

Perche' cambiare?

- Utilizzo di strumenti di programmazione più moderni:
 - “orientati agli oggetti” (O.O.)
 - “multithreaded”
- Supporto per linguaggi di programmazione O.O.
 - C++, python, java
- Ci aspettiamo:
 - tempi di sviluppo ridotti
 - architettura più “pulita”: meno errori
 - Manutenzione ed evoluzione più facili

Criteria per la scelta

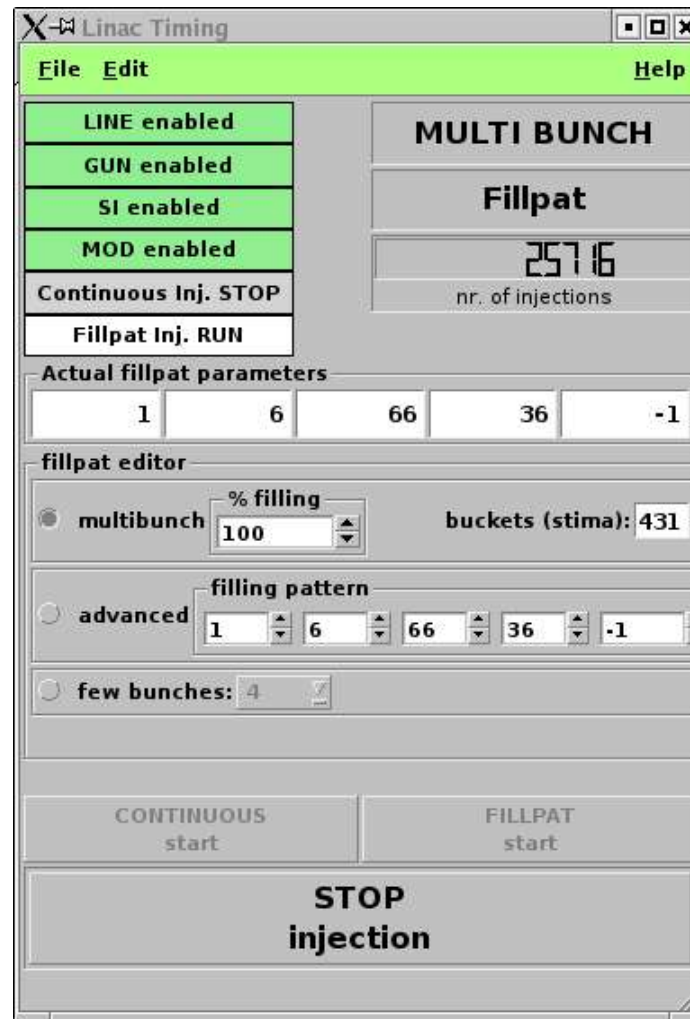
- compatibilità con Linux (i386,PPC) e HP-UX
- Disponibili in formato “sorgente”
- Prodotti moderni ma maturi
- Ampia e/o qualificata platea di utilizzatori
- Progetto curato e coerente
- Documentazione
- Possibilmente “open source”

G.U.I. library

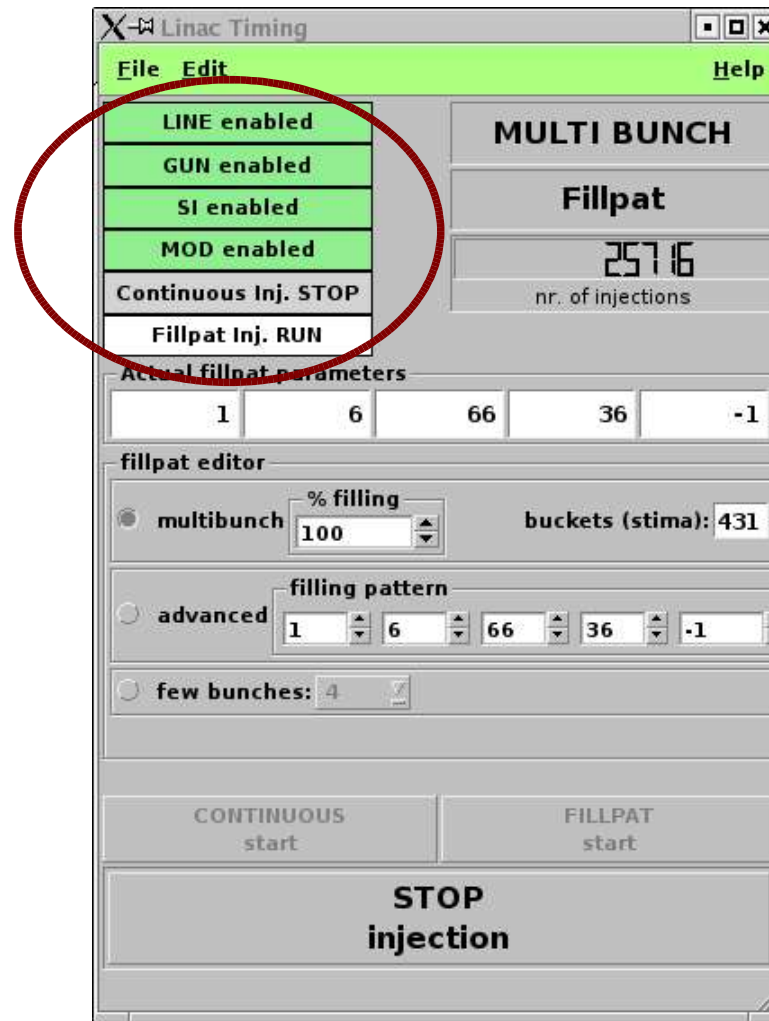
- “Qt” sviluppata da Trolltech AS *
 - Supporta anche Windows e Mac
 - Base del progetto KDE, uno dei migliori desktop manager per Linux
 - Supporto per database (Oracle,MySQL)
 - Ricca libreria di oggetti grafici (widget)
 - Facilmente estendibile ed integrabile
 - C++ (nativo)
 - python

* <http://www.trolltech.com>

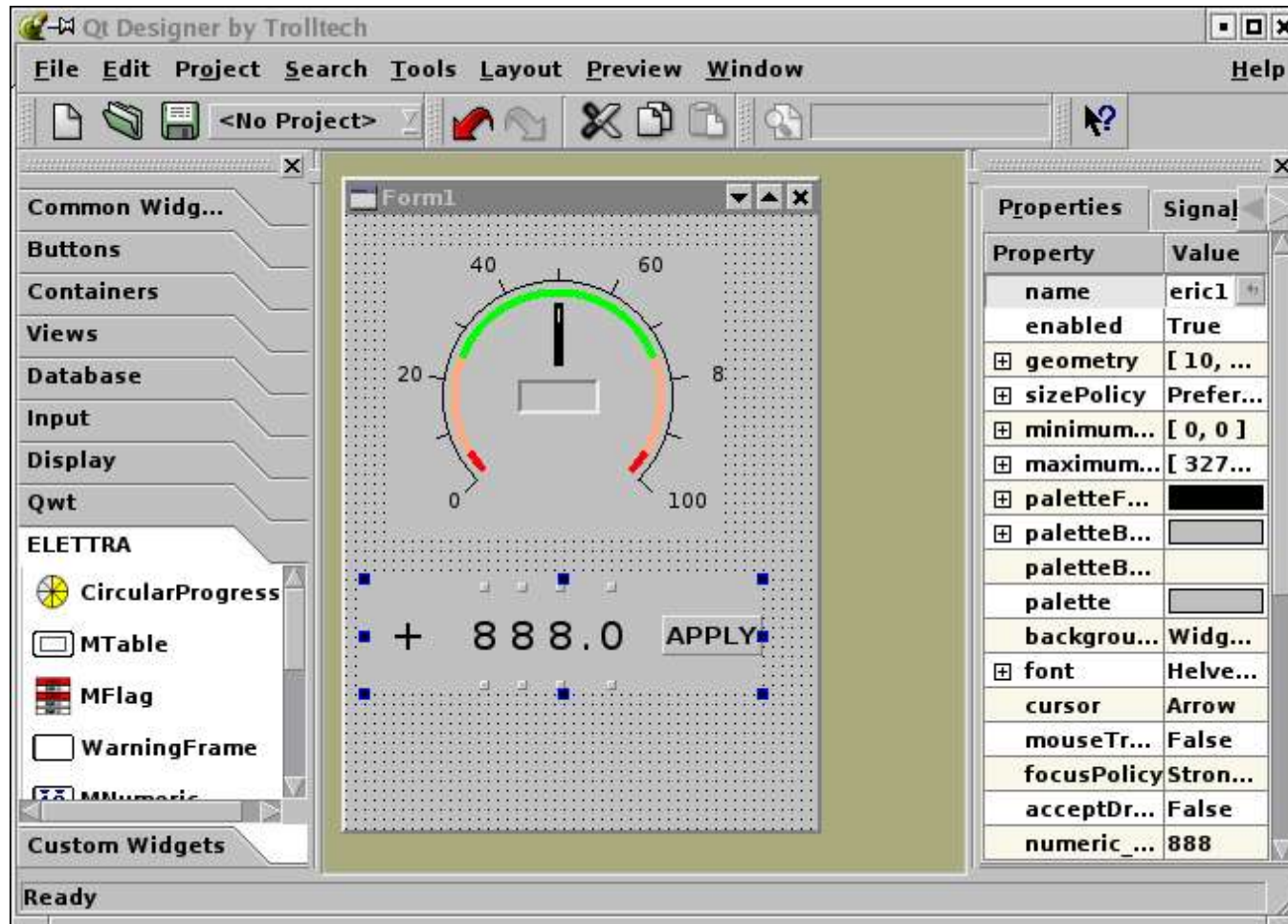
Qt: un assaggio (I)



Qt: un assaggio (I)



Qt: un assaggio (II)



Communications Library: CORBA (I)

- Common Object Request Broker Architecture
 - Standard definito dall' Open Management Group
 - Varie implementazioni: commerciali e open source
 - Binding per C++, java, python...
- Modello di programmazione a “oggetti distribuiti”
 - oggetto: dati + operazioni (metodi)
 - Dati modificabili solo per mezzo dei metodi
 - Server: implementa effettivamente l' oggetto
 - Client: agisce sull' oggetto per mezzo di un' interfaccia

Communications Library: CORBA (II)

- L' interfaccia dell' oggetto viene descritta per mezzo di un linguaggio specializzato: IDL
- A partire dall' IDL vengono generati dei sorgenti da utilizzare per server e client
 - invio/ricezione dei dati, codifica errori...
- server: implementa effettivamente l' oggetto
- client: utilizza l' oggetto per mezzo di un' interfaccia che in realta' trasferisce tutte le richieste al server

Communications Library: CORBA (III)

- CORBA definisce anche un certo numero di "servizi" standard, ad esempio:
 - Naming
 - Events
 - Notification
- I servizi CORBA sono "interoperabili" indipendentemente dalla particolare implementazione di CORBA e dal linguaggio di programmazione utilizzato per scrivere client e server.
- CORBA e' solo uno strumento per implementare sistemi distribuiti.

Middleware

Generic Interface:

modello “universale” per tutti gli oggetti da controllare: un' unica libreria (+)
flessibile (+)
run-time checking (-) --> prestazioni ridotte (-)
facile introdurre e integrare nuovi oggetti da controllare(+)

Specific Interface:

modello “specifico” per ciascuno degli oggetti da controllare: molte librerie (-)
rigido (-)
compile-time checking (+) --> prestazioni ottime (+)
conoscenza “a priori” di cosa si deve controllare: cicli di sviluppo piu' lenti (-)
richiede un processo di sviluppo strutturato e ben disciplinato

Dilemma non definitivamente risolvibile !

Middleware

Generic Interface:

modello “universale” per tutti gli oggetti da controllare: un' unica libreria (+)
 flessibile (+)
 run-time checking (-) --> prestazioni ridotte (-)
 facile introdurre e integrare nuovi oggetti da controllare(+)



Specific Interface:

modello “specifico” per ciascuno degli oggetti da controllare: molte librerie (-)
 rigido (-)
 compile-time checking (+) --> prestazioni ottime (+)
 conoscenza “a priori” di cosa si deve controllare: cicli di sviluppo piu' lenti (-)
 richiede un processo di sviluppo strutturato e ben disciplinato

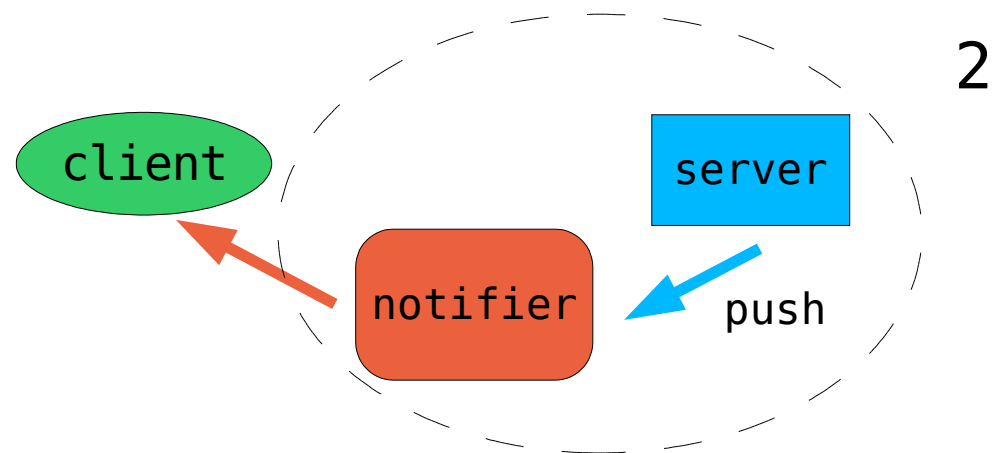
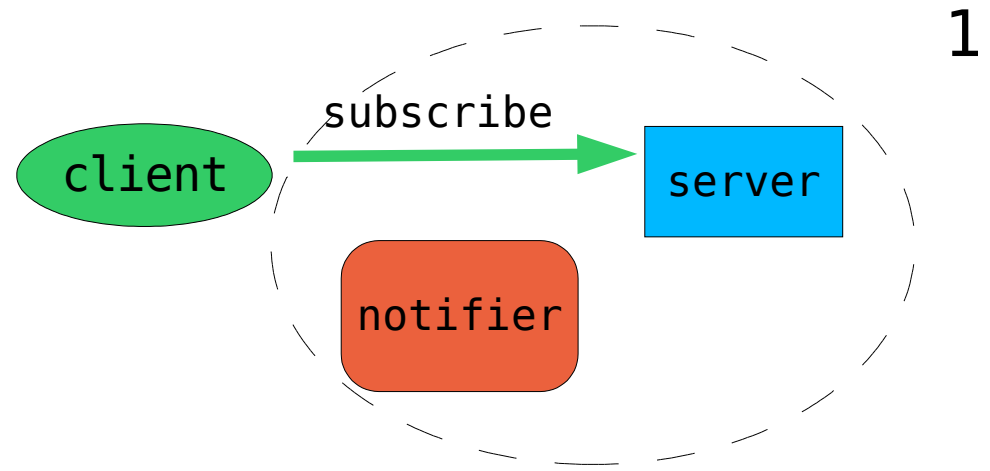
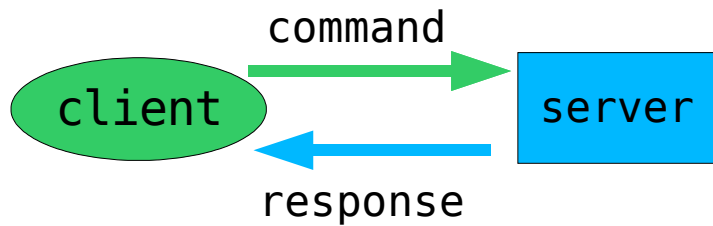
Dilemma non definitivamente risolvibile !

Requisiti del middleware

- Basato su CORBA, completamente O.O.
- Modellato sul concetto di “device”
 - Attributi : variabili scritte/lette
 - Comandi: operazioni con dati in ingresso e uscita
 - Proprietà: variabili di configurazione
- Capacità di introspezione
- Notifica di eventi con modello publish/subscribe
- Configurazione gestibile da database

→ *Sosh-98 model*

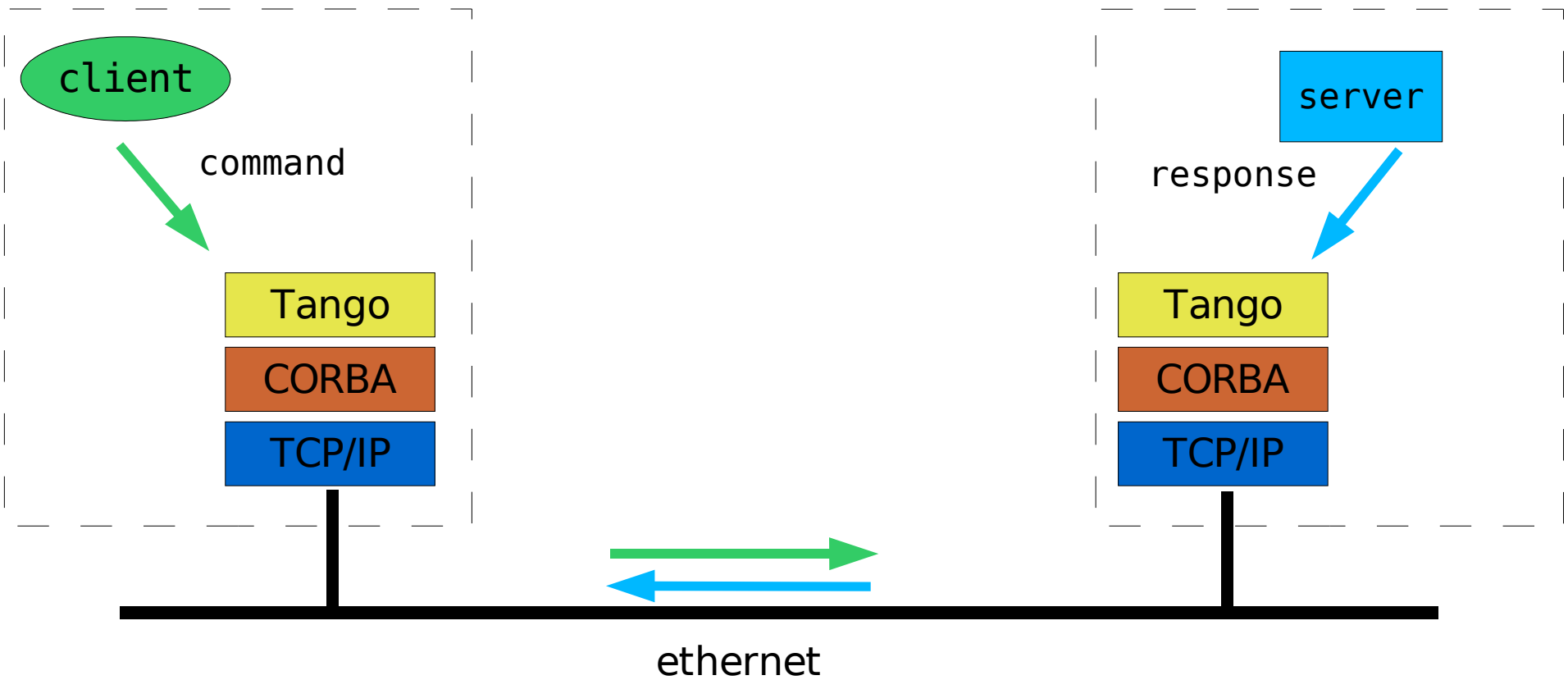
publish/subscribe



Tango Control System

- Soddisfa tutti i nostri requisiti
- Veloce, robusto
- Codice e documentazione di ottima qualità
- Buone prospettive di sviluppo e supporto futuro
- Adottato da ESRF, Soleil ed ELETTRA:
 - Collaboration Agreement firmato nel giugno del 2004
 - ELETTRA e' membro *paritario* della collaborazione

CORBA e Tango



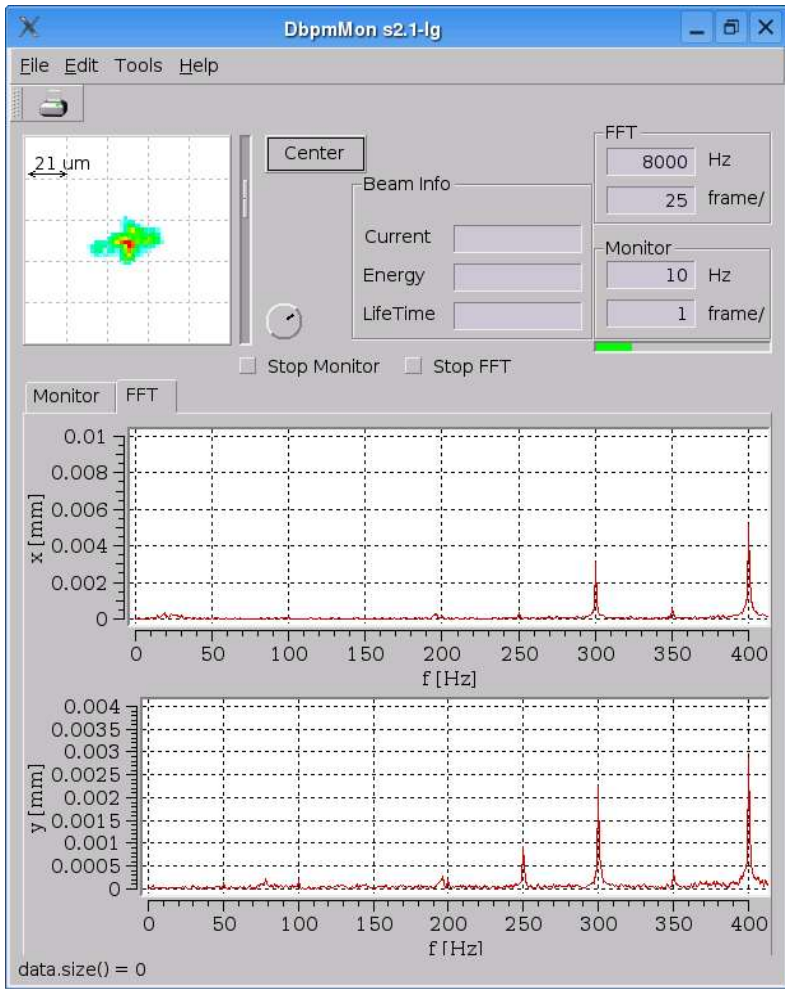
Tango e Qt ad ELETTRA

- Monitor **Digital BPM**
- Supervisore **Local Feedback**
- **Master Oscillator**
- **Phase Shifter**

Dall' inizio del run #92 alcuni device critici sono in configurazione “Booster Control System”:

- Motorola PPC/Linux
- Tango
- Qt

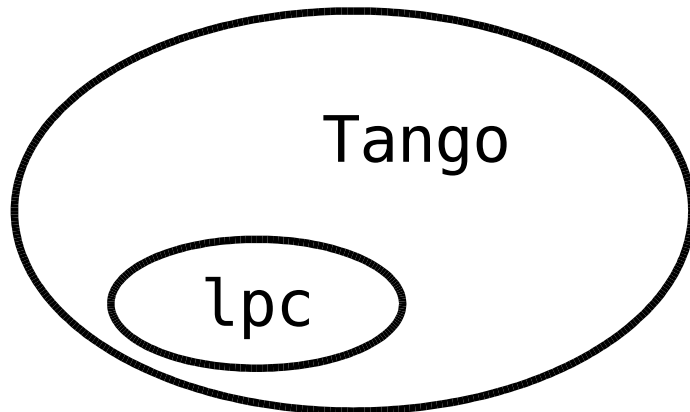
Tango e Qt ad ELETTRA



The screenshot shows the 'LOF sez. 2' window. It features a menu bar (File, Edit, Tools, Help) and two large buttons: 'Feedback ON' and 'Feedback OFF'. Below these is a 'Stop DC Loop' button and a gauge with a scale from 0 to 100. A 'Force DC Correction' button is also present. At the bottom, there is a vertical stack of eight green status indicators: 'HOR PID ON', 'VER PID ON', 'HOR HS ON', 'VER HS ON', 'RUNNING', 'CURRENT OK', 'BUTTONS OK', 'HOR NOT SATURATING', and 'VER NOT SATURATING'.

The screenshot shows the 'RF-MASTER_OSCILLATOR' window. It has a menu bar (File, Help) and a 'Main parameters' section. The 'frequency' is set to '499.653400 MHz' with 'Synchronous' and 'Remote (ON)' buttons. Below are 'FM off' and 'PHM off' buttons. A 'Standard' tab is selected, with other tabs for 'FM', 'PHM', 'Sweep', and 'Direct Setting'. The 'set frequency MHz' field shows '+ 4 9 9 . 6 5 3 4' with an 'APPLY' button. At the bottom, there are 'Synchronous' and 'Asynchronous' buttons, and a 'Remote Control' section with 'Stop FM', 'Stop PHM', and 'Sweep reset' buttons, and a 'Local Control' button.

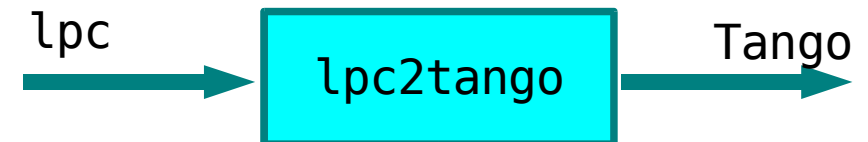
Transizione (e coesistenza)



La funzionalità' del "vecchio" sistema di controllo e' riproducibile con Tango

I nuovi "device" vanno progettati in modo opportuno

E' stato sviluppato un programma che "traduce" al volo le chiamate *lpc* nelle opportune chiamate *Tango*



I "vecchi" programmi accedono ai nuovi device senza alcuna modifica

Esperienze acquisite su Tango

- modello di *device*: adeguato e versatile
- copre ampiamente le necessita' dei nostri sistemi di controllo
- sviluppo di nuovi *device* rapido e sicuro:
 - design delle librerie
 - strumenti di test e prototipazione rapida
 - documentazione
 - libreria di *device* di base (serial-line, gpib...)
- fa molto di più e meglio del “vecchio” sistema
- competitivo rispetto ad altre soluzioni

Evoluzione o Rivoluzione?

Nuovi “concetti”: *programmazione 0.0. multithreading publish/subscribe*

Nuovi strumenti: *Qt Tango*

Nuovi linguaggi di programmazione: *C++ python java*

Verranno avviati dei corsi di formazione:

- programmazione O.O. e C++ (docente esterno)
- Tango control system, Qt (con risorse interne)

Crescita tecnologica del laboratorio
Crescita professionale del personale