur d'une maille et P_K^i la puissance échangée à la maille i. On peut exprimer les températures des mailles de la manière suivante :

$$\frac{dT_{i+1}}{dt}(t) = -\frac{1}{\tau_m} T_{i+1}(t) + \frac{1}{\tau_m} T_i(t) + \frac{1}{\tau_m Q_p C_p} P_K^i(t)$$

pour $i = \{1, \dots, n\}$. Enfin, puisque la puissance échangée dans la maille est proportionnelle au nombre de neutrons dans la maille i noté n_i tel que $P_K^i(t) = K_n n_i(t)$ $\forall\, i \in \{1, \dots, n\}$, on a :

$$\frac{dT_{i+1}}{dt}(t) = -\frac{1}{\tau_m} T_{i+1}(t) + \frac{1}{\tau_m} T_i(t) + \frac{K_n}{\tau_m Q_p C_p} n_i(t)$$

avec K_n le coefficient de conversion des neutrons en puissance.

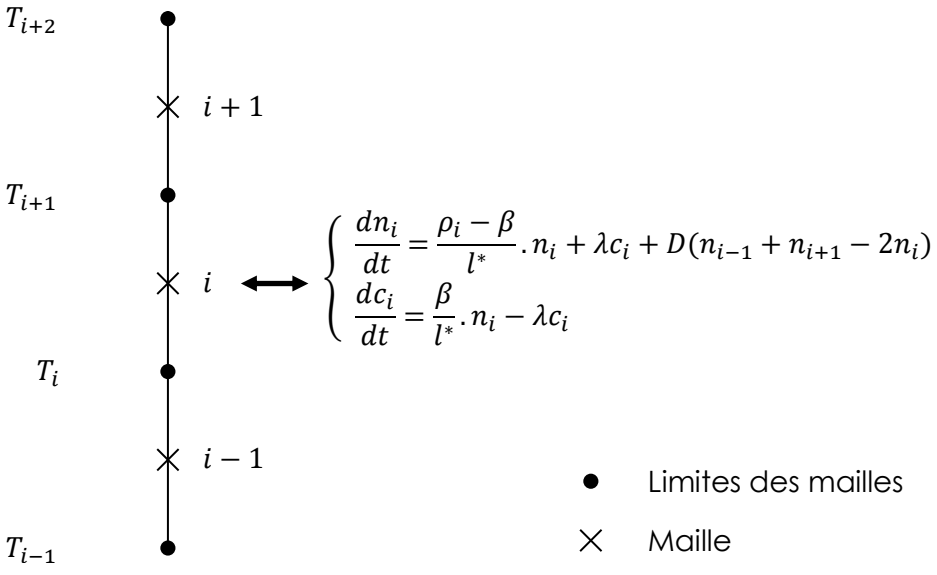


Figure 126: Illustration du couplage thermo-hydraulique des mailles

Annexe G. Démonstration du développement en série de Taylor de l'équation non-linéaire

Posons $n(t) = n^0 + \delta n(t)$ avec δn variable d'écart de n par rapport à l'équilibre. L'équation 11.e devient :

$$\frac{d\delta n}{dt}(t) = \frac{\rho(t)-\beta}{l^*}(n^0 + \delta n(t)) + \lambda(c^0 + \delta c(t))$$

$$\delta\dot{n}(t) = \frac{n^0}{l^*}\rho(t) - \frac{\beta}{l^*}\delta n(t) + \lambda\delta c(t) + \frac{\rho(t)\delta n(t)}{l^*} - \frac{\beta}{l^*}n^0 + \lambda c^0$$

D'après l'équation 11.e il vient aisément à l'équilibre que :

$$-\frac{\beta}{l^*}n^0 + \lambda c^0 = 0$$

De plus, ρ et δn étant proche de 0, on peut négliger le produit $\rho \times \delta n$ devant $\frac{n^0}{l^*}\rho$ et $\frac{\beta}{l^*}\delta n$. L'équation 30 devient donc :

$$\delta\dot{n}(t) = \frac{n^0}{l^*}\rho(t) - \frac{\beta}{l^*}\delta n(t) + \lambda\delta c(t)$$

Annexe H. Commande LQ

Considérons le système (Σ) régi par les équations d'état suivantes :

$$(\Sigma)\ \begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \quad x(0) = x_0 \\ y(t) = x(t) \\ z(t) = C_z x(t) \end{cases}$$

Où $x(t) \in \mathbb{R}^n$ désigne le vecteur d'état, $u(t) \in \mathbb{R}^m$ le vecteur des commandes, $y(t) = x(t)$ le vecteur des sorties mesurées et $z(t) \in \mathbb{R}^P$ le vecteur des sorties régulées.

Le problème de commande Linéaire Quadratique consiste alors à trouver un régulateur $K_{LQ} \in \mathbb{R}^{m \times n}$ par retour d'état garantissant :

- La stabilité interne du système bouclé (valeurs propres de $(A - BK_{LQ})$ à parties réelles négatives).

- La minimalité de $J_{LQ}(K_{LQ}) = \int_0^{t_f} x(t)^T Q_x x(t) + u(t)^T R u(t) dt$ un critère quadratique où $u(t) = -Kx(t)$.

Les matrices Q_x et R sont les matrices de pondération du critère LQ.

Lemme : *On suppose que* $\mathbf{R > 0}$ *et* $\mathbf{Q_x = Q_x^T \geq 0}$**.** *L'équation algébrique de Riccati* $\mathbf{A^T P + PA - PBR^{-1}P + Q_x = 0}$ *admet une solution unique* $\mathbf{P \geq 0}$ *si le triplet* $(\mathbf{Q_x, A, B})$ *est stabilisable et détectable.*

Sous les hypothèses du Lemme 1, la loi de commande par retour d'état $u(t) = -Kx(t)$ solution du problème de commande linéaire quadratique est obtenue pour $K_{LQ} = R^{-1}B^T P$ si P obeit à l'équation algébrique de Riccati et $P \geq 0$. On a alors $J_{LQ}(K_{LQ}) = \int_0^{t_f} x(t)^T Q_x x(t) + u(t)^T R u(t) dt = x_0^T P x_0$.

Annexe I. Résultats de simulation du régulateur à gains séquencés seul

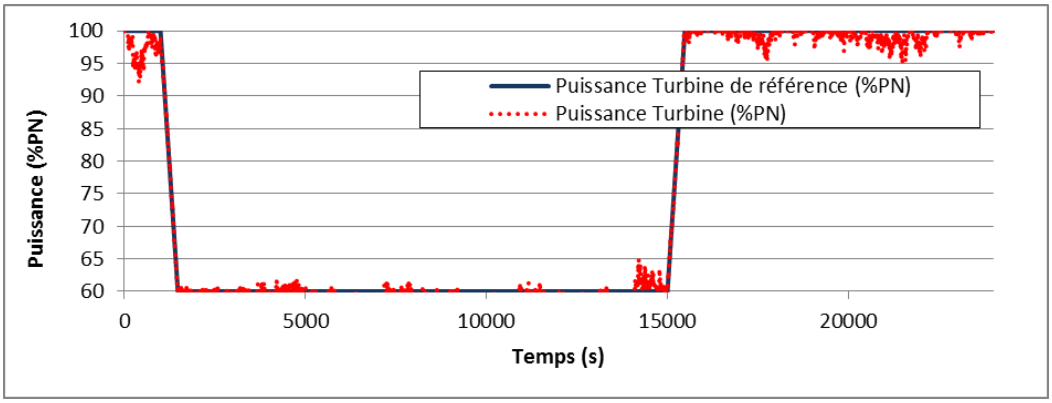


Figure 127: Tracé de la puissance turbine – Rampe de puissance 100%PN à 60%PN à 5%PN/min

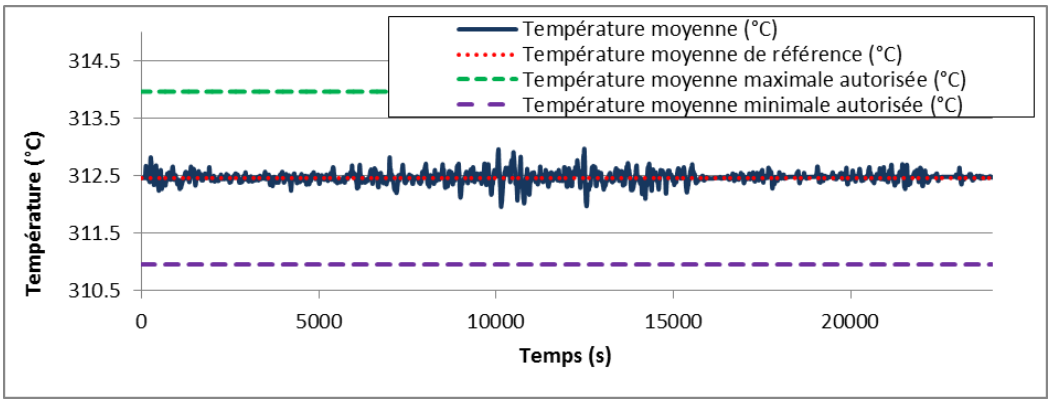


Figure 128: Tracé de la température moyenne - Rampe de puissance 100%PN à 60%PN à 5%PN/min

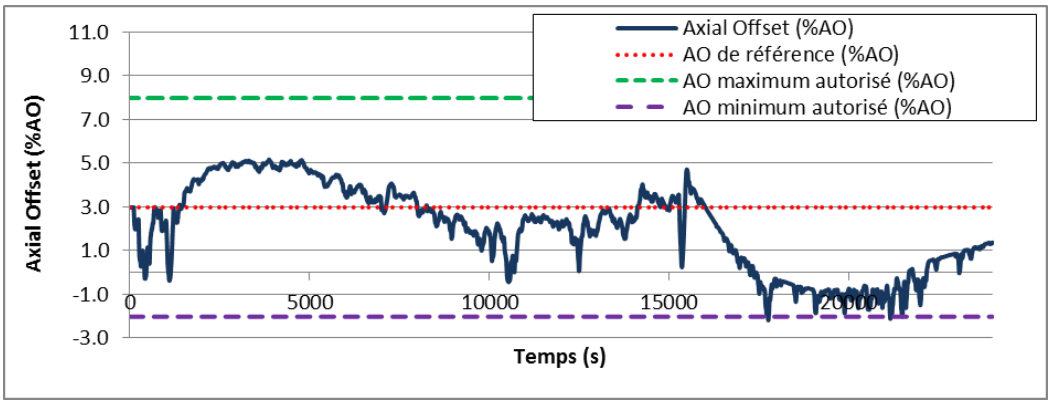


Figure 129: Tracé de l'AO - Rampe de puissance 100%PN à 60%PN à 5%PN/min

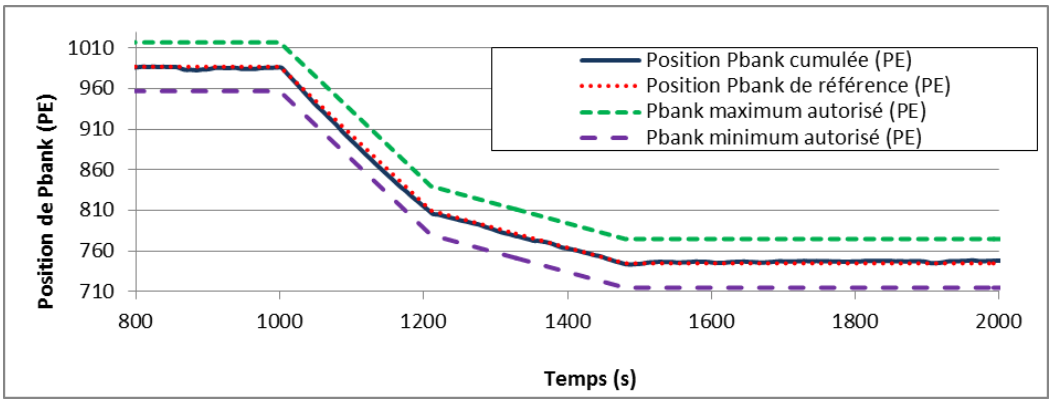


Figure 130: Tracé de la position de Pbank - Rampe de puissance 100%PN à 60%PN à 5%PN/min

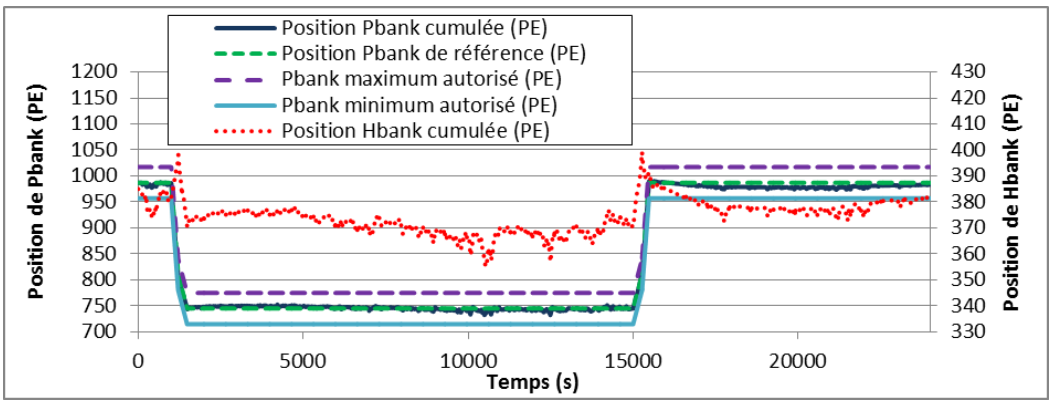


Figure 131: Tracé de la position des groupes - Rampe de puissance 100%PN à 60%PN à 5%PN/min

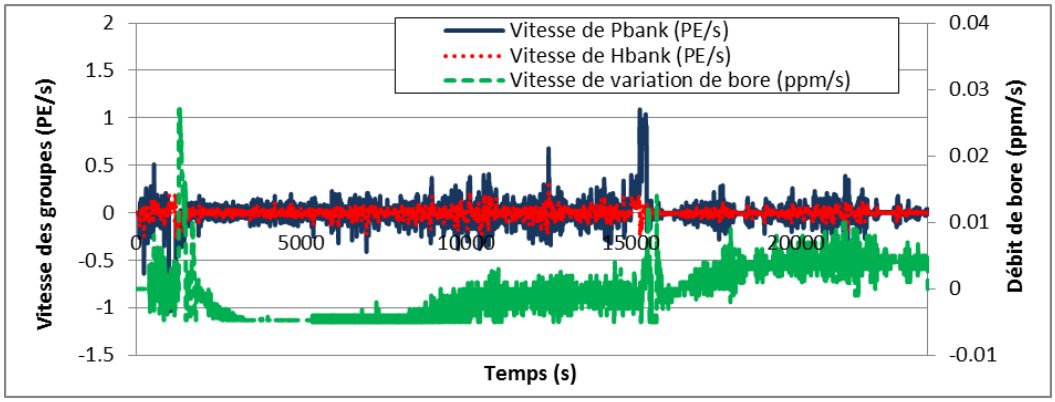


Figure 132: Tracé des vitesses et débit des actionneurs - Rampe de puissance 100%PN à 60%PN à 5%PN/min

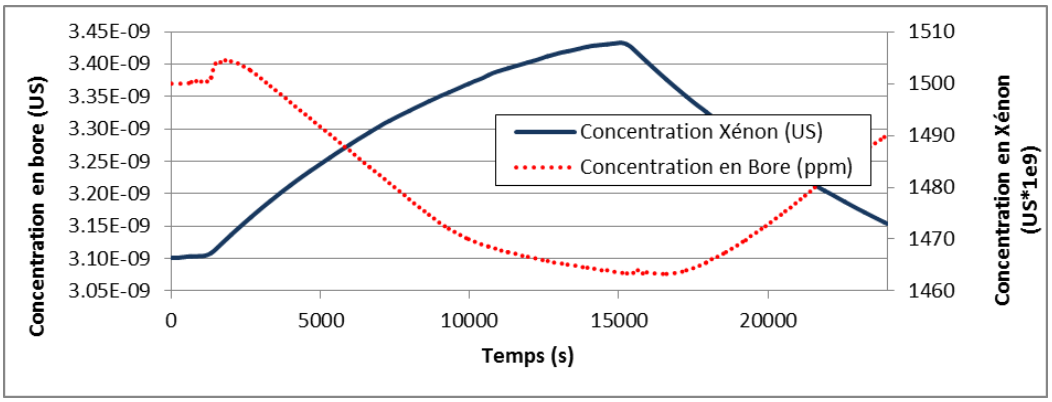


Figure 133: Xénon vs. Concentration en bore - Rampe de puissance 100%PN à 60%PN à 5%PN/min

Annexe J. Résultats de simulation Superviseur seul. Rampe de puissance 100%PN à 60%PN à 5%PN/min

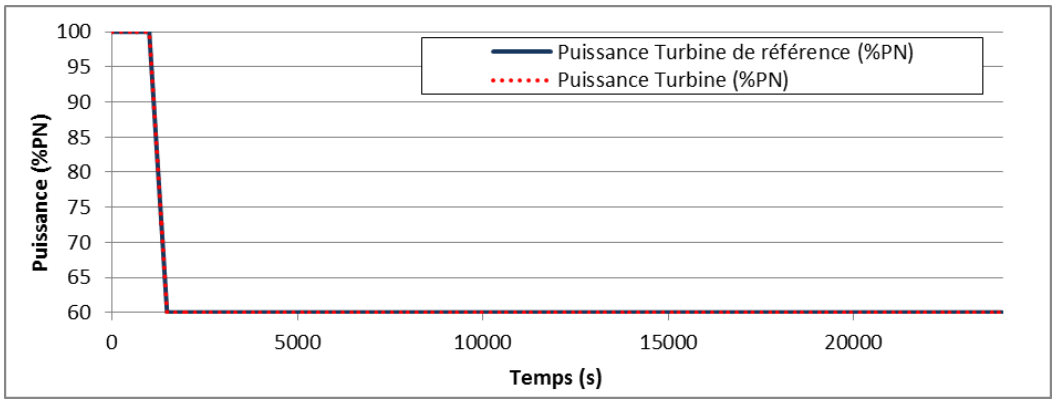


Figure 134: Tracé de la puissance turbine – Rampe de puissance 100%PN à 60%PN à 5%PN/min

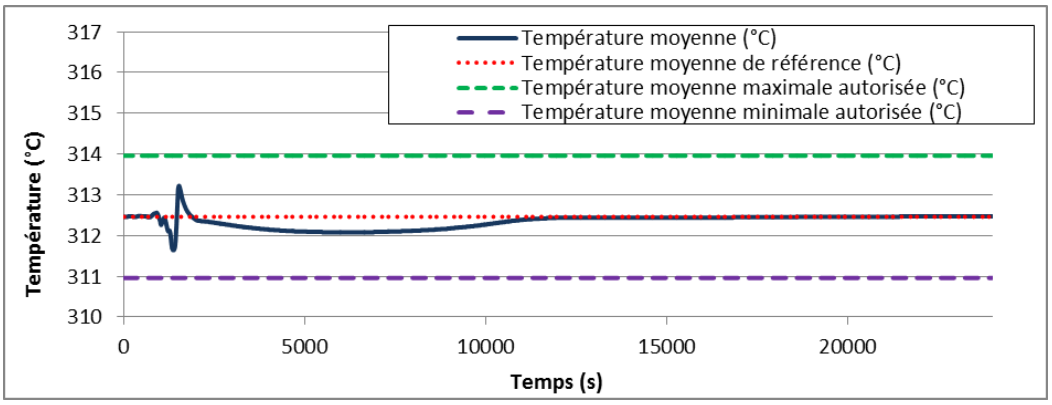


Figure 135: Tracé de la température moyenne - Rampe de puissance 100%PN à 60%PN à 5%PN/min

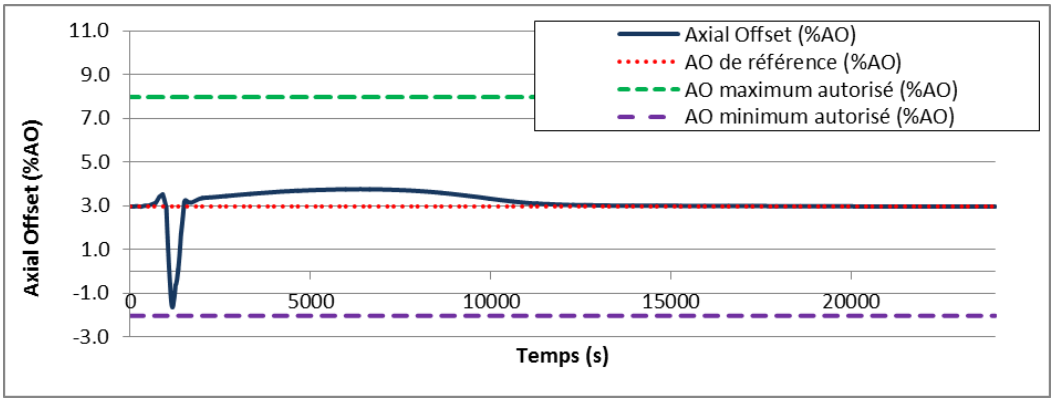


Figure 136: Tracé de l'AO - Rampe de puissance 100%PN à 60%PN à 5%PN/min

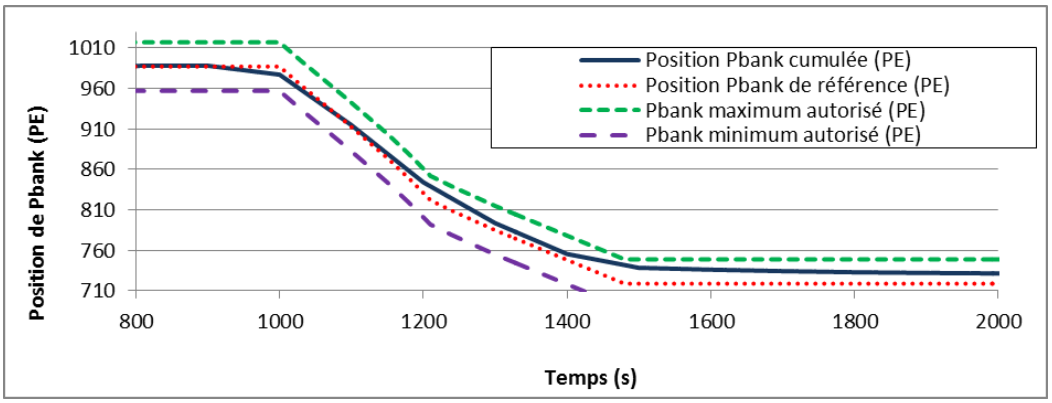


Figure 137: Tracé de la position de Pbank - Rampe de puissance 100%PN à 60%PN à 5%PN/min

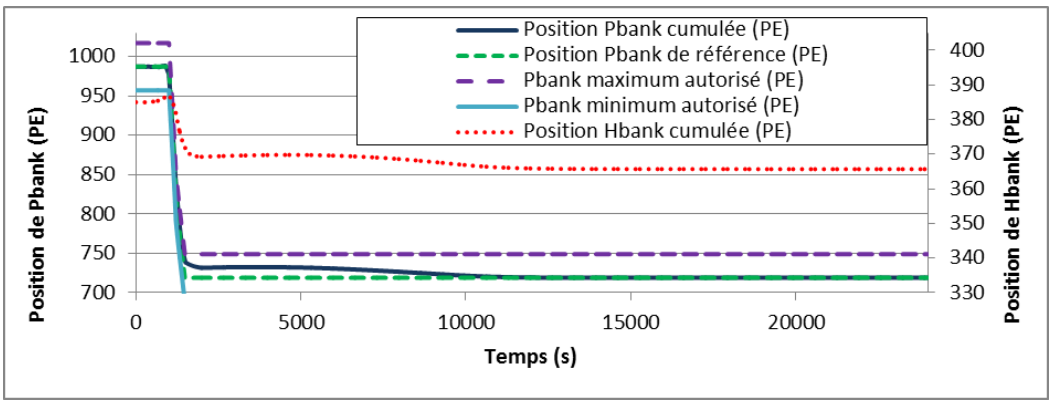


Figure 138: Tracé de la position des groupes - Rampe de puissance 100%PN à 60%PN à 5%PN/min

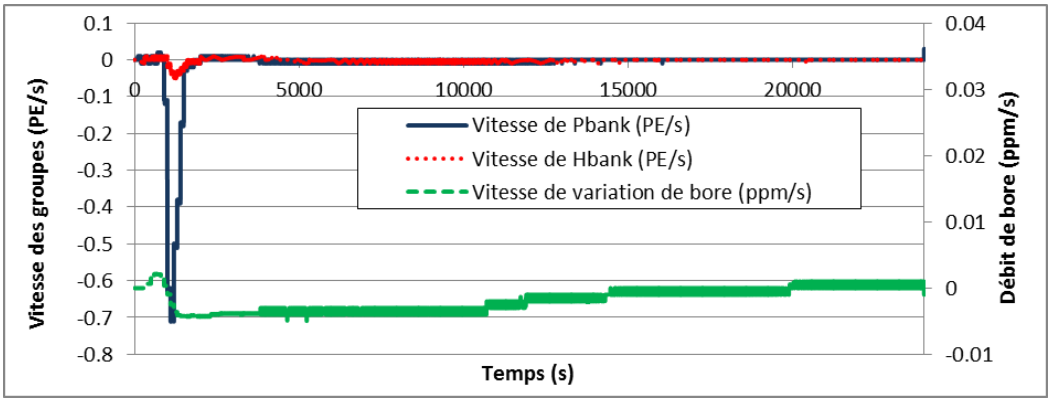


Figure 139: Tracé des vitesses et débit des actionneurs - Rampe de puissance 100%PN à 60%PN à 5%PN/min

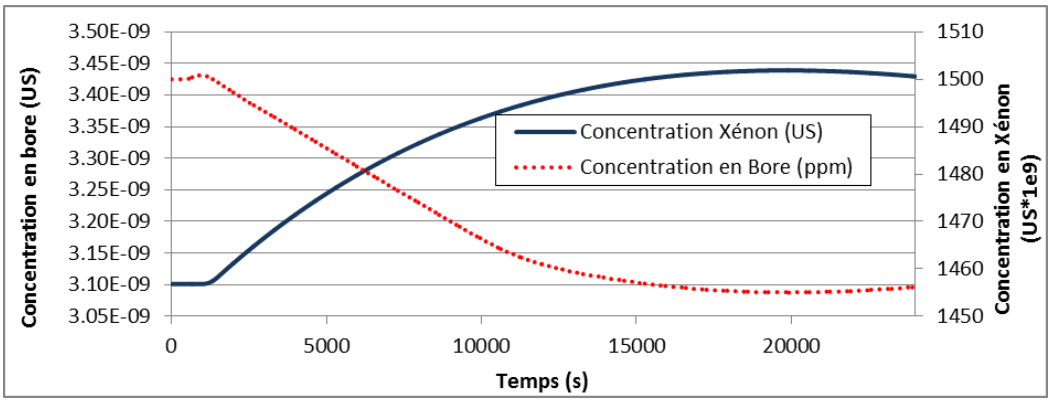


Figure 140: Xénon vs. Concentration en bore - Rampe de puissance 100%PN à 60%PN à 5%PN/min

Annexe K. Passage des exigences aux spécifications

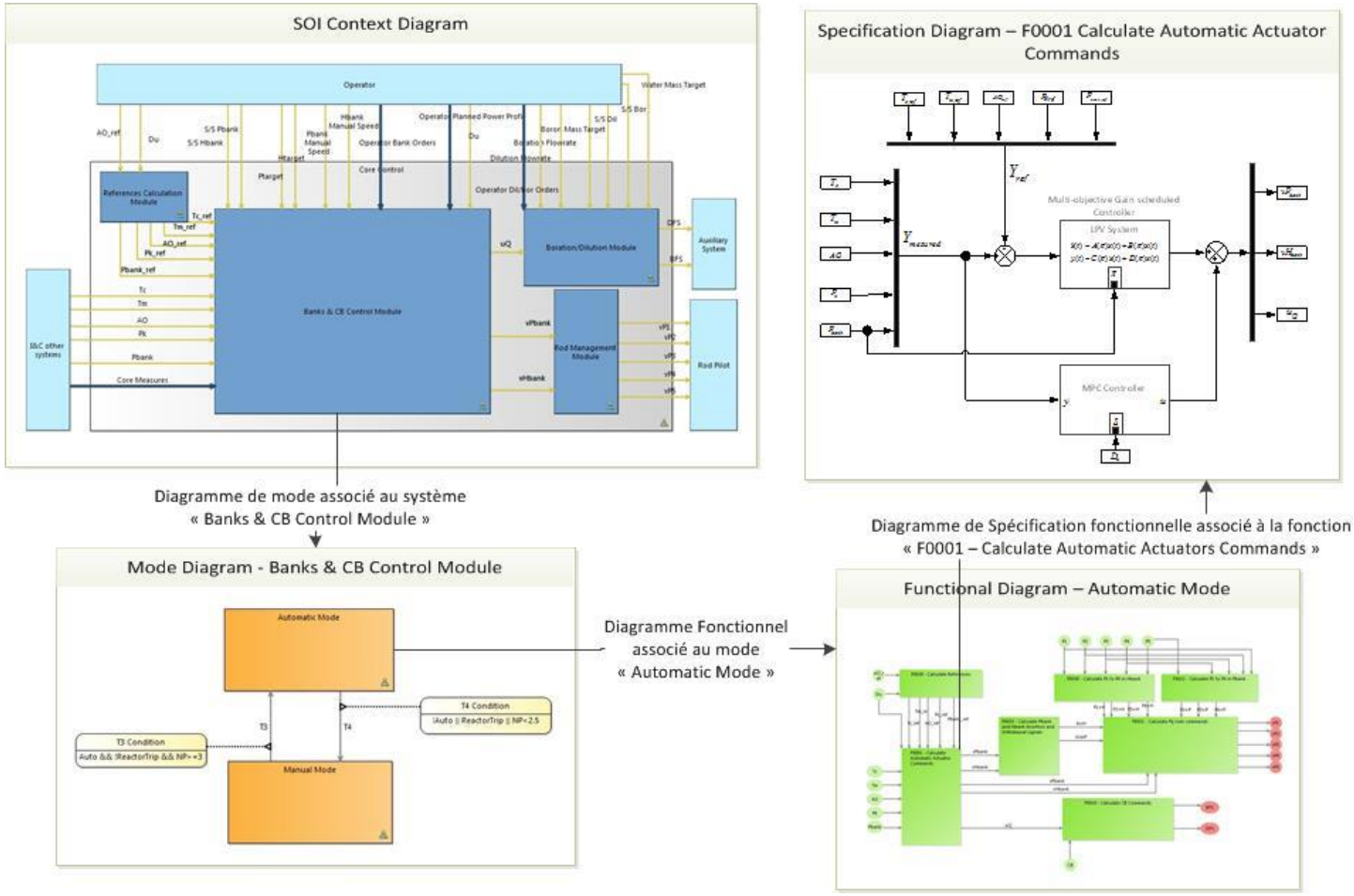


Figure 141: Illustration du passage du système aux spécifications

Annexe L. Allocation des exigences

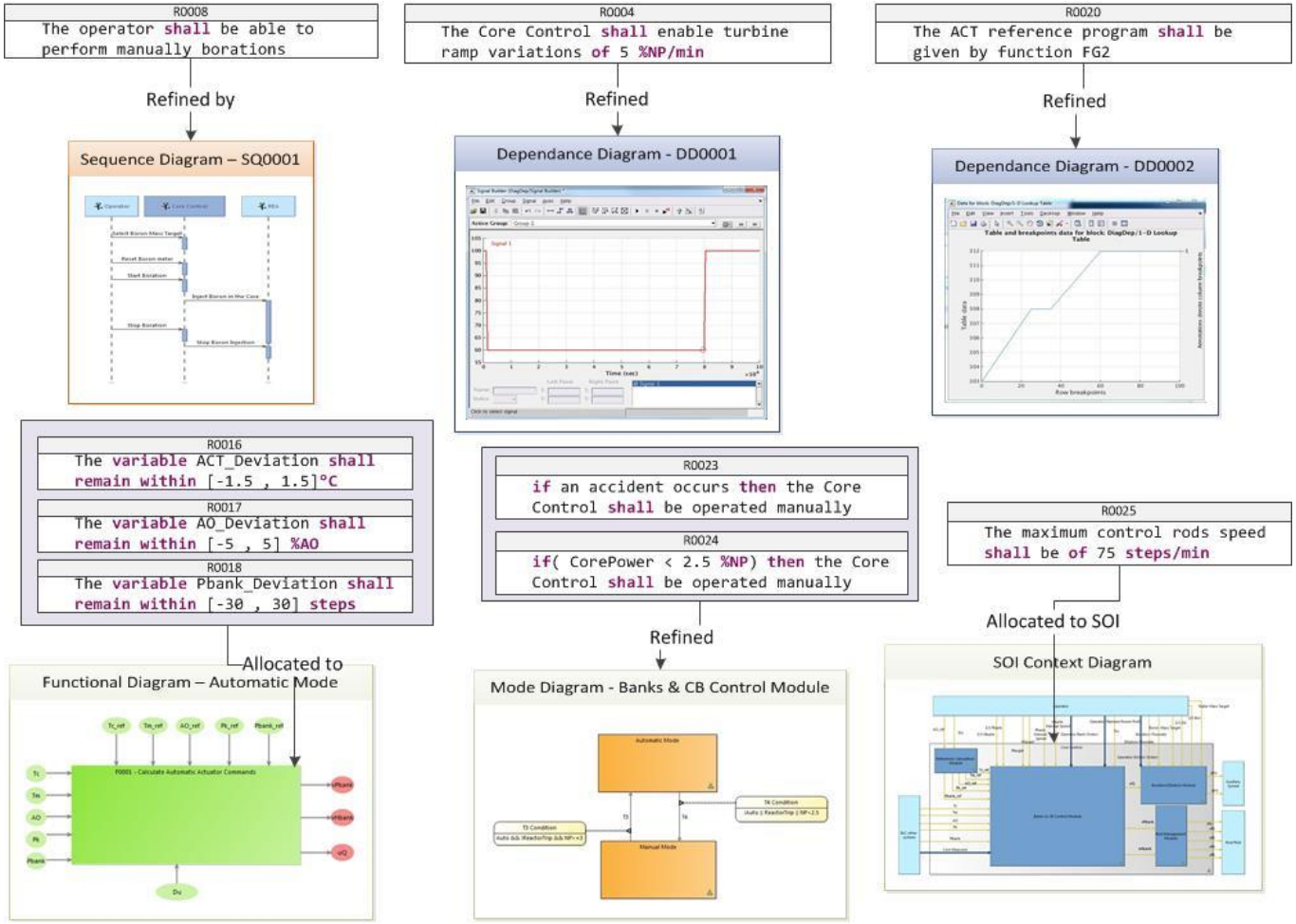


Figure 142 Illustration de l'allocation des exigences

Annexe M. Interactions entre les DSML

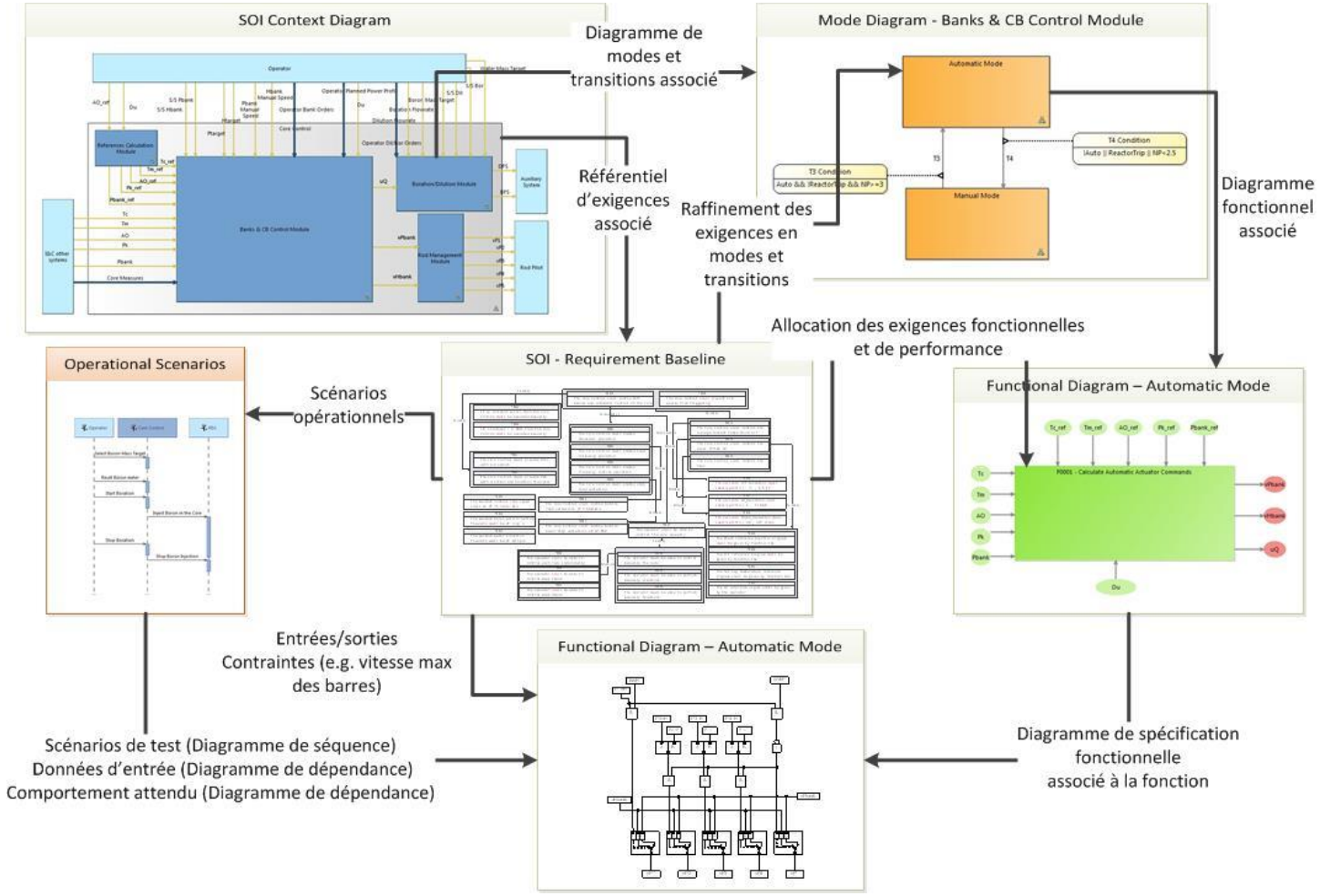


Figure 143: Schéma global des interactions entre les DSML

Annexe N. Diagrammes de spécification

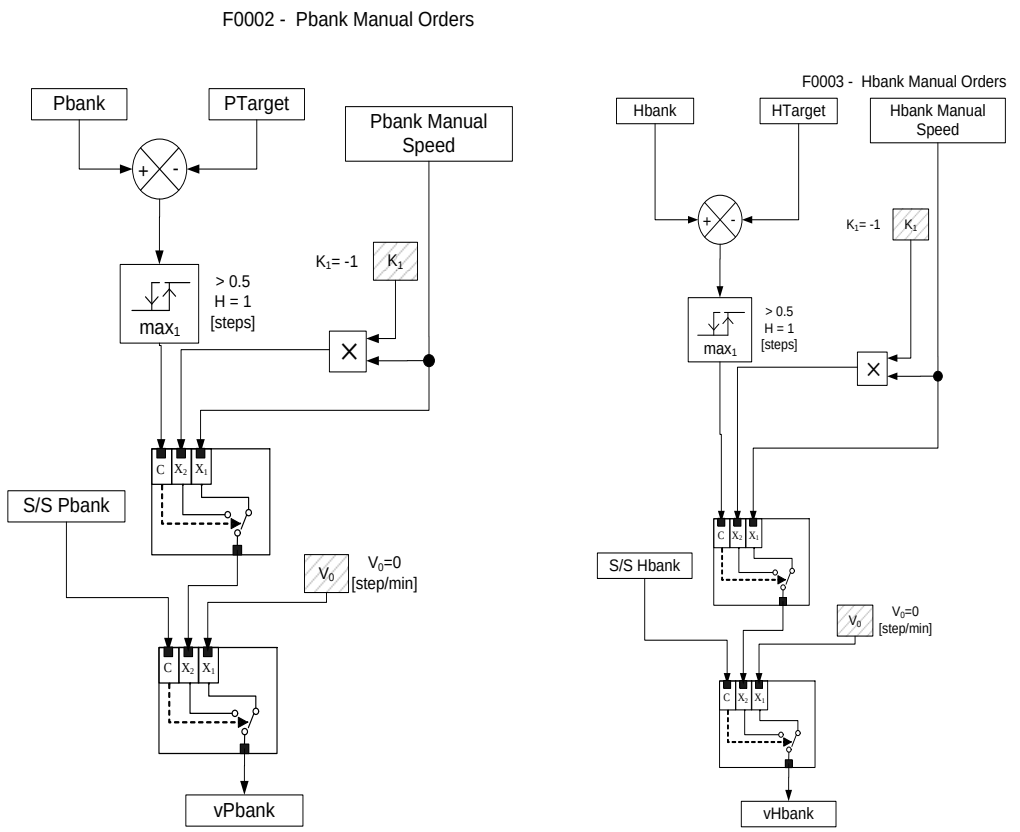


Figure 144 : Diagrammes de Spécification fonctionnelle des fonctions F0002 et F0003

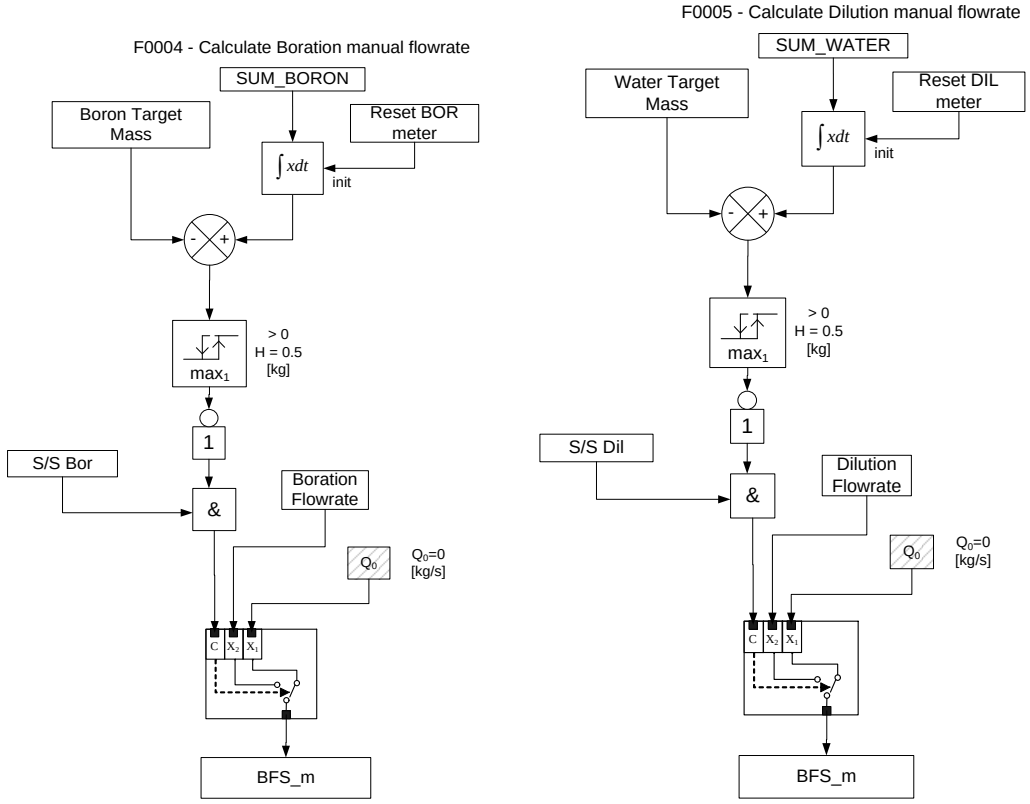


Figure 145:Diagrammes de spécification fonctionnelle des fonctions F0004 et F0005

F0020 - Calculate References

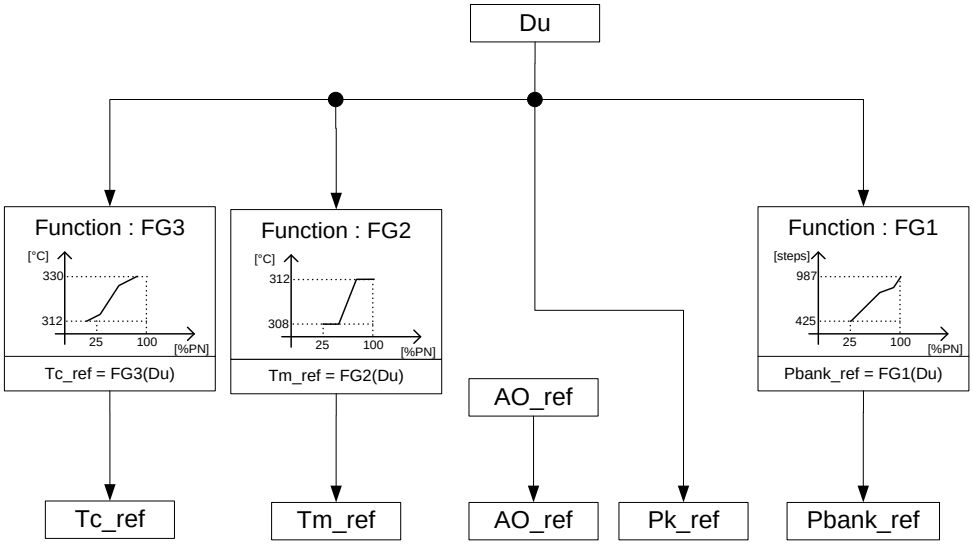


Figure 146: Diagramme de spécification fonctionnelle de la fonction F0020

F0010 – Calculate CB Commands

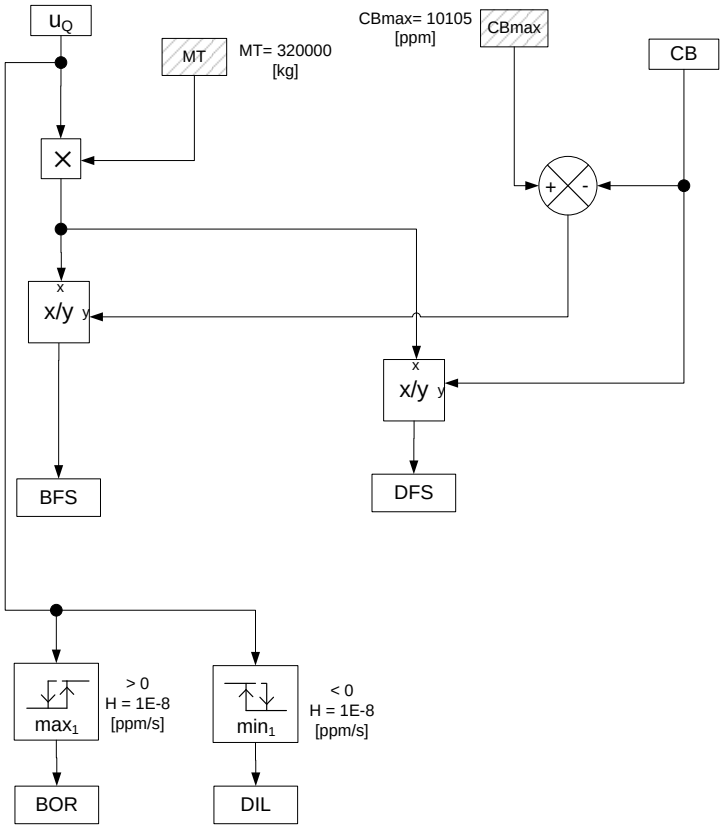


Figure 147: Diagramme de spécification fonctionnelle de la fonction F0010

Annexe O. Diagramme fonctionnel du mode manuel

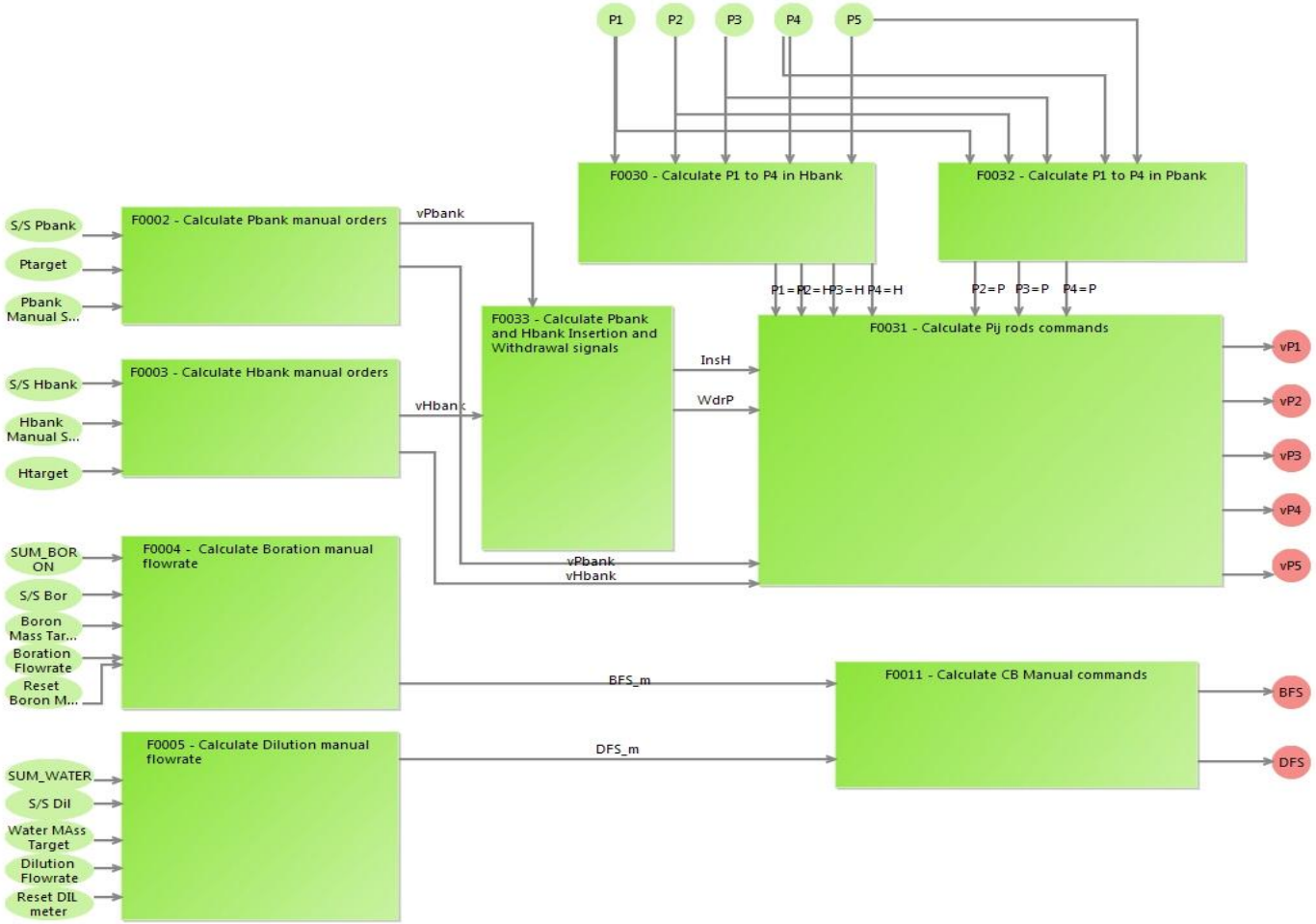


Figure 148: Diagramme fonctionnel du mode Manuel du Core Control

Annexe P. Diagrammes fonctionnels du Core Control

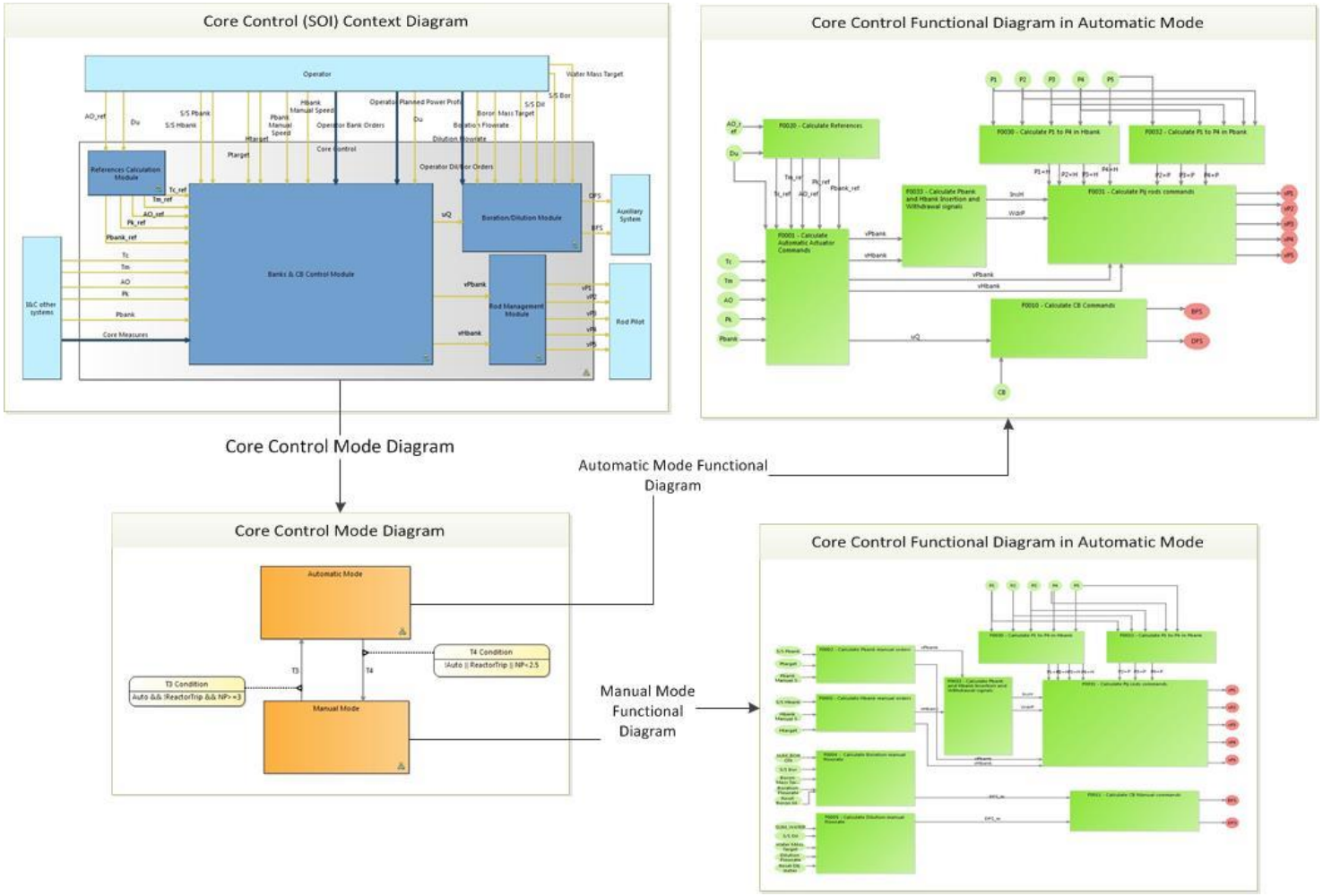


Figure 149: Les diagrammes fonctionnels du Core Control

Annexe Q. Résultats de simulation mode T – Rampe 100-60-100%PN à 5%PN/min

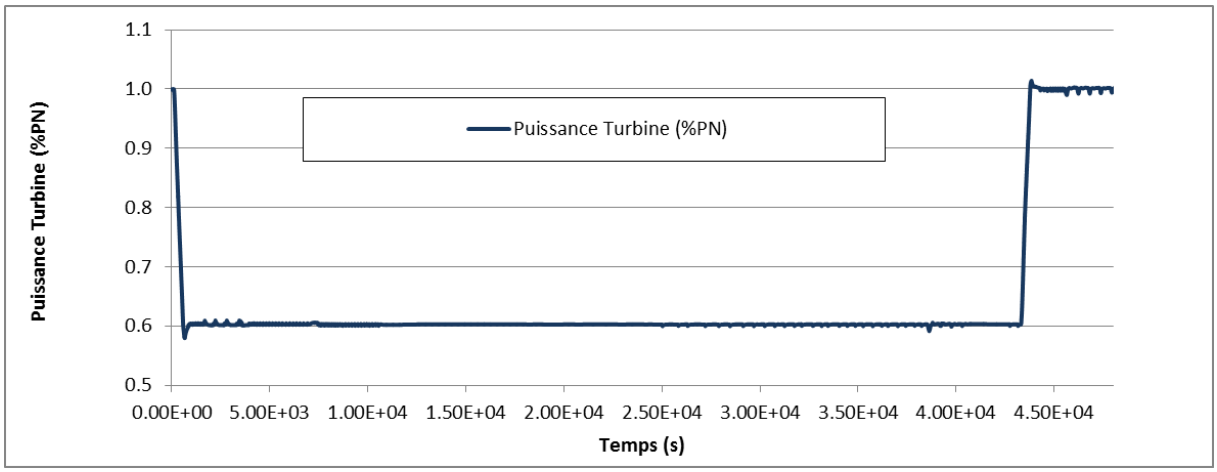


Figure 150: Simulation Mode T - Puissance turbine - Rampe 100-60-100%PN à 5%/min

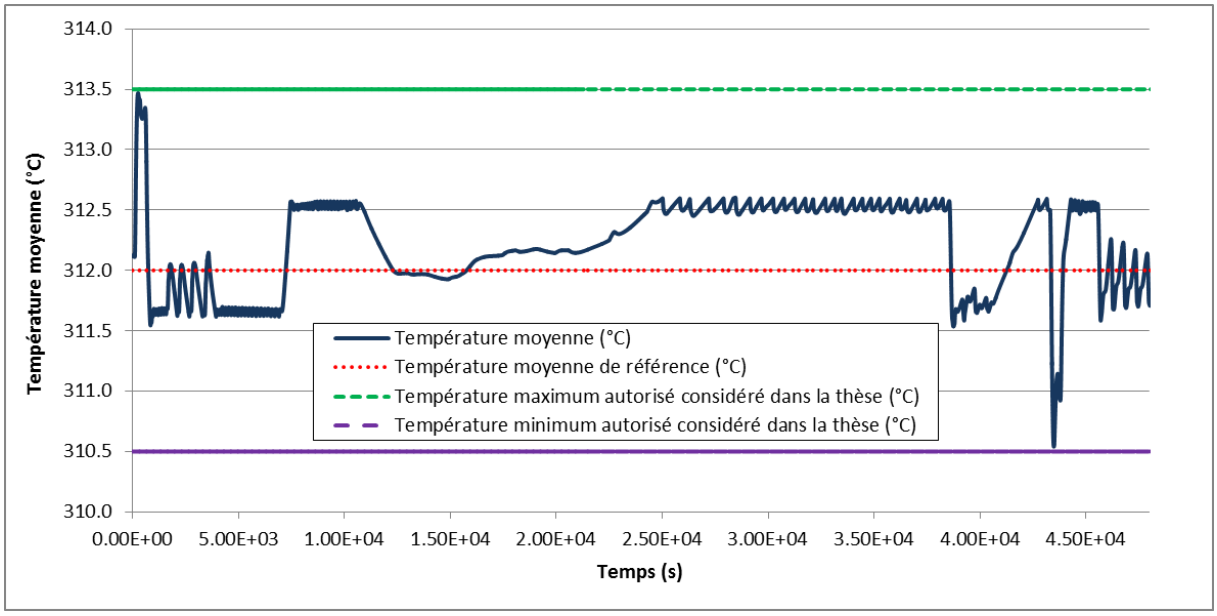


Figure 151: Simulation Mode T – Température moyenne - Rampe 100-60-100%PN à 5%/min

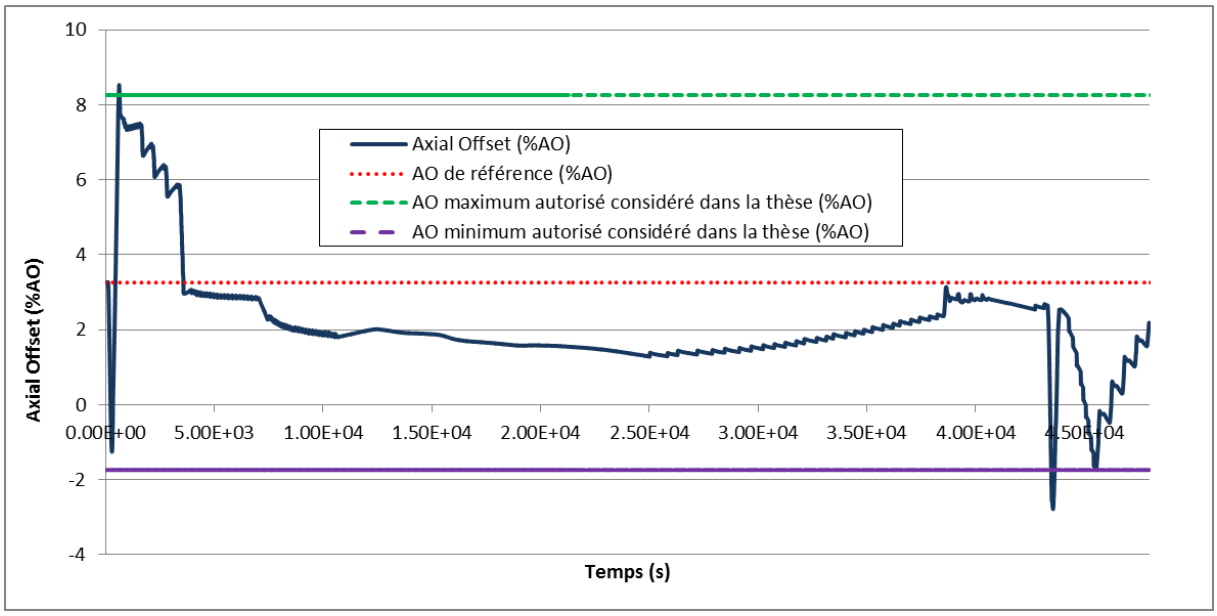


Figure 152: Simulation Mode T – Axial Offset - Rampe 100-60-100%PN à 5%/min

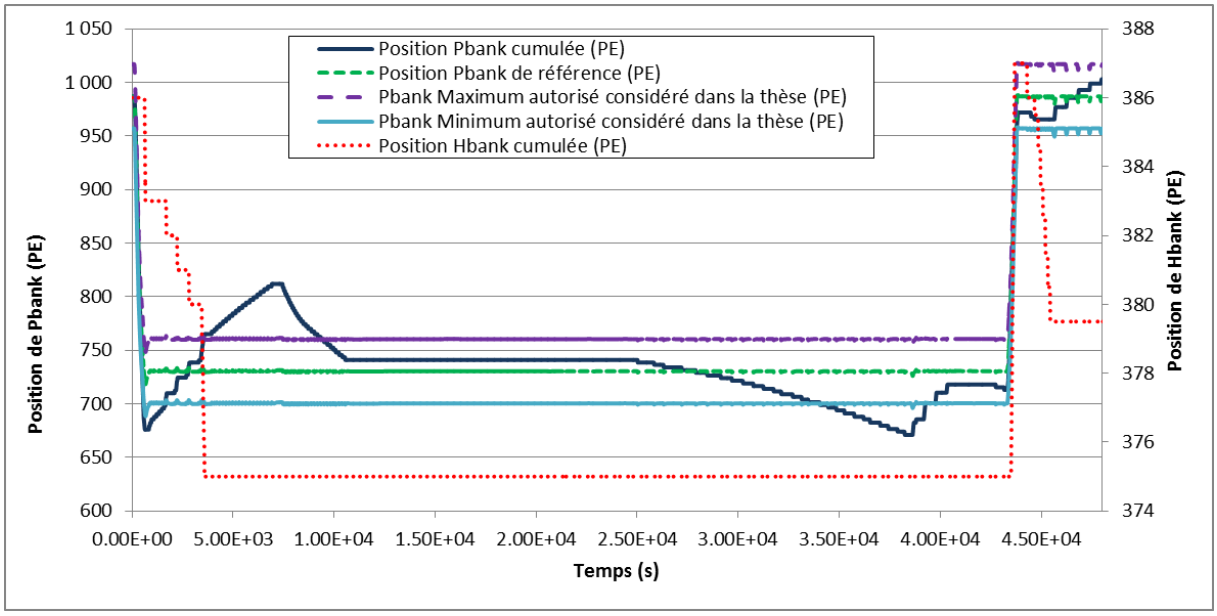


Figure 153: Simulation Mode T – Groupes de grappes Pbank et Hbank - Rampe 100-60-100%PN à 5%/min

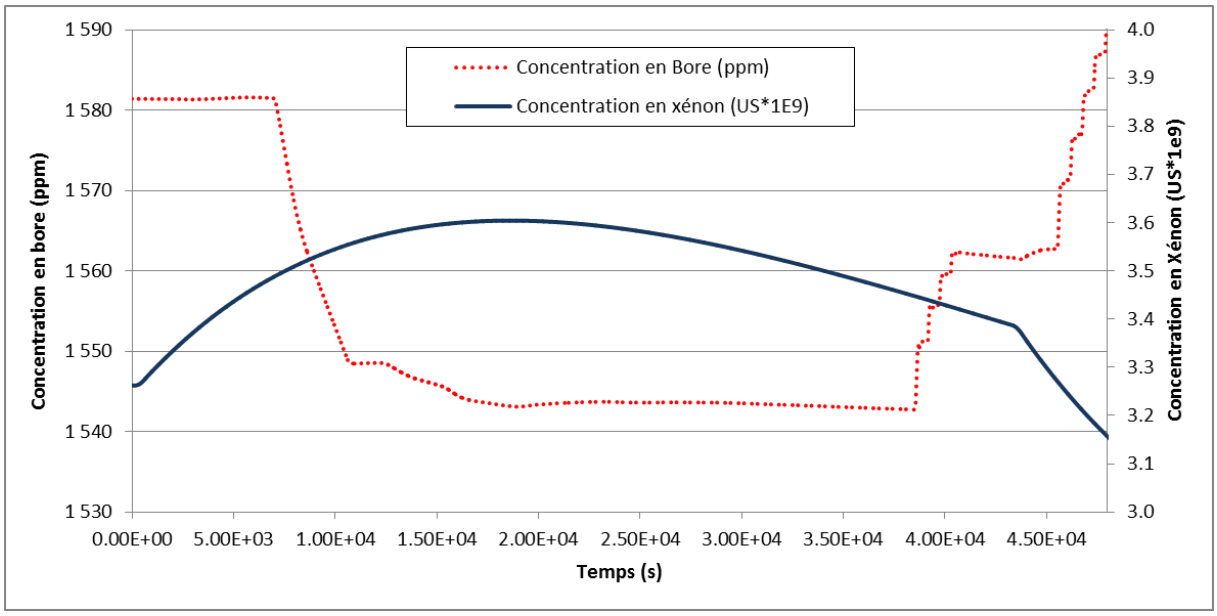


Figure 154: Simulation Mode T – Xénon et Concentration en Bore - Rampe 100-60-100%PN à 5%/min

Annexe R. Résultats de simulation mode T – Réglage de fréquence

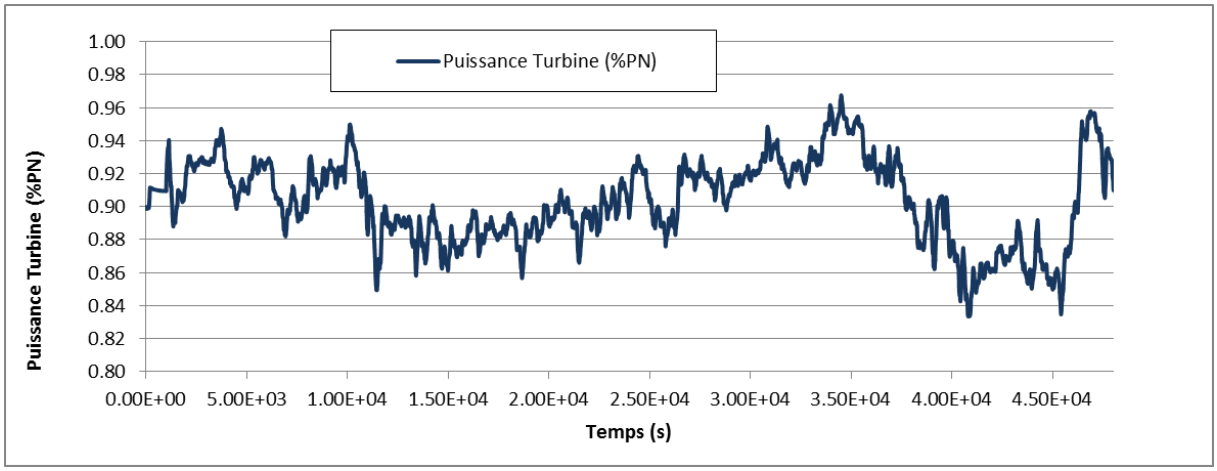


Figure 155: Simulation Mode T – Puissance turbine - Réglage de fréquence

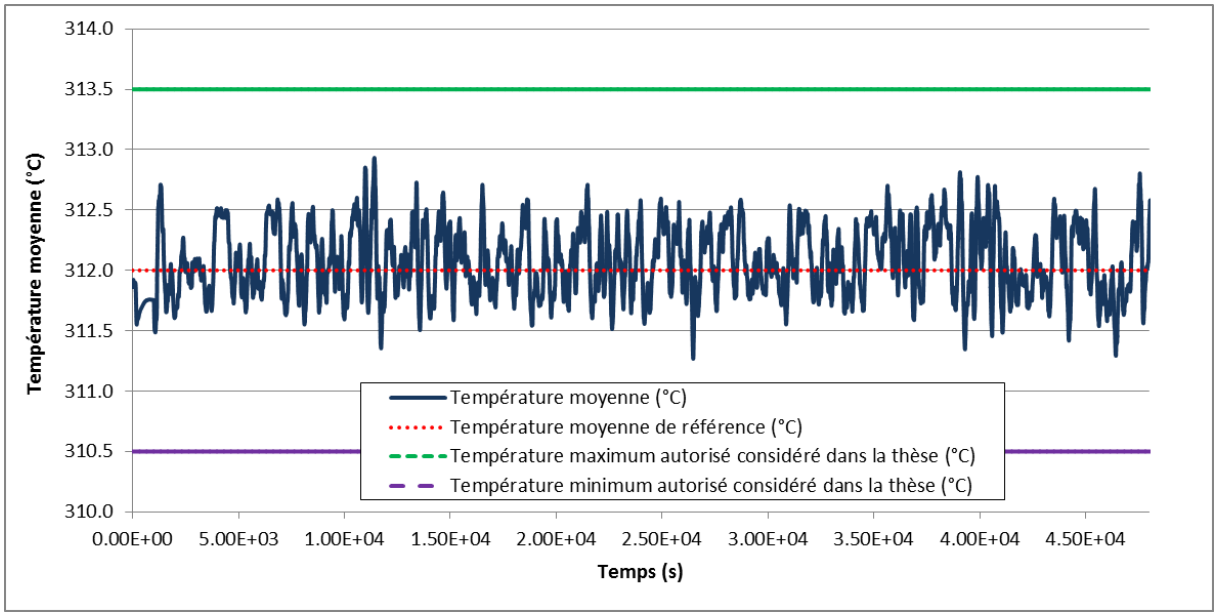


Figure 156: Simulation Mode T – Température moyenne - Réglage de fréquence

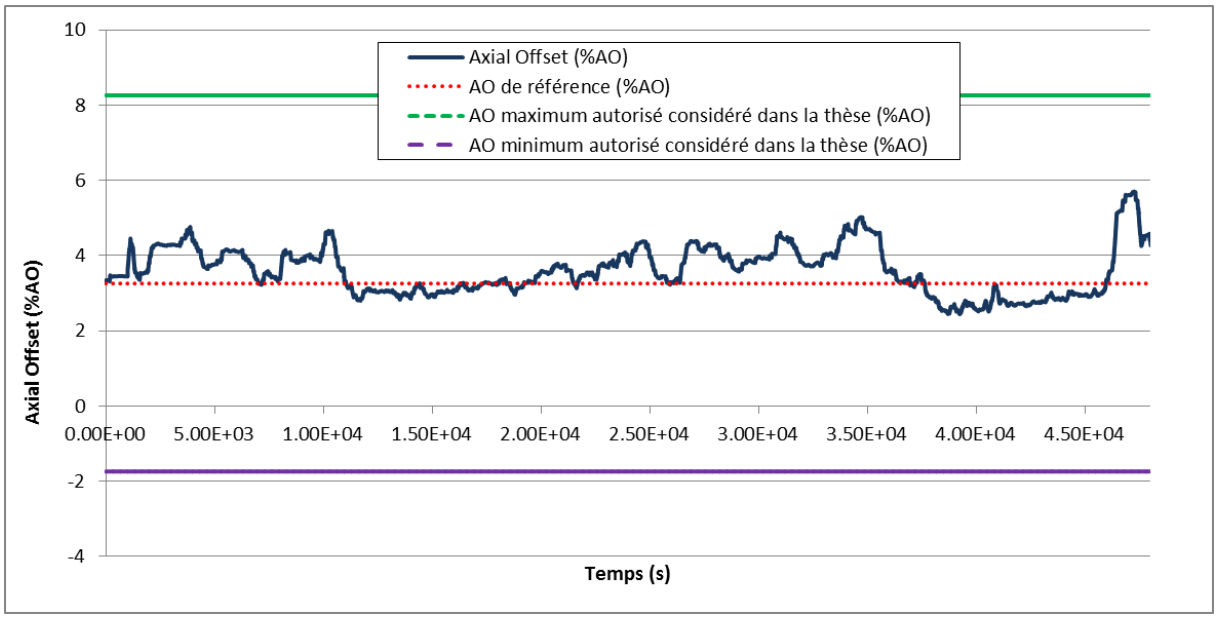


Figure 157: Simulation Mode T – Axial Offset - Réglage de fréquence

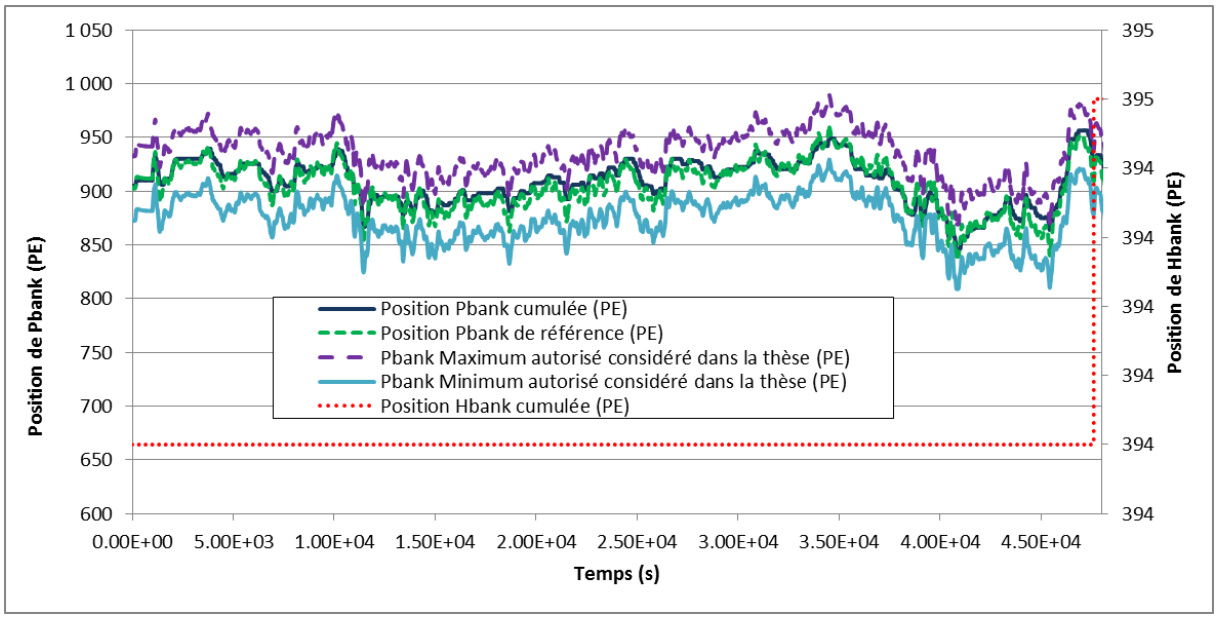


Figure 158: Simulation Mode T – Groupes de grappes Pbank et Hbank - Réglage de fréquence

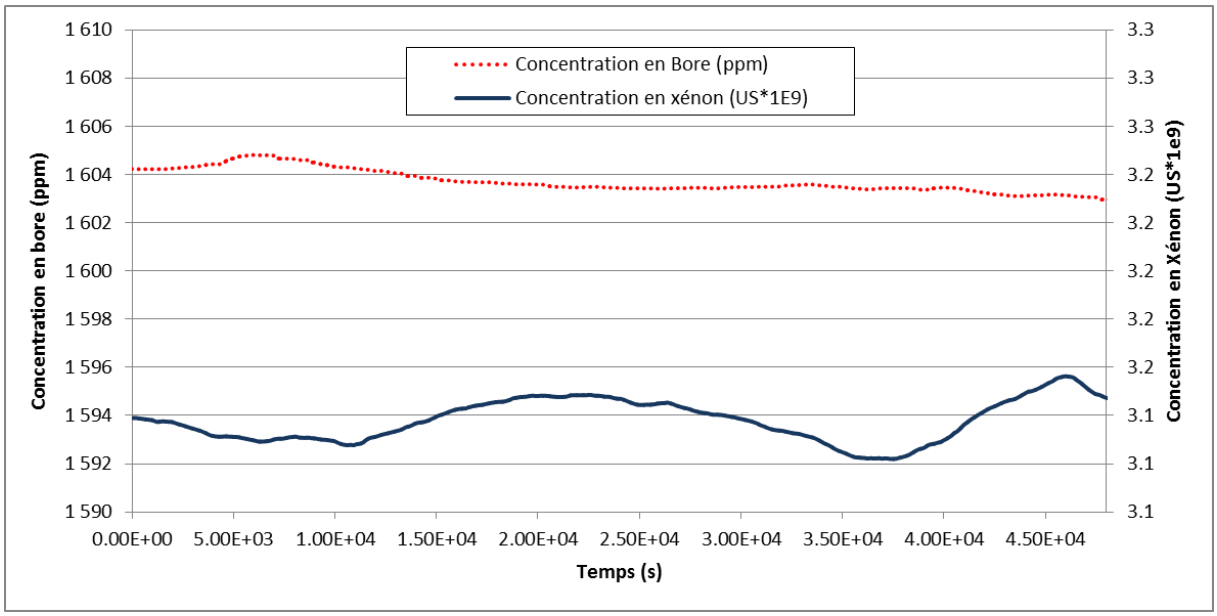


Figure 159: Simulation Mode T – Xénon et Concentration en Bore - Réglage de fréquence

Bibliographie

[Ablay, 2013] G. Ablay. A modeling and control approach to advanced nuclear power plants with gas turbines. *Energy Conversion and Management,* 76, pages 899–909, 2013.

[AFIS, 2012] *Découvrir et comprendre l'Ingénierie Système*. Collection AFIS sous la direction de Serge Fiorèse et Jean-Pierre Meinadier, CEPADUES Editions, ISBN : 97802036493.005.6 , Avril 2012.

[Amokrane *et al.,* 2015] N. Amokrane, V. Chapurlat, A. L. Courbis, T. Lambolais et M. Rahhou. Requirements Authoring and Verification for SMEs' Information Systems Engineering. *World Congress of the International Federation of Automatic Control*, 48 (3), pages 2238-2243, 2015.

[Amokrane, 2016] Nawel Amokrane. De l'Ingénierie des Besoins à l'Ingénierie des Exigences : Vers une Démarche Méthodologique d'Ingénierie de Systèmes Complexes, de Vérification et de Validation Appliquée à l'Informatisation des PME. Thèse de doctorat, Université Montpellier, 2016.

[Apkarian *et al.*, 2007a] P. Apkarian, D. Noll et A. Rondepierre. Nonsmooth optimization algorithm for mixed H_2/H_∞ synthesis. In Proc. of IEEE Conference on Decision and Control, New Orleans, December 2007.

[Apkarian *et al.,* 2007b] P. Apkarian, V. Bompart, and D. Nöll. Nonsmooth optimization techniques for structured controller design. *International Conference on High Performance Scientific Computing*, 2007.

[Apkarian *et al.*, 2008] P. Apkarian, D. Noll et P. Rondepierre. Mixed H_2/H_∞ Control via Nonsmooth Optimization. *SIAM J. Control and Optimization*, 47(3), pages 1516–1546, 2008.

[Apkarian et Gahinet, 1995] P. Apkarian et Gahinet. A convex characterization of gain-scheduled H= controllers. *IEEE Transactions on Automatic Control*, AC-40(5), pages 853-864, 1995.

[Apkarian et Noll, 2006a] P. Apkarian et D. Noll. Nonsmooth $H\infty$ Synthesis. *IEEE Transactions on Automatic Control*, 51(1), pages 71–86, 2006.

[Apkarian et Noll, 2006b] P. Apkarian et D. Noll. Nonsmooth optimization for multidisk $H\infty$ synthesis. *European Journal of Control*, 12(3), pages 229–244, 2006.

[Apkarian et Noll, 2006c] P. Apkarian et D. Noll. IQC analysis and synthesis via nonsmooth optimization. *Systems and Control Letters*, 55(12), pages 971–981, 2006.

[Apkarian et Noll, 2006d] P. Apkarian et D. Noll. Controller design via nonsmooth multidirectional search. *SIAM J. Control Optim.*, 44(6), pages1923–1949, 2006.

[Apkarian, 2013] P. Apkarian. Tuning Controllers Against Multiple Design Requirements. *American Control Conference*, , pages 3888–3893, 2013.

[Arab-Alibeik, 2003] H. Arab-Alibeik. Improved Temperature Control of a PWR Nuclear Reactor Using an LTQ/LTR Based Controller. *IEEE Transactions on Nuclear Science*, 50, pages 211-218, 2003.

[Astrom et Kumar, 2014] K. J. Astrom et P.R. Kumar. Control: A perspective. *Automatica*, 50(1), pages 3-43, 2014.

[Balas, 2002] G. J. Balas. Linear Parameter-Varying Control and its Application to Aerospace Systems. *International council of aeronautical sciences,* 2002.

[Bauer et Poncelet, 1974] D. C. Bauer, C. G. Poncelet. Practical Xenon Spatial Control. *Nuclear Technology,* 21, pages 165 – 189, 1974.

[Becker et Packard, 1991] G. Becker and A. Packard. Gain-Scheduled State-Feedback with Quadratic Stability for Uncertain Systems. Proc. *American Control Conf.*, pages 2702–2703, 1991.

[Bendotti *et al.*, 1994] P. Bendotti et B. Bodenheimer. Identification and H∞ Control Design for Pressurized Water Reactor. *IEEE Conference on Decision and Control*, 2, pages 1072-1077, 1994.

[Bodenheimer et al., 1995] B. Bodenheimer et P. Bendotti. Optimal Linear Parameter Varying Control Design for a Pressurized Water Reactor. *Proceedings of the 34th Conference on Decision & Control*, pages 182-187, 1995.

[Bompart *et al.*, 2007] V. Bompart, P. Apkarian, D. Nöll. Non-smooth techniques for stabilizing linear systems. *American Control Conference*, 2007.

[Bompart, 2007] V. Bompart. *Optimisation Non Lisse pour la Commande des Systèmes de l'Aéronautique*. Thèse de doctorat, Université Paul Sabatier, Toulouse, 2007.

[Boyd et Barrat, 1991] S.P. Boyd et C.H. Barrat. Linear Control Design, Limits of Performance. *Prentice Hall*, 1991.

[Boyd, 1994] S.P. Boyd. Linear Matrix Inequalities in System and Control Theory. *SIAM*, 1994.

[BPMN, 2011] Business Process Model And Notation Specification, version 2.0. Object Management Group, 2011.

[Bresch-Pietri *et al*., 2014] D. Bresch-Pietri, J. Chauvin et N. Petit. Prediction-Based Stabilization of Linear Systems Subject to Input-Dependent Input Delay of Integral-Type. IEEE Transactions on Automatic Control, 59(9), pages 2385-2399, 2014.

[Bruisma et Steinbuch, 1990] N. A. Bruisma et M. Steinbuch. A Fast Algorithm to Compute the H∞-Norm of a Transfer Function Matrix. *System Control Letters*, 14, pages 287-293, 1990.

[Burke *et al.*, 2002] J. V. Burke, A. S. Lewis et M. L. Overton. Two numerical methods for optimizing matrix stability. *Linear Algebra and its Applications*, pages 117–145, 2002.

[Byrd *et al.*, 1999] R. H. Byrd, E. M. Hribar et J. Nocedal. An Interior Point Algorithm for Large-Scale Nonlinear Programming. *SIAM Journal on Optimization*, 9(4), pages 877–900, 1999.

[CESAR, 2012] Cost-efficient methods and processes for safety relevant embedded systems. http://www.cesarproject.eu, 2012.

[Chapurlat et Daclin, 2013] V. Chapurlat et N. Daclin. Proposition of a guide for investigating, modeling and analyzing system operating modes: OMAG. *International Conference on Complex System Design and Management*, Paris, 2013.

[Chapurlat, 2007] V. Chapurlat. Vérification et validation de modèles de systèmes complexes: application à la Modélisation d'Entreprise. Rapport de HDR, Université de Montpellier II, 2007.

[Chen et Allgöwer, 1998] H. Chen et F. Allgöwer. A Quasi-Infinite Horizon Nonlinear Predictive Control Scheme with Guaranteed Stability. *Automatica*, 34(10), pages 1205-1218, 1998.

[Chevrel et Berrini, 2010] P. Chevrel et M. Berriri. Implementation des schemas de commande a retards distribues : Predicteur de Smith generalise versus predicteur d'etat. *Journal Européen des Systèmes Automatisés* , Lavoisier, 44(9-10), pages 1019-1045, 2010.

[Claveau, 2005] F. Claveau. *Contribution à l'analyse et la commande structurée des grands systèmes*. Thèse de doctorat, Ecole Centrale de Nantes, Université de Nantes, 2005.

[Clement, 2001] B. Clement. Synthèse Multi-objectif et Séquencement de Gains : Application au Pilotage d'un Lanceur Spatial. Thèse de doctorat, Université Paris XI Orsay, 2001.

[Coleman et Li, 1996] T. F. Coleman et Y. Li. An Interior, Trust Region Approach for Nonlinear Minimization Subject to Bounds. *SIAM Journal on Optimization*, 6, pages 418–445, 1996.

[Dong, 2011] Z. Dong. Nonlinear State-Feedback Dissipation Power Level Control for Nuclear Reactors. *IEEE Transactions on Nuclear Science*, 58, pages 241-257, 2011.

[Doyle *et al.*, 1989] J.C. Doyle, K. Glover, P.P. Khargonekar et B.A. Francis. State-space solutions to standard H_2 and H_∞ control problems. *IEEE Transactions on Automatic Control*, 34(8), 1989.

[Doyle, 1978] J. Doyle. Guaranteed margins for LQG regulators. *IEEE Transactions on Automatic Control*, 23(4), pages 756-757, 1978.

[Doyle, 1983] J.C. Doyle. Synthesis of robust controllers and filters. *IEEE Conference on Decision and Control*, 1983.

[Duc et Font, 1999] G. Duc, S. Font. Commande H∞ et µ-analyse : des outils pour la robustesse. *Hermès*, 1999.

[Dufour, 2000] P. Dufour. *Contribution à la Commande Prédictive des Systèmes à Paramètres Répartis Non-Linéaires*. Thèse de Doctorat, Université Claude Bernard Lyon 1, 2000.

[D'Andrea-Novel *et al.*, 1994] B. D'Andrea-Novel et M. Cohen De Lara. Commande Linéaire Des Systèmes Dynamiques. *Masson*, 1994.

[Edwards *et al.*, 1990] R. M. Edwards, K. Y. Lee et M. A. Schultz. State Feedback Assisted Classical Control: An Incremental Approach To Control Modernization Of Existing And Future Nuclear Reactors And Power Plants. *Nuclear Technologies*, 92(2), 1990.

[Edward *et al.*, 1992] R. M. Edwards, K.Y. Lee et A. Ray. Robust Optimal Control of Nuclear Reactors and Power Plants. *Nuclear Technology*, 98, pages 137-148, 1992.

[EIA632, 1999] EIA 632 (1999) – Processes for Engineering a System.

[Eliasi *et al.*, 2012] H. Eliasi, M. B. Menhaj et H. Davilu. Robust nonlinear model predictive control for a PWR nuclear power plant. *Progress in Nuclear Energy*, 54(1), pages 177–185, 2012.

[Emara *et al.,* 2013] H.M. Emara, A.A. Hanafy, M.M.Z. Abdelaal et S. Elaraby. Anovel, Robust Control Methodology Application to Nuclear Reactors. *Nuclear Science Engineering*, 174(1), pages 87– 95, 2013.

[Farfeleder *et al.*, 2011] S. Farfeleder, T. Moser, A. Krall, T. Stålhane, H. Zojer et C. Panis. DODT: Increasing Requirements Formalism using Domain Ontologies for Improved Embedded Systems Development. *DDECS*, pages 271-274. 2011.

[Fossard *et al.*, 1993] A. J. Fossard et D. Normand-Cyrot. Systèmes non linéaires. 3. Commande. *Masson*, 1993.

[Fuchs, 2006] N. E. Fuchs, K. Kaljurand et G. Schneider (2006). Attempto Controlled English Meets the Challenges of Knowledge Representation, Reasoning, Interoperability and User Interfaces. *FLAIRS Conference*, 12, pages 664-669, 2006.

[Gahinet et Apkarian, 2013] P. Gahinet et P. Apkarian. Automated Tuning of Gain-Scheduled Control Systems. *IEEE Conference on Decision and Control*, (3), pages 2740-2745, 2013.

[Greenfield et Short, 2003] J. Greenfield et K. Short. Software factories: assembling applications with patterns, models, frameworks and tools. *Proceeding OOPSLA*, pages 16-27, 2003.

[Grosdidier et al., 1988] P. Grosdidier, B. Froisy et M. Hamman. The IDCOM-M controller. In T.J. Mc Avoy, Y. Arkun and E. Zafiriou, editors, Proc. of the 1988 IFAC Workshop on Model Based Process Control, pages 31-36, 1988.

[Grossetête, 2007] A. Grossetête. Le pilotage de l'EPR : mode T. *Revue Générale du Nucléaire*, pages 37-41, 2007.

[Grossetête, 2008] A. Grossetête. EPR: high load variation performances with the "Tmode" core control". *International Congress on Advances in Nuclear Power Plants*, 42, pages 21-25, 2008.

[Grossetête, 2014] A. Grossetête. ATMEA1&EPR mode T Core Control Innovative Features for High Operating Flexibility. *Transactions of the American Nuclear Society*, pages 1095-1098, 2014.

[Gumussoy *et al.*, 2009] S. Gumussoy, D. Henrion, M. Millstone et M. L. Overton. Multiobjective Robust Control with HIFOO 2.0. *IFAC Symposium on RObust CONtrol Design*, 2009.

[Hardy, 1915] G. H. Hardy. The mean value of the modulus of an analytic function. *Proc. London Math. Soc.*, 14, pages 269-277, 1915.

[Hindi *et al.*, 1998] H.A. Hindi, B. Hassibi, S. Boyd. Multiobjective H_2/H_∞ - optimal control via finite dimensional Q-parametrization and linear matrix inequalities. *American Control Conference*, 5, pages 3244-3249 1998.

[Hull et al., 2002] E. Hull, K. Jackson and J. Dick. Requirements Engineering. First Edition, *Springer*, 2002.

[IEEE Std 1220, 2005] IEEE Std 1220 (2005) – IEEE Standard for Application and Management of the Systems Engineering Process.

[ISO15288, 2015] ISO/IEC/IEEE 15288 (2015) – Systems and software engineering – System life cycle processes.

[ISO 29148, 2011] ISO/IEC/IEEE 29148 (2011) – Systems and software engineering –Life cycle processes – Requirements engineering.

[ITU-T, 2008] User Requirements Notation (URN) – Language Definition Z.151, Telecommunication Standardization Sector of ITU, 2008.

[Keerthi et Gilbert, 1988] S. S. Keerthi et E. G. Gilbert. Optimal infinite-horizon feedback laws for a general class of constrained discrete-time systems : stability and moving horizon approximations. *J. Opt. Theory and Appl.*, 57(2), pages 265-293, 1988.

[Kerlin *et al.*, 1976] T. W. Kerlin, E.M. Katz, J. G. Thakkar et J. E. Strange. Theoretical and Experimental Dynamic Analysis of the H. B. Robinson Nuclear Plant. *Nuclear Technology*, 30(3), 1976.

[Lamsweerde et al.,1991] A. Van Lamsweerde, A. Dardenne, B. Delcourt et F. Dubisy. The KAOS project: Knowledge acquisition in automated specification of software. *Proceedings of the AAAI Spring Symposium Series*, Stanford University, 1991.

[Leite et Peres, 2004] V. Leite et P. Peres. Robust control through piecewise Lyapunov functions for discrete time-varying uncertain systems. *International Journal of Control*, 3(77), pages 230–238, 2004.

[Lemazurier *et al.*, 2016] L. Lemazurier et A. Grossetete. PHD Advanced Control/SE methodology - 2nd annual report. *Framatome Internal Publication*, ref: D02-DTIPC-F-17-0022, 2016.

[Lemazurier *et al.*, 2017a] L. Lemazurier, M. Yagoubi, Ph. Chevrel and A. Grossetête. Multi-Objective H_2/H_∞ Gain-Scheduled Nuclear Core Control Design. *The 20th World Congress of the International Federation of Automatic Control,* 50(1), pages 3256-3262, 2017.

[Lemazurier *et al.*, 2017b] L. Lemazurier, V. Chapurlat et A. Grossetête. An MBSE Approach to Pass from Requirements to Functional Architecture. *The 20th World Congress of the International Federation of Automatic Control,* 50(1), pages 7260-7265, 2017.

[Lemazurier *et al.*, 2018a] L. Lemazurier, Ph. Chevrel, M. Yagoubi and A. Grossetête. A Multi-Objective Nuclear Core Control Performing Hot and Cold Leg Temperature Control. *European Control Conference,* 2018.

[Lemazurier *et al.*, 2018b] L. Lemazurier, Ph. Chevrel, M. Yagoubi and A. Grossetête. An Advanced Multi-Objective Nuclear Core Control for PWR Reactors. *Control Engineering Practice,* 2018*, soumis.*

[Lewis, 2007] A. S. Lewis. Nonsmooth optimization and robust control. *Annual Reviews in Control*, 31, pages 167–177, 2007.

[Li, 2014] G. Li. Modeling and LQG/LTR control for power and axial power difference of load-follow PWR core. *Annals of Nuclear Energy*, 68, pages 193-203, 2014.

[Li, 2016] G. Li. Modeling and control of nuclear reactor cores for electricity generation: A review of advanced technologies. *Renewable and Suitable Energy Reviews*, 60, pages 116-128, 2016.

[Maciejowski, 1989] J. M. Maciejowski. Multivariable Feedback Design. *Addison-Wesley*, 1989.

[MDA, 2014] Model Driven Architecture (MDA) MDA Guide, version 2.0. Object Management Group, *http://www.omg.org/mda/*, document ormsc/2014-06 01, Juin 2014.

[Meinadier, 1998] J. P. Meinadier. Ingénierie et Intégration des systèmes. *Hermes*, Paris 1998.

[Michalska et Mayne, 1993] H. Michalska et D.Q. Mayne. Robust Receding Horizon Control Of Constrained Nonlinear Systems. *IEEE Trans. Automat. Contr.*, AC-38(11), pages1623-1633, 1993.

[Micouin, 2008] P. Micouin. Toward a Property Based Requirements Theory: System Requirements Structured as a Semilattice. *System Engineering*, 11(3), pages 235-245, 2008.

[Mourlevat, 2007] J.-L. Mourlevat. Evolution des modes de pilotage. *Revue Générale du Nucléaire*, 3, pages 29-36, 2007.

[MOF, 2015] The Meta Object Facility (MOF), version 2.5.1. Object Management Group, *http://www.omg.org/spec/MOF/2.5.1*, Juin 2015.

[Nastov *et al.*, 2015] B. Nastov, V. Chapurlat, C. Dony et F. Pfister. A verification approach from MDE applied to model based systems engineering: XeFFBD dynamic semantics. *Complex Systems Design and Management - Proceedings of the 5th International Conference on Complex Systems Design and Management*, pages 225–235, 2015.

[Nastov et al., 2016a] B. Nastov, V. Chapurlat, C. Dony et F. Pfister. A Tooled Approach for Designing Executable and Verifiable Modeling Languages. *INSIGHT Q. Mag. Int. Counc. Syst. Eng.*, 18(4), pages 31–33, 2016.

[Nastov *et al.*, 2016b] B. Nastov, V. Chapurlat, C. Dony et F. Pfister. Towards semantical DSMLs for complex or cyber-physical systems. *Proceedings of the 11th International Conference on Evaluation of Novel Software Approaches to Software Engineering,* pages115-123*,* 2016.

[Nastov, 2016] B. Nastov. Contribution à une méthode outillée pour la conception de langages de modélisation métier interopérables, analysables et prouvables pour l'Ingénierie Système basée. Thèse de doctorat (en anglais), Université de Montpellier, 2016.

[Na, Shin et Kim, 2003] M. G. Na, S.H. Shin et W.C. Kim. A model Predictive Controller for Nuclear Reactor Power. *Journal of the Korean Nuclear Society*. 35(5), pages 399 – 411, 2003.

[Oliveira *et al.*, 2005] R. Oliveira, V. Leite, M. D. Oliveira et P. Peres. An LMI characterization of polynomial parameter dependent Lyapunov functions for robust stability. *Proceeding of the 44th IEEE Conference on Decision and Control, and the European Control Conference*, pages 12–15, 2005.

[Oliveira et al., 2007] R. Oliveira, V. Montagner, P. Peres et P. Bliman. LMI relaxations for $H\infty$ control of time-varying polytopic systems by means of parameter-dependent quadratically stabilizing gains. *Proc. of 3rd IFAC Symposium on System, Structure and Control*, 2007.

[Oliveira et Peres, 2005] R. Oliveira et P. Peres. Stability of polytopes of matrices via affine parameter-dependent Lyapunov functions : Asymptotically exact LMI conditions. *Linear algebra and its applications*, 405, pages 209–228, 2005.

[Oliveira et Peres, 2007] R. Oliveira et P. Peres (). Parameter-dependent LMIs in robust analysis: Characterization of homogeneous polynomially parameter-dependent solutions via LMI relaxations. *IEEE Transactions on Automatic Control*, 52(7), pages 1334–1340, 2007.

[Overbeek, 2006] J.F. Overbeek. Meta Object Facility (MOF): investigation of the state of the art. *http://purl.utwente.nl/essays/57286*, 2006.

[Partington, 1991] J. R. Partington. Approximation of delay systems by fourier-laguerre series. *Automatica*, 27(3), pages 569-572, 1991.

[Peaucelle, 2000] D. Peaucelle. Formulation générique de problèmes en analyse et commande robuste par les fonctions de Lyapunov dépendant des paramètres. Thèse de doctorat, LAAS, Toulouse, 2000.

[Pénalva, 1997] J-M. Pénalva. *La modélisation par les systèmes en situations complexes*. Thèse de doctorat, Université Paris XI, Orsay, 1997.

[Pfister *et al.*, 2012] F. Pfister, V. Chapurlat, M. Huchard, C. Nebut et J.-L. Wippler. A proposed meta-model for formalizing Systems Engineering knowledge, based on functional architectural patterns. *International Journal Systems Engineering*, INCOSE Eds., 2012.

[Popoi, 1963] A.I. Propoi. Use of linear programming methods for synthesizing sampled-data automatic systems. *Automn. Remote Control*, 24(7) pages 837-844, 1963.

[Powell, 1978] M. J. D. Powell. A Fast Algorithm for Nonlinearly Constrained Optimization Calculations. *Numerical Analysis*, ed. G. A. Watson, *Lecture Notes in Mathematics, Springer-Verlag*, 630, 1978.

[Puyou, 2005] G. Puyou. Conception Multi-objectif de Lois de Pilotage pour un Avion de Transport Civil. Thèse de doctorat, ENSAE Toulouse, 2005.

[Ramos et Peres, 2002] D. Ramos et P. Peres. An LMI condition for the robust stability of uncertain continuous-time linear systems. *IEEE Transactions on Automatic Control*, 47, pages 675 – 678, 2002.

[Renier *et al.*, 2011] R. Renier, R. Chenouard. De Sysml à Modelica Aide à La Formalisation de Modeles de Simulation en Conception Preliminaire. *12ème Colloque National AIP PRIMECA*, 2011.

[Richalet *et al.*, 1976] J. Richalet, A. Rault, J. L. Testud et J. Papon. Algorithmic control of industrial processes. *In Proc. 4th IFAC Symposium on Identication and System Parameter Estimation*, pages 1119-1167, 1976.

[Roques, 2016] P. Roques. MBSE with the ARCADIA Method and the Capella Tool. *8th European Congress on Embedded Real Time Software and Systems*, 2016.

[Rugh et Shamma, 2000] W. Rugh et J. Shamma. Reasearch on gain Scheduling. *Automatica*, pages 1401 1425, 2000.

[Sadabadi et Peaucelle, 2016] M. Sadabadi et D. Peaucelle. From Static Output Feedback to Structured Robust Static Output Feedback: A Survey. Annual Reviews in Control, 42, pages 11-26, 2016.

[SAGACE, 1999] Méthode SAGACE, le systémographe, CEA 1999.

[SBVR, 2008] Semantics of Business Vocabulary and Business Rules (SBVR), version 1.0. Object Management Group, Janvier 2008.

[Scherer et Kose, 2012] C. Scherer et E. Kose. Gain-scheduled control synthesis using dynamic d-scales. *IEEE Trans. Aut. Control*, 57, pages 2219–2234, 2012.

[Scherer, 1995] C.W. Scherer. Multiobjective H2/H∞ control. *IEEE Transactions on Automatic Control*, 40(6), pages 1054–1062, 1995.

[Scherer, 2000] C.W. Scherer. An Efficient Solution to Multi-Objective Control Problem with LMI Objectives. *Systems and Control Letters*, 40, pages 43–57, 2000.

[Schneider *et al.*, 2012] F. Schneider, H. Naughton et B. Berenbach. A modeling language to support early lifecycle requirements modeling for systems engineering. *Procedia Computer Science*, 8, pages 201-206, 2012.

[SEBoK, 2016] BKCASE Editorial Board. The Guide to the Systems Engineering Body of Knowledge (SEBoK), version 1.6. R.D. Adcock (EIC). Hoboken, NJ: The Trustees of the Stevens Institute of Technology. http://www.sebokwiki.org/, 2016.

[SysML, 2012] *OMG Systems Modeling Language*. Object Management Group (2012). http://sysml.org/sysml-specifications/, 2012.

[Tahk et Speyer, 1987] Tahk et J. Speyer. Modeling of parameter variations and asymptotic LQG synthesis. *IEEE Trans. Automat. Contr.*, AC-32, pages 793-801, 1987.

[Tan et Packard, 2000] W. Tan et A. K. Packard. Quasi-LPV Modeling and LPV Control of a Generic Missile. *Conference on Decision and Control*, 2000.

[Troffino et Souza, 2001] A. Troffino et C. D. Souza. Biquadratic stability of uncertain linear systems. *IEEE Transactions on Automatic Control*, 46, pages 1303–1307, 2001.

[Troffino, 1999] A. Troffino. Parameter dependent Lyapunov functions for a class of uncertain linear systems : an LMI approach. *In proccedings of the 38th IEEE Conference on Decision and Control*, 1, pages 2341–2346, 1999.

[Ukai et Iwazumi, 1994] H. Ukai et Tetsuo Iwazumi. A New Approach to Control of Xenon Spatial Oscillation During Load Follow Operation via Robust Servo Systems. *IEEE Transactions On Nuclear Science*, 41(6), pages 2675-2685, 1994.

[Vanbiervliet *et al.*, 2009] J. Vanbiervliet, B. Vandereycken, W. Michiels, S. Vandewall et M. Diehl. The smoothed spectral abscissa for robust stability optimization. *Siam J. Optim. Society for Industrial and Applied Mathematics*, 20(1), pages 156–171, 2009.

[Vitech, 2011] *A primer to model Based System Engineering*, 2nd version. Vitech Corp, 2011.

[Vroemen et Jager, 1997] B. Vroemen et B. De Jager. Multiobjective control : An overview. *Proceedings of the IEEE CDC*, 1997.

[Wang *et al.*, 2014] L. Wang, Y. Liu et F Zhao. A Feasibility Study Of Load Follow Operation Without Boron Adjustment For CPR1000. *Nuclear Technology*, 186, pages 33-44, 2014.

[Williams *et al.*, 2014] S. Williams, R. Power et A. Third. How easy is it to learn a controlled natural language for building a knowledge base?. *Controlled Natural Language*, Springer International Publishing, pages 20-32, 2014.

[Winokur et Tepper, 1984] M. Winokur et L. Tepper. Extension of Load Follow Capability of a PWR Reactor by Optimal Control. *IEEE Transactions on Nuclear Science*, 31(2), pages 932–939, 1984.

[Zames, 1981] G. Zames. Feedback and optimal sensitivity: Model reference transformations, multiplicative seminorms, and approximate inverses. *IEEE Transactions on Automatic Control*, 26(2), pages 301-320, 1981.

[Zelm *et al.*, 1995] M. Zelm, F. B. Vernadat et K. Kosanke. The CIMOSA business Modelling Process. *Computers in industry*, 27(2), pages 123-142, 1995.

[Zhang *et al.*, 2015] Y. Zhang, P. Wang, F. Zhao, G. Li. Improvement of core control strategy for CPR1000: Load follow without boron adjustment. *Progress in Nuclear Energy*, 81, pages 98-103, 2015.

[Zhou et al., 1995] K. Zhou, J. C. Doyle et K. Glover. Robust and Optimal Control. Prentice-Hall, 1995.

UNIVERSITE BRETAGNE LOIRE

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Thèse de Doctorat

Lori LEMAZURIER

Conception d'un Système Avancé de Réacteur PWR Flexible par les Apports Conjoints de l'Ingénierie Système et de l'Automatique

Conception of an Advanced Flexible PWR Reactor System using Systems Engineering and Control Theories

Résumé

Devant l'augmentation de la part des énergies renouvelables en France, cette thèse propose d'étudier l'augmentation de la flexibilité des réacteurs à eau pressurisée en croisant deux disciplines pour, chacune, atteindre des objectifs complémentaires : l'Ingénierie Système (IS) et l'Automatique.

Dans le contexte de l'ingénierie de systèmes complexes et du Model Based Systems Engineering, ce travail propose dans un premier temps une méthode de conception se fondant sur les principes normatifs de l'IS et respectant les habitudes et les pratiques courantes en ingénierie de Framatome. Cette méthode a pour vocation de formaliser et assurer le passage des exigences aux architectures et d'améliorer les capacités de vérification des modèles développés lors de la conception. Elle s'organise autour de langages de modélisation interopérables, couvrant l'ensemble des processus promus par l'IS. La méthode proposée est appliquée sur le système dont les performances sont les plus limitantes dans le contexte de l'augmentation de flexibilité : le Core Control. Ce composant algorithmique du réacteur assure le contrôle des paramètres de fonctionnement du cœur : la température moyenne, la distribution axiale de puissance et la position des groupes de grappes.

La thèse propose ensuite des contributions techniques relevant du champ de l'Automatique. Il s'agit de concevoir un système de régulation répondant aux exigences issues de la formalisation IS évoquée ci-dessus. La solution proposée repose sur une stratégie de commande hiérarchisée, utilisant la complémentarité des approches dites de commande multi-objectif, de séquencement de gains et enfin de commande prédictive. Un modèle de réacteur nucléaire simplifié innovant est développé à des fins de conception du système de régulation et de simulations intermédiaires. Les résultats obtenus ont montré les capacités d'adaptation de la démarche proposée à des spécifications diverses. Les performances atteintes sont très encourageantes lorsque évaluées en simulation à partir d'un modèle réaliste et comparées à celles obtenues par les modes de pilotages classiques.

Mots clés

Contrôle-Commande de Réacteur Nucléaire, Commande Multi-Objectif, Régulateur à Gains Séquencés, Commande Prédictive, Ingénierie Système, MBSE, Vérification et Validation.

Abstract

In the event of increasing renewable energies in France, this thesis proposes to study the flexibility increase of pressurized water reactors (PWR) throughout two different engineering disciplines aiming at complementary objectives: Systems Engineering (SE) and Control theory.

In a first phase, within the frame of complex systems design and Model Based Systems Engineering, this work proposes a SE method based on SE standard principles and compliant with Framatome's practices and addressing the revealed issues. This SE contribution is twofold: formalize and ensure the path from requirements to system architectures and enhance the capabilities of models verification. The method revolves around interoperable modeling languages, covering the SE processes: from requirement engineering to system architecture design. The method is applied to the system, which performances are the most limiting in the context of flexibility increase: the Core Control. This algorithmic reactor component ensures the control of: the average coolant temperature, the axial offset and the rod bank position, three of the core main functioning parameters.

In order to provide a technical contribution relying on some advanced control methodologies. It consists in designing a control system meeting the requirements defined by the SE method application. The proposed solution is in a two-layer control strategy using the synergies of multi-objective control, gain-scheduling and predictive control strategies. A simplified innovative nuclear reactor model is employed to conceive the control algorithm, simulate and verify the developed models. The results obtained from this original approach showed the ability to adapt to various specifications. Compared to conventional core control modes, the simulation results showed very promising performances, while meeting the requirements, when evaluated on a realistic reactor model.

Keywords

Nuclear Core Control, Multi-Objective Control, Gain-Scheduling, Predictive Control, Systems Engineering, MBSE, Verification and Validation.