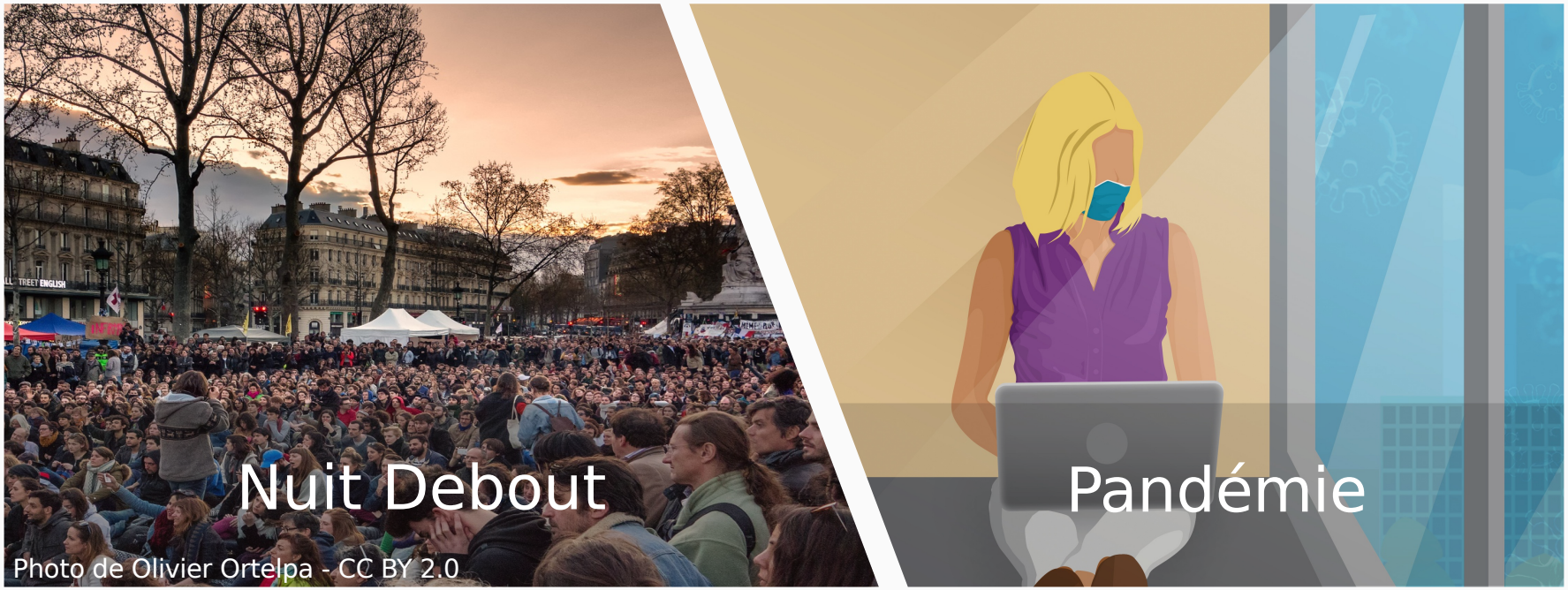
Réplication sécurisée dans les infrastructures pair-à-pair de collaboration

Victorien Elvinger

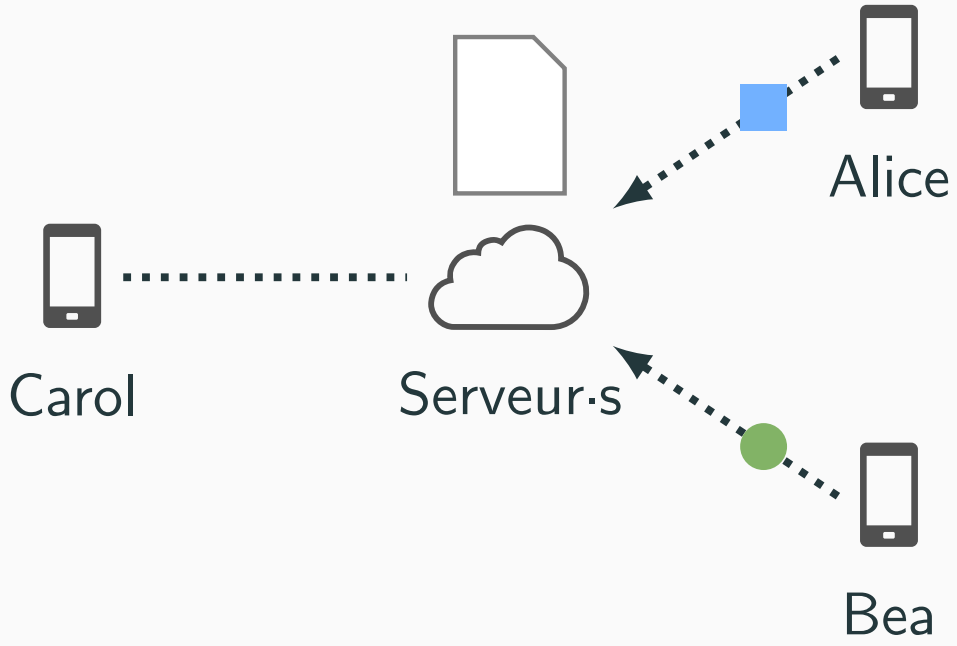

Juin 2021

Membres du jury :	
Steve Kremer	directeur de recherche Inria Grand Est
Emmanuelle Anceaume	directrice de recherche IRISA
Pascal Molli	professeur à l'Université de Nantes
Esther Pacitti	professeure à l'Université de Montpellier 2
François Charoy	professeur à l'Université de Lorraine
Gérald Oster	maître de conférence à l'Université de Lorraine

Applications de collaboration



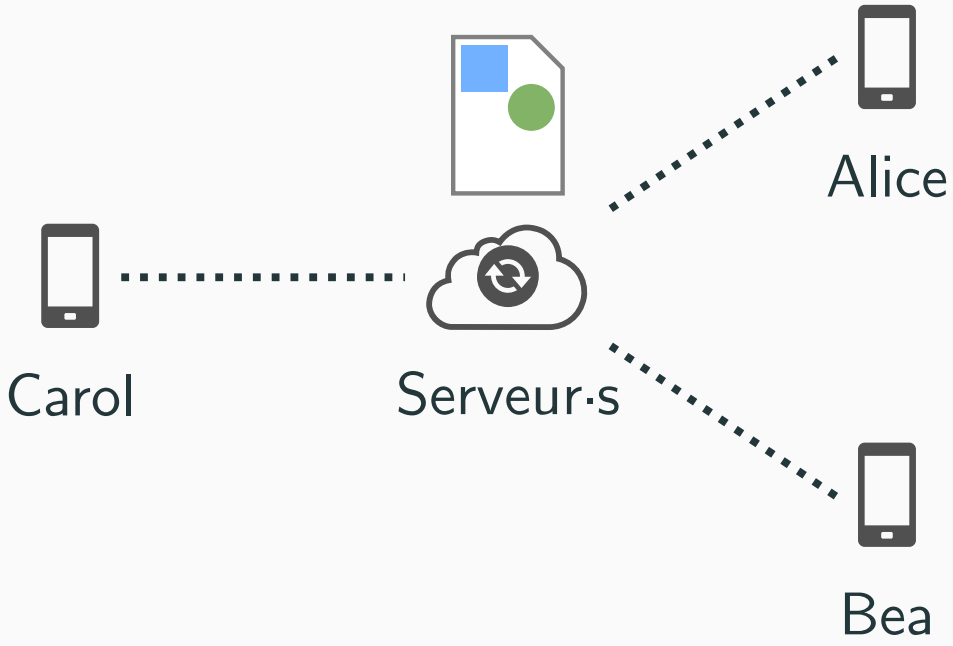
- plusieurs individus **modifient ensemble** un contenu
  - localisés à des **endroits différents**
  - modifications en **simultané** ou à des **moments distincts**
- **collaborations massives**
  - met en lumière les limites des applications existantes



Limites

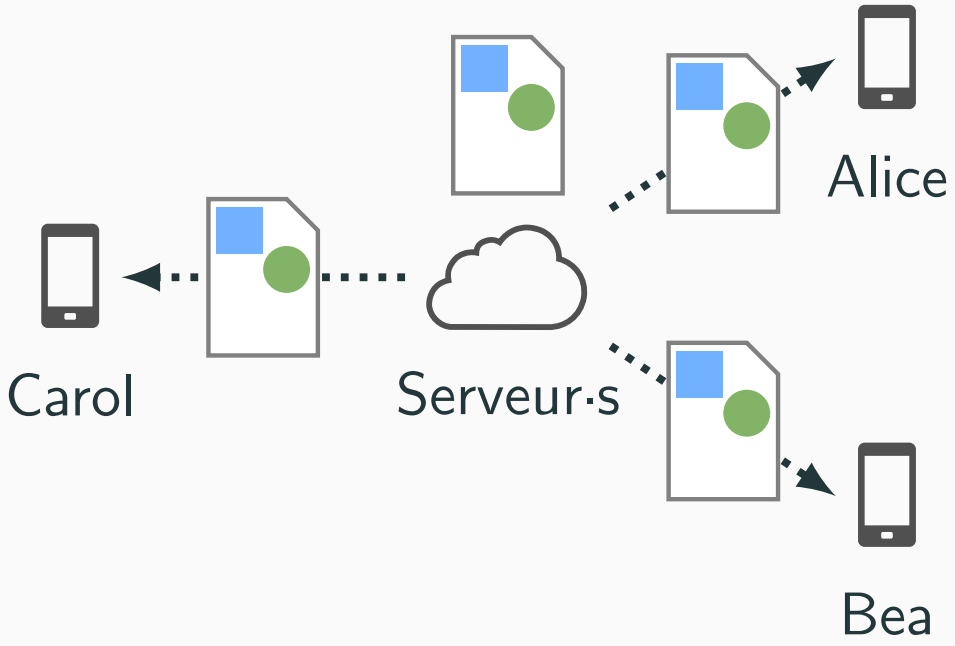
- disponibilité
- latence
- sécurité et vie privée
- propriété des contenus
- passage à l'échelle
- modes de collaboration (hors ligne)

Infrastructure centralisée de collaboration



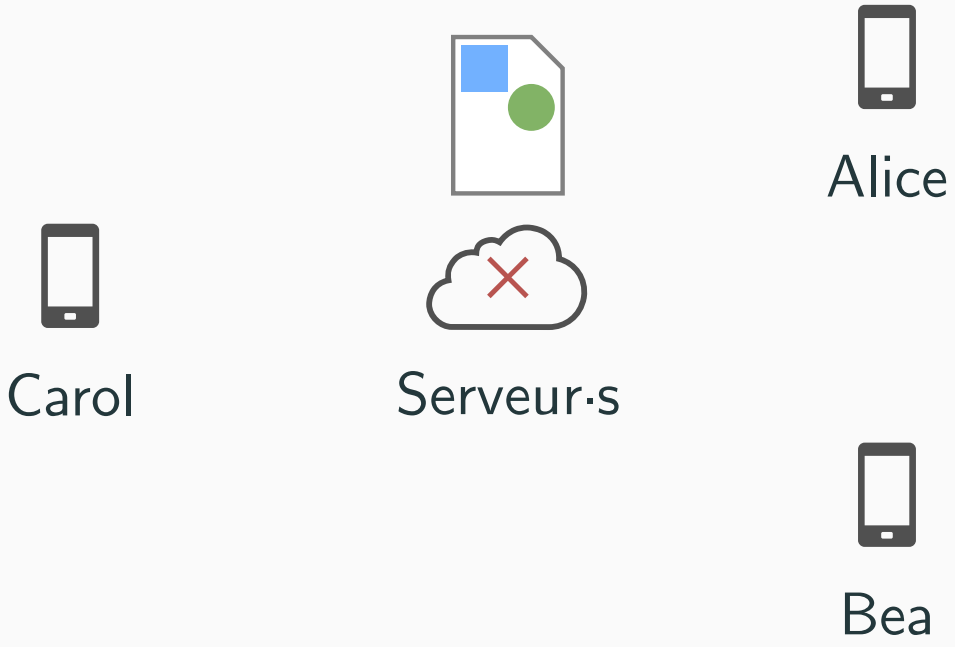
Limites

- disponibilité
- latence
- sécurité et vie privée
- propriété des contenus
- passage à l'échelle
- modes de collaboration (hors ligne)



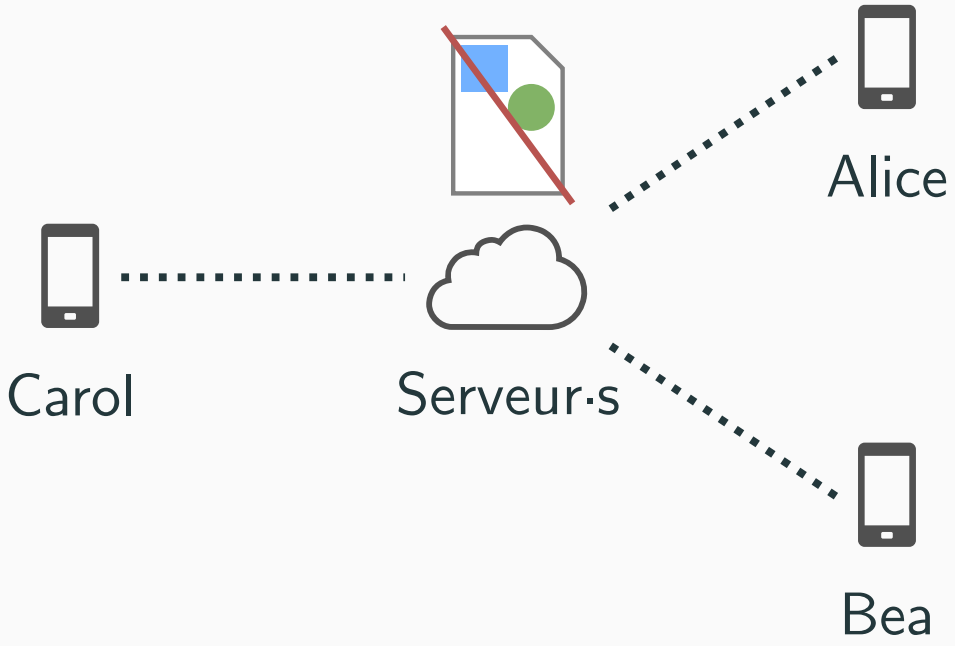
Limites

- disponibilité
- latence
- sécurité et vie privée
- propriété des contenus
- passage à l'échelle
- modes de collaboration (hors ligne)



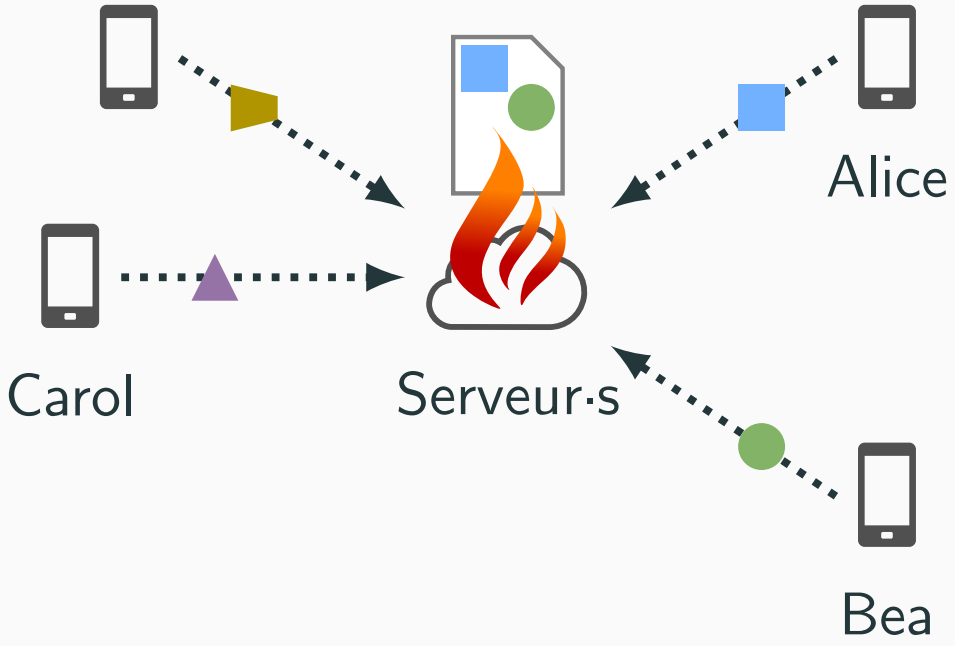
Limites

- disponibilité
- latence
- sécurité et vie privée
- propriété des contenus
- passage à l'échelle
- modes de collaboration (hors ligne)



Limites

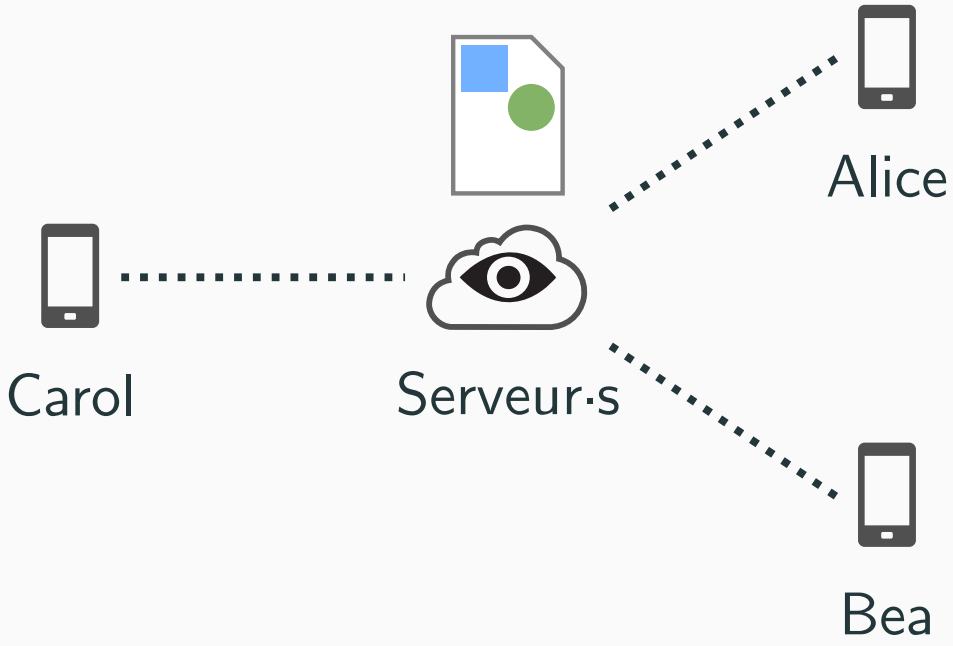
- disponibilité
- latence
- sécurité et vie privée
- propriété des contenus
- passage à l'échelle
- modes de collaboration (hors ligne)



Limites

- disponibilité
- latence
- sécurité et vie privée
- propriété des contenus
- passage à l'échelle
- modes de collaboration (hors ligne)

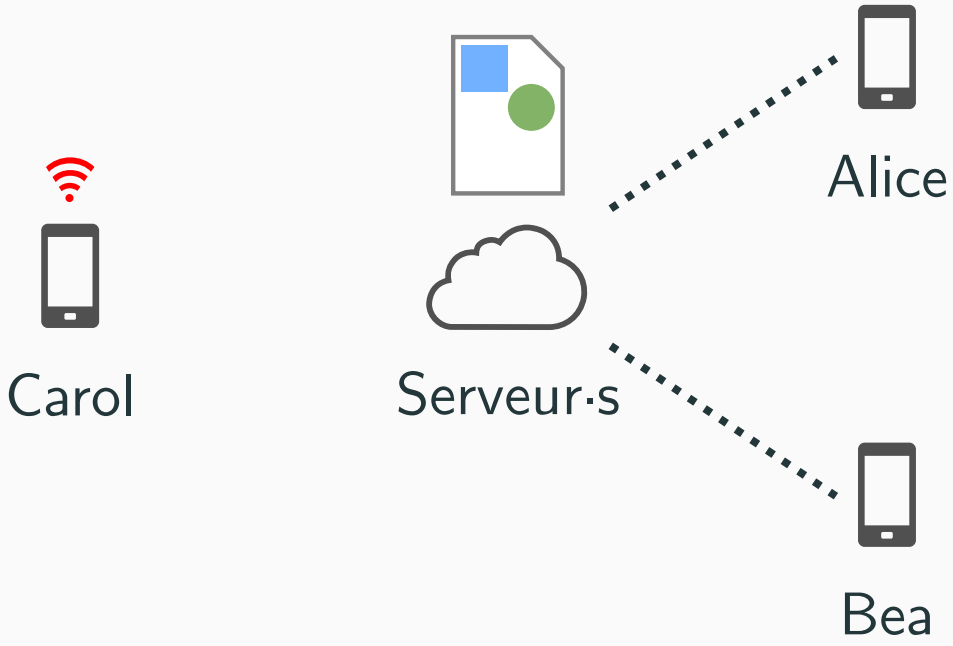
Infrastructure centralisée de collaboration



Limites

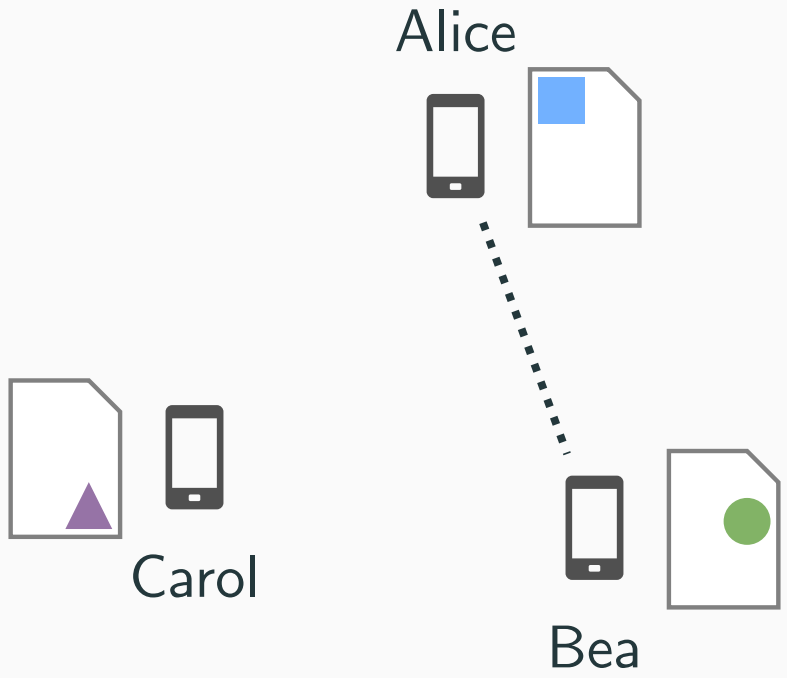
- disponibilité
- latence
- sécurité et vie privée
- propriété des contenus
- passage à l'échelle
- modes de collaboration (hors ligne)

Infrastructure centralisée de collaboration

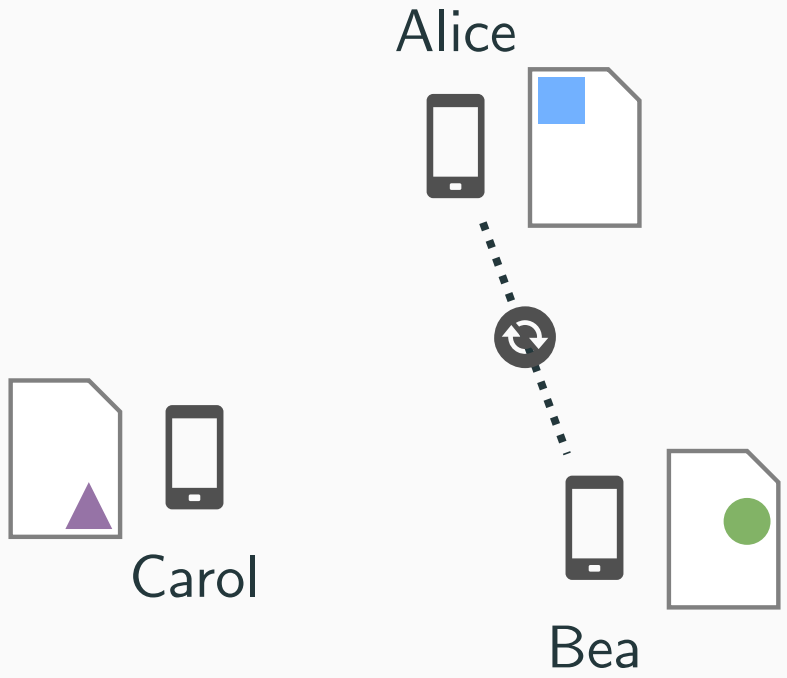


Limites

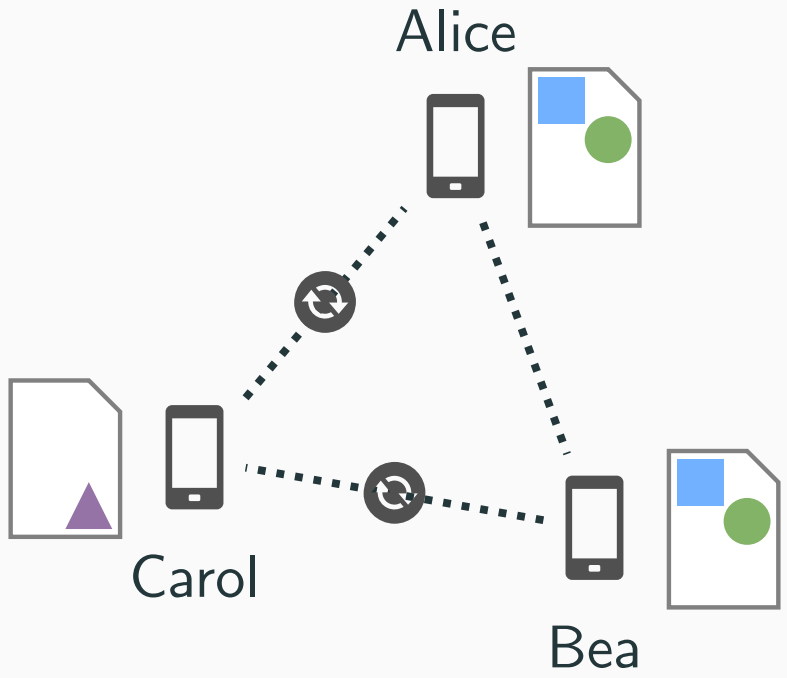
- disponibilité
- latence
- sécurité et vie privée
- propriété des contenus
- passage à l'échelle
- modes de collaboration (hors ligne)



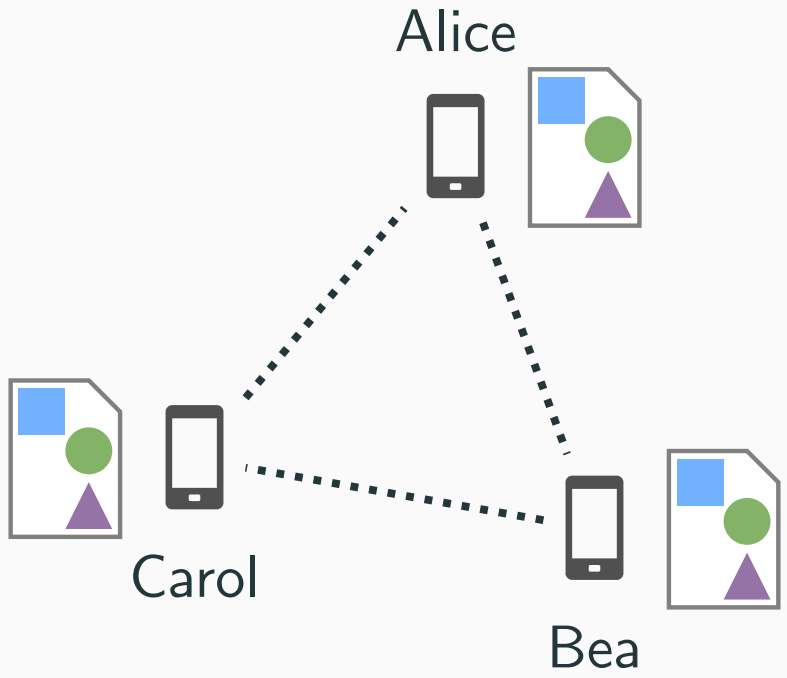
- les pairs possèdent leur **propre copie** du contenu
  - modifiable à **tout moment** et **sans coordination**
  - **convergence à terme**
- la convergence des copies est une propriété essentielle



- les pairs possèdent leur **propre copie** du contenu
  - modifiable à **tout moment** et **sans coordination**
  - **convergence à terme**
- la convergence des copies est une propriété essentielle



- les pairs possèdent leur **propre copie** du contenu
  - modifiable à **tout moment** et **sans coordination**
  - **convergence à terme**
- la convergence des copies est une propriété essentielle

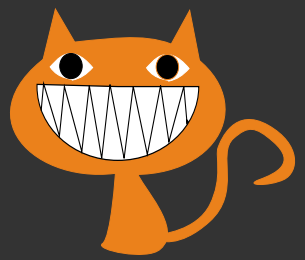


- les pairs possèdent leur **propre copie** du contenu
  - modifiable à **tout moment** et **sans coordination**
  - **convergence à terme**
- la convergence des copies est une propriété essentielle

Problématique 1 : peut-on assurer la convergence des copies en présence de pairs malintentionnés ?

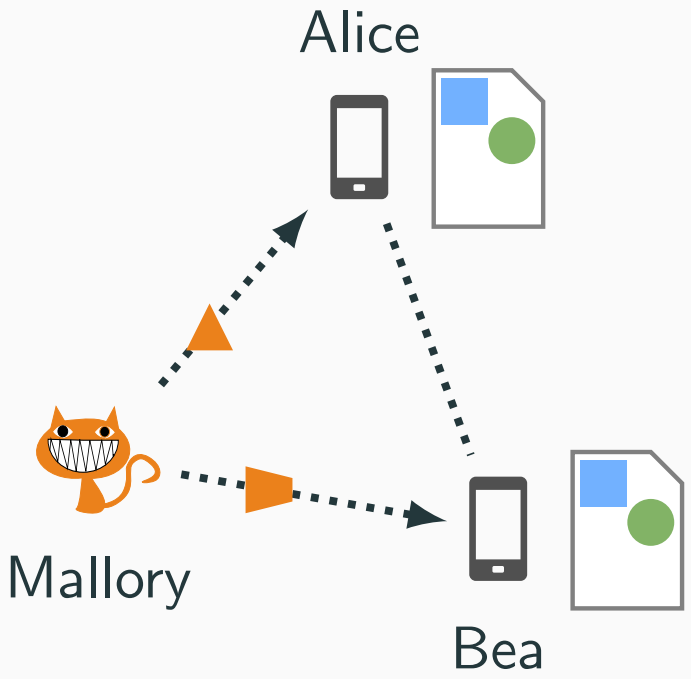
Problématique 2 : peut-on concevoir un protocole pour la co-édition de texte qui résiste mieux aux aléas du réseau et aux longues périodes de déconnexion ?

Problématique 1 : peut-on assurer la convergence des copies en présence de pairs malintentionnés ?



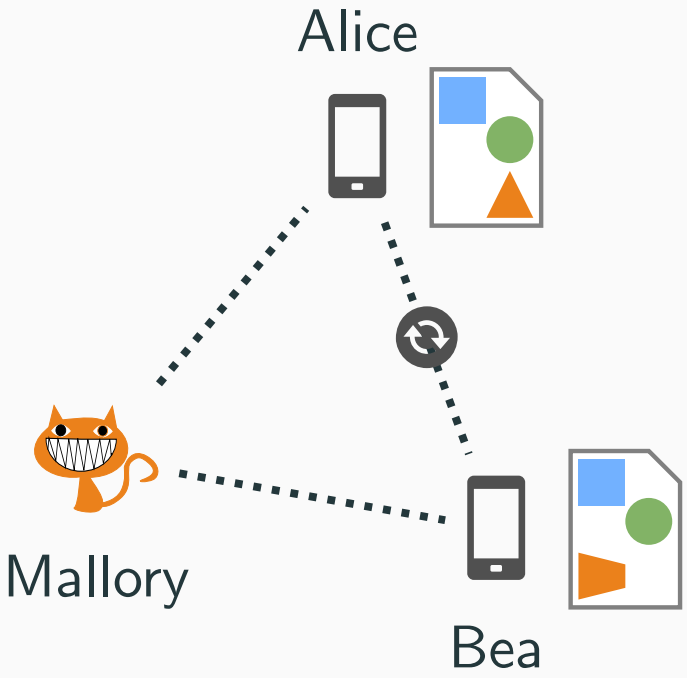
Mallory

Qu'est ce qu'un pair malintentionné ?



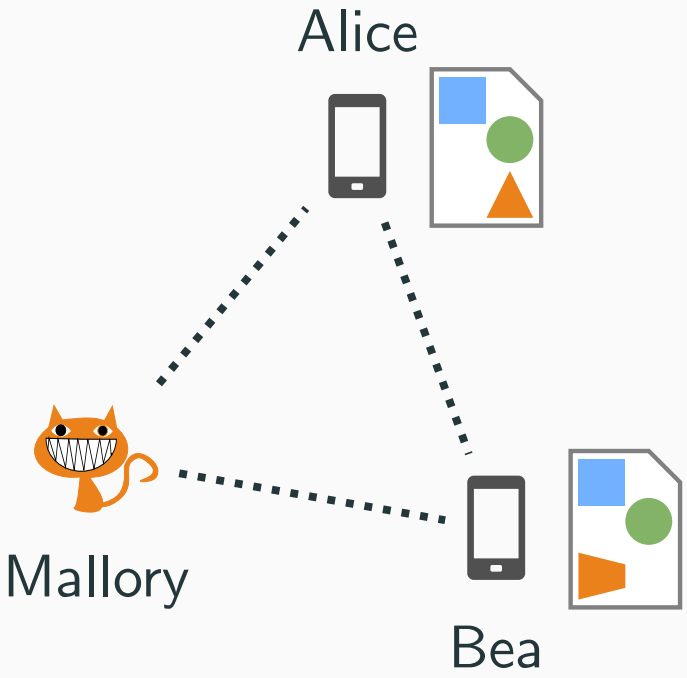
- adversaire de la collaboration
  - objectif : **empêcher la convergence** des copies
  - *Byzantin* : **pairs malintentionnés + réseau asynchrone corrompu**
- un pair malintentionné peut compromettre la convergence par **équivoques** = modifications distinctes perçues comme identiques

Qu'est ce qu'un pair malintentionné ?



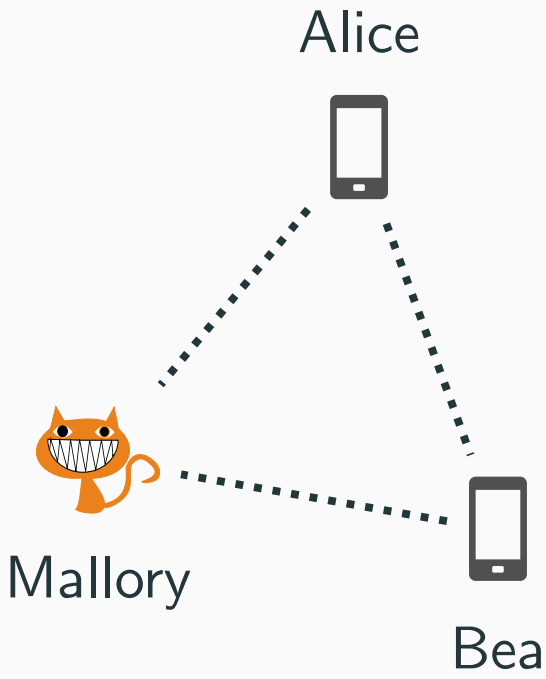
- adversaire de la collaboration
  - objectif : **empêcher la convergence** des copies
  - *Byzantin* : **pairs malintentionnés + réseau asynchrone corrompu**
- un pair malintentionné peut compromettre la convergence par **équivoques** = modifications distinctes perçues comme identiques

Qu'est ce qu'un pair malintentionné ?



- adversaire de la collaboration
  - objectif : **empêcher la convergence** des copies
  - *Byzantin* : **pairs malintentionnés + réseau asynchrone corrompu**
- un pair malintentionné peut compromettre la convergence par **équivoques** = modifications distinctes perçues comme identiques

Journaux de modifications



journal de Alice

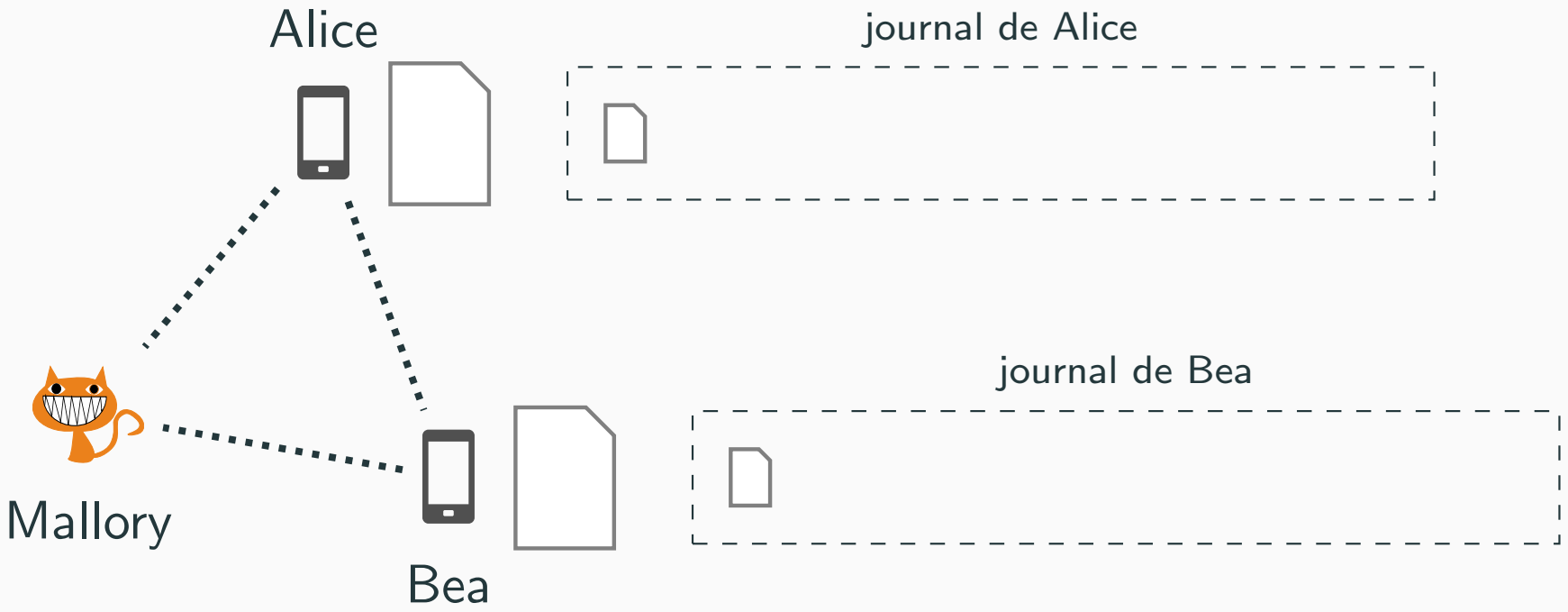
journal de Bea

- un pair enregistre dans son journal les **modifications qu'il intègre**
- arrangement horizontal = ordre d'ajout des modifications

Journaux de modifications

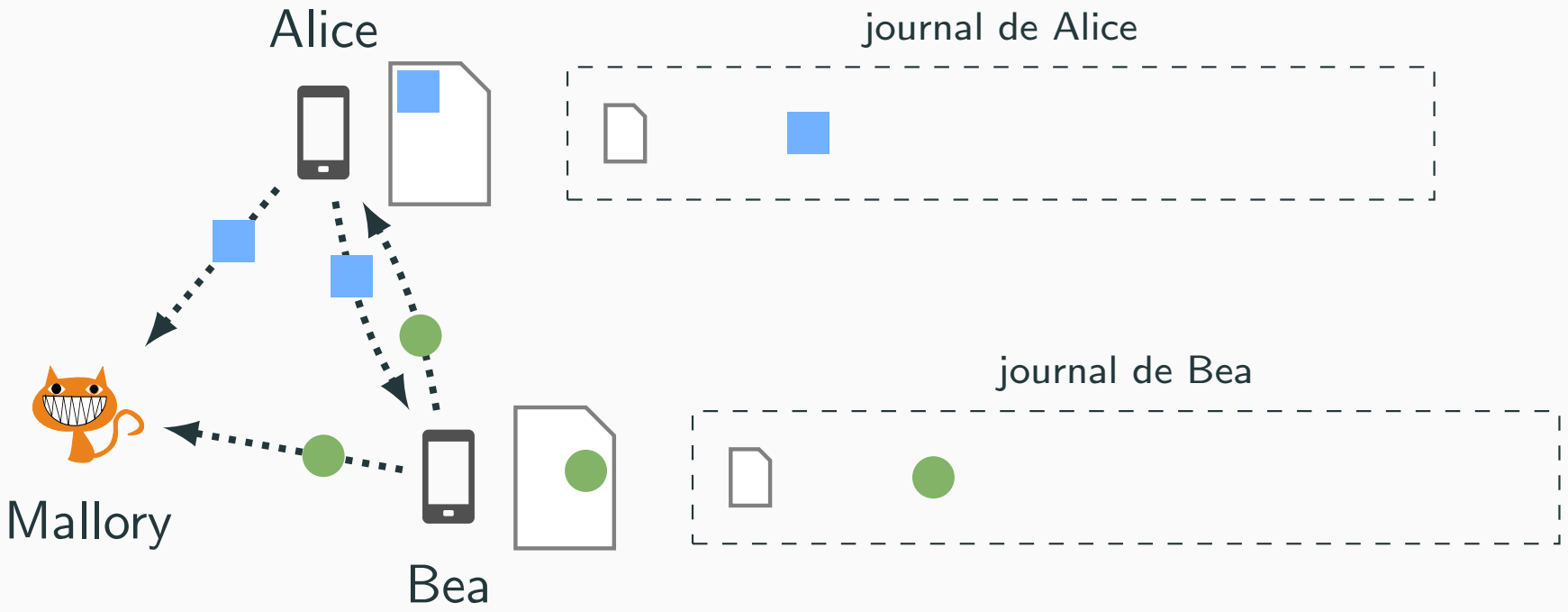
- un pair enregistre dans son journal les **modifications qu'il intègre**
- arrangement horizontal = ordre d'ajout des modifications

Journaux de modifications



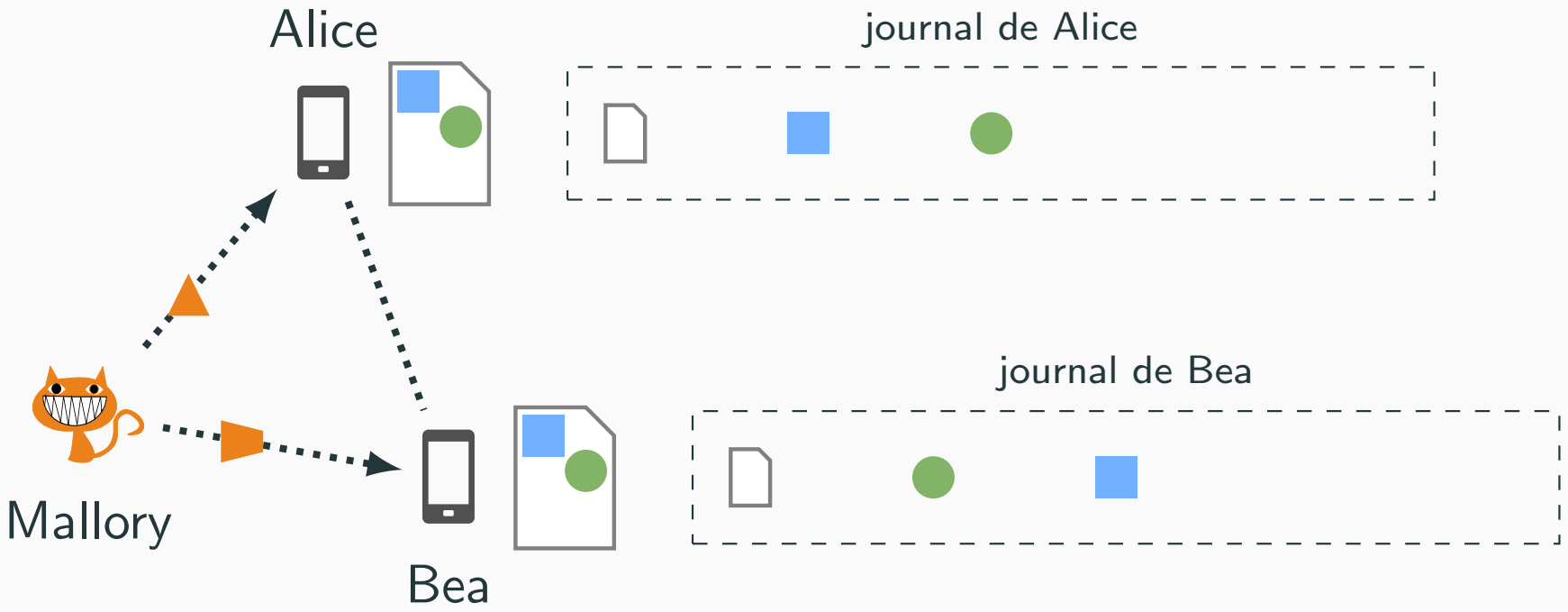
- un pair enregistre dans son journal les **modifications qu'il intègre**
- arrangement horizontal = ordre d'ajout des modifications

Journaux de modifications



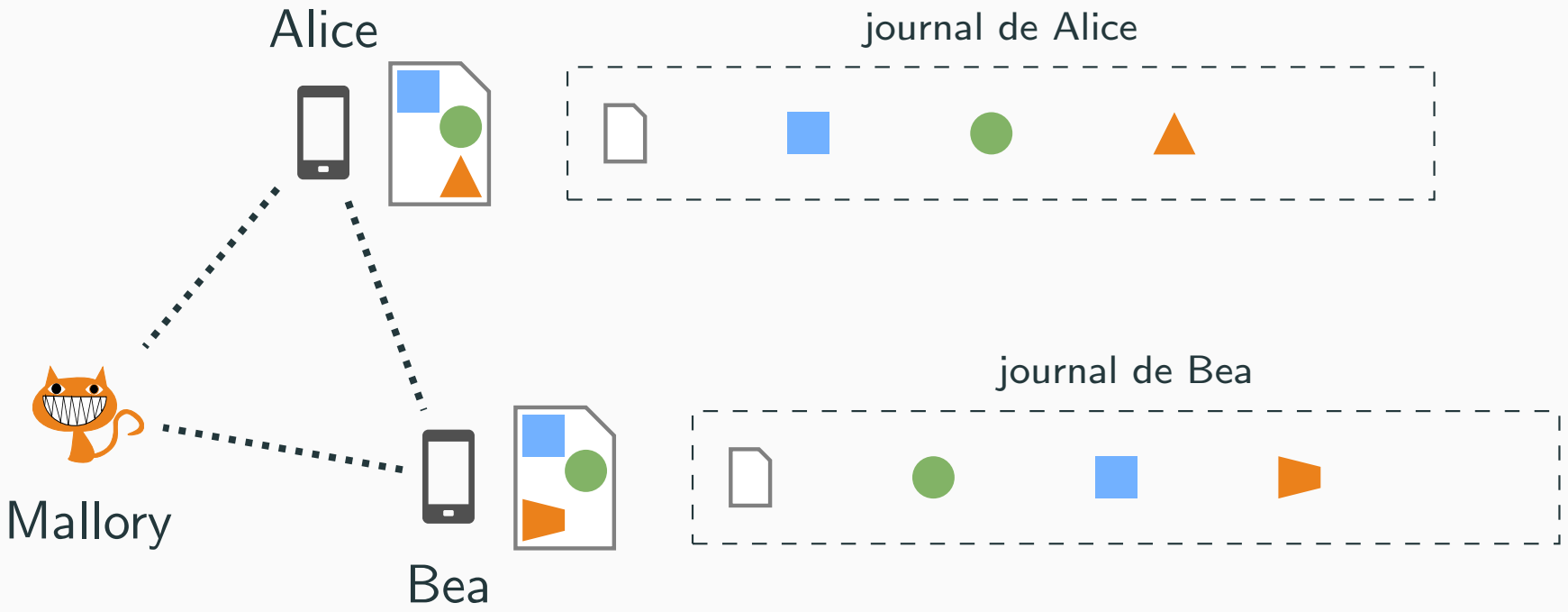
- un pair enregistre dans son journal les **modifications qu'il intègre**
- arrangement horizontal = ordre d'ajout des modifications

Journaux de modifications



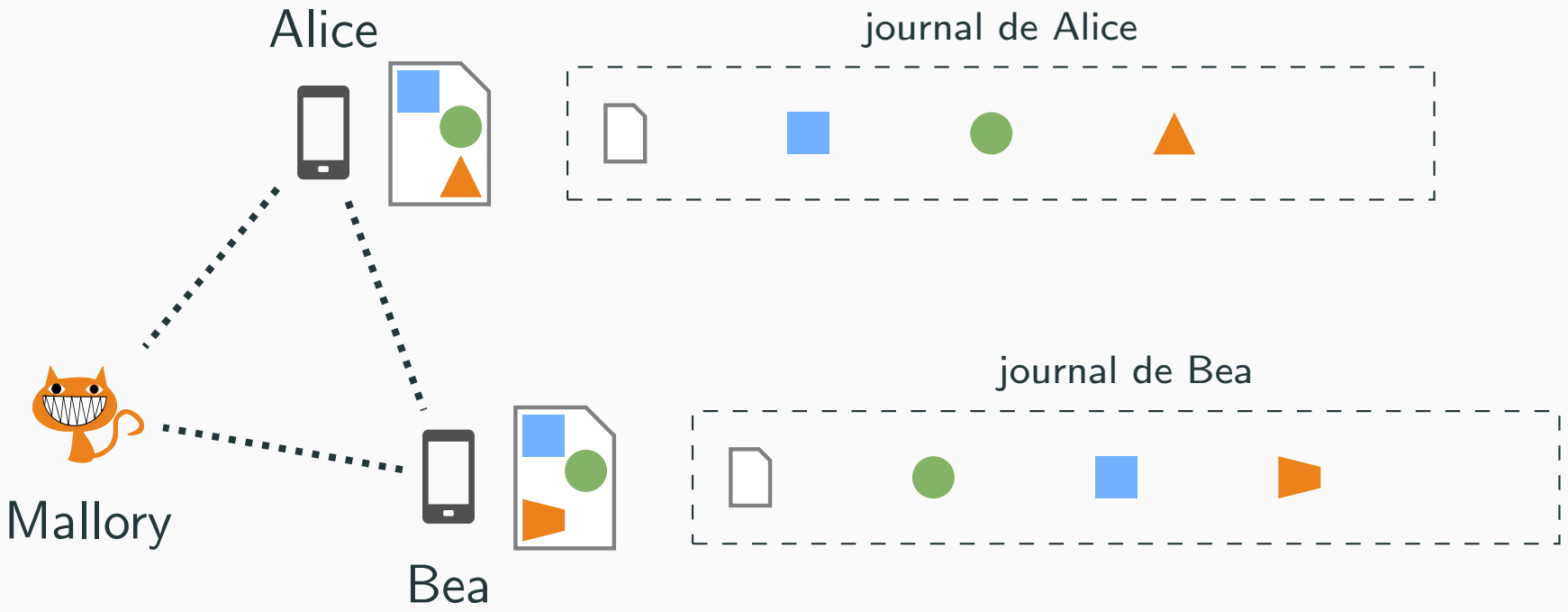
- un pair enregistre dans son journal les **modifications qu'il intègre**
- arrangement horizontal = ordre d'ajout des modifications

Journaux de modifications



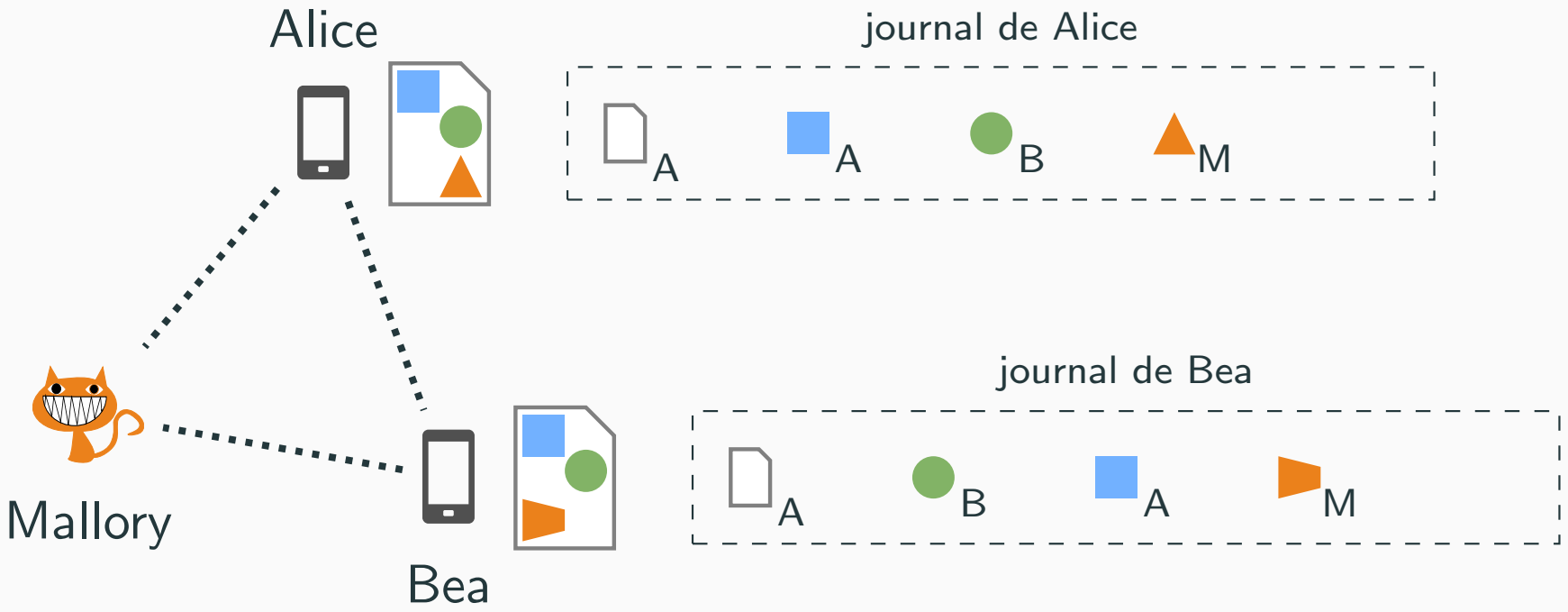
- un pair enregistre dans son journal les **modifications qu'il intègre**
- arrangement horizontal = ordre d'ajout des modifications

Journaux infalsifiables

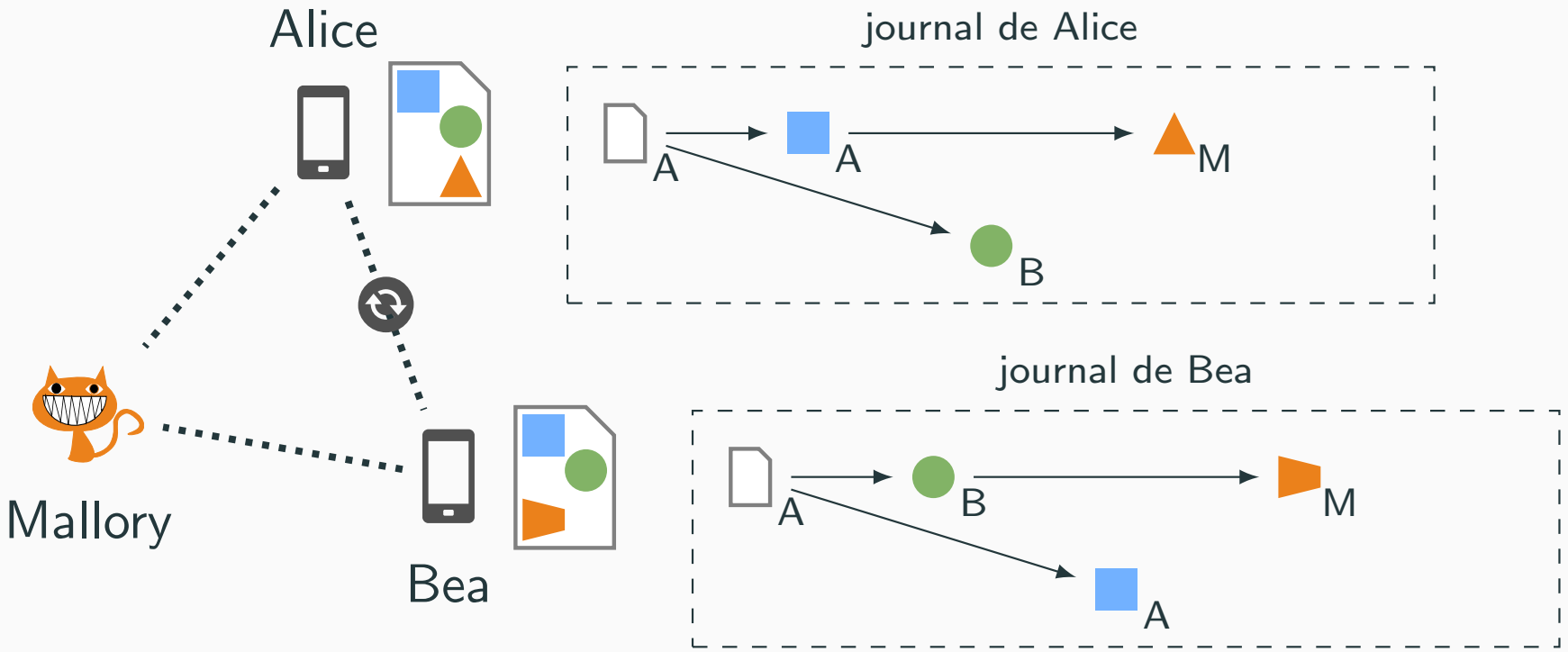


- **modifications signées** par leur auteur·ice
- **déclaration infalsifiable des dépendances** à l'aide de **hashes**
  - un pair malintentionné peut déclarer des dépendances arbitraires
- journaux avec **mêmes modifications** $\Longrightarrow$ **copies convergentes**

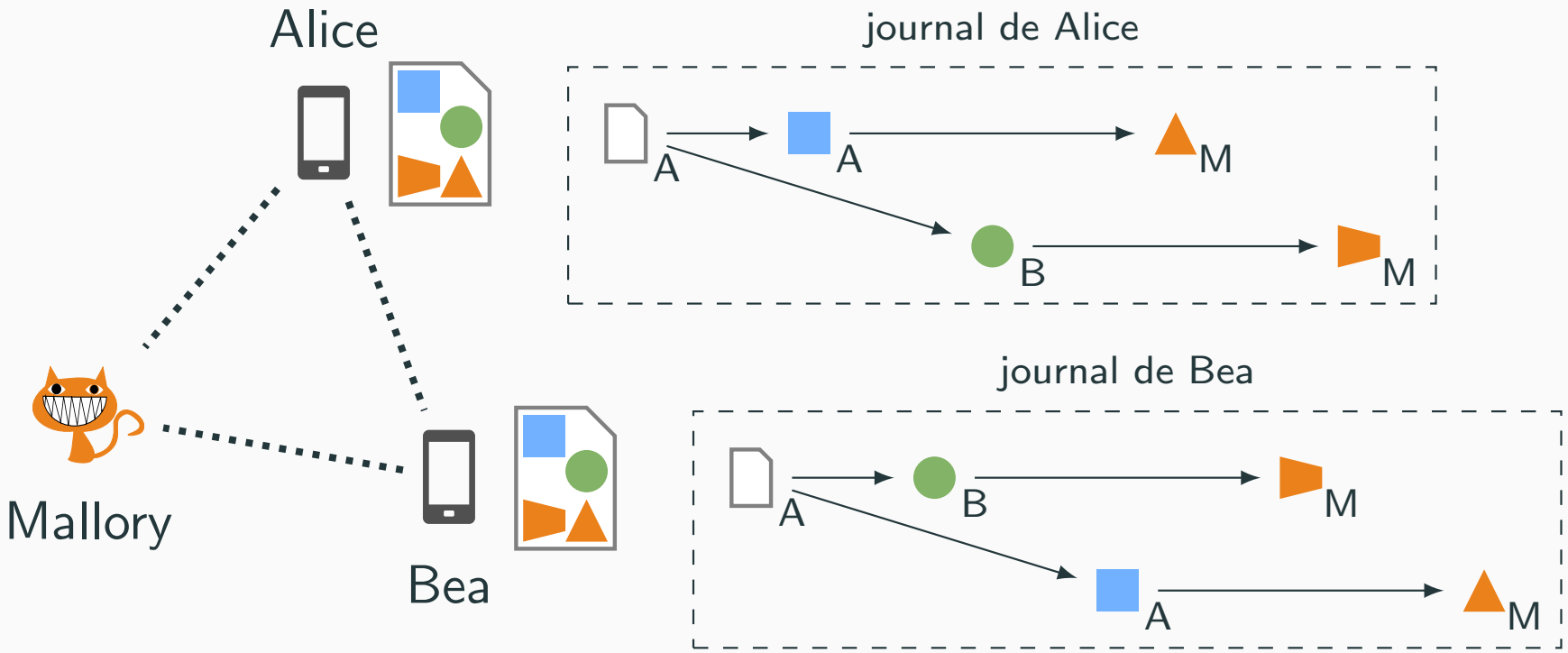
Journaux infalsifiables



- **modifications signées** par leur auteur·ice
- **déclaration infalsifiable des dépendances** à l'aide de **hashes**
  - un pair malintentionné peut déclarer des dépendances arbitraires
- journaux avec **mêmes modifications** $\Longrightarrow$ **copies convergentes**

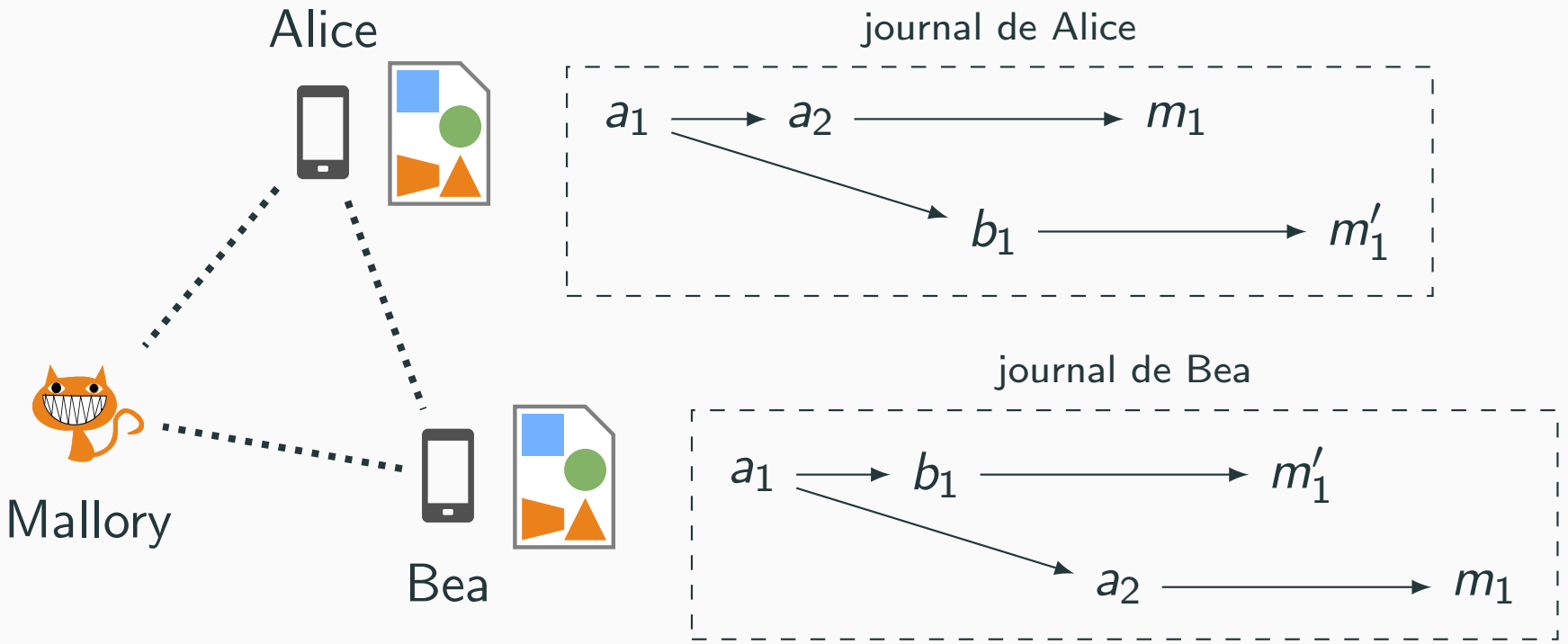


- **modifications signées** par leur auteur·ice
- **déclaration infalsifiable des dépendances** à l'aide de **hashes**
  - un pair malintentionné peut déclarer des dépendances arbitraires
- journaux avec **mêmes modifications** $\Longrightarrow$ **copies convergentes**

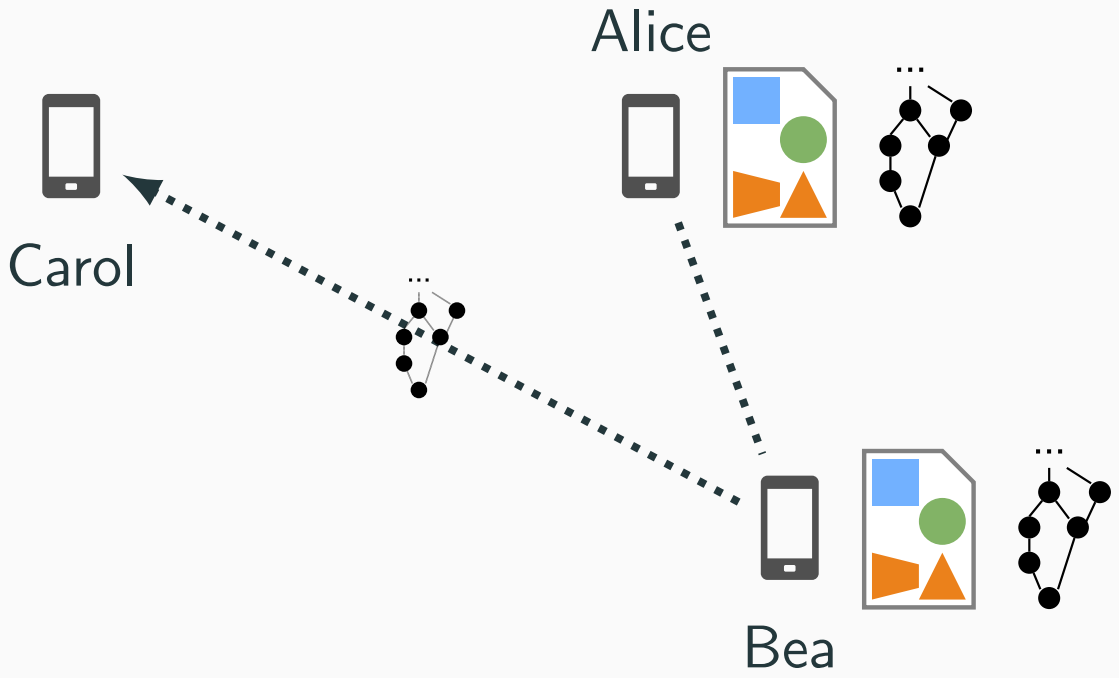


- **modifications signées** par leur auteur·ice
- **déclaration infalsifiable des dépendances** à l'aide de **hashes**
  - un pair malintentionné peut déclarer des dépendances arbitraires
- journaux avec **mêmes modifications** $\Longrightarrow$ **copies convergentes**

Journaux infalsifiables



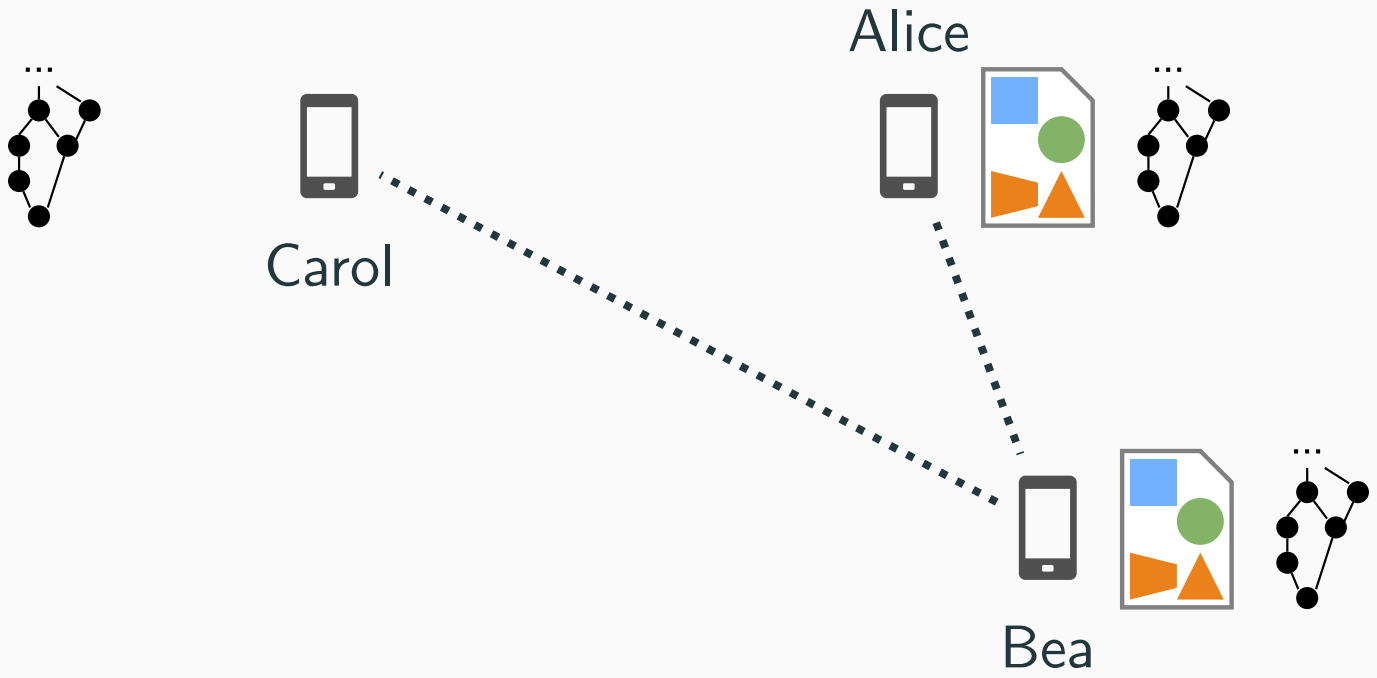
- **modifications signées** par leur auteur·ice
- **déclaration infalsifiable des dépendances** à l'aide de **hashes**
  - un pair malintentionné peut déclarer des dépendances arbitraires
- journaux avec **mêmes modifications** $\Longrightarrow$ **copies convergentes**



mémoire : les pairs conservent leur journal

communication : un nouveau pair récupère un journal

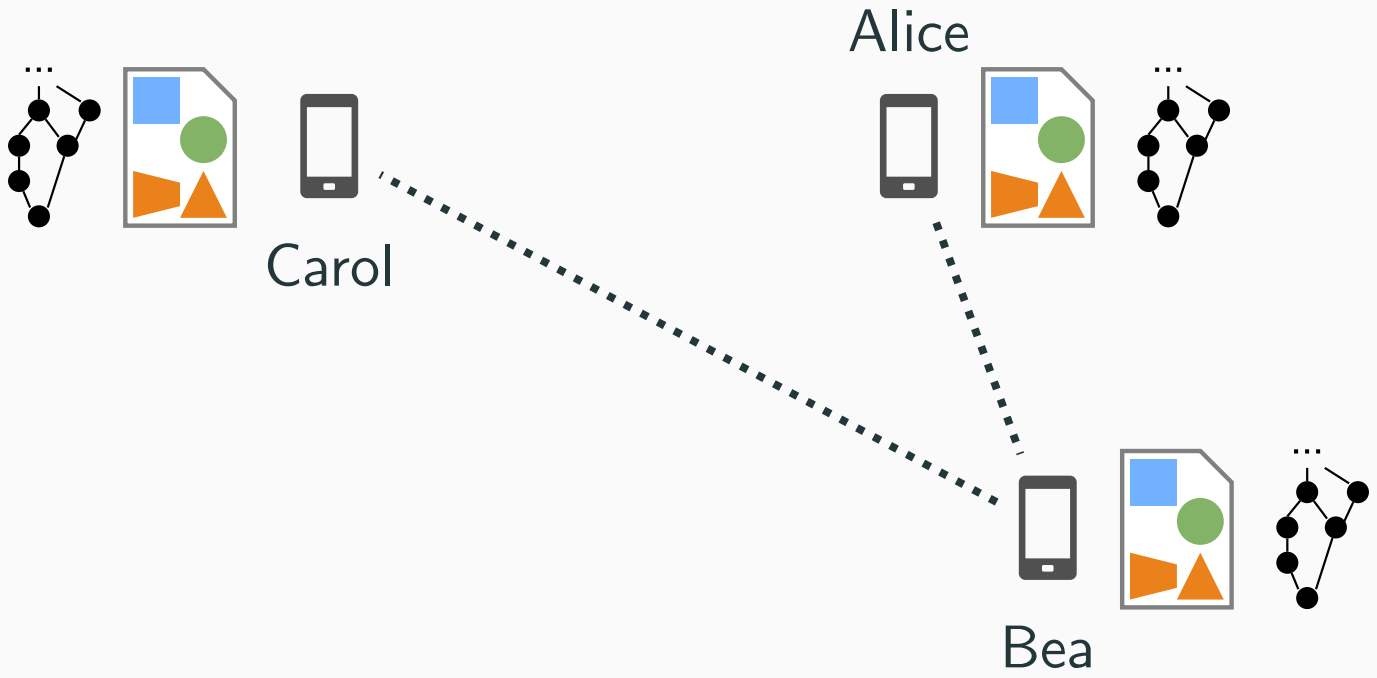
calcul : un nouveau pair intègre chaque modification



mémoire : les pairs conservent leur journal

communication : un nouveau pair récupère un journal

calcul : un nouveau pair intègre chaque modification

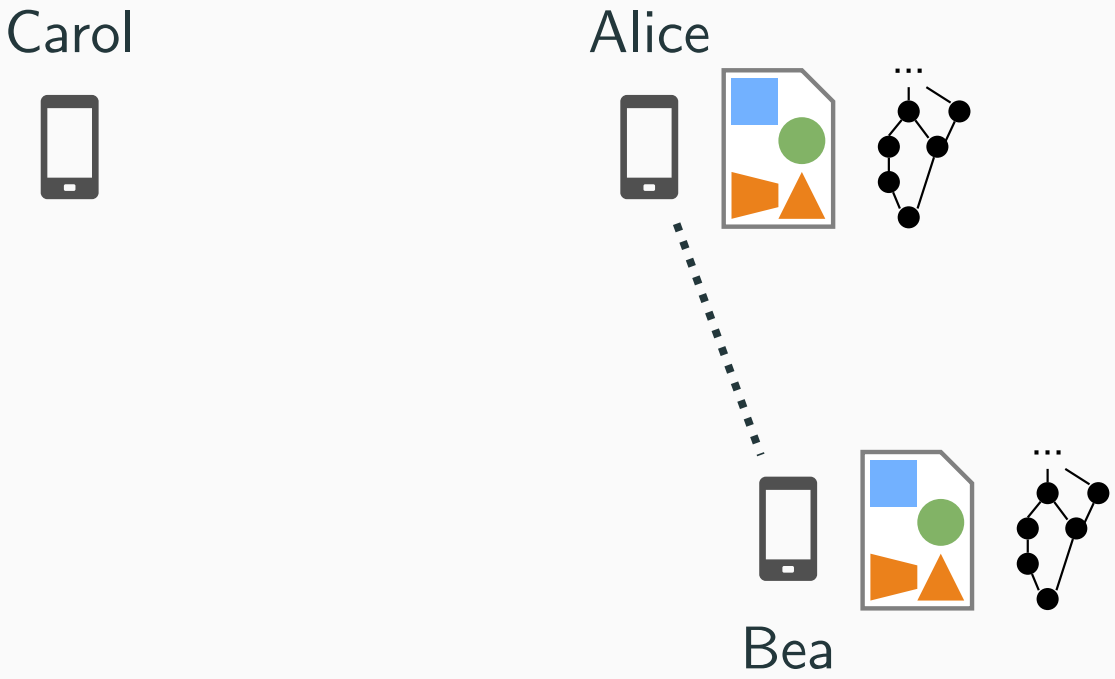


mémoire : les pairs conservent leur journal

communication : un nouveau pair récupère un journal

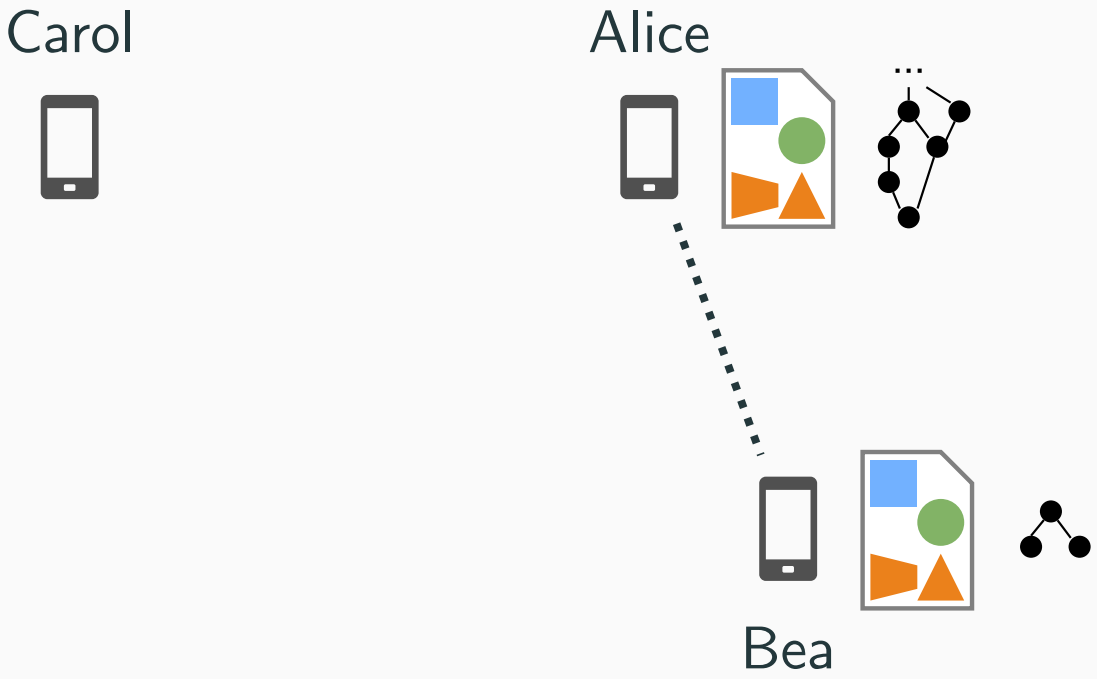
calcul : un nouveau pair intègre chaque modification

Que font les applications de collaboration ?



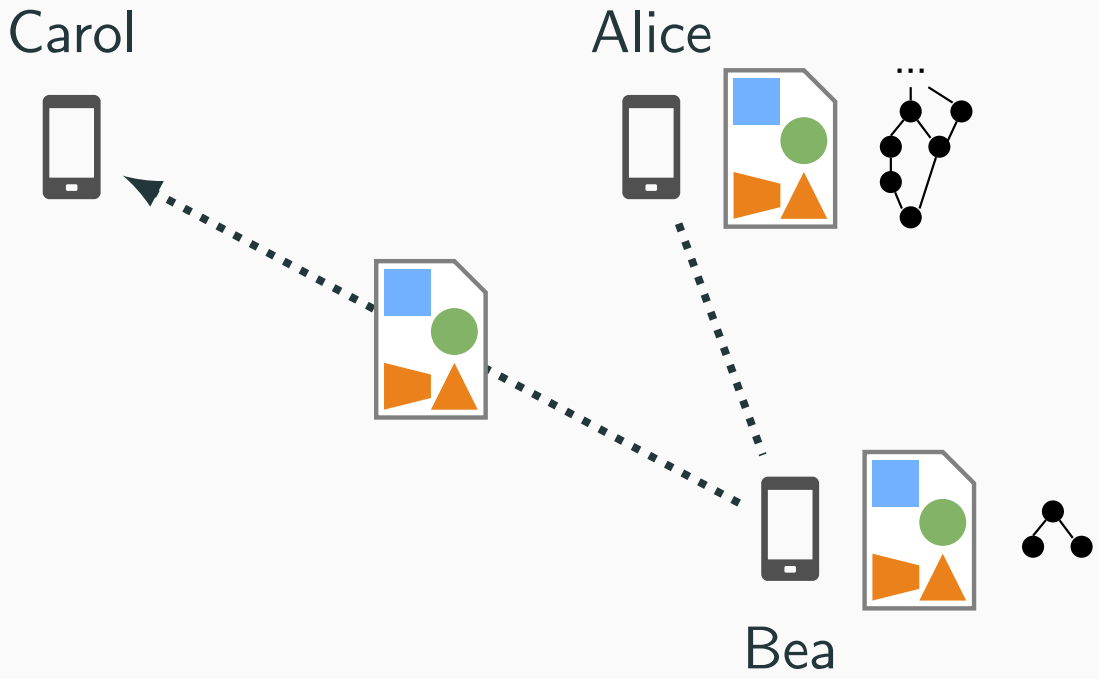
- hypothèse : taille contenu $<$ taille journal
- **troncature** du journal **sans coordination**
- **transmission de l'état de la copie** aux nouveaux pairs

Que font les applications de collaboration ?



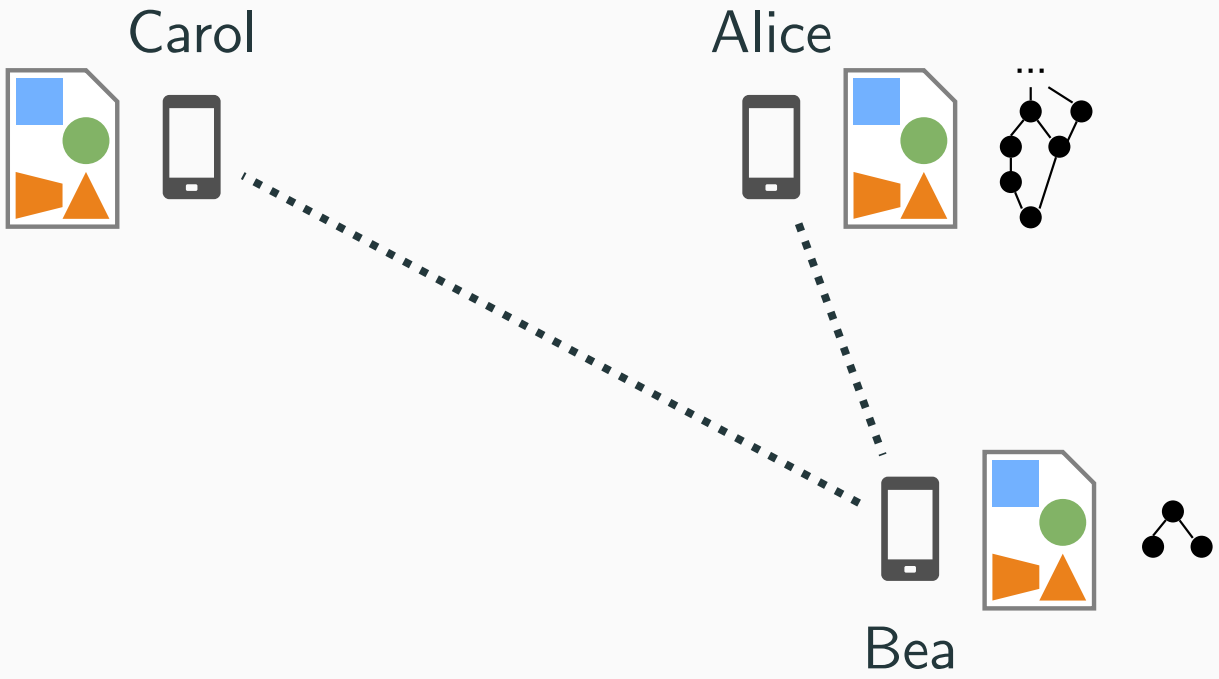
- hypothèse : taille contenu $<$ taille journal
- **troncature** du journal **sans coordination**
- **transmission de l'état de la copie** aux nouveaux pairs

Que font les applications de collaboration ?



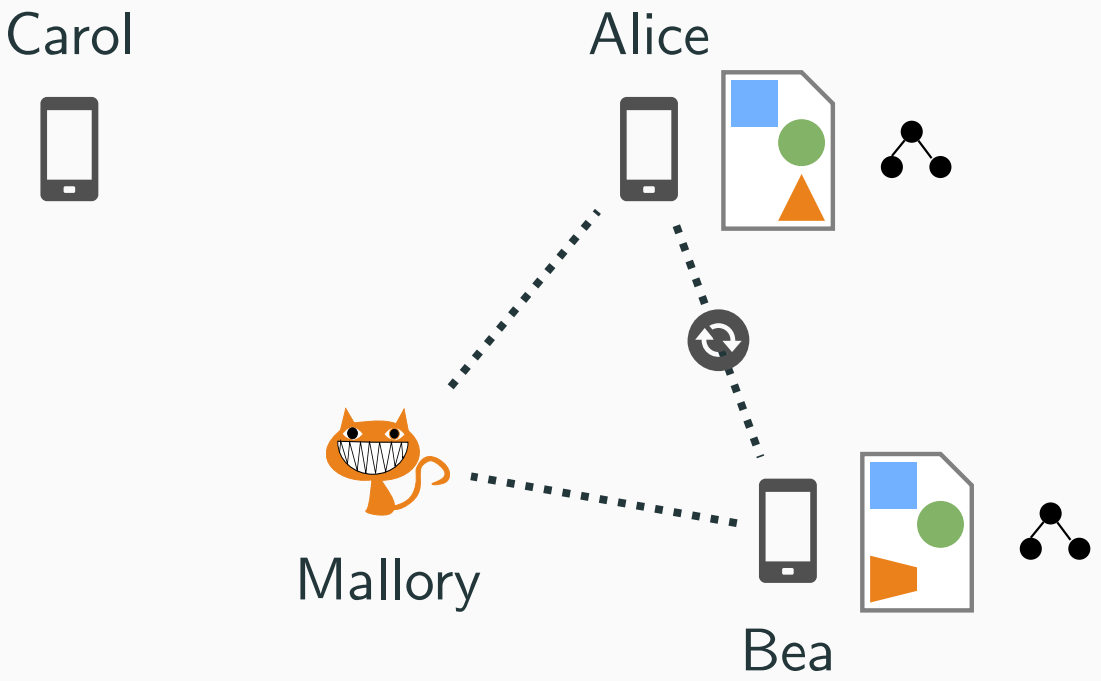
- hypothèse : taille contenu $<$ taille journal
- **troncature** du journal **sans coordination**
- **transmission de l'état de la copie** aux nouveaux pairs

Que font les applications de collaboration ?



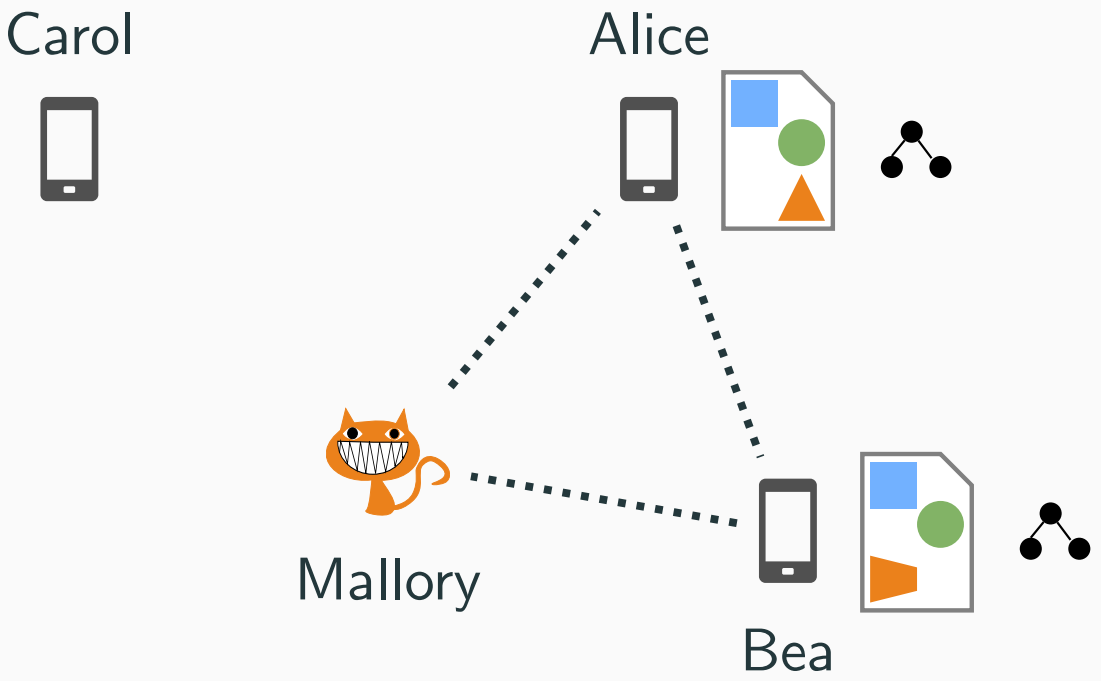
- hypothèse : taille contenu $<$ taille journal
- **troncature** du journal **sans coordination**
- **transmission de l'état de la copie** aux nouveaux pairs

Diffucltés en présence de pairs malintentionnés



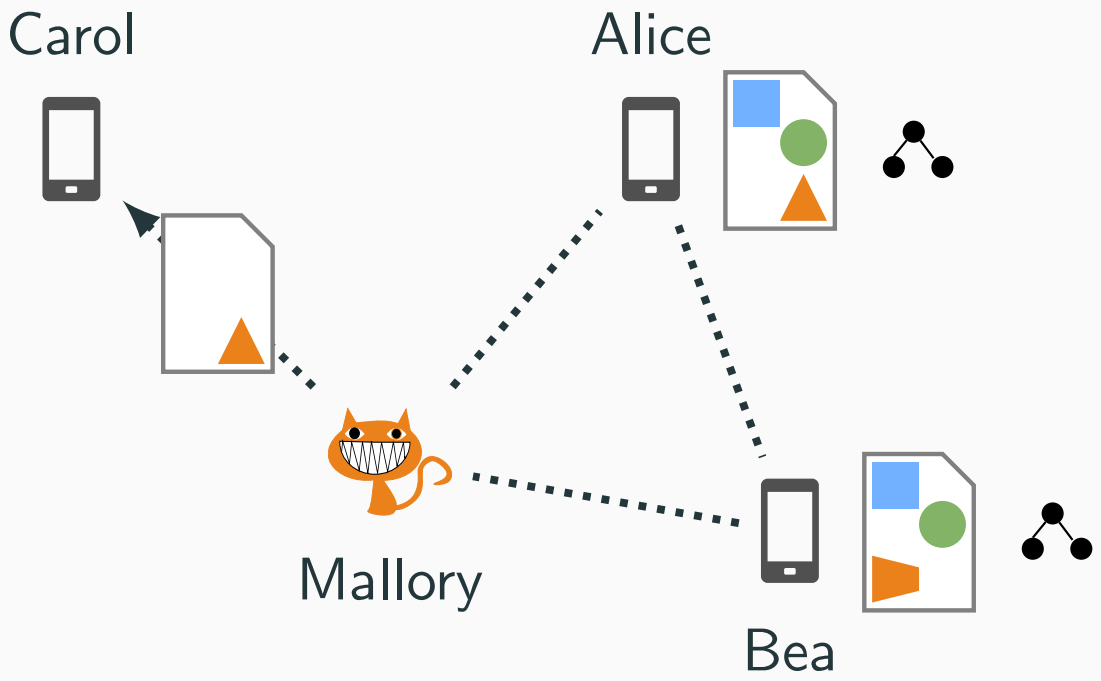
- le journal **ne peut pas être tronqué arbitrairement**
  - la détection d'équivoque pourrait être compromise
- un pair malintentionné peut transmettre un **état falsifié**

Diffucltés en présence de pairs malintentionnés



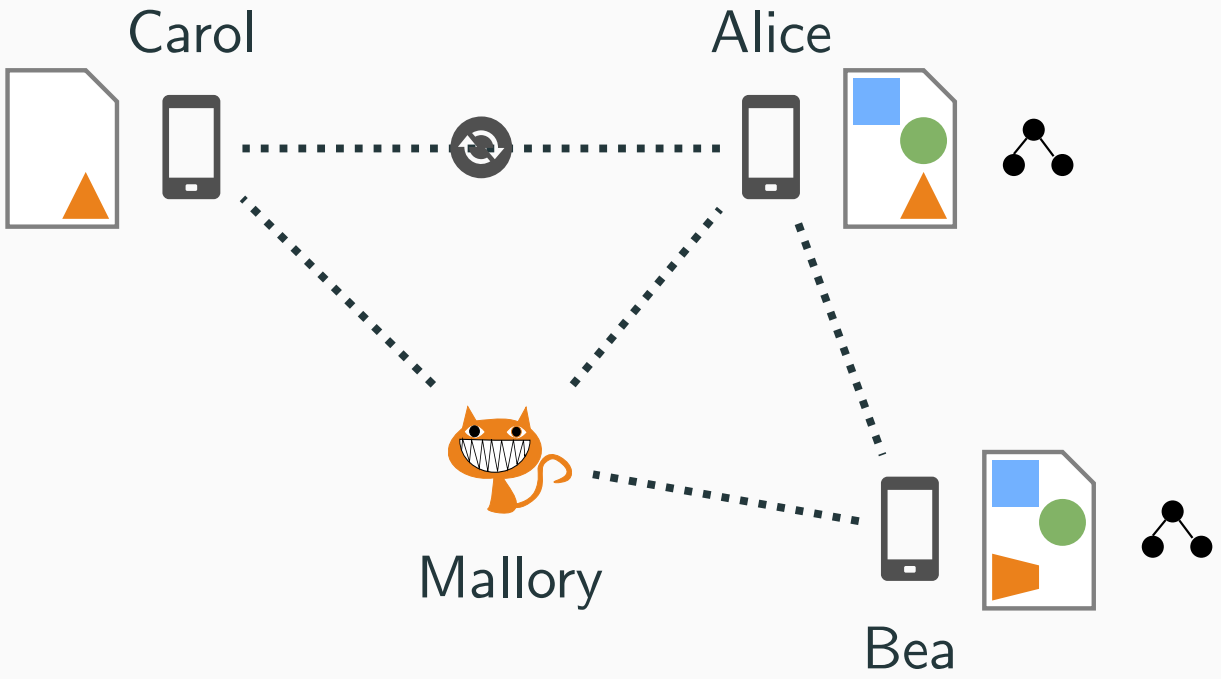
- le journal **ne peut pas être tronqué arbitrairement**
  - la détection d'équivoque pourrait être compromise
- un pair malintentionné peut transmettre un **état falsifié**

Diffucltés en présence de pairs malintentionnés



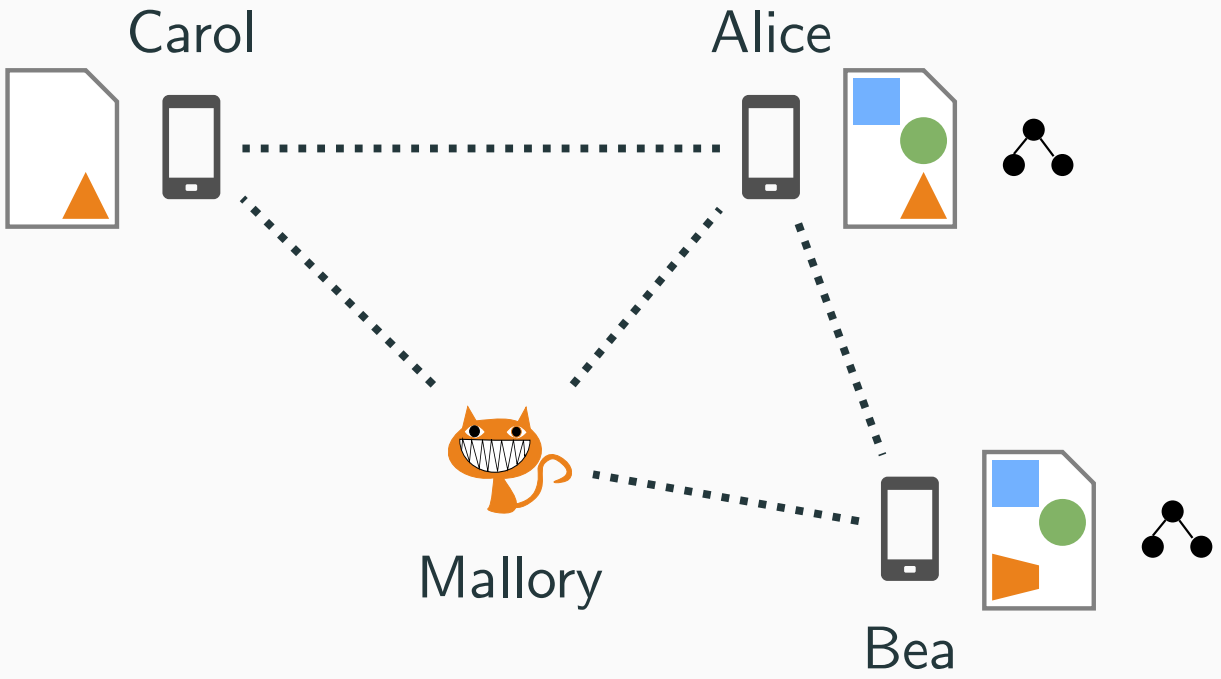
- le journal **ne peut pas être tronqué arbitrairement**
  - la détection d'équivoque pourrait être compromise
- un pair malintentionné peut transmettre un **état falsifié**

Diffucltés en présence de pairs malintentionnés

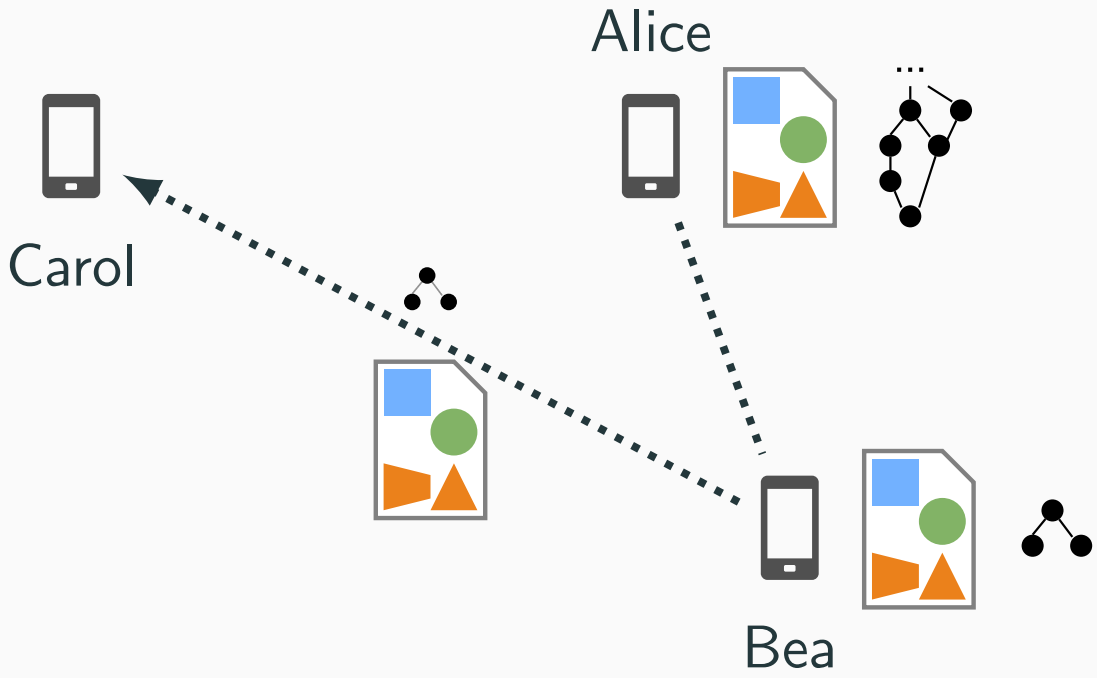


- le journal **ne peut pas être tronqué arbitrairement**
  - la détection d'équivoque pourrait être compromise
- un pair malintentionné peut transmettre un **état falsifié**

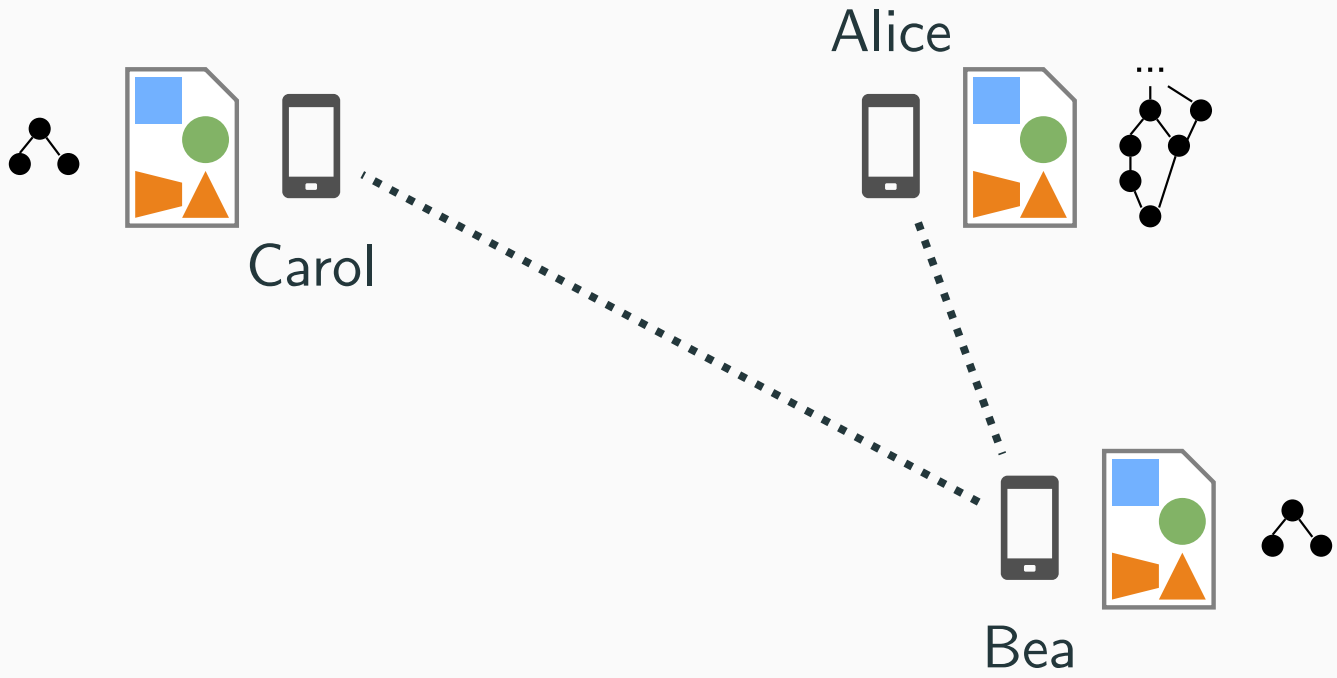
Diffucltés en présence de pairs malintentionnés



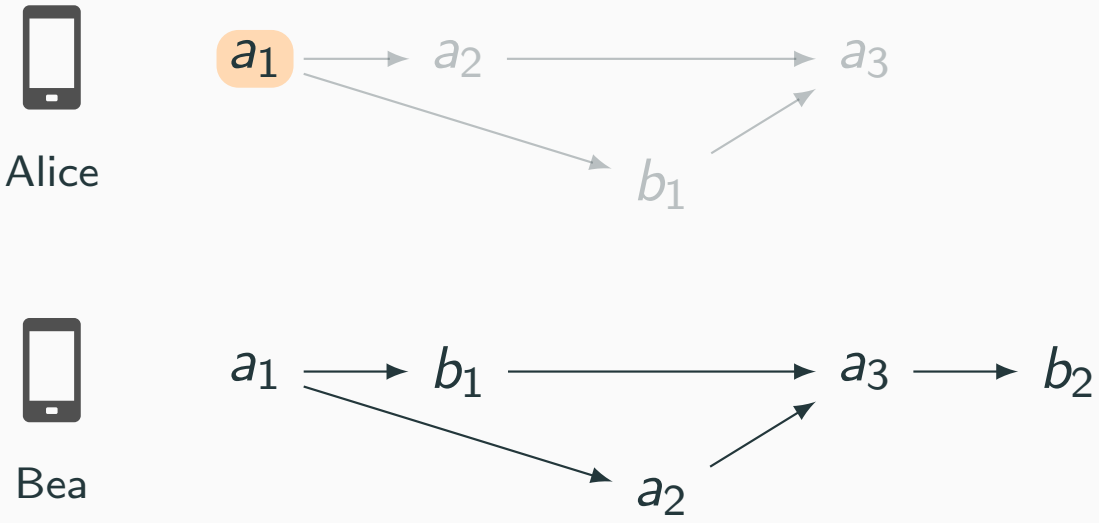
- le journal **ne peut pas être tronqué arbitrairement**
  - la détection d'équivoque pourrait être compromise
- un pair malintentionné peut transmettre un **état falsifié**



- **troncature sans coordination** reposant sur le concept de **stabilité**
  - conservation de modifications pour la détection d'équivoques
- transmission d'un **état authentifiable** et journal tronqué
  1. vérifie la cohérence du journal tronqué
  2. authentifie le contenu à partir du journal

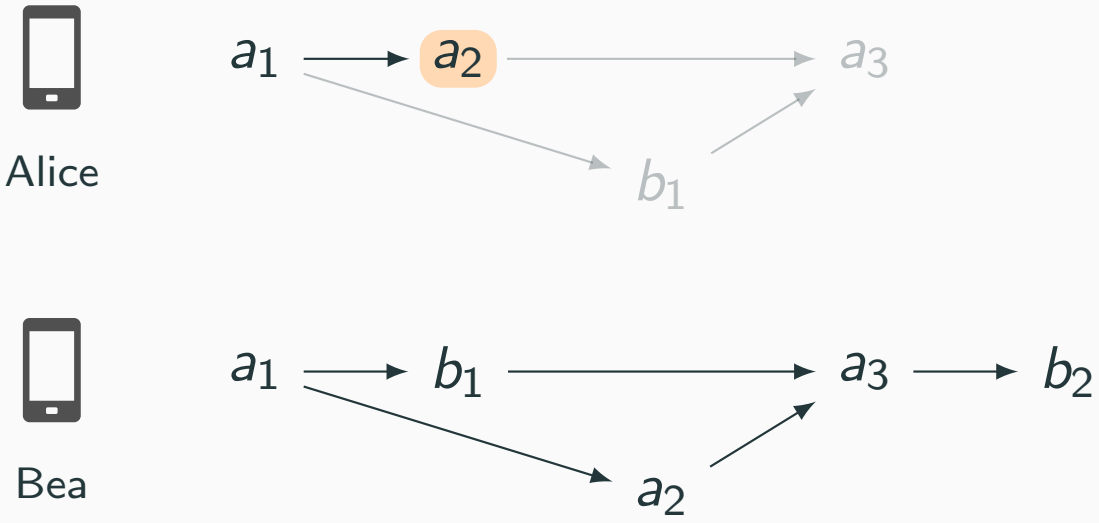


- **troncature sans coordination** reposant sur le concept de **stabilité**
  - conservation de modifications pour la détection d'équivoques
- transmission d'un **état authentifiable** et journal tronqué
  1. vérifie la cohérence du journal tronqué
  2. authentifie le contenu à partir du journal



Invariants

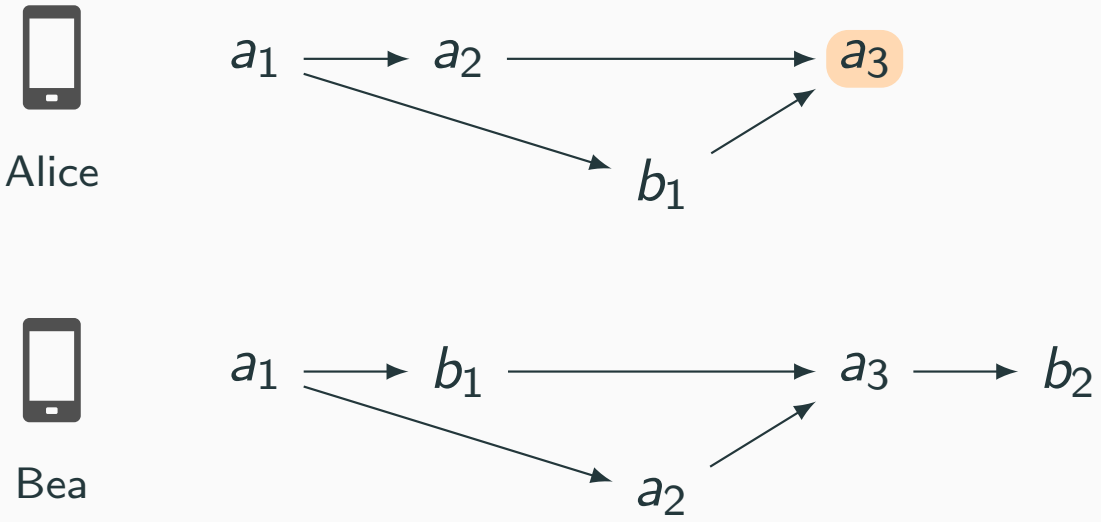
1. une modification du possesseur du journal dépend des précédentes
2. ordre linéaire des modification de chaque pair



Invariants

1. une modification du possesseur du journal dépend des précédentes
2. ordre linéaire des modification de chaque pair

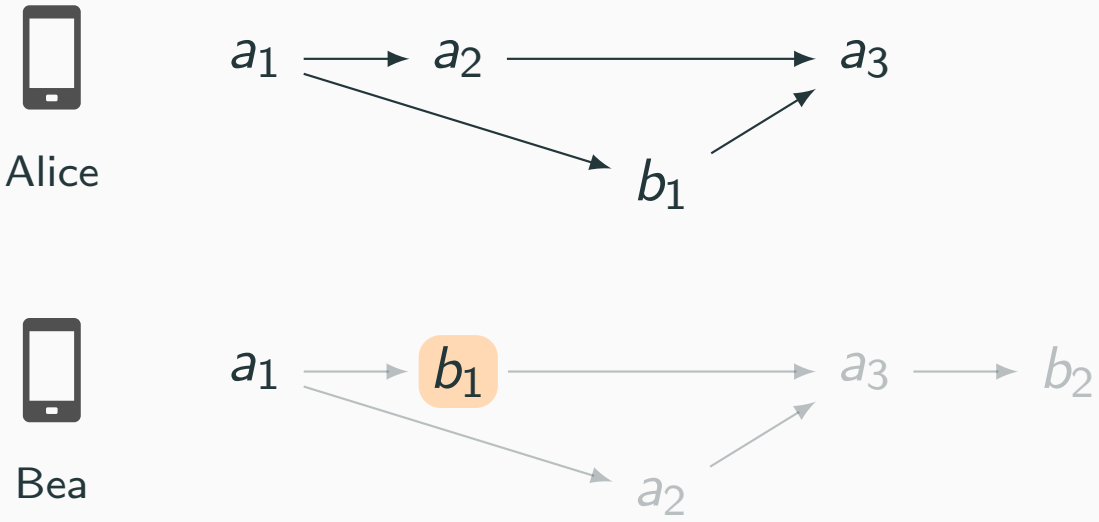
Journal causal



Invariants

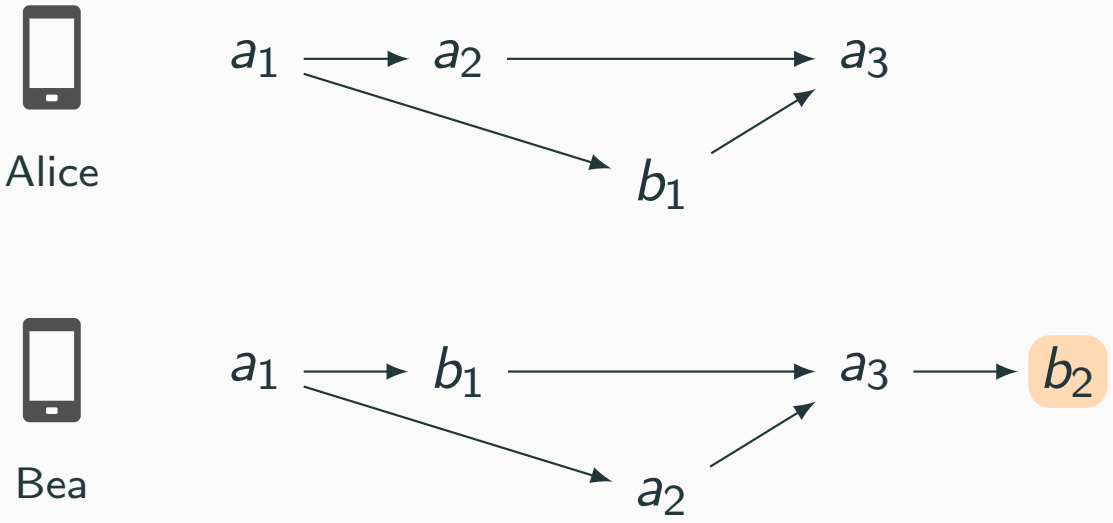
1. une modification du possesseur du journal dépend des précédentes
2. ordre linéaire des modification de chaque pair

Journal causal



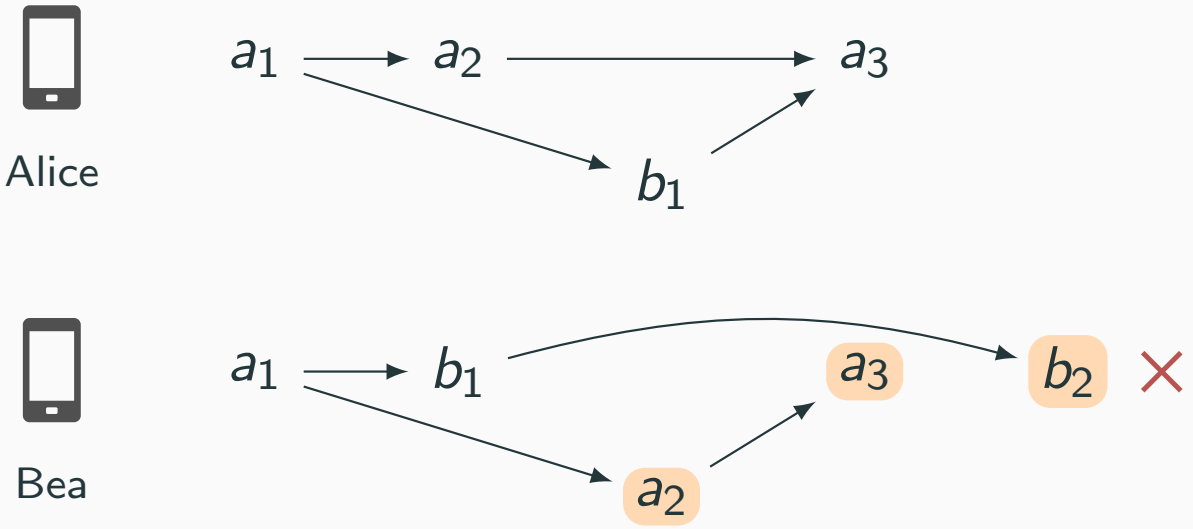
Invariants

1. une modification du possesseur du journal dépend des précédentes
2. ordre linéaire des modification de chaque pair



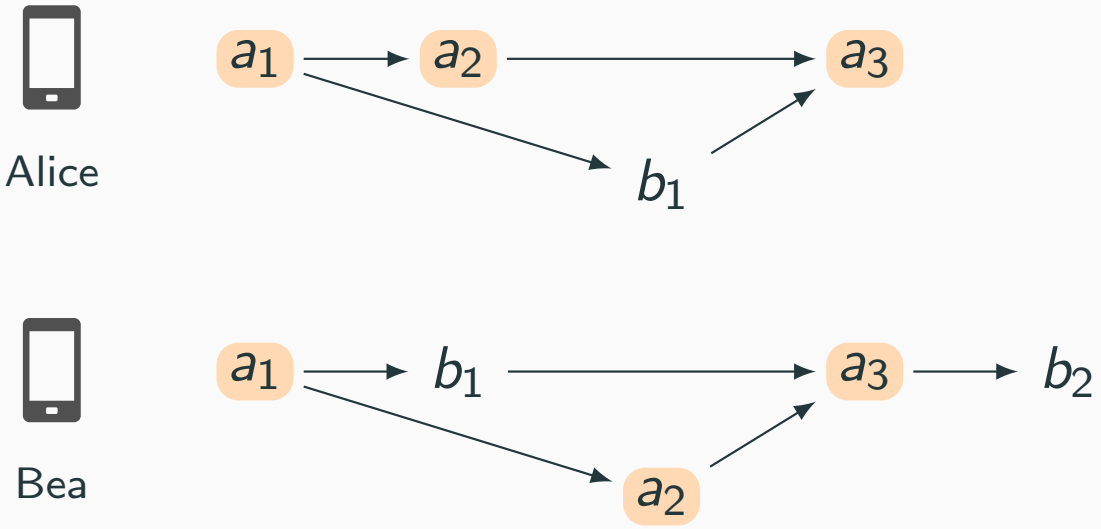
Invariants

1. une modification du possesseur du journal dépend des précédentes
2. ordre linéaire des modification de chaque pair



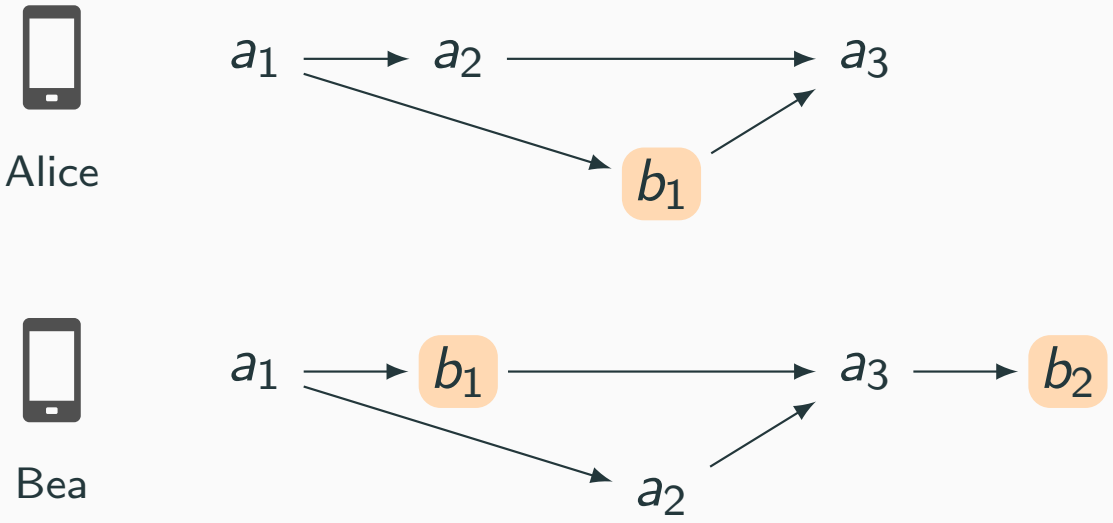
Invariants

1. une modification du possesseur du journal dépend des précédentes
2. ordre linéaire des modification de chaque pair



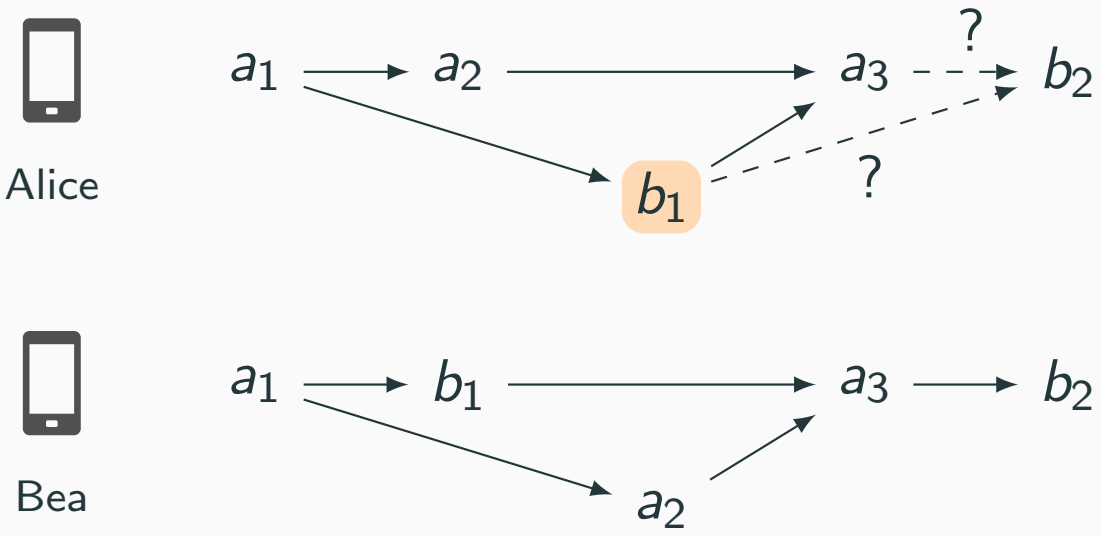
Invariants

1. une modification du possesseur du journal dépend des précédentes
2. ordre linéaire des modification de chaque pair



Invariants

1. une modification du possesseur du journal dépend des précédentes
2. ordre linéaire des modification de chaque pair

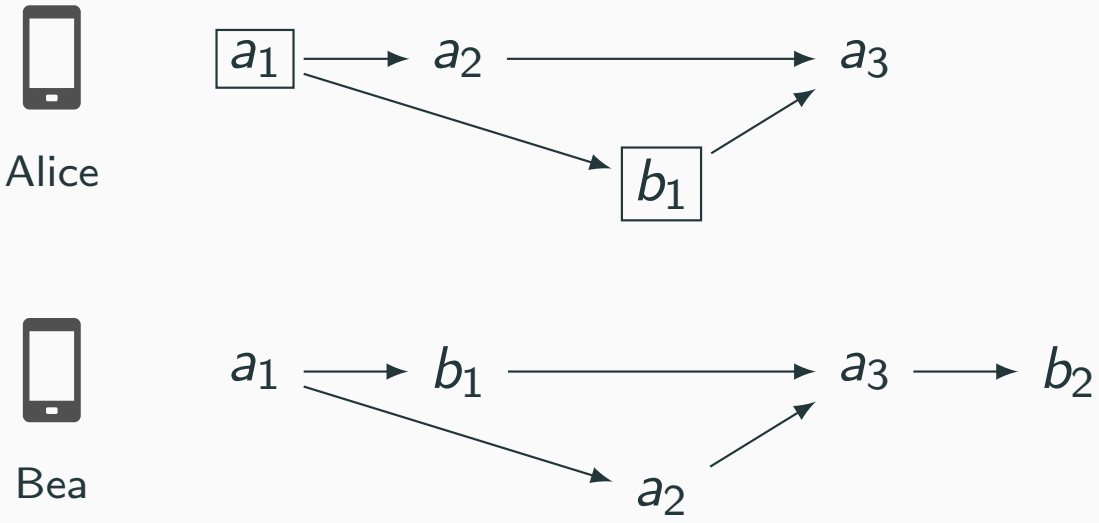


Définition

modification stable $\iff$ dépendance de toute modification **ultérieurement acceptée** dans le journal.

Théorème

modification stable $\iff$ *observée par chaque pair au sein du journal.*

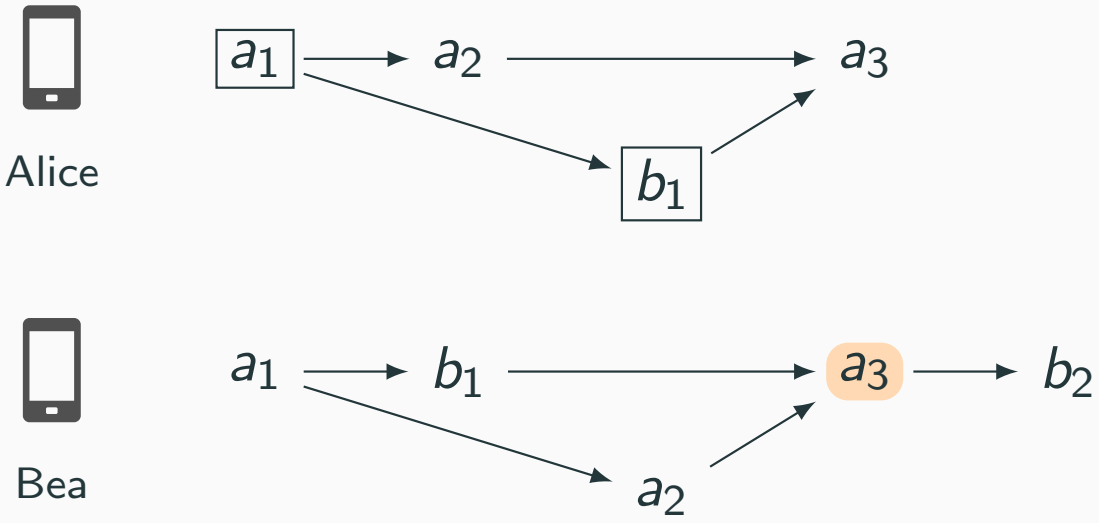


Définition

modification stable $\iff$ dépendance de toute modification **ultérieurement acceptée** dans le journal.

Théorème

modification stable $\iff$ *observée par chaque pair au sein du journal.*

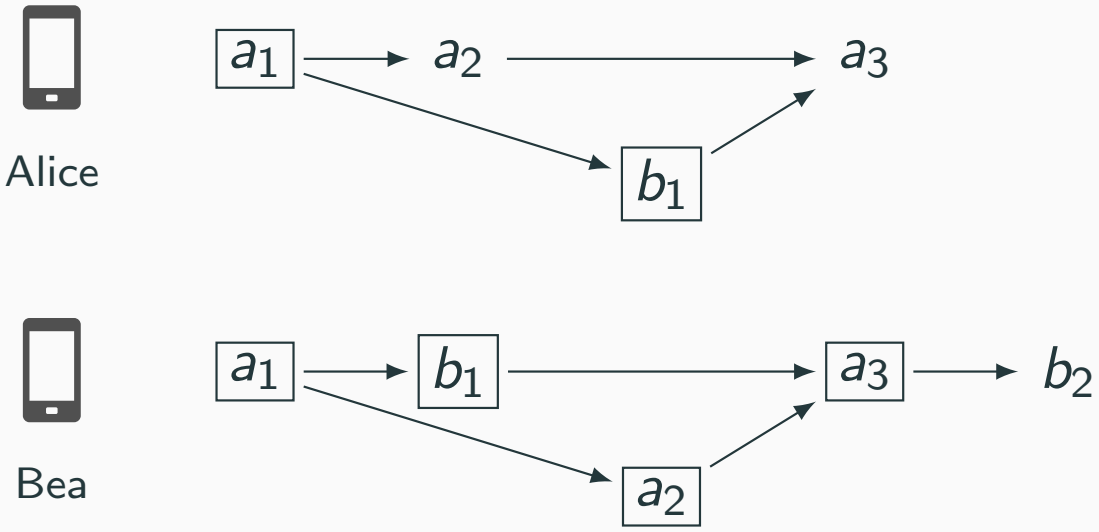


Définition

modification stable $\iff$ dépendance de toute modification **ultérieurement acceptée** dans le journal.

Théorème

modification stable $\iff$ *observée par chaque pair au sein du journal.*

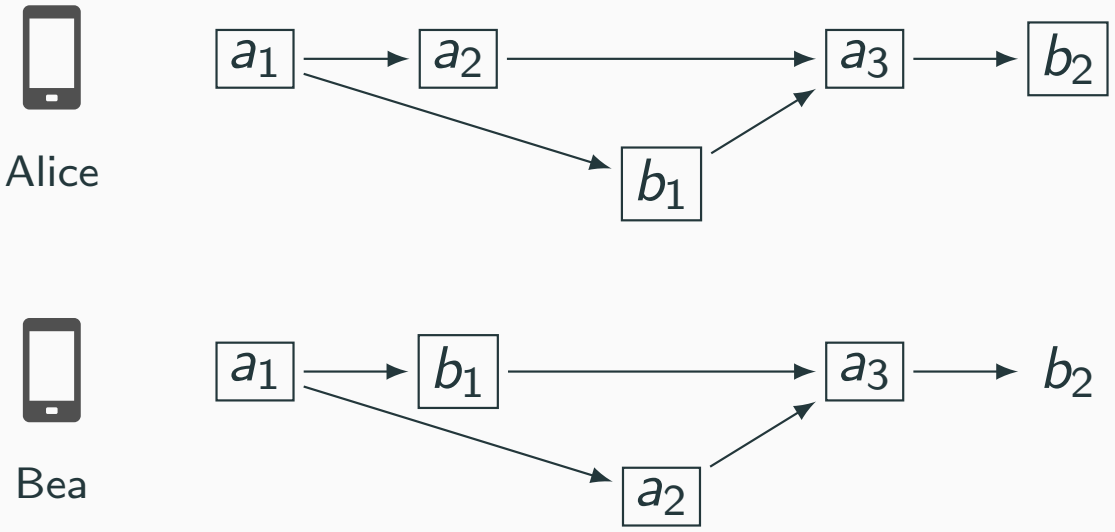


Définition

modification stable $\iff$ dépendance de toute modification **ultérieurement acceptée** dans le journal.

Théorème

modification stable $\iff$ *observée par chaque pair au sein du journal.*

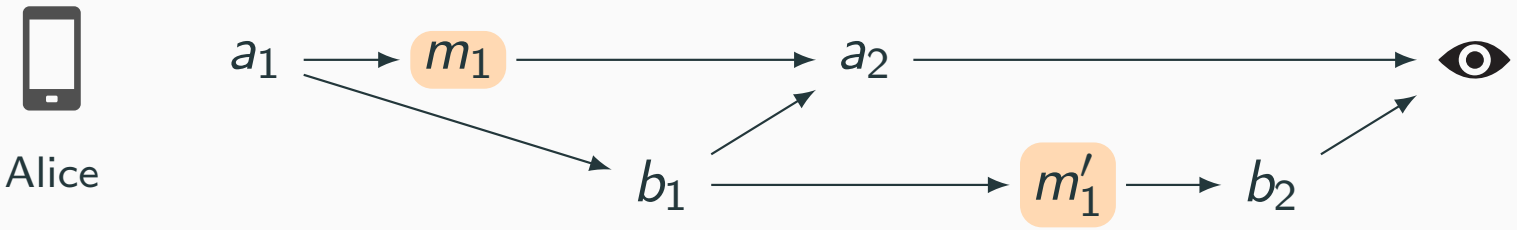


Définition

modification stable $\iff$ dépendance de toute modification **ultérieurement acceptée** dans le journal.

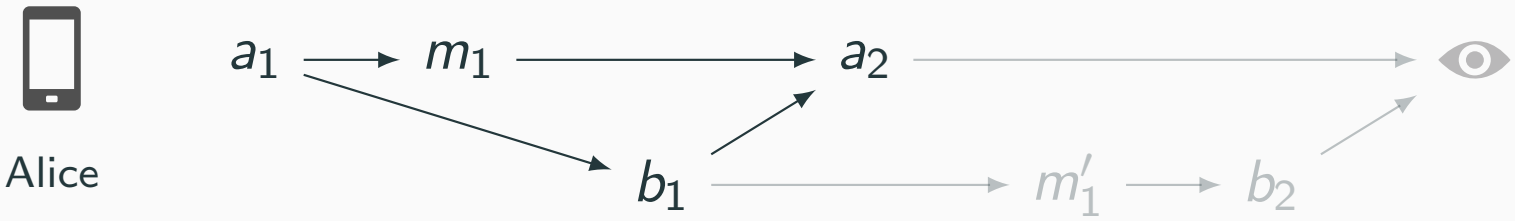
Théorème

modification stable $\iff$ *observée par chaque pair au sein du journal.*



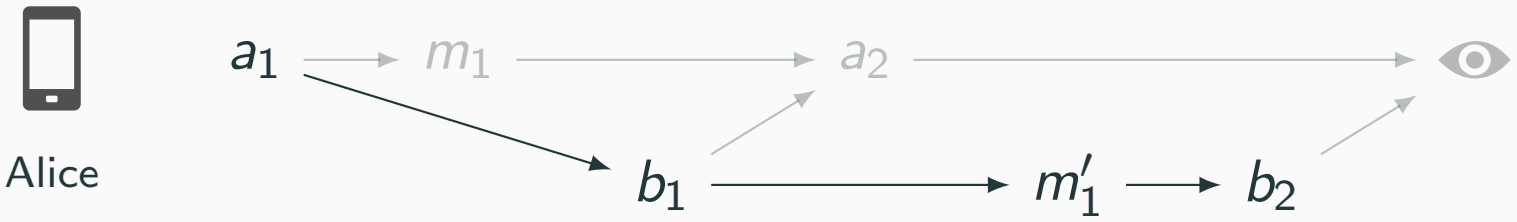
Invariants

1. une modification du possesseur du journal dépend des précédentes
2. ordre linéaire des modification de chaque pair honnête
3. une modification **dépend directement** de modifications **présumées linéaires**
   - une modification non-linéaire est acceptée indirectement
   - le possesseur du jorunal ne peut plus accepter directement de modifications d'un pair reconnu malintentionné



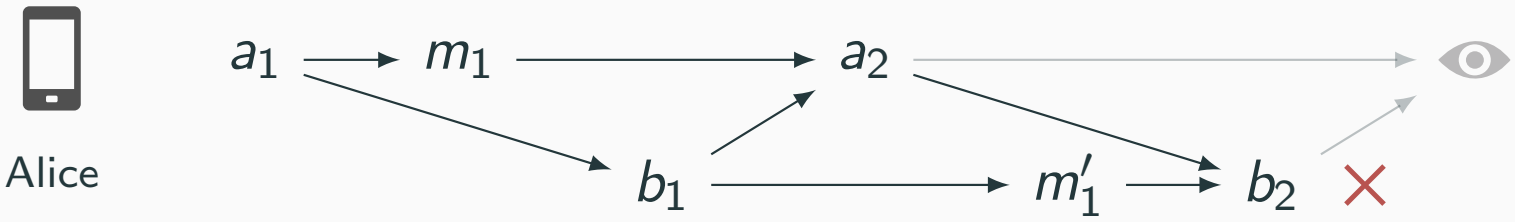
Invariants

1. une modification du possesseur du journal dépend des précédentes
2. ordre linéaire des modification de chaque pair honnête
3. une modification **dépend directement** de modifications **présumées linéaires**
   - une modification non-linéaire est acceptée indirectement
   - le possesseur du jorunal ne peut plus accepter directement de modifications d'un pair reconnu malintentionné



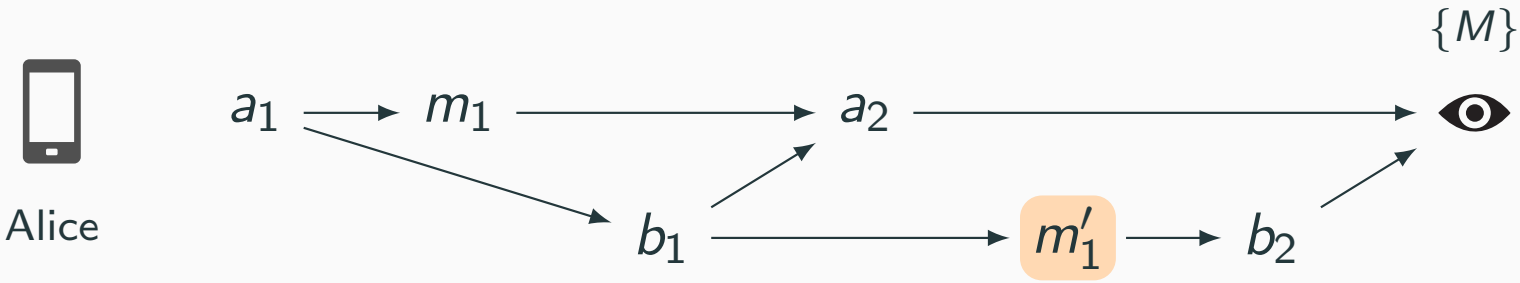
Invariants

1. une modification du possesseur du journal dépend des précédentes
2. ordre linéaire des modification de chaque pair honnête
3. une modification **dépend directement** de modifications **présumées linéaires**
   - une modification non-linéaire est acceptée indirectement
   - le possesseur du jorunal ne peut plus accepter directement de modifications d'un pair reconnu malintentionné



Invariants

1. une modification du possesseur du journal dépend des précédentes
2. ordre linéaire des modification de chaque pair honnête
3. une modification **dépend directement** de modifications **présumées linéaires**
   - une modification non-linéaire est acceptée indirectement
   - le possesseur du jorunal ne peut plus accepter directement de modifications d'un pair reconnu malintentionné



Invariants

1. une modification du possesseur du journal dépend des précédentes
2. ordre linéaire des modification de chaque pair honnête
3. une modification **dépend directement** de modifications **présumées linéaires**
   - une modification non-linéaire est acceptée indirectement
   - le possesseur du jorunal ne peut plus accepter directement de modifications d'un pair reconnu malintentionné

	environnement honnête	environnement malintentionné
groupe statique	journal causal [1] stabilité causale [2]	journal VFJC [3] **stabilité VFJC**
groupe dynamique	**journal DynCausale** **stabilité DynCausale**	**journal DynVFJC** **stabilité DynVFJC**

groupe statique = groupe avec un ensemble défini de pairs

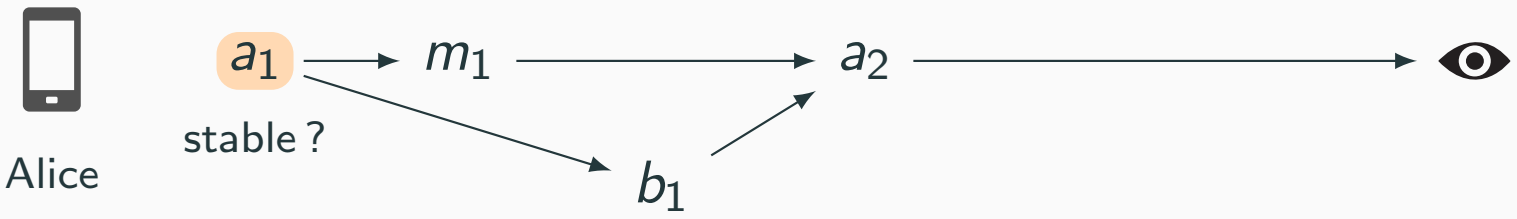
groupe dynamique = groupe dont la composition évolue

---

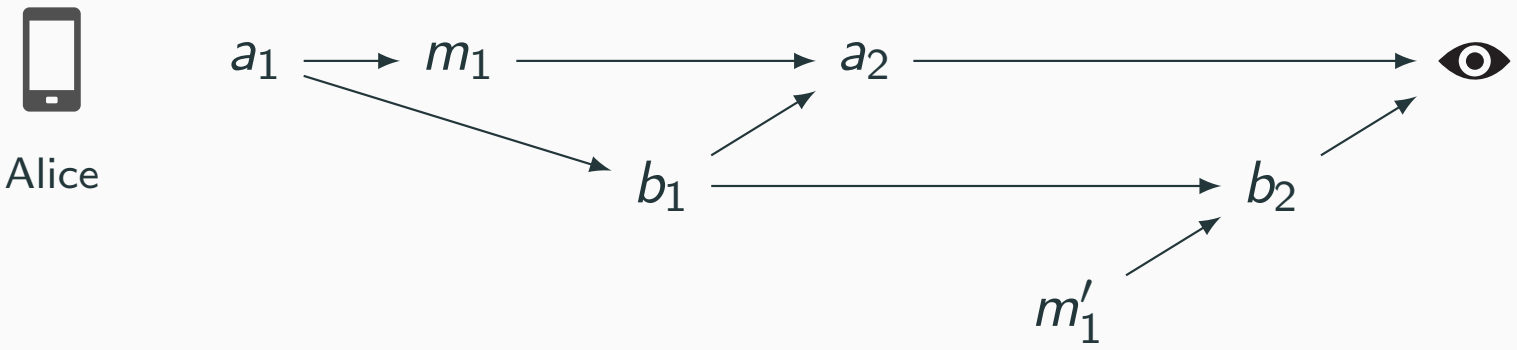
[1]. Hutto et al., "Slow Memory : Weakening Consistency to Enchance Concurrency in Distributed Shared Memories".

[2]. Baquero et al., "Pure Operation-Based Replicated Data Types".

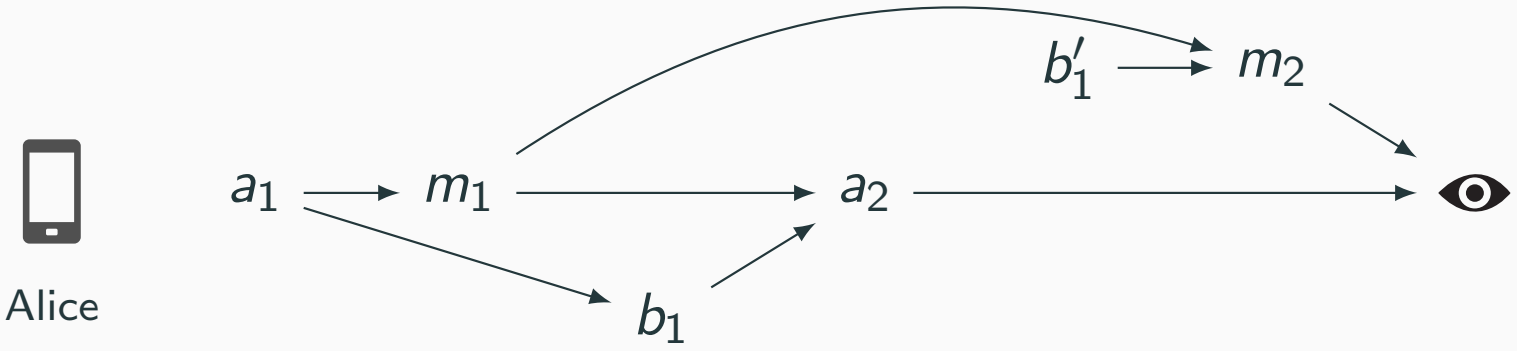
[3]. Mahajan et al., *Consistency, Availability, and Convergence*.



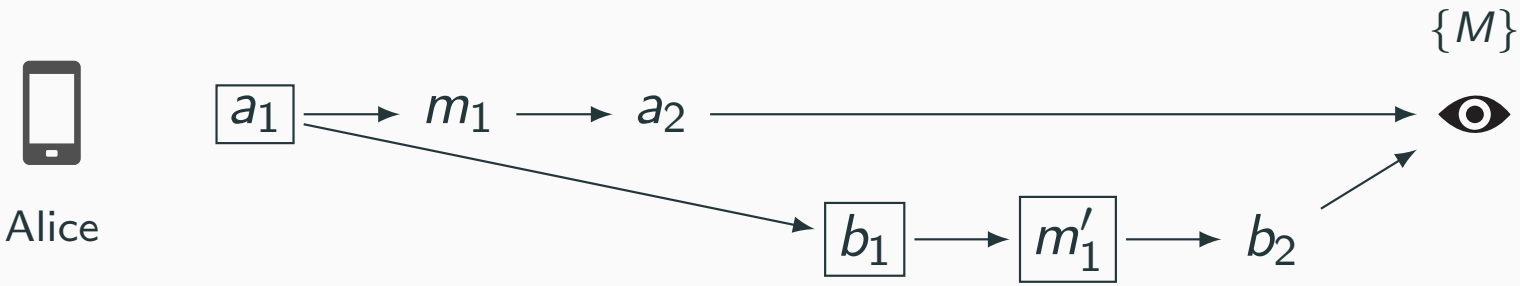
- le possesseur du journal **présume honnêtes** les autres pairs
  - il s'attend à ce qu'ils soient **potentiellement malintentionnés**



- le possesseur du journal **présume honnêtes** les autres pairs
  - il s'attend à ce qu'ils soient **potentiellement malintentionnés**



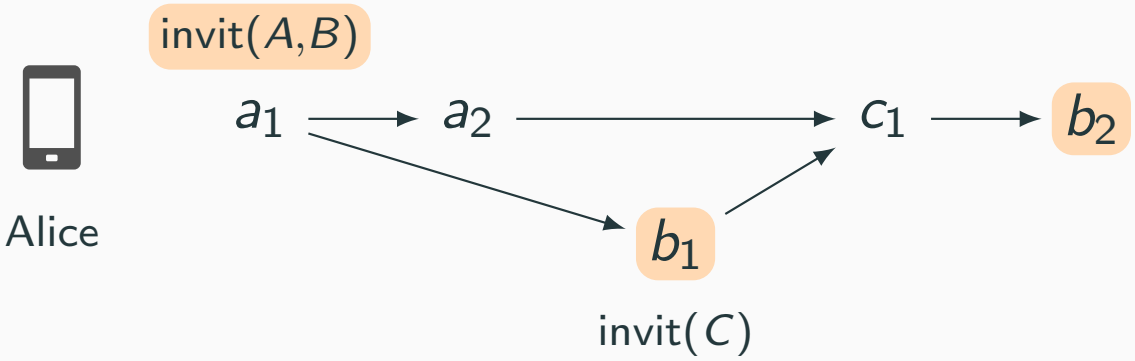
- le possesseur du journal **présume honnêtes** les autres pairs
  - il s'attend à ce qu'ils soient **potentiellement malintentionnés**



permutation	chaîne de dépendance
M, B, A	$m'_1, b_2,$ 👁
B, M, A	$b_2,$ 👁, 👁

Théorème

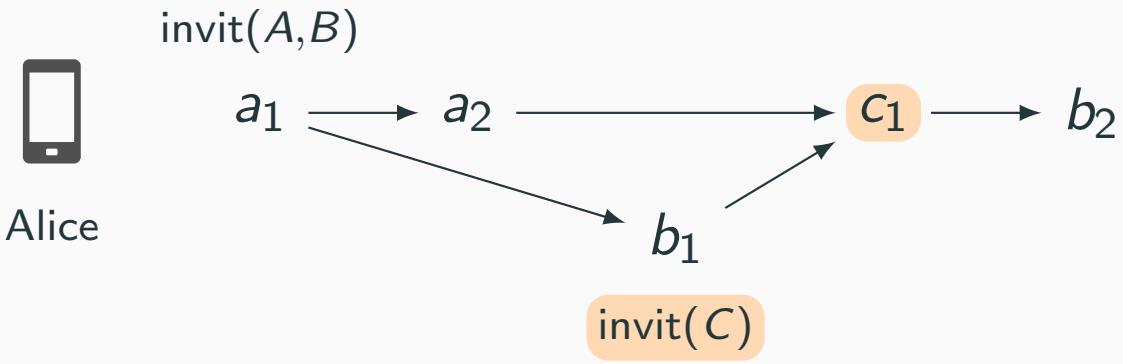
modification x stable $\iff$ *pour toute permutation* $p_1, \ldots, p_N$ *des pairs avec* p_N *le possesseur du journal, il existe une chaîne de dépendances* $x, x_1, \ldots, x_N$ *telle que* x_i *a pour auteur ou reconnaît malintentionné* p_i.



- suivi de la composition du groupe via les invitations
- les modifications d'un pair dépendent des dépendances de son invit.

Invariants

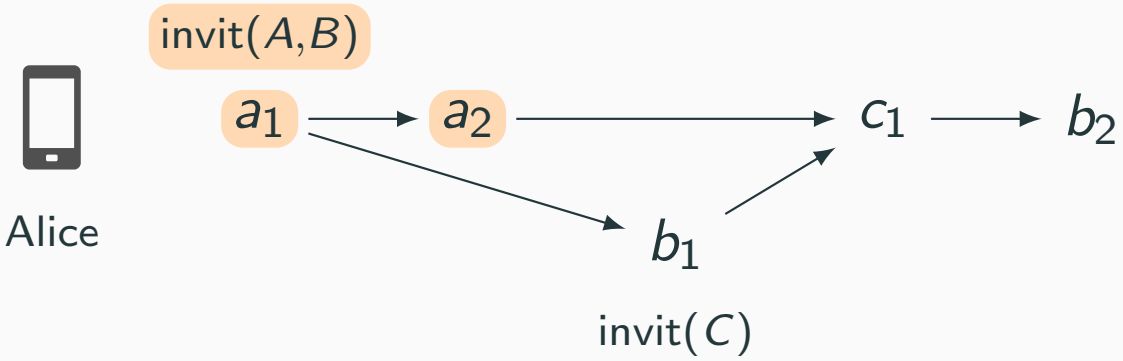
1. une modification du possesseur du journal dépend des précédentes
2. ordre linéaire des modifications de chaque pair
3. les modifications d'un pair dépendent de son invitation



- suivi de la composition du groupe via les invitations
- les modifications d'un pair dépendent des dépendances de son invit.

Invariants

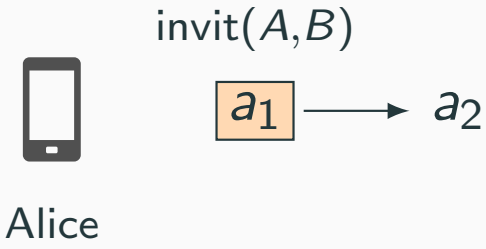
1. une modification du possesseur du journal dépend des précédentes
2. ordre linéaire des modifications de chaque pair
3. les modifications d'un pair dépendent de son invitation



- suivi de la composition du groupe via les invitations
- les modifications d'un pair dépendent des dépendances de son invit.

Invariants

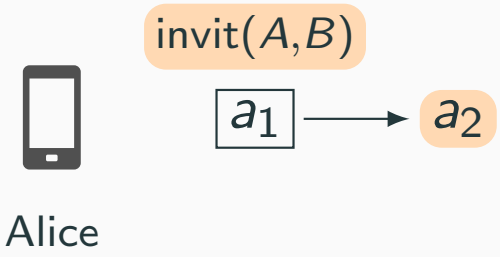
1. une modification du possesseur du journal dépend des précédentes
2. ordre linéaire des modifications de chaque pair
3. les modifications d'un pair dépendent de son invitation



- observateurs honnêtes requis =
  - \+ pairs invités dans les dépendances
  - \+ pairs invités en concurrence

Théorème

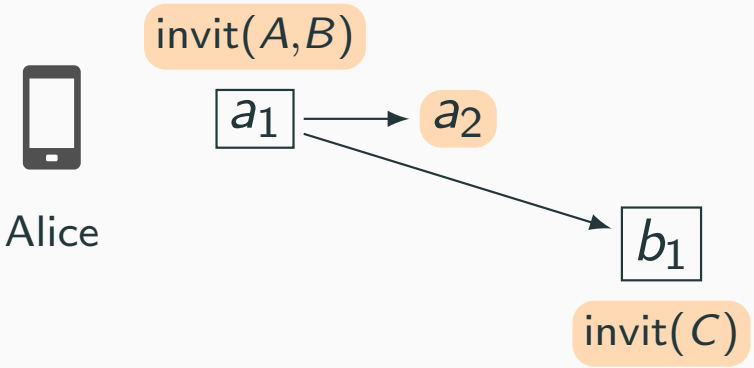
modification stable $\iff$ *elle est observée par chacun de ses observateurs honnêtes requis au sein du journal.*



- observateurs honnêtes requis =
  - \+ pairs invités dans les dépendances
  - \+ pairs invités en concurrence

Théorème

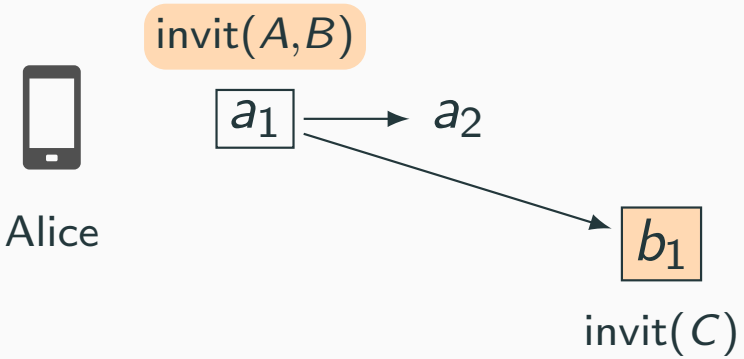
modification stable $\iff$ *elle est observée par chacun de ses observateurs honnêtes requis au sein du journal.*



- observateurs honnêtes requis =
  - \+ pairs invités dans les dépendances
  - \+ pairs invités en concurrence

Théorème

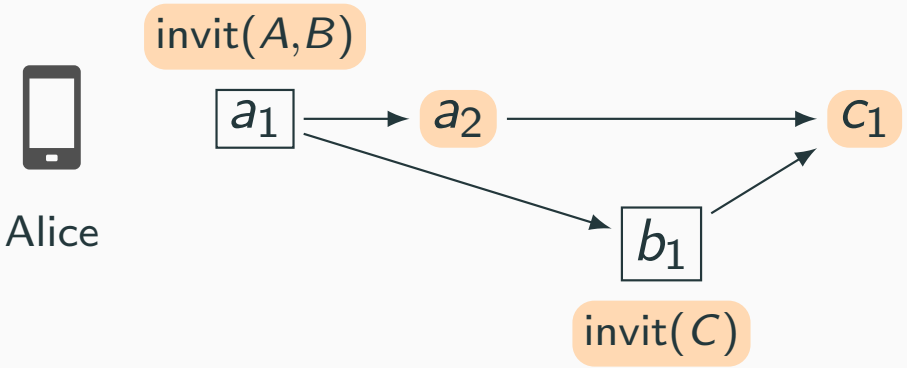
modification stable $\iff$ *elle est observée par chacun de ses observateurs honnêtes requis au sein du journal.*



- observateurs honnêtes requis =
  - \+ pairs invités dans les dépendances
  - \+ pairs invités en concurrence

Théorème

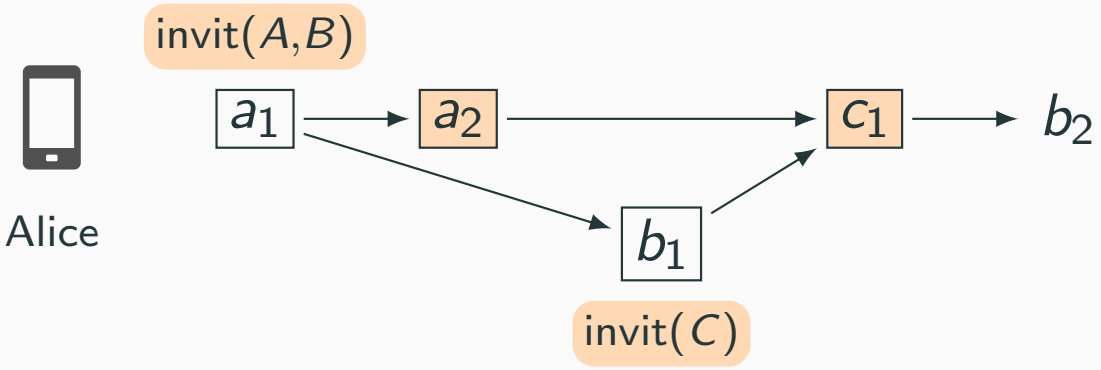
modification stable $\iff$ *elle est observée par chacun de ses observateurs honnêtes requis au sein du journal.*



- observateurs honnêtes requis =
  - \+ pairs invités dans les dépendances
  - \+ pairs invités en concurrence

Théorème

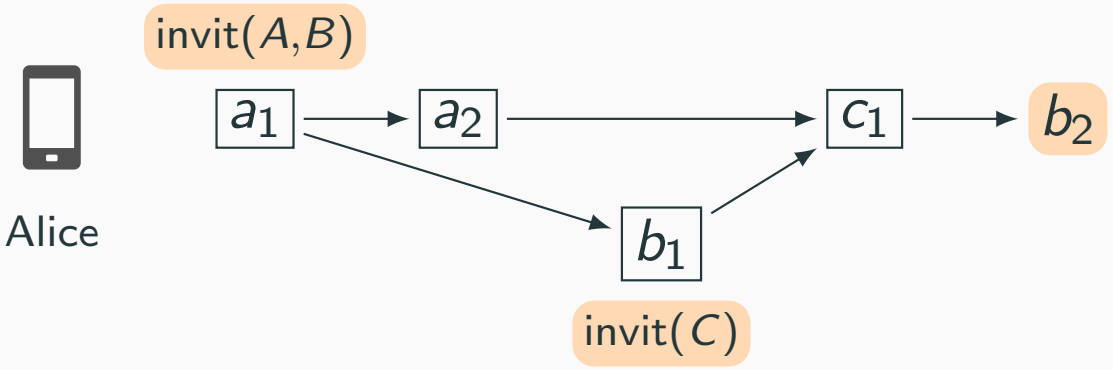
modification stable $\iff$ *elle est observée par chacun de ses observateurs honnêtes requis au sein du journal.*



- observateurs honnêtes requis =
  - \+ pairs invités dans les dépendances
  - \+ pairs invités en concurrence

Théorème

modification stable $\iff$ *elle est observée par chacun de ses observateurs honnêtes requis au sein du journal.*



- observateurs honnêtes requis =
  - \+ pairs invités dans les dépendances
  - \+ pairs invités en concurrence

Théorème

modification stable $\iff$ *elle est observée par chacun de ses observateurs honnêtes requis au sein du journal.*

Résumé des contributions

- contributions
  - formalisation et généralisation du **concept de stabilité**
  - formalisation des journaux et leurs invariants
  - protocole à **journaux complets** et protocole à **journaux tronqués**
  - formalisation et preuves des garanties offertes par les protocoles
  - **algorithme à complexité linéaire**[*] pour déterminer un sous-ensemble de modifications stables
- notre approche
  - protége la convergence avec des **journaux infalsifiables tronqués**
  - tolère la présence de **pairs malintentionnés**
  - est adapté aux **groupes dynamiques**
  - **sans coordination**
- limite : des pairs inactifs peuvent **bloquer la troncature**

*. par rapport à la taille du journal

Problématique 2 :
peut-on concevoir un protocole pour la co-édition de texte qui résiste mieux aux aléas du réseau et aux longues périodes de déconnexion ?

Alice

l	a	d	y

Bea

l	a	d	y

- document = **séquence** de caractères
  - ins(c, n) : insertion d'un caractère c après le n-ième caractère
  - del(n) : suppression du n-ième caractère

Alice

l	a	d	i	y

Bea

l	a	d	y

ins(3, *i*)

- document = **séquence** de caractères
  - ins(c, n) : insertion d'un caractère c après le n-ième caractère
  - del(n) : suppression du n-ième caractère

Alice

l	a	d	i	~~y~~

Bea

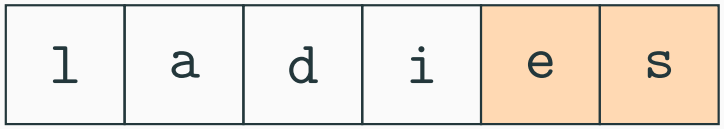
l	a	d	y

ins(3, *i*)

del(5)

- document = **séquence** de caractères
  - ins(c, n) : insertion d'un caractère c après le n-ième caractère
  - del(n) : suppression du n-ième caractère

Alice



ins(3, *i*)
del(5)
ins(4, *e*)
ins(5, *s*)

Bea

ins(4, !)

- document = **séquence** de caractères
  - ins(c, n) : insertion d'un caractère c après le n-ième caractère
  - del(n) : suppression du n-ième caractère

Alice

l	a	d	i	!	e	s

Bea

l	a	d	i	e	s	!

ins(3, *i*)
del(5)
ins(4, *e*)
ins(5, *s*)
ins(4, !)

ins(4, !)
ins(3, *i*)
del(5)
ins(4, *e*)
ins(5, *s*)

- document = **séquence** de caractères
  - ins(c, n) : insertion d'un caractère c après le n-ième caractère
  - del(n) : suppression du n-ième caractère

Alice

l	a	d	i	e	s

Bea

l	a	d	y	!

ins(3, *i*)
del(5)
ins(4, *e*)
ins(5, *s*)

ins(4, !)

- Conflict-free Replicated Data Types (CRDTs)
- l'exécution d'une opération génère un **message de synchronisation**
- les messages générés par des **opérations concurrentes** peuvent être **intégrés dans un ordre quelconque**

Alice

l	a	d	i	e	s	!

ins(3, *i*)
del(5)
ins(4, *e*)
ins(5, *s*)
message de Bea

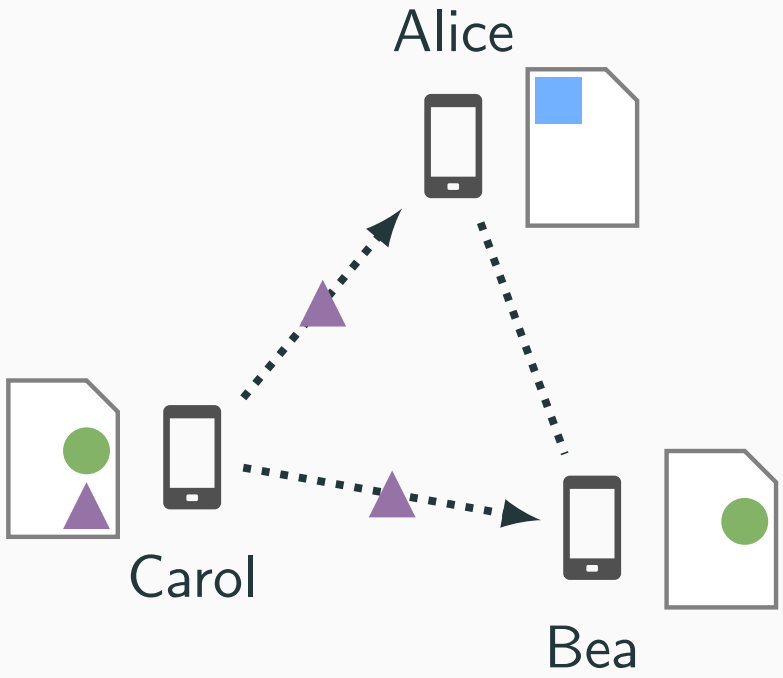
Bea

l	a	d	i	e	s	!

ins(4, !)
message·s de Alice

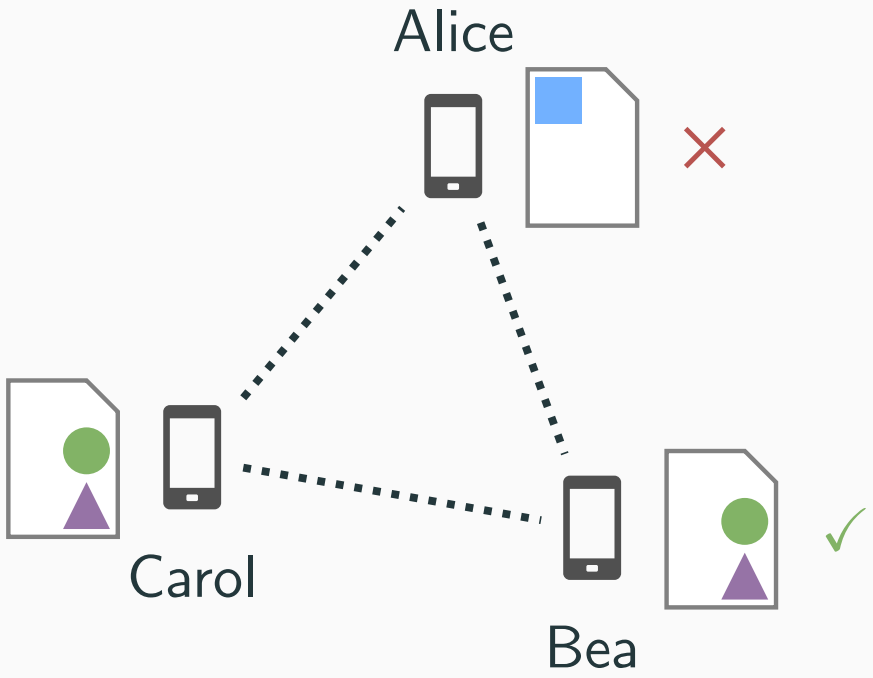
- Conflict-free Replicated Data Types (CRDTs)
- l'exécution d'une opération génère un **message de synchronisation**
- les messages générés par des **opérations concurrentes** peuvent être **intégrés dans un ordre quelconque**

Synchronisation par opérations (1)



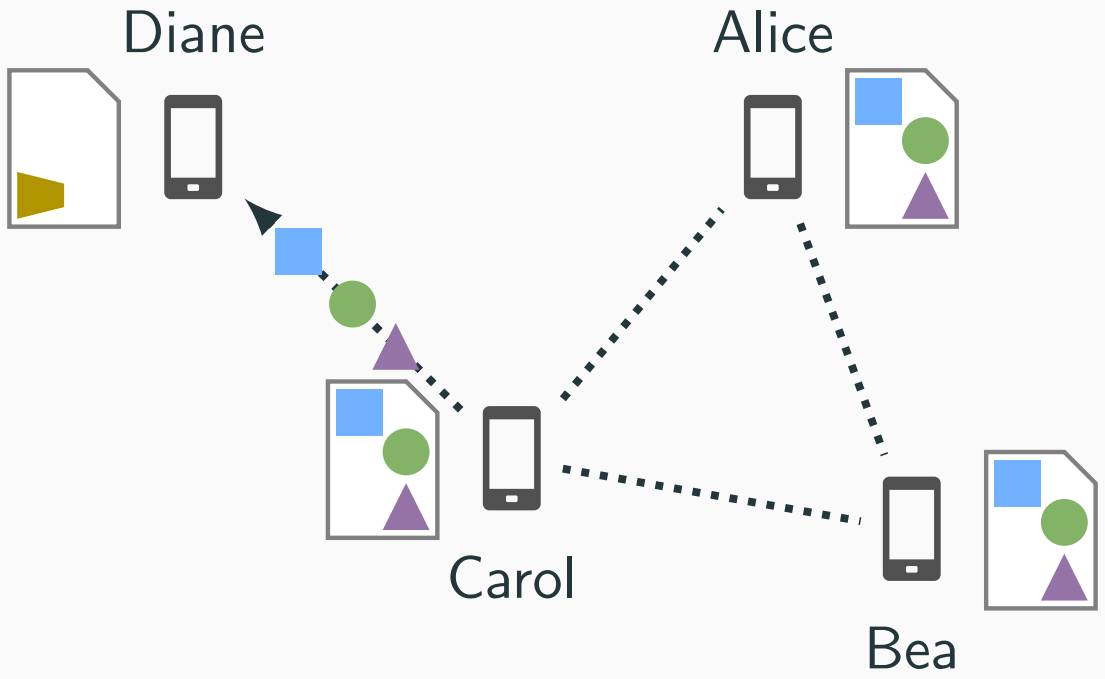
1. un message est intégré **exactement une fois**
2. ordre spécifique d'intégration de certains messages
   - l'**insertion** d'une valeur est intégrée **avant sa suppression**
3. **intégration causale** des messages (implique 2.)

Synchronisation par opérations (1)



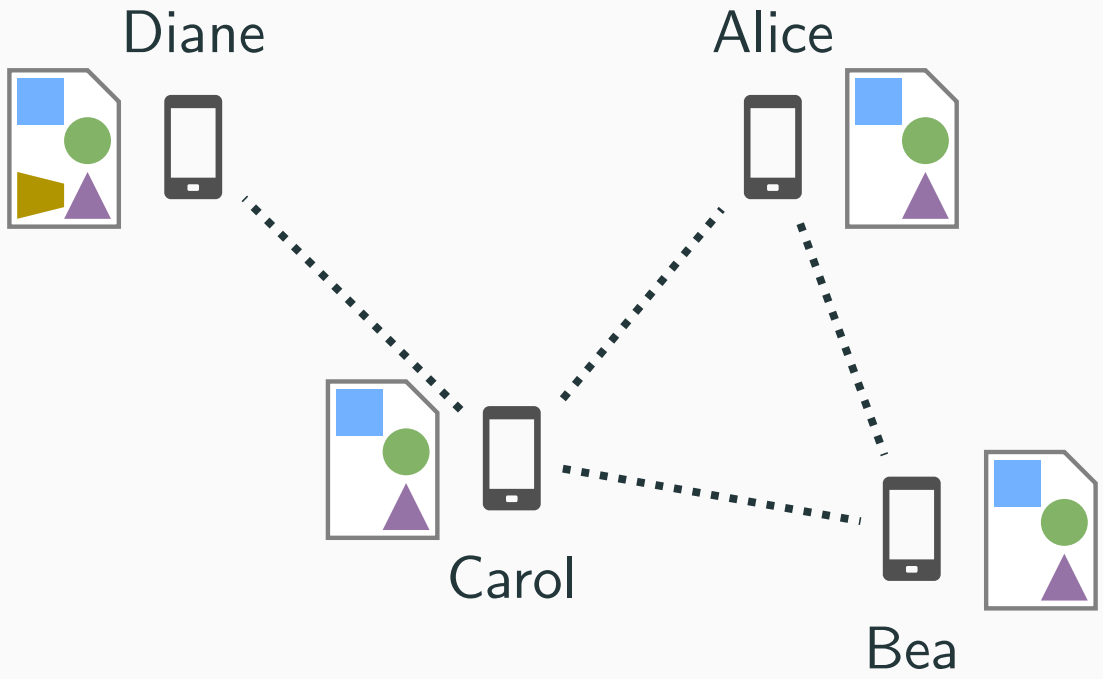
1. un message est intégré **exactement une fois**
2. ordre spécifique d'intégration de certains messages
   - l'**insertion** d'une valeur est intégrée **avant sa suppression**
3. **intégration causale** des messages (implique 2.)

Synchronisation par opérations (2)



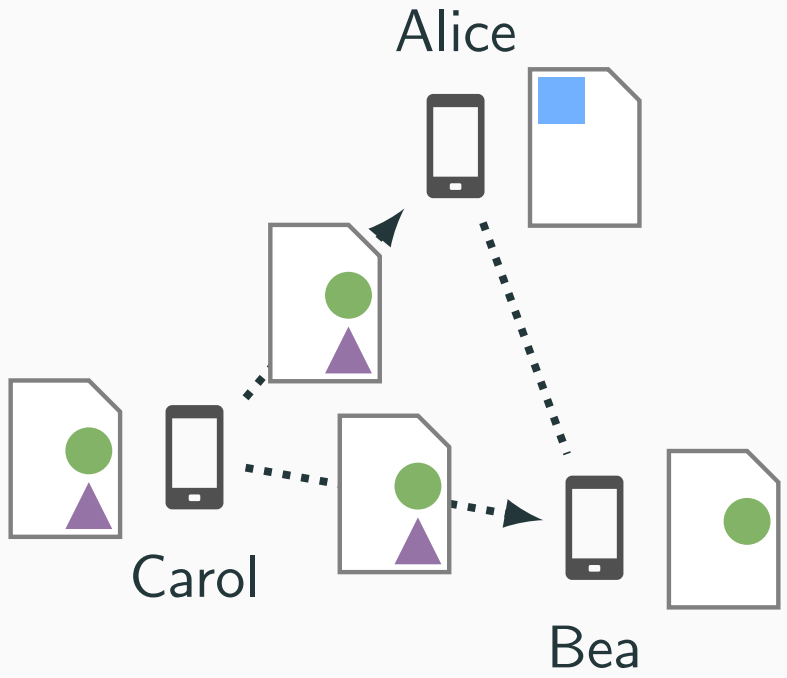
+ messages de petite taille
- exige une dissémination fiable des message
- la perte d'un message peut **propager des ralentissements**
- coût en communication et en calcul élevé
  - après une **longue période de déconnexion**

Synchronisation par opérations (2)



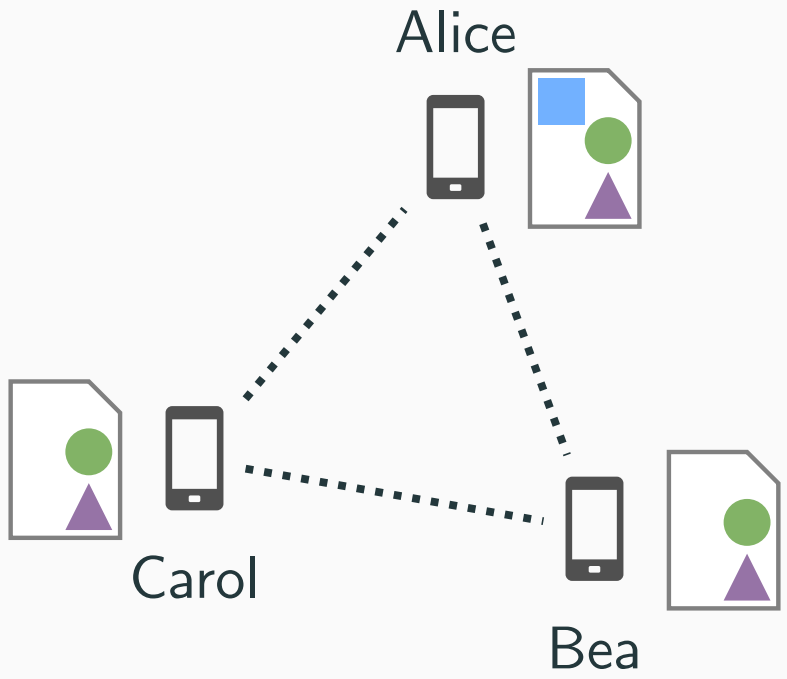
+ messages de petite taille
- exige une dissémination fiable des message
- la perte d'un message peut **propager des ralentissements**
- coût en communication et en calcul élevé
  - après une **longue période de déconnexion**

Synchronisation par états (1)



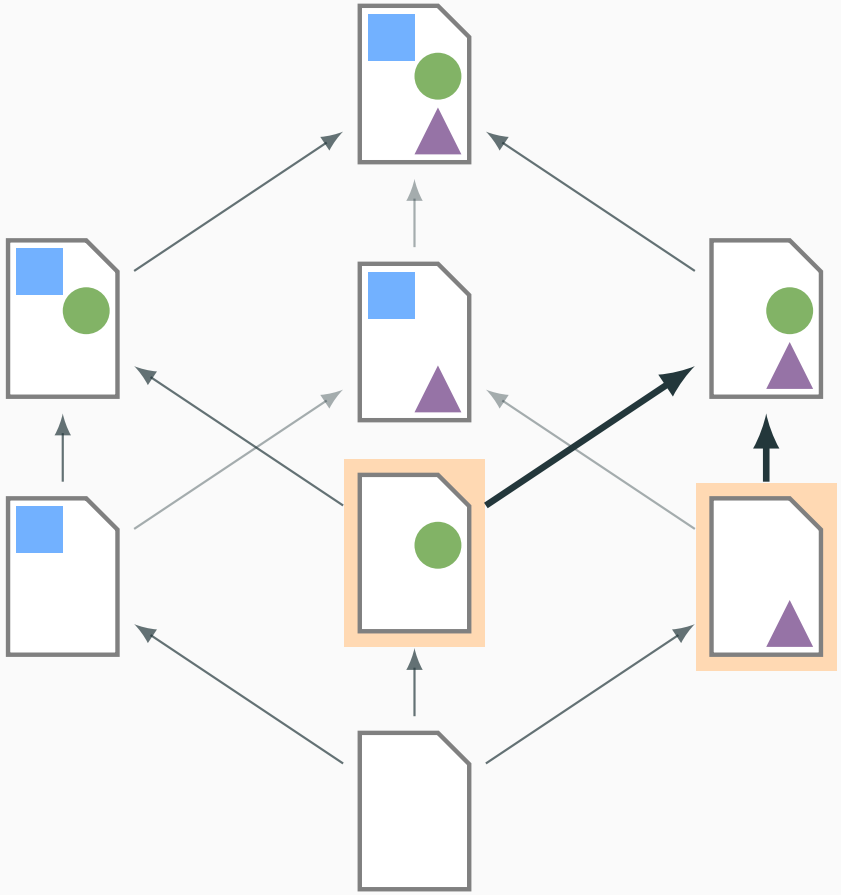
- les pairs **transmettent** directement l'**état mis à jour**
- ils **fusionnent** leur état actuel avec les états reçus

Synchronisation par états (1)

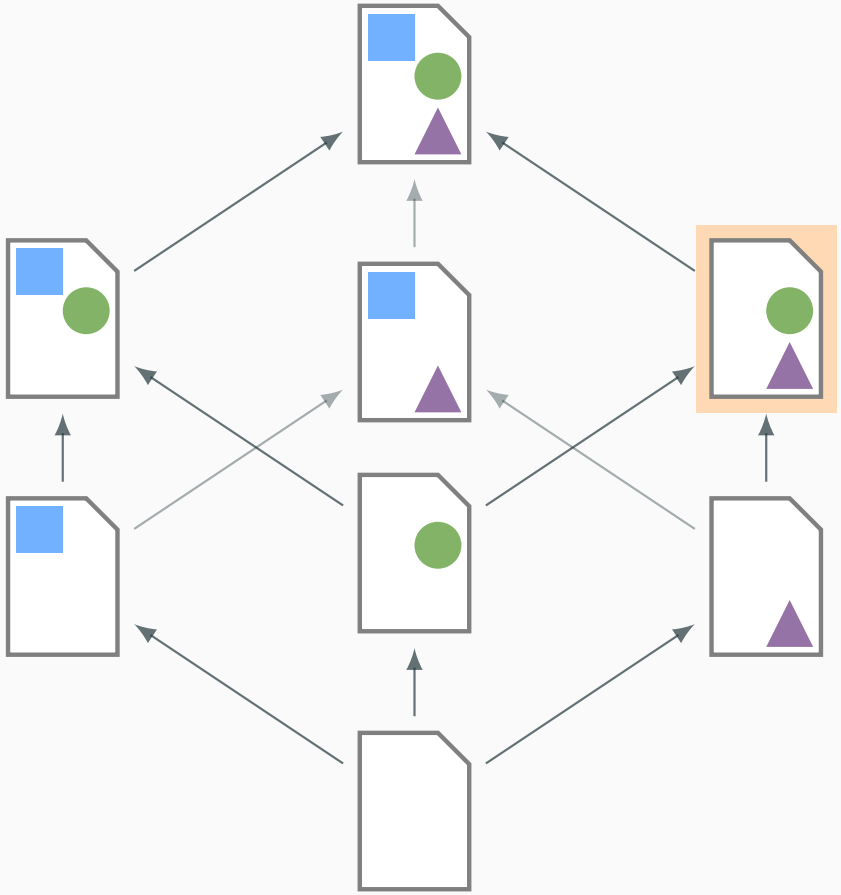


- les pairs **transmettent** directement l'**état mis à jour**
- ils **fusionnent** leur état actuel avec les états reçus

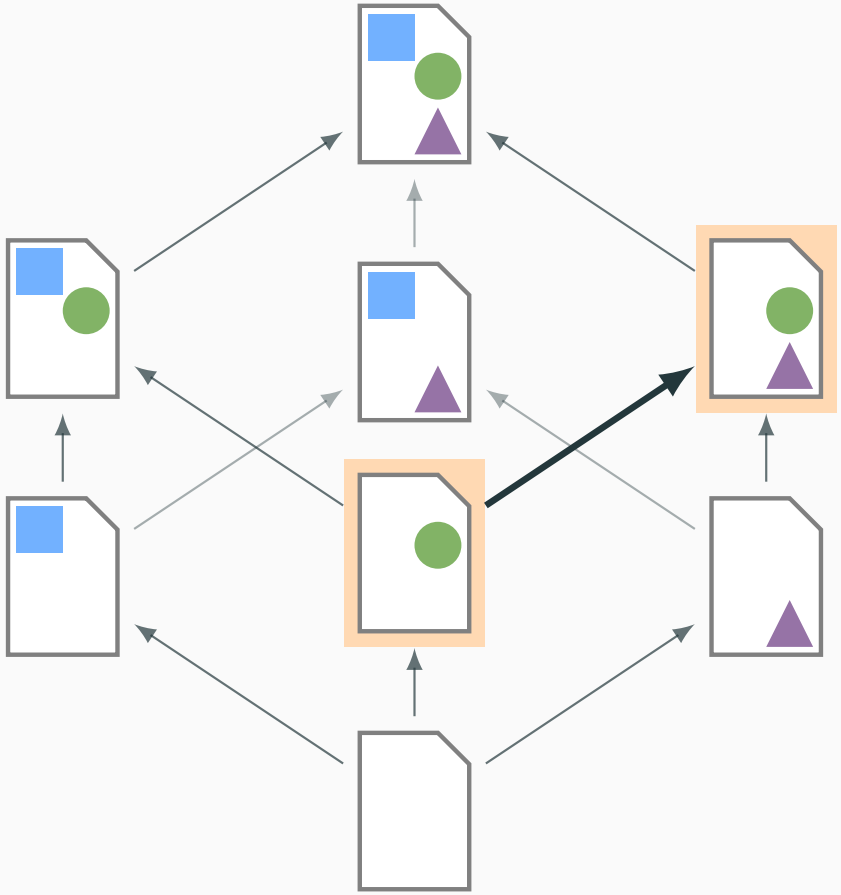
Synchronisation par états (2)



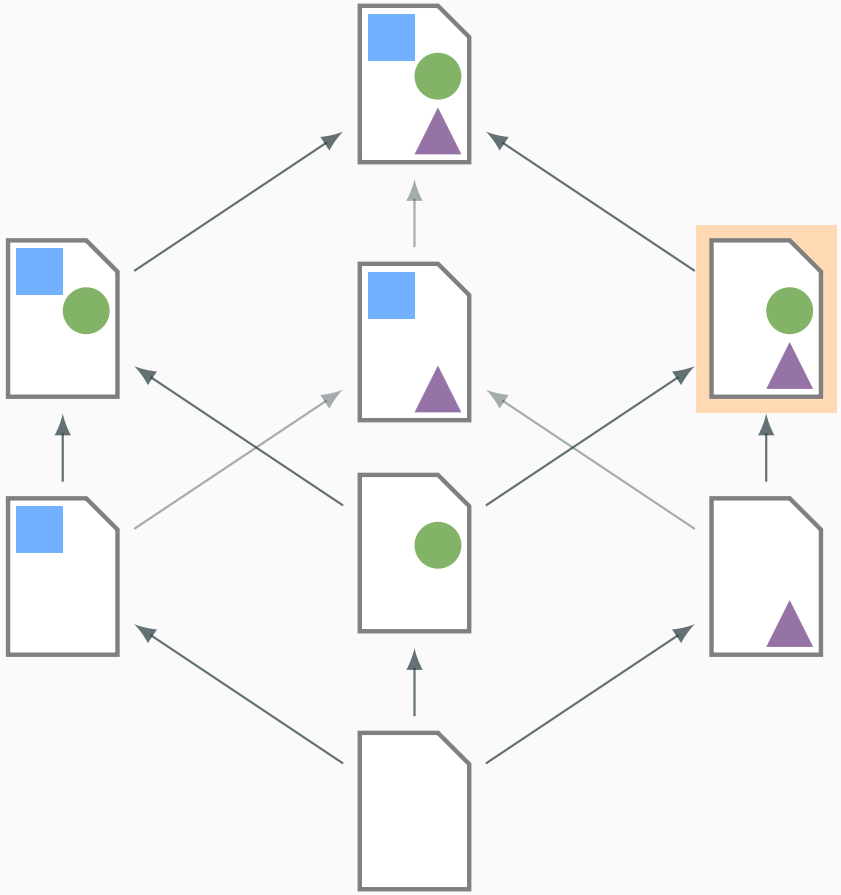
- l'ensemble des états forme un **sup demi-treillis**
  - fusion de 2 états donne le plus petit état qui leur est $\geq$
  - fusion : commutative, associative, idempotente



- l'ensemble des états forme un **sup demi-treillis**
  - fusion de 2 états donne le plus petit état qui leur est $\geq$
  - fusion : commutative, associative, idempotente

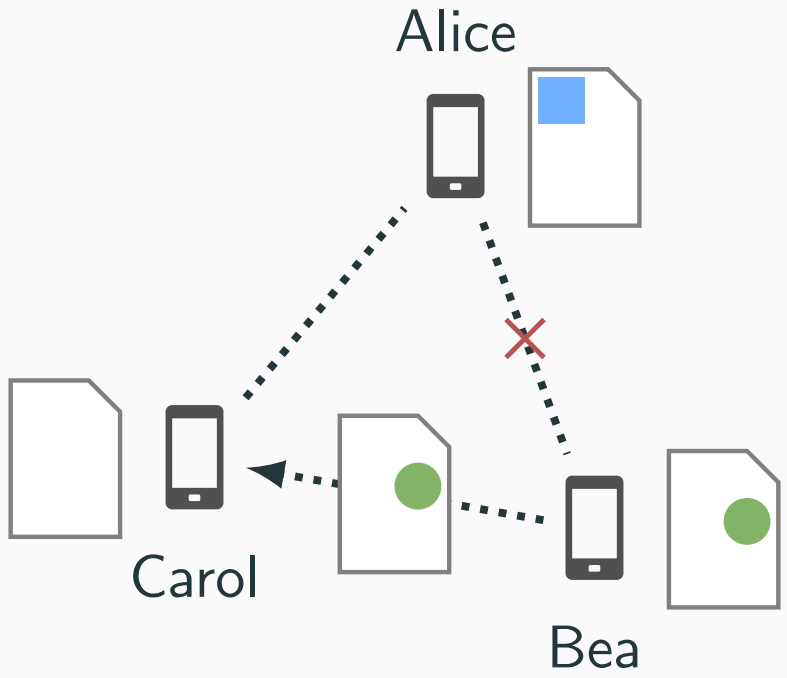


- l'ensemble des états forme un **sup demi-treillis**
  - fusion de 2 états donne le plus petit état qui leur est $\geq$
  - fusion : commutative, associative, idempotente



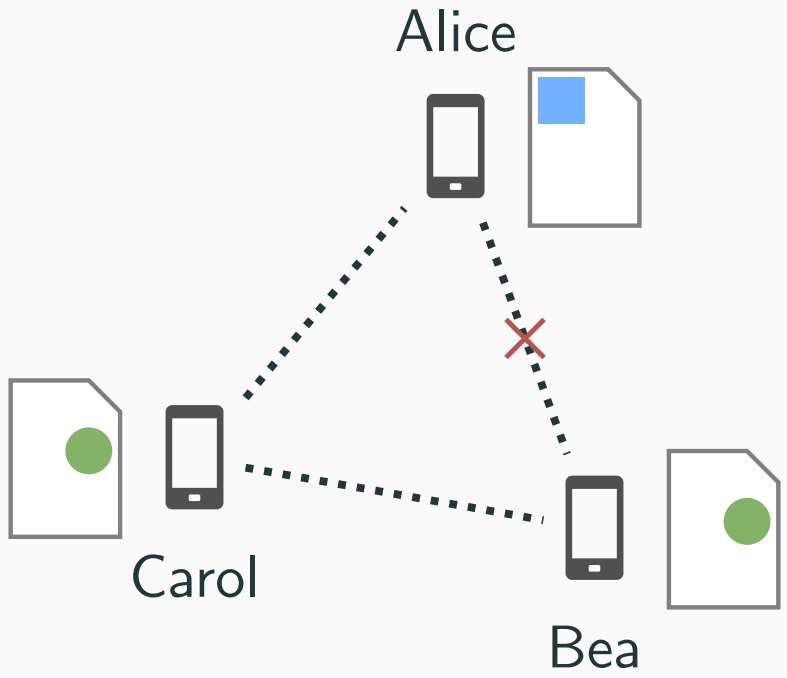
- l'ensemble des états forme un **sup demi-treillis**
  - fusion de 2 états donne le plus petit état qui leur est $\geq$
  - fusion : commutative, associative, idempotente

Synchronisation par états (3)



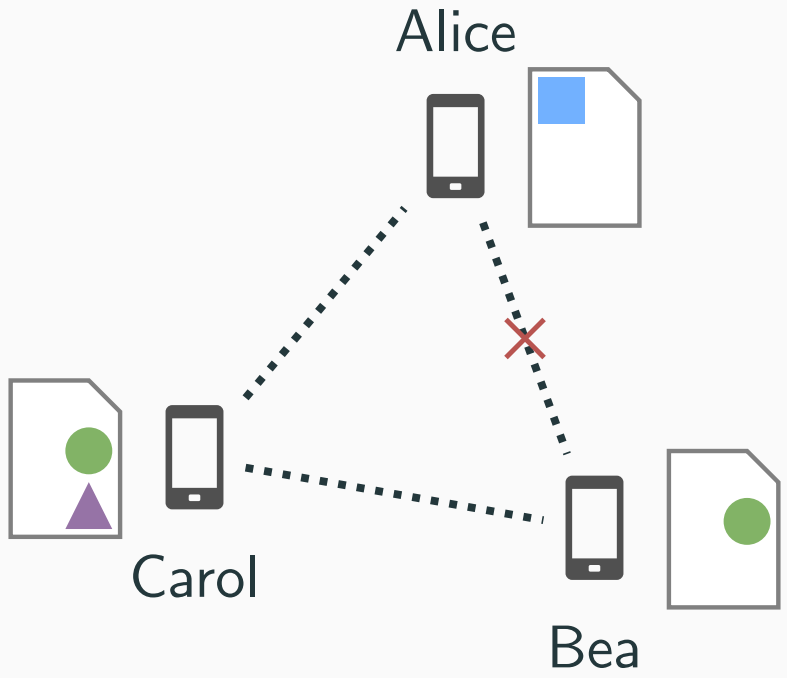
+ **résiste aux aléas du réseau**
  - tolère perte, duplication, et réordonnement des messages
- adapté aux **CRDTs avec une faible occupation mémoire**

Synchronisation par états (3)



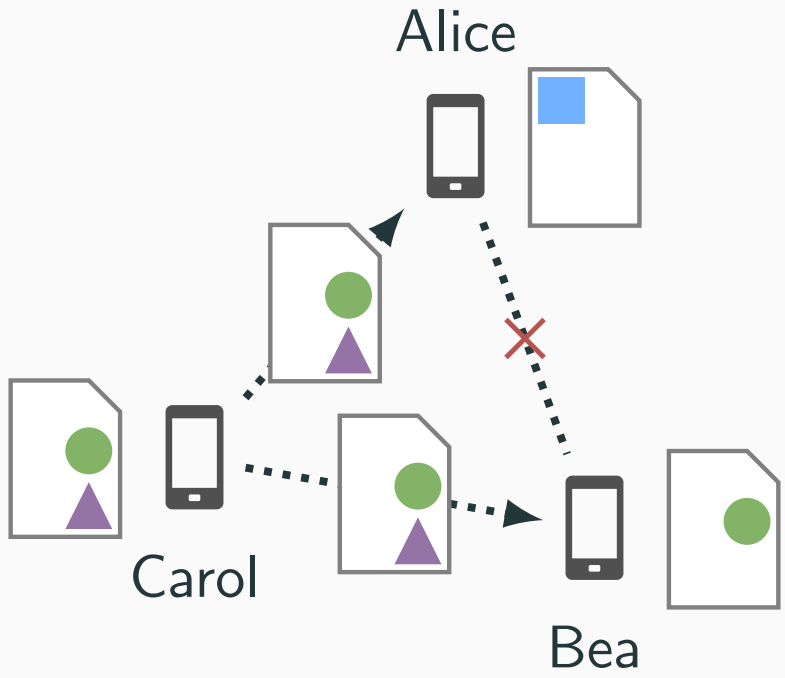
+ **résiste aux aléas du réseau**
  - tolère perte, duplication, et réordonnement des messages
- adapté aux **CRDTs avec une faible occupation mémoire**

Synchronisation par états (3)



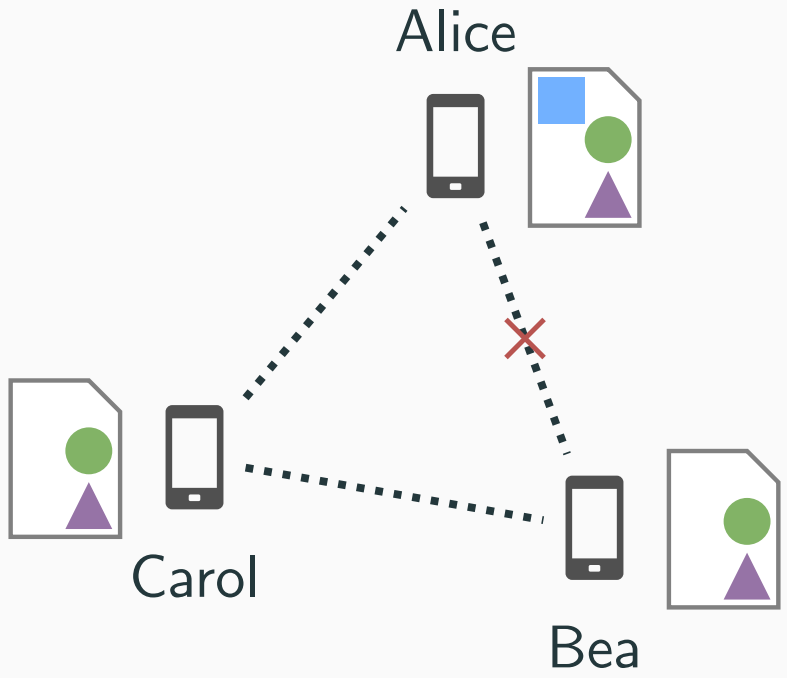
+ **résiste aux aléas du réseau**
  - tolère perte, duplication, et réordonnement des messages
- adapté aux **CRDTs avec une faible occupation mémoire**

Synchronisation par états (3)



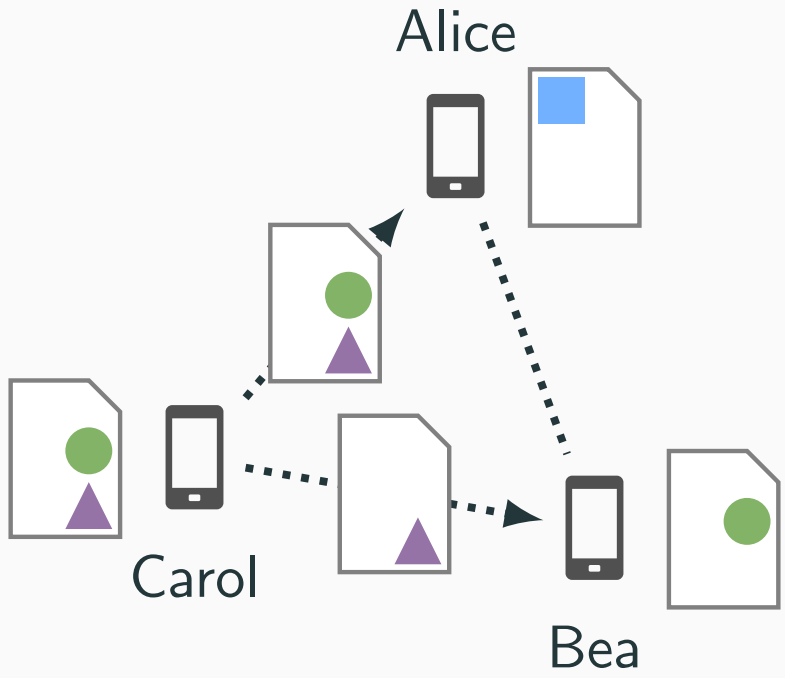
+ **résiste aux aléas du réseau**
  - tolère perte, duplication, et réordonnement des messages
- adapté aux **CRDTs avec une faible occupation mémoire**

Synchronisation par états (3)



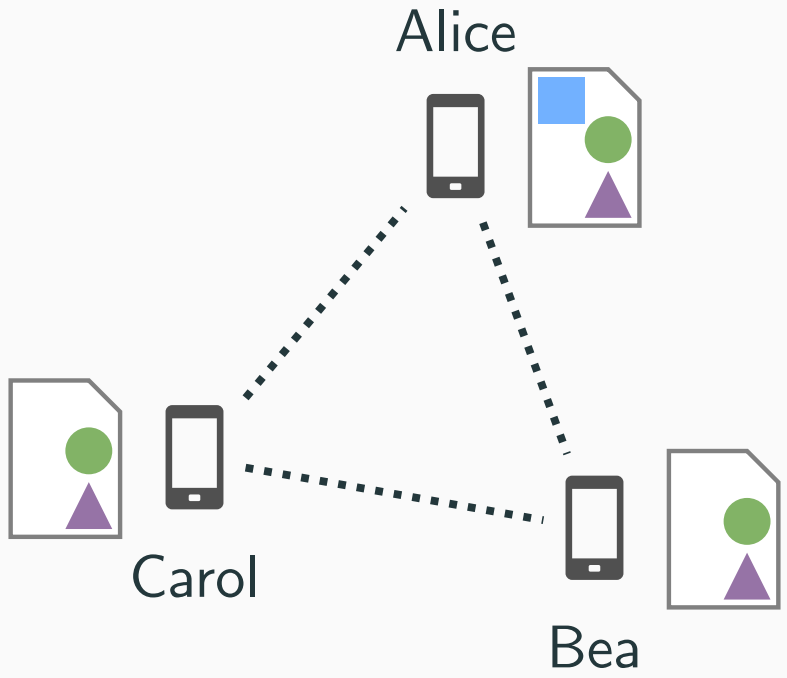
+ **résiste aux aléas du réseau**
  - tolère perte, duplication, et réordonnement des messages
- adapté aux **CRDTs avec une faible occupation mémoire**

Synchronisation par différences (d'états)



- un état est **décomposable en états irréductibles**
- différence entre l'état initial et l'état mis à jour
- \+ flexibilité pour la synchronisation
  - synchronisation par états
  - **synchronisation par états partiels** (différences d'états)

Synchronisation par différences (d'états)



- un état est **décomposable en états irréductibles**
- différence entre l'état initial et l'état mis à jour
- \+ flexibilité pour la synchronisation
  - synchronisation par états
  - **synchronisation par états partiels** (différences d'états)

Logoot [4]

- chaque caractère à une **position unique** et immuable
- **ordre total et dense entre les positions**
  - une position peut toujours être générée entre deux autres
- chaque caractère à un **identifiant unique** (*dot*)
  - identifiant du pair qui l'a inséré
  - entier incrémenté avant chaque insertion du pair

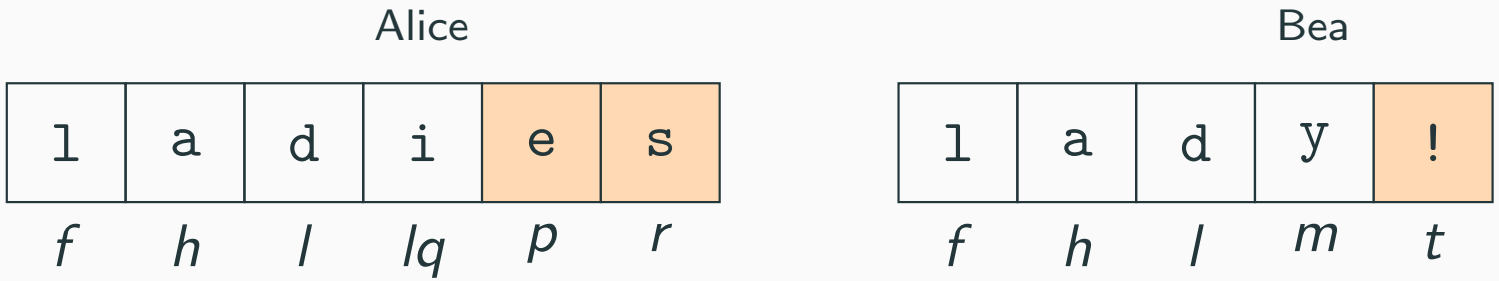
---

[4]. Weiss et al., "Logoot : A Scalable Optimistic Replication Algorithm for Collaborative Editing on P2P Networks".

- chaque caractère à une **position unique** et immuable
- **ordre total et dense entre les positions**
  - une position peut toujours être générée entre deux autres
- chaque caractère à un **identifiant unique** (*dot*)
  - identifiant du pair qui l'a inséré
  - entier incrémenté avant chaque insertion du pair

---

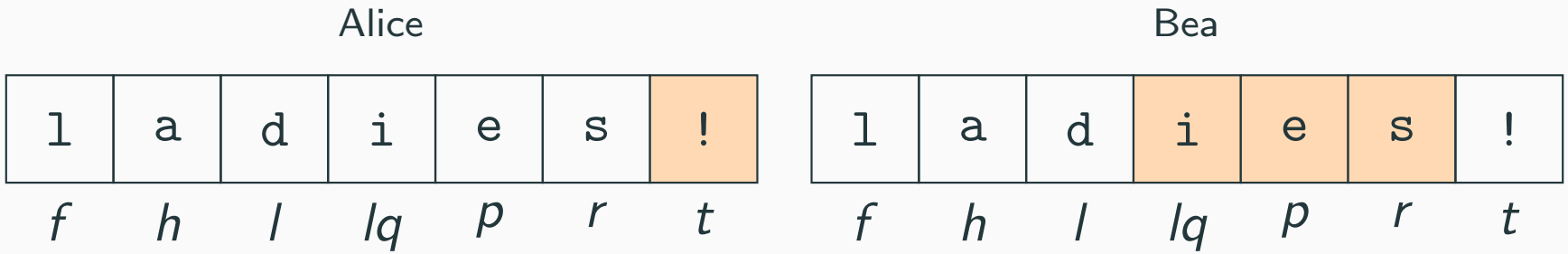
[4]. Weiss et al., "Logoot : A Scalable Optimistic Replication Algorithm for Collaborative Editing on P2P Networks".



- chaque caractère à une **position unique** et immuable
- **ordre total et dense entre les positions**
  - une position peut toujours être générée entre deux autres
- chaque caractère à un **identifiant unique** (*dot*)
  - identifiant du pair qui l'a inséré
  - entier incrémenté avant chaque insertion du pair

---

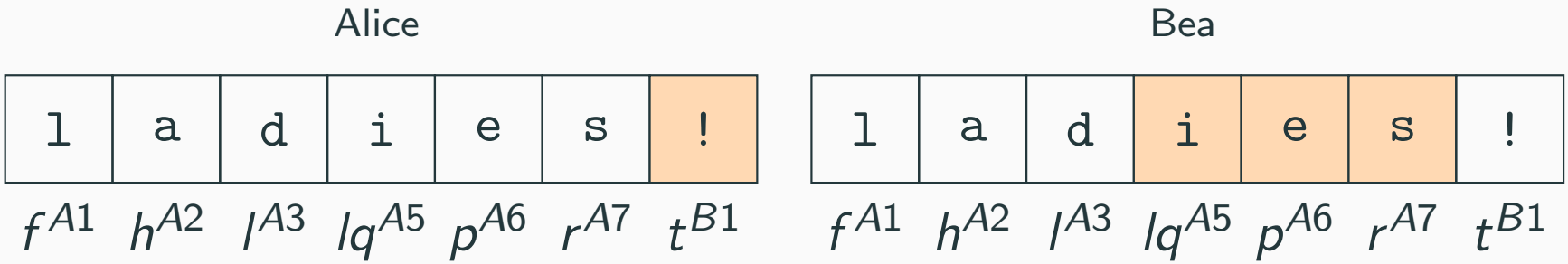
[4]. Weiss et al., "Logoot : A Scalable Optimistic Replication Algorithm for Collaborative Editing on P2P Networks".



- chaque caractère à une **position unique** et immuable
- **ordre total et dense entre les positions**
  - une position peut toujours être générée entre deux autres
- chaque caractère à un **identifiant unique** (*dot*)
  - identifiant du pair qui l'a inséré
  - entier incrémenté avant chaque insertion du pair

---

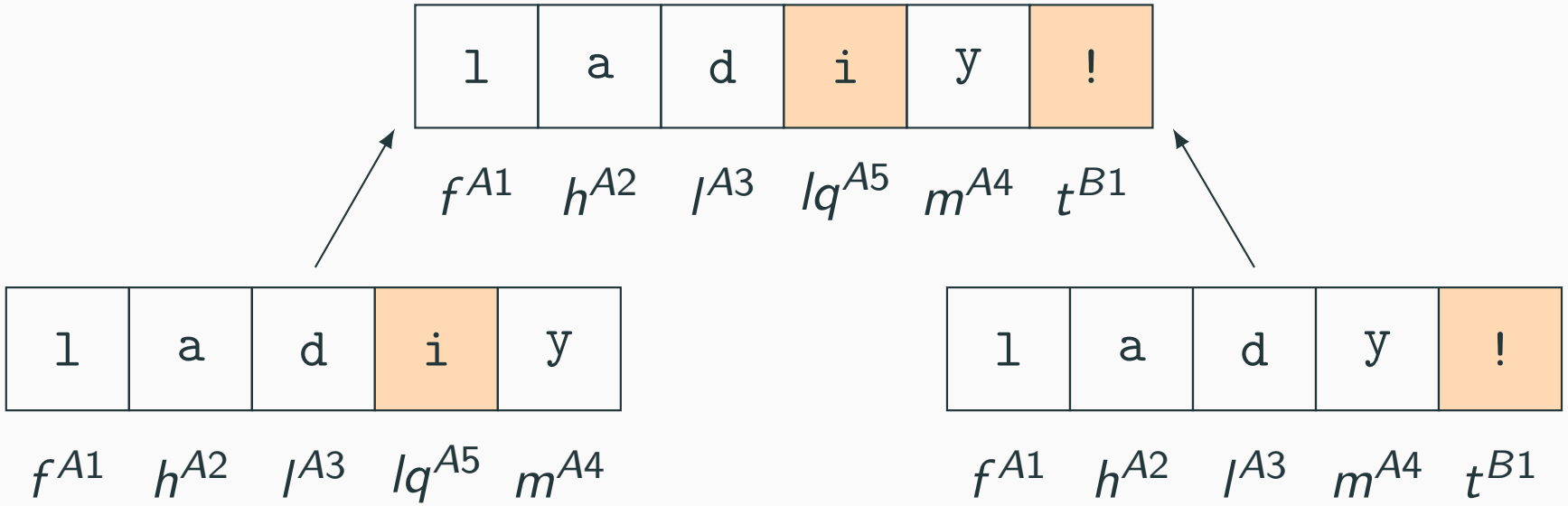
[4]. Weiss et al., "Logoot : A Scalable Optimistic Replication Algorithm for Collaborative Editing on P2P Networks".



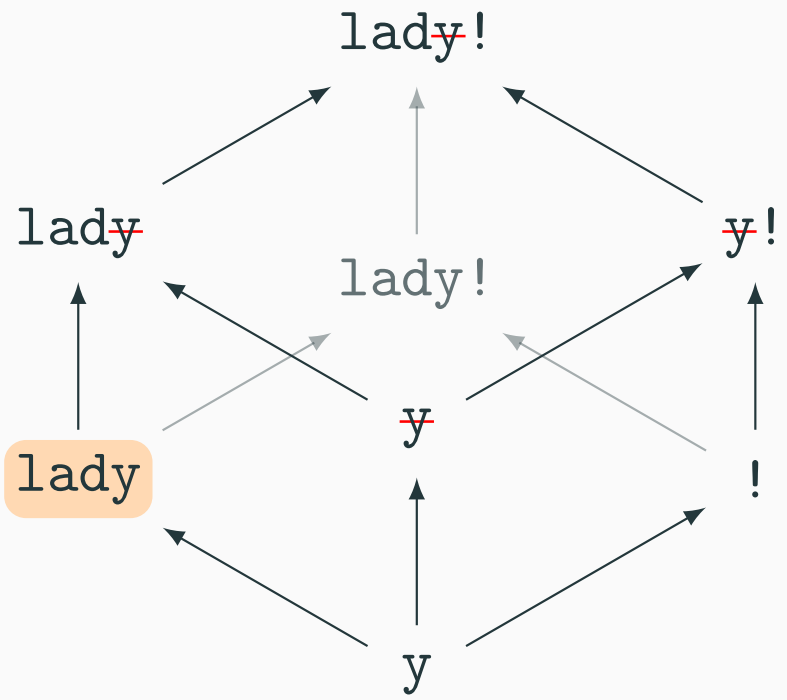
- chaque caractère à une **position unique** et immuable
- **ordre total et dense entre les positions**
  - une position peut toujours être générée entre deux autres
- chaque caractère à un **identifiant unique** (*dot*)
  - identifiant du pair qui l'a inséré
  - entier incrémenté avant chaque insertion du pair

---

[4]. Weiss et al., "Logoot : A Scalable Optimistic Replication Algorithm for Collaborative Editing on P2P Networks".

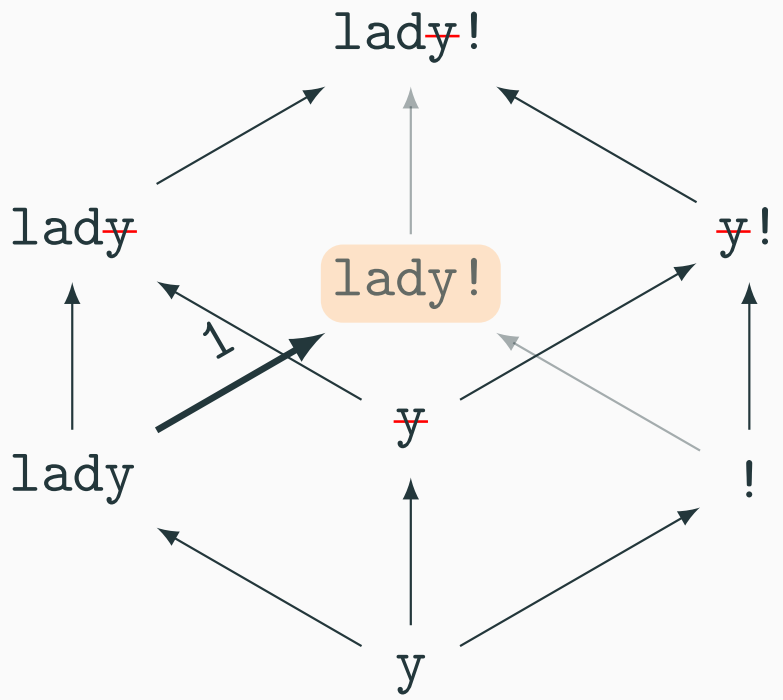


- **Logoot sans suppression** synchronisé par différences d'états
  - union des ensembles de pairs valeur-position
- valeur supprimée $\cong$ valeur qui n'a pas encore été insérée



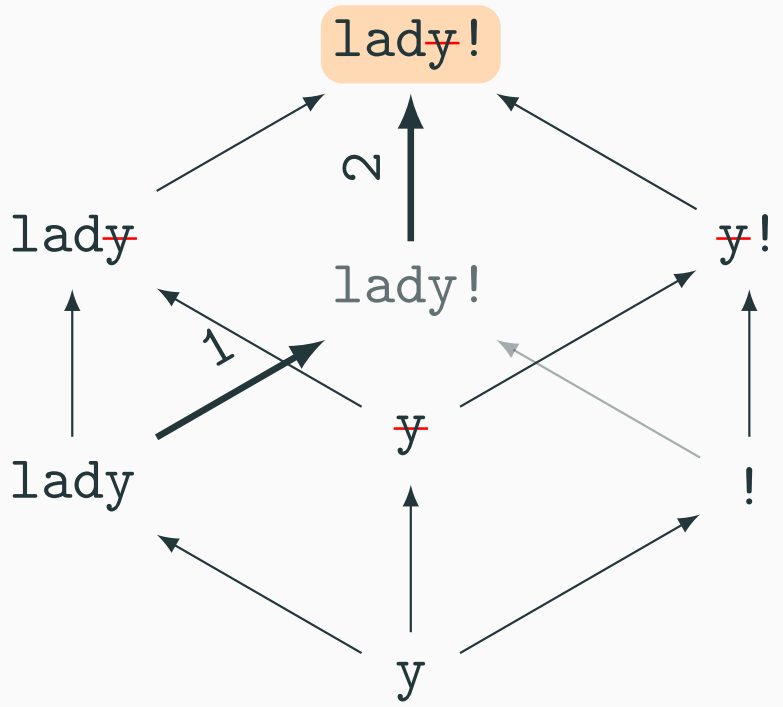
1. intègre '!'
2. intègre '~~y~~'

- deux états irréductibles pour chaque couple caractère-position
  - un qui représente la présence du couple
  - un qui représente la suppression du couple
  - le premier est plus petit que le second



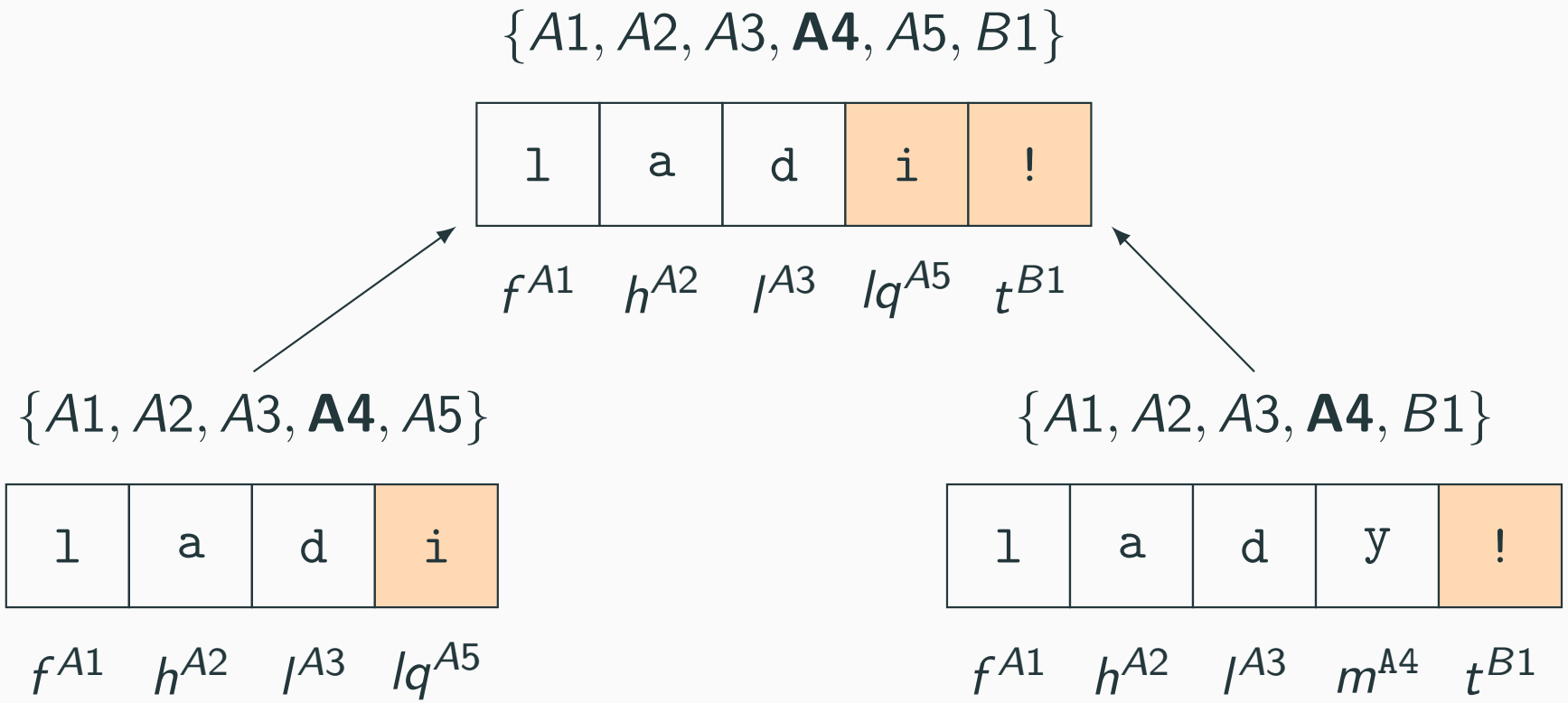
1. intègre '!'
2. intègre 'y'

- deux états irréductibles pour chaque couple caractère-position
  - un qui représente la présence du couple
  - un qui représente la suppression du couple
  - le premier est plus petit que le second



1. intègre '!'
2. intègre 'y'

- deux états irréductibles pour chaque couple caractère-position
  - un qui représente la présence du couple
  - un qui représente la suppression du couple
  - le premier est plus petit que le second



- état :
  - l'ensemble des valeurs et leurs positions
  - l'ensemble des identifiants des valeurs insérées
- le deuxième ensemble évite la réinsertion d'une valeur déjà insérée

Résumé des contributions

- contributions
  - **formalisation des séquences** à positions densément ordonnées
  - synchronisation par différences d'état de ces séquences (e.g. *Logoot*)
  - nouvelle structure de positions : **Dotted LogootSplit**
    - **optimisée** pour la synchronisation par différences d'états
    - **positions agrégeables** pour former des blocs
    - occupent moins d'espace mémoire que les positions LogootSplit
- notre approche
  - **résiste** mieux aux **aléas du réseau**
  - peut être appliquée à plusieurs séquences répliquées
  - implémentée* et testée en conditions réelles

*. `https://github.com/coast-team/dotted-logootsplit`

Synthèse

- sécurité et passage à l'échelle des collaborations pair-à-pair

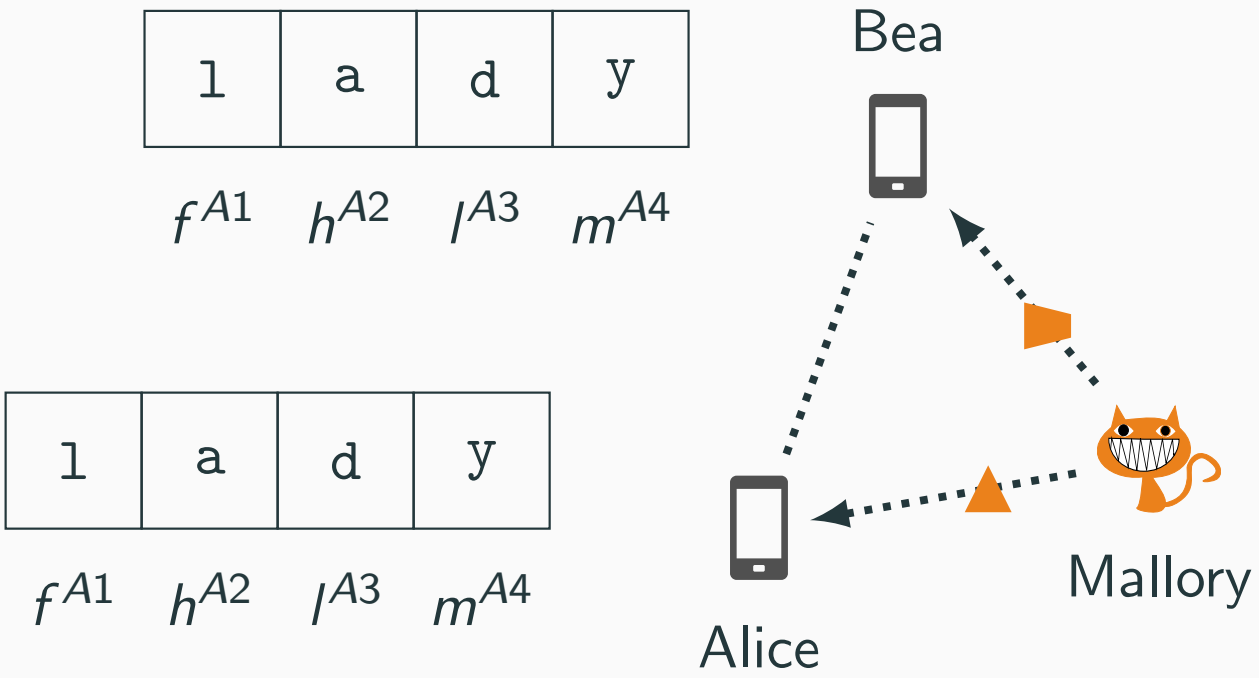
Protection de la convergence

- formalisation du concept de stabilité
- protocole à journaux complets et à journaux tronqués
  - tolère la présence de pairs malintentionnés
  - adapté aux groupes dynamiques
  - sans aucune coordination

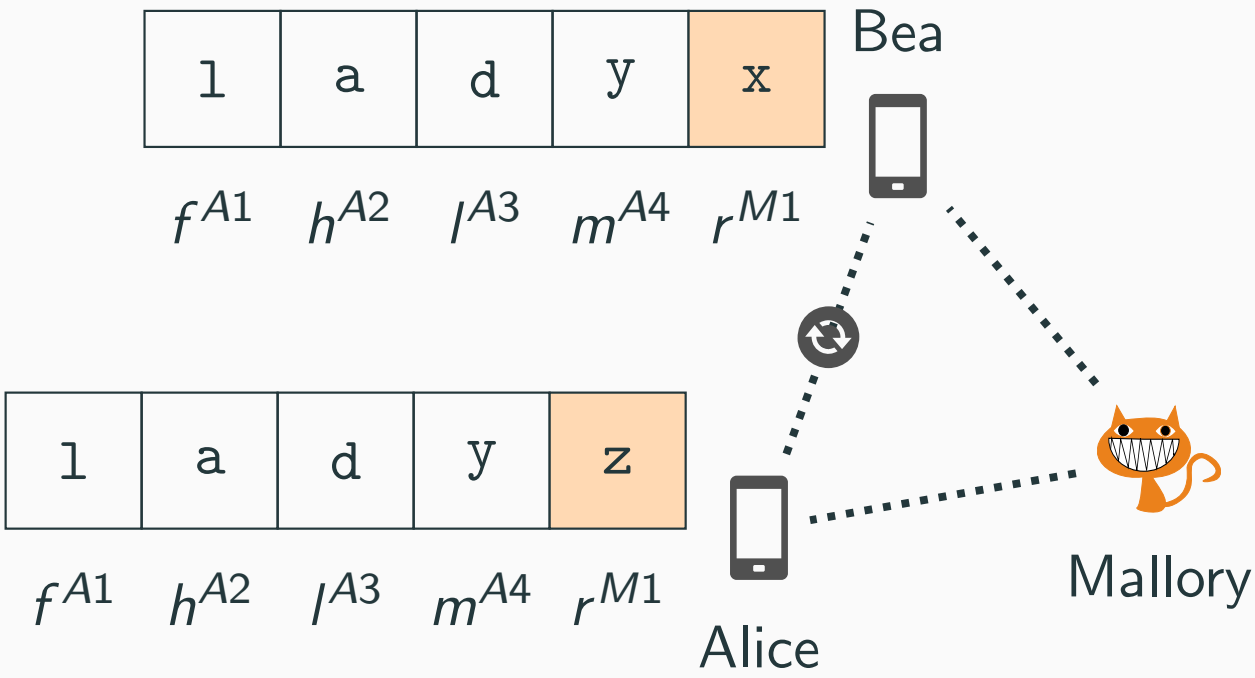
Co-édition massive de texte

- formalisation des séquences à positions densément ordonnées
- synchronisation de séquences par différences d'états
  - résiste mieux aux aléas du réseau
- nouvelle structure de positions : Dotted LogootSplit

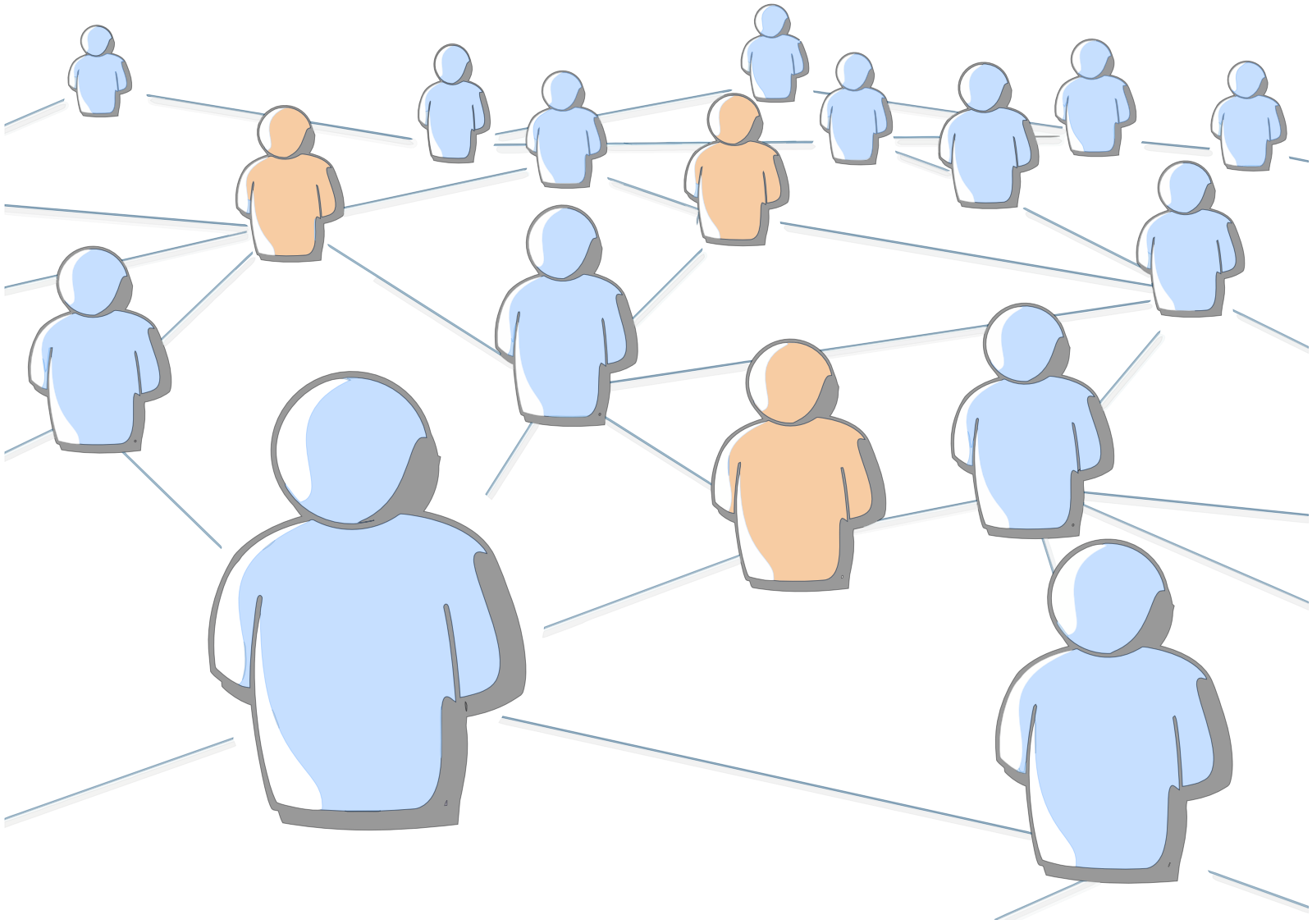
- les pairs inactifs ou non-coopératifs peuvent bloquer la troncature
  - évincer les pairs inactifs ou non-coopératifs
  - stabilité au sein de sous-groupes
- la synchronisation par différences d'états est flexible
  - questionne les **stratégies de synchronisation à adopter**



- les CRDTs utilisent des **identifiants pour distinguer des valeurs**
- un pair malintentionné peut associer à un **même identifiant des valeurs distinctes**



- les CRDTs utilisent des **identifiants pour distinguer des valeurs**
- un pair malintentionné peut associer à un **même identifiant des valeurs distinctes**



Publications

Prunable Authenticated Log and Authenticable Snapshot in Distributed Collaborative Systems in proceedings of the 4th IEEE International Conference on Collaboration and Internet Computing, (CIC 2018) – Victorien Elvinger, Gérald Oster, François Charoy

MUTE : A Peer-to-Peer Web-based Real-time Collaborative Editor in proceedings of the 15th European Conference on Computer Supported Cooperative Work (ECSCW 2017) – Matthieu Nicolas, Victorien Elvinger, Gérald Oster, Claudia-Lavinia Ignat, François Charoy

A Generic Undo Support for State-Based CRDTs in proceedings of the 23rd International Conference on Principles of Distributed Systems – Weihai Yu, Victorien Elvinger, Claudia-Lavinia Ignat

Références

[1] Phillip W. Hutto et al. Slow Memory : Weakening Consistency to Enchance Concurrency in Distributed Shared Memories. In *Proceedings of the 10th International Conference on Distributed Computing Systems (ICDCS 1990)*, pages 302-309, mai 1990. DOI : 10.1109/ICDCS.1990.89297.

[2] Carlos Baquero et al. Pure Operation-Based Replicated Data Types. *CoRR*, abs/1710.04469, 2017. arXiv : 1710.04469. URL : http://arxiv.org/abs/1710.04469.

[3] Prince Mahajan et al. Consistency, Availability, and Convergence. Rapport technique TR-11-22, Department of Computer Science, The University of Texas at Austin, 2011.

[4] Stéphane Weiss et al. Logoot : A Scalable Optimistic Replication Algorithm for Collaborative Editing on P2P Networks. In *Proceedings of the 29th IEEE International Conference on Distributed Computing Systems, ICDCS 2009*, pages 404-412, juin 2009. DOI : 10.1109/ICDCS.2009.75.

Transparents supplémentaires

Contribution : Stabilité

Définition

modification stable $\iff$ dépendance de toute modification ultérieurement **acceptée** dans le journal.

- primitive pour définir quelles modifications peuvent être supprimées
- ensemble de dépendances communes qui croît
  - nécessité de **limiter les concurrences** à l'aide d'invariants
  - le protocole garantit les invariants

Jounal View-Fork-Join-Causal (VFJC) dynamique

Invariants

1. une modification du possesseur du journal dépend des précédentes
2. ordre linéaire des modification de chaque pair honnête
3. une modification dépend directement de modifications présumées linéaires
4. les modification d'un pair dépendent de son invitation
5. l'auteur de chaque dépendance directe d'une modification est connu dans chacune des dépendances directes

Stabilité View-Fork-Join-Causal (VFJC) dynamique

Théorème

modification x stable $\iff$ *pour toute* **énumération** $p_1, \ldots, p_N$ *des pairs avec* p_N *le possesseur du journal, il existe une chaîne de dépendances* $x, x_1, \ldots, x_n$ $(n \leq N)$ *telle que* x_i *a pour auteur, reconnaît malintentionné, ou invite* p_i. *Si* $N \neq n$, *alors* x_n *invite* p_n.

Logoot [5]

[5]. Weiss et al., "Logoot : A Scalable Optimistic Replication Algorithm for Collaborative Editing on P2P Networks".

Logoot [5]

Alice

$f \quad h \quad l \quad lq \quad m$

Bea

$f \quad h \quad l \quad m \quad t$

$\text{ins}(3, \texttt{i}) \xrightarrow{\text{génére}} \langle \text{ins}, lq, \texttt{i} \rangle$

$\text{ins}(4, !) \xrightarrow{\text{génére}} \langle \text{ins}, t, ! \rangle$

---

[5]. Weiss et al., "Logoot : A Scalable Optimistic Replication Algorithm for Collaborative Editing on P2P Networks".

Logoot [5]

$$\text{ins}(3, \texttt{i}) \xrightarrow{\text{génére}} \langle \text{ins}, lq, \texttt{i} \rangle$$

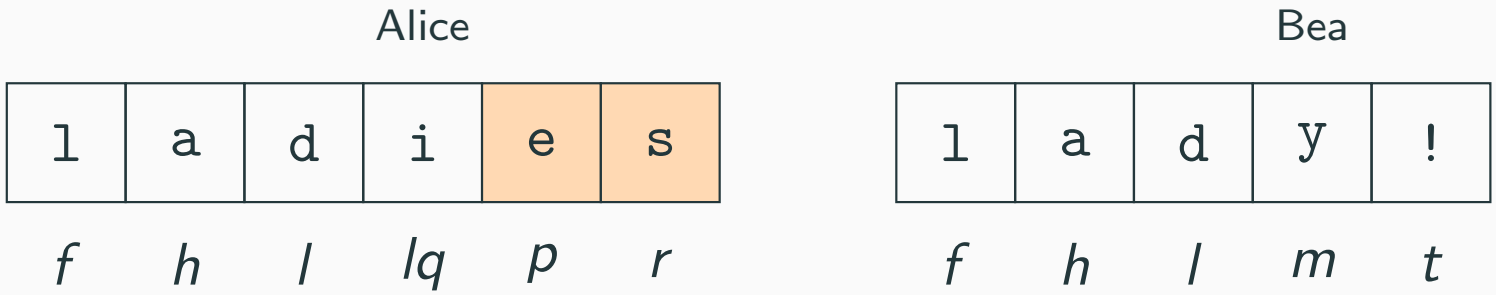
$$\text{del}(4) \xrightarrow{\text{génére}} \langle \text{del}, m \rangle$$

$$\text{ins}(4, !) \xrightarrow{\text{génére}} \langle \text{ins}, t, ! \rangle$$

---

[5]. Weiss et al., "Logoot : A Scalable Optimistic Replication Algorithm for Collaborative Editing on P2P Networks".

Logoot [5]



$\text{ins}(3, \texttt{i}) \xrightarrow{\text{génére}} \langle \text{ins}, lq, \texttt{i} \rangle$

$\text{del}(4) \xrightarrow{\text{génére}} \langle \text{del}, m \rangle$

$\text{ins}(4, \texttt{e}) \xrightarrow{\text{génére}} \langle \text{ins}, p, \texttt{e} \rangle$

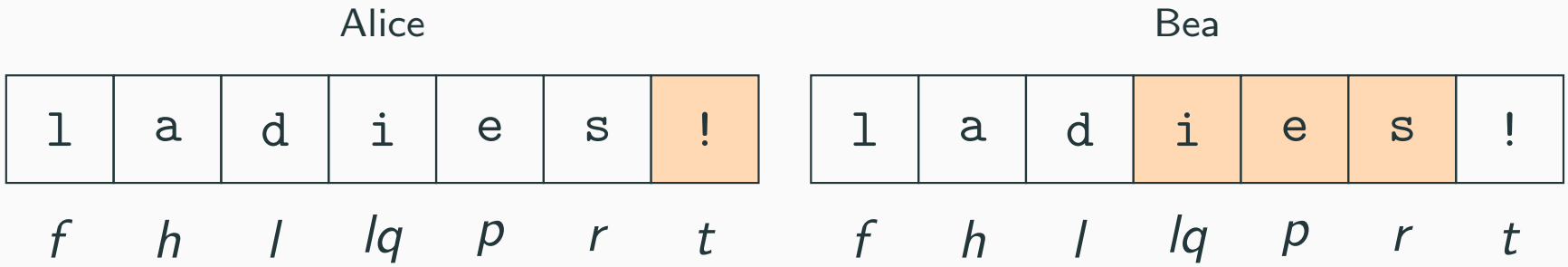
$\text{ins}(5, \texttt{s}) \xrightarrow{\text{génére}} \langle \text{ins}, r, \texttt{s} \rangle$

$\text{ins}(4, !) \xrightarrow{\text{génére}} \langle \text{ins}, t, ! \rangle$

---

[5]. Weiss et al., "Logoot : A Scalable Optimistic Replication Algorithm for Collaborative Editing on P2P Networks".

Logoot [5]



ins(3, i) —génére→ ⟨ins, *lq*, i⟩

del(4) —génére→ ⟨del, *m*⟩

ins(4, e) —génére→ ⟨ins, *p*, e⟩

ins(5, s) —génére→ ⟨ins, *r*, s⟩

intègre ⟨ins, *t*, !⟩

ins(4, !) —génére→ ⟨ins, *t*, !⟩

intègre ⟨ins, *lq*, i⟩

intègre ⟨del, *m*⟩

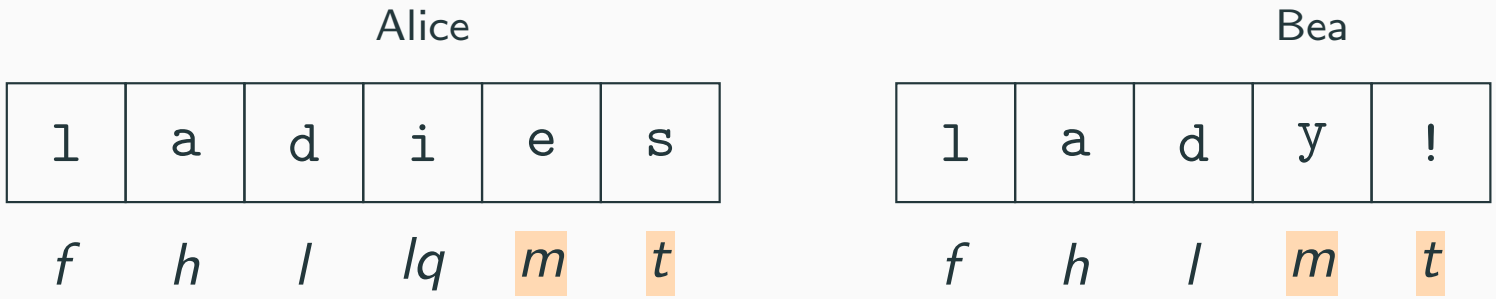
intègre ⟨ins, *p*, e⟩

intègre intègre ⟨ins, *r*, s⟩

---

[5]. Weiss et al., "Logoot : A Scalable Optimistic Replication Algorithm for Collaborative Editing on P2P Networks".

Logoot [5]



$\text{ins}(3, \texttt{i}) \xrightarrow{\text{génére}} \langle \text{ins}, lq, \texttt{i} \rangle$

$\text{del}(4) \xrightarrow{\text{génére}} \langle \text{del}, m \rangle$

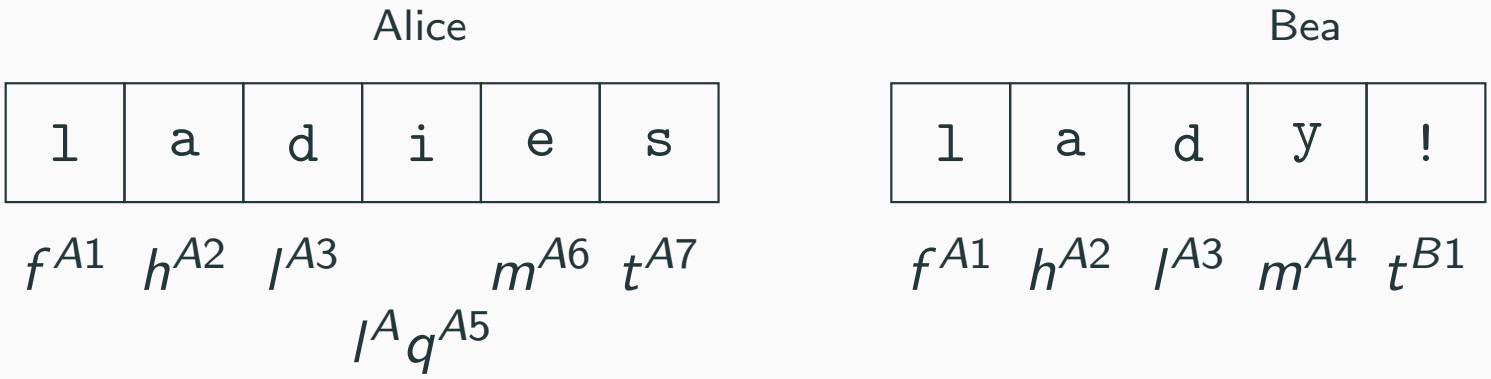
$\text{ins}(4, \texttt{e}) \xrightarrow{\text{génére}} \langle \text{ins}, m, \texttt{e} \rangle$

$\text{ins}(5, \texttt{s}) \xrightarrow{\text{génére}} \langle \text{ins}, t, \texttt{s} \rangle$

$\text{ins}(4, !) \xrightarrow{\text{génére}} \langle \text{ins}, t, ! \rangle$

[5]. Weiss et al., "Logoot : A Scalable Optimistic Replication Algorithm for Collaborative Editing on P2P Networks".

Logoot [5]



$\text{ins}(3, \texttt{i}) \xrightarrow{\text{génére}} \langle \text{ins}, l^A q^{A5}, \texttt{i} \rangle$

$\text{del}(4) \xrightarrow{\text{génére}} \langle \text{del}, m^{A6} \rangle$

$\text{ins}(4, \texttt{e}) \xrightarrow{\text{génére}} \langle \text{ins}, m^{A6}, \texttt{e} \rangle$

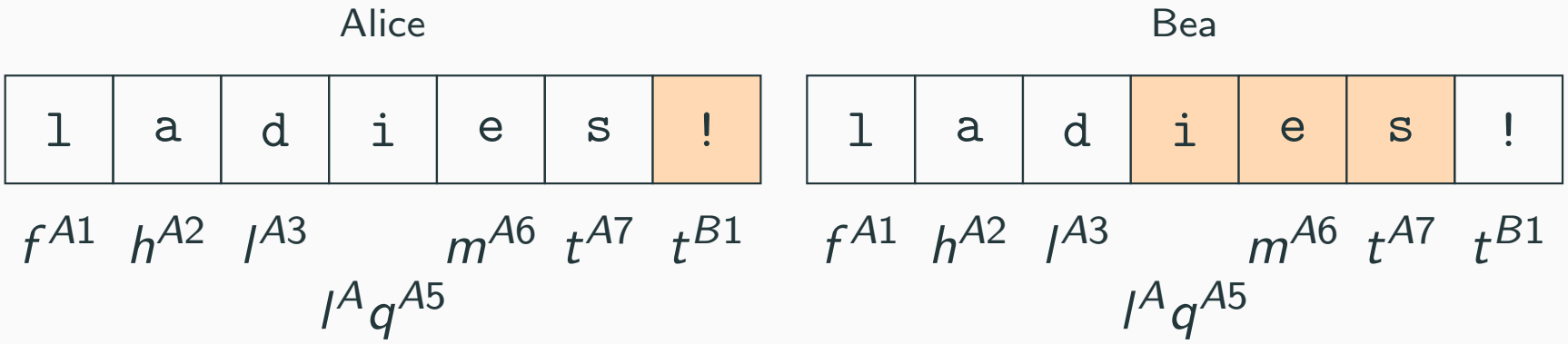
$\text{ins}(5, \texttt{s}) \xrightarrow{\text{génére}} \langle \text{ins}, t^{A7}, \texttt{s} \rangle$

$\text{ins}(4, !) \xrightarrow{\text{génére}} \langle \text{ins}, t^{B1}, ! \rangle$

---

[5]. Weiss et al., "Logoot : A Scalable Optimistic Replication Algorithm for Collaborative Editing on P2P Networks".

Logoot [5]



$\text{ins}(3, \texttt{i}) \xrightarrow{\text{génére}} \langle \text{ins}, l^A q^{A5}, \texttt{i} \rangle$

$\text{del}(4) \xrightarrow{\text{génére}} \langle \text{del}, m^{A6} \rangle$

$\text{ins}(4, \texttt{e}) \xrightarrow{\text{génére}} \langle \text{ins}, m^{A6}, \texttt{e} \rangle$

$\text{ins}(5, \texttt{s}) \xrightarrow{\text{génére}} \langle \text{ins}, t^{A7}, \texttt{s} \rangle$

intègre $\langle \text{ins}, t^{B1}, ! \rangle$

$\text{ins}(4, !) \xrightarrow{\text{génére}} \langle \text{ins}, t^{B1}, ! \rangle$

intègre $\langle \text{ins}, l^A q^{A5}, \texttt{i} \rangle$

intègre $\langle \text{del}, m^{A4} \rangle$

intègre $\langle \text{ins}, m^{A6}, \texttt{e} \rangle$

intègre intègre $\langle \text{ins}, t^{A7}, \texttt{s} \rangle$

---

[5]. Weiss et al., "Logoot : A Scalable Optimistic Replication Algorithm for Collaborative Editing on P2P Networks".