



Un système multi-agent adaptatif pour la distribution de l'intelligence dans les réseaux électriques



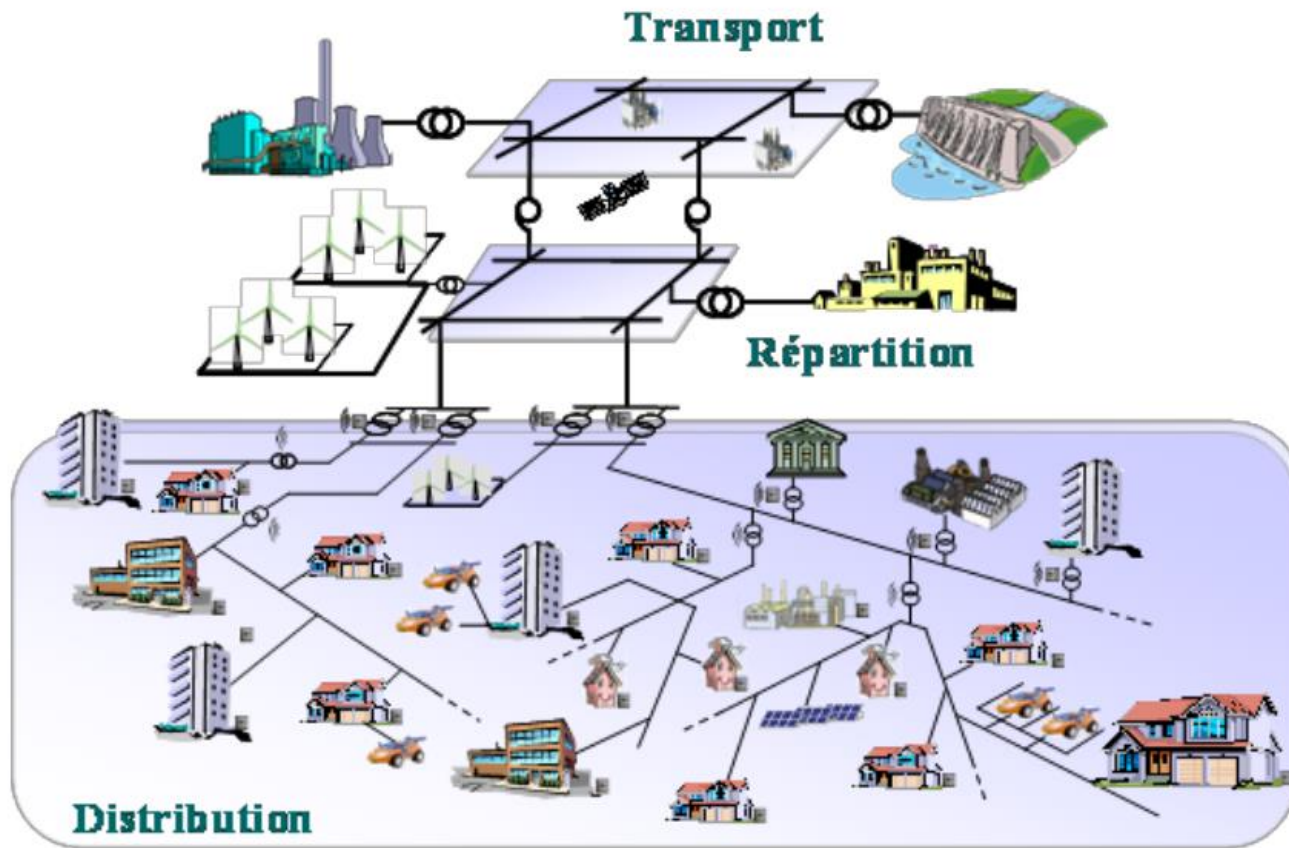
Plan de la présentation

1. Contexte
2. Système développé
3. Exemple de résolution



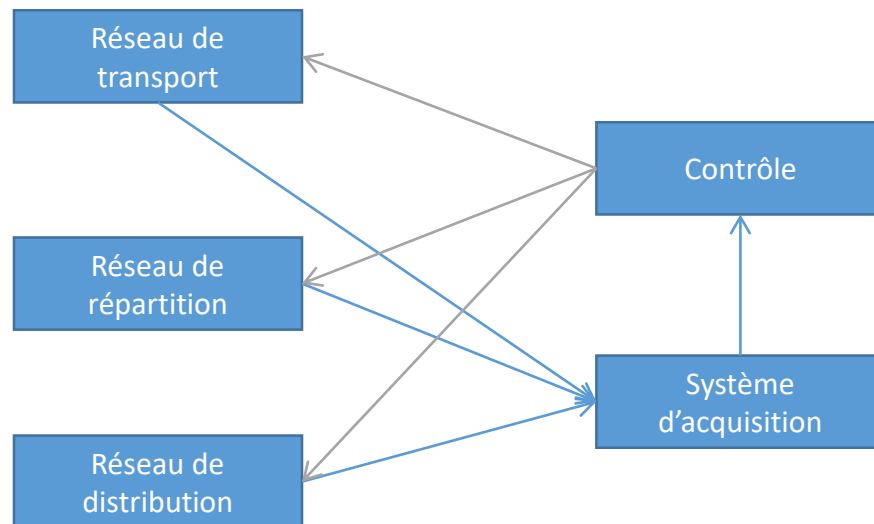
Contexte

Réseau électrique



- > 1 000 000 km de lignes
- Dizaines de millions de consommateurs

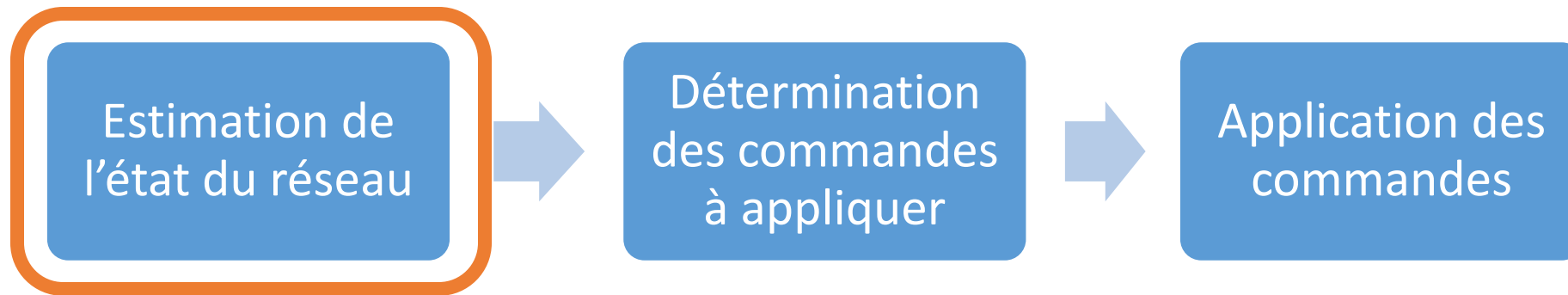
Gestion actuelle de réseaux



- Centralisation des données
- Contrôle centralisé

Objectifs du projet

- Régulation de tension



Résolution **émergente** de problème, **anytime**
Système Multi-Agent



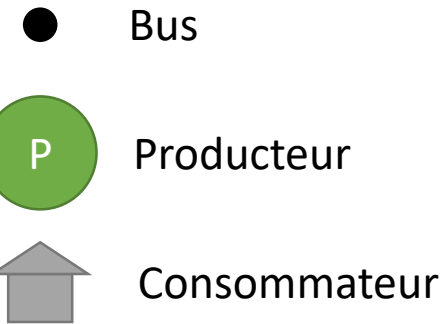
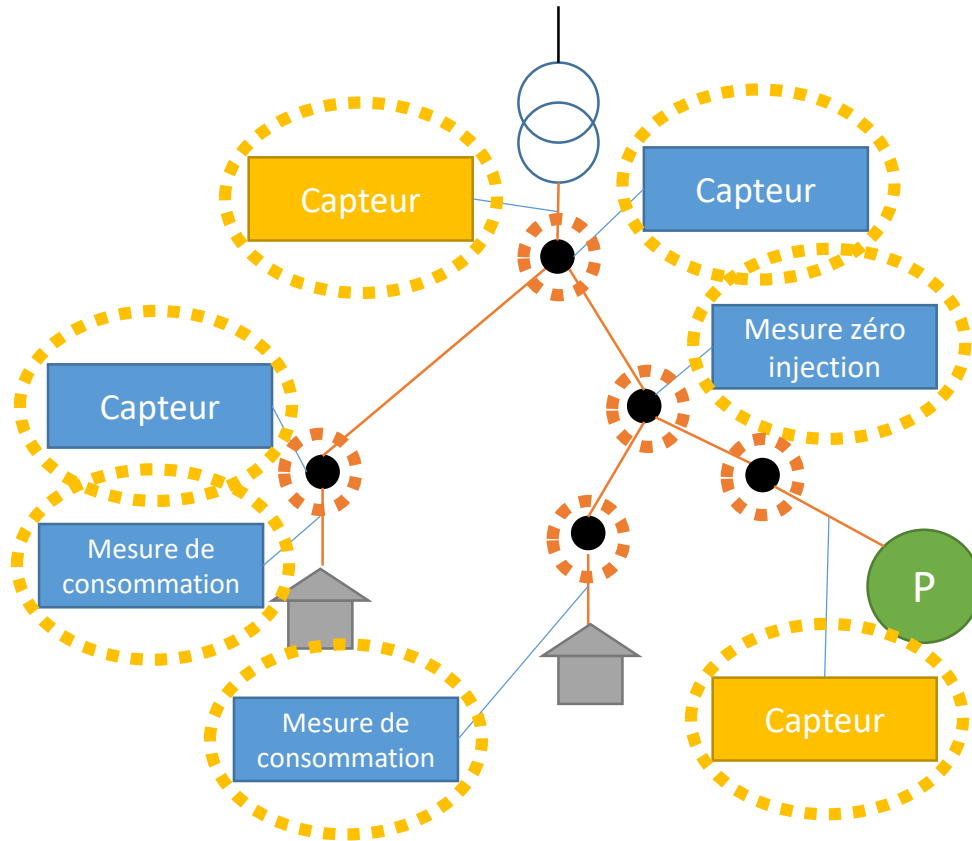
Contributions



Objectifs

- Contrôle intelligent du réseau → Problème complexe
- Estimation d'état pour la régulation de tension
- Propriétés
 - Généricité
 - Ouverture
 - Passage à l'échelle
 - Robustesse

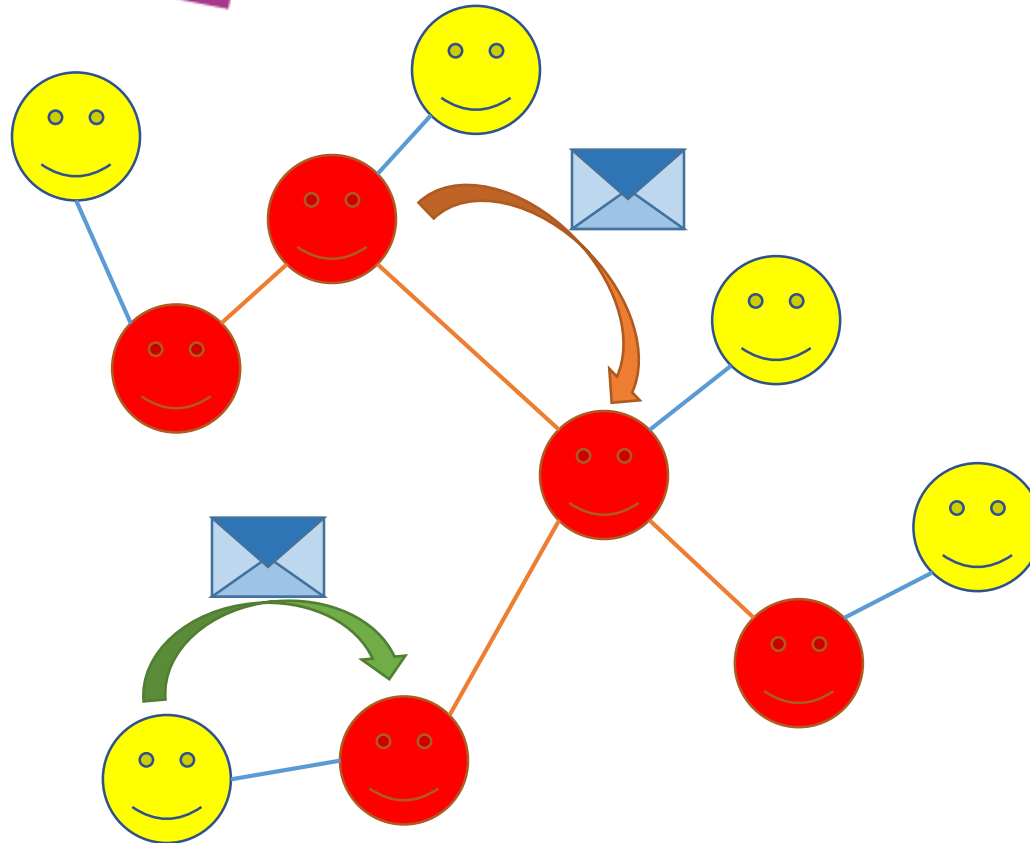
Agentification



Mesures

- Ensemble des capteurs
- Valeurs connues (zero injection)
- Modèles de consommation

Définition du voisinage des agents



Agent Measure

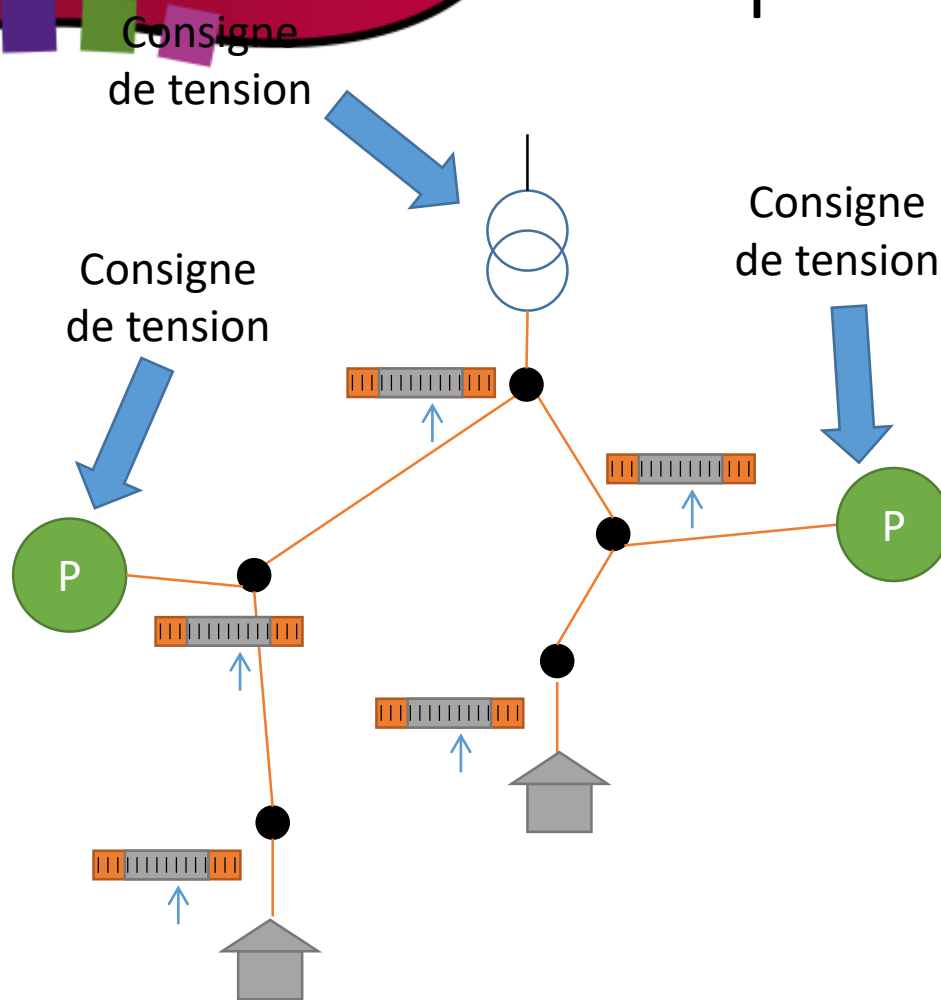


Agent Bus

- Interaction

- Entre agents Bus reliés via une ligne
- Entre agent Bus et agent Measure associés

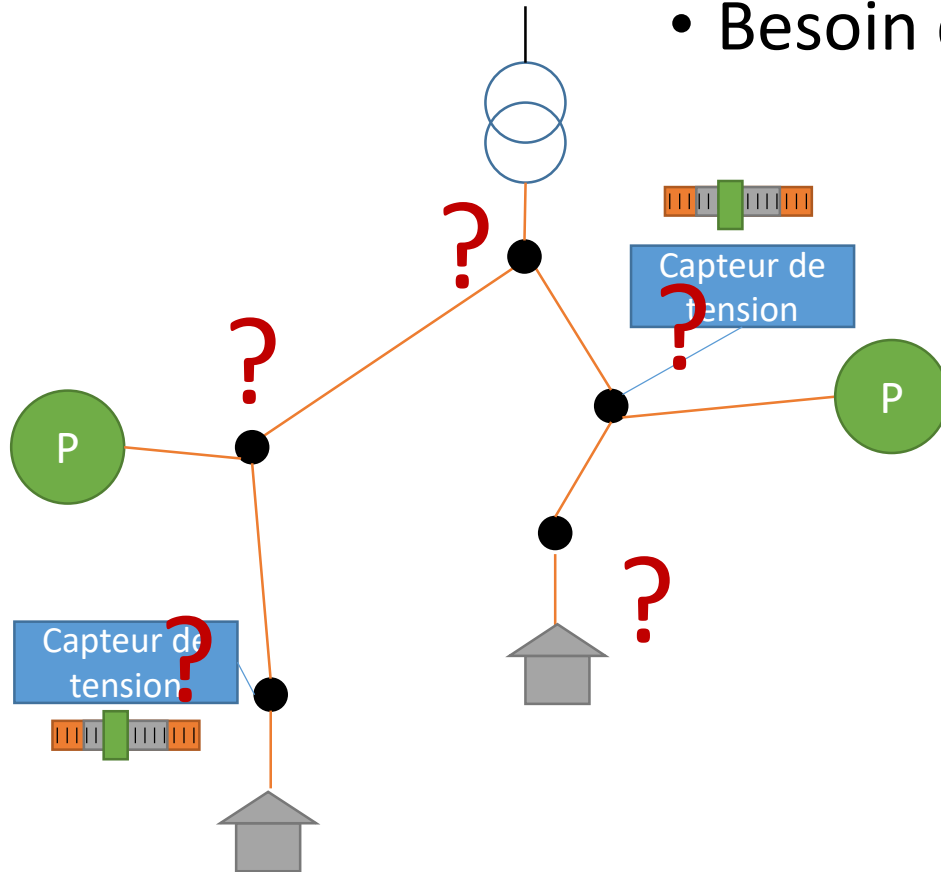
Pourquoi l'estimation d'état ?



- Caractéristique
 - Régulation de tension
- Déterminer les consignes de tension
 - Garantir que la tension en tout point sera comprise dans un intervalle prédéfini
 - Éviter les sous-tensions et sur-tensions

Régulation de tension

- Besoin d'avoir une idée de l'état des réseaux

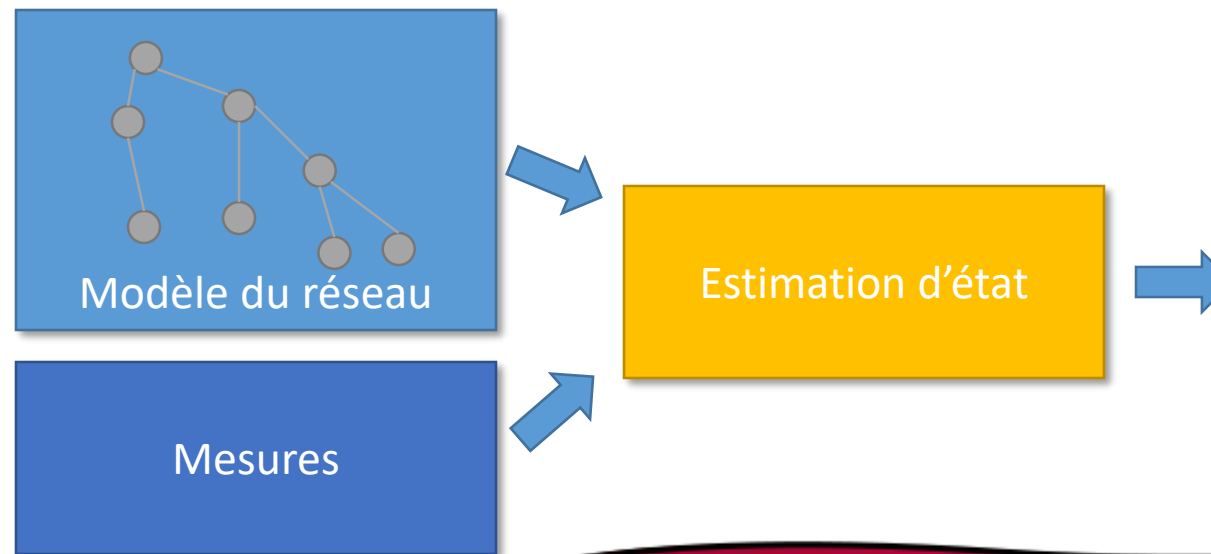


- Ajout de capteurs de tension et de puissance
 - Coût important
 - Manque de précision

Estimation d'état

Estimation d'état

Trouver l'état le plus vraisemblable d'un système étant donnés certaines quantités mesurées et le modèle du système et **filtrer les erreurs**



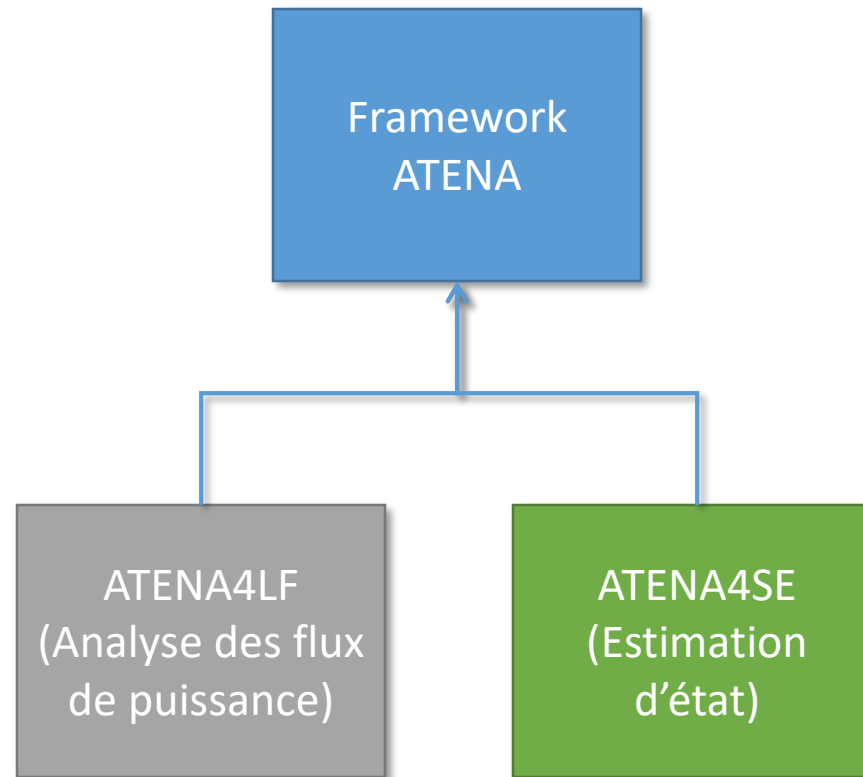
Maximum de vraisemblance

- Minimisation de la distance entre les valeurs mesurées et les valeurs calculées [Aldrich 1997]
- Chercher les paramètres de la loi qui maximisent la probabilité d'avoir observé un ensemble de valeurs
 - Distributions normales
- Formulation en moindres carrés pondérés
 - Somme des carrés des distances entre les valeurs des mesures (z_i) et les valeurs calculées ($h_i(x)$) pondérées par la précision des mesures (σ_i)

$$\sum_{i \in \text{Mesures}} \left(\frac{z_i - h_i(x)}{\sigma_i} \right)^2$$

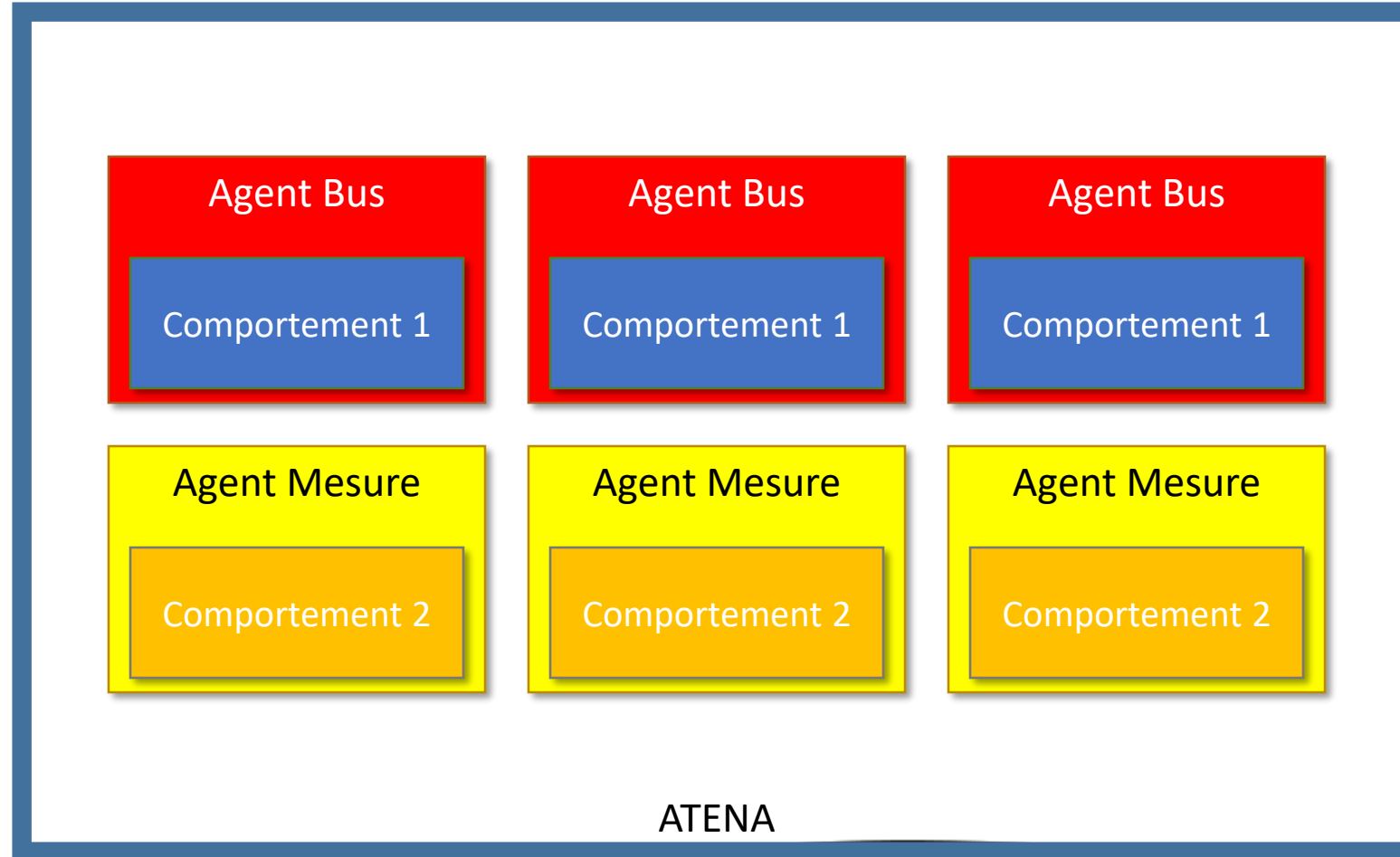
**Fonction objectif
à minimiser**

Instanciación du framework

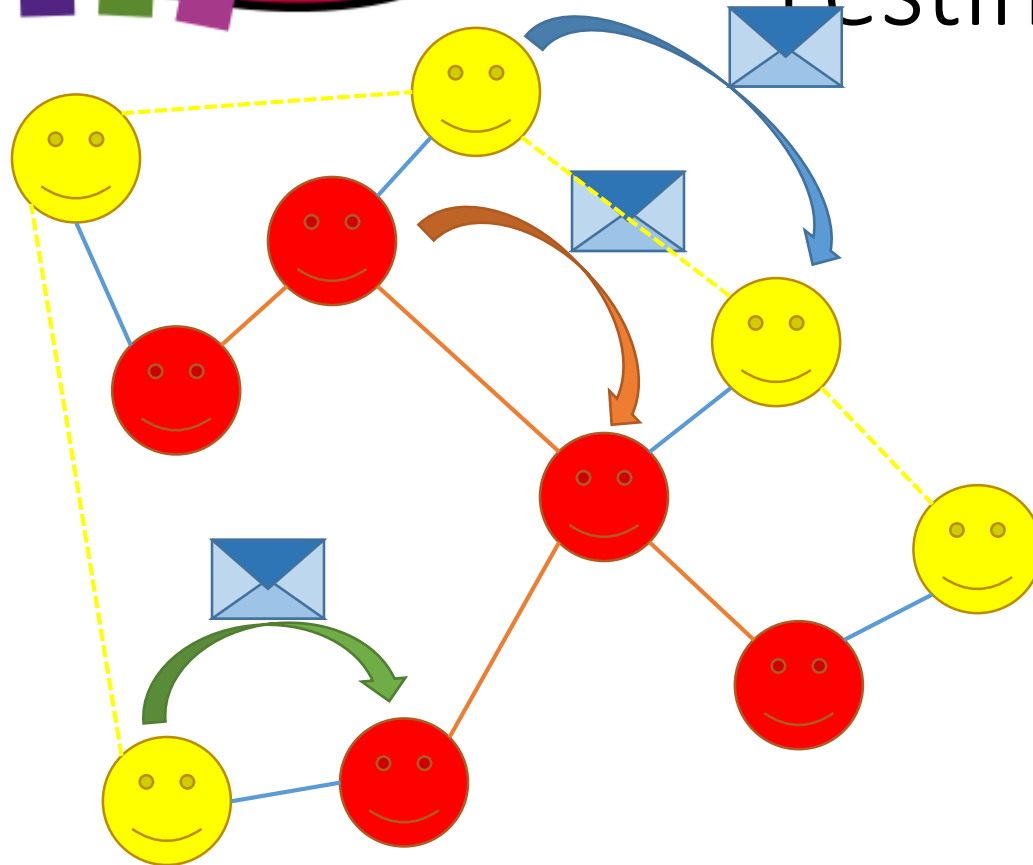


ATENA pour l'estimation d'état

ATENA4SE



Voisinage des agents pour l'estimation d'état



Agent Mesure



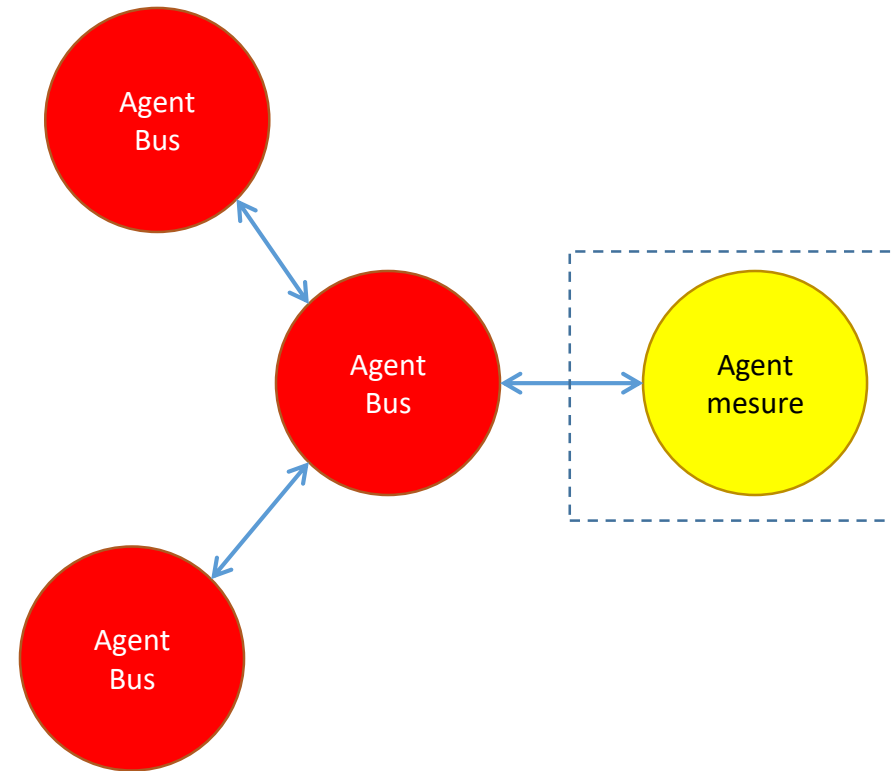
Agent Bus

• Communication

- Entre agents Bus reliés via une ligne
- Entre agent Bus et agent Mesure associés
- Entre agents Mesure
 - Pair à pair
 - Graphe connexe

Agent Bus

- But local
 - Déterminer les valeurs qui satisfont la loi des nœuds
- Criticité
 - Ecart à la loi des nœuds
- Voisinage
 - Agents Bus
 - Agent Mesure
- Perception locale
 - Les valeurs estimées par les voisins
- Décision
 - Réévaluer ses valeurs grâce aux perceptions
- Action
 - Envoyer ses nouvelles valeurs

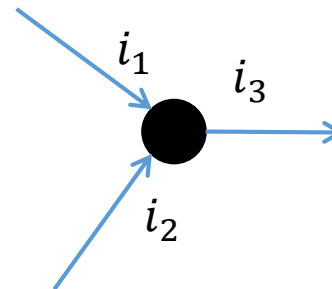


Loi des nœuds

« La somme des courants entrants dans un nœud est égale à la somme des courants sortants de ce même nœud » Kirchhoff

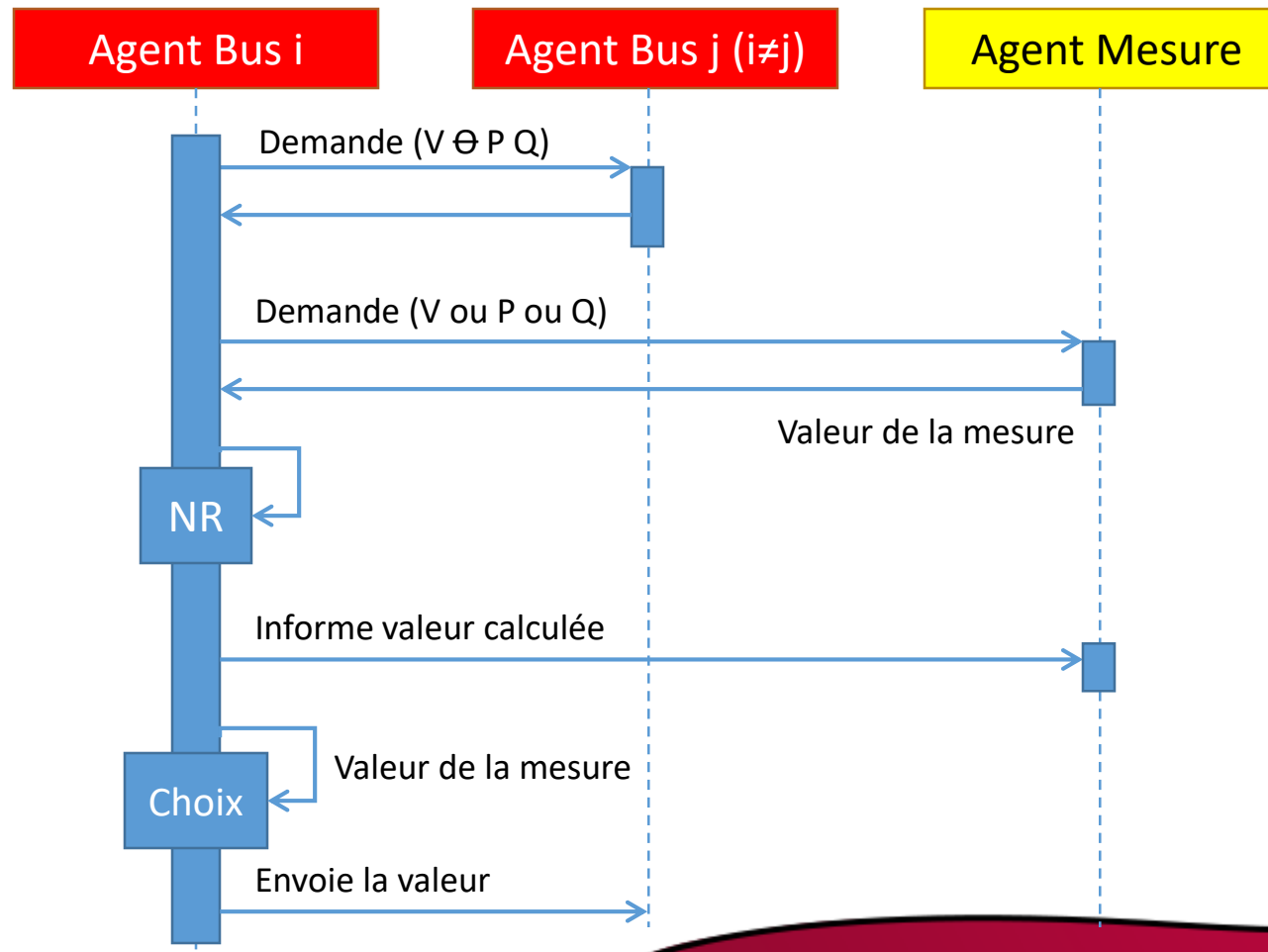
- Pour chaque nœud :

$$i_1 + i_2 = i_3$$



- Newton-Raphson
 - Méthode itérative de recherche de racine de fonction couramment utilisée
- Inversion de matrice
 - Complexité polynomiale d'ordre supérieur à 2

Décision de l'agent Bus



SNC de l'agent Bus

Non respect de
la loi des nœuds
(Conflit)

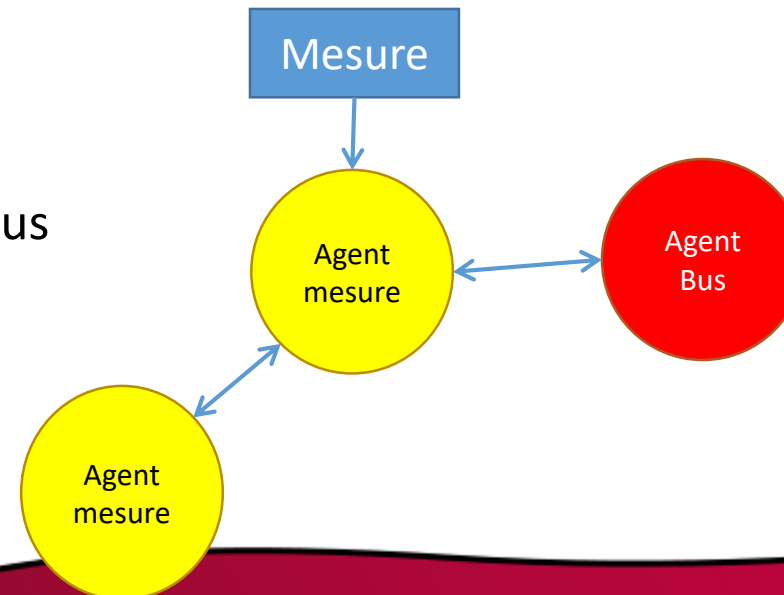
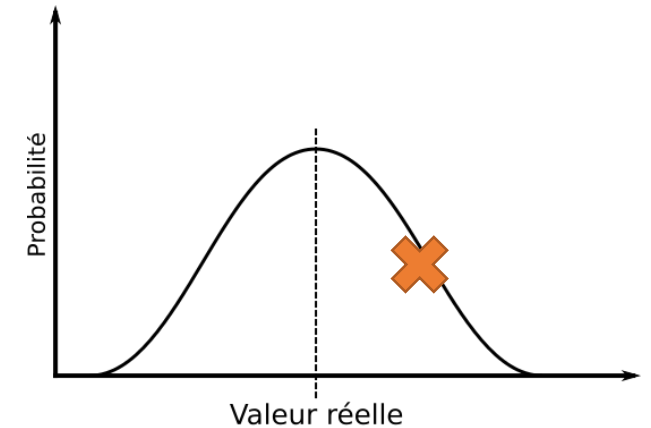
- Itération locale de Newton-Raphson pour satisfaire la loi des noeuds

Différence avec
l'agent Mesure
(Conflit)

- Modification de la valeur estimée au profit de celle de l'agent mesure

Agent Mesure

- But local
 - Essayer de corriger les mesures
- Criticité
 - Ecart à la mesure
- Voisinage
 - Agents Mesure
 - 1 Agent Bus
- Perception locale
 - Les valeurs estimées du voisin agent bus
- Décision
 - Réévaluer la valeur estimée
- Action
 - Envoyer sa nouvelle valeur



SNC de l'agent Mesure

Différence
avec la valeur
de l'agent Bus
(Conflit)

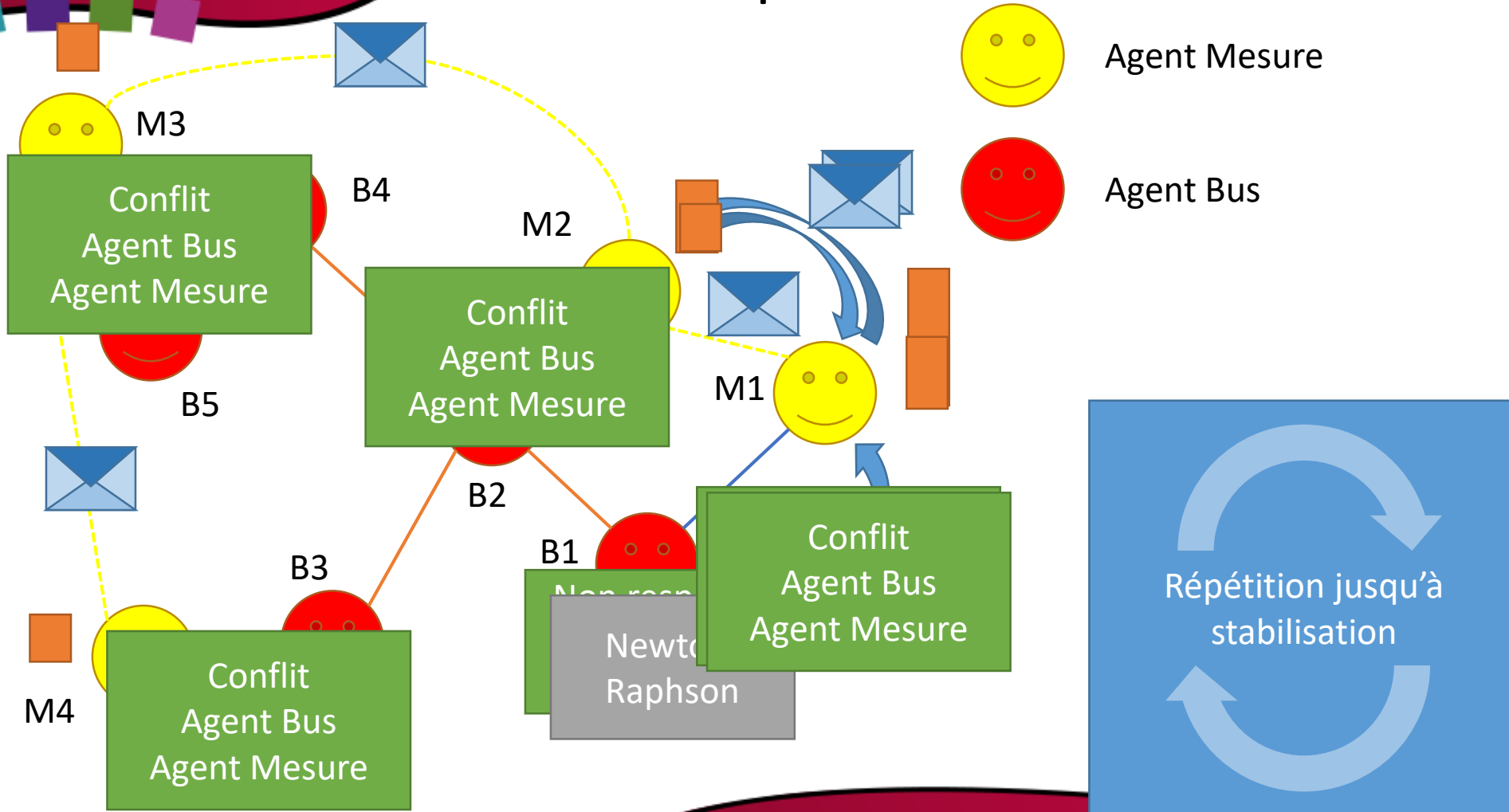
- Rapprocher sa valeur de celle de l'agent Bus (erreur peut augmenter)

Attitude coopérative

Agent Mesure
qui cherche à
compenser
son erreur

- Prise en charge d'une partie de l'erreur (si pas plus critique)

Exemple de résolution



Synthèse

- Deux types d'agents : Bus et Mesure
- Interactions entre les agents
- Code à réutiliser
 - Patron de conception
 - Modules de calcul
- Analyse des flux de puissance
- Estimation d'état

```
67
68
69
70 public NRA() {
71     v1Mat = MatrixUtils.createRealMatrix(2, 1);
72     pqMat = MatrixUtils.createRealMatrix(2, 1);
73     j = MatrixUtils.createRealMatrix(2, 2);
74 }
75
76 /**
77  * Compute an iteration of the Newton-Raphson
78  *
79  * @param v1 voltage magnitude of the concerned bus
80  * @param o1 voltage phase angle of the concerned bus
81  * @param neighborVoltages Voltages of the neighbors
82  * @param residual Distance to the Kirchhoff's Current Law
83  *
84  * @return the new voltage for the bus
85  */
86 public Complex computeNextValues(final double v1, final double o1, final Map<
87     final Complex residual) {
88     final Complex tmpV1 = ComplexUtils.polar2Complex(v1, o1);
89
90     v1Mat.setEntry(0, 0, tmpV1.getReal());
91     v1Mat.setEntry(1, 0, tmpV1.getImaginary());
92
93     Complex pq = Complex.ZERO;
94     for (final Map.Entry<Line, Complex> e : neighborVoltages.entrySet()) {
95         final Complex tmpV2 = ComplexUtils.polar2Complex(e.getValue().abs(),
96
97             pq = pq.add(f(tmpV1.getReal(), tmpV1.getImaginary(), tmpV2.getReal(),
98                 e.getKey().getV1(), e.getKey().getV2()));
99
100     }
101
102     pq = pq.subtract(residual);
103
104     pqMat.setEntry(0, 0, pq.getReal());
105     pqMat.setEntry(1, 0, pq.getImaginary());
106
107     Complex pqdvr1 = Complex.ZERO;
108     Complex pqdvi1 = Complex.ZERO;
109     for (final Map.Entry<Line, Complex> e : neighborVoltages.entrySet()) {
110         final Complex tmpV2 = ComplexUtils.polar2Complex(e.getValue().abs(),
111             pqdvr1 = pqdvr1.add(dfvr1(tmpV1.getReal(), tmpV1.getImaginary(), tmpV2
112                 e.getKey().getV1(), e.getKey().getV2()));
113             pqdvi1 = pqdvi1.add(dfvi1(tmpV1.getReal(), tmpV1.getImaginary(), tmpV2
114                 e.getKey().getV1(), e.getKey().getV2()));
```



Résolution

