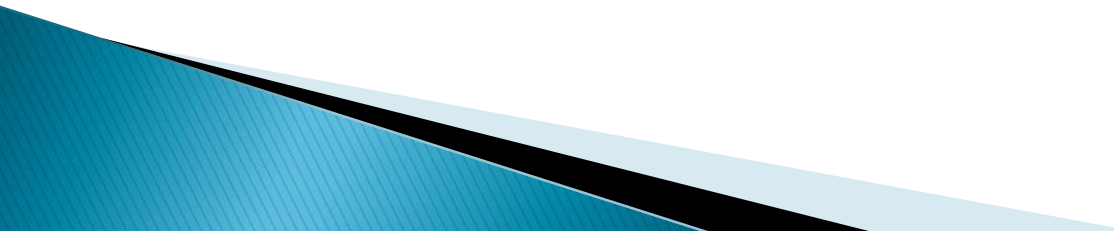


Projet CDC: Salons de discussion distribués

Johan Jobin et Julien Clément
Université de Fribourg, 2017



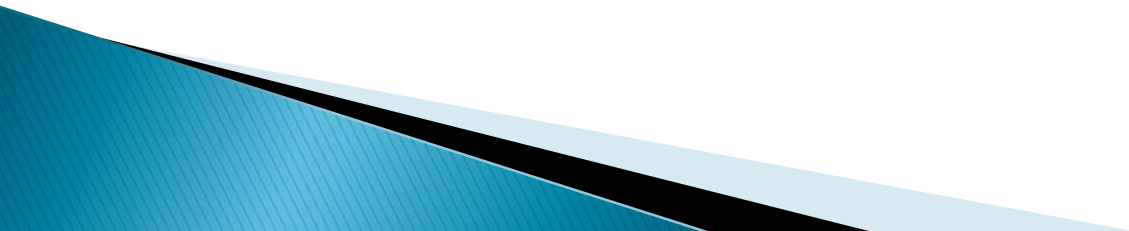
Sommaire

- ▶ Présentation du projet
 - ▶ Architecture et implémentation
 - Déploiement
 - Fonctionnement
 - ▶ Problème principal: exclusion mutuelle
 - Système de «token»
 - ▶ Evaluation de la solution
 - ▶ Résultats
 - ▶ Conclusion
- 

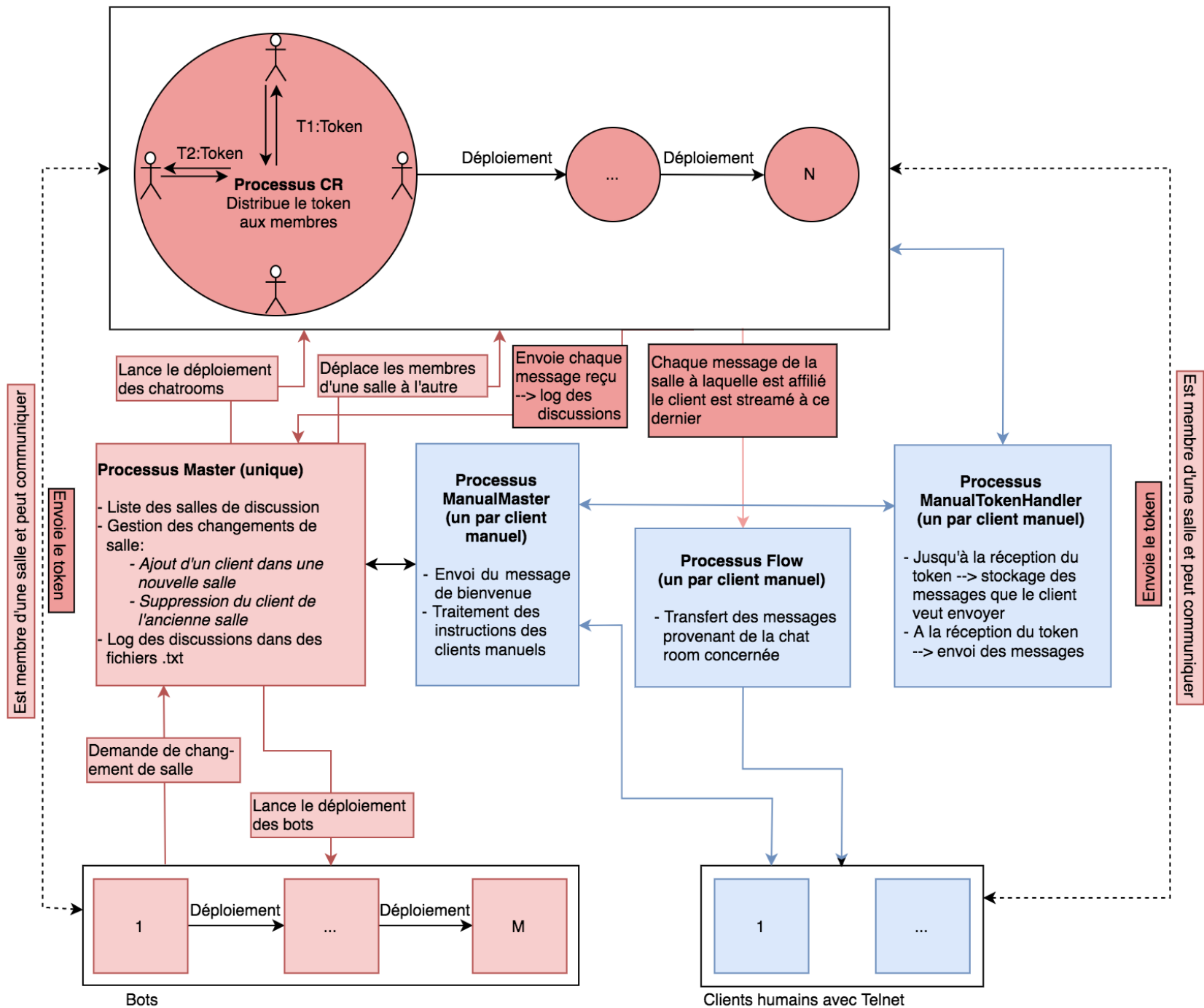
Présentation du projet

- ▶ Salons de discussion distribués composés de:
 - Bots: automatisés
 - Clients manuels: Connexion Telnet en TCP
- ▶ Système de token

Architecture

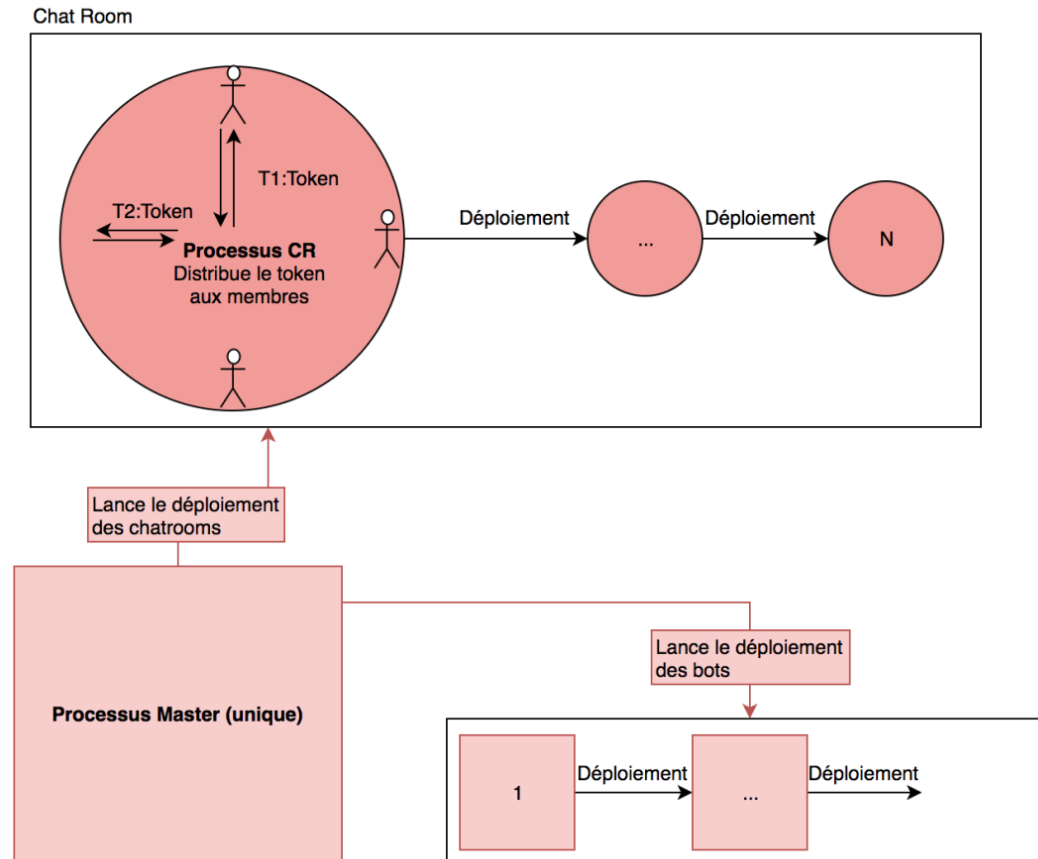


Chat Room



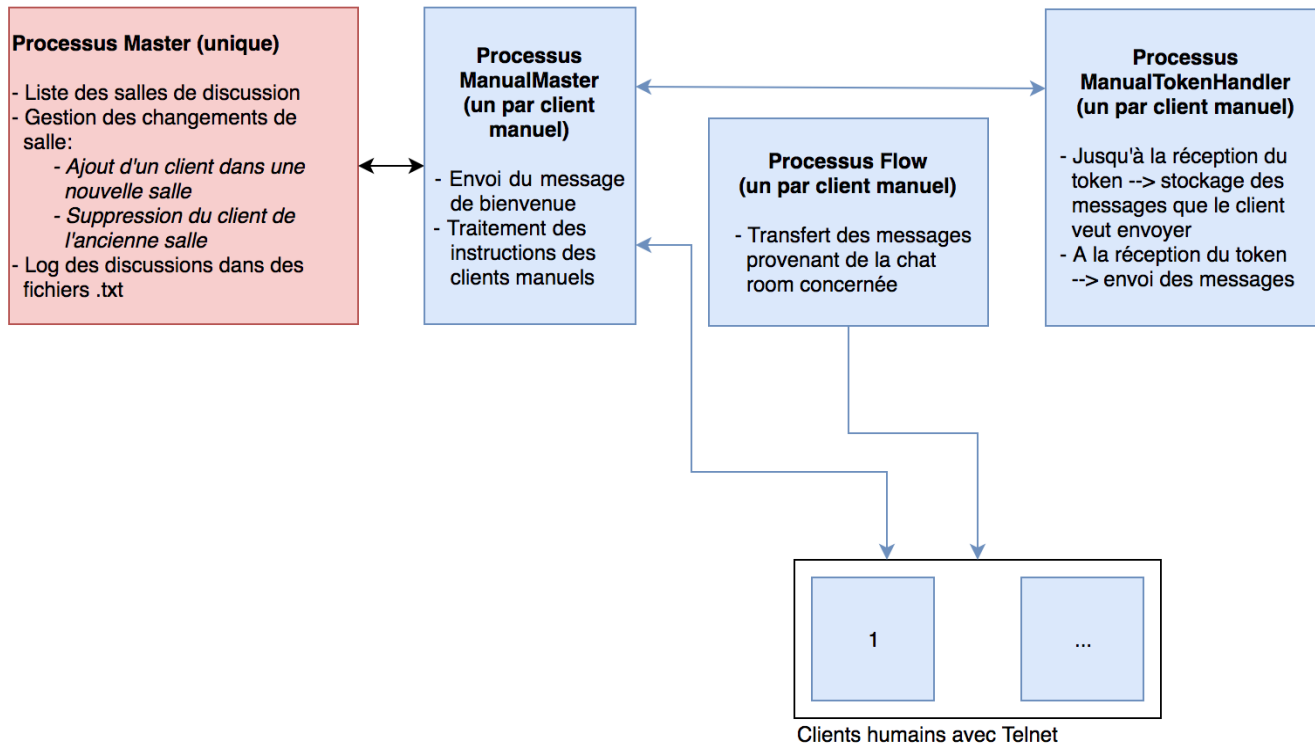
Salles de chat et bots: déploiement

- ▶ Processus master -> Salons de discussion -> Bots
- ▶ Une salle de discussion par ordinateur sur teDA
- ▶ Processus bots sur le reste des ordinateurs disponibles puis tout sur le même ordinateur

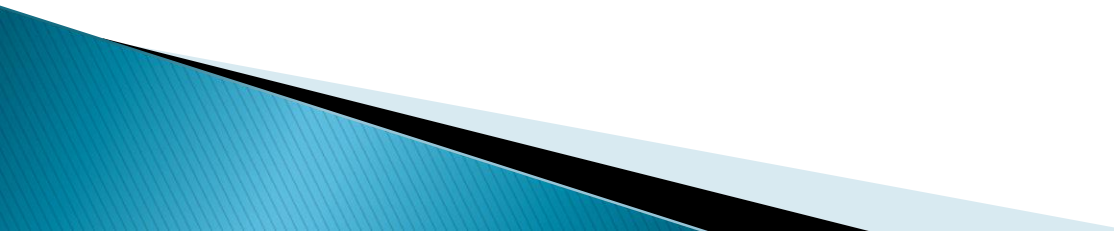


Clients humains: déploiement

- ▶ Master → 3 processus:
 - **ManualMaster**: attend la connexion TCP et gère la communication
 - **Flow**: stream les conversations
 - **ManualTokenHandler**: queue de message et gère le token du client manuel



Processus master

- ▶ Après le déploiement:
 - Liste des salles de discussion disponibles
 - Changement de salle d'un client (bot ou client manuel)
 - Log des discussions
 - Communication avec ManualMaster pour effectuer les commandes demandées par les clients manuels
- 

Salles de chat : fonctionnement

- ▶ Liste de leurs membres
 - Ajout/Suppression de membre de la liste
- ▶ Distribution du token aux membres
- ▶ Envoi de chaque message reçu:
 - Au Master pour le log
 - Vers Flow pour le stream

Bots : fonctionnement

- ▶ En possession du token, 2 choix:
 - Envoi d'un message (probabilité de 90%)
 - Phrase piochée au hasard dans un fichier texte
 - Changement de salle (probabilité de 10%)
- Dans tous les cas, renvoi du token à la salle de chat

ManualMaster: fonctionnement

- ▶ ManualMaster: attend la connexion TCP, envoie le menu



WELCOME

Johan Jobin, Julien Clement, University of Fribourg, 2017.

Commands :

send a message -> S_yourMessage

get the list of rooms -> G_

choose a room -> C_roomNumber, where roomNumber = 1, 2, ...

receive mode (stream of messages coming from the chat room) -> R_

suspend the incoming stream of messages -> P_

quit -> Q_

Flow: fonctionnement

- ▶ Envoi de chaque message de de la salle de chat vers Flow
 - Si le mode du client manuel = R_
 - Envoi instantané de messages vers le client manuel
 - Si le mode du client manuel = P_
 - Stockage dans une queue de messages et envoi de toute la queue lorsque le client repasse en mode R_

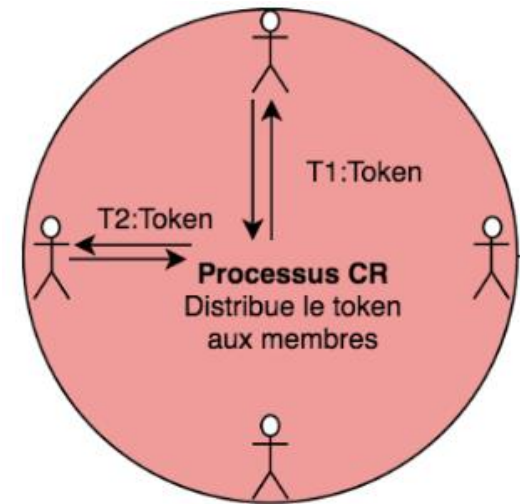
Exclusion mutuelle: token

► But:

- Garantir l'ordre de parole dans la salle de discussion
- Empêcher que deux clients parlent en même temps

► Fonctionnement:

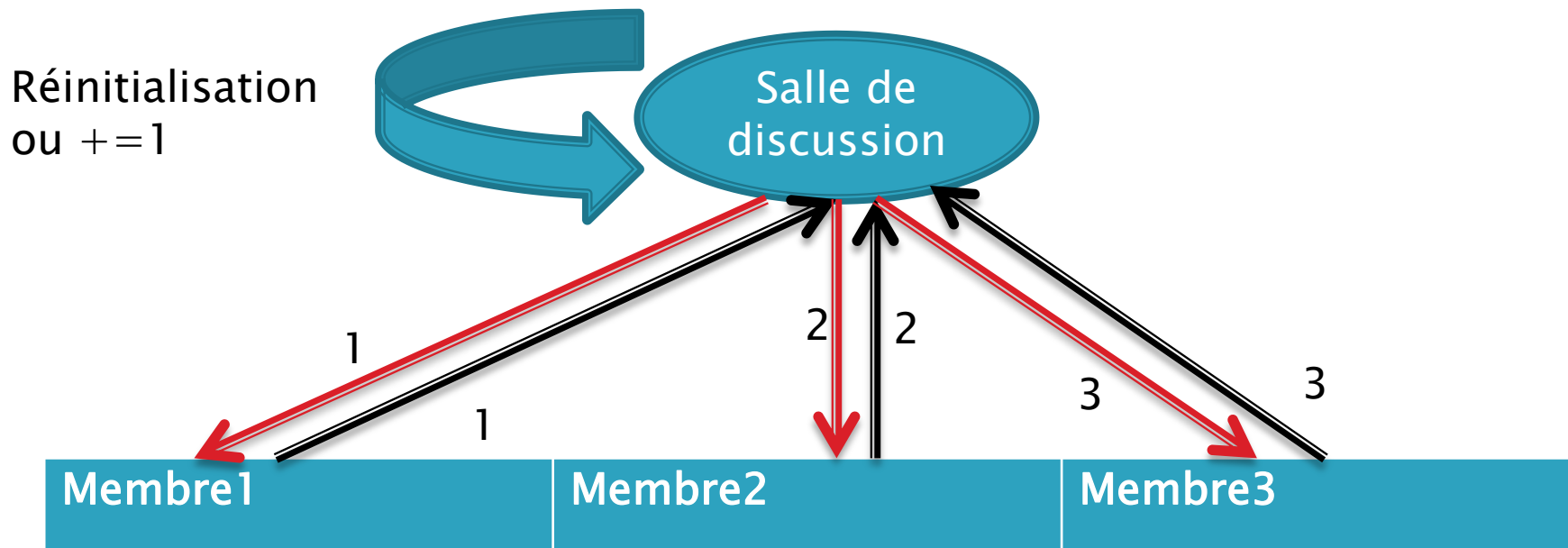
- Chaque client qui reçoit le token peut parler ou changer de salle
- Une fois l'action effectuée, il le renvoie à la salle de discussion qui le renvoie au prochain membre



Système de token (suite)

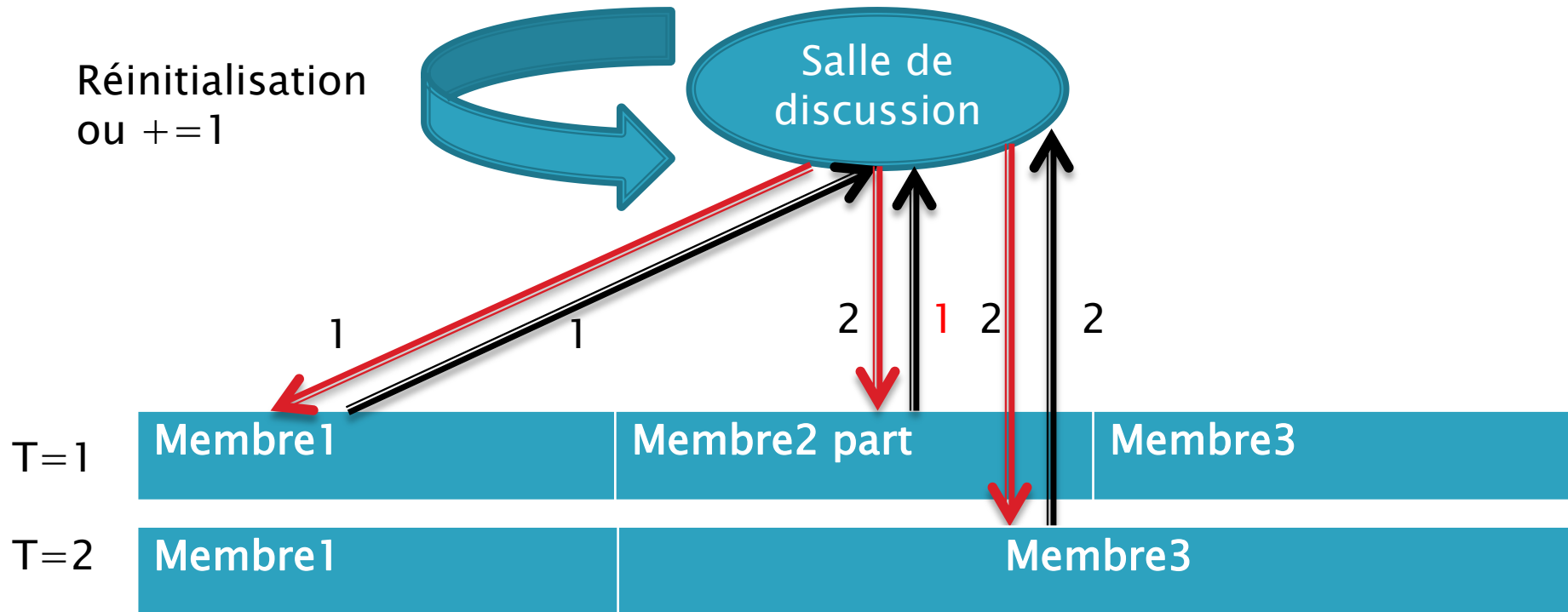
► Implémentation:

- Entier compris entre 1 et le nombre de membres de la salle de discussion, correspondant à sa place dans la liste de membres



Système de token (suite)

► Situation de changement de salle



Evaluation de la solution

► Système testé avec plusieurs scénarios

1. Un salon de discussion, N bots

- I. Vérification du déploiement
- II. Vérification du système de token

2. N salons de discussion, N bots

- I. Changements de salle

3. N salons de discussion, N bots et N clients manuels

- I. Communication en utilisant TCP
- II. Commandes G_, C_, S_, P_, R_, Q_
- III. Stream de Flow
- IV. Processus ManualTokenHandler (queue de messages en attendant de recevoir le token)

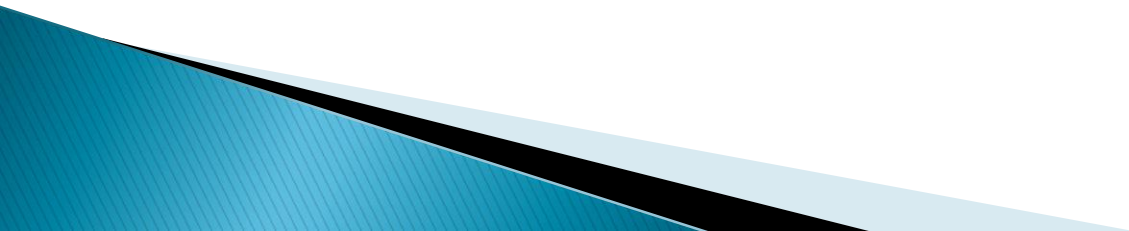
Résultats

1. Les bots parlent les uns après les autres, le token est bien réinitialisé à chaque tour et la discussion garde un ordre cohérent.
2. Les changements de salle de discussion, peu importe quand ils surviennent, se font sans poser problème.
3. Connexion TCP fonctionnelle, toutes les commandes sont utilisables et `manualTokenHandler` remplit son rôle.

Conclusions

- ▶ Avantages de notre système:
 - Système efficace pour l'exclusion mutuelle
 - Mise en œuvre relativement simple
 - Stable
 - Code facilement extensible pour ajouter de nouvelles fonctionnalités (ajout de salles de discussion, de nouvelles commandes etc..)
- ▶ Inconvénients
 - Inefficace dans le cas de gros salons car possibilité de communiquer uniquement en possession du token

**Merci de votre
attention !**



Démonstration

