Conception et réalisation d'un intergiciel schizophrène pour la mise en œuvre de systèmes répartis interopérables

Thèse de doctorat de l'Université Pierre et Marie Curie

Thomas QUINOT

sous la direction de

Fabrice KORDON et Laurent PAUTET

Université Pierre et Marie Curie

École nationale supérieure des télécommunications

Plan

- Intergiciels et interopérabilité
- Architecture schizophrène
- PolyORB, un intergiciel schizophrène
- Conclusion et perspectives

Intergiciels et interopérabilité

Répartition, intergiciels et interopérabilité

Modèle de répartition : formalisme de description des composants applicatifs et de leurs interactions

- passage de messages
- appel de sous-programmes à distance
- objets répartis

Intergiciel : mise en œuvre d'un modèle

$\Rightarrow$ transparence + interopérabilité entre architectures

Interopérabilité entre modèles :

- réutiliser des services/composants existants
- mutualiser les efforts de développement

Objectif

Les intergiciels

- rendent les applications réparties indépendantes des architectures/systèmes d'exploitation
- *mais* introduisent de nouvelles dépendances :
  - choix d'un modèle de répartition
  - choix d'une mise en œuvre de ce modèle

Objectif

Les intergiciels

- rendent les applications réparties indépendantes des architectures/systèmes d'exploitation
- *mais* introduisent de nouvelles dépendances :
  - choix d'un modèle de répartition
  - choix d'une mise en œuvre de ce modèle

↓

Paradoxe de l'intergiciel

Problématique M2M (*Middleware-to-Middleware*)

Notre objectif :

répondre efficacement au paradoxe de l'intergiciel par l'interopérabilité entre modèles + intergiciels.

État de l'art

- passerelles statiques ?
  - explosion combinatoire
  - coût de l'agrégation d'intergiciels
  - point de passage obligé
    $\rightarrow$ affaiblit les performances et la sûreté
- intergiciels interopérables :
  - protocole fédérateur ?
    - pas toujours possible
  - architecture générique ?
    - instanciation d'une même architecture de base sous forme d'une *personnalité*
    - mais les architectures génériques existantes n'offrent pas l'interopérabilité

Architecture schizophrène

Fonctions récurrentes des intergiciels

Nous identifions les fonctions réalisées par les intergiciels :

- adressage
- transport
- protocole
- représentation
- liaisons
- activation
- exécution

Ces fonctions sont accessibles à travers :

- des interfaces applicatives (objets locaux)
- des interfaces protocolaires (intergiciels distants)

Traits architecturaux retenus

générique

adaptation des fonctions à un modèle,

décomposition modulaire

Traits architecturaux retenus

générique

adaptation des fonctions à un modèle,
décomposition modulaire

configurable

besoins de l'application,
ressources de l'environnement

Traits architecturaux retenus

générique

adaptation des fonctions à un modèle,
décomposition modulaire

configurable

besoins de l'application,
ressources de l'environnement

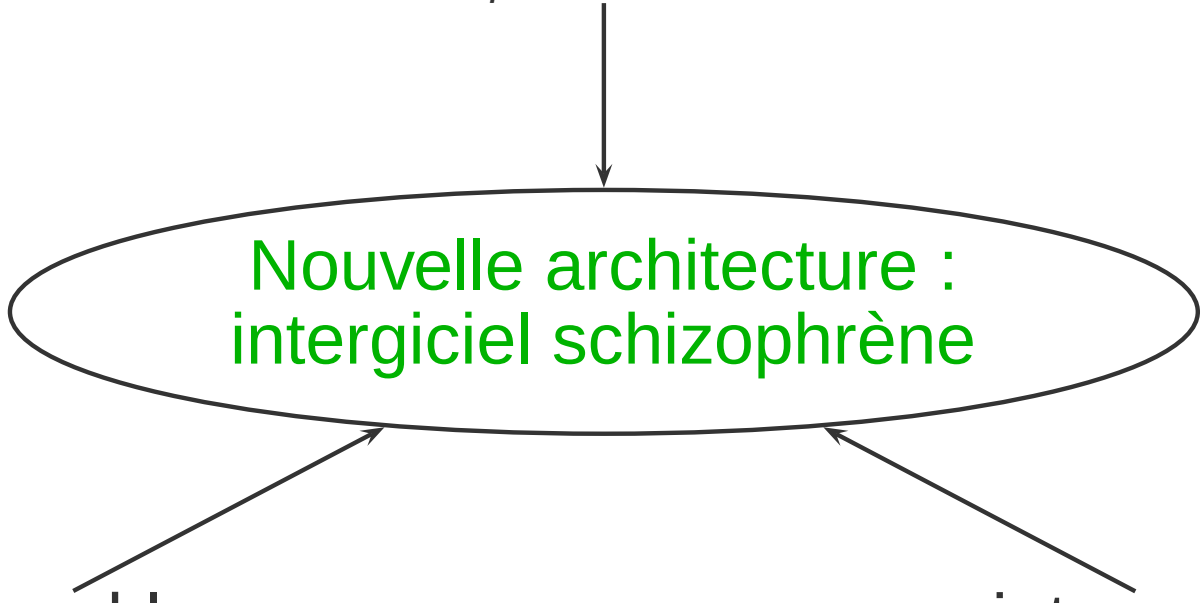
interopérable

passerelles dynamiques
entre personnalités

Traits architecturaux retenus

générique

adaptation des fonctions à un modèle,
décomposition modulaire



configurable

besoins de l'application,
ressources de l'environnement

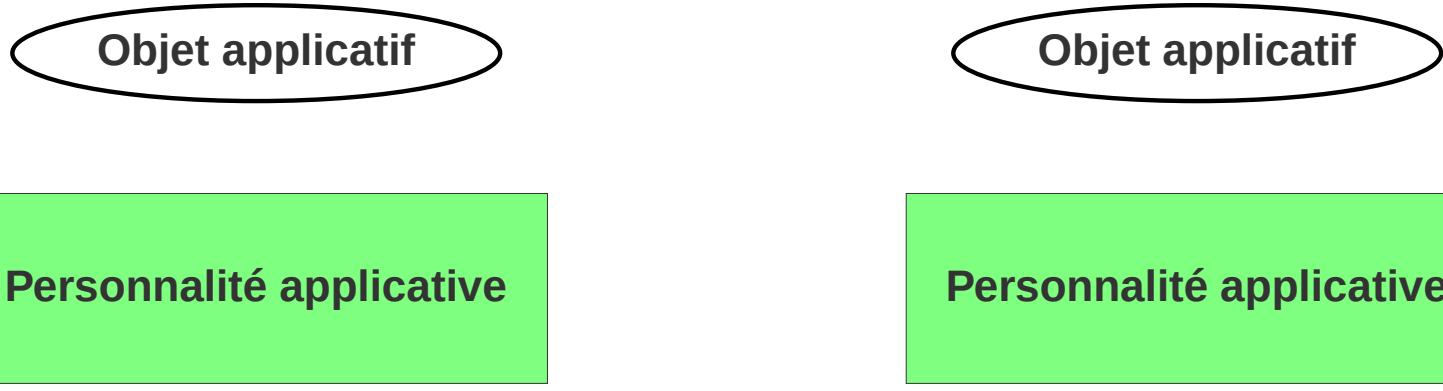
interopérable

passerelles dynamiques
entre personnalités

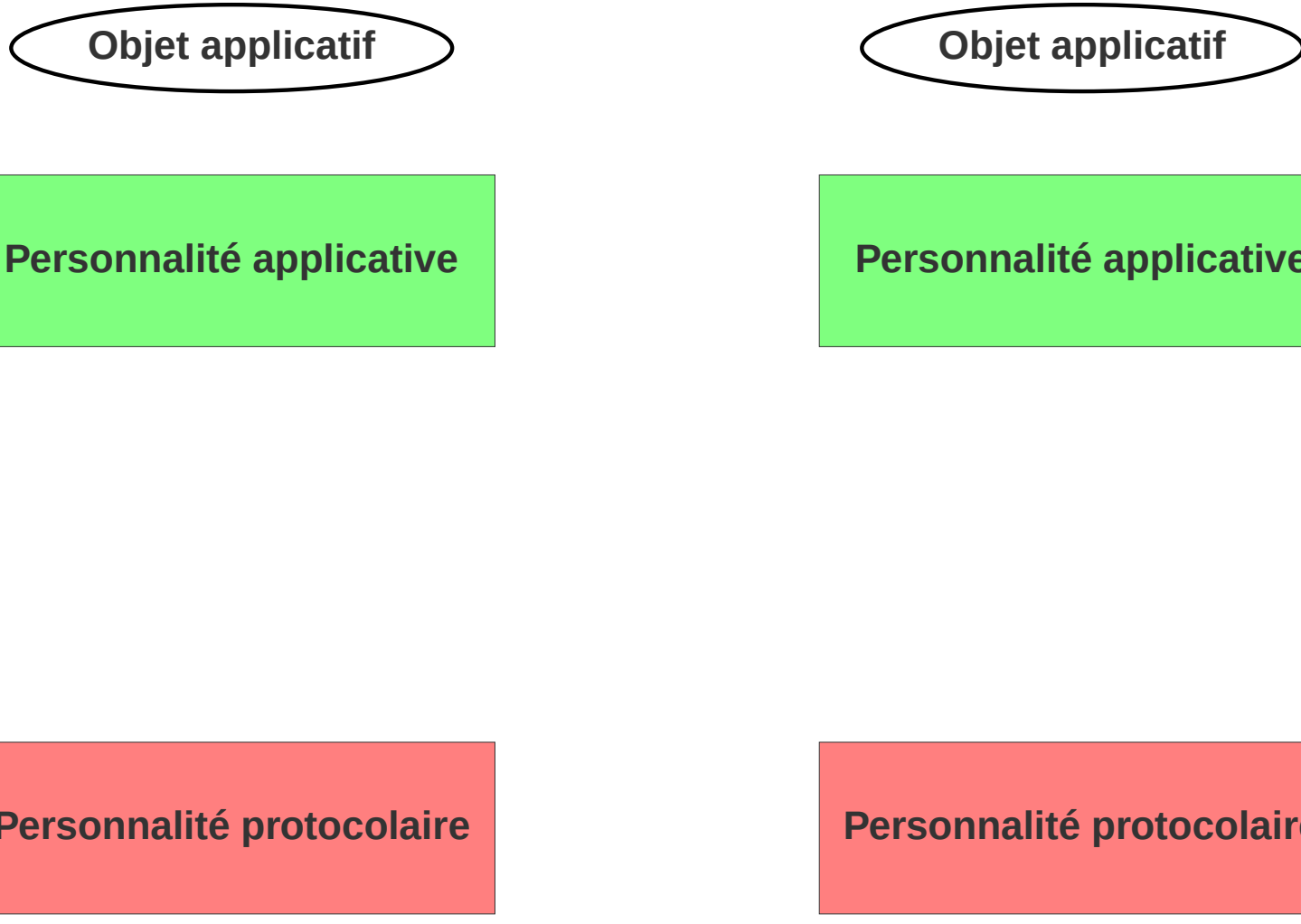
Définition : intergiciel schizophrène

Un intergiciel capable de disposer, simultanément, de plusieurs personnalités applicatives et protocolaires et de les faire interagir efficacement.

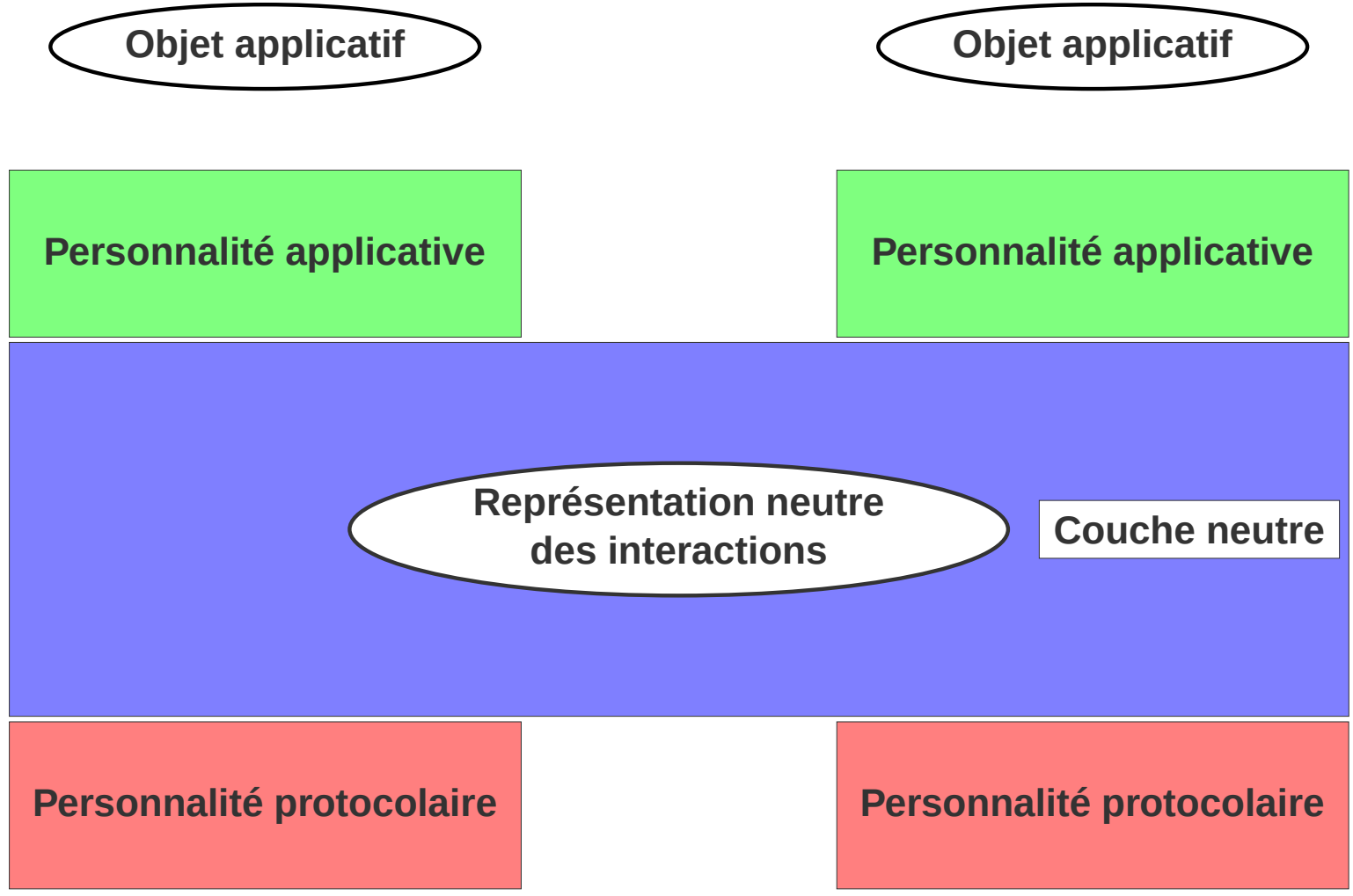
Anatomie d’un intergiciel schizophrène



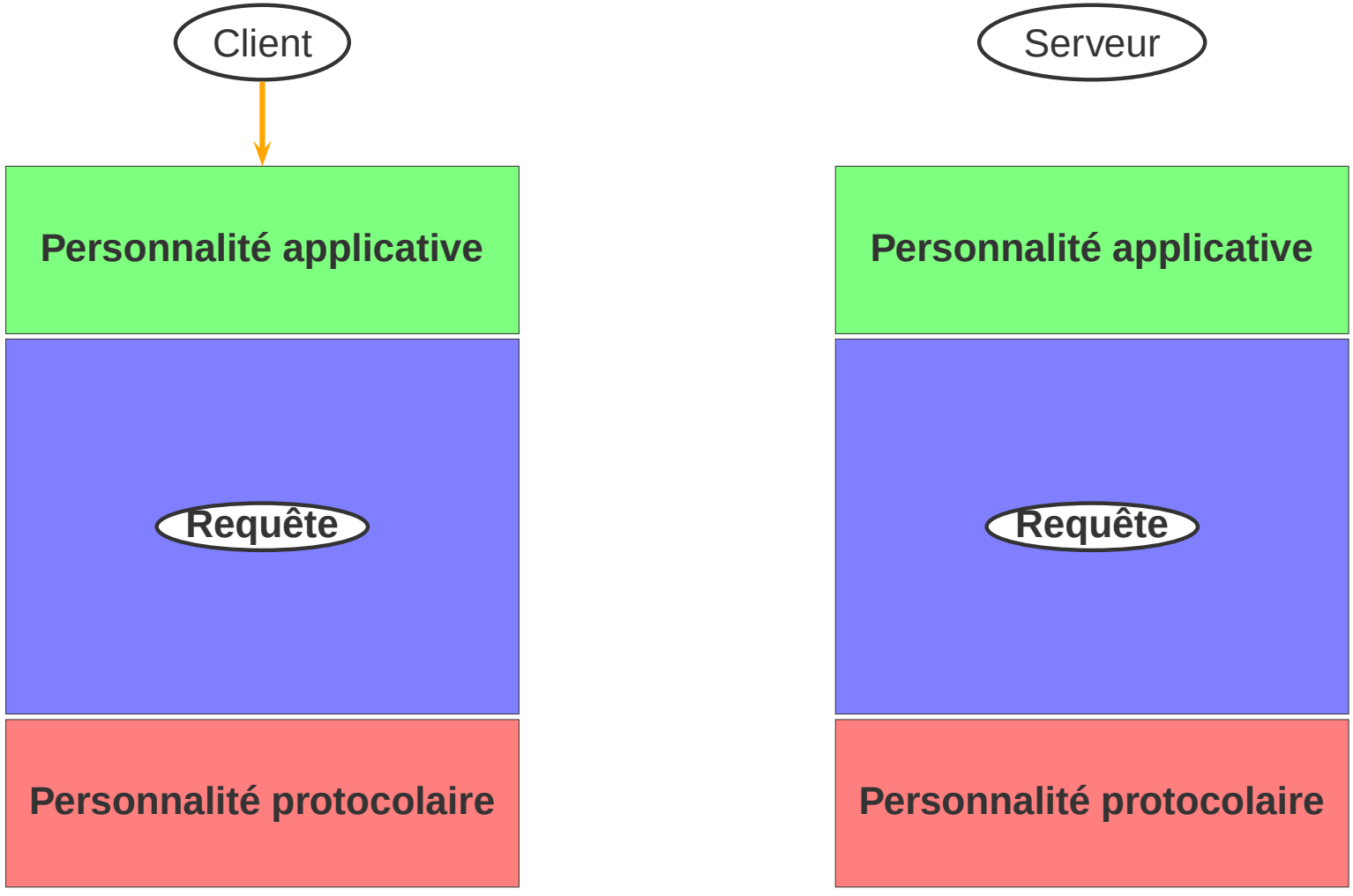
Anatomie d’un intergiciel schizophrène



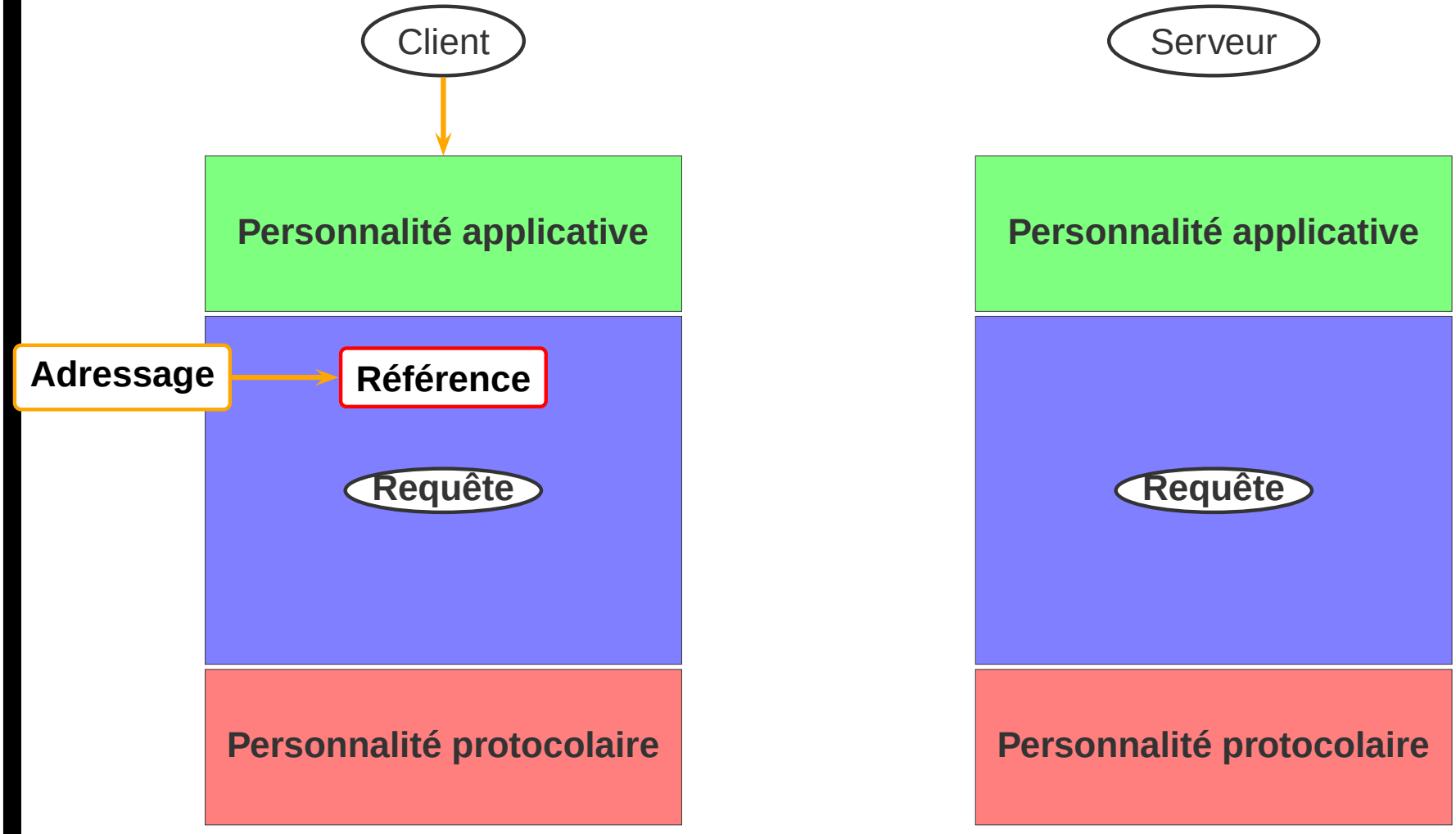
Anatomie d’un intergiciel schizophrène



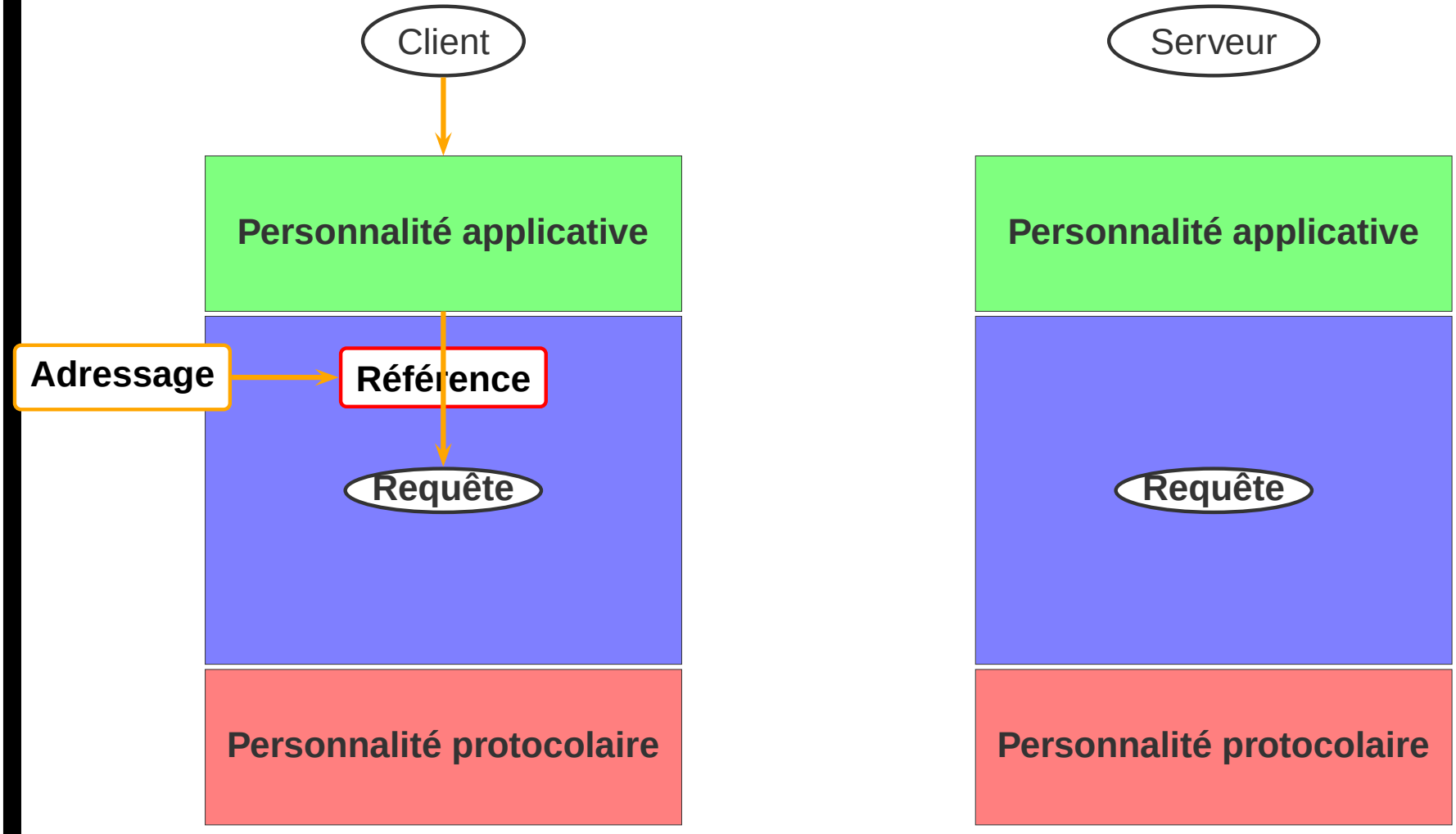
Déroulement d'une interaction



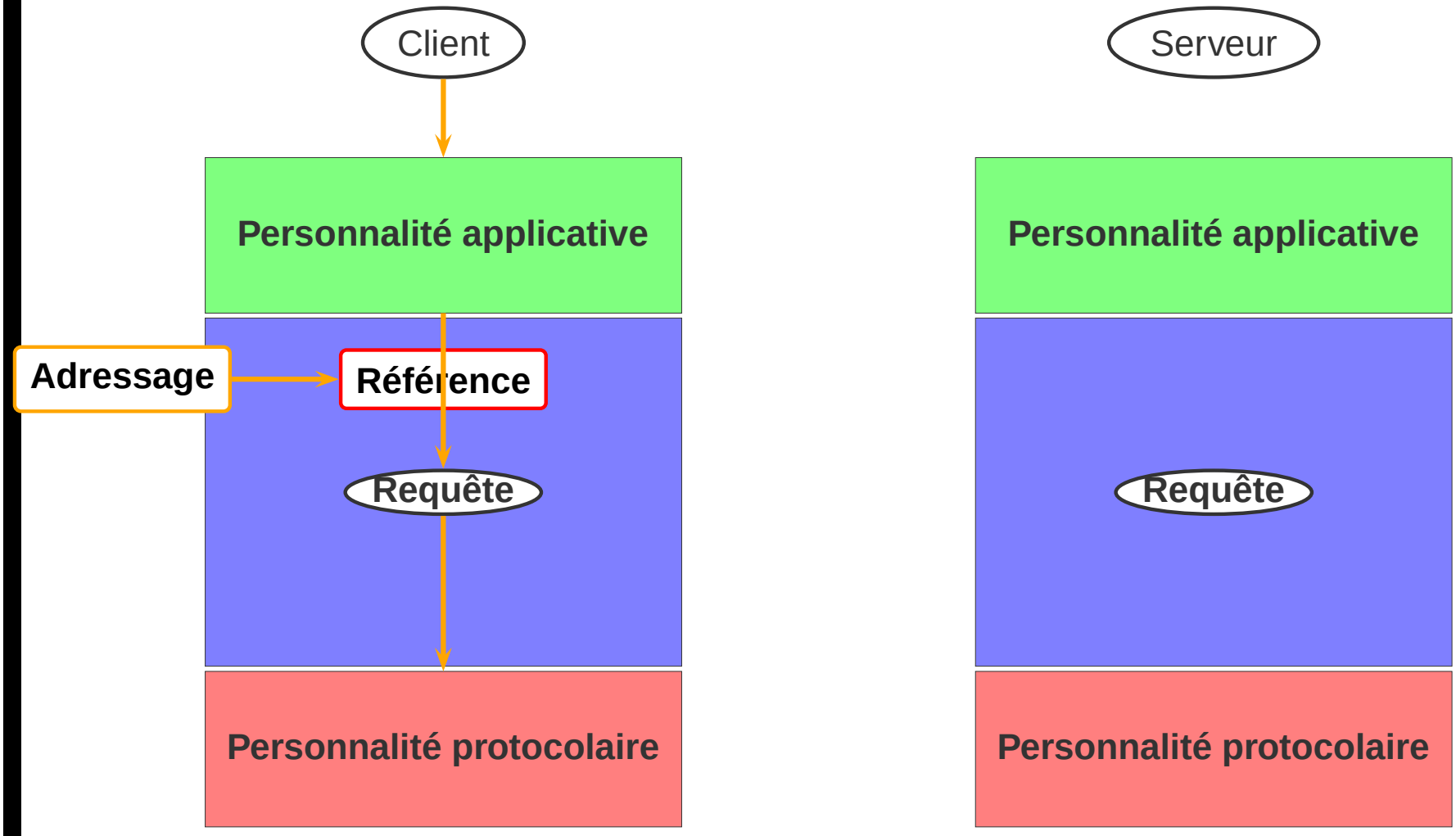
Déroulement d'une interaction



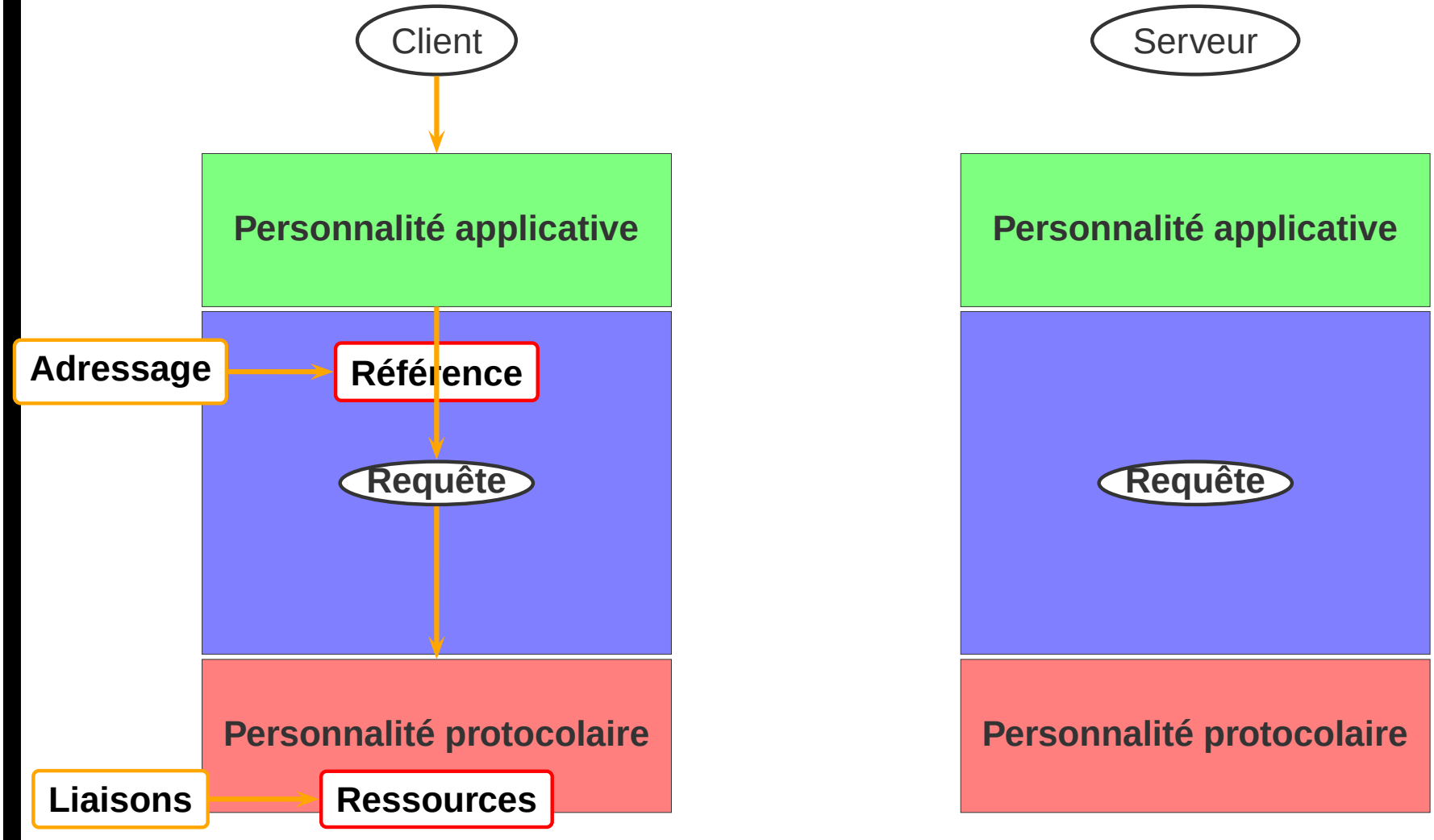
Déroulement d'une interaction



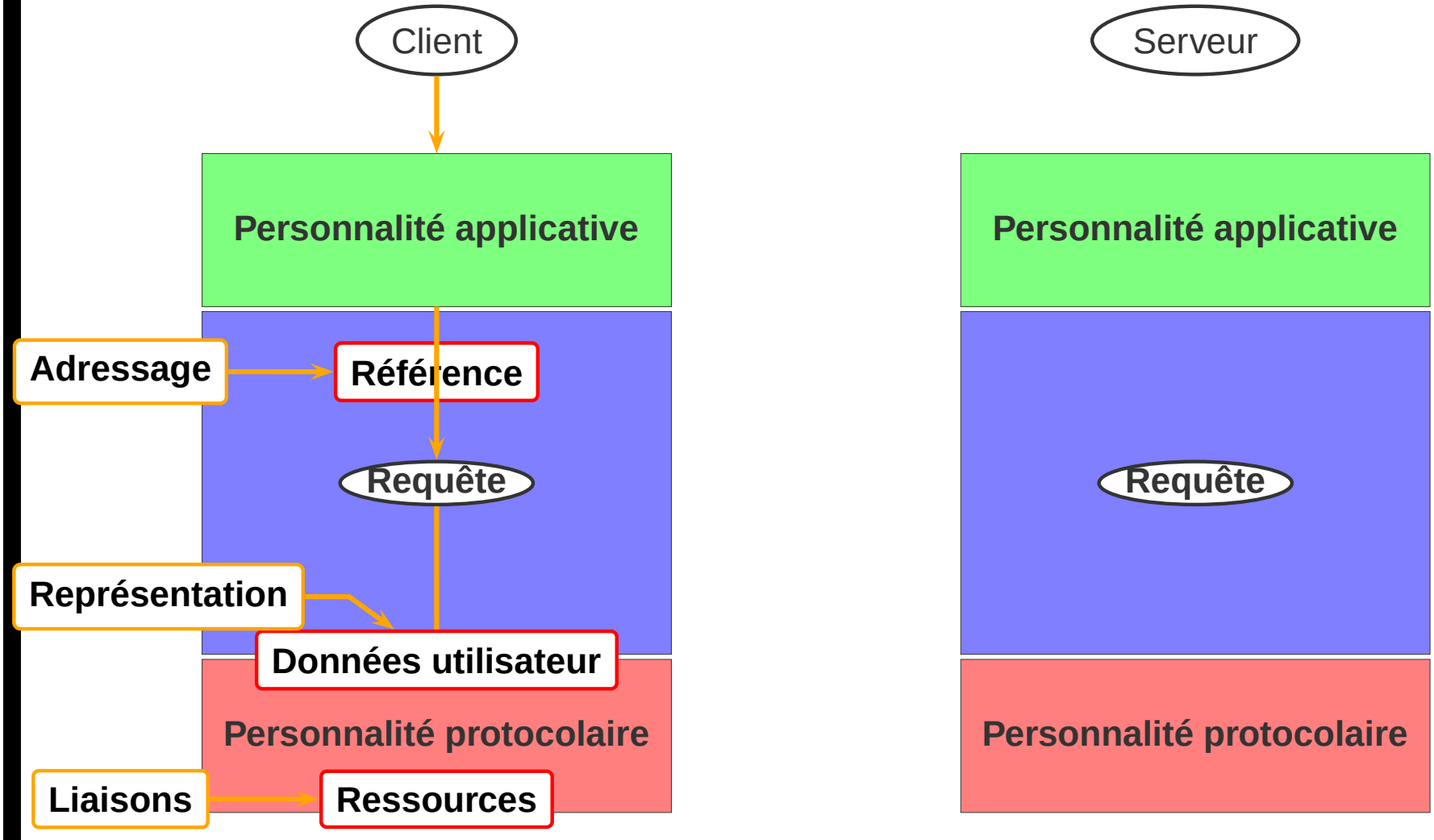
Déroulement d'une interaction



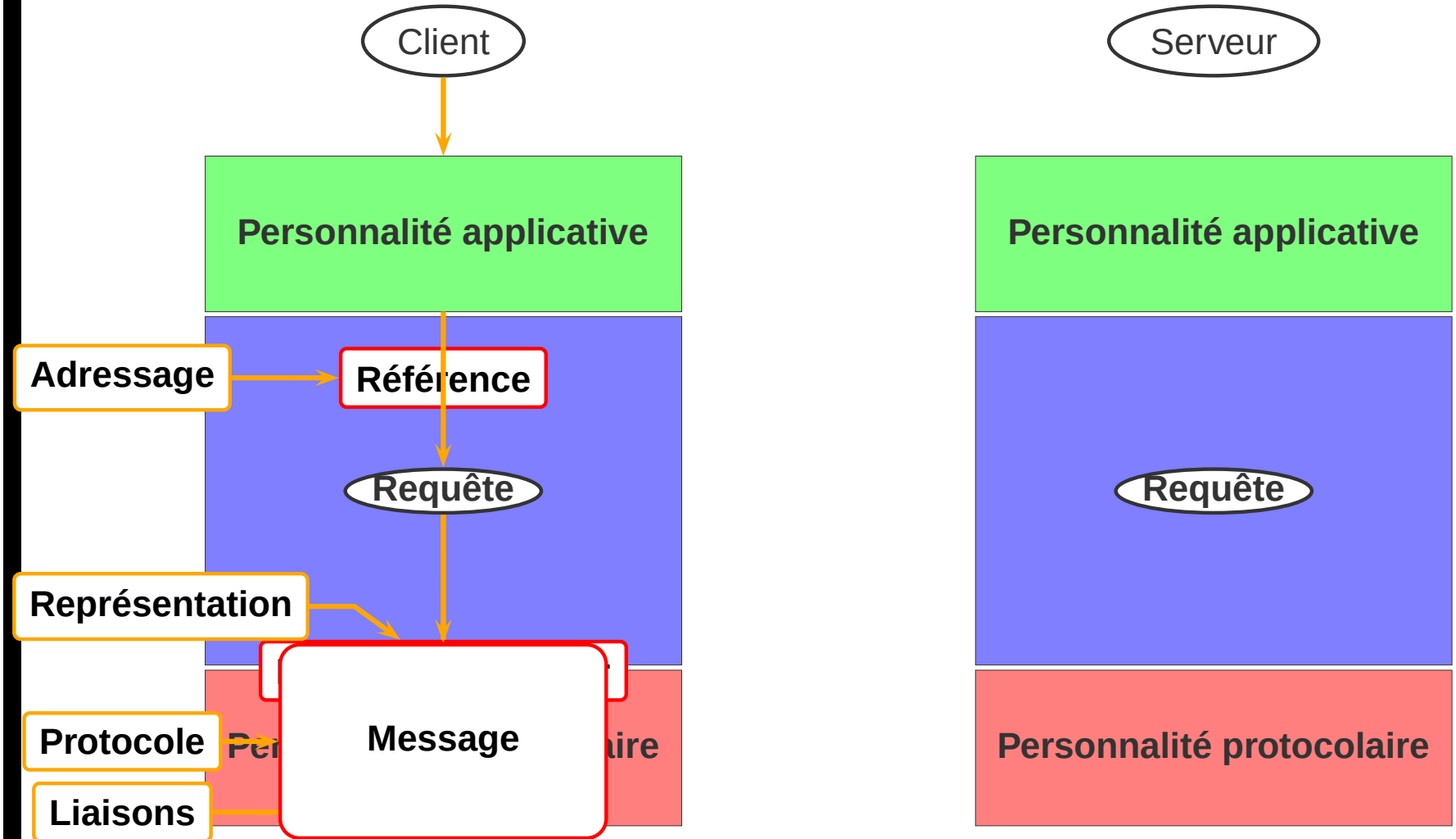
Déroulement d'une interaction



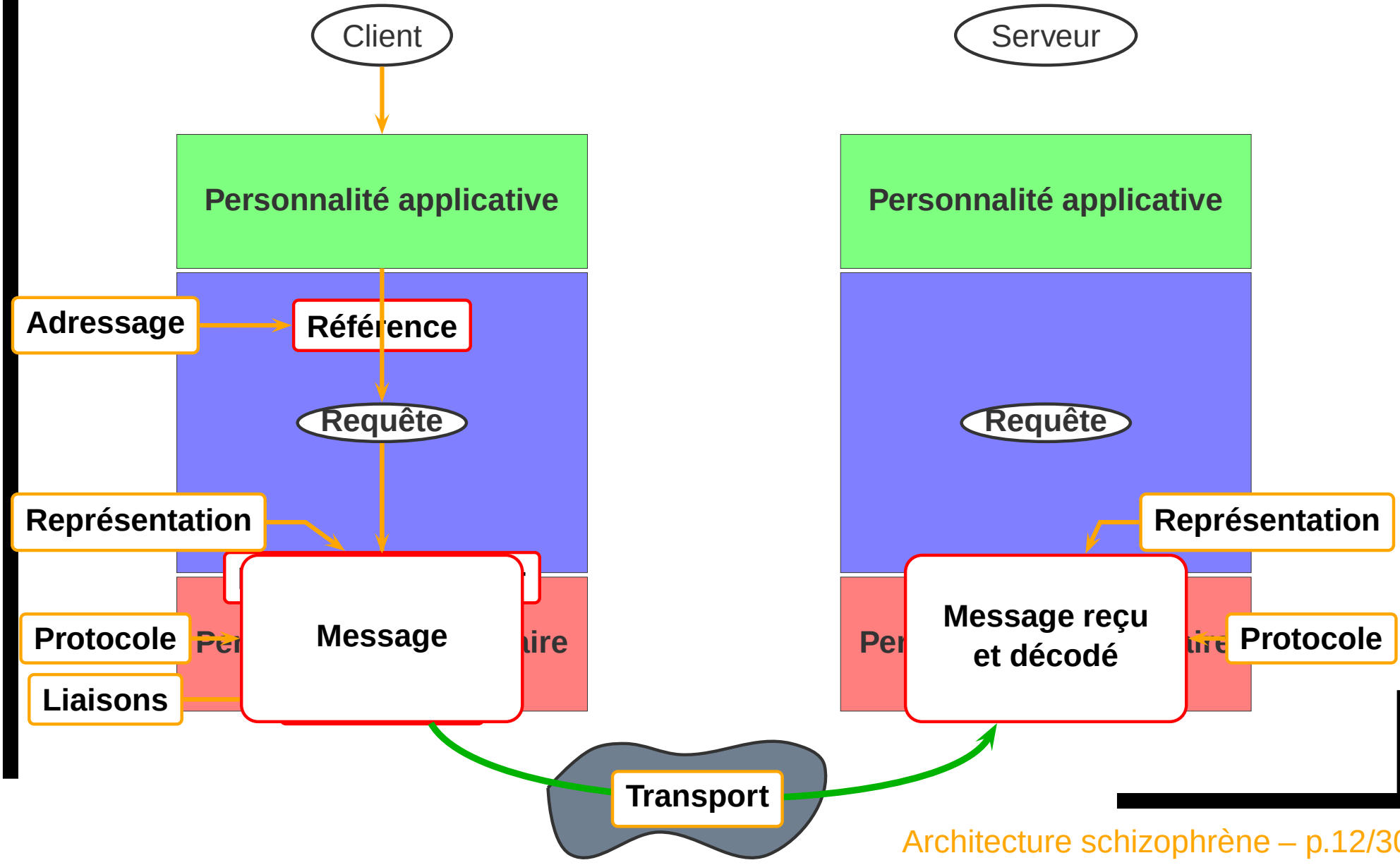
Déroulement d'une interaction



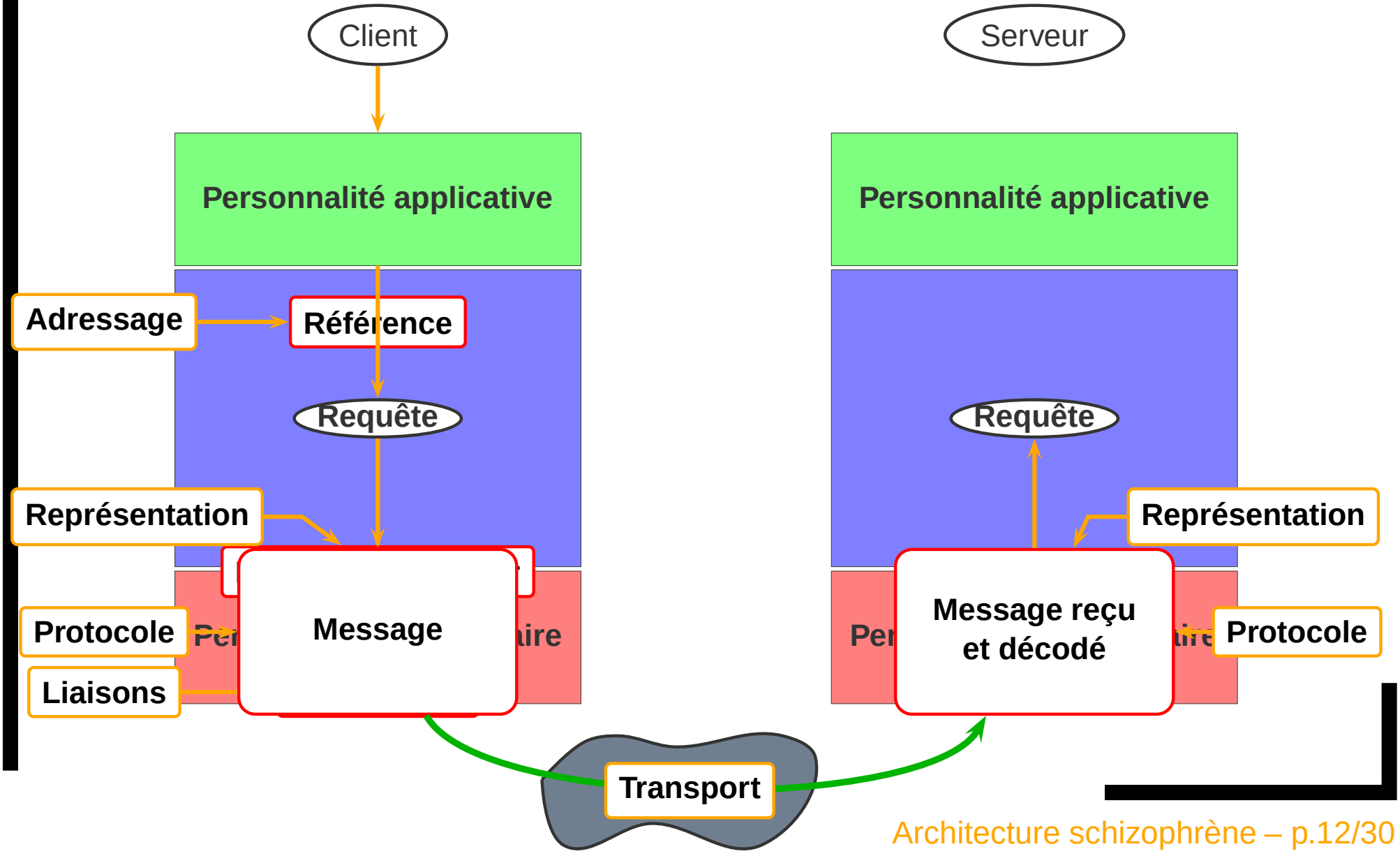
Déroulement d'une interaction



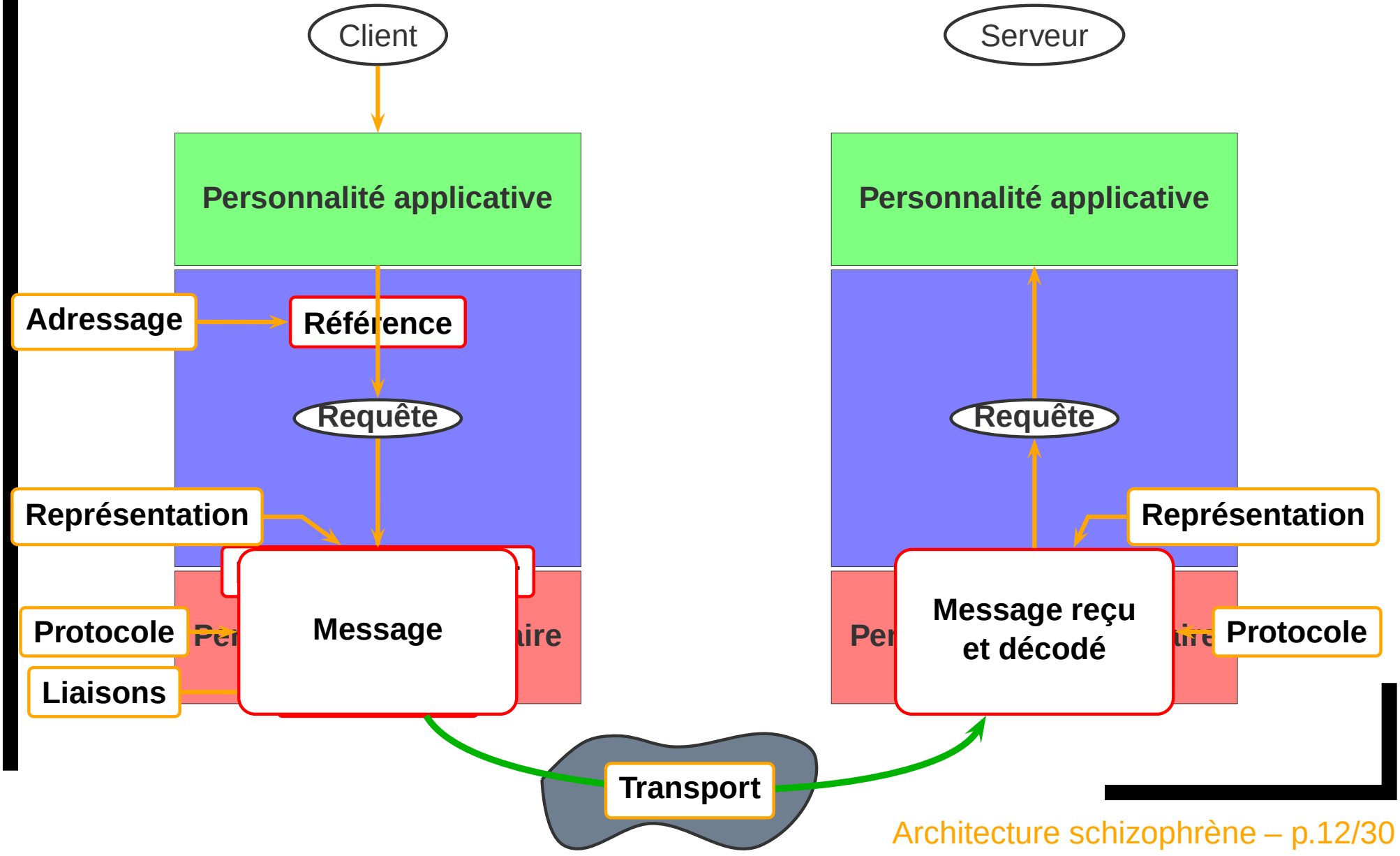
Déroulement d'une interaction



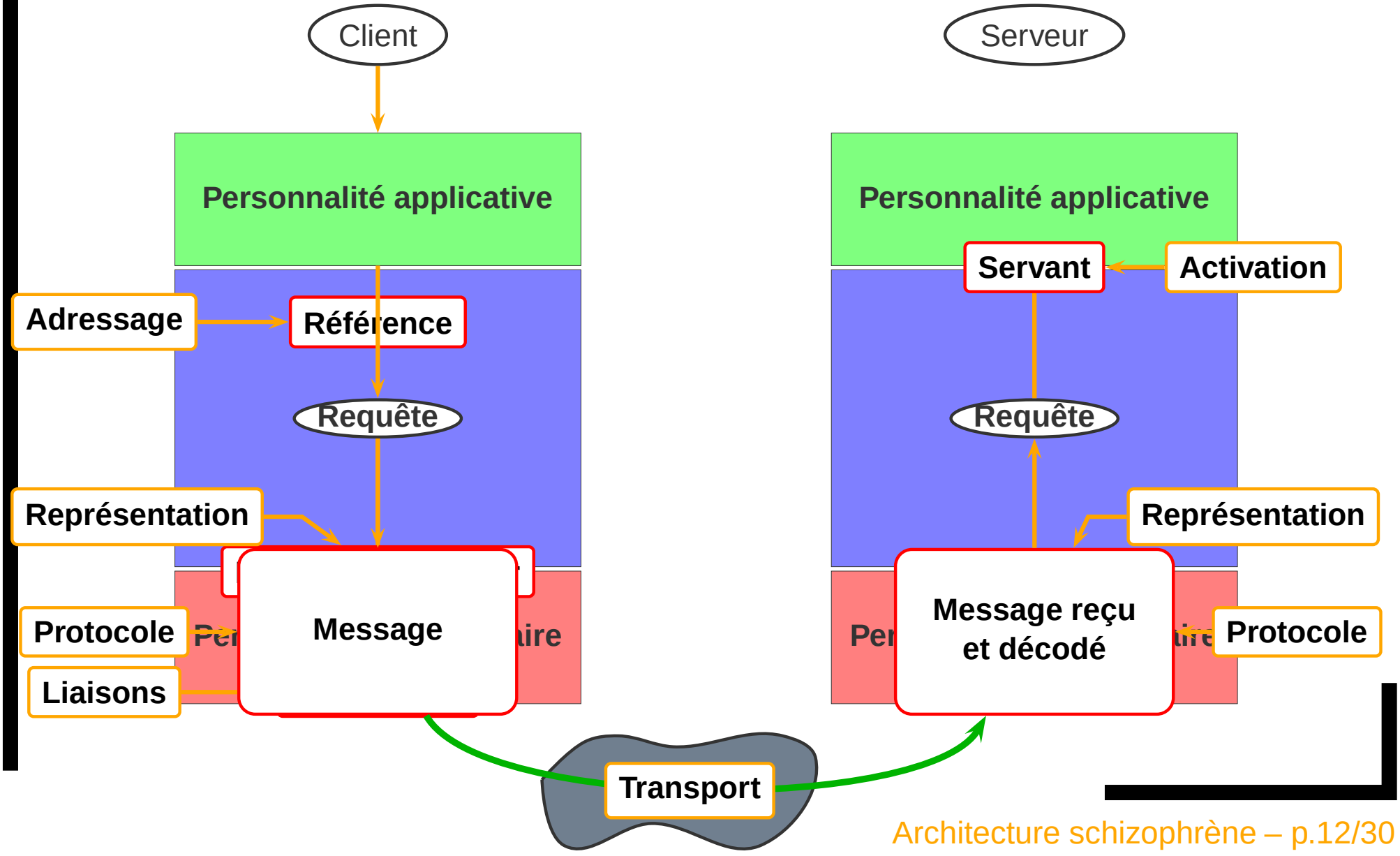
Déroulement d'une interaction



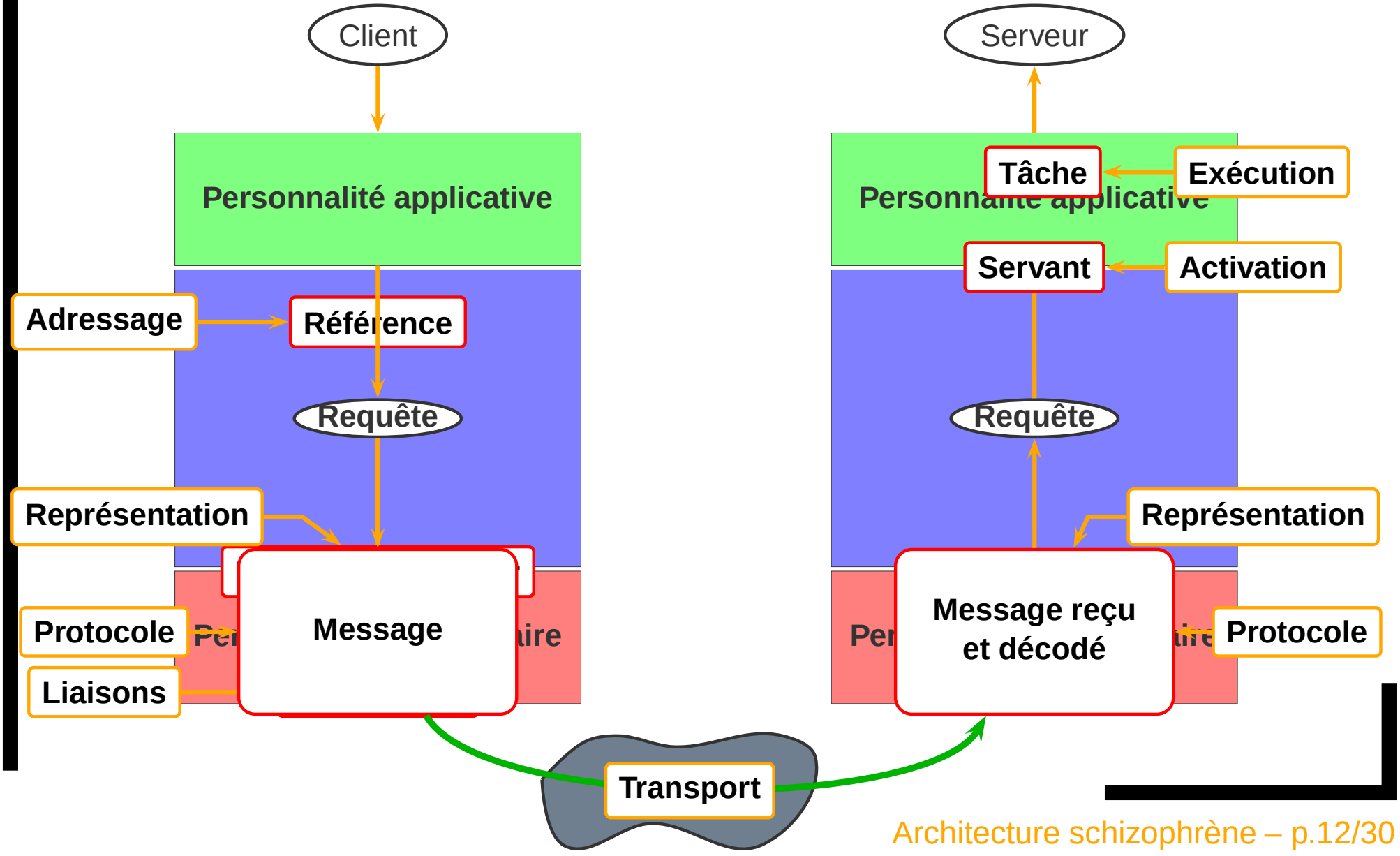
Déroulement d'une interaction



Déroulement d'une interaction



Déroulement d'une interaction



Quatre contributions à l'interopérabilité

L'architecture répond à son objectif en proposant simultanément :

- Protocole fédérateur
- Protocoles multiples
- Passerelles dynamiques par composition de personnalités protocolaires
- Développement *indépendant* des personnalités applicatives et protocolaires

PolyORB, un intergiciel schizophrène

Services de base

- Bibliothèque de parallélisme *configurable*
  profils : sans/avec parallélisme, Ravenscar

Services de base

- Bibliothèque de parallélisme *configurable* profils : sans/avec parallélisme, Ravenscar
- Réalisation de fonctions de hachage parfaites statiques et dynamiques

Services de base

- Bibliothèque de parallélisme *configurable*
  profils : sans/avec parallélisme, Ravenscar
- Réalisation de fonctions de hachage parfaites statiques et dynamiques
- Nouveaux patrons de conception :
  - *Component*
    ajout de méthodes à des types
  - *Annotation*
    ajout d'attributs à des types

utilisés par l'ensemble de l'intergiciel

Couche neutre

- Noyau générique

Couche neutre

- Noyau générique
  - Usine de références — adressage
  - Ordonnanceur-orienteur
  - Lieur inspiré des liaisons de Jonathan — liaisons

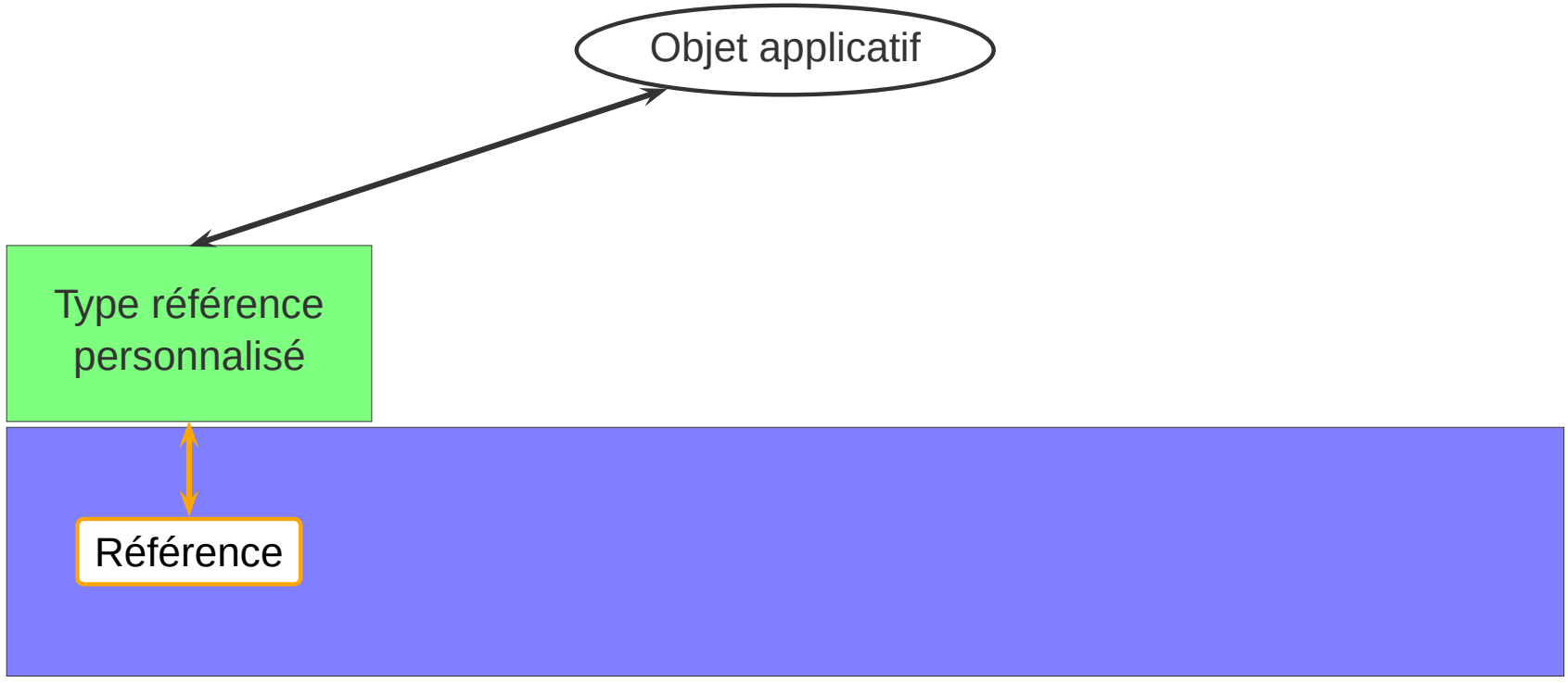
Couche neutre

- Noyau générique
  - Usine de références — adressage
  - Ordonnanceur-orienteur
  - Lieur inspiré des liaisons de Jonathan — liaisons
- Adaptateurs d'objets génériques inspirés de CORBA — activation, exécution

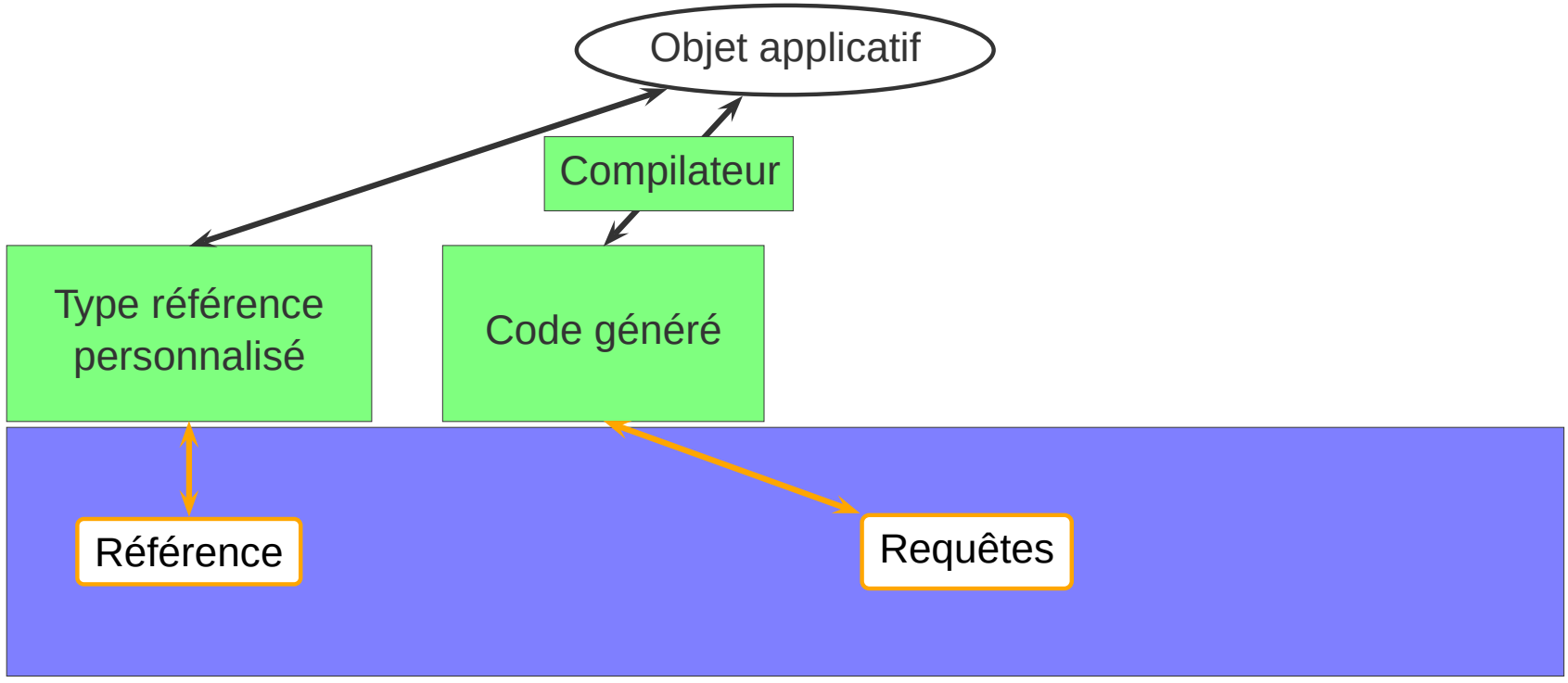
Couche neutre

- Noyau générique
  - Usine de références — adressage
  - Ordonnanceur-orienteur
  - Lieur inspiré des liaisons de Jonathan — liaisons
- Adaptateurs d'objets génériques inspirés de CORBA — activation, exécution
- Infrastructure de transport
  - filtres inspirés de *x*-kernel, netgraph
  - patrons de TAO — transport

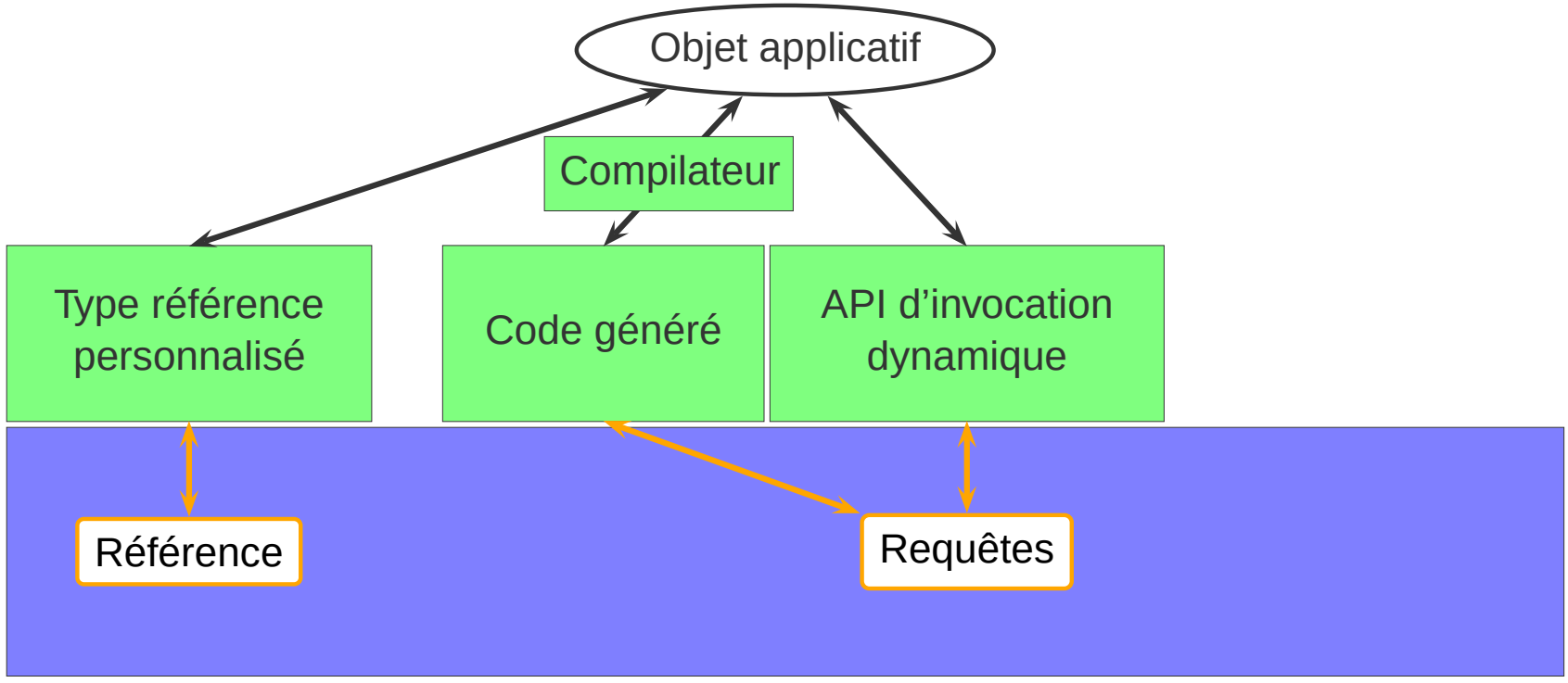
Anatomie d’une personnalité applicative



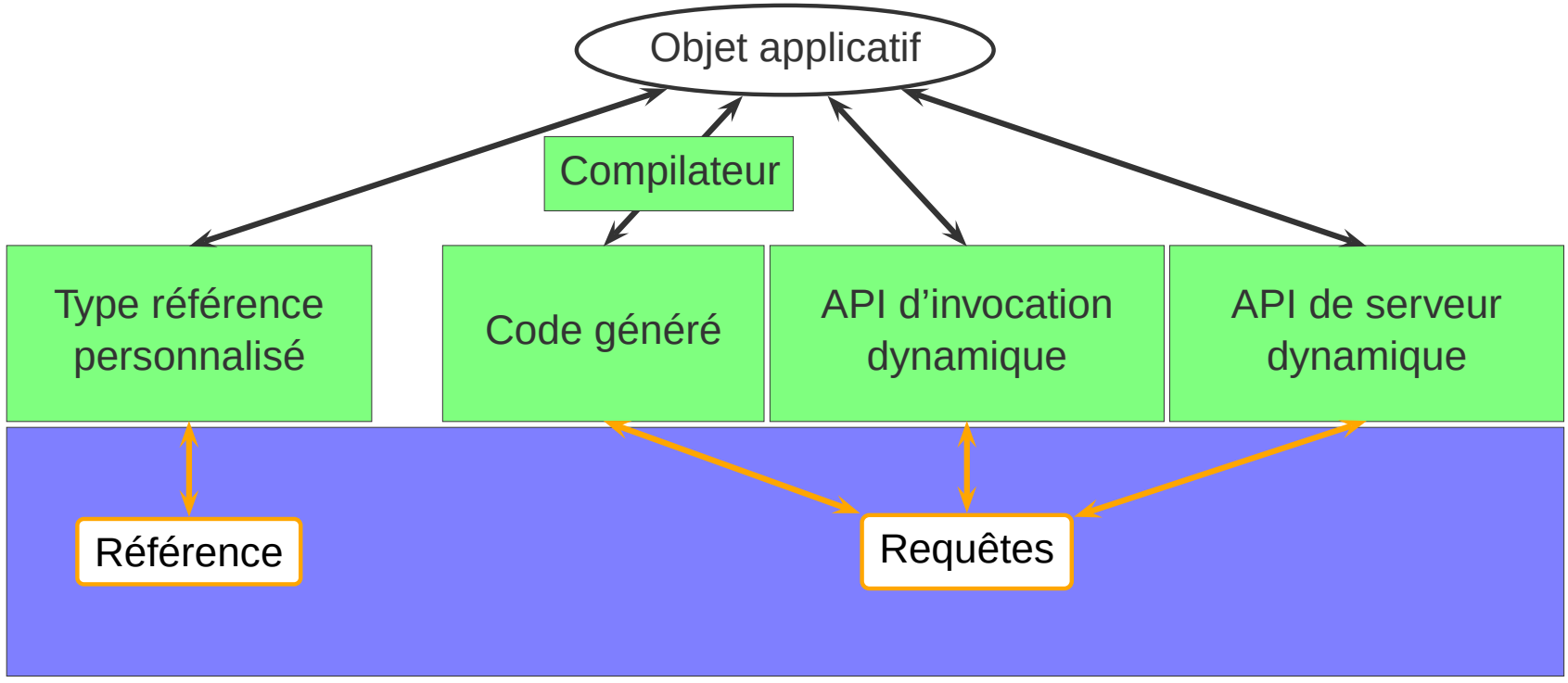
Anatomie d’une personnalité applicative



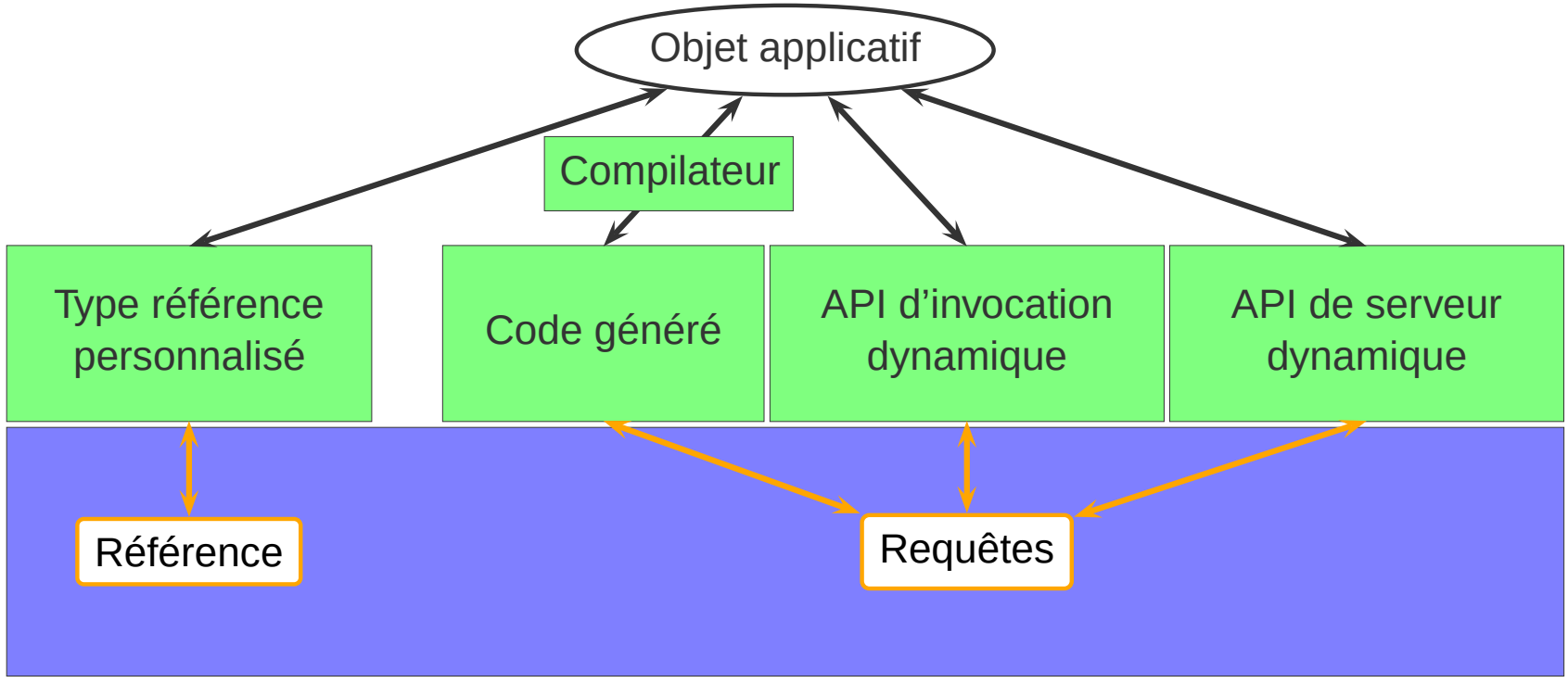
Anatomie d’une personnalité applicative



Anatomie d’une personnalité applicative



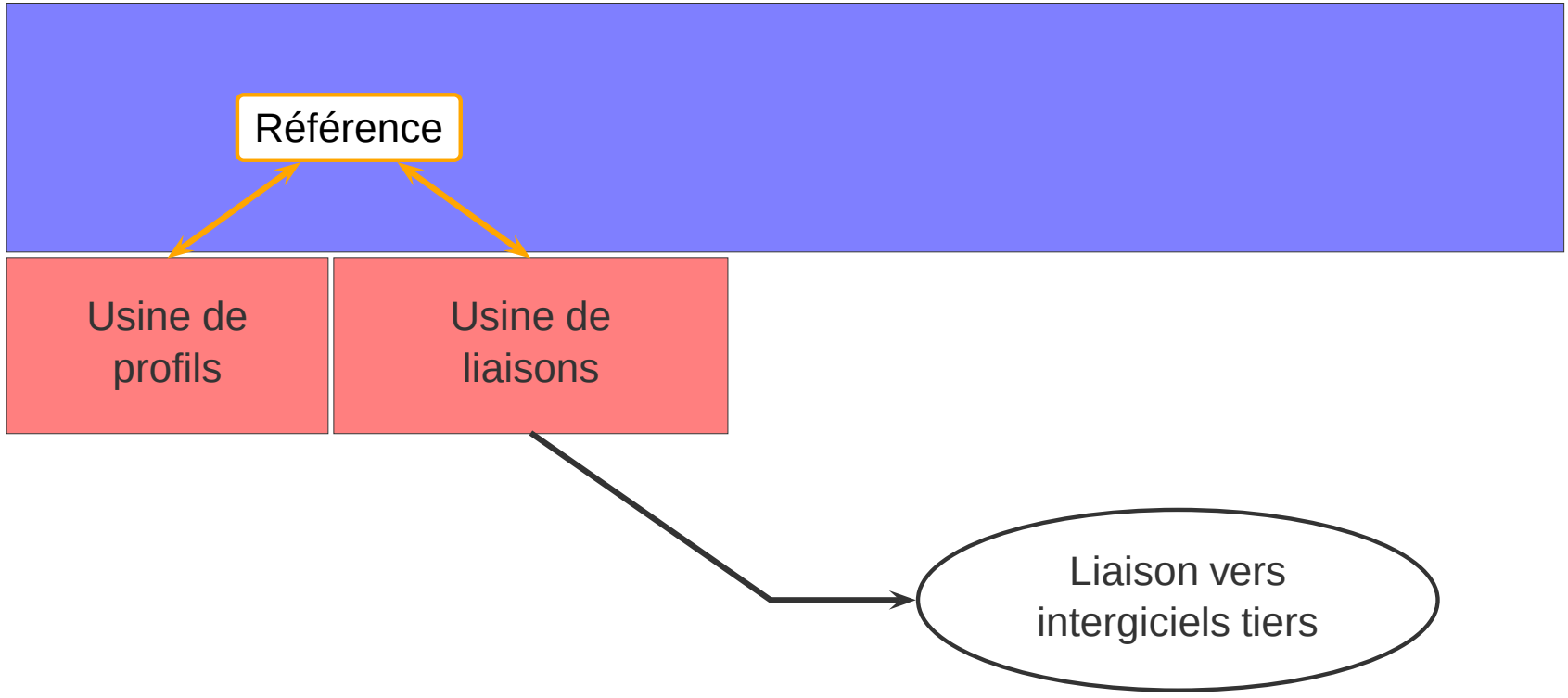
Anatomie d'une personnalité applicative



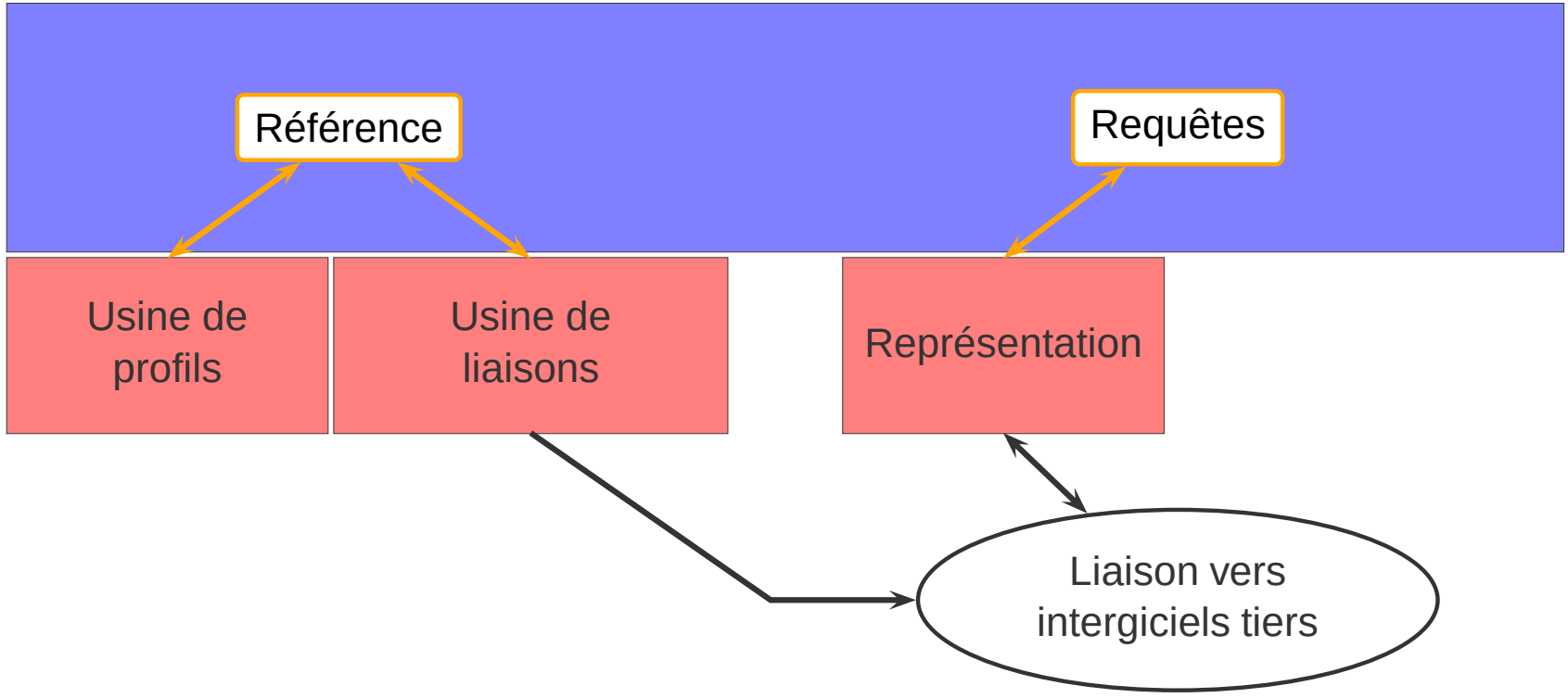
- CORBA (système de communication, compilateur IDL `idlac`)
- DSA (système de communication, compilateur Ada 95 GNAT/GCC)
  DSA = Ada 95 Distributed Systems Annex
- MOMA (système de communication)

Anatomie d’une personnalité protocolaire

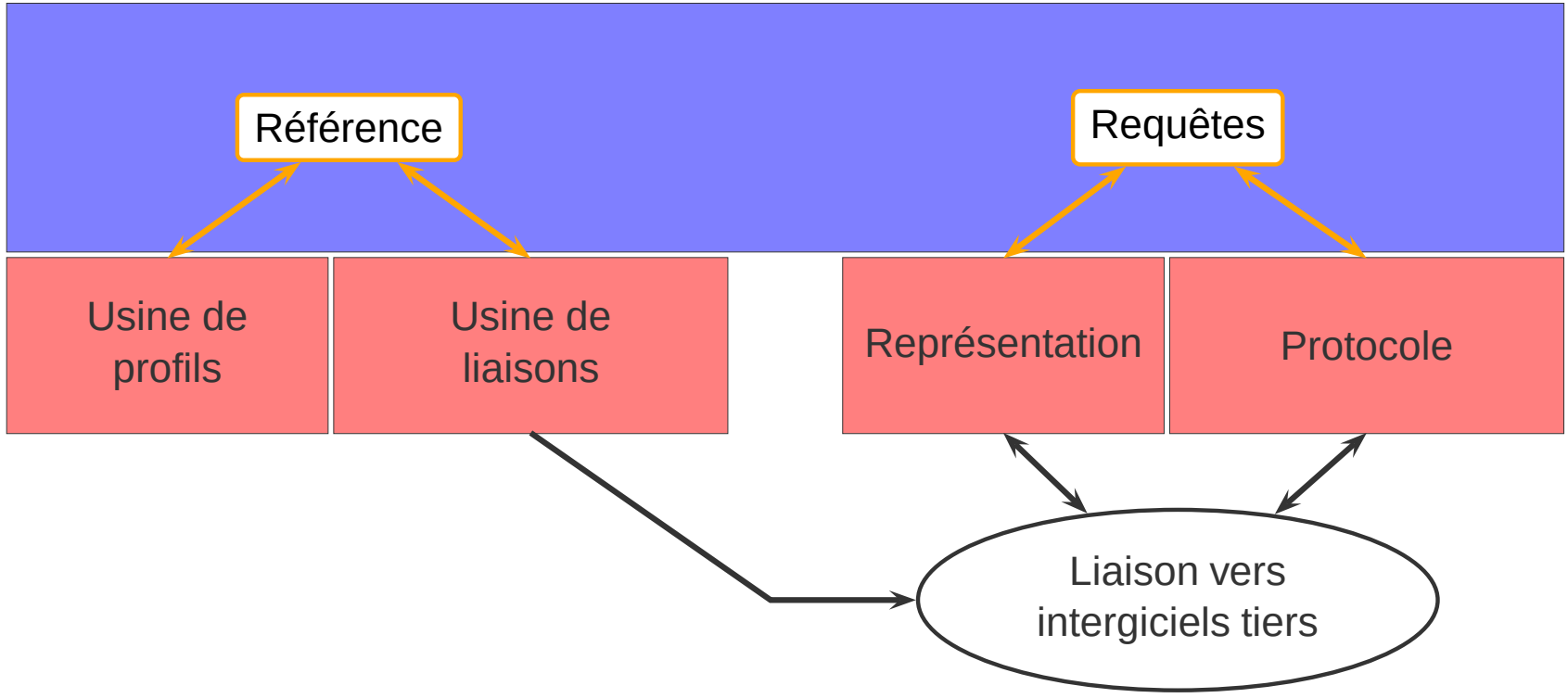
Anatomie d'une personnalité protocolaire



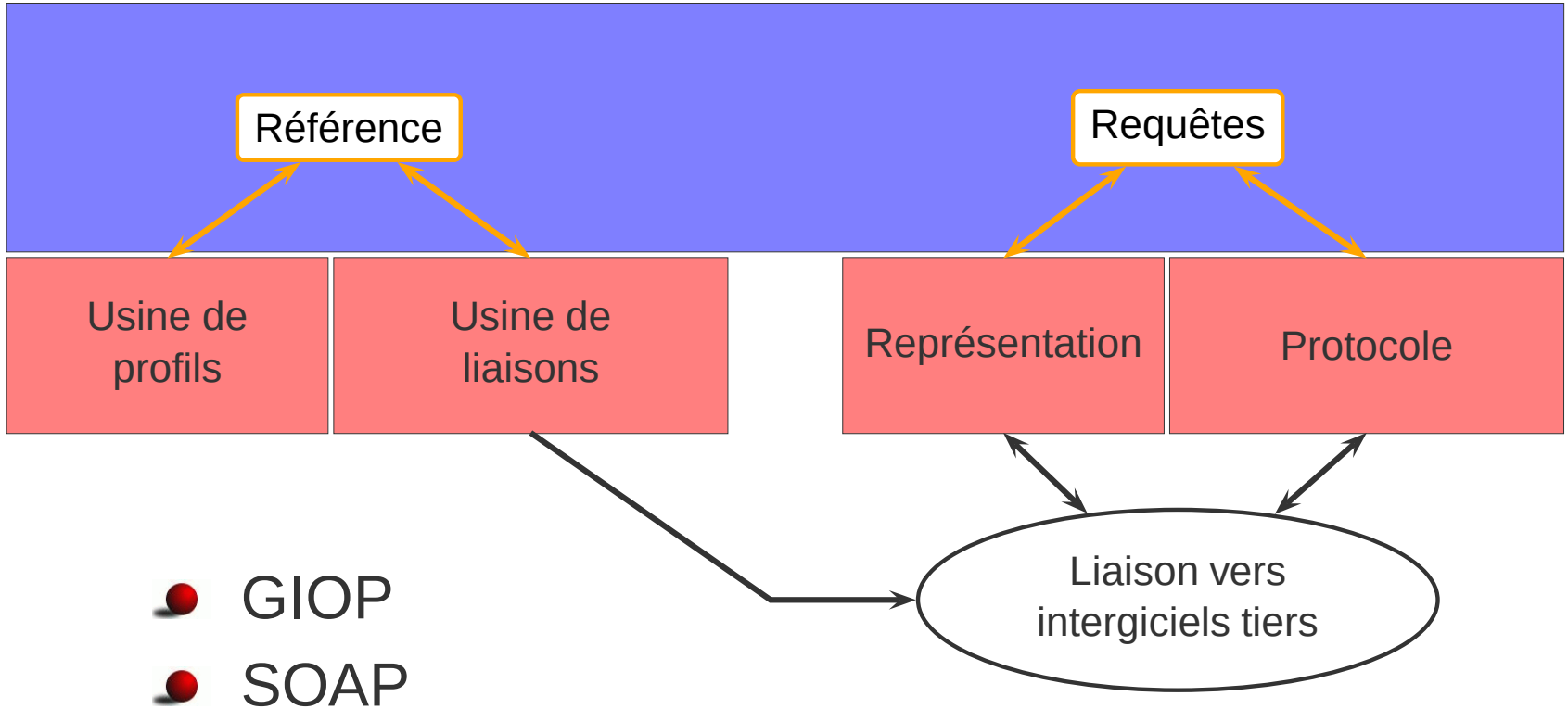
Anatomie d'une personnalité protocolaire



Anatomie d'une personnalité protocolaire



Anatomie d'une personnalité protocolaire



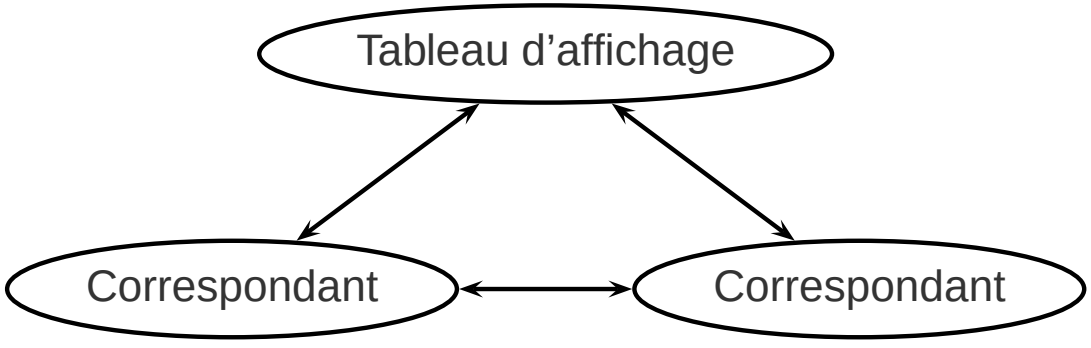
- GIOP
- SOAP

Tests fonctionnels du prototype

- Tests homogènes
  - Tests CORBA d'AdaBroker
  - Tests DSA ACATS (suite normalisée de vérification de conformité)
  - Tests GIOP : interaction avec omniORB, Jonathan

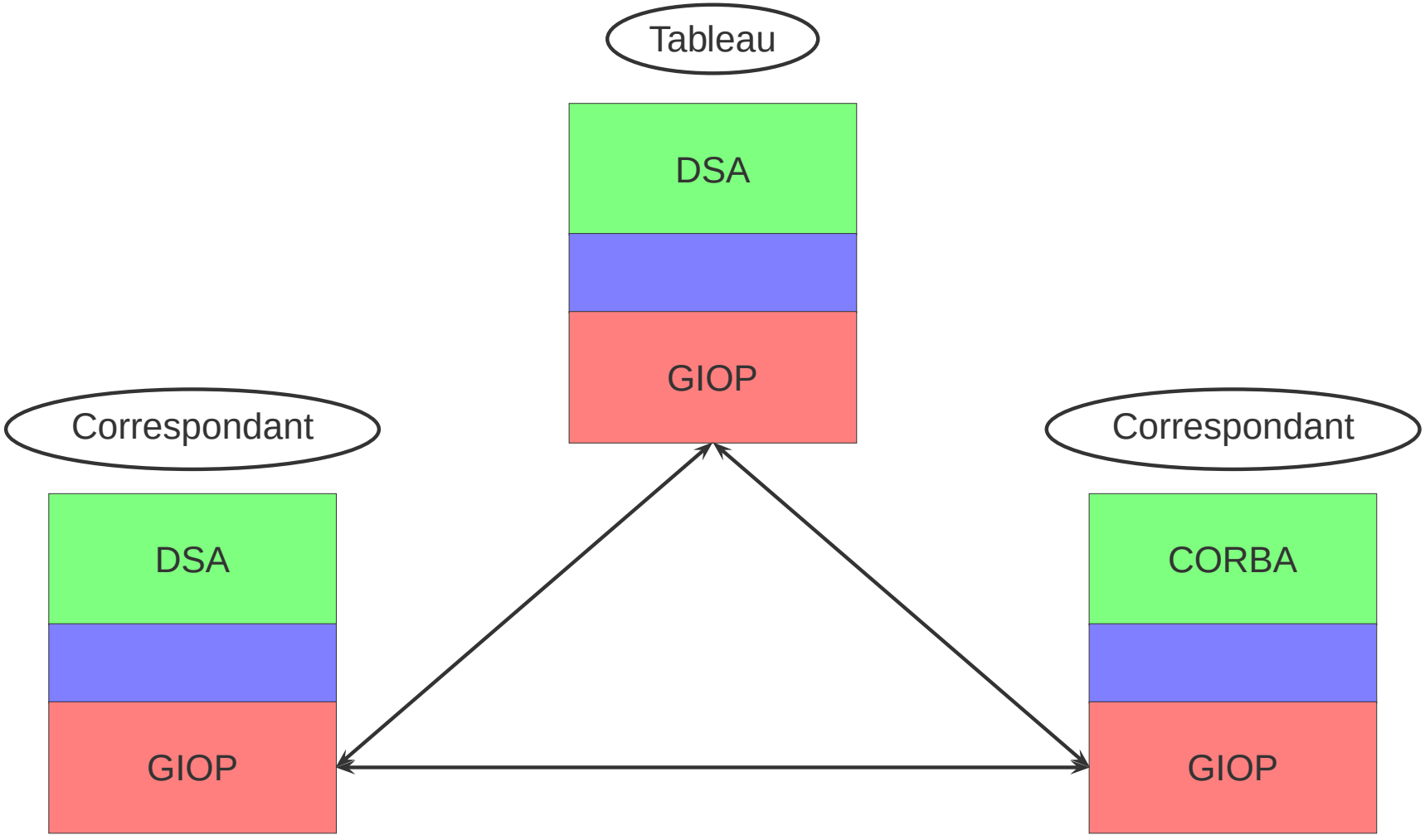
Tests fonctionnels du prototype

- Tests homogènes
  - Tests CORBA d'AdaBroker
  - Tests DSA ACATS (suite normalisée de vérification de conformité)
  - Tests GIOP : interaction avec omniORB, Jonathan
- Tests d'interopérabilité entre personnalités et entre modèles : application témoin

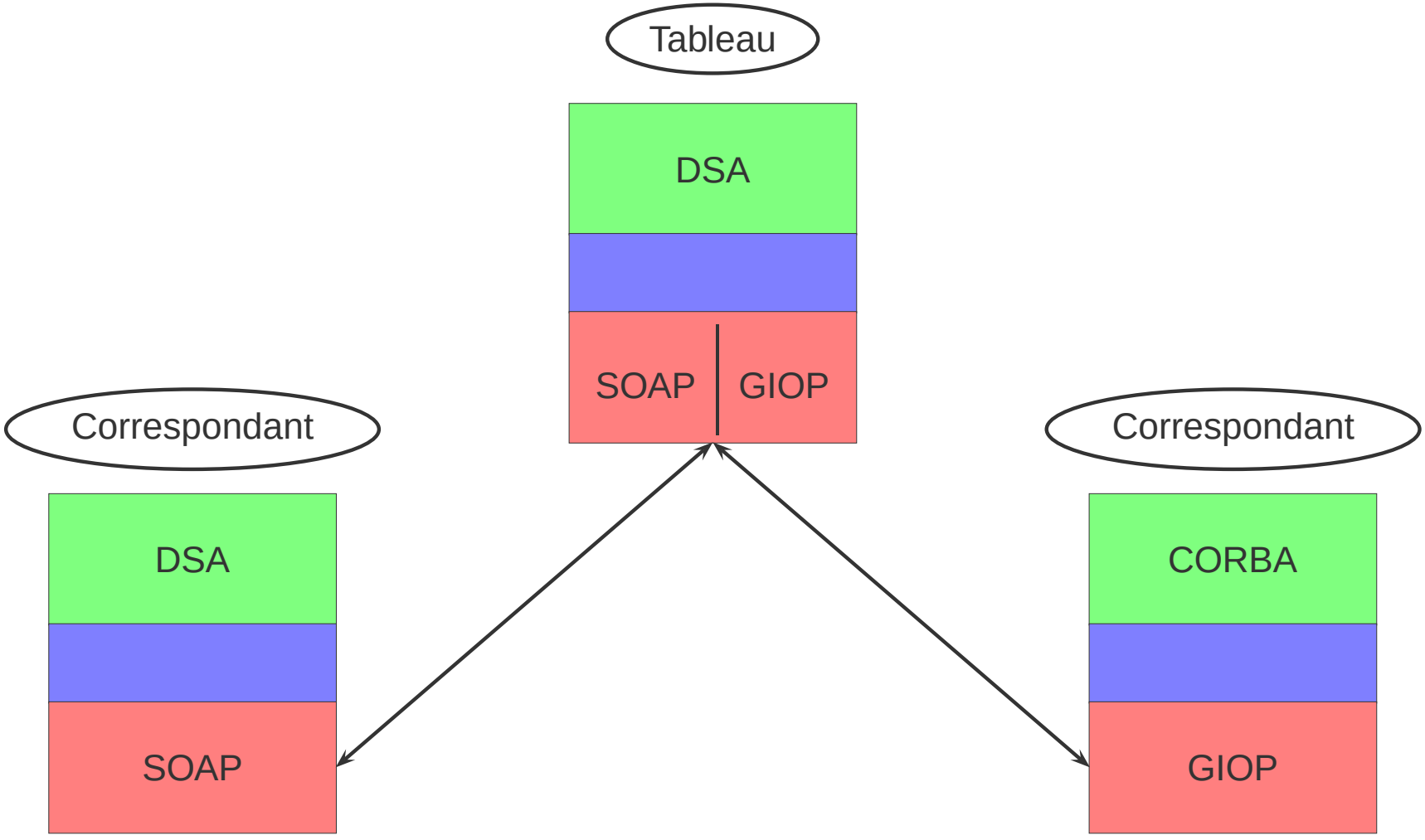


- publication/abonnement
- point à point

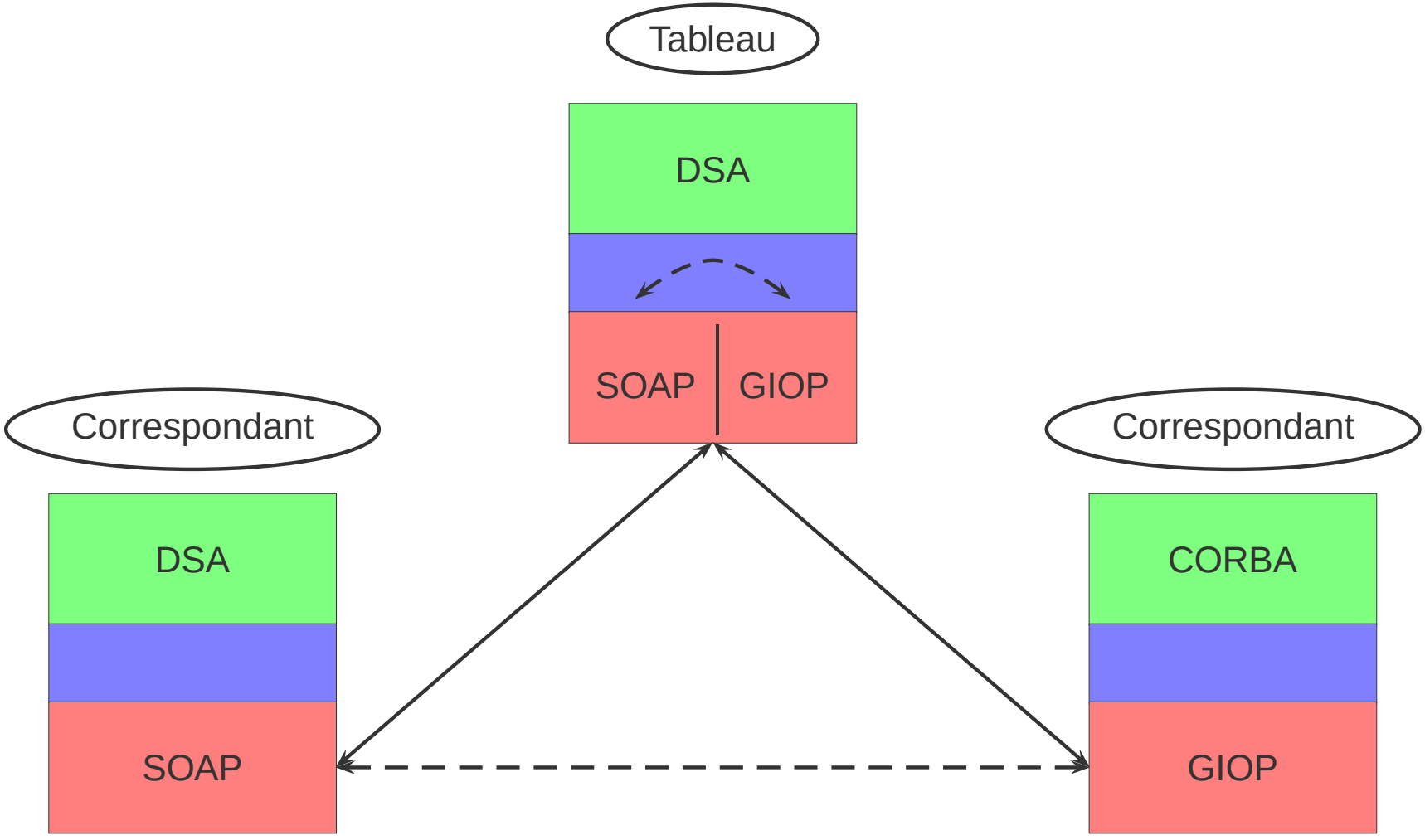
Configuration « Protocole fédérateur »



Configuration « Protocoles multiples »



Configuration « Passerelle dynamique »



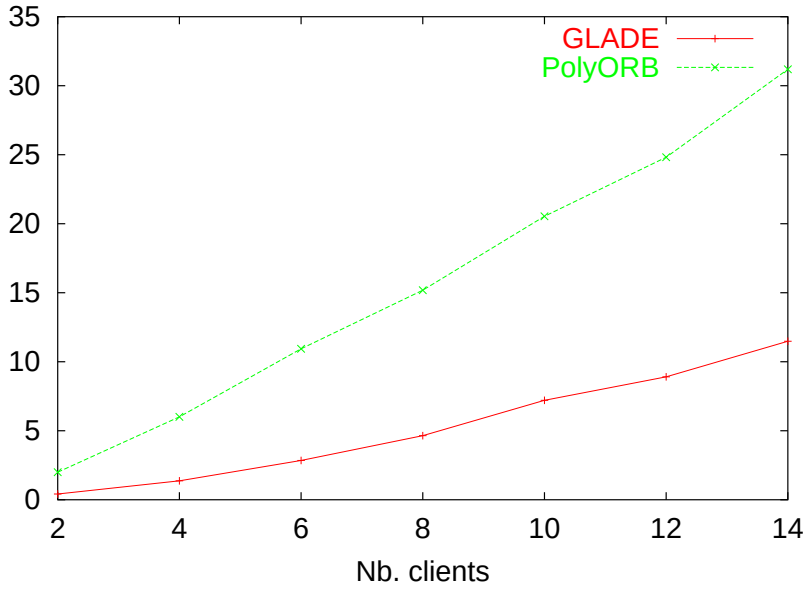
Performances : comparaison CORBA

Temps (secondes) d'exécution sur 10 000 appels de méthodes simples

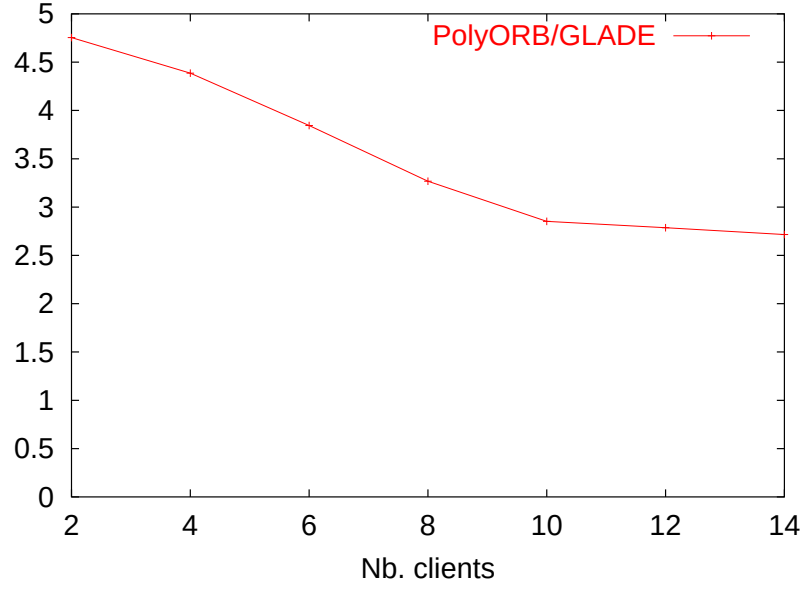
Client \ Serveur	omniORB (DSI)	Jonathan (statique)	PolyORB sans //	PolyORB avec //
omniORB (DII)	**1,12**	**3,71**	**2,33**	**4,90**
PolyORB sans //	**1,20**	**4,60**	**2,92**	**5,72**
Jonathan (statique)	**4,17**	**7,09**	**5,31**	**8,26**

Performances : comparaison DSA

Temps d'exécution sur l'échange de 100 messages entre n correspondants de l'application témoin



Temps d'exécution

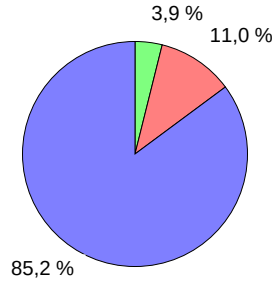
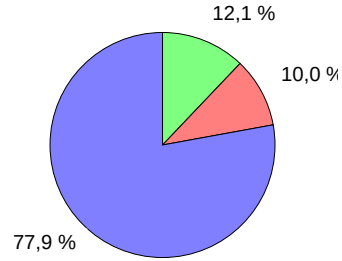
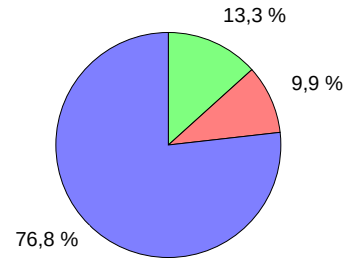
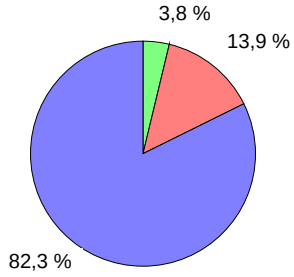
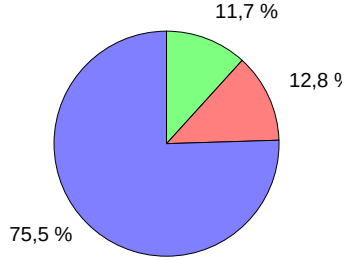
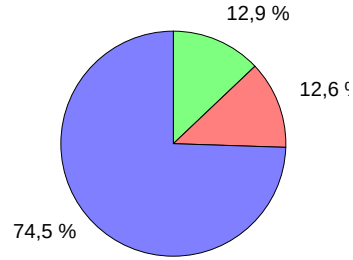


Rapport $\frac{PolyORB}{GLADE}$

Performances : conclusions

- Proches ou meilleures que certains ORB déjà industrialisés (par exemple Jonathan)
- Pas de surcoût du passage à l'échelle
- Font apparaître le coût de l'architecture schizophrène
- Prototype centré sur les fonctionnalités, pas d'optimisation au départ
- Certains modules issus de projets d'étudiants
- Nombreuses optimisations en cours : parallélisme, adaptateur d'objets, compilateur
- Coût marginal par rapport à omniORB et GLADE : généralement 2,5 à 2,7

Code fortement mutualisé

	DSA	CORBA	MOMA
GIOP			
SOAP			

Couche neutre Personnalité protocolaire Personnalité applicative

Conclusion et perspectives

Contributions

Une solution au paradoxe de l'intergiciel (M2M) : *l'architecture schizophrène*.

- découplage entre *personnalités applicatives* et *personnalités protocolaires*
- au moyen d'une *couche neutre*
- qui mutualise les *fonctions récurrentes* d'un intergiciel
- et permet la *cohabitation efficace* de différentes personnalités
- architecture validée par un prototype opérationnel

Réalisation : PolyORB

- Une couche neutre
- Trois personnalités applicatives
  - CORBA, DSA, MOMA
  - compilateurs IDLAC (CORBA), GNAT (DSA)
- Deux personnalités protocolaires
  - GIOP
  - SOAP
- Personnalités combinables à volonté
- Intergiciel configurable
- Distribué sous forme de logiciel libre (GMGPL)
  `http://libre.act-europe.fr/polyorb/`

Perspectives

- Recherche :
  - nouvelles personnalités, guide méthodologique de personnalisation
  - outil générique d'aide au déploiement
  - compilateur générique
  - intergiciel temps réel
  - services génériques (cycle de vie, stockage partagé)
- Industrielles :
  - optimisation
  - achèvement des personnalités existantes
  - intégration à l'offre d'ACT Europe