



Università degli Studi di Palermo
Dottorato di Ricerca in Ingegneria Informatica (XVI ciclo)

Dipartimento di Ingegneria Informatica

**Peer-To-Peer Architectures in
Distributed Data Management Systems
for Large Hadron Collider Experiments**

Candidato
Giuseppe Lo Presti

Collaborazioni:



European Organization for Nuclear Research
PH Department, CMS TriDAS Group

Tutor
Prof. Salvatore Gaglio



Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad alte prestazioni

Coordinatore
Prof. Alessandro Genco

Anno Accademico 2003 – 2004

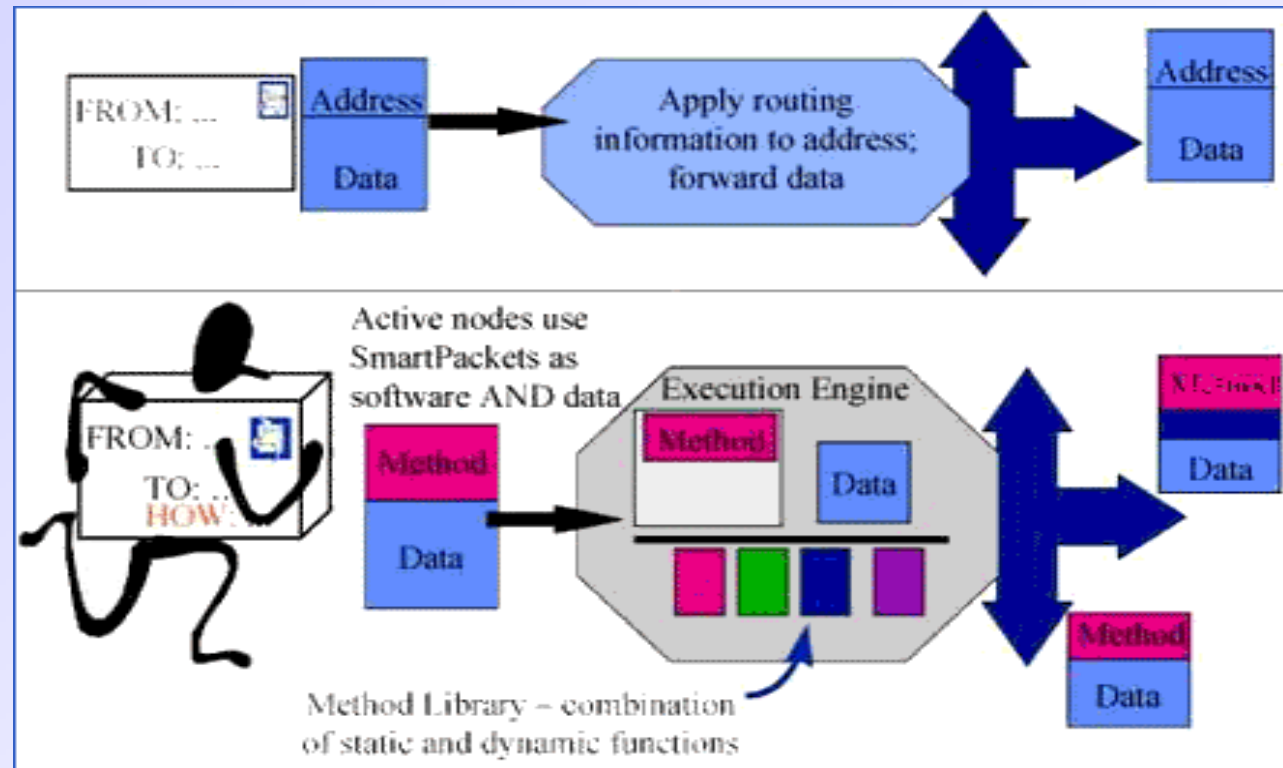
Palermo, 4 Febbraio 2005

Sommario

- ❖ Infrastrutture per applicazioni distribuite: stato dell'arte
 - Active Networking
 - Peer-to-Peer
- ❖ ANgate: un'architettura di gestione per reti attive
 - Gestione di AN e Applicazioni Attive
 - Caso di studio: gestione intelligente di rete
- ❖ JxtaPT: un'architettura per il *discovery* in sistemi di acquisizione dati per esperimenti di fisica delle alte energie
 - Il contesto: il CERN, il LHC e il *Data Acquisition* di CMS
 - La piattaforma Jxta per applicazioni P2P
 - Caso di studio: un servizio di discovery per XDAQ
- ❖ Conclusioni e sviluppi futuri

Sistemi distribuiti: le Reti Attive

- ❖ Le Reti Attive (**Active Networks**) introducono programmabilità nella rete
- ❖ Caratteristiche chiave
 - Deployment di soluzioni *ad hoc* e di protocolli dedicati nei nodi attivi
 - Applicazioni “attive” distribuite invece di end-to-end
 - Duale rispetto al paradigma della programmazione ad *Agenti Mobili*



[DARPA, 1998]

Problemi aperti:

- Sicurezza (security & safety)
- Gestione
- Prestazioni
- Killer applications



Piattaforme per AN: PLAN

- ❖ Packet Language for Active Networks
 - Un linguaggio ed un ambiente di esecuzione per la programmazione di reti attive secondo il modello a **capsule**
 - Supporto per servizi locali residenti nei nodi
- ❖ Linguaggio
 - Basato su ML
- ❖ Maturità e supporto
 - Prototipo. Sviluppato da UPenn ma non più supportato attivamente
- ❖ Vantaggi
 - API flessibile per lo sviluppo di nuovi protocolli e la sperimentazione di applicazioni distribuite in rete
- ❖ Svantaggi
 - **Gestione della sicurezza assente**
 - Prestazioni, fault tolerance

Sistemi distribuiti: le reti Peer-to-Peer

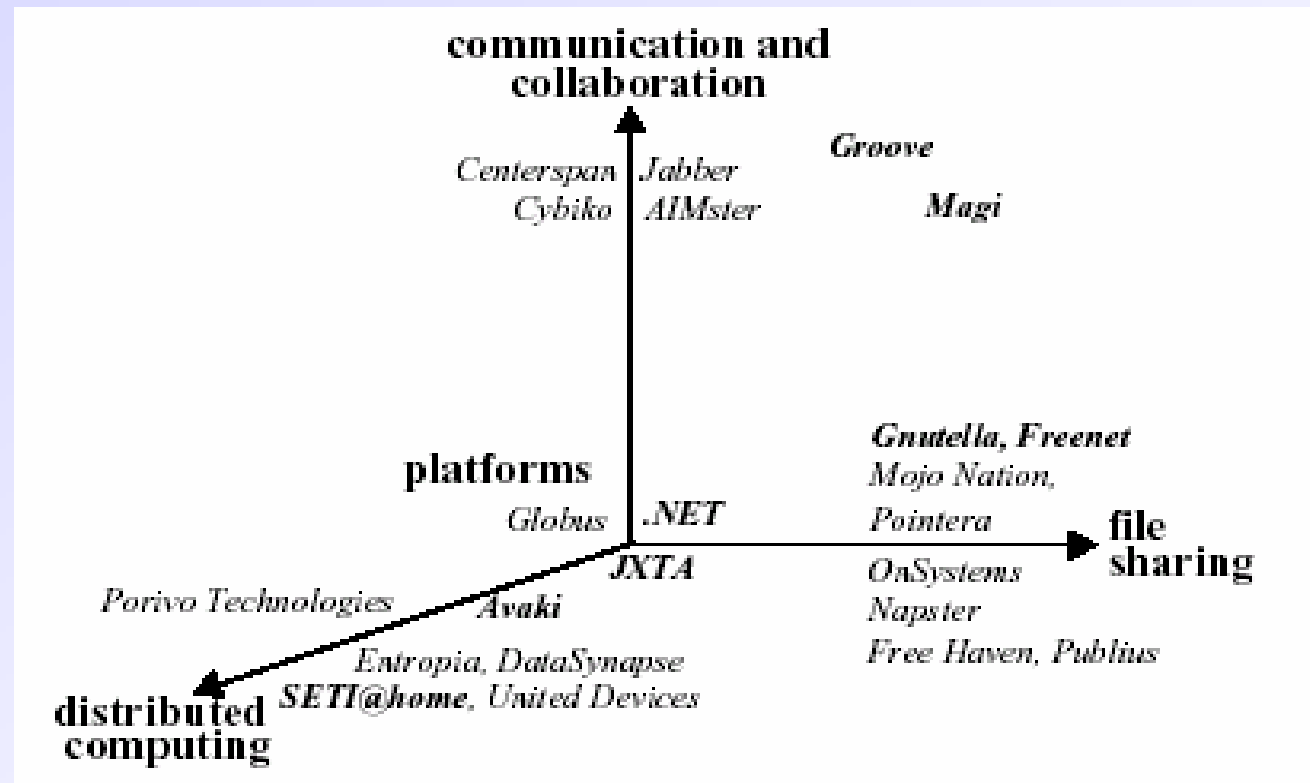
❖ Una definizione di P2P:

Classe di sistemi ed applicazioni che impiega **risorse distribuite** per svolgere un **compito critico** in **maniera decentralizzata**

- **Risorse distribuite**: CPU, spazio di storage, banda, contenuti, presenze umane
- **Compito critico**: computazione distribuita, condivisione e scambio di dati, comunicazione e collaborazione, servizi di piattaforma
- **Decentralizzazione**: algoritmi, dati e meta-dati

Lo spazio delle applicazioni

- ❖ Condivisione di dati
- ❖ Computazione distribuita
- ❖ Piattaforme di middleware
- ❖ Ambienti collaborativi



Source: Peer-to-Peer Computing, HP labs, March 2002

Problemi aperti:

- Gestione
- Sicurezza (security)
- Scalabilità



Piattaforme per P2P: SETI@home

- ❖ Un sistema **C/S** “rovesciato” per **distributed computing**
 - Ciascun “peer” riceve un job e lo esegue durante i tempi di non utilizzo (tipicamente il software client è uno screen-saver)
 - Un database **centrale** mantiene tutti gli account (nessun discovery)
 - Si sta trasformando in un’applicazione costruita sopra la piattaforma *BOINC*
- ❖ Protocolli
 - Proprietari su TCP/IP
- ❖ Maturità
 - Sistema stabile (iniziato nel 1996). Ha ispirato altri progetti *@home
- ❖ Supporto
 - Sponsorizzato tra gli altri da Sun
- ❖ Vantaggi
 - **Scalabilità dimostrata** (~ 5.300.000 utenti) dato che il costo di comunicazione è molto più basso del costo computazionale.
 - **Buona fault resilience**: i client memorizzano il loro stato ogni 1-2 minuti e il server è un grado di riassegnare un job dopo un timeout
- ❖ Svantaggi
 - Orientato specificamente a computazioni coarse-grain e CPU-intensive, **la piattaforma non è general purpose.**



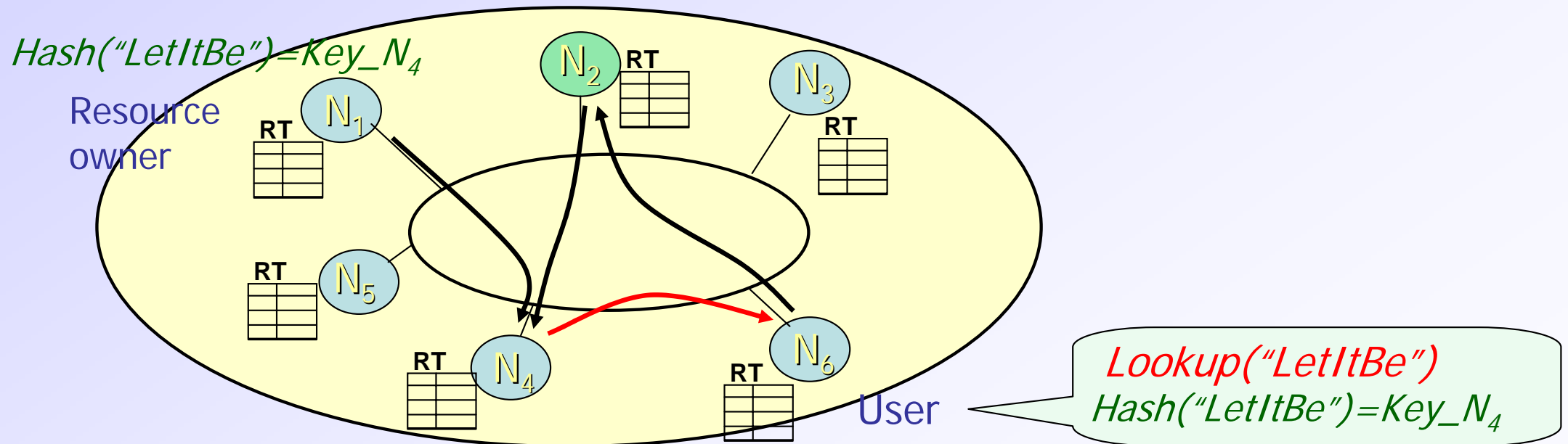
Piattaforme per P2P: Gnutella

- ❖ Un **protocollo** di P2P **puro** per il **file sharing**
 - Query basate su **flooding**: ciascun peer fa il broadcast delle query
 - numero di messaggi richiesti ~ **$O(d \wedge TTL)$** , d = grado medio dei nodi
 - **Autodiscovery distribuito**: ciascun peer si annuncia agli altri inviando un comando PING e risponde ad altri annunci inviando un comando PONG
- ❖ Protocollo
 - Protocollo proprietario molto basilare basato su TCP (comandi supportati: PING/PONG per il discovery, QUERY/QUERYHIT per le query, PUSH per attraversare i firewall)
- ❖ Maturità
 - Sistema ben studiato in quanto ne è stato fatto il reverse engineering. Sono presenti diversi cloni e diverse applicazioni Internet basate su questo protocollo
- ❖ Vantaggi
 - Fault tolerance (anche se non esplicitamente prevista nel protocollo) se la rete overlay include un numero sufficiente di peer
- ❖ Svantaggi
 - L'overhead di comunicazione elevato (flooding) **limita la scalabilità**
 - Non garantisce risultati sulle query: non ci sono limiti temporali, e **una query può anche fallire** (cioè la risorsa cercata non viene trovata pur se presente nella rete)

- ❖ **Piattaforma general-purpose e open source per sistemi P2P**
 - **Jxta = Juxtapose**: peers vs. C/S
 - Architettura Orientata agli Oggetti
 - Supporto per **super peers** e **peer groups** per overlay networks strutturate
- ❖ **Protocolli**
 - **basati su XML**, trasporto utilizzato TCP/IP, Bluetooth, etc.
- ❖ **Linguaggi**
 - Tra le altre sono disponibili implementazioni Java e C
- ❖ **Configurazione**
 - Sviluppo e deployment di applicazioni Java-based relativamente semplice
 - Integrazione con C/C++ più complessa in quanto l'API è ancora in fase di sviluppo ed è disponibile poca documentazione
- ❖ **Maturità e supporto**
 - Livello prototipale. Attivamente supportato da Sun e dalla comunità Java
- ❖ **Vantaggi**
 - **Servizi distribuiti di *discovery/indexing* forniti nativamente**
 - Supporto alla localizzazione di risorse presenti dietro firewall o NAT
 - Rappresentazione dei dati e dei metadati basata su XML
- ❖ **Svantaggi**
 - Scalabilità nel **naming a livello globale**: non è garantito che gli ID siano unici

Discovery: indicizzazione distribuita delle risorse

- ❖ Obiettivo: mantenere un database *indicizzato* delle risorse *online*
 - Vanno considerate nuove risorse rese disponibili e quelle non più disponibili
 - Va distribuito tra alcuni o tutti i peer partecipanti
 - L'indicizzazione deve prescindere da ID dipendenti dalla rete sottostante
- ❖ Approccio tramite **Distributed Hash Tables**
 - Complessità di lookup minima
 - Gli indici vanno tenuti aggiornati – operazione potenzialmente costosa
 - Che succede se il peer che indicizzava una risorsa diventa offline?



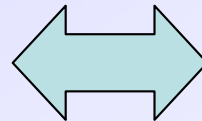
**ANgate:
un'architettura di gestione per Reti Attive**

Motivazioni e scopi

❖ Le Reti Attive per il management

- Disponibilità di informazioni nei nodi intermedi
- Capacità di calcolo lungo il percorso
- Possibilità di adottare strategie distribuite (cfr. Agenti Mobili)

Queste caratteristiche consentono una gestione di rete più flessibile



Una Rete Attiva necessita di una gestione più sofisticata

- Gestione delle AA

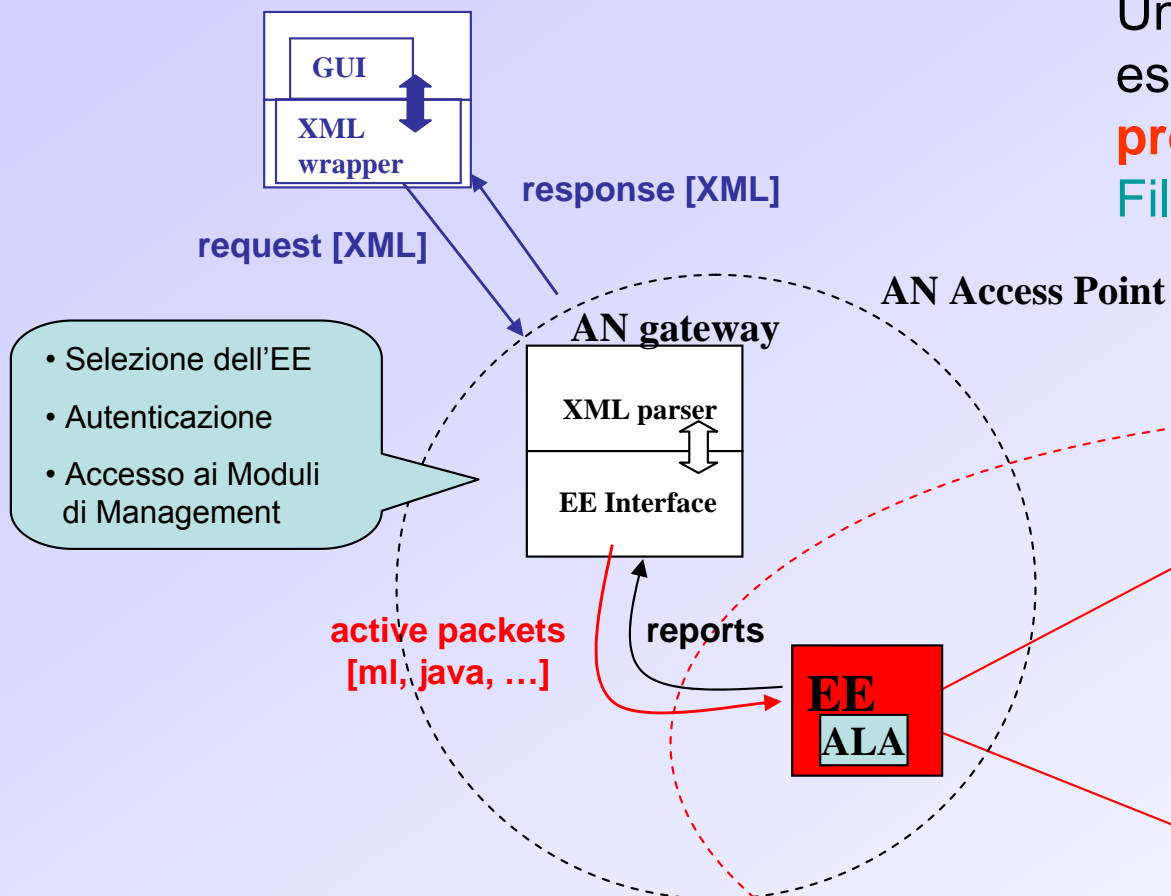
❖ Scopo:

- Progettazione di un prototipo di architettura per la gestione di rete che utilizza la tecnologia delle AN

ANgate

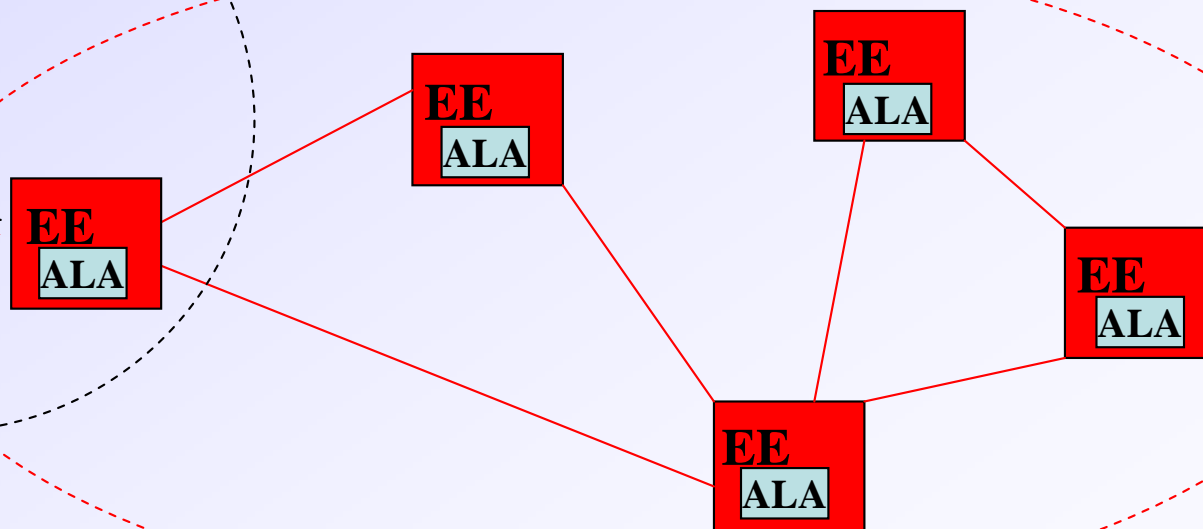
Architettura di ANgate

Modulo di Management



Un **Agente Locale Attivo** (**ALA**) è in esecuzione sui nodi e fornisce servizi **programmabili** secondo un modello **Filtri-Eventi-Azioni**.

Active Network



EE = Execution Environment

Moduli di Management

- Editor di topologie
- Setup della rete (solo admin)
- Discovery della topologia
- Stato di nodi e link
- Immissione (*injection*) di capsule attive
- GUI verso i servizi di ALA
- Tracing e debugging di applicazioni attive
- Misure di performance

The image shows two overlapping software windows. The top window, titled 'AN Manager', displays a network topology diagram with nodes 0 through 9 and connecting links. A sidebar on the right shows details for 'Node id: 6', including its real host, IP address, EE port, interface ports, state, and ALA version. The bottom window, titled 'AA Monitor', shows a tree view of the network structure under 'ANet' and a table of active capsules.

Node id	AA name	Var name	Type	Value
8	timestamp	back	string	1027951513.664335
0	pubSample	string	string	foobar
3	timestamp	back	string	1027951554.284142
3	timestamp	fwd	string	1027951554.240450
8	timestamp	fwd	string	1027951513.645486

```
updating vars table.
<msg>: msg from 147.163.4.13:3601: START PLANet Navigation along a tree rooted at:
      angate.cere.pa.cnr.it:3601 (147.163.4.13:3601) (gwport=4444) (RB=9999)
<msg>: timestamp results from 147.163.4.13:3601:
(answer from 147.163.4.13:3601,
 {fwd=1027947965.846663, back=1027947965.952026})
(answer from 147.163.4.13:3601
```

Caso di studio: un sistema di gestione intelligente

❖ Verso l'**autonomia**

- ANgate è un'interfaccia per l'amministratore di rete
- Entro che limiti è possibile automatizzare i processi di gestione?

❖ Obiettivo: design di un sistema **logico** di Network Management

❖ Uso di tecniche di Intelligenza Artificiale per

- Diagnosi delle **cause generatrici** dei fault di rete
- **Correlazione** tra eventi diversi e inferenza di singoli eventi complessi

Il contesto di IA: il Situation Calculus

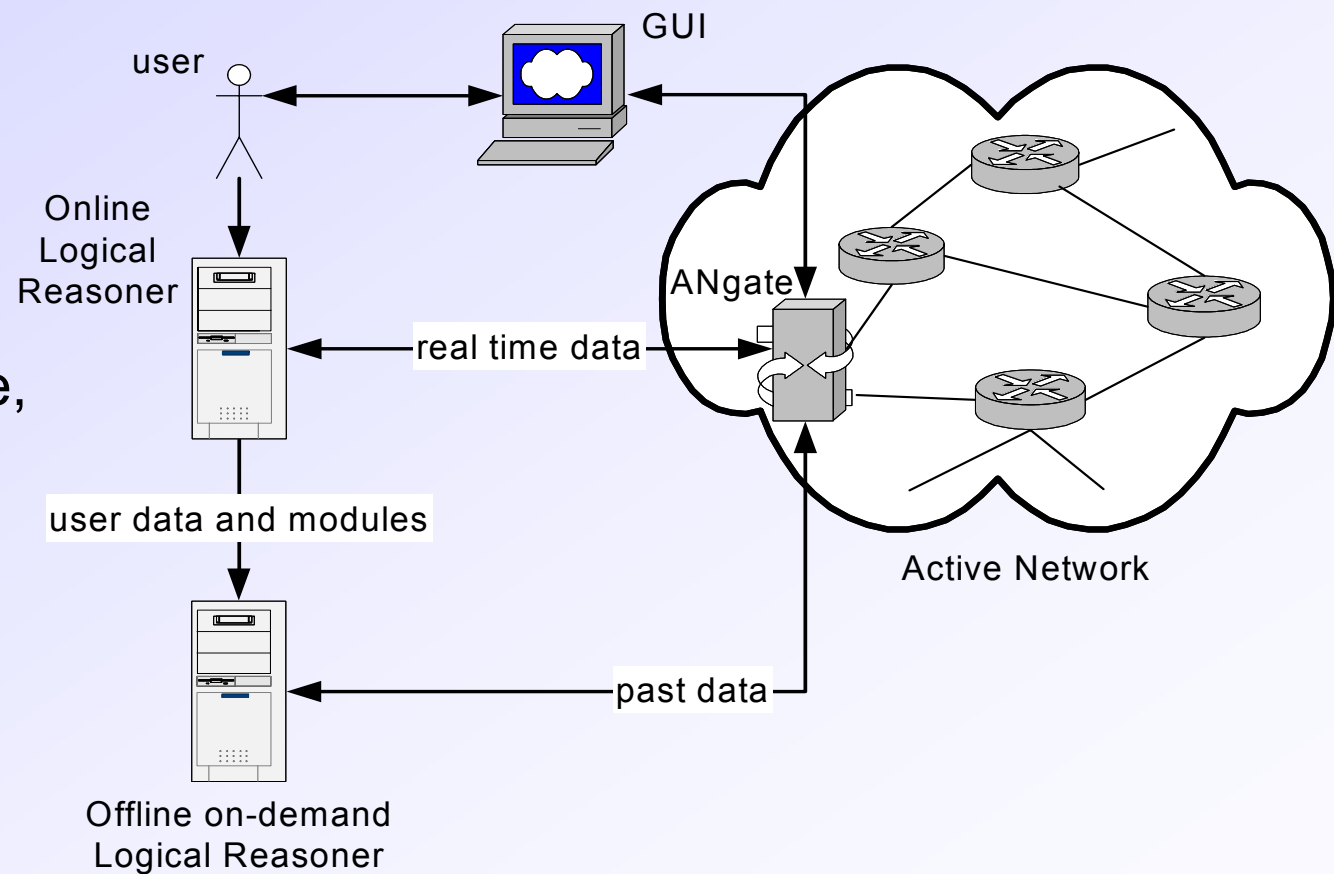
- ❖ Dialetto della **Logica del Primo Ordine**
- ❖ **Situazione** = stato di un sistema dinamico
 - Il tempo non viene gestito direttamente ma l'ordine temporale di ciascuna situazione viene preservato
- ❖ Implementato tramite il linguaggio logico **Golog** [Reiter]
- ❖ Concetti chiave:
 - **azioni primitive**, per catturare i cambiamenti del mondo
 - **situazioni**, per catturare lo stato
 - **fluenti**, predicati logici il cui valore è funzione dello stato
- ❖ **Reactive Golog**
 - Un'estensione per tenere conto delle azioni del mondo esterno
 - Introdotto il concetto di **procedura**:
gruppo di azioni eventualmente asserite dall'esterno durante il processo di ragionamento in risposta a nuove situazioni

Architettura del ragionatore logico

- ❖ La rete è controllata dal sistema ANgate
 - Disaccoppiamento rispetto all'EE
 - Interazione ragionatore ↔ rete tramite comandi XML

- ❖ Il ragionatore *online* inferisce continuamente lo stato della rete

- ❖ Quando richiesto dall'utente, un'altra istanza *offline* può svolgere indagini su eventi passati



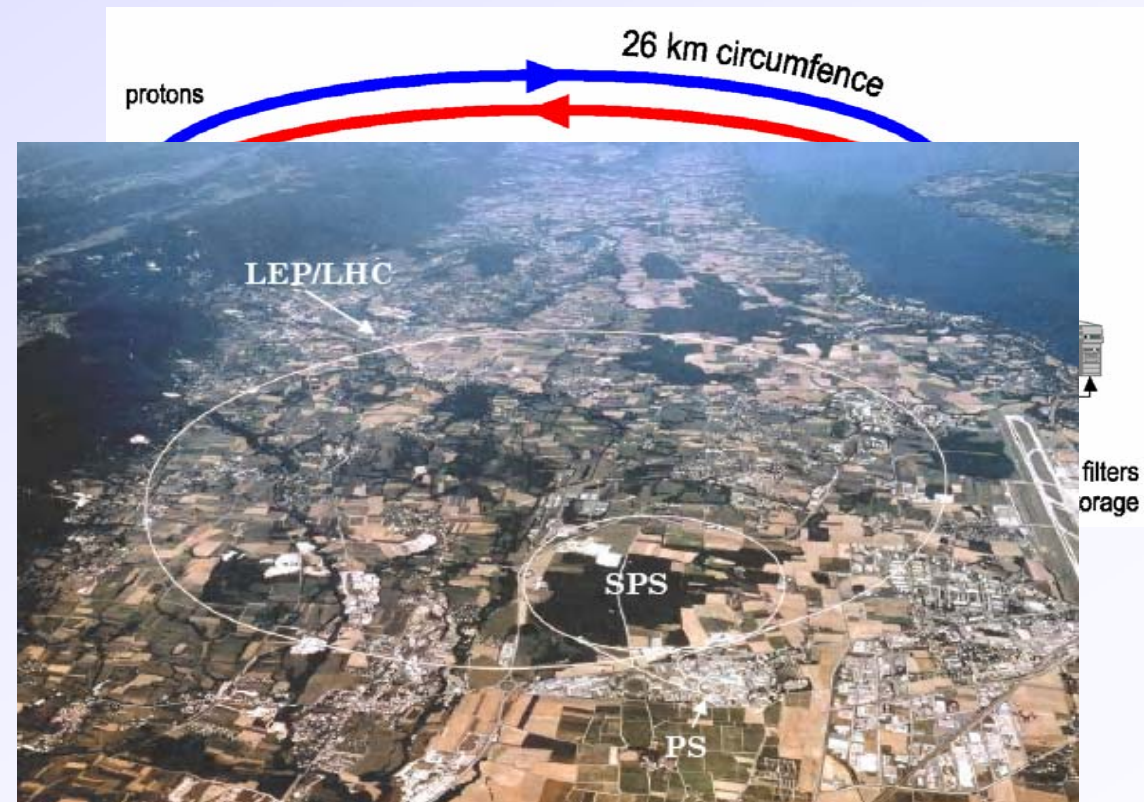
ANgate: esperimenti e risultati ottenuti

- ❖ Guasti trattati: livello di rete
 - Caduta di nodi e link
 - Errori nelle tabelle di routing
- ❖ Scenario: generazione di guasti artificiali nella rete
 - Distribuzione uniforme nello spazio e poissoniana nel tempo
- ❖ Risultati: tutti i guasti sono stati identificati entro **$O(10 \text{ sec})$**
 - Nei casi applicabili, le **azioni di ripristino** delle condizioni di funzionamento sono state inferite dal sistema ed eseguite da ANgate
 - **La complessità del processo di inferenza logica limita la scalabilità del sistema**

JxtaPT: un'architettura per il discovery in sistemi di acquisizione dati

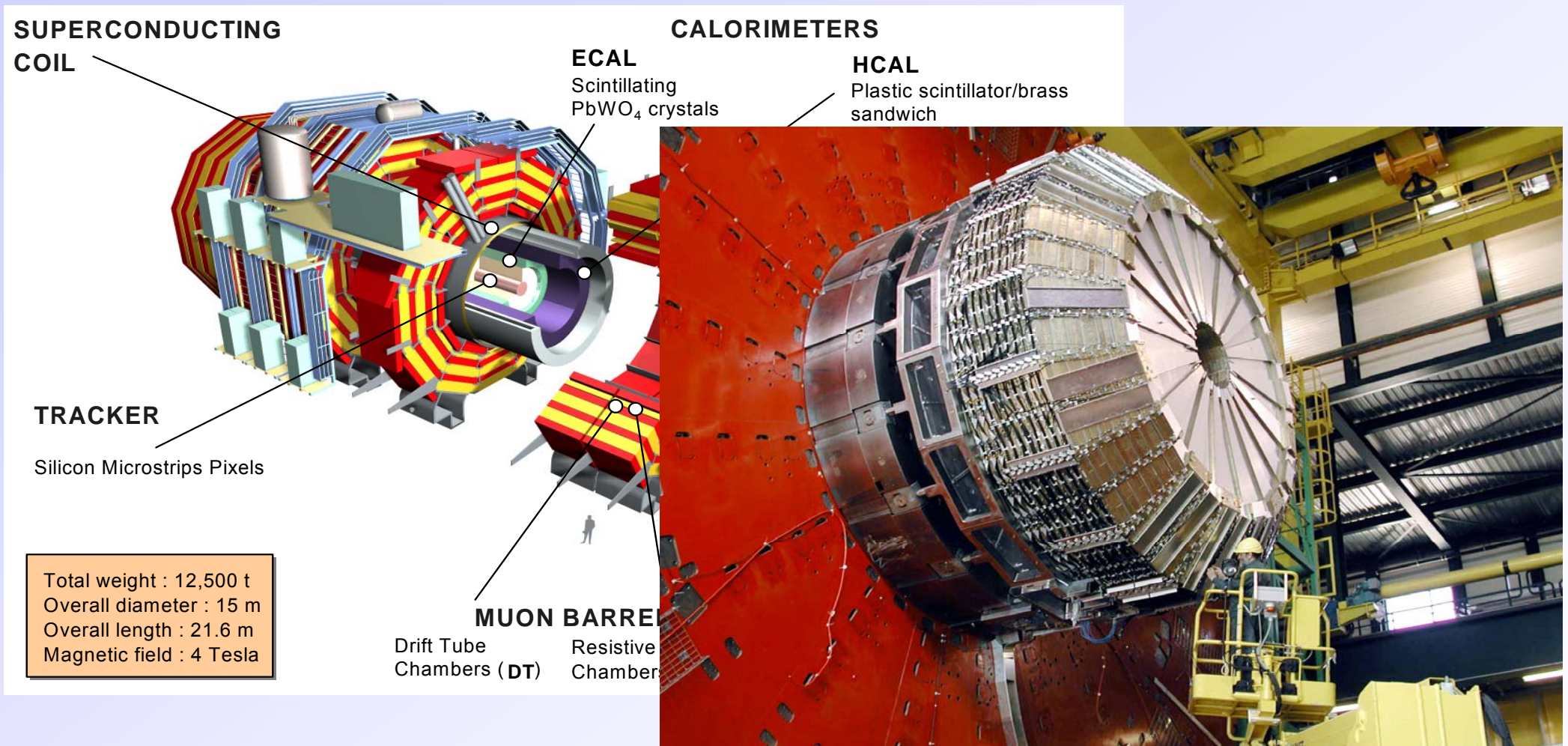
Il CERN e il Large Hadron Collider

- ❖ Il CERN, Organizzazione Europea per la Ricerca Nucleare, si occupa di costruire e supportare *Acceleratori di particelle* per la comunità di ricercatori in Fisica delle Alte Energie (HEP, High Energy Physics)
- ❖ Il *Large Hadron Collider (LHC)* è un acceleratore protoni-protoni ad alte prestazioni, dove le particelle vengono portate a collisione all'interno di opportuni detector
 - In funzione a partire dal 2007
 - Energia massima superiore al TeV
 - Frequenza di lavoro: **40 MHz** (collisioni al secondo)



L'esperimento CMS

- ❖ CMS (*Compact Muon Solenoid*) è un detector per lo studio della Fisica delle particelle che sfrutta la massima *luminosità* del LHC.

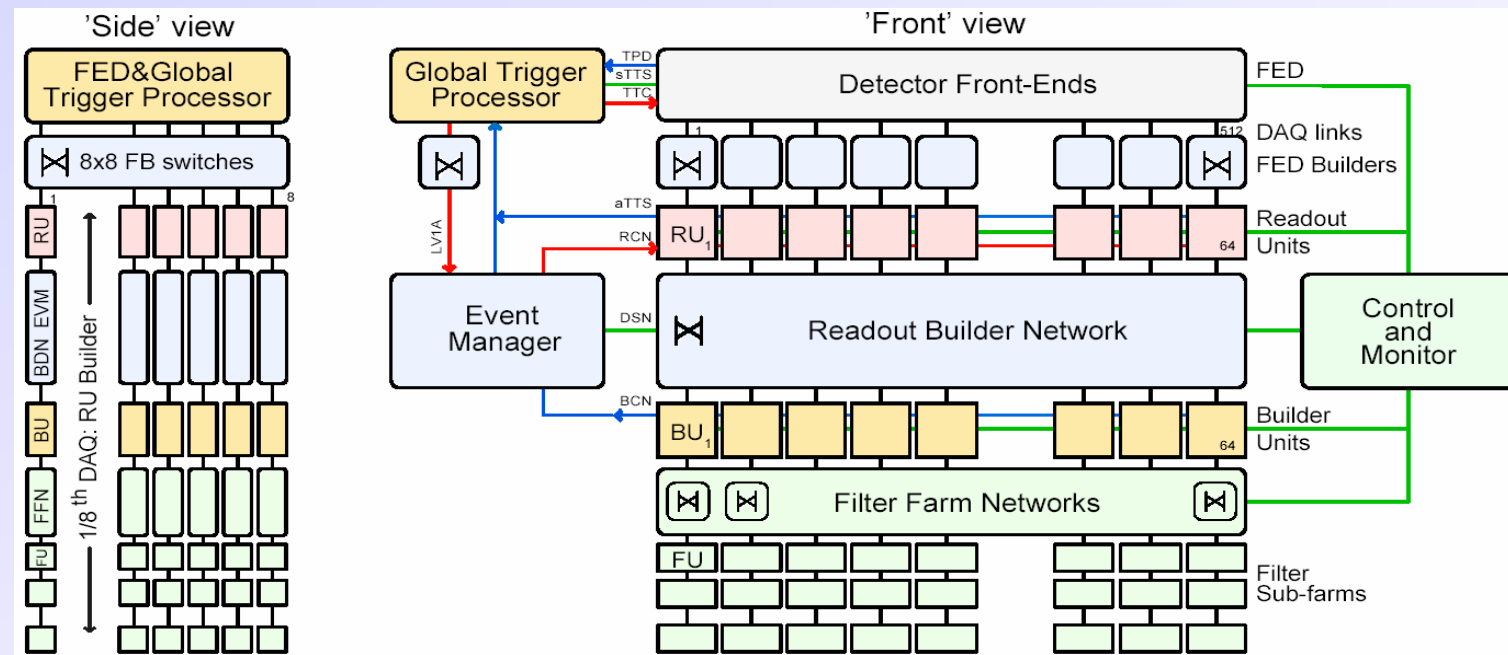


Sistemi di Data Acquisition (DAQ)

❖ Un sistema di acquisizione online di dati (DAQ) per esperimenti HEP deve combinare tutti i dati misurati dai detector e filtrare tramite algoritmi veloci gli eventi significativi per un'analisi successiva più accurata.

❖ Alcuni parametri per CMS:

- flusso entrante: 100 Gb/sec
- flusso uscente: 100 Mb/sec
- dimensione farm: circa 5000 PC, 10^6 MIPS
- dimensione media evento da trattare: 1 Mb



❖ Piattaforma Peer-to-Peer per il Data Acquisition (DAQ)

- Consente di sviluppare applicazioni di acquisizione dati online per esperimenti di fisica delle alte energie (HEP)
- Supporta diverse piattaforme hardware (**cross platform**) dedicate al DAQ esponendo una interfaccia di programmazione omogenea
- Include il supporto per applicazioni con interfaccia **web**

❖ Protocolli

- SOAP per messaggi di controllo testuali, **I₂O/IP** per trasporto di dati binari

❖ Linguaggi

- Implementazione C++

❖ Maturità e supporto

- In produzione da Dic 2004. Supportato dal gruppo di lavoro CMS DAQ del CERN

❖ Vantaggi

- Scalabile, overhead limitato in termini di memoria occupata
- Supporto per diverse piattaforme
- Modulare ed estensibile

❖ Svantaggi

- **Non prevede un servizio di discovery distribuito delle risorse**

Caso di studio: un servizio di discovery per XDAQ

- ❖ **Scopo:** implementazione e deployment di un protocollo per il discovery distribuito per la piattaforma XDAQ
 - Sistema a configurazione zero
 - Integrazione in XDAQ come modulo caricabile dinamicamente
- ❖ **Perché XDAQ?**
 - L'intrinseca complessità di questo ambiente consente di testare l'efficacia dei servizi di discovery in un contesto reale
 - Un servizio di discovery distribuito può essere utile alle applicazioni XDAQ
 - XDAQ è già una piattaforma P2P

Quale piattaforma Peer-to-Peer?

❖ JXTA

- Piattaforma generica open-source per applicazioni P2P
- Indicizzazione distribuita e *discovery* delle risorse forniti nativamente
 - Basata su *DHT* “*loosely coupled*”
- Progetto molto attivo e supportato
- Basato su standard di mercato (XML, SOAP, ...)
- **Facilità di personalizzazione**
 - Eventuali problemi minori possono essere risolti con interventi mirati, grazie alla modularità del codice sorgente

Jxta per il discovery in XDAQ

❖ Scenario

- Deve esistere un *superpeer* (*rendez-vous*) in esecuzione su ciascun nodo di rete che appartiene a più segmenti diversi (i.e. gateway/firewall).
- La rete deve supportare l'IP multicast.

❖ Peer Jxta-J2SE

- Implementano i *rendez-vous* e mantengono un registro distribuito di tutti i peer e le risorse online.

❖ Demoni XDAQ

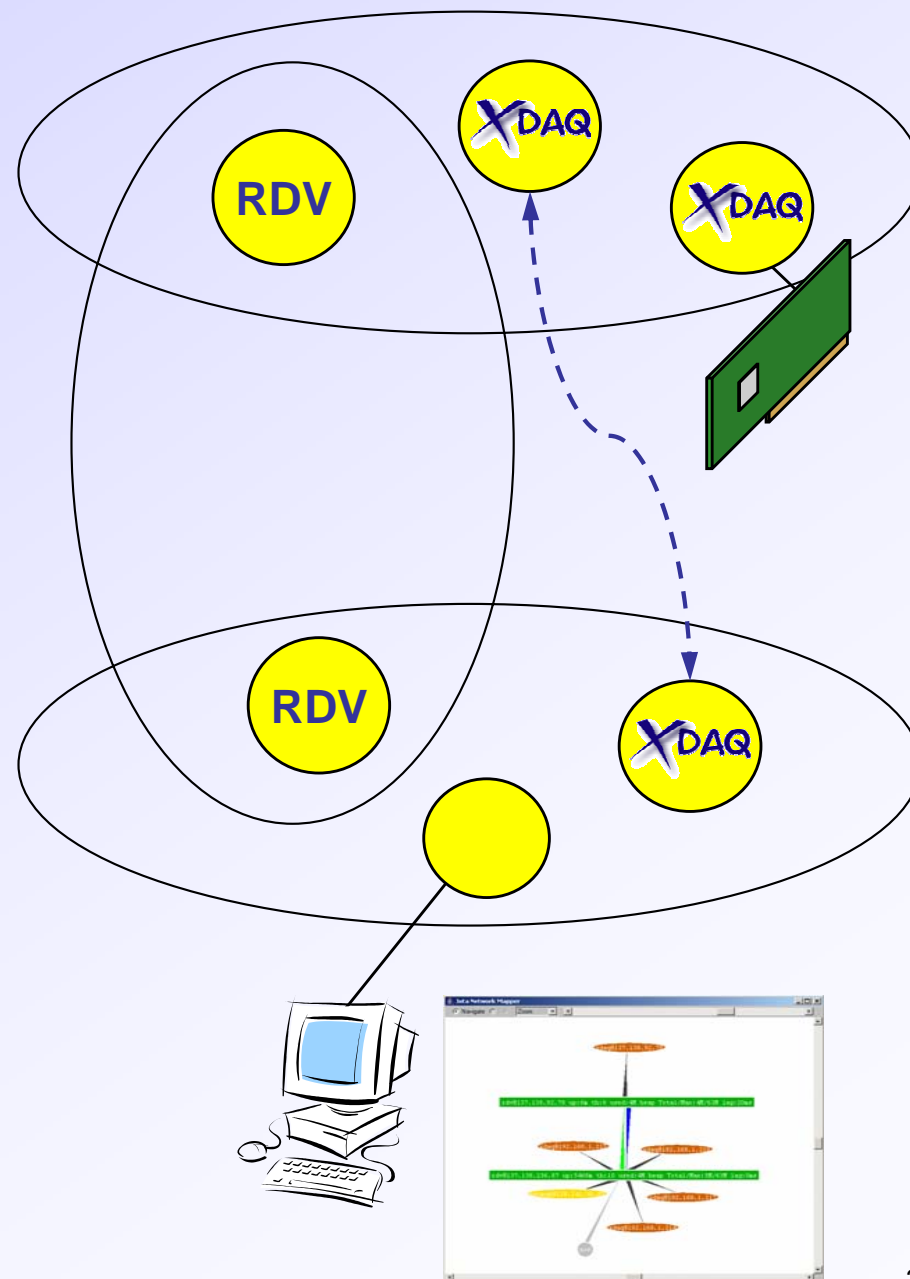
- possono caricare il modulo di discovery per pubblicare la propria presenza in rete e partecipare alla rete overlay di Jxta.

❖ GUI Java per il monitoraggio

- basata su Touch Graph
- include un peer Jxta-J2SE
- consente query verso i peer XDAQ presenti online

❖ Forwarding di messaggi

- La overlay network è trasparente ai bordi tra reti pubbliche/private diverse

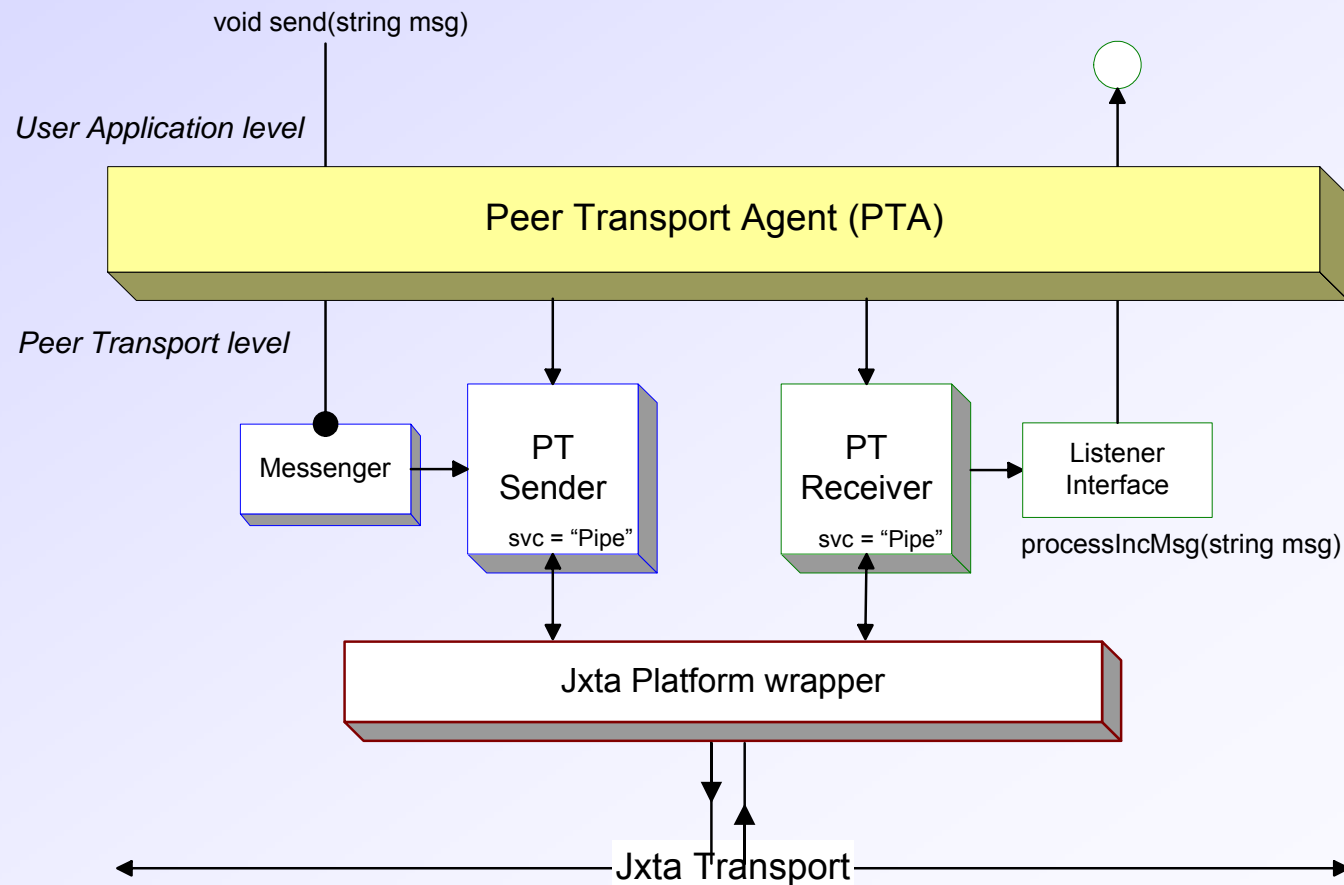


Architettura di JxtaPT

❖ **JxtaPT**: un *PeerTransport* per consentire ad applicazioni XDAQ di interagire con la rete Jxta

❖ **Jxta Platform Wrapper**:

- Un'interfaccia API astratta per esporre i servizi Jxta
- Un package di classi C++ che implementano i servizi esposti
 - Integra l'implementazione **Jxta-C**
 - Un'altra implementazione può essere integrata senza modificare l'API astratta
- Una *XDAQ application* fornisce un'interfaccia **web** alla rete Jxta.



JxtaPT: esperimenti e risultati ottenuti

- ❖ Test scenari di discovery
 - Tutti i demoni XDAQ sono stati identificati a livello Jxta
- ❖ Instaurazione di rete *overlay* su rete pubblica/privata
 - Il PeerTransport consente la messaggistica ove il livello rete (TCP, HTTP) è bloccato
- ❖ Misura dell'occupazione di risorse
 - Unico parametro significativo: RAM
 - Occupazione trascurabile per il modulo JxtaPT
 - Nessuna funzionalità di super-peer
 - Occupazione più consistente per i super-peer
 - Le prestazioni di XDAQ non sono inficiate

Conclusioni

- ❖ Sono stati investigati nuovi paradigmi per il design e lo sviluppo di architetture distribuite
- ❖ Sono stati affrontati casi di studio innovativi per proporre soluzioni originali a problemi attuali
- ❖ I prototipi sviluppati sono stati testati in laboratorio
- ❖ I risultati ottenuti sono stati oggetto di pubblicazioni scientifiche
- ❖ Sviluppi futuri
 - Discovery API per XDAQ

Grazie per l'attenzione

❖ Contatti

Giuseppe Lo Presti

CERN – Ginevra (Svizzera)

Giuseppe.LoPresti@cern.ch

www.cern.ch/lopresti

❖ Ringraziamenti

- Dr. Johannes Gutleber, CERN, PH/CMD
- Prof. Giuseppe Lo Re, CNR, ICAR
- Dr. Luciano Orsini, CERN, PH/CMD

