

INFORMATIQUE  
ET SYSTÈMES  
D'INFORMATION

Information - Commande - Communication

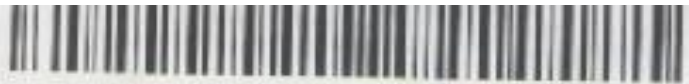
# Systemes temps réel 1

*techniques de description  
et de vérification*

*sous la direction de*  
Nicolas Navet

*hermes*

*Lavoisier*



2-004-275-1/1



# Systemes temps réel 1

*techniques de description et de vérification*

*sous la direction de*

**Nicolas Navet**

**Hermès  
Science**  
— PUBLICATIONS —

*Lavoisier*

# Table des matières

<b>Préface</b> . . . . .	19
Nicolas NAVET	
<b>Chapitre 1. Réseaux de Petri temporels : méthodes d'analyse et vérification avec TINA</b> . . . . .	25
Bernard BERTHOMIEU et François VERNADAT	
1.1. Introduction . . . . .	25
1.2. Les réseaux de Petri temporels . . . . .	26
1.2.1. Réseaux temporels . . . . .	26
1.2.2. États et relation d'accessibilité. . . . .	27
1.2.3. Illustration . . . . .	28
1.2.4. Quelques théorèmes généraux . . . . .	29
1.3. Graphes de classes préservant marquages et propriétés <i>LTL</i> . . . . .	30
1.3.1. Classes d'états . . . . .	30
1.3.2. Illustration . . . . .	31
1.3.3. Vérification à la volée du caractère borné. . . . .	32
1.3.4. Variantes . . . . .	34
1.3.4.1. Sensibilisations multiples . . . . .	34
1.3.4.2. Préservation des seuls marquages . . . . .	34
1.4. Graphes de classes préservant états et propriétés <i>LTL</i> . . . . .	35
1.4.1. Domaines d'horloges . . . . .	35
1.4.2. Construction du <i>SSCG</i> . . . . .	36
1.4.3. Variantes . . . . .	37
1.5. Graphes de classes préservant états et propriétés <i>CTL</i> . . . . .	37
1.6. Calculs d'échéanciers . . . . .	40
1.6.1. Systèmes d'échéanciers . . . . .	40
1.6.2. Délais (dates relatives) et dates (absolues) . . . . .	41
1.6.3. Illustration . . . . .	41

1.7. Mise en œuvre, l'environnement <i>Tina</i> . . . . .	42
1.8. La vérification de formules <i>SE-LTL</i> dans <i>Tina</i> . . . . .	43
1.8.1. La logique temporelle <i>SE-LTL</i> . . . . .	44
1.8.2. Préservation des propriétés <i>LTL</i> par les constructions de <i>Tina</i> . . . . .	46
1.8.3. <i>selt</i> : le vérificateur <i>SE-LTL</i> de <i>Tina</i> . . . . .	46
1.8.3.1. Technique de vérification . . . . .	46
1.8.3.2. La logique de <i>selt</i> . . . . .	47
1.9. Quelques exemples d'utilisation de <i>selt</i> . . . . .	49
1.9.1. John et Fred . . . . .	49
1.9.1.1. Énoncé . . . . .	49
1.9.1.2. Les contraintes temporelles figurant dans ce scénario sont-elles consistantes ? . . . . .	49
1.9.1.3. Est-il possible que Fred ait pris le bus et que John ait utilisé le covoiturage ? . . . . .	51
1.9.1.4. A quels horaires Fred a pu partir ? . . . . .	51
1.9.2. Le protocole du bit alterné . . . . .	52
1.10. Conclusion . . . . .	54
1.11. Bibliographie . . . . .	55

## Chapitre 2. Combinaison entre vérification et test pour la validation de systèmes réactifs . . . . .

59

Camille CONSTANT, Thierry JÉRON, Hervé MARCHAND et Vlad RUSU

2.1. Introduction . . . . .	59
2.2. Le modèle des IOSTS . . . . .	62
2.2.1. Syntaxe des IOSTS . . . . .	62
2.2.2. Sémantique des IOSTS . . . . .	63
2.2.3. Produit parallèle . . . . .	65
2.2.4. Blocage et IOSTS suspendu . . . . .	66
2.2.5. IOSTS déterministe et déterminisation . . . . .	67
2.3. Vérification et test de conformité pour des IOSTS . . . . .	67
2.3.1. Vérification . . . . .	67
2.3.1.1. Vérification de propriétés de sûreté . . . . .	68
2.3.1.2. Vérification de propriétés d'accessibilité . . . . .	70
2.3.1.3. Combinaisons d'observateurs . . . . .	71
2.3.2. Test de conformité . . . . .	72
2.4. Génération de tests et conformité . . . . .	73
2.5. Sélection des tests . . . . .	77
2.6. Étude de cas . . . . .	80
2.6.1. Spécification . . . . .	80
2.6.2. Propriétés . . . . .	80
2.6.3. Génération du cas de test . . . . .	82
2.6.3.1. Suspension . . . . .	82

2.6.3.2. Déterminisation et testeur canonique . . . . .	82
2.6.3.3. Sélection . . . . .	82
2.7. Conclusion et travaux connexes . . . . .	83
2.8. Bibliographie . . . . .	87
<b>Chapitre 3. <i>Model checking</i> : éléments de base . . . . .</b>	<b>89</b>
Stephan MERZ	
3.1. Introduction . . . . .	89
3.2. Exemple : contrôle d'un ascenseur . . . . .	90
3.3. Systèmes de transition et leurs exécutions . . . . .	93
3.3.1. Modèles sémantiques de systèmes . . . . .	93
3.3.2. Vérification d'invariants . . . . .	94
3.4. Logiques temporelles . . . . .	96
3.4.1. La logique temporelle linéaire . . . . .	96
3.4.2. Les logiques temporelles arborescentes . . . . .	98
3.4.3. $\omega$ -automates . . . . .	101
3.4.4. Automates et <i>PTL</i> . . . . .	103
3.5. Algorithmes de <i>model checking</i> . . . . .	106
3.5.1. Algorithme local pour <i>PTL</i> . . . . .	106
3.5.2. Algorithme global pour <i>CTL</i> . . . . .	108
3.5.3. Algorithmes symboliques . . . . .	110
3.6. Quelques sujets actuels . . . . .	113
3.7. Bibliographie . . . . .	115
<b>Chapitre 4. Vérification par automates temporisés . . . . .</b>	<b>121</b>
Patricia BOUYER et François LAROUSSINIE	
4.1. Introduction . . . . .	121
4.2. Automates temporisés, exemples et sémantique . . . . .	122
4.2.1. Quelques notations . . . . .	122
4.2.2. Automates temporisés, syntaxe et sémantique . . . . .	123
4.2.3. Composition parallèle . . . . .	124
4.3. Décidabilité de l'accessibilité . . . . .	125
4.4. Autres problèmes de vérification . . . . .	128
4.4.1. Vérification et langages temporisés . . . . .	128
4.4.2. Logique de temps arborescent . . . . .	129
4.4.3. Logique de temps linéaire . . . . .	130
4.4.4. Logiques modales avec points fixes . . . . .	131
4.4.5. Automates de test . . . . .	131
4.4.6. Équivalences comportementales . . . . .	131
4.5. Quelques extensions des automates temporisés . . . . .	132

4.5.1. Les contraintes « diagonales »	132
4.5.2. Les contraintes additives	133
4.5.3. Les actions internes	135
4.5.4. Les mises à jour	135
4.5.5. Les automates hybrides linéaires	137
4.6. Quelques sous-classes des automates temporisés	138
4.6.1. Automates <i>event-recording</i>	138
4.6.2. Automates à une ou deux horloges	138
4.6.3. Modèles à temps discret	140
4.7. Algorithmique de la vérification	140
4.7.1. Les zones, une représentation symbolique pour les automates temporisés	141
4.7.2. Analyse en arrière des automates temporisés	141
4.7.3. Analyse en avant des automates temporisés	142
4.7.4. Les DBM, une structure de donnée pour les systèmes temporisés	144
4.8. L'outil Uppaal	145
4.9. Bibliographie	146

## Chapitre 5. Modélisation et analyse de systèmes asynchrones avec CADP . . . . 151

Radu MATEESCU

5.1. Introduction	151
5.2. La boîte à outils CADP	152
5.2.1. Le langage LOTOS	153
5.2.2. Systèmes de transitions étiquetées	153
5.2.3. Quelques outils de vérification	154
5.3. Modélisation d'une unité de perçage	158
5.3.1. Architecture	161
5.3.2. Dispositifs physiques et contrôleurs locaux	163
5.3.2.1. Table rotative	163
5.3.2.2. Verrou	164
5.3.2.3. Perceuse	165
5.3.2.4. Testeur	165
5.3.3. Contrôleur principal – Version séquentielle	166
5.3.4. Contrôleur principal – Version parallèle	168
5.3.5. Environnement	169
5.4. Analyse du fonctionnement de l'unité de perçage	170
5.4.1. Vérification par équivalences	170
5.4.2. Vérification par logiques temporelles	173
5.5. Conclusion et travaux futurs	175
5.6. Bibliographie	177

<b>Chapitre 6. Vérification de programmes synchrones avec Lustre/Lesar</b> . . .	181
Pascal RAYMOND	
6.1. Approche synchrone. . . . .	182
6.1.1. Le domaine des systèmes réactifs . . . . .	182
6.1.2. L'approche synchrone . . . . .	182
6.1.3. Les langages synchrones . . . . .	183
6.2. Le langage Lustre. . . . .	183
6.2.1. Principes . . . . .	183
6.2.2. Exemple du compteur de balises . . . . .	184
6.3. Vérification de programme . . . . .	185
6.3.1. Notion de propriété temporelle . . . . .	185
6.3.2. Sûreté et vivacité . . . . .	185
6.3.3. Exemple du compteur de balises . . . . .	186
6.3.4. Machine à mémoire . . . . .	186
6.3.5. Automate explicite . . . . .	186
6.3.6. Principe du <i>model-checking</i> . . . . .	187
6.3.7. Exemple d'abstraction . . . . .	187
6.3.8. Abstraction conservative et propriétés de sûreté . . . . .	188
6.4. Expression des propriétés . . . . .	189
6.4.1. Le schéma général du <i>model-checking</i> . . . . .	189
6.4.2. <i>Model-checking</i> des programmes synchrones . . . . .	190
6.4.3. Observateurs . . . . .	190
6.4.4. Exemples. . . . .	190
6.4.5. Hypothèses . . . . .	191
6.4.6. <i>Model-checking</i> des programmes synchrones . . . . .	191
6.5. Algorithmique . . . . .	192
6.5.1. Automate booléen . . . . .	192
6.5.2. Automate explicite associé . . . . .	192
6.5.3. Fonctions « pre » et « post » . . . . .	193
6.5.4. Les états remarquables . . . . .	193
6.5.5. Principe de l'exploration . . . . .	193
6.6. Algorithme énumératif . . . . .	194
6.7. Méthodes symboliques et graphes binaires de décision . . . . .	195
6.7.1. Notations. . . . .	195
6.7.2. Manipulation de prédicats . . . . .	196
6.7.3. Représentation des prédicats . . . . .	196
6.7.3.1. Décomposition de Shannon . . . . .	196
6.7.3.2. Graphe binaire de décision . . . . .	198
6.7.4. Interface typique d'une librairie de BDDs . . . . .	198
6.7.5. Implémentation de BDDs . . . . .	199
6.7.6. Opérations sur les BDDs . . . . .	199
6.7.6.1. Négation . . . . .	199

6.7.6.2. Opérations binaires . . . . .	199
6.7.6.3. Cofacteurs et quantificateurs . . . . .	200
6.7.7. Notes sur la complexité des BDDs . . . . .	201
6.7.8. Graphes de décision typés . . . . .	202
6.7.8.1. Fonctions positives . . . . .	202
6.7.8.2. TDG . . . . .	203
6.7.8.3. Implémentation des TDGs . . . . .	203
6.7.8.4. Gain engendré par les TDGs . . . . .	203
6.7.9. Espace d'intérêt et cofacteur généralisé . . . . .	204
6.7.9.1. Opérations « sachant que » . . . . .	204
6.7.9.2. Cofacteur généralisé . . . . .	204
6.7.9.3. Restriction . . . . .	205
6.7.9.4. Propriétés algébriques du cofacteur généralisé . . . . .	205
6.8. Exploration symbolique en avant . . . . .	206
6.8.1. Schéma général . . . . .	206
6.8.2. Implémentation détaillée . . . . .	206
6.8.3. Calcul symbolique d'image . . . . .	207
6.8.4. Calcul optimisé d'image . . . . .	209
6.8.4.1. Principes . . . . .	209
6.8.4.2. Image universelle . . . . .	209
6.8.4.3. Cas d'une seule fonction de transition . . . . .	210
6.8.4.4. Décomposition de Shannon de l'image . . . . .	210
6.9. Exploration symbolique en arrière . . . . .	211
6.9.1. Schéma général . . . . .	211
6.9.2. Calcul d'image inverse . . . . .	212
6.9.3. Comparaisons entre les deux méthodes . . . . .	213
6.10. Conclusion et travaux en relation . . . . .	213
6.11. Démonstrations . . . . .	214
6.11.1. Lemme : propriété fondamentale du co-facteur . . . . .	214
6.12. Bibliographie . . . . .	215

## Chapitre 7. Lucid Sychrone, un langage de programmation des systèmes réactifs . . . . .

Paul CASPI, Grégoire HAMON et Marc POUZET

217

7.1. Introduction . . . . .	217
7.1.1. Langages pour les systèmes réactifs . . . . .	217
7.1.1.1. Les langages synchrones . . . . .	218
7.1.1.2. Développement par modèle . . . . .	220
7.1.1.3. Des besoins convergents . . . . .	221
7.1.2. Lucid Sychrone . . . . .	221
7.2. Lucid Sychrone . . . . .	223

7.2.1. Un Langage à la ML sur des flots . . . . .	224
7.2.1.1. Les flots comme objets de base . . . . .	224
7.2.1.2. Opérations temporelles : décalage et initialisation . . . . .	224
7.2.2. Fonctions de flots . . . . .	225
7.2.3. Systèmes multi-horloges . . . . .	227
7.2.3.1. L'opérateur d'échantillonnage when . . . . .	228
7.2.3.2. L'opération de combinaison merge . . . . .	229
7.2.3.3. Sur-échantillonnage . . . . .	230
7.2.3.4. Contraintes d'horloges et synchronisme . . . . .	232
7.2.4. Valeurs statiques . . . . .	233
7.2.5. Types de données et filtrage . . . . .	234
7.2.6. Une construction pour partager la mémoire . . . . .	235
7.2.7. Signaux . . . . .	236
7.2.7.1. Les signaux vus comme des abstractions d'horloges . . . . .	237
7.2.7.2. Tester la présence et filtrer des signaux . . . . .	237
7.2.8. Machines à états et systèmes mixtes . . . . .	239
7.2.8.1. Prémption faible et prémption forte . . . . .	239
7.2.8.2. ABRO et la réinitialisation modulaire . . . . .	240
7.2.8.3. Définitions locales à un état . . . . .	241
7.2.8.4. Communication entre états et mémoires partagées . . . . .	242
7.2.8.5. Actions par défaut et le rôle particulier de l'état initial . . . . .	242
7.2.8.6. Réinitialiser ou prolonger un état . . . . .	243
7.2.9. Machines d'états paramétrées . . . . .	244
7.2.10. Machines d'états et signaux . . . . .	245
7.2.11. Traits d'ordre supérieur . . . . .	246
7.2.12. Deux exemples classiques . . . . .	248
7.2.12.1. Le pendule inversé . . . . .	248
7.2.12.2. Un contrôleur de chaudière . . . . .	250
7.3. Discussion . . . . .	252
7.3.1. Programmation fonctionnelle pour le réactif ou la description de circuits . . . . .	253
7.3.2. Types, horloges et analyses statiques . . . . .	253
7.3.3. Définition de systèmes mixtes . . . . .	254
7.4. Conclusion . . . . .	255
7.5. Remerciements . . . . .	256
7.6. Bibliographie . . . . .	256
<b>Chapitre 8. Vérification de systèmes probabilisés : méthodes et outils . . . . .</b>	<b>261</b>
Serge HADDAD et Patrice MOREAUX	
8.1. Introduction . . . . .	261
8.2. Évaluation de performance de modèles markoviens . . . . .	262

8.2.1. Un modèle stochastique de systèmes à événements discrets . . . . .	262
8.2.2. Chaînes de Markov à temps discret . . . . .	264
8.2.2.1. Présentation . . . . .	264
8.2.2.2. Comportement transitoire et stationnaire d'une DTMC . . . . .	265
8.2.3. Chaînes de Markov à temps continu . . . . .	267
8.2.3.1. Présentation . . . . .	267
8.2.3.2. Comportement transitoire et stationnaire d'une CTMC . . . . .	267
8.3. Modèles stochastiques de haut niveau . . . . .	269
8.3.1. Les réseaux de Petri à lois générales . . . . .	269
8.3.1.1. Politique de choix . . . . .	270
8.3.1.2. Politique de service . . . . .	271
8.3.1.3. Politique de mémoire . . . . .	271
8.3.2. Les réseaux de Petri à lois exponentielles . . . . .	272
8.3.3. Indices de performance des réseaux de Petri . . . . .	272
8.3.4. Panorama des modèles et des méthodes d'évaluation de performance . . . . .	273
8.3.5. L'outil GreatSPN . . . . .	274
8.3.5.1. Les modèles supportés . . . . .	275
8.3.5.2. Analyse qualitative des réseaux de Petri . . . . .	275
8.3.5.3. Analyse de performance des réseaux de Petri stochastiques . . . . .	275
8.3.5.4. Architecture logicielle . . . . .	275
8.4. Vérification probabiliste des chaînes de Markov . . . . .	275
8.4.1. Limites des indices de performance standard . . . . .	276
8.4.2. Une logique temporelle pour les chaînes de Markov . . . . .	276
8.4.3. Algorithme de vérification . . . . .	278
8.4.4. Panorama de la vérification probabiliste des chaînes de Markov . . . . .	279
8.4.5. L'outil ETMCC . . . . .	280
8.4.5.1. Langage des modèles de systèmes . . . . .	281
8.4.5.2. Langage des propriétés . . . . .	281
8.4.5.3. Résultats calculés . . . . .	281
8.4.5.4. Architecture logicielle . . . . .	281
8.5. Les processus de décision markoviens . . . . .	281
8.5.1. Présentation des processus de décision markoviens . . . . .	281
8.5.2. Une logique temporelle pour les processus de décision markoviens . . . . .	282
8.5.3. Algorithme de vérification . . . . .	283
8.5.4. Panorama de la vérification des processus de décision markoviens . . . . .	287
8.5.5. L'outil PRISM . . . . .	288
8.5.5.1. Langage des modèles de systèmes . . . . .	288
8.5.5.2. Langage des propriétés . . . . .	288
8.5.5.3. Résultats calculés . . . . .	288
8.5.5.4. Architecture logicielle . . . . .	288
8.6. Bibliographie . . . . .	289

<b>Chapitre 9. La boîte à outils IF pour la modélisation et la vérification de systèmes temps réel</b> .....	293
Marius BOZGA, Susanne GRAF, Laurent MOUNIER et Iulian OBER	
9.1. Motivation .....	294
9.2. Architecture globale de la boîte à outils .....	296
9.2.1. Le niveau langage utilisateur .....	296
9.2.2. Le niveau de description intermédiaire (IF) .....	297
9.2.3. Les niveaux plate-forme d'exploration et graphe d'états .....	298
9.3. La notation IF .....	298
9.3.1. Modélisation des aspects fonctionnels .....	299
9.3.2. Modélisation des aspects non fonctionnels .....	301
9.3.3. Expression de propriétés à l'aide d'observateurs .....	303
9.4. Les outils .....	304
9.4.1. Les composants noyaux .....	304
9.4.2. Les outils d'analyse statique .....	308
9.4.3. Les outils de validation .....	309
9.4.4. L'outil de transformation depuis UML vers IF .....	309
9.4.4.1. La modélisation avec UML .....	310
9.4.4.2. Les principes de transformation .....	310
9.5. Etude de cas : le programme de vol d'Ariane 5. ....	312
9.5.1. Aperçu du programme de vol d'Ariane 5. ....	312
9.5.2. Vérification de propriétés fonctionnelles .....	314
9.5.3. Vérification de propriétés non fonctionnelles .....	318
9.5.4. Vérification modulaire et utilisation d'abstractions .....	320
9.6. Discussion .....	320
9.7. Bibliographie .....	322
<b>Chapitre 10. Description d'architectures pour le temps réel : l'approche AADL</b> .....	327
Anne-Marie DÉPLANCHE et Sébastien FAUCOU	
10.1. Introduction .....	327
10.2. Principes généraux des langages de description d'architecture .....	330
10.3. ADL et systèmes temps réel .....	331
10.3.1. Analyse des besoins .....	331
10.3.2. Vues architecturales .....	334
10.3.3. Tour d'horizon de l'existant .....	336
10.4. Le langage AADL .....	337
10.4.1. Le langage en bref .....	338
10.4.2. Les outils associés .....	340
10.4.2.1. Outils d'analyse offerts par OSATE .....	341
10.4.3. Cahier des charges de l'exemple .....	342

10.4.4. Description d'une architecture . . . . .	342
10.4.4.1. Organisation du modèle . . . . .	342
10.4.4.2. Éléments de la plate-forme . . . . .	343
10.4.4.3. Éléments de l'architecture logicielle . . . . .	344
10.4.4.4. Construction de l'architecture opérationnelle . . . . .	346
10.4.5. Analyse de l'architecture candidate . . . . .	350
10.4.5.1. Analyse d'ordonnançabilité . . . . .	350
10.4.5.2. Analyse des latences entrées/sorties . . . . .	351
10.4.6. Quelques compléments . . . . .	354
10.4.6.1. Modes de fonctionnement . . . . .	354
10.4.6.2. Description des mécanismes de tolérance aux fautes . . . . .	355
10.4.6.3. Mécanismes d'extension du langage . . . . .	355
10.5. Conclusion . . . . .	357
10.6. Bibliographie . . . . .	359
<b>Index . . . . .</b>	<b>363</b>

