

# Plan

- HLA, un survol
  - Raison d'être, histoire et domaine d'application
  - Terminologie et composants
  - Descriptions des services de l'interface normative
- yaRTI
  - Pourquoi une RTI HLA en Ada?
  - Éléments de conception
  - Quelques points d'implémentation
- Conclusions

## Objectifs de HLA

- HLA = **High Level Architecture**
- Faire des simulations **réutilisables**
  - Expression formelle de leurs capacités
  - Paradigme commun de développement
- Faire des simulations **inter opérables**
  - Spécification d'interface (I/F Spec.)
  - Infrastructure d'exécution (RTI)

# HLA, l 'histoire

## Hier

- Définition initiale par la DARPA (programme ADS)
- Reprise par le DMSO et l'AMG in 1995
- Première base du standard approuvée 1996, Sep 10
- Version 1.3 stable et utilisable en 1998
- Premières RTIs (DMSO) en 1998

## Maintenant

- Plusieurs RTIs opérationnelles, DMSO ou commerciales
- Standard IEEE (P1516) approuvé

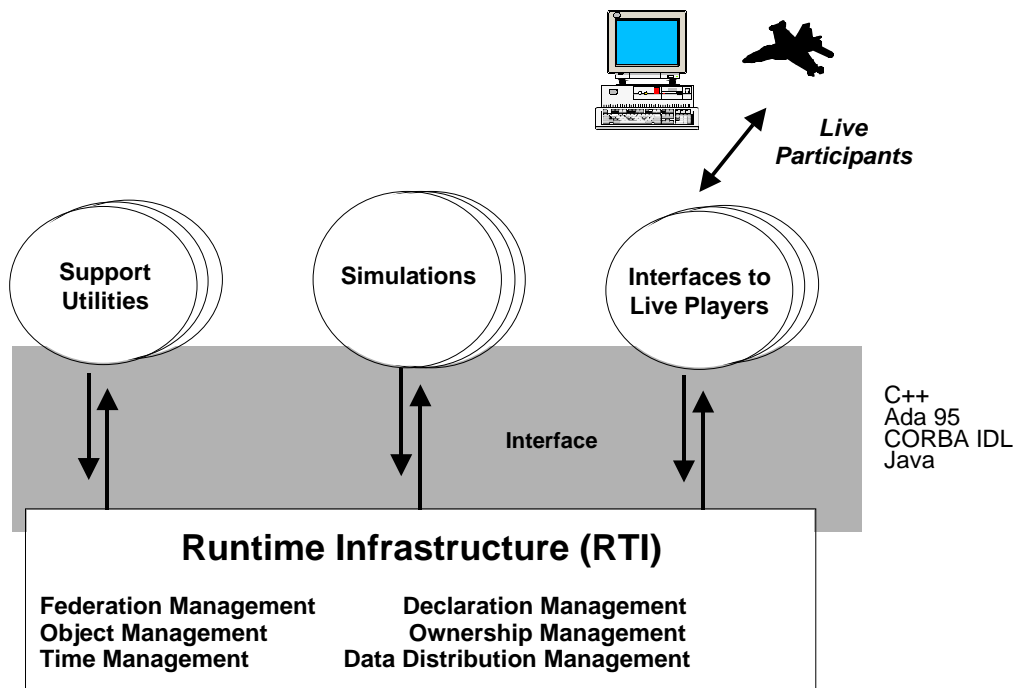
## Demain

- D'autres RTIs, d'autres domaines d'application
- Des évolutions du standard

## Domaines d 'application de HLA

- Types d'utilisation
  - Entraînement, planification, étude technique, ingénierie
  - Études technico-opérationnelles
  - Acquisition
- Applications
  - simulations purement logicielles,
  - simulations à opérateur humain,
  - composants réels (C3, systèmes réels instrumentés)

# Illustration



10 Nov. 2000

Ada-France

5

## Un peu de terminologie

- **Fédération** : un ensemble de simulations dotées d'un modèle commun utilisées ensemble pour constituer une simulation plus vaste
- **Fédéré** : simulation membre d'une fédération
  - Peut représenter une entité unique (simulateur)
  - Peut représenter de nombreuses entités, (simulation de trafic aérien à l'échelle d'un pays)
  - Peut représenter des fragments fonctionnels d'entités (modèles répartis)
- **Exécution de fédération** : session, exercice, run, ...

10 Nov. 2000

Ada-France

6

## Encore un peu

- **Objet** : Instance d'une classe d'objets, entité du domaine simulé par la fédération qui
  - Présente de l'intérêt pour plus qu'un seul fédéré
  - Est connue et gérée par l'infrastructure d'exécution
- **Interaction** : Événement transitoire produit par l'un des fédérés et reçu par d'autres (via la RTI), instance d'une classe d'interactions
- **Attribut** : Donnée nommée définie dans un modèle objet associée à chaque instance d'une classe d'objets
- **Paramètre** : Donnée nommée définie dans un modèle objet associée à chaque instance d'une classe d'interactions

## Les composants du standard

- Dix **règles** pour définir les relations entre les composants d'une fédération
- Un **méta modèle objet (OMT)** qui dit comment décrire
  - Les capacités d'un fédéré (SOM)
  - Les échanges possibles dans une fédération (FOM)
- Une **spécification d'interface (I/F Spec.)** qui décrit comment les fédérés interagissent pendant l'exécution de la fédération

# La Spécification d 'Interface

- Définit le fonctionnement de l 'interface entre les fédérés et la RTI
  - Comporte 6 groupes de services
- Pour chaque définition de service :
  - Nom, sens du flot de contrôle et description
  - Arguments en entrée
  - Arguments en sortie
  - Pré conditions
  - Post conditions
  - Exceptions
  - Services connexes
- APIs normatives en IDL, C++, Ada 95 et Java

## Contenu de l'I/F Spec.

- Les six groupes de services
  - Federation Management (20 services)
  - Declaration Management (12 services)
  - Object Management (17 services)
  - Ownership Management (16 services)
  - Time Management (23 services)
  - Data Distribution Management (13 services)
- Les annexes :
  - Support Services (29 services)
  - Management Object Model (MOM)
  - APIs

# Les catégories de services

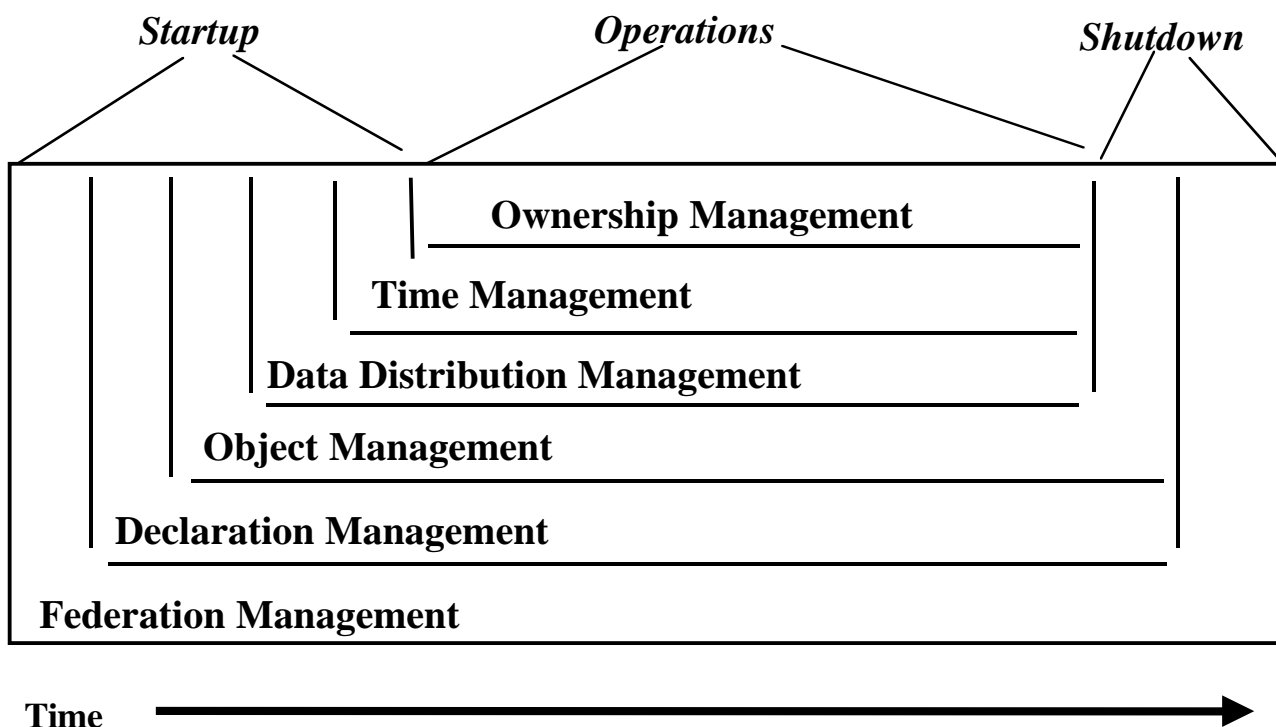
<b>Catégorie</b>	<b>Fonctionnalités</b>
<b>Federation Management</b>	Créer et détruire une exécution de fédération Rejoindre et quitter une fédération Point de rendez-vous, sauvegarde et restauration
<b>Declaration Management</b>	Établir son intention de publier ou souscrire des attributs d 'objets ou des interactions
<b>Object Management</b>	Créer et détruire des objets Mettre à jour des attributs, envoyer des interactions Recevoir des mises à jour et des interactions
<b>Ownership Management</b>	Transférer la responsabilité d'attributs
<b>Time Management</b>	Faire avancer le temps logique (temps simulé) de manière coordonnée entre fédérés
<b>Data Distribution Mgmt</b>	Optimiser les échanges

10 Nov. 2000

Ada-France

11

# Les Services et l'exécution



10 Nov. 2000

Ada-France

12

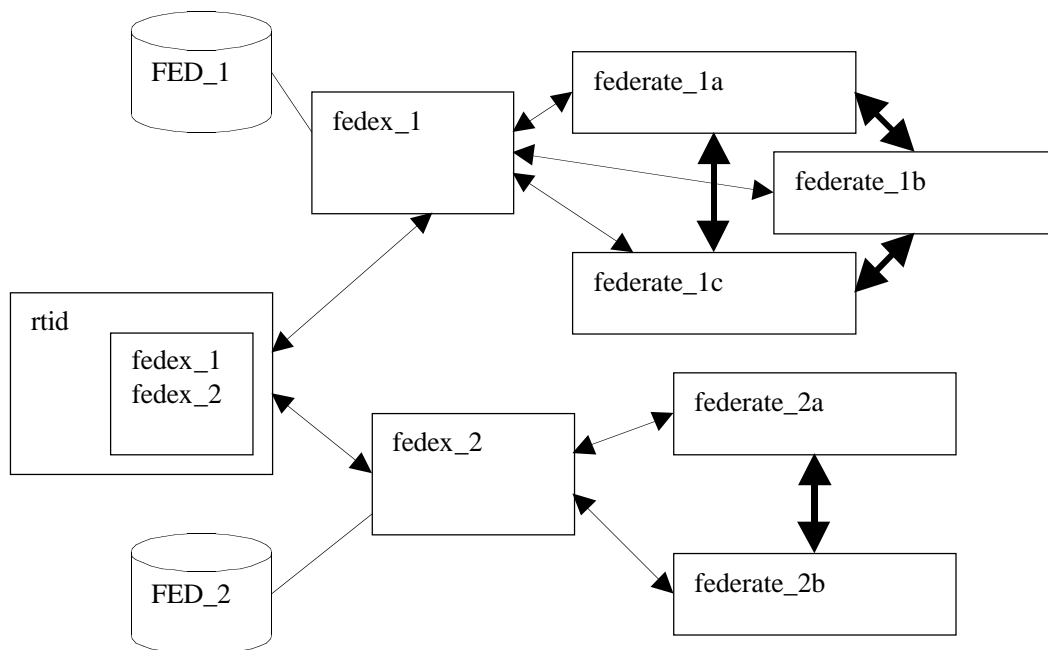
## yaRTI : pourquoi une RTI en Ada 95?

- Beaucoup de développements de simulation sont en Ada
- Le besoin est réel d'une RTI d'architecture ouverte et orientée vers l'étude algorithmique, sans préoccupation de plus bas niveau
- L'Annexe E est la base technique idéale pour ce type de développement
- Portabilité : GNAT/GLADE sont disponibles sur de nombreuses plates-formes
  
- Et puis c'est marrant :-)

## Conception de yaRTI : idées directrices

- Arriver à **une implémentation totalement distribuée**, sans aucun traitement centralisé
- Obtenir **une portabilité maximale** en n'utilisant que des traits du langage (Annexe E, tasking, objets protégés, etc.)
- Bien **modulariser** les services pour faciliter les **changements d'algorithmes**
- Fournir une API et un comportement au standard 1.3, avec la possibilité d'évolution vers le standard IEEE P1516

## yaRTI - architecture globale - 1



10 Nov. 2000

Ada-France

15

## yaRTI - architecture globale - 2

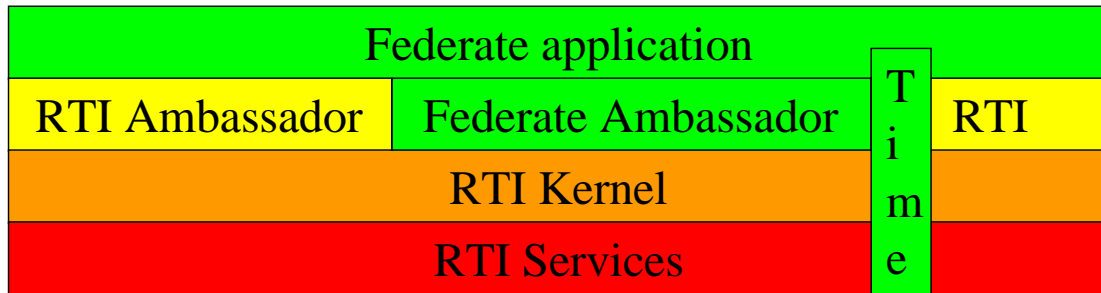
- ‘[rtid](#)’ n’est pas la RTI. Il ne gère que les exécutions de fédérations. [rtid](#) contient l’unique paquetage RCI.
- ‘[fedex](#)’ n’est pas la RTI. Il gère une exécution de fédération en fournissant à chaque fédéré :
  - Un RACW pour lui-même et les autres fédérés
  - Les données d’exécution (extraites du FOM), pour être sûr que tous les fédérés ont les mêmes
- La vraie RTI est distribuée, « liée » avec les applications fédérées.

10 Nov. 2000

Ada-France

16

# yaRTI, architecture



- ‘RTI Ambassador’ : services à disposition du fédéré
- ‘Federate Ambassador’ : services à disposition de la RTI (callbacks)
- ‘RTI’ : types de base
- ‘RTI Kernel’ : tâches et gestion de base
- Services : implémentation de l’I/F Spec.
- Le temps est défini par l ’utilisateur (type abstrait)

10 Nov. 2000

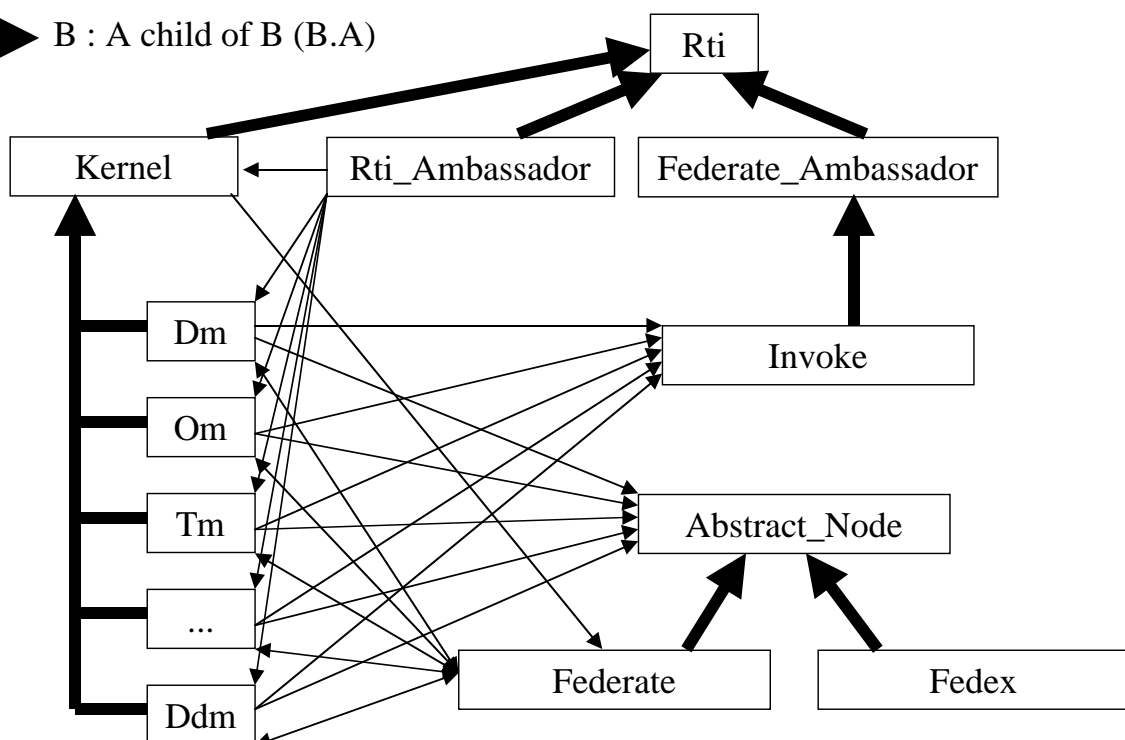
Ada-France

17

## yaRTI, hiérarchie de paquetages (fédéré)

A → B : A with B

A → B : A child of B (B.A)



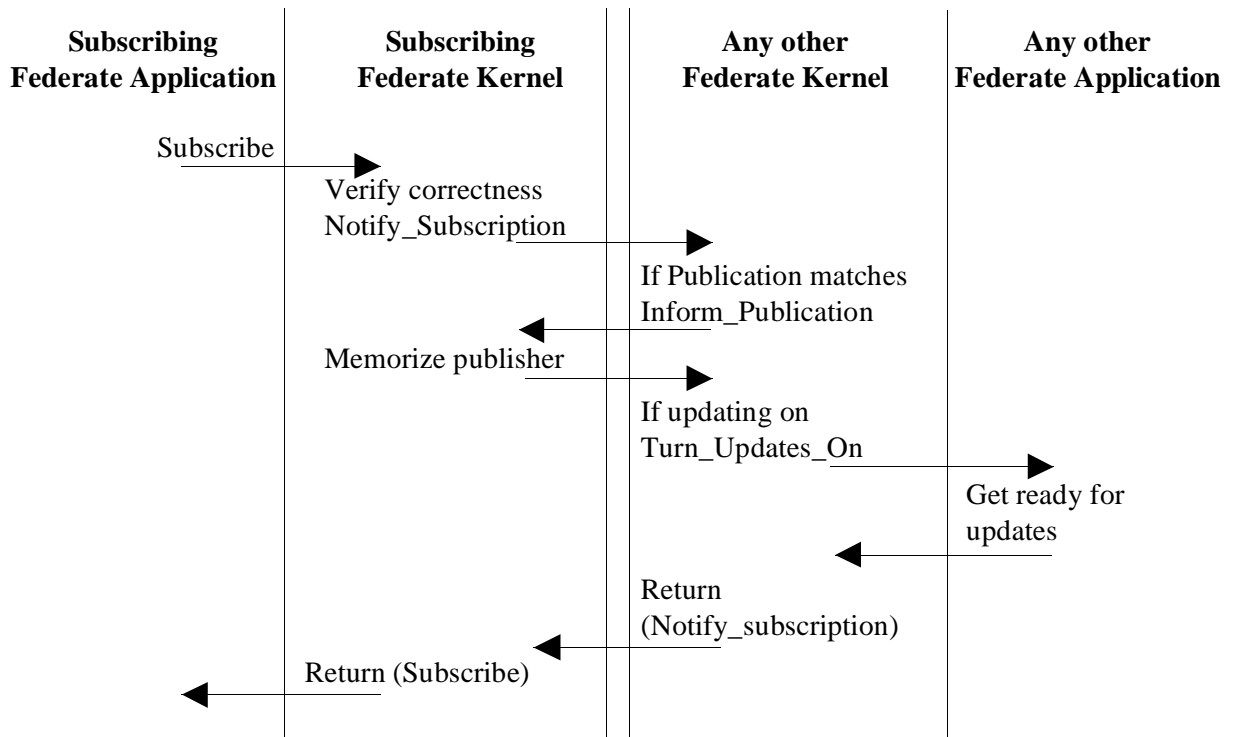
10 Nov. 2000

Ada-France

18



# Exemple d 'implémentation : la souscription

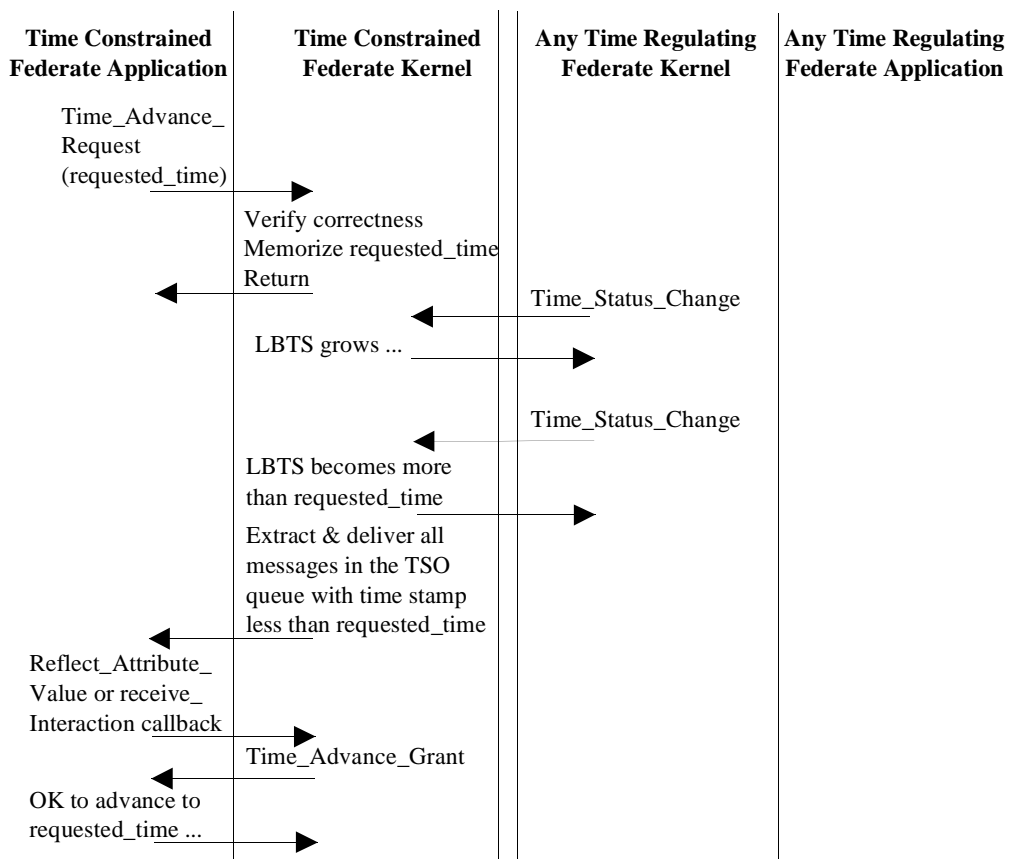


10 Nov. 2000

Ada-France

21

# Autre exemple : avance coordonnée du temps



10 Nov. 2000

Ada-France

22

# Implémentation : transport et remise des données

- Deux qualités de transport : ‘reliable’ et ‘best effort’.
  - Le transport fiable utilise des appels RACW normaux (résultat garanti, mais latence)
  - Le transport ‘Best effort’ utilise des appels RACW asynchrones (pas d’attente de fin d’exécution => latence réduite)
- Deux modes de remise : RO (Receive Ordering) et TSO (Time Stamp Ordering). Seuls les fédérés régulateurs du temps peuvent émettre des messages TSO, seuls les fédérés asservis au temps peuvent en recevoir. L’ordre est global à la fédération.
  - Les messages RO entrants sont gérés par une FIFO
  - Les messages TSO entrants sont gérés par une file triée
  - Une tâche consomme chaque file, quand le temps peut avancer

## Quelques données

- Moins de 10 000 lignes de cœur algorithmique
  - moins de 2000 h de conception - réalisation
  - utilisée dans des applications réparties réelles
- Performance globale de yaRTI environ 30% en-dessous de la RTI NG du DMSO, sans effort d’optimisation particulier

# Conclusion

- Développée à très faible coût, yaRTI est:
  - un bon support pour l'étude des algorithmes distribués d'une RTI HLA,
  - réellement utilisable,
  - ouverte et libre (GPL adaptée)
- Reste à faire :
  - MOM à compléter
  - Data Distribution Management à terminer
  - Ownership Management à implémenter
  - Performance à améliorer

## Conclusion - 2 - Open Source

- Bon pour la paranoïa des utilisateurs Défense :
  - Pas de chevaux de Troie possibles
  - Souscription prise en compte à la source (contrairement à la RTI du DMSO)
  - Cryptage possible par canal entre couple de fédérés
- Bon pour le développement et la qualité du produit
  - Les utilisateurs peuvent critiquer et suggérer, voire améliorer et tester leurs idées
  - Des communautés non militaires peuvent trouver des applications (télé opération, applications civiles de simulation, etc...)

# Many Thanks...

- To DMSO 's Chief Scientist Judith Dahmann, for very large parts of the HLA presentation,
- To Ada Core Technologies and especially the GLADE team (Laurent, Sam and all the gang).
- To Laurent Guerby and Philippe Annic, for the initial 'why not...'

## Links

- yaRTI is available at:

<http://perso.wanadoo.fr/dominique.canazzi/dominique.htm>

- HLA documentation, guidance, software and more are available at:

<http://hla.dmsso.mil>