Université de Rennes 1 – 17/06/2003

Éric CARIOU

Contribution à un Processus de Réification d'Abstractions de Communication

IRISA

Contexte

- Une application est composée d'un ensemble d'objets ou composants :
  - Qui ne sont pas autonomes
  - Qui communiquent, collaborent et interagissent entre eux
- Applications distribuées :
  - Mise en avant de la communication, de l'interaction à distance

⇒ *La communication et les interactions entre composants sont des points essentiels*

- Importance de représenter et d'utiliser des abstractions de communication ou d'interaction

Plan

1. Introduction aux abstractions de communication
2. Introduction aux composants de communication (médiums)
3. Le processus de réification d'abstractions de communication
4. Une plateforme d'implémentation de médiums
5. Conclusion et perspectives

Abstraction de communication

- Système, protocole, service de communication ou d'interaction de tout type, de toute nature et de toute complexité
- Exemples d'abstractions de communication :
  - Appel de procédure à distance (RPC)
  - Diffusion d'événements
  - Mémoire partagée
  - Protocole de consensus
  - Système de vote
  - Diffusion de flux vidéo

Abstraction de communication

- Traitement souvent différent selon la phase du cycle de développement du logiciel :
  - À la spécification/conception : abstractions de haut niveau
  - À l'implémentation : abstractions de bas niveau, adaptées au support de communication (intergiciel, plateforme de composants)
- Pas de continuité de la manipulation d'une abstraction de communication
- Plusieurs travaux permettent tout de même de conserver cette continuité, de la spécification à l'implémentation

Travaux traitant des abstractions de communication

- Langages de description d'architecture (ADL) :
  C2 [Medvidovic & Taylor 96], Unicon [Zelesnik & Shaw 96], Wright [Allen & Garlan 97], ...
- Notion de connecteur :
  - Réifie une abstraction de communication
  - Réutilisable
  - Existe souvent à la spécification et à l'implémentation : génération de code ou plateforme d'exécution
- Limites de ces approches :
  - Au niveau spécification, l'abstraction est proche de l'implémentation
  - Pas de possibilité de variantes d'implémentation

Travaux traitant des abstractions de communication

- Catalysis [D'Souza, Wills & Cameron 98] :
  méthodologie et processus de développement d'applications à base de composants (et de connecteurs)
  - Insiste beaucoup sur les collaborations entre composants/objets
  - Offre un processus de raffinement
  - Propose plusieurs niveaux, de la spécification abstraite à l'implémentation
- Limites de Catalysis :
  - Connecteurs de bas niveau à l'implémentation
  - Transformations/raffinements généraux, pas spécialisés pour les abstractions de communication

Contribution de la thèse

- Un processus de réification d'abstractions de communication sous forme de composants logiciels (composants de communication ou médiums)
- Permet de manipuler une abstraction de communication pendant tout le cycle de développement $\Rightarrow$ de la spécification abstraite à l'implémentation
- Composé de 3 éléments :
  - Une méthodologie de spécification de médiums en UML à un niveau abstrait, indépendamment de toute implémentation
  - Une architecture de déploiement de médium
  - Un processus de raffinement $\Rightarrow$ de la spécification abstraite à une ou plusieurs spécifications d'implémentation

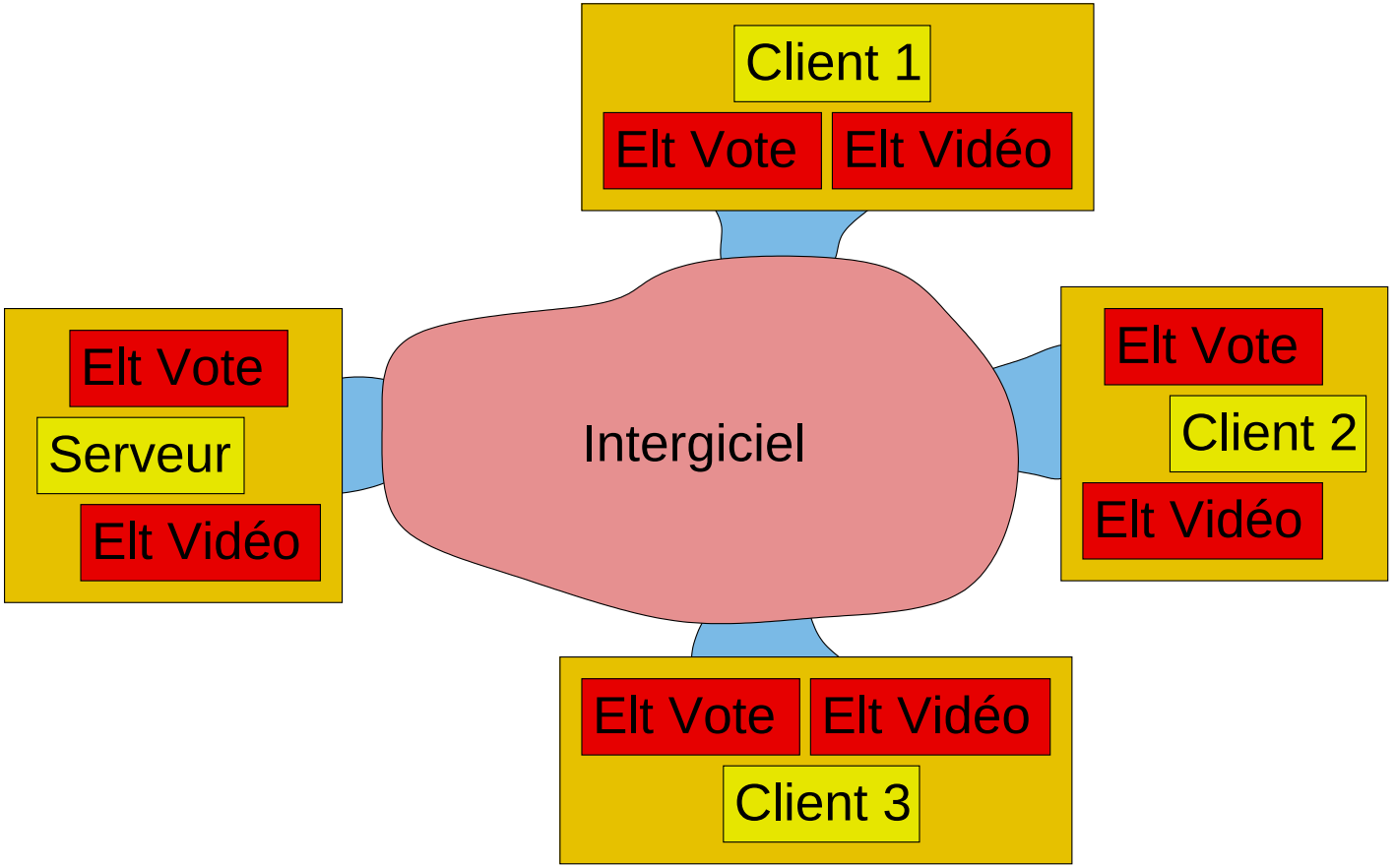
Plan

1. Introduction aux abstractions de communication
2. *Introduction aux composants de communication (médiums)*
   - *Étude d'une architecture d'application*
   - Caractéristiques des composants de communication
3. Le processus de réification d'abstractions de communication
4. Une plateforme d'implémentation de médiums
5. Conclusion et perspectives

Application de vidéo interactive

- Diffusion d'un film vidéo
  - Par un serveur
  - Visualisé par plusieurs clients
- Un vote est lancé à la fin de chaque de film par le serveur
- Les clients votent
- Le film le plus choisi est celui qui est diffusé

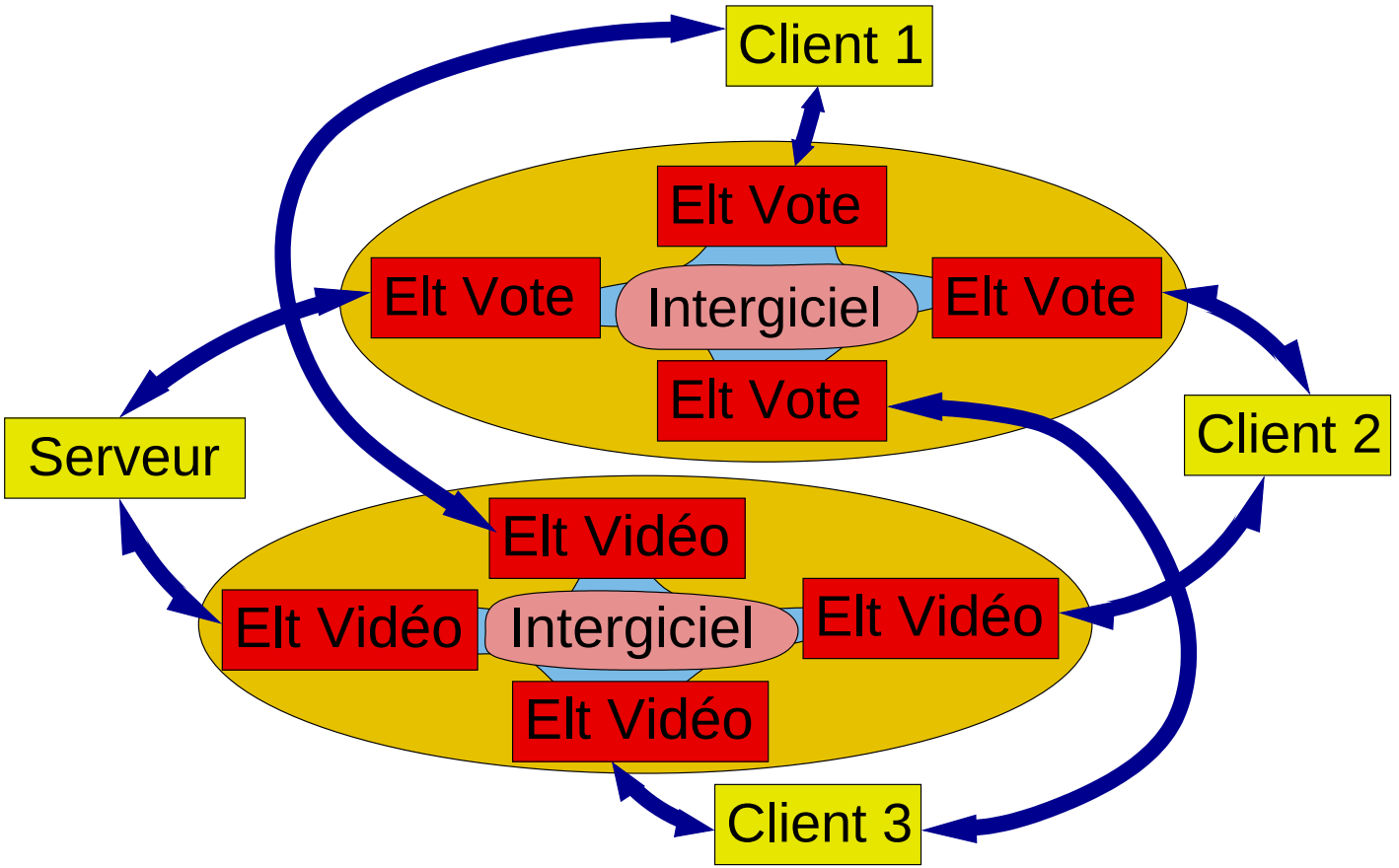
Architecture « classique »



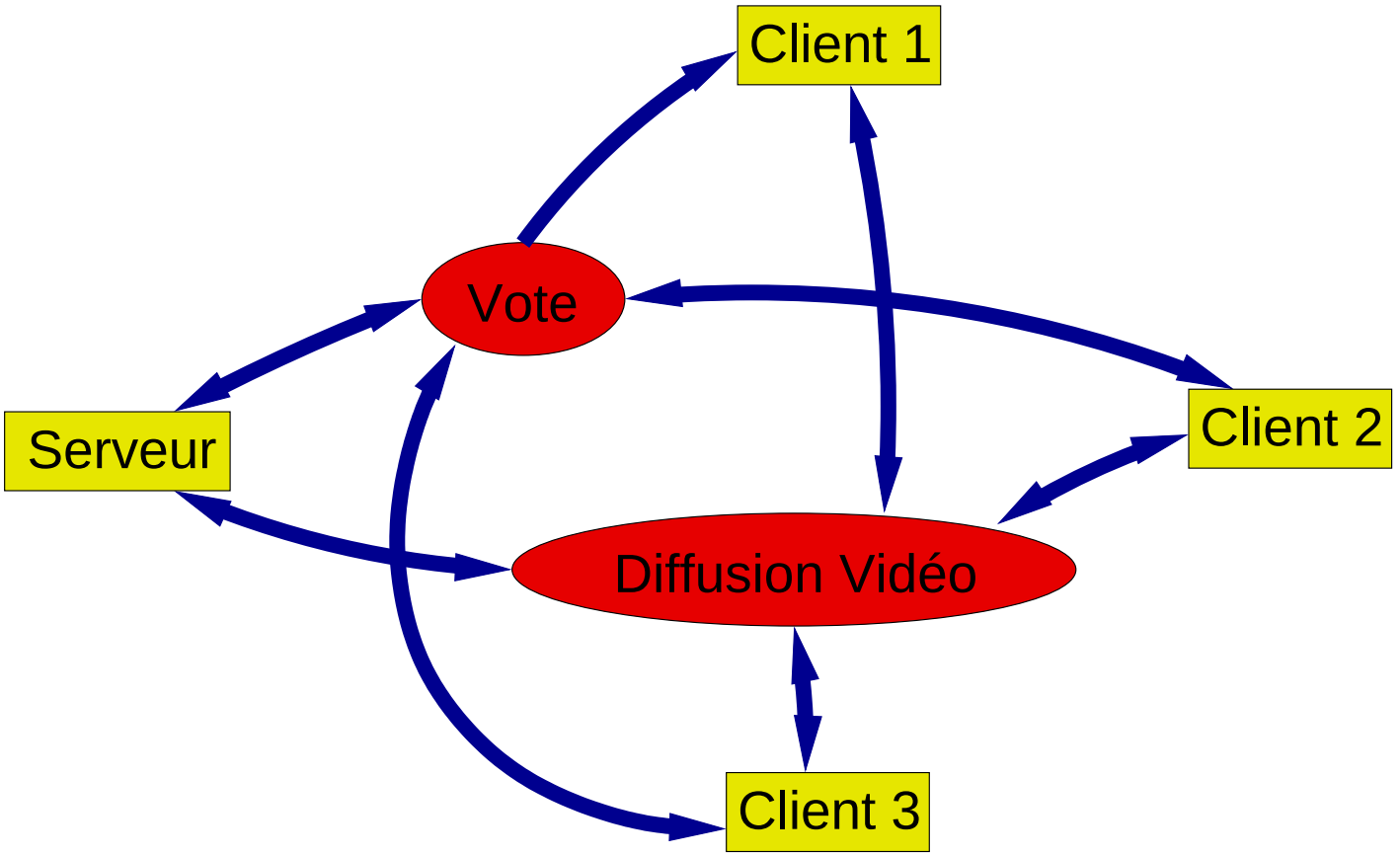
Critique de l'architecture

- Restrictions de cette approche classique :
  - Mélange des interactions (vote, diffusion de flux) avec le reste
  - Peut poser des problèmes d'évolutivité et est peu réutilisable
- Solution : rassembler ces « morceaux » dans un composant
  - *un composant de communication (ou médium)*

Nouvelle architecture



Mise en avant des médiums



Plan

1. Introduction aux abstractions de communication
2. *Introduction aux composants de communication (médiums)*
   - Étude d'une architecture d'application
   - *Caractéristiques des composants de communication*
3. Le processus de réification d'abstractions de communication
4. Une plateforme d'implémentation de médiums
5. Conclusion et perspectives

Les composants logiciels

- Entité logicielle ayant comme caratéristiques :
  - Spécifie clairement ses interfaces de services offerts et requis
  - Sujet à composition pour former d'autres composants ou des applications
  - Substituable et réutilisable
- Un composant existe pendant tout le cycle de développement :
  - À la spécification/conception : définition d'un contrat (liste des services et de leur sémantique)
  - À l'implémentation : entité instantiable

Les composants de communication (médiums)

- Médium = réification d'une abstraction de communication dans un composant $\Rightarrow$ même propriétés que pour un composant :
  - Existe à la spécification et à l'implémentation
  - Réutilisable
  - Substituable (autorise des variantes d'implémentation)
  - Composable avec d'autres composants (métier ou fonctionnels) pour réaliser leur communication
  - Séparation des communications des autres préoccupations
  - ➥ *permet de conserver l'unité et la cohérence d'une abstraction de communication pendant tout le cycle de développement*

Plan

1. Introduction aux abstractions de communication
2. Introduction aux composants de communication (médiums)
3. *Le processus de réification d'abstractions de communication*
   - *Méthodologie de spécification de médiums en UML*
   - Architecture de déploiement de médium
   - Processus de raffinement de spécifications de médium
4. Une plateforme d'implémentation de médiums
5. Conclusion et perspectives

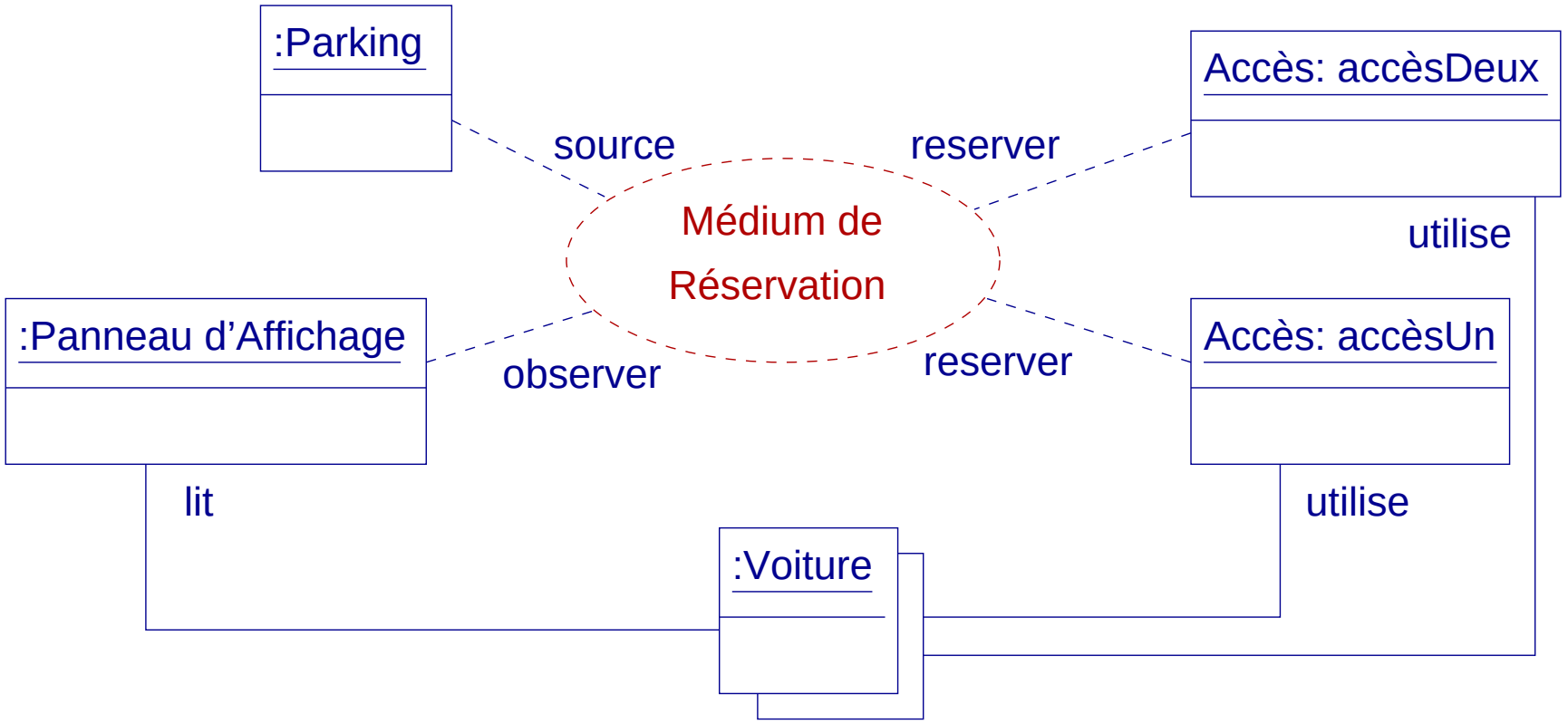
Méthodologie de spécification en UML

- But : définir précisément le contrat d'usage d'un composant de communication à un niveau abstrait
  - *Liste des services offerts et requis avec leur sémantique*
- En UML (Unified Modeling Language) $\Rightarrow$ standard de modélisation en génie logiciel
  - Un médium est représenté par une collaboration UML
  - Contraintes OCL (Object Constraint Language) : invariants de classes, pré et post-conditions sur méthodes
  - Diagrammes d'état

Gestion des entrées d'un parking

- Un parking est composé de places
- Les voitures entrent par deux accès
- Une voiture occupe une place identifiée
- Un panneau indique le nombre de places disponibles

Gestion des entrées d'un parking

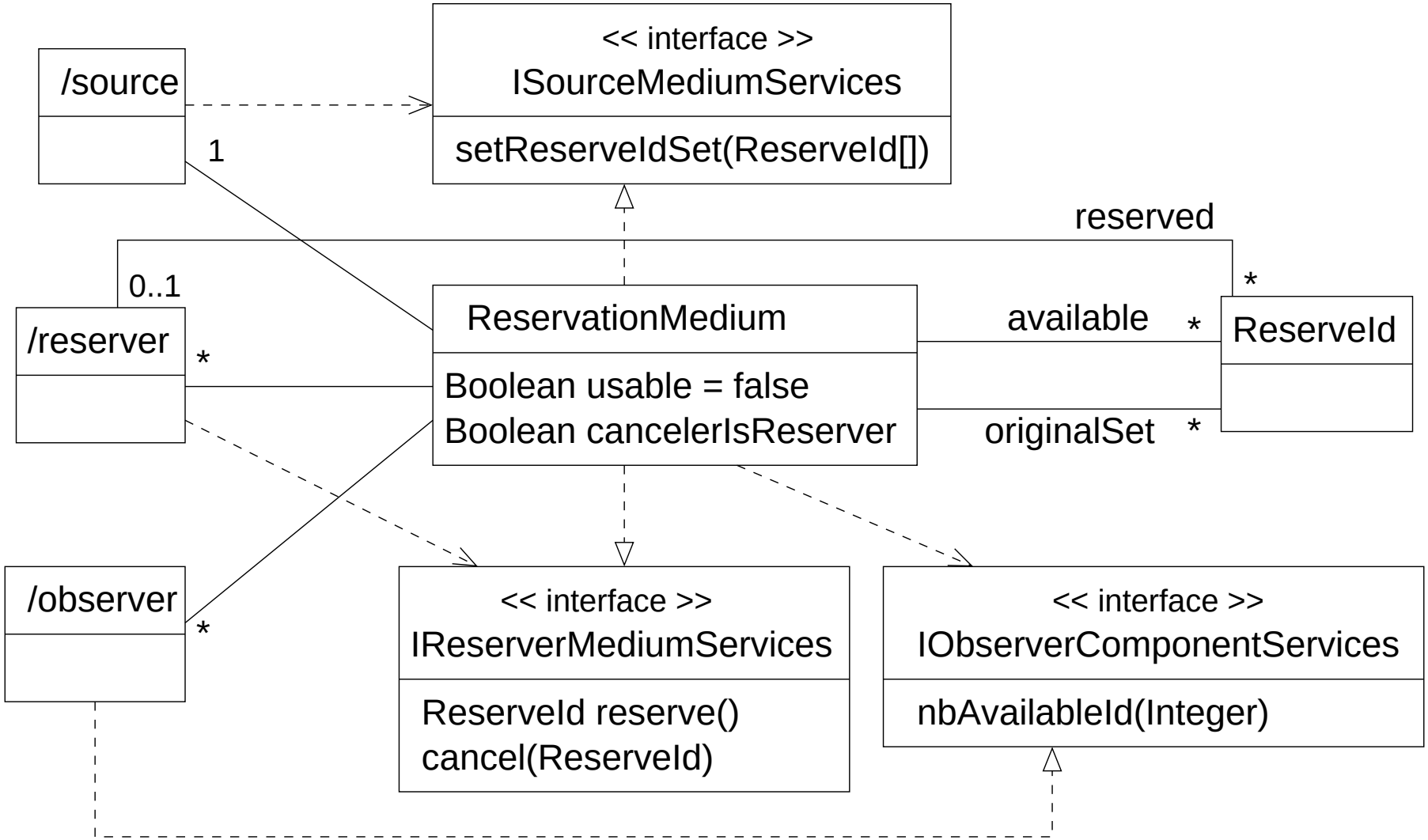


- Les 4 composants inter-agissent via une collaboration UML $\Rightarrow$ via un médium

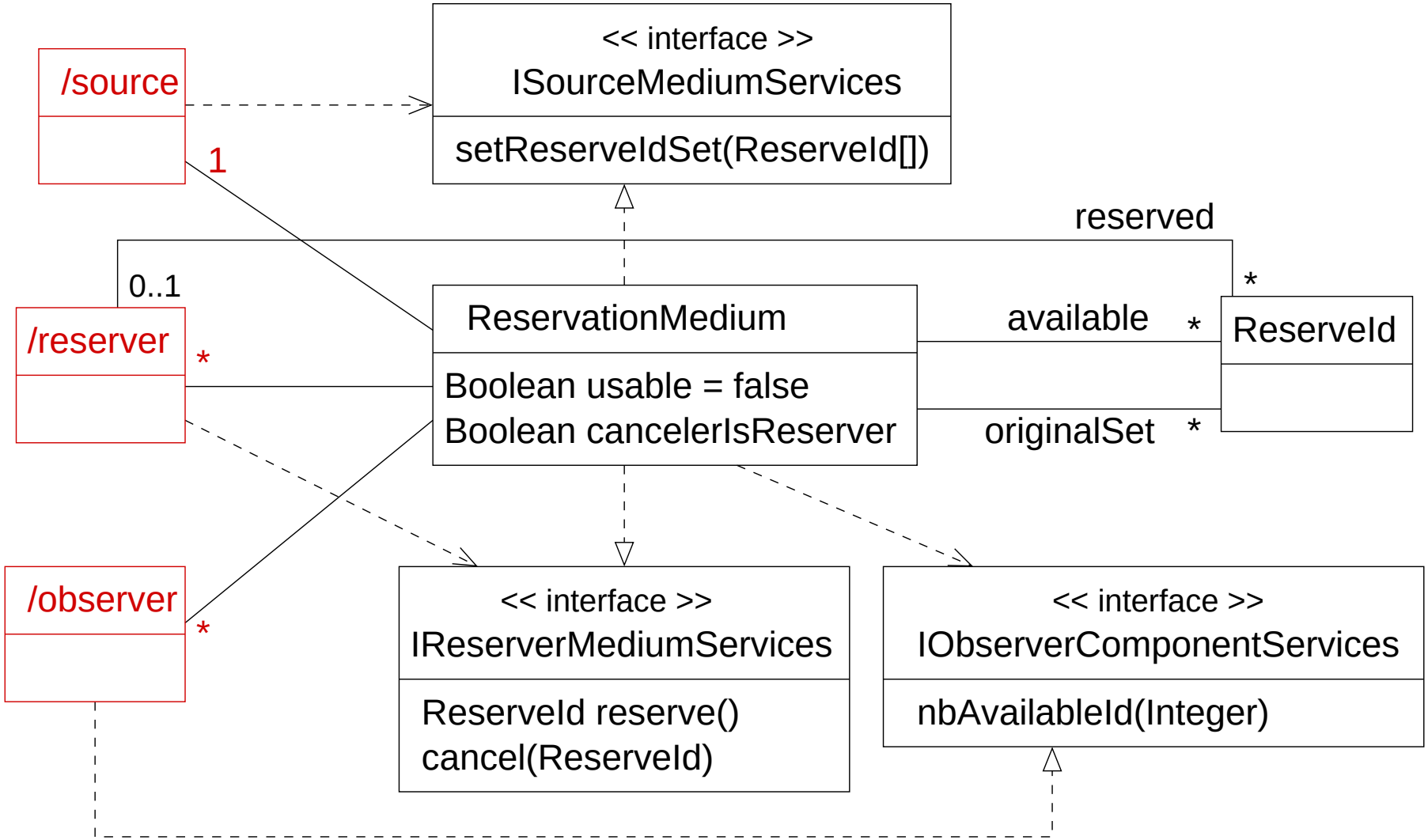
Le médium de réservation

- Sert à réserver des identificateurs :
  - Un composant *source* définit un ensemble d'identificateurs
  - Des composants *reserver* retirent des identificateurs de l'ensemble et en replacent
  - Des composants *observer* sont informés du nombre d'identificateurs disponibles dès qu'il change

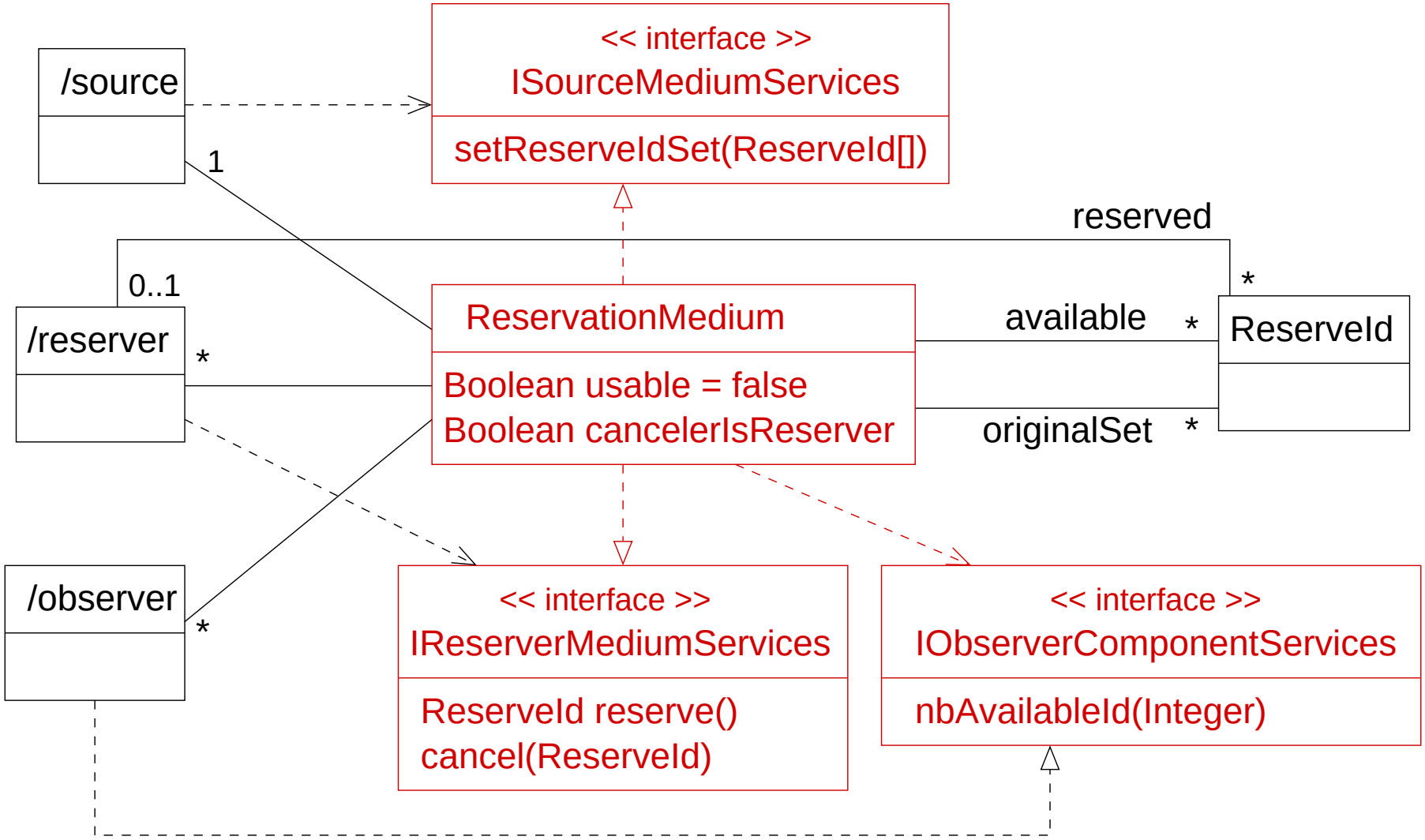
Le médium de réservation



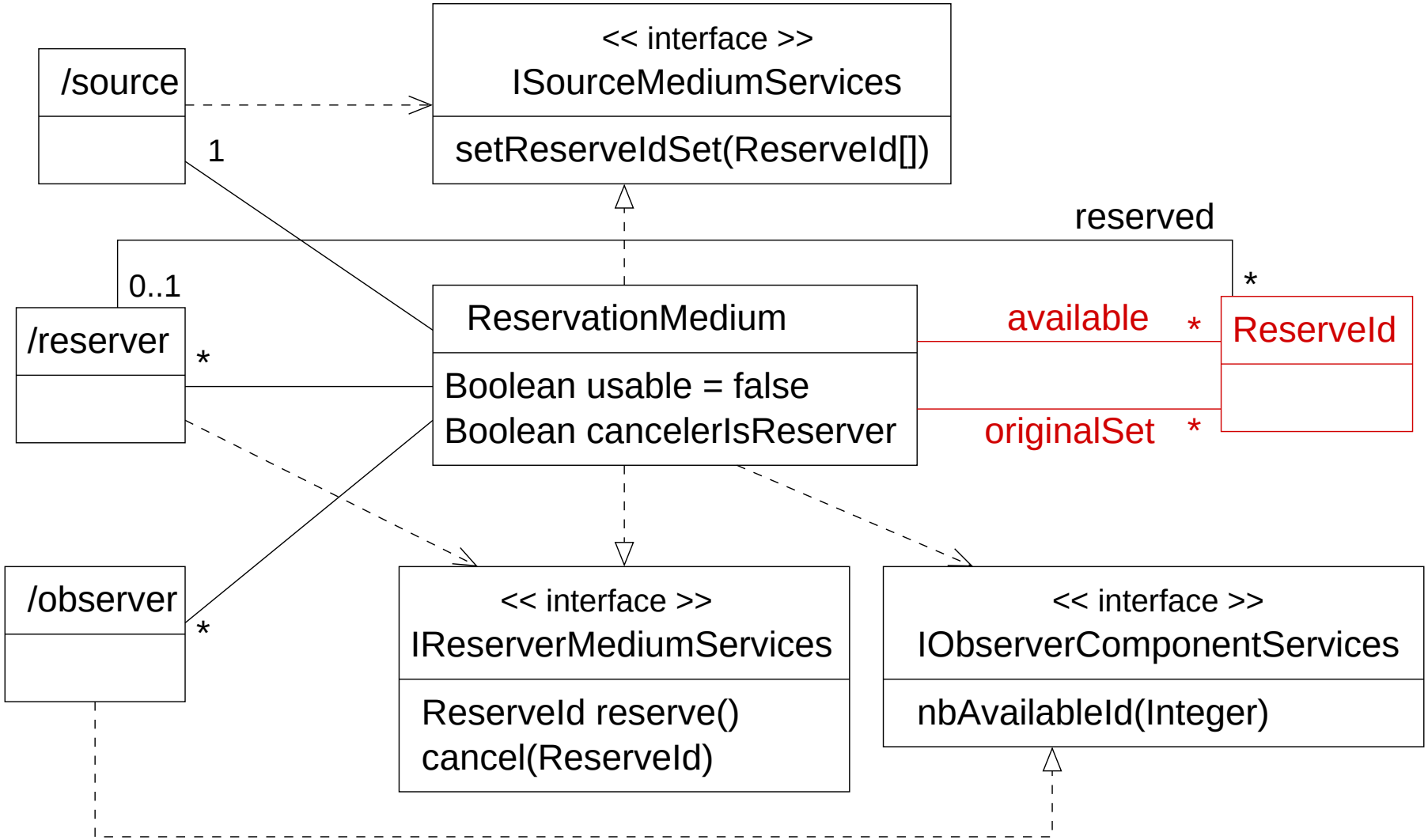
Le médium de réservation



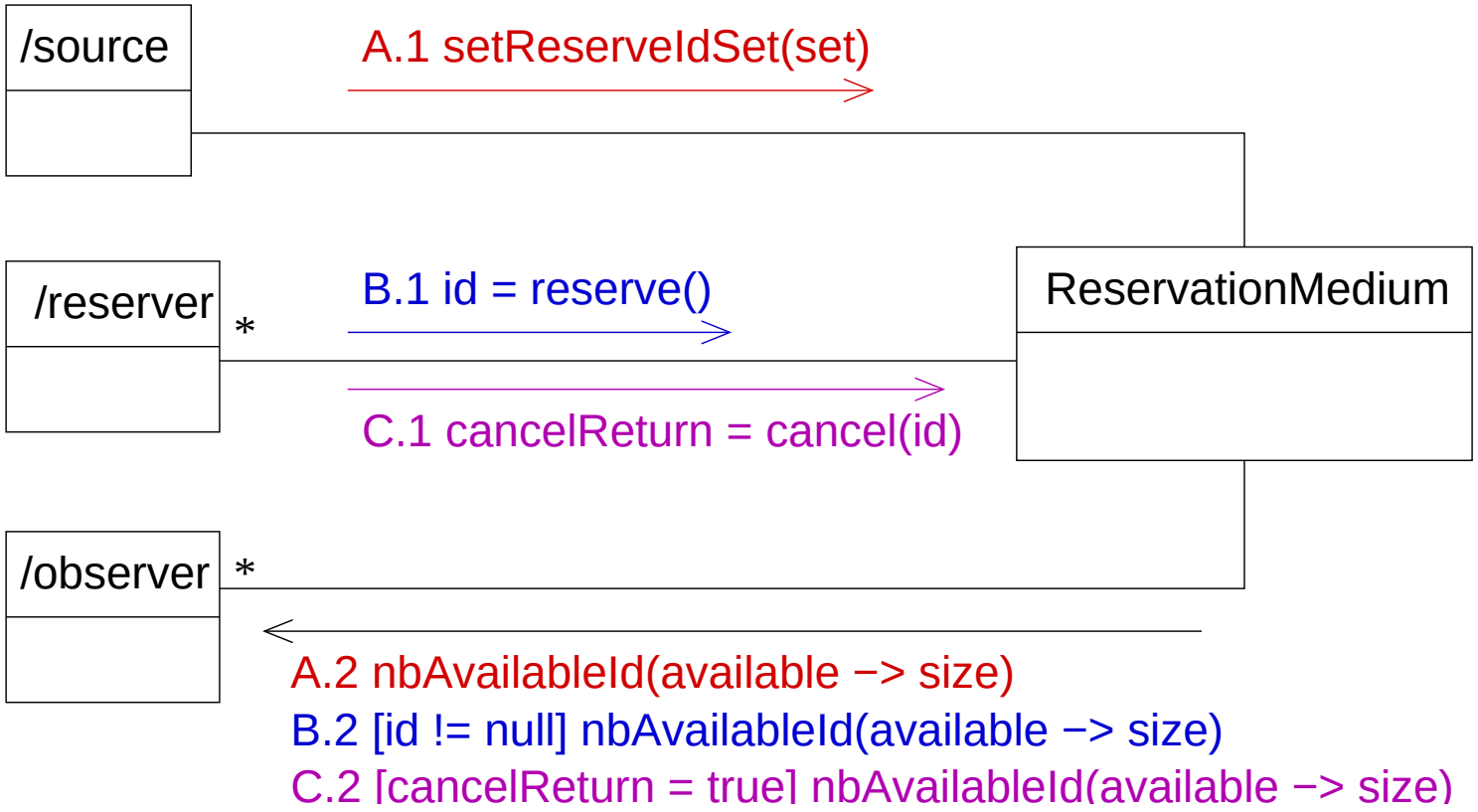
Le médium de réservation



Le médium de réservation



Vue dynamique de la collaboration



- Spécifie quand notifier les observateurs du changement du nombre d'identificateurs.

Contraintes OCL

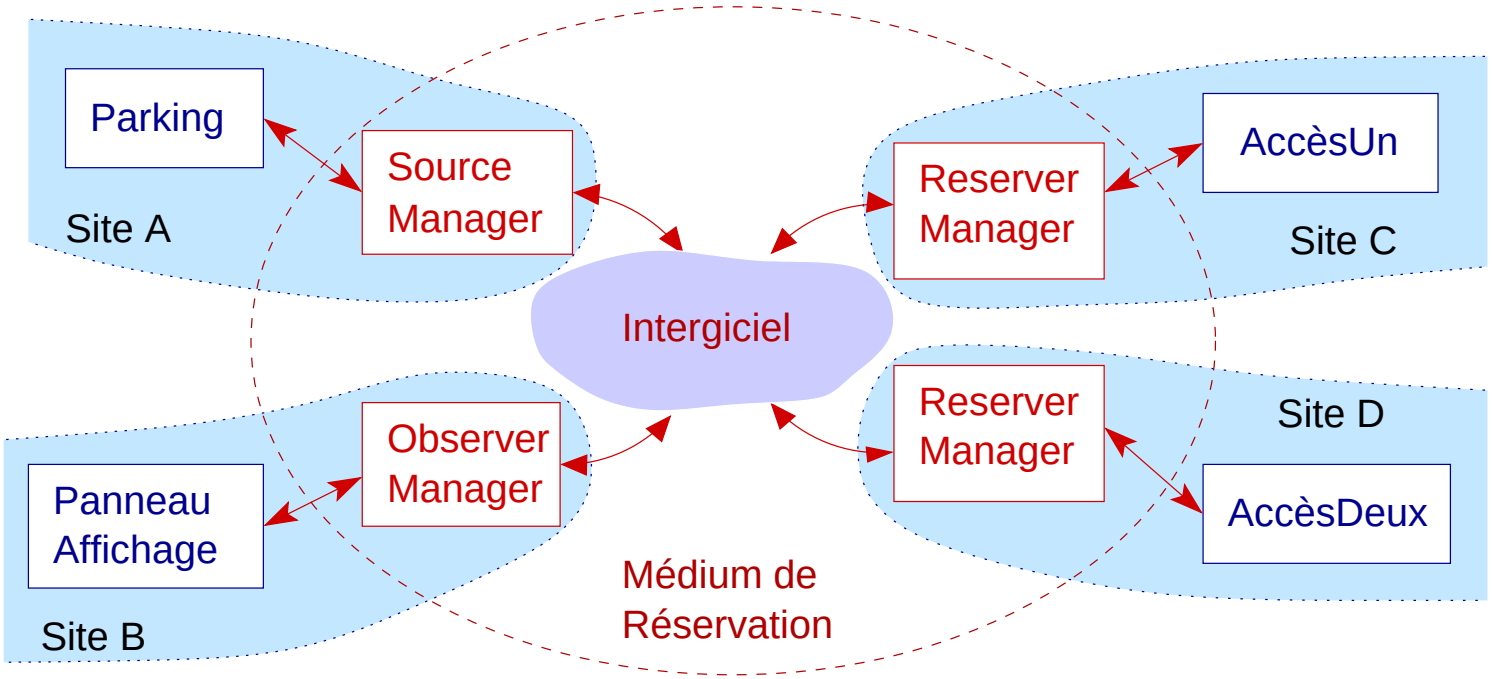
context ReservationMedium : :reserve() : ReserveId
pre : usable = true -- le médium est dans un état utilisable
post :
if available –> isEmpty -- pas d'identificateur disponible
then result = null
else -- un identificateur est retourné
originalSet –> includes(result)
and available@pre –> includes(result)
and available –> excludes(result)
and *caller*.reserved –> includes(result)
endif

➤ Spécification du service `reserve` permettant de faire une réservation d'identificateur

Plan

1. Introduction aux abstractions de communication
2. Introduction aux composants de communication (médiums)
3. *Le processus de réification d'abstractions de communication*
   - Méthodologie de spécification de médiums en UML
   - *Architecture de déploiement de médium*
   - Processus de raffinement de spécifications de médium
4. Une plateforme d'implémentation de médiums
5. Conclusion et perspectives

Architecture de déploiement de médium



- Un gestionnaire local à chaque composant connecté
- Adapté à la réalisation de diverses abstractions de communication
- Autorise des variantes d'implémentations

Plan

1. Introduction aux abstractions de communication
2. Introduction aux composants de communication (médiums)
3. *Le processus de réification d'abstractions de communication*
   - Méthodologie de spécification de médiums en UML
   - Architecture de déploiement de médium
   - *Processus de raffinement de spécifications de médium*
4. Une plateforme d'implémentation de médiums
5. Conclusion et perspectives

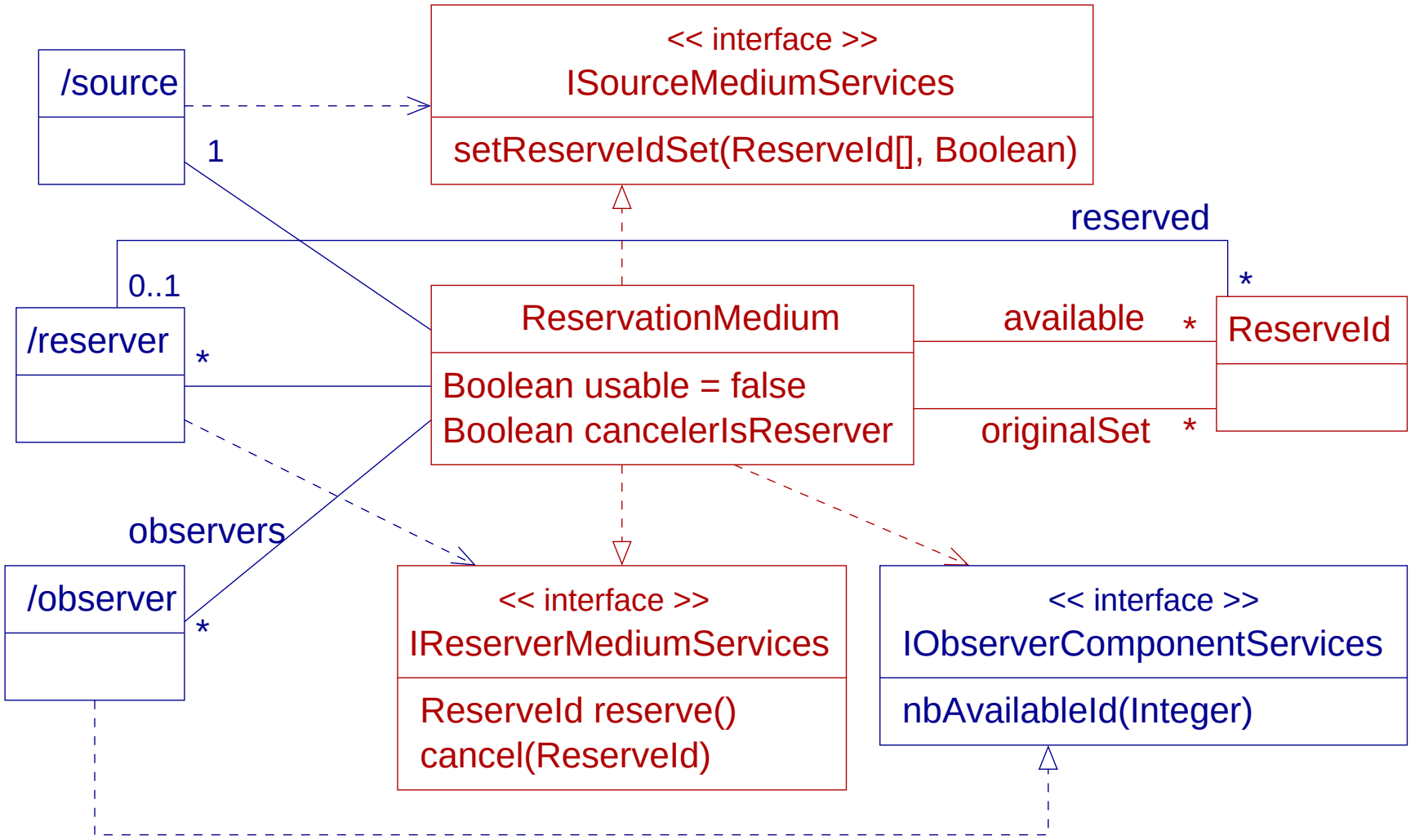
Le processus de raffinement

- But : transformer une spécification abstraite en une spécification d'implémentation
  - En tenant compte de l'architecture de déploiement
  - Plusieurs spécifications d'implémentation pour une même spécification abstraite
  - Respect du contrat du médium lors des transformations
  - Chaque étape transforme entièrement la spécification (collaboration, contraintes OCL, diagrammes d'état)
- En pratique :
  - Remplacer la classe « Medium » par des gestionnaires de rôle

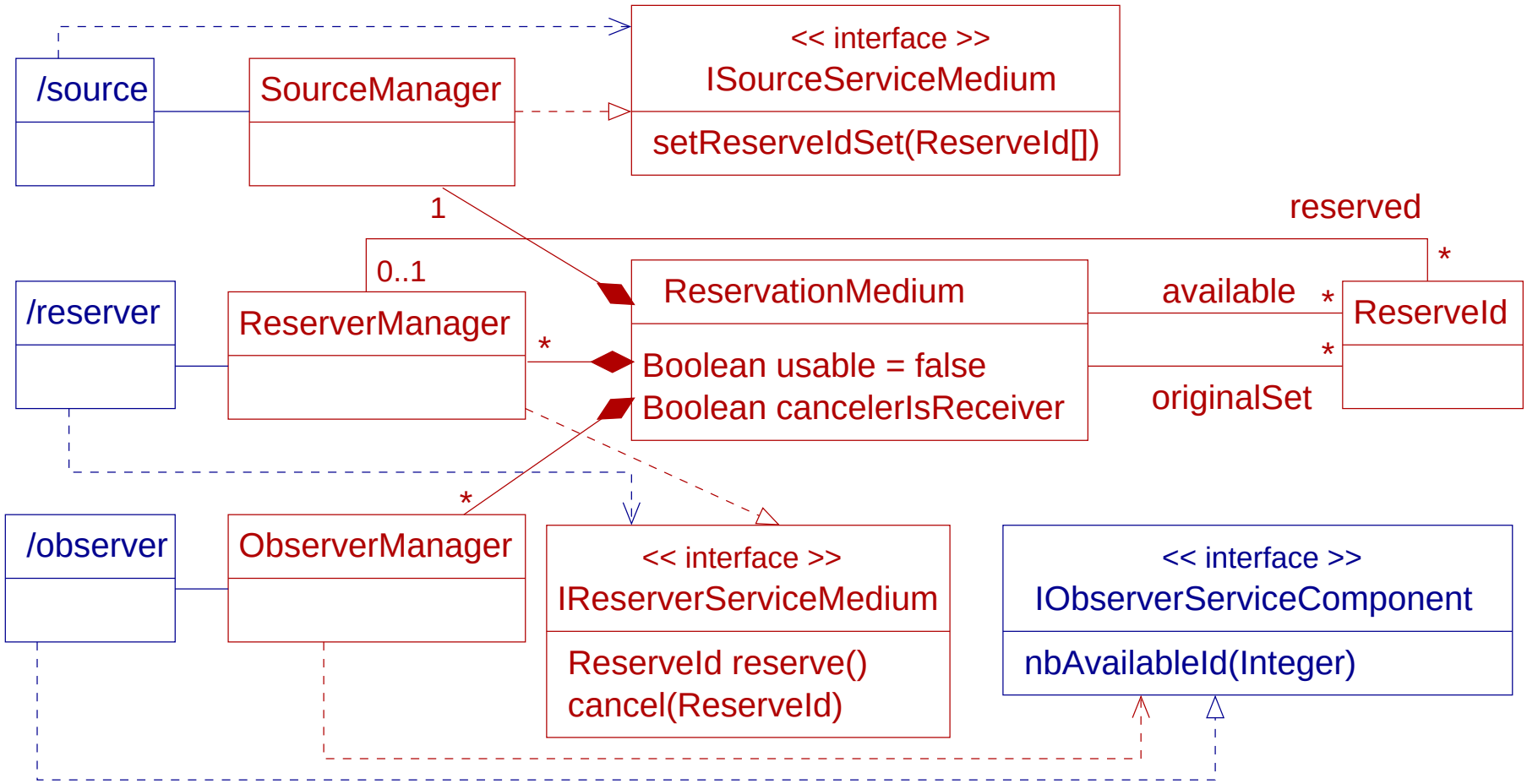
Les étapes du processus

- 3 étapes :
  - La classe « Medium » = aggrégation des gestionnaires de rôle
  - Disparation de cette classe, spécification avec uniquement les gestionnaires $\Rightarrow$ choix de conception, d'implémentation
  - Choix de déploiement pour chaque spécification d'implémentation
- Exemple : médium de réservation, deux choix d'implémentation
  - L'ensemble des identificateurs géré par un seul gestionnaire (version centralisée)
  - L'ensemble des identificateurs partagé et distribué entre plusieurs gestionnaires (version distribuée)

Étape 0 : spécification abstraite

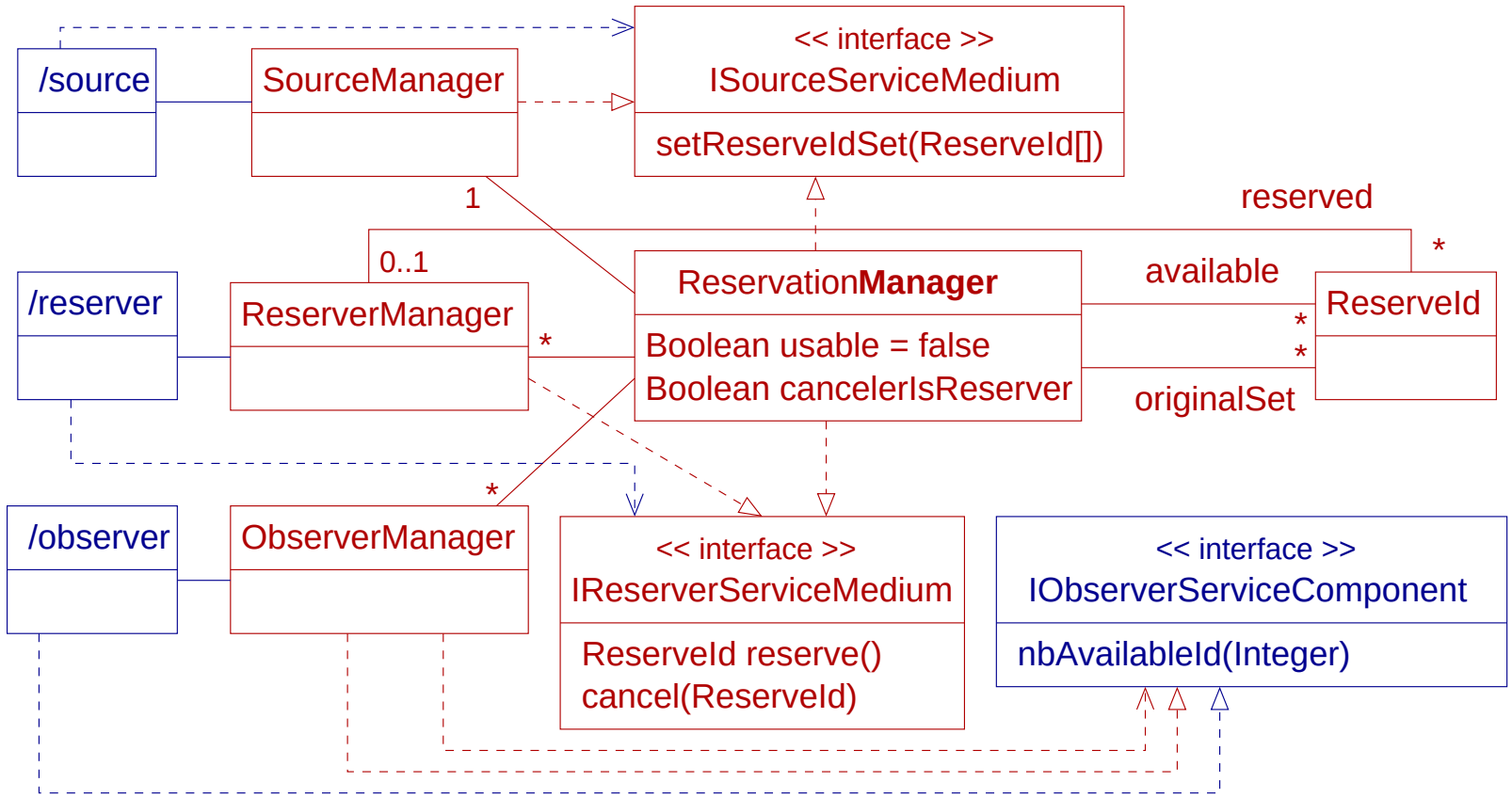


Étape 1 : introduction des gestionnaires



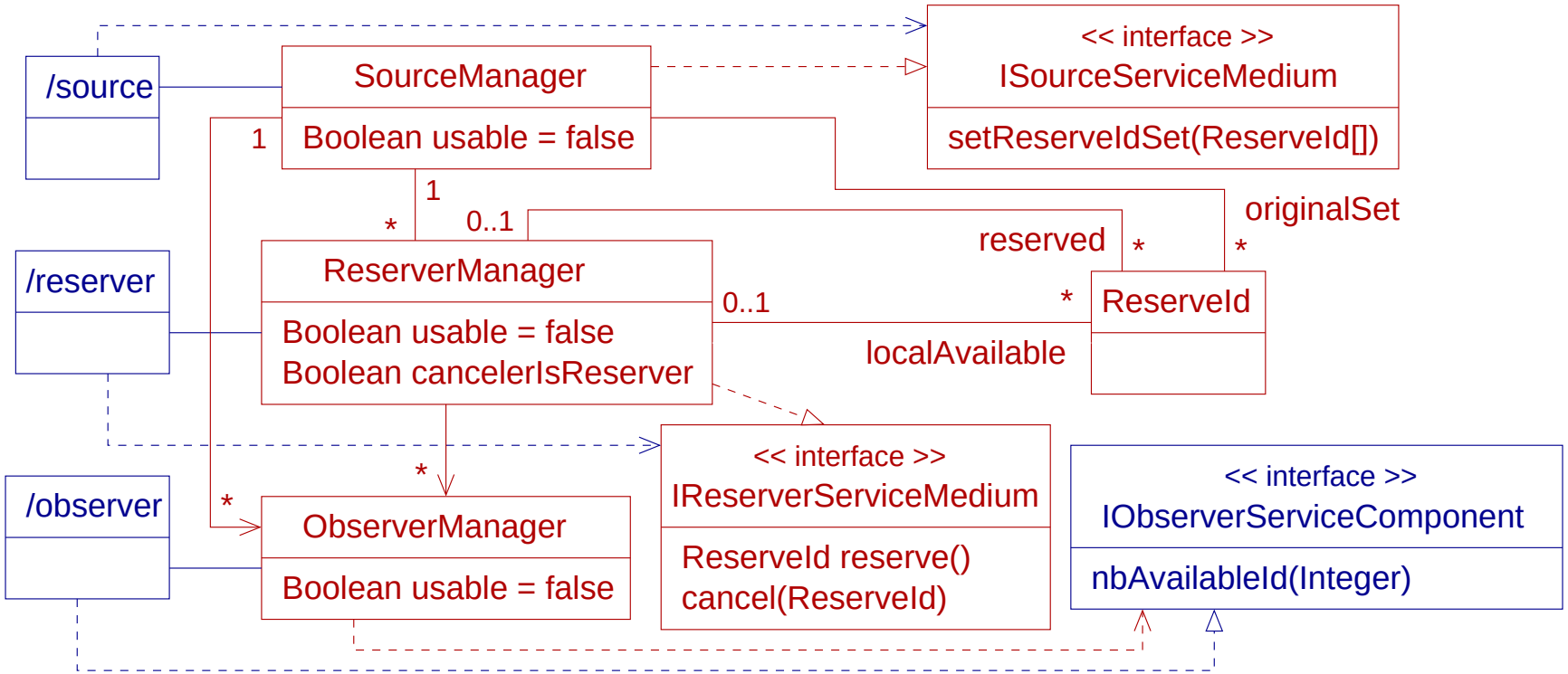
- Contrat totalement respecté par cette transformation

Étape 2 : choix de conception (centralisé)



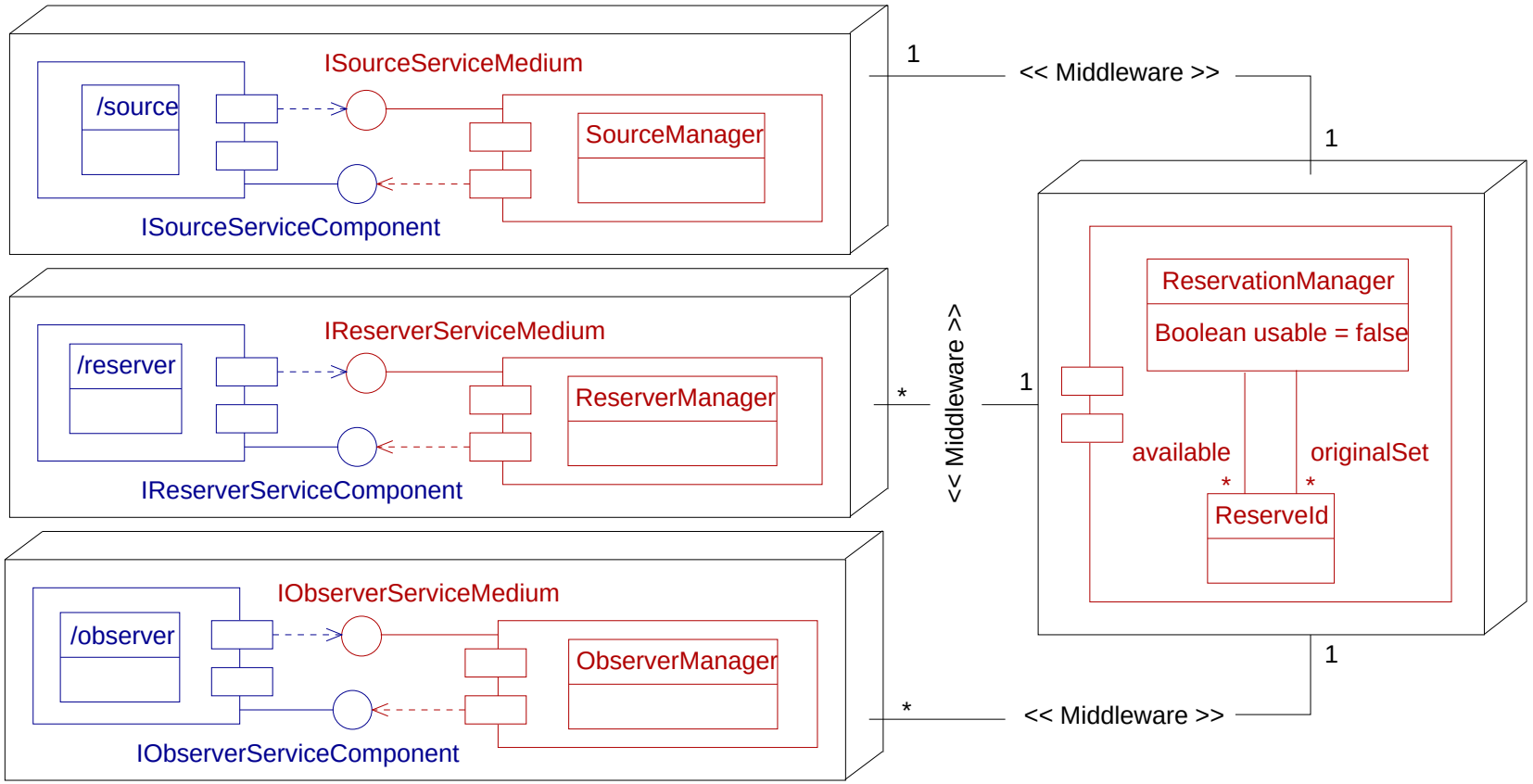
- Nouveau gestionnaire qui gère tous les identificateurs
- Disparition de la classe représentant le médium

Étape 2 : choix de conception (distribué)



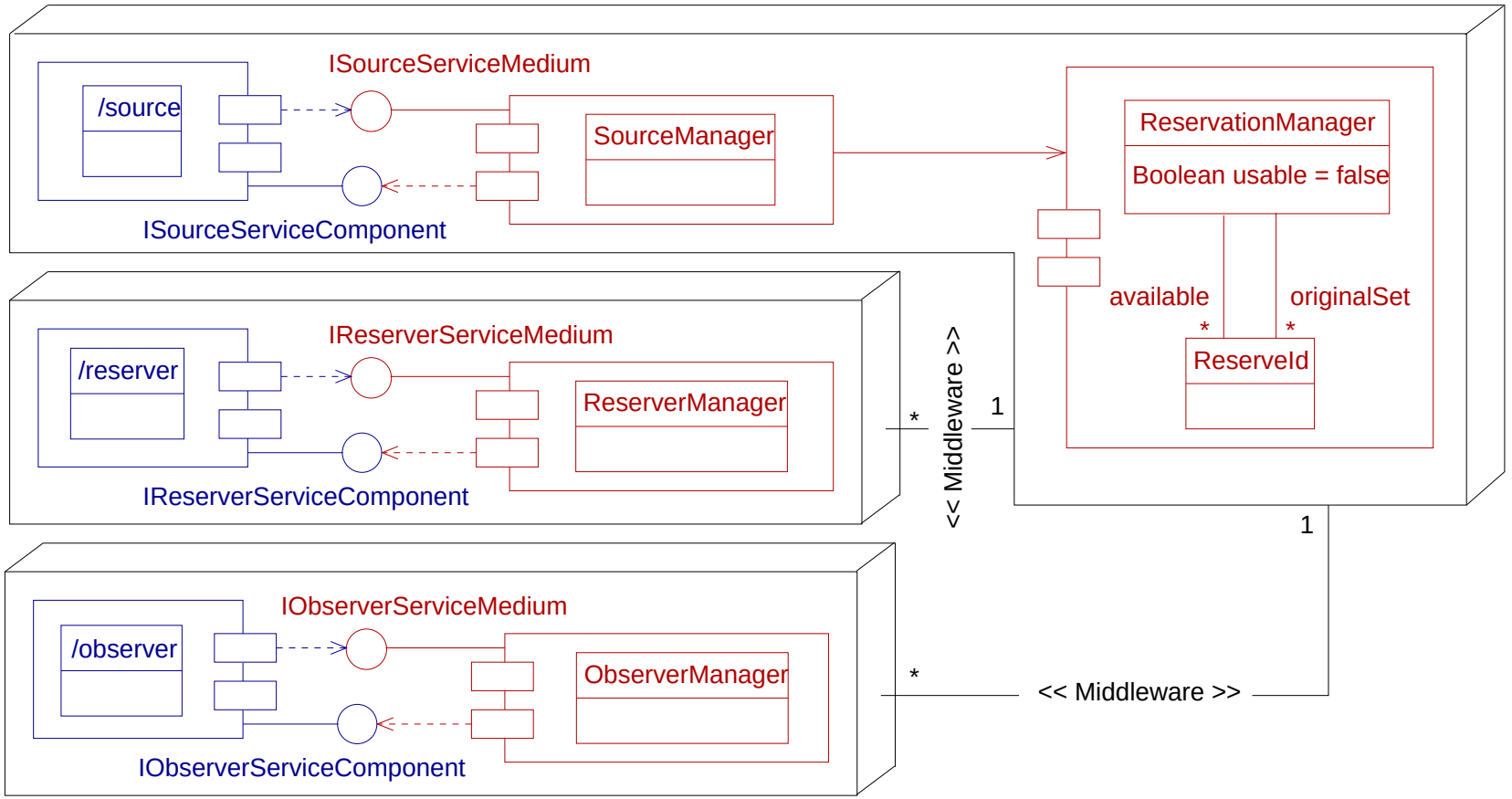
- Les identificateurs sont partagés entre tous les gestionnaires de réserveur
- Difficulté de s'assurer du respect du contrat

Étape 3 : choix de déploiement



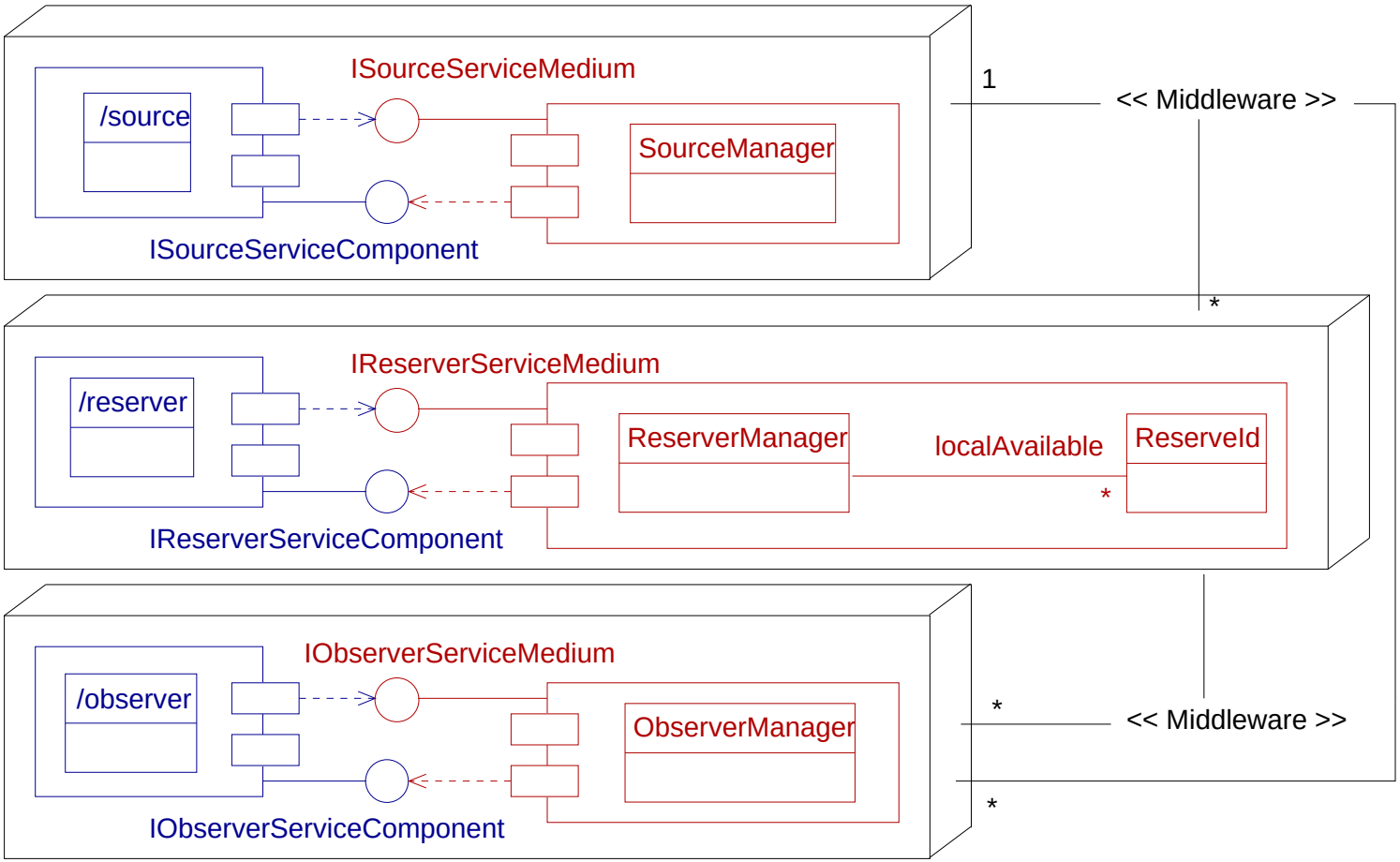
- Version centralisée : gestionnaire de réservation à part
- Pas de modification du contrat

Étape 3 : choix de déploiement



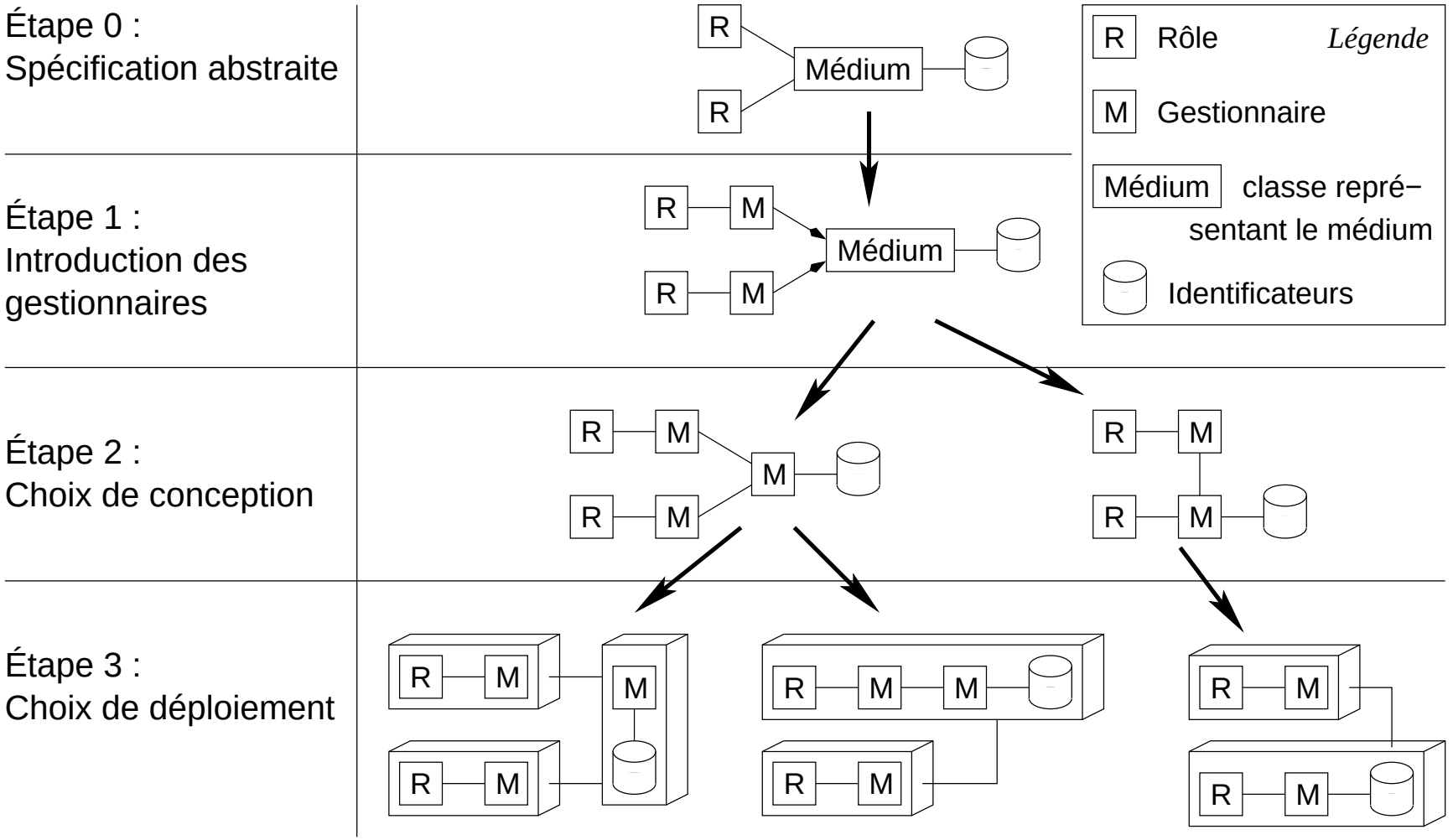
- Version centralisée : gestionnaire de réservation avec le gestionnaire de source

Étape 3 : choix de déploiement



- Version distribuée : un seul déploiement

Résumé des différentes étapes



Plan

1. Introduction aux abstractions de communication
2. Introduction aux composants de communication (médiums)
3. Le processus de réification d'abstractions de communication
4. *Une plateforme d'implémentation de médiums*
5. Conclusion et perspectives

Une plateforme d'implémentation de médiums

- Écrite en Java, distribuée sous forme de logiciel libre (licence GPL)
- Deux buts :
  - Aider à la réalisation de médiums : squellettes de gestionnaires et primitives pour leur communication
  - Support d'exécution, dans un contexte distribué
- Permet de gérer une partie du contrat :
  - La topologie : contraintes sur le nombre de composants connectés
  - L'utilisabilité : dans quelles conditions le médium est utilisable

Expérimentations avec cette plateforme

- Réalisation de l'application de vidéo interactive :
  - Médium de diffusion de vidéo complexe
  - La complexité de l'application est dans les médiums
- Réalisation du médium de réservation en plusieurs versions :
  - Transparence aux variantes d'implémentation pour les composants utilisant le médium
- Bilan sur l'implémentation de médiums :
  - Ne simplifie pas forcément l'implémentation d'abstrations de communication
  - Permet facilement de réutiliser une abstraction et d'utiliser des variantes d'implémentations

Conclusion

- Contribution : un processus de réification d'abstractions de communication sous forme de composants logiciels
  - Manipulation d'une même abstraction pendant tout le cycle de développement
  - Séparation des communications des autres préoccupations
  - Réutilisation d'une abstraction
  - Variantes d'implémentation pour une même abstraction
- Composé de 3 éléments :
  - Une méthodologie de spécification abstraite
  - Une architecture de déploiement
  - Un processus de raffinement

Perspectives

- Phase d'analyse (avant la spécification) :
  - Recherche des abstractions de communication « pertinentes »
  - Recherche de la frontière et de la responsabilité d'un médium
- Processus de raffinement :
  - Définition de patrons de distribution, de transformation de spécifications
  - Aides, guides pour passer à la spécification d'implémentation