Università degli Studi di Padova
Facoltà di Ingegneria

Finito di scrivere il giorno 7 aprile 2011 utilizzando LaTeX 2ε

UNIVERSITÀ DEGLI STUDI DI PADOVA
FACOLTÀ DI INGEGNERIA

—

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

—

TESI DI LAUREA IN INGEGNERIA INFORMATICA

PERSONALIZZAZIONE DEL PACCHETTO SOFTWARE "CONTROL SYSTEM STUDIO" PER IL PROGETTO SPES

RELATORE: CH.MO PROF. FRANCO BOMBI
CORRELATORE: DOTT. ALBERTO ANDRIGHETTO
CORRELATORE: PI. MAURO GIACCHINI

LAUREANDO:LORIS GIOVANNINI

ANNO ACCADEMICO 2010-2011

Alle persone che hanno creduto in me, che mi hanno sostenuto nei momenti difficili ed hanno saputo farmi reagire

"Se incontri qualcuno persuaso di sapere tutto e di essere capace di fare tutto non potrai sbagliare, costui è un imbecille. "

Confucio

Indice

Capitolo 1

Produzione ed utilizzo di fasci di ioni esotici

1.1 Nuclei stabili, nuclei instabili, e nuclei esotici

Le proprietà degli elementi presenti in natura sono determinate dall'unità di base, l'atomo; le proprietà chimiche sono dovute principalmente alla nube elettronica esterna, mentre le proprietà fisiche derivano dalla combinazione di più atomi disposti ed organizzati in reticoli cristallini. La quasi totalità della massa dell'atomo (più del 99,9%) si concentra nel nucleo, che è composto da particelle che la comunità scientifica definisce nucleoni: i protoni (a carica positiva) e i neutroni (elettricamente neutri); entrambi hanno una massa circa 1800 volte più grande degli elettroni [1]. Il comportamento e la stabilità del nucleo atomico e le interazioni protoni-neutroni sono il campo d'indagine della fisica nucleare. Il nucleo garantisce la propria stabilità grazie alla presenza di una forza attrattiva molto intensa tra i nucleoni, l'interazione forte, che manifestandosi efficacemente a distanze inferiori alle dimensioni nucleari, bilancia la repulsione elettrostatica tra i protoni carichi positivamente. L'interazione forte produce altri importanti effetti: essa vincola il moto dei nucleoni attorno alla massa nucleare centrale e, proprio per la sua intensità, rende la rottura del nucleo un processo che richiede molta energia. I nuclei stabili hanno approssimativamente lo stesso numero di neutroni

e protoni [2], e costituiscono la cosiddetta "valle di stabilità" nella carta dei nuclidi (Figura 1.1); i nuclei con numero di protoni uguale al numero di neutroni vantano una maggiore stabilità dovuta principalmente al fatto che l'interazione tra neutrone e protone è leggermente più intensa delle interazioni protone-protone e neutrone-neutrone. La Figura 1.1 mostra che per nuclei con numero di massa $A \geq 40$ (si ricorda che $A = Z + N$, con Z = *numero di protoni* ed N = *numero di neutroni*), la forza di Coulomb sposta la linea di stabilità lontano dalla retta in cui giacciono i nuclei con numero di protoni uguale al numero di neutroni verso nuclei ricchi di neutroni che, essendo privi di carica, non alimentano la forza repulsiva elettrostatica. Inoltre la forza coulombiana limita l'esistenza di elementi super pesanti, dato che il corto raggio di azione della forza nucleare forte non permette un'efficace opposizione alla forza elettrostatica, dato che essa invece agisce a più lungo raggio.

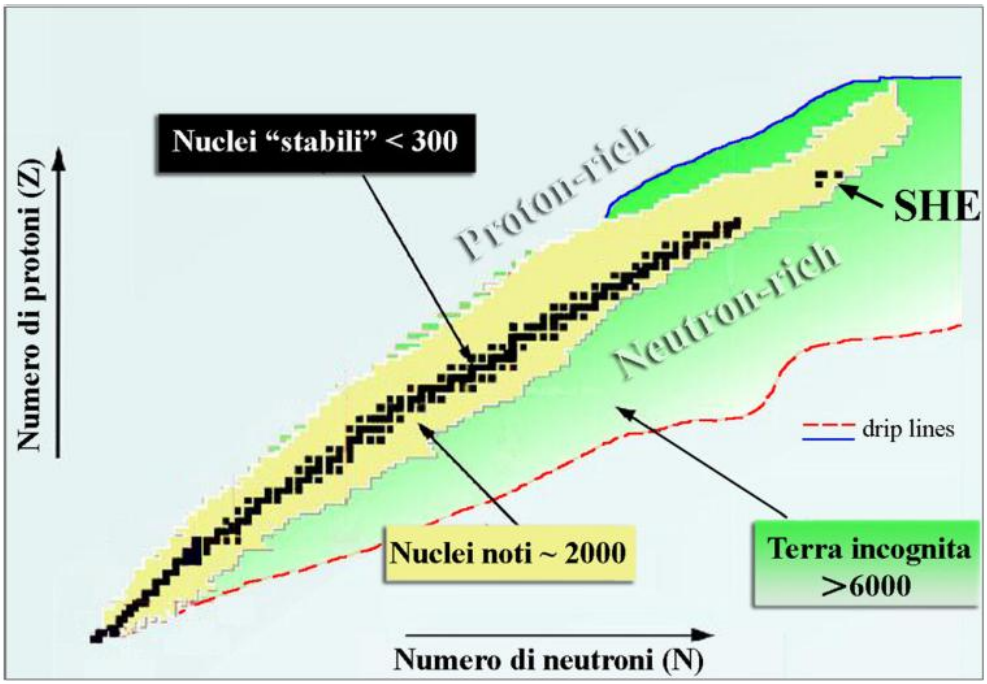

Figura 1.1: Carta dei nuclidi.

I nuclei che presentano eccesso o difetto di neutroni, e sono quindi lontani dalla valle di stabilità, sono radioattivi e soggetti a decadimento con emissione

di particelle (tra le quali le particelle alfa, le particelle beta, i neutrini) e raggi γ. Questi nuclei instabili si definiscono "esotici"; al momento ne sono stati prodotti e caratterizzati circa 2000 nei laboratori di ricerca, ma è possibile attraverso calcoli teorici intuire l'esistenza di un numero di nuclei esotici assai più elevato, addirittura oltre 6000; quindi è possibile che molti di essi siano presenti nella cosiddetta "terra incognita", la quale comprende inoltre la regione ricca di neutroni (*neutron-rich*) e quella dei nuclei *superpesanti* (SHE, *Super Heavy Elements*).

In base a ciò che si può osservare in figura, nella carta dei nuclidi si possono individuare e classificare i nuclei in base al numero di protoni (Z) e al numero di neutroni (N). I nuclei definiti "stabili" (quadratini neri) sono quelli non radioattivi oppure aventi tempo di decadimento comparabile o addirittura superiore all'età della terra. L'area indicata in giallo è caratterizzata dalla presenza di nuclei artificiali, che possono avere vita più o meno breve a seconda dei casi. Aggiungendo neutroni o protoni ad un nucleo ci si allontana dalla valle di stabilità fino a raggiungerne i limiti, detti *drip lines*. Ciò provoca una diminuzione della forza di attrazione tra neutroni e protoni, il che a sua volta si traduce in un incremento dell'instabilità del nucleo: grazie a calcoli teorici è stato dimostrato che, al di fuori delle *drip lines*, i nuclei emettono nucleoni molto rapidamente per formare nuovi nuclei, con combinazioni di protoni e neutroni tali da poter rientrare nell'area di potenziale stabilità, nella quale l'interazione forte è nuovamente capace di garantire per il nucleo il grado di coesione necessario.

La regione di colore verde è ancora inesplorata, ed è comunemente definita "terra incognita": essa comprende nuclei radioattivi con rapporti N/Z molto piccoli o molto grandi; si può vedere in figura che l'area *proton-rich* è relativamente ben definita (in linea teorica), mentre quella *neutron-rich* risulta più vasta ed indefinita.

Lo studio dei nuclei instabili, ed in particolar modo dei nuclei esotici, ha permesso l'apertura di nuovi campi di ricerca in fisica nucleare; grazie a ciò si sono potute confermare precedenti ipotesi di fondamentale importanza; inoltre ha favorito l'avvio di promettenti applicazioni in fisica dello stato solido ed in medicina.

1.2 Il metodo ISOL per la produzione di fasci radioattivi (RIB)

La produzione di ioni radioattivi di questo tipo e il loro conseguente utilizzo pratico dev'essere preceduta dalla costruzione di sistemi acceleratori ed attrezzature in grado di garantire fasci ionici (RIB, *Radioactive Ion Beams*) di elevata purezza, intensità ed energia (*facility*). Vi sono molteplici *facility* che operano per la produzione di fasci radioattivi sia in Europa che nel resto del mondo; la maggior parte di queste sono basate sulla tecnica ISOL (*Isotope Separation On-Line*).

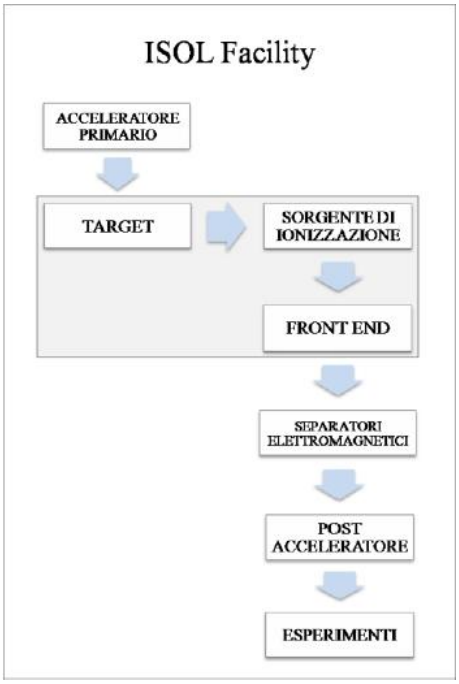

Figura 1.2: Schema di una facility di tipo ISOL per la produzione di fasci di ioni esotici.

Come si può vedere in Figura 1.2, il metodo ISOL utilizza la separazione degli isotopi in linea[3]. I principali costituenti di tale tipologia di *facility* sono:

- l'acceleratore primario;
- il complesso *target*, sistema di estrazione e sistema di ionizzazione;
- i separatori di massa ed isobari;

- il post acceleratore.

Dall'acceleratore primario viene emesso un fascio di particelle della voluta energia; il fascio viene poi fatto collidere con un bersaglio (*target*) di materia fissile; come risultato si ottiene così la produzione degli isotopi radioattivi attraverso delle reazioni nucleari (quali fissione, spallazione, frammentazione, ecc..). Tramite un apposito sistema vengono estratti e ionizzati i radioisotopi, che così potranno essere accelerati per differenza di potenziale. Il primo livello dell'accelerazione avviene nel *Front End*: esso attira gli ioni e li invia verso dei separatori elettromagnetici, all'interno dei quali il fascio viene poi opportunamente selezionato e purificato. Il vantaggio dei separatori elettromagnetici sta nel fatto che permettono di ottenere un fascio chimicamente e isobaricamente puro. Successivamente gli ioni vengono post accelerati al livello di energia richiesto dal particolare esperimento.

La seguente equazione descrive l'intensità del fascio radioattivo prodotto:

$$I = \sigma * \Phi * N * \epsilon_1 * \epsilon_2 * \epsilon_3 \tag{1.1}$$

dove σ è la sezione d'urto per le reazioni nucleari, Φ è l'intensità del fascio primario, N è lo spessore del *target*, ϵ_1 è l'efficienza di rilascio del *target*, ϵ_2 è l'efficienza di ionizzazione e ϵ_3 è l'efficienza del processo di estrazione.
Ne deriva che una corretta configurazione del complesso costituito dal target, dal sistema di estrazione e dal sistema di ionizzazione è d'importanza cruciale ai fini dell'efficiente funzionamento di una *facility* di tipo ISOL. Gli obiettivi che stanno alla base del dimensionamento sono:

1. la riduzione del tempo di ritardo;
2. la massimizzazione della produzione senza deterioramento della purezza del fascio.

I processi di separazione dei prodotti radioattivi dal substrato del *target* e di estrazione dei nuclei esotici dipendono strettamente dalle condizioni di temperatura: l'innalzamento del livello termico provoca un'accelerazione della diffusione

delle particelle. Ne consegue che il sistema deve essere mantenuto alla più alta temperatura possibile, dato che risulta ovvio che tanto più breve è la vita media degli atomi radioattivi, tanto più rapido deve essere il tempo di rilascio.

A livello europeo la comunità scientifica ha osservato favorevolmente le opportunità e i vantaggi offerti dai RIB, che però sono controbilanciati da problemi tecnologici. Per questo si è arrivati a proporre la costruzione di una rete di *facility* complementari, dette di "intermedia generazione". Si è trattato di un passo molto importante che ha permesso di arrivare all'ideazione di un'unica grande *facility* europea di tipo ISOL, chiamata EURISOL [4]. Questa iniziativa, che coinvolge i principali laboratori di fisica nucleare d'Europa, si esplica nello studio e nella progettazione di uno strumento tecnologico in grado di produrre fasci radioattivi di qualità significativamente superiore a quella attualmente raggiungibile. Questo scopo è perseguito anche nei Laboratori Nazionali di Legnaro (LNL): qui è in previsione la costruzione di una *facility* ISOL per la produzione di fasci di ioni esotici. Si tratta di ciò che viene definito "Progetto SPES" (*Selective Production of Exotic Species*), un programma coordinato a livello nazionale che prevede la collaborazione tra sei sezioni dell'INFN (Istituto Nazionale di Fisica Nucleare), l'ENEA (Bologna), i Dipartimenti di Ingegneria Meccanica, di Ingegneria dell'Informazione e di Scienze Chimiche dell'Università degli Studi di Padova e, a livello internazionale, strette collaborazioni con il CERN di Ginevra (Svizzera) ed i Laboratori di Oak Ridge (USA).

1.3 Campi di applicazione dei fasci di ioni esotici

Lo studio dei fasci di ioni esotici ha attirato un forte interesse da parte del mondo della ricerca, e questo grazie alle molteplici facoltà di utilizzo non solo nel campo della fisica nucleare ma anche nei settori dell'astrofisica, della medicina e della fisica dello stato solido.

1.3.1 Applicazioni nel settore della fisica nucleare

Miglioramento e verifica del Modello Standard

Il cosiddetto Modello Standard della fisica delle particelle è una teoria che descrive insieme tre delle quattro forze fondamentali, cioè l'interazione nucleare forte, l'elettromagnetismo e l'interazione nucleare debole (queste ultime due unificate nell'interazione elettrodebole), nonché la funzione e le proprietà di tutte le particelle (note ed osservate) che costituiscono la materia. Questo modello ha avuto molto successo tra gli addetti ai lavori, ma nonostante ciò non è del tutto soddisfacente poiché dipende in modo sostanziale da alcune assunzioni fatte *ad hoc*. Attraverso particolari esperimenti di fisica nucleare, suggeriti da convincenti basi teoriche, si è cercato di chiarire l'origine di queste assunzioni per arrivare così all'unificazione delle interazioni fondamentali. Tali esperimenti prevedono precise misure delle proprietà di decadimento di alcuni nuclei, le misurazioni delle quali si avvalgono dei fasci di ioni radioattivi prodotti dalle *facility* come sorgente pura di ioni[1].

Studio della struttura di nuclei complessi

E' dimostrato che i nucleoni (protoni e neutroni) sono a loro volta costituiti da subparticelle chiamate *quark*: esse esercitano un effetto fisico anche oltre i nucleoni nei quali sono confinate e, in particolare, le interazioni tra i nucleoni all'interno del nucleo sono diverse da quelle esistenti tra due nucleoni liberi, in quanto esse dipendono anche dalla densità di protoni e neutroni associata al particolare tipo di nucleo. Attualmente manca una formula generale in grado di quantificare l'entità

delle interazioni nucleari per tutti i nuclei rappresentati nella Figura 1.1, e questo perché i calcoli quantomeccanici sono applicabili unicamente ai nuclei più leggeri. Attraverso lo studio dei fasci di ioni radioattivi la fisica nucleare si pone l'obiettivo di riuscire a ottenere una trattazione unitaria che:

- permetta di derivare l'effettiva interazione tra le particelle nucleari;
- elimini le incongruenze dei modelli correnti;
- sia applicabile anche ai nuclei esotici (cioè aventi rapporto *protoni/neutroni* estremo).

Misura della dimensione del nucleo: i nuclei "Halo"

Quando ci si riferisce ai nuclei stabili, si afferma che la dimensione del nucleo è legata al numero totale di nucleoni che lo costituiscono: indicando con A il numero di nucleoni si ha che la semplice relazione

$$R = R_0 A^{1/3} \tag{1.2}$$

permette di determinare il raggio nucleare R in funzione di tale parametro, mentre R_0 è una costante pari a 1.2 $fermi$ [1].
Però quando ci si allontana dalla valle di stabilità si possono incontrare notevoli deviazioni da tale legge, in quanto le energie di legame tra le particelle di uno stesso nucleo possono diventare così piccole da causare la formazione di nuclei particolari, chiamati "ad anello" (nuclei *"halo"*). Questi sono caratterizzati da una diversa collocazione dei neutroni: possiedono infatti molti più neutroni dei rispettivi isotopi stabili; inoltre uno o due neutroni, essendo debolmente legati al nucleo, orbitano attorno ad esso (neutroni di valenza).
Un esempio di nucleo "halo" è il ^{11}Li: esso ha una dimensione media del nucleo paragonabile a quella del ^{48}Ca; se però si considera l'alone racchiuso dalle orbite dei due elettroni di valenza presenti, si avrà che il nucleo assume dimensioni paragonabili a quelle del ^{208}Pb (Figura 1.3).

[1] 1 fermi = 10^{-15}m

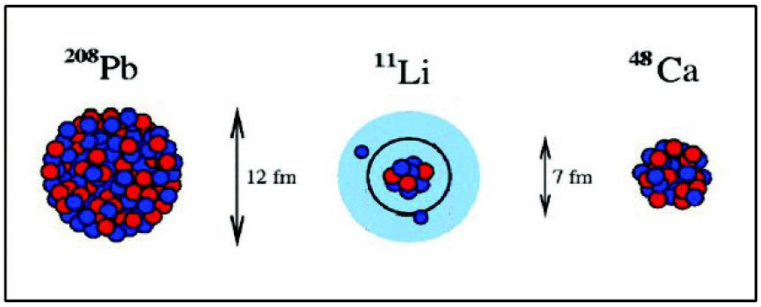

Figura 1.3: Paragone tra le dimensioni del nucleo di ^{11}Li e quelle di altri nuclei più massivi.

Quindi il nucleo ^{11}Li è un sistema a tre corpi (i due neutroni "esterni" ed il *core*) e rappresenta un esempio naturale di sistema *Borromeico* (Figura 1.4): in topologia i tre anelli borromeici sono legati l'uno all'altro in modo tale che la rottura di uno di essi permette la separazione degli altri due.

Figura 1.4: Gli anelli Borromeici.

Di conseguenza i nuclei ad anello sono chiamati anche "nuclei Borromeici" proprio perché se uno dei loro costituenti viene a mancare, gli altri divengono immediatamente instabili e a loro volta si possono allontanare facilmente.
Attualmente si usano fasci radioattivi di bassa energia e luce laser collineata per la misura della distribuzione dei protoni, sulla base di esperimenti di spettroscopia atomica; per la determinazione di tutti i nucleoni vengono invece impiegati fasci radioattivi ad alta energia.

Produzione di elementi superpesanti

Gli elementi chimici presenti in natura sono circa 90, dall'Idrogeno all'Uranio. Però gli esperimenti nei laboratori attraverso l'utilizzo di reazioni di fusione nucleare hanno condotto alla sintesi di nuovi elementi artificiali di elevato peso atomico, estendendo il numero dei costituenti della Tavola Periodica sino all'elemento avente numero atomico 112 e prospettando un'ulteriore estensione a 116. Questi elementi, detti superpesanti, in cui si ha una fortissima repulsione coulombiana, sembra riescano a formarsi attorno alla cosiddetta "Isola di Stabilità" (una combinazione di 114 protoni e 184 neutroni che sembra garantire la stabilità del nucleo).
La recente disponibilità di fasci intensi, costituiti da nuclei instabili ricchi di neutroni (*n-rich*) accoppiati a target stabili, anch'essi ricchi di neutroni, potrebbe finalmente permettere un'accurata indagine di tale fenomeno.

1.3.2 Applicazioni nei campi della fisica dello stato solido

Una delle principali applicazioni dei fasci di ioni esotici negli studi di fisica dello stato solido è l'implementazione della tecnica *Radio Tracer Diffusion*, nata nel 1920, che consiste nell'impiantare dei nuclei radioattivi all'interno di un sistema solido e nello studiarne il decadimento, rilevando le particelle o la radiazione gamma da esse emessa. Tale tecnica permette di captare segnali anche da pochissimi atomi ed è uno dei metodi più usati per studiare i processi di diffusione atomica nei solidi [5].
Il sistema ospitante può essere drogato con i radioisotopi "sonda" per diffusione, tramite reazione nucleare, oppure per impianto ionico. La scelta dell'atomo radioattivo da utilizzare per un determinato esperimento viene fatta in base alla sua natura chimica e alle proprietà nucleari.
L'uso della tecnica *Radio Tracer Diffusion* consente di:

- osservare, tramite i prodotti di decadimento, l'interazione tra l'atomo sonda e il reticolo che lo circonda;
- ottenere informazioni riguardanti il campo elettrico e magnetico all'interno del cristallo;

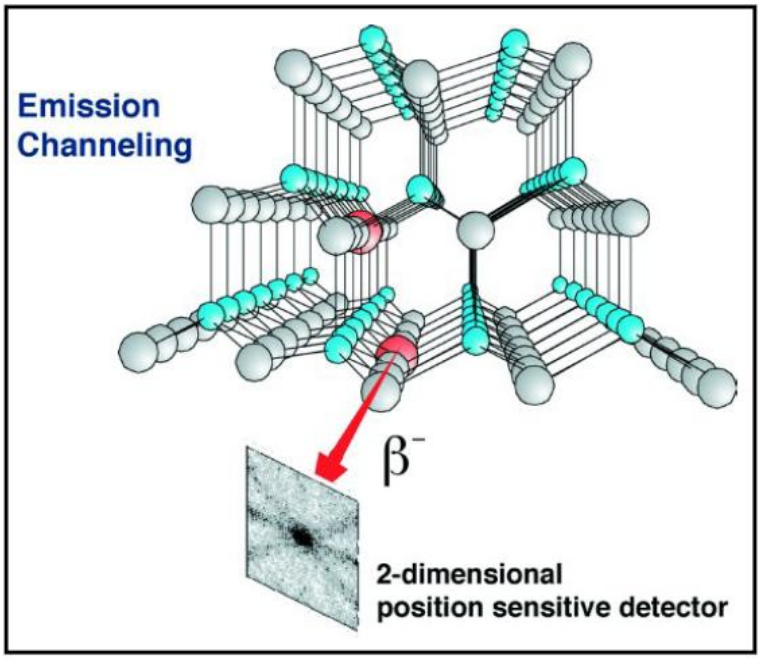

Figura 1.5: Emission Channeling degli elettroni emessi da atomi radioattivi situati in una riga atomica del reticolo.

- studiare i processi diffusivi e le interazioni tra gli atomi sonda;
- individuare i tipi di difetti presenti nel cristallo.

Drogaggio dei semiconduttori

Per arrivare allo sviluppo di semiconduttori di dimensioni ridotte con caratteristiche ottiche ed elettriche ottimali è necessario un controllo completo dei difetti che influiscono su tali proprietà, sia intrinseci (ad esempio le vacanze interstiziali) che estrinseci (come i droganti e le impurità atomiche): per tale motivo sia la ricerca di base che quella applicata si stanno concentrando notevolmente nello studio dei difetti e dell'attivazione elettrica dei droganti di diversi semiconduttori.

Come gli isotopi stabili, anche gli isotopi radioattivi influenzano le proprietà elettroniche ed ottiche dei semiconduttori a seconda della loro natura chimica e della loro posizione all'interno del reticolo cristallino: in particolare, poiché le proprietà elettroniche ed ottiche dei semiconduttori dipendono, oltre che dal tipo di semiconduttore, anche dalle sue dimensioni, è stato dimostrato che in semicon-

duttori molto piccoli tali proprietà possono essere sensibilmente alterate da un difetto presente con concentrazione minore di 10^{12} $atomi/cm^3$. Quindi per tenere sotto controllo le prestazioni dei semiconduttori (e garantirne l'affidabilità) c'è bisogno di tecniche sperimentali che combinino un'alta sensibilità chimica con un'alta sensibilità per la determinazione di basse concentrazioni di difetti.
In passato la principale tecnica in uso per la rilevazione delle impurità all'interno di un cristallo era il *channeling*: esso prevede che un fascio di ioni venga guidato lungo le righe atomiche o lungo i piani del cristallo (canali). Con questo processo tuttavia non è possibile determinare concentrazioni di difetti inferiori a 10^{18} $atomi/cm^3$, anche se la sensibilità di tale tecnica può essere incrementata di diversi ordini di grandezza impiantando all'interno del cristallo impurezze radioattive che emettono particelle cariche (*emission channeling*). La misura dell'emissione lungo differenti direzioni cristalline permette la determinazione del sito cristallografico dell'atomo emittente con un'accuratezza di pochi decimi di $\dot{A}$.

Studio delle proprietà magnetiche di superfici e monostrati atomici metallici

Un campo di ricerca di notevole interesse è costituito dalle superfici e dalle interfacce dei solidi. In particolare, è possibile studiare le proprietà magnetiche degli strati metallici ultrasottili e le variazioni che esse subiscono passando da uno strato atomico ad un altro. Per testare sperimentalmente predizioni teoriche molto dettagliate si utilizzano diversi atomi radioattivi "sonda". Un esempio di applicazione dei fasci di ioni radioattivi riguarda la determinazione delle proprietà magnetiche e strutturali di film atomici di Palladio su bulk[2] di Nichel. In questo ambito lo studio può essere condotto impiantando isotopi radioattivi di Piombo o Cadmio a differenti distanze dall'interfaccia.

In questo modo è stato possibile determinare, attraverso una risoluzione strutturale mai raggiunta prima, il comportamento ferromagnetico di uno strato ul-

[2]bulk: nell'ambito dei fenomeni di trasporto, indica la parte del fluido (o del solido) abbastanza lontana dalle regioni del fluido stesso in cui avvengono gli scambi di materia, quantità di moto e calore, da non percepirne gli effetti.

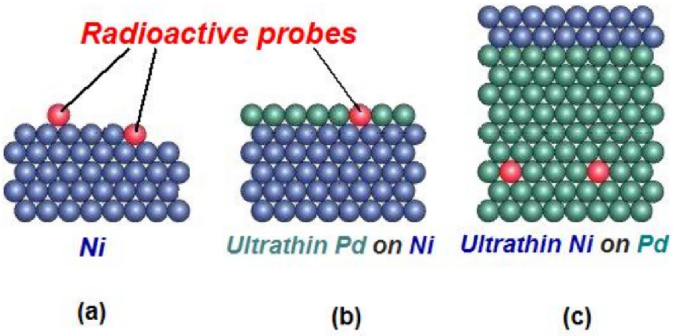

Figura 1.6: Esempio di analisi di proprietà magnetiche.

trasottile di Palladio cresciuto su Nichel, e le proprietà magnetiche indotte sul Palladio da uno strato ultrasottile di Nichel.

1.3.3 Applicazioni in ambiente medico: la Tomografia ad Emissione Positronica

Innanzitutto bisogna tener presente che per "antimateria" si intende la materia composta da antiparticelle, cioè particelle aventi la stessa massa e caratteristiche opposte a quelle che costituiscono la materia ordinaria; e per "positrone" (altrimenti detto antielettrone) ci si riferisce all'equivalente di antimateria dell'elettrone, con carica elettrica pari a +1. Quando un positrone si annichila con un elettrone, la massa di entrambi viene convertita in energia, sotto forma di due fotoni ad altissima energia nella banda dei raggi gamma. Un positrone può essere generato dal decadimento radioattivo con emissione di positroni o dall'interazione con la materia di fotoni con energia superiore a $1,022\ MeV$. L'antimateria è utilizzata principalmente per studiare le interazioni tra particelle elementari; ma i ricercatori ne hanno sviluppato anche un'applicazione tecnologica: la Tomografia ad Emissione Positronica (PET, *Positron Emission Tomography*), una tecnica di medicina nucleare che utilizza l'emissione di positroni per realizzare immagini tridimensionali o mappe ad alta risoluzione degli organi interni dei pazienti, con notevoli vantaggi nel campo della diagnostica medica, dal momento che si possono individuare molte delle principali forme tumorali allo stadio iniziale della

manifestazione neoplasica.

Figura 1.7: Scanner impiegato nella tecnica di rilevazione PET.

La procedura PET inizia con l'iniezione nel soggetto da esaminare, generalmente per via endovenosa, di un isotopo tracciante con tempo di dimezzamento molto breve, legato chimicamente ad una molecola attiva a livello metabolico. Dopo un tempo di attesa durante il quale la molecola metabolicamente attiva (spesso uno zucchero) raggiunge una determinata concentrazione all'interno dei tessuti organici da analizzare, il soggetto viene posizionato nello scanner.
L'isotopo di breve vita media decade, emettendo un positrone. Dopo un percorso che può raggiungere al massimo pochi millimentri, il positrone si annichila con un elettrone, producendo una coppia di fotoni (di energia paragonabile a quella dei raggi gamma) emessi in direzioni opposte tra loro (sfasate di 180° lungo la stessa retta). Tali fotoni vengono successivamente rilevati dal dispositivo di scansione, grazie anche all'impiego di speciali tubi fotomoltiplicatori. Nodo cruciale di questa tecnica è la rilevazione simultanea di coppie di fotoni: i fotoni che non raggiungono il rilevatore in coppia, cioè entro un intervallo di pochi nanosecondi, non sono presi in considerazione. Dalla misurazione della posizione in cui i fotoni colpiscono il rilevatore (ogni coppia di fotoni individua una retta) si può ricostru-

ire la posizione del corpo da cui sono stati emessi (teoricamente con due coppie di fotoni, e dunque con due rette, è possibile individuare il punto di emissione dei fotoni), permettendo la determinazione dell'attività o dell'utilizzo chimico della molecola iniettata all'interno delle parti del corpo investigate. Lo scanner utilizza la rilevazione delle coppie di fotoni per mappare la densità dell'isotopo nel corpo; la mappa risultante rappresenta i tessuti in cui la molecola campione si è maggiormente concentrata, e può così essere letta ed interpretata da uno specialista in medicina nucleare o in radiologia al fine di determinare una diagnosi ed il conseguente trattamento. Spesso, e sempre più frequentemente, le scansioni della Tomografia a Emissione di Positroni sono raffrontate con le scansioni a Risonanza Magnetica Nucleare, fornendo informazioni sia morfologiche che metaboliche, cioè sull'anatomia del tessuto o dell'organo di interesse e sulla loro conformazione in quel momento. La PET è usata estensivamente in oncologia clinica per avere rappresentazioni dei tumori e per la ricerca di metastasi e nelle ricerche cardiologiche e neurologiche.

Ad ogni modo, mentre gli altri metodi di scansione, come la TAC e la RMN, permettono di identificare alterazioni organiche e anatomiche nel corpo umano, le scansioni PET sono in grado di rilevare alterazioni a livello biomolecolare che spesso precedono l'alterazione anatomica, attraverso l'uso di marcatori molecolari che presentano un diverso ritmo di assorbimento a seconda del tessuto interessato.
Con la scansione PET è possibile visualizzare e quantificare con discreta precisione il cambio di afflusso sanguigno nelle varie strutture anatomiche (attraverso la misurazione della concentrazione dell'emettitore di positroni iniettato). I radionuclidi utilizzati nella scansione PET sono generalmente isotopi con breve tempo di dimezzamento, come ^{11}C ($\sim 20min$), ^{13}N ($\sim 10min$), ^{15}O ($\sim 2min$) e ^{18}F ($\sim 110min$). Per via del loro basso tempo di dimezzamento, i radioisotopi devono essere prodotti da un ciclotrone posizionato in prossimità dello scansionatore PET [5], però non tutti gli ospedali sono in grado di mantenere le strutture per la produzione di isotopi radioattivi.
La PET gioca un ruolo sempre maggiore nella verifica della risposta alla terapia,

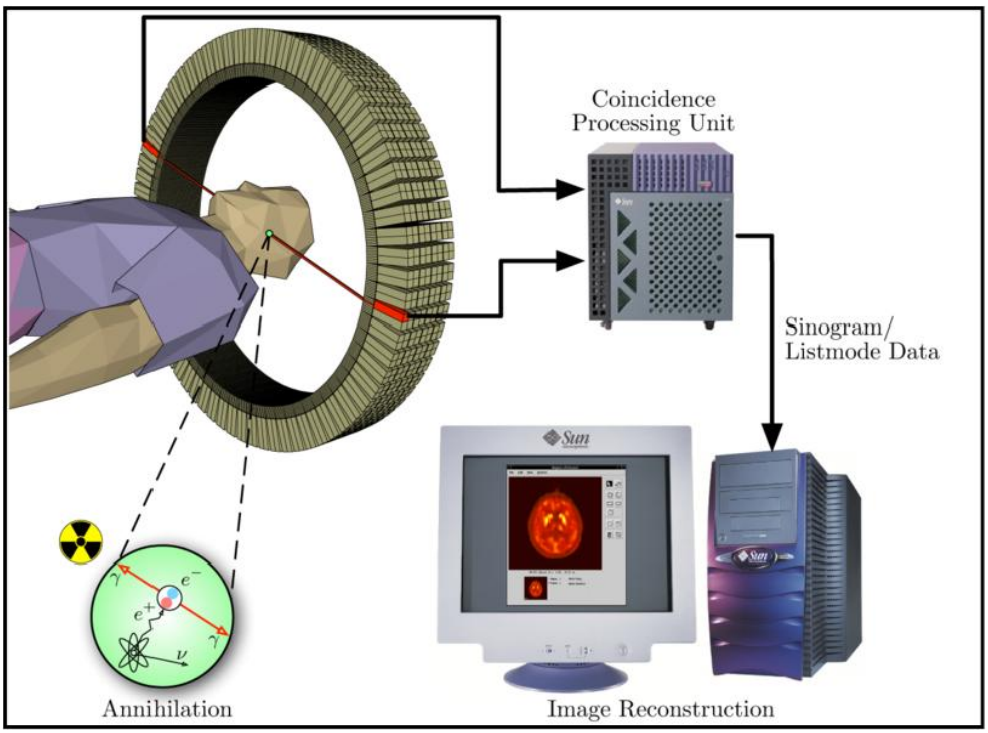

Figura 1.8: Schema riassuntivo della tecnica PET e modalità di acquisizione e presentazione dei dati.

specialmente in particolari rapie anti-cancro; si prospettano dunque per essa sempre maggiori applicazioni e sviluppi. Metodi simili, in cui si sfrutta la radioattività di isotopi di elementi biologici, sono usati in biochimica per evidenziare i movimenti di molecole attive metabolicamente, per lo studio di dettaglio di reazioni chimiche, per quantificare la diffusione delle sostanze all'interno delle cellule o per la fotografia del cariotipo umano durante le analisi del DNA.

1.3.4 Applicazioni nel settore dell'astrofisica

La disciplina che studia la comprensione della struttura, evoluzione e composizione dell'Universo e dei suoi costituenti è l'astrofisica nucleare. Le stelle generano energia attraverso reazioni nucleari che coinvolgono sia nuclei stabili che nuclei radioattivi. A volte il consumo del carburante nucleare è esplosivo e dura pochi minuti o secondi, altre volte procede stabilmente e dura bilioni di anni. Nelle differenti fasi di vita delle stelle vengono sintetizzati nuovi elementi chimici

sia attraverso processi di nucleosintesi che seguono strettamente la valle di stabilità, sia attraverso processi che si svolgono in un territorio sconosciuto. Al fine di sviluppare un modello di descrizione del meccanismo di nucleosintesi, si devono misurare le rese delle reazioni nucleari relative ai principali cicli astrofisici e le caratteristiche di decadimento di molti nuclei tuttora sconosciuti. Queste essenziali informazioni includono i tempi di vita, le masse ed i principali canali di decadimento di un numero di nuclei chiave lontani dalla stabilità.
Le reazioni nucleari che riguardano nuclei instabili possono essere misurate unicamente con un fascio di ioni radioattivi: per questo motivo si prevede che la nuova generazione di *facility* per la produzione di fasci radioattivi risulterà di fondamentale importanza per la comprensione della sintesi elementare nell'Universo.[1]

Capitolo 2

Produzione di fasci di ioni esotici e Front End del Progetto SPES

Il progetto SPES (Selective Production of Exotic Species) dei Laboratori Nazionali di Legnaro ha come fine principale quello di arrivare a fornire fasci di ioni ricchi di neutroni che permettano di svolgere ricerca all'avanguardia nei campi della fisica nucleare. I fasci esotici *neutron-rich* sono prodotti attraverso la fissione nucleare indotta bombardando il *target* di produzione, realizzato in carburo di uranio, con un fascio di protoni di 40 MeV e 200 μA di corrente (per una potenza totale di 8kW)[6].

In questo capitolo verrà esposta la struttura di fondo della *facility* SPES ed i principali elementi che la compongono. Si passerà poi alla descrizione del Front End *offline* che attualmente viene utilizzato nei LNL per le attività di ricerca e sviluppo. Infine verrà fatta una panoramica sugli strumenti software adottati per interfacciarsi con il sistema di controllo EPICS.

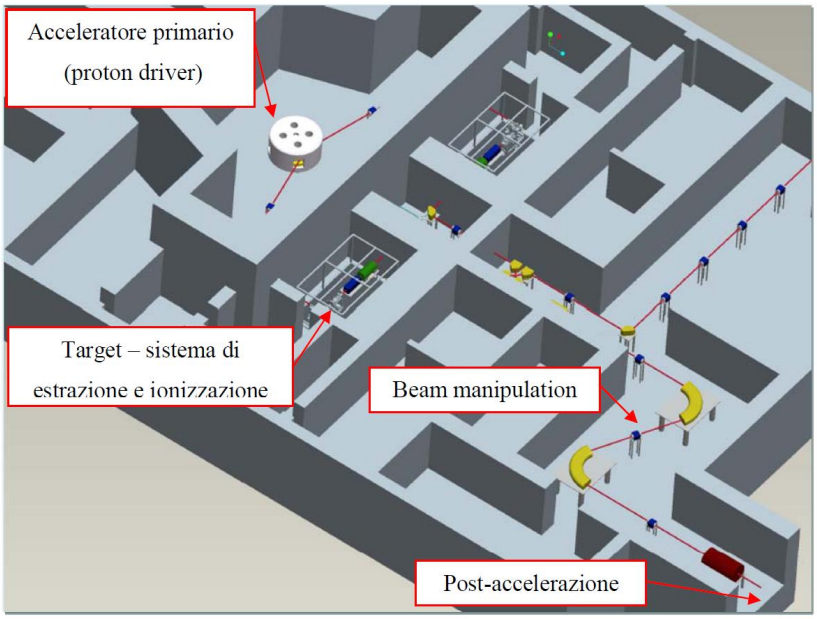

Figura 2.1: Schema della struttura della *facility* SPES (RIB *facility*).

2.1 La facility SPES di Legnaro

L'acceleratore SPES, in fase di realizzazione ai Laboratori Nazionali di Legnaro, è una *facility* di tipo ISOL (*Isotope Separation On Line*), ed è possibile identificare le principali strutture che la costituiscono, di cui poi verrà fornita una descrizione più dettagliata (Figura 2.1):

- l'acceleratore primario (che fornisce il fascio di protoni necessario per la produzione degli isotopi attraverso la collisione con il target);
- il complesso costituito dal target e dal sistema di ionizzazione;
- i separatori di massa ed i separatori isobari (necessari per la manipolazione del fascio);
- il sistema di post accelerazione.

2.1.1 L'acceleratore primario

L'acceleratore primario ha la funzione di produrre il fascio di particelle che deve essere direzionato verso il *target* di produzione, sede della reazione nucleare[6]. La

facility SPES utilizzerà come acceleratore primario un Ciclotrone, atto a fornire un fascio di protoni. Al giorno d'oggi sono reperibili in commercio ciclotroni che forniscono fasci di intensità ed energia molto vicini a quanto richiesto dal progetto SPES, e per questo motivo ci si è orientati verso la scelta di una soluzione commerciale.
Un esempio di ciclotrone commerciale adatto al progetto SPES è il Cyclone© 70, sviluppato dalla IBA, che si può osservare in Figura 2.2: esso è in grado di fornire due fasci di protoni indipendenti fino a 70MeV di energia ed aventi una corrente massima di 750 μA.

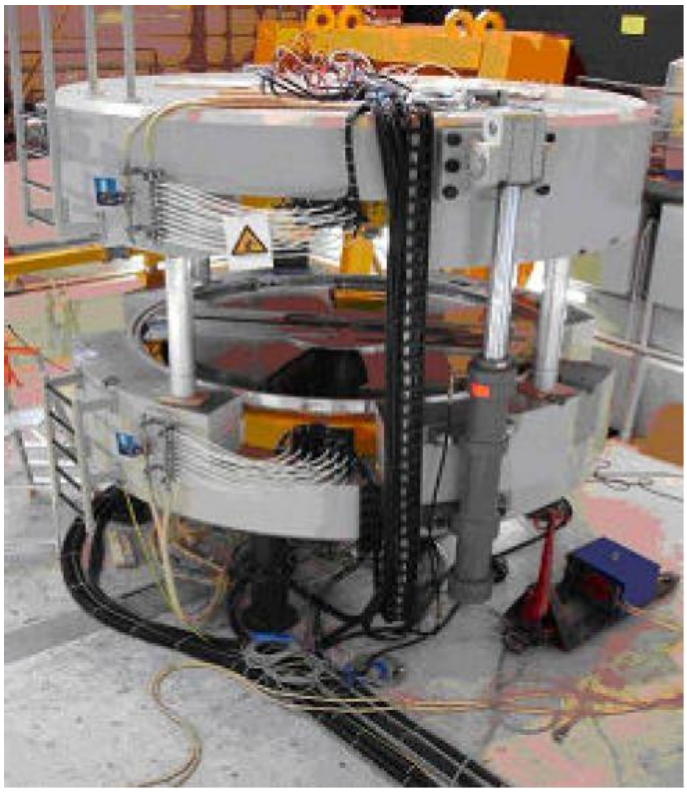

Figura 2.2: Il ciclotrone *Cyclone 70*.

2.1.2 Il target di produzione ed il sistema di estrazione

Sia il *target* di produzione dei radioscopi, sia il sistema di estrazione e ionizzazione degli stessi sono contenuti all'interno di una camera di forma cilindrica (camera *target*), la quale viene raffreddata mediante un opportuno circuito, viste le elevate temperature in gioco (Figura 2.3). Inoltre a causa delle alte temperature, per evitare l'ossidazione dei componenti presenti, l'interno della camera viene mantenuto in condizioni di alto vuoto (con pressione dell'ordine dei 10^{-6} $mbar$); la mancanza di atmosfera è necessaria anche per aumentare il cammino libero medio delle particelle radioattive prodotte. Il volume della camera è delineato dallo spallamento di una flangia (*plate*) e da un coperchio (*cover*) a forma di pentola, entrambi realizzati in lega di alluminio, mentre la tenuta a vuoto viene garantita da una guarnizione.

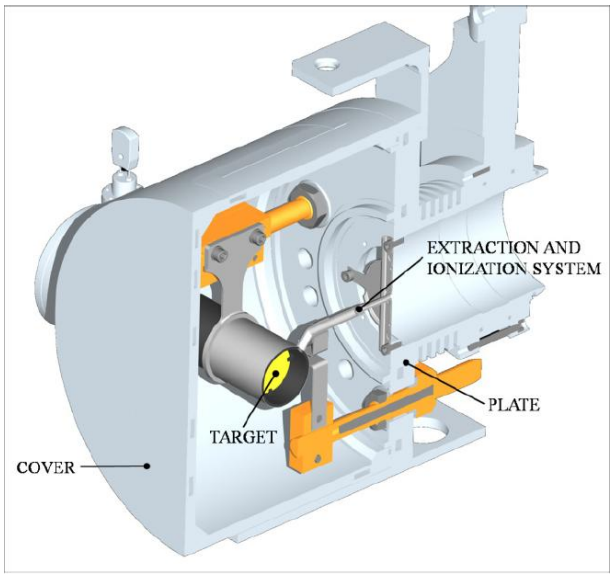

Figura 2.3: Configurazione della camera target.

Il *target* di produzione, come si può vedere in Figura 2.4, è composto principalmente da sette dischi coassiali in UC_x opportunamente distanziati al fine di dissipare, attraverso radiazione termica, la potenza di $8kW$ sviluppata dal fascio di protoni; i dischi sono contenuti in una scatola (*box*) costituita da un tubo cavo di grafite. Il fascio di protoni attraversa due finestre di grafite, interagisce con i

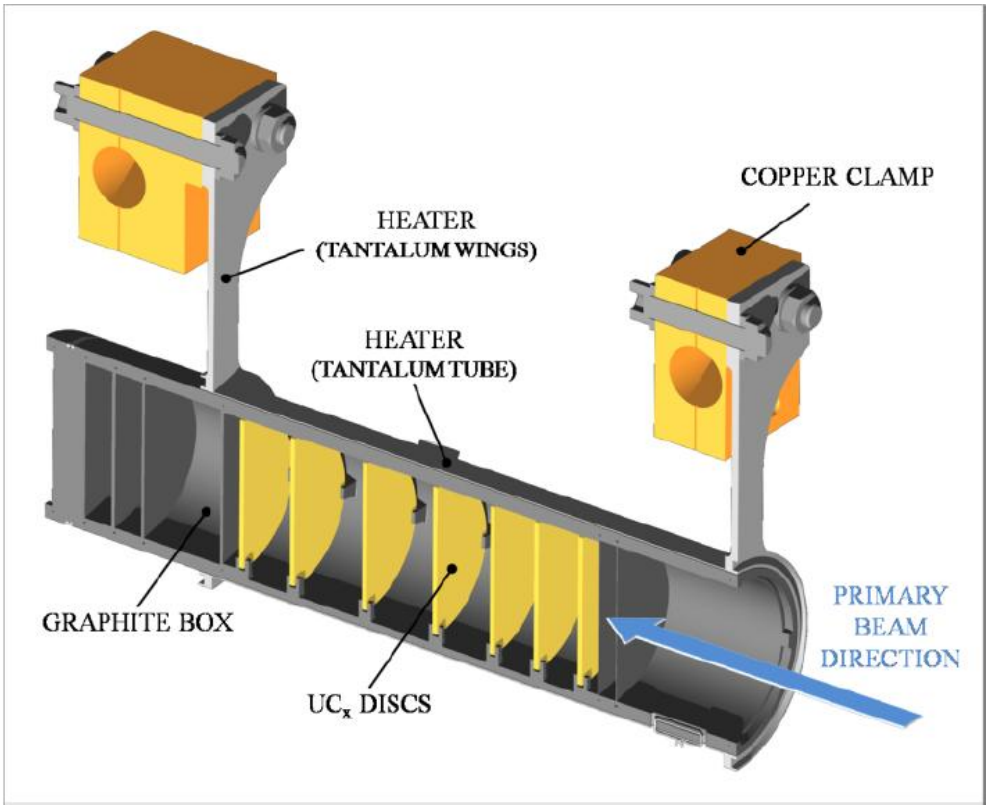

Figura 2.4: Rappresentazione del prototipo di bersagli diretto del progetto SPES.

bersagli di produzione in UC_x ed infine viene assorbito da un sistema di *dumper* in grafite[7].

Nella scatola e nei dischi contenuti in essa si deve mantenere la temperatura di esercizio di circa $2000°C$, in modo da ottimizzare l'estrazione dei prodotti di fissione. Poiché la potenza del fascio di protoni non è sufficiente a portare il *target* al livello di temperatura richiesto, bisogna disporre di un dispositivo indipendente che svolga la funzione di riscaldare e schermare il *target*. Il sistema di riscaldamento supplementare deve essere in grado di sostenere il target durante i transitori, evitando quegli improvvisi pericolosi sbalzi di temperatura che metterebbero a repentaglio l'integrità strutturale dei dischi. Il riscaldatore (*heater*), come si può vedere in Figura 2.4, è composto da un tubo molto sottile (*tube*) di tantalio saldato ai bordi delle due ali (*wings*) direttamente collegate ai morsetti in rame (*clamps*), tramite i quali per effetto Joule viene dissipato dal riscaldatore il quantitativo richiesto di potenza. Questo, in aggiunta al calore sviluppato dal-

l'interazione del fascio protonico con il target, fa sì che la temperatura del sistema scatola-dischi sia mantenuta al valore di utilizzo. Per la struttura del riscaldatore è stato volutamente scelto il tantalio perché esso è un metallo altamente resistente alla corrosione, in grado di condurre energia elettrica e termica e di raggiungere temperature molto elevate.
Il processo di fissione nucleare provoca la creazione di nuclei con una massa compresa tra gli 80 ed i 160 *uma*. Per la produzione di un RIB (Radioactive Ion Beam), gli isotopi devono quindi essere estratti e ionizzati.

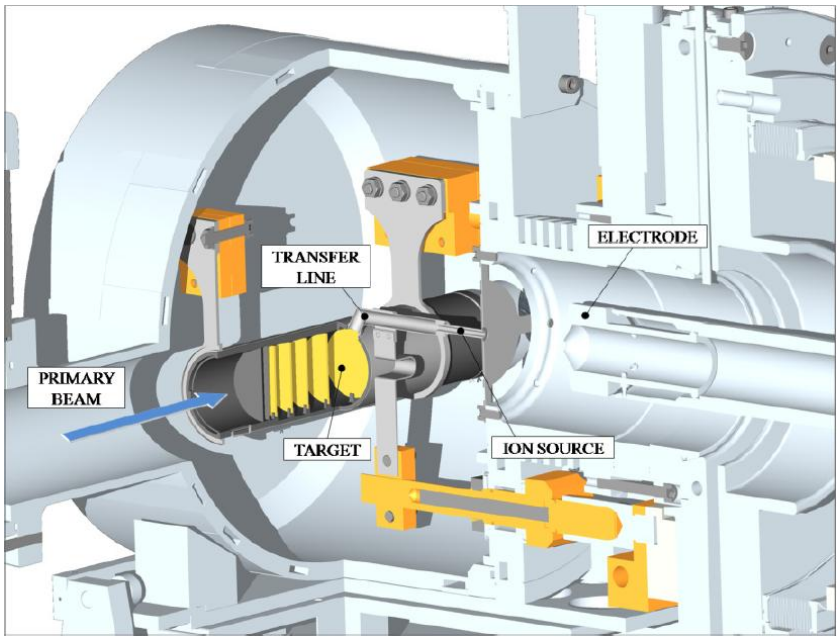

Figura 2.5: Rappresentazione del sistema di estrazione e ionizzazione del progetto SPES.

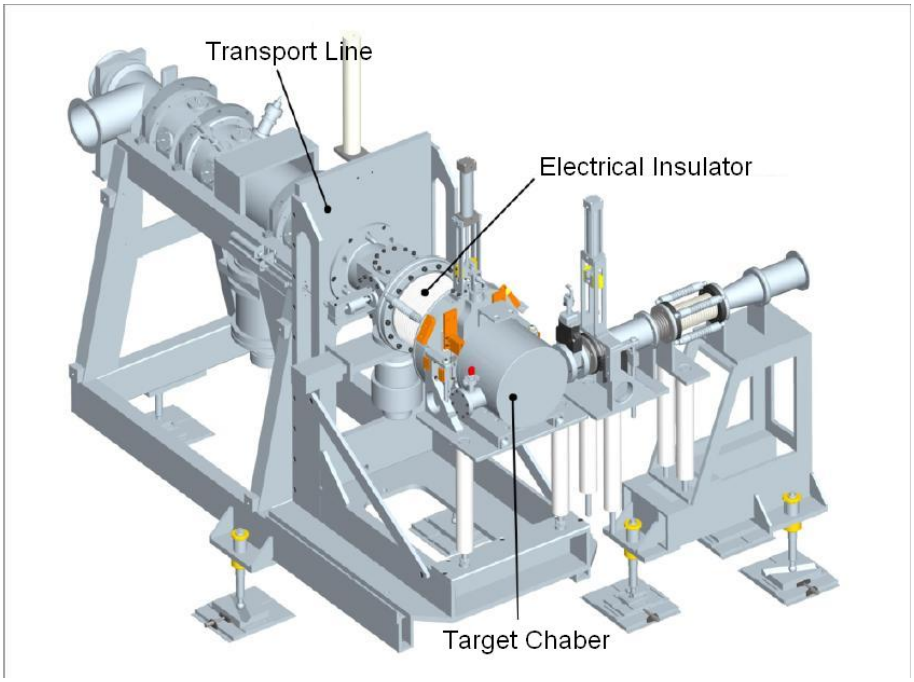

Figura 2.6: Rappresentazione del Front End.

La linea di trasferimento che collega il target con la sorgente di ionizzazione è interessata dal processo di estrazione degli isotopi. Nell'attuale configurazione la linea di trasferimento, che si può vedere in Figura 2.5, è costituita da un tubo sottile di tantalio saldato al riscaldatore ad un'estremità e connesso meccanicamente alla sorgente di ionizzazione. Come accade per il riscaldatore, anche il sistema linea di trasferimento - sorgente di ionizzazione viene riscaldato mediante dissipazione di potenza per effetto Joule; in questo modo la temperatura della sorgente arriva a sfiorare i $2400°C$.

Tra la camera target ed il sistema di trasporto del fascio di protoni è presente una differenza di potenziale ($V_{camera} - V_{transport}$) pari a 60 kV: quindi, per evitare il contatto diretto, bisogna interporre un isolante elettrico (*electrical insulator*) come rappresentato in Figura 2.6. La differenza di potenziale presente permette quindi l'accelerazione degli ioni, formando in questo modo il fascio che verrà inviato alle sale sperimentali.

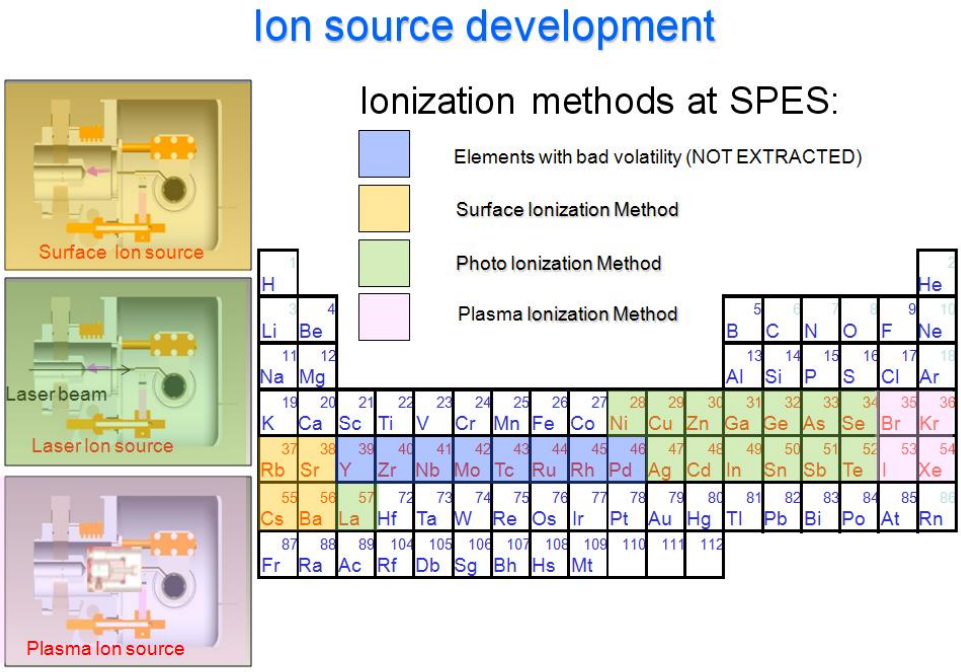

Figura 2.7: Schema delle differenti metodologie di ionizzazione in fase di sviluppo per il progetto SPES: è possibile individuare per ogni tecnica quali elementi della tavola periodica è possibile ionizzare.

2.1.3 Il processo di ionizzazione

La camera target descritta in precedenza trova alloggio all'interno del Front End. Gli isotopi prodotti nel target, per divenire fasci di nuclei accelerati, devono essere necessariamente ionizzati. Nel Front End si possono produrre isotopi appartenenti soltanto ad un sottoinsieme degli elementi presenti in natura: possono essere impiegati, come visibile in Figura 2.7, unicamente gli elementi compresi tra numero atomico $Z = 28$ (Nichel) e $Z = 57$ (Lantanio). Inoltre questo insieme si sfoltisce ulteriormente perché alcuni elementi non possono essere estratti a causa della loro scarsa volatilità (gli elementi compresi tra Y e Pd); ma nonostante questo il range di specie utilizzabili rimane sufficientemente ampio.
Il processo di ionizzazione è regolato dalle caratteristiche chimiche dell'elemento utilizzato: in generale l'energia di ionizzazione decresce lungo un gruppo della tavola periodica ed incrementa da sinistra a destra attraverso il periodo. E' quindi

possibile classificare gli elementi prodotti nel target, indicati in rosso nella tavola periodica di Figura 2.7, in quattro gruppi distinti:

- elementi che non sono estratti dal target a causa della loro scarsa volatilità;
- elementi con bassa energia di prima ionizzazione ($\leq 5eV$);
- elementi con energia di prima ionizzazione compresa tra $5eV$ e $10eV$;
- elementi con alto valore di energia di prima ionizzazione ($\leq 15eV$).

Ciò implica che vi siano quindi tre differenti metodi di ionizzazione e, per ognuno di essi rispettivamente, una particolare sorgente di ionizzazione per la produzione del fascio desiderato. Vengono ora prese in esame le tre diverse metodologie e le rispettive sorgenti di ionizzazione utilizzate all'interno del progetto SPES.

Metodo di ionizzazione superficiale

Nel complesso Front End, attraverso la *Surface Ion Source*, la ionizzazione avviene a partire dall'urto degli isotopi provenienti dal target sulla superficie interna di un tubo di tungsteno (*hot cavity*). In conseguenza di tali urti gli isotopi cedono un elettrone e vengono quindi caricati positivamente (ioni +1). Questo è possibile se la minima energia necessaria per rimuovere un elettrone da una superficie (funzione di lavoro) è maggiore del potenziale di ionizzazione dell'isotopo. Nel caso in esame si riescono a produrre con elevata efficienza ioni positivi per elementi con potenziale di ionizzazione inferiore alla funzione di lavoro del tugsteno (pari a circa $\leq 5eV$). Per mezzo della ionizzazione superficiale è quindi possibile produrre un fascio di ioni utilizzando gli elementi caratterizzati da una bassa energia di ionizzazione, come gli elementi alcalini ed alcalino-terrosi indicati in Figura 2.7.
Lo svantaggio principale di tale metodo sta nel fatto che la ionizzazione non è selettiva, cioè non vengono ionizzati soltanto gli isotopi della specie desiderata; bisogna perciò predisporre dei separatori elettromagnetici in grado di selezionare, in base alla massa, le particelle presenti nel fascio.
In ogni caso la purezza del fascio non è garantita: vi sono infatti isotopi di diversa specie tra loro isobari, aventi cioè lo stesso numero di massa A ma un diverso

numero atomico Z (come ad esempio ^{132}Cs e lo ^{132}Sn). Per separare tali elementi sono necessari dei separatori isobari; tali dispositivi, oltre ad essere molto complicati e costosi, sono anche poco affidabili e riducono notevolmente l'intensità del fascio.
Questa tipologia di sorgente permette di ottenere anche ioni negativi. La produzione di fasci di atomi esotici ionizzati negativamente è comunque molto complicata, non tanto per la produzione degli ioni (per cui è sufficiente realizzare la *hot cavity* con un materiale la cui funzione di lavoro sia più piccola dell'affinità elettronica degli isotopi) bensì per la loro accelerazione. La differenza di potenziale presente tra la camera target, la linea di trasporto del fascio ed i sistemi di post accelerazione deve essere invertita, e questo è possibile soltanto se si dispone di particolari dispositivi denominati *charge exchange devices*.

Metodo di fotoionizzazione

Ad oggi il più potente strumento per la produzione di fasci di ioni per le *facility* di tipo ISOL è la fotoionizzazione, di cui si può vedere il principio di funzionamento in Figura 2.8: questo metodo assicura un processo di ionizzazione selettiva e garantisce la soppressione di contaminazioni indesiderate a livello della sorgente di ionizzazione. Come sorgente di ionizzazione viene impiegato un fascio laser, ottenuto come sovrapposizione di più fasci laser aventi diverse lunghezza d'onda. Tale fascio viene diretto all'interno della *hot cavity* in modo da interagire con gli isotopi e fornire loro l'energia necessaria ad attivare una ionizzazione "selettiva". All'elettrone più esterno vengono fatti compiere dei salti quantici verso orbitali più esterni fino alla separazione dall'atomo: in questo modo si ha la formazione di uno ione positivo +1.
La metodologia di fotoionizzazione permette quindi di produrre soltanto ioni della specie di interesse, riuscendo ad ottenere un fascio nel quale le contaminazioni sono minime.
Per avere un'efficienza di ionizzazione elevata è di fondamentale importanza limitare e controllare il disallineamento della *hot cavity* causato dall'espansione termica: se la *hot cavity* si disallinea viene a ridursi la zona di azione del laser e

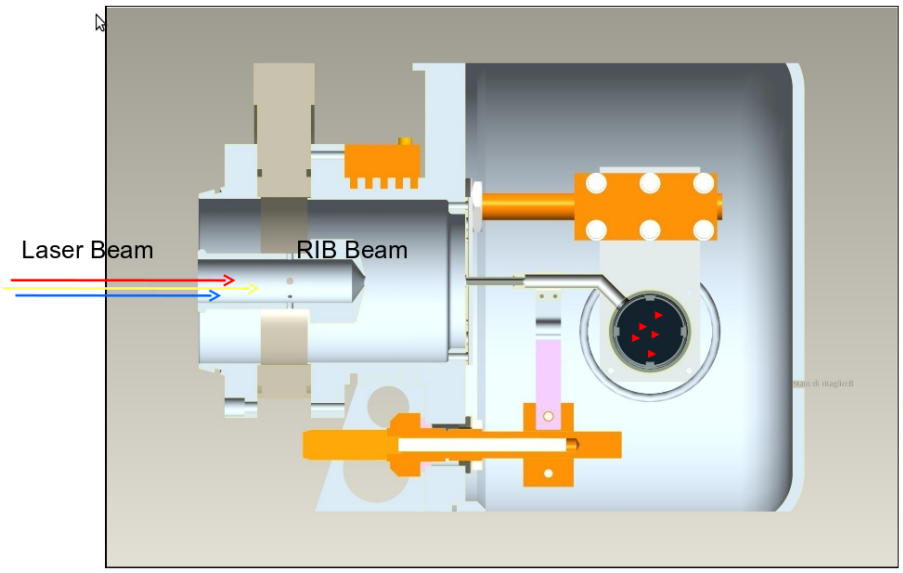

Figura 2.8: Schematizzazione del sistema di ionizzazione RILIS.

di conseguenza si riduce anche l'efficienza di ionizzazione.

Metodo di ionizzazione al plasma

La *Plasma Ion Source* è una tecnica che permette di ionizzare anche gli elementi riportati nelle ultime due colonne della tavola periodica, caratterizzati da un alto potenziale di ionizzazione (Figura 2.7). Il principale problema di questa metodologia è lo scarso livello di selettività: in effetti l'energia media fornita è tale da ionizzare tutti gli elementi prodotti nel target, con la conseguenza che il fascio di ioni prodotto risulta troppo impuro per poter essere utilizzato. In conclusione il metodo è meno efficiente rispetto alle tecniche laser e superficiale.
Attualmente ai Laboratori Nazionali di Legnaro è in fase di studio e progettazione una nuova sorgente di ionizzazione che permetta di utilizzare questa metodologia.

2.1.4 Purificazione nei separatori elettromagnetici

Una volta estratto dalla sorgente di ionizzazione, il fascio viene sottoposto ad un primo momento di purificazione in cui grazie al filtro di *Wien* viene catturata una grande quantità di contaminanti. La risoluzione di tale separatore di massa è tale da permettere di separare in base alla massa i vari elementi.
Dopo di che viene utilizzato uno spettrometro ad alta risoluzione con potere risolutivo 1/15000. Questo strumento permette la separazione degli isotopi, ad esempio gli isobari ^{132}Cs e ^{132}Sn. Per migliorare la capacità selettiva dei separatori isobari bisogna operare con un'energia di ingresso dell'ordine dei 200 keV: questo è possibile se sia il target sia il primo separatore vengono montati su una piattaforma di alto voltaggio a 250 kV.
A questo punto il fascio è pronto per venire convogliato direttamente alle sale sperimentali, ed utilizzato in esperimenti che richiedono fasci radioattivi di bassissima energia; oppure può essere post accelerato. Prima dell'iniezione del fascio attraverso PIAVE, che rappresenta il primo stadio di post accelerazione, esso viene diretto verso un *charge breeder*, un dispositivo in grado di incrementare la carica degli ioni.
Occorre tener conto del fatto che la corrente finale del fascio radioattivo dipende dall'efficienza di molti processi chimico-fisici (secondo l'equazione 1.1); complessivamente ci si aspetta, alle sale sperimentali, un fascio avente un rapporto di circa $10^6 \div 10^9$ $isotopi/s$ (molto inferiore alle 10^{13} $fissioni/s$ date dalla fissione nucleare).
Il diagramma di Figura 2.9 mostra l'intensità del fascio radioattivo, calcolata tenendo conto delle efficienze di emissione, di ionizzazione e di accelerazione, per diverse specie di isotopi.

2.1.5 La post accelerazione

Dopo la prima accelerazione, realizzata nella *facility* PIAVE, il fascio viene ora guidato verso l'acceleratore ALPI (acceleratore LINAC superconduttore).
Il complesso PIAVE-ALPI è in funzione da molti anni presso i Laboratori Nazionali di Legnaro, e negli ultimi tempi ha subito significativi miglioramenti al fine

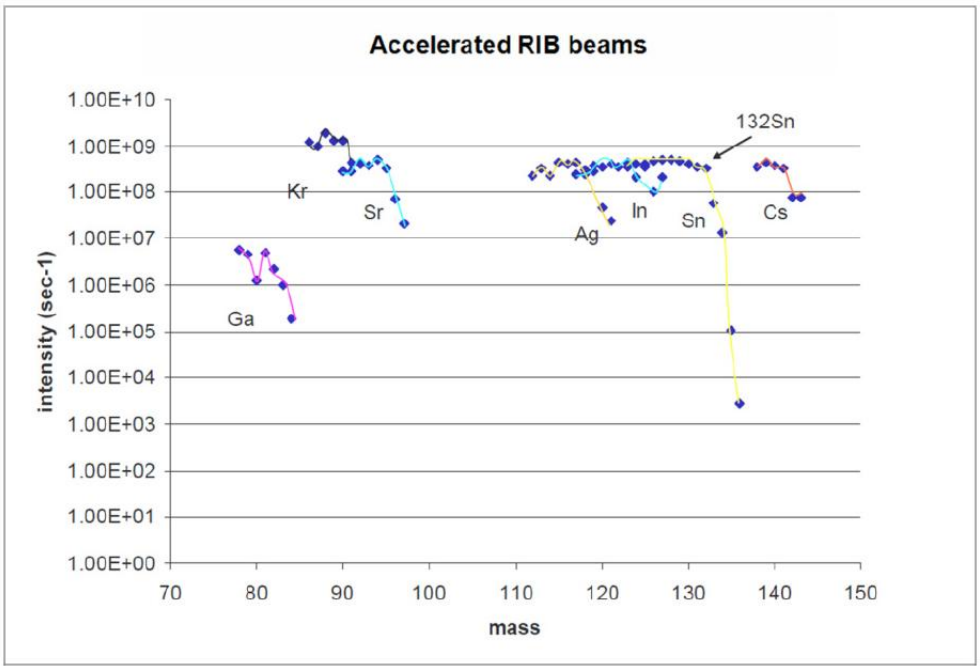

Figura 2.9: Intensità finale del fascio, calcolata tenendo conto delle efficienze di emissione, di ionizzazione e di accelerazione, per diverse specie di isotopi.

di poterlo impiegare come acceleratore di RIB.

2.2 Il Front End del progetto SPES

Come si è potuto vedere nei precedenti paragrafi, la tecnica ISOL si basa su un processo di separazione degli isotopi in linea: il fascio di particelle, proveniente dall'acceleratore primario, viene fatto collidere con il *target* di produzione: mediante reazioni nucleari si ottiene così la formazione di isotopi radioattivi. Essi vengono poi estratti e ionizzati in maniera da poter essere accelerati; questo primo stadio di accelerazione avviene nel *Front End*.
In Figura 2.10 si possono osservare gli elementi che compongono l'apparato di produzione di ioni. Bisogna prestare molta attenzione al fatto che in questa particolare sede, a causa della formazione di materiale radioattivo, si rendono necessarie particolari dispositivi di sicurezza, nello specifico per quanto riguarda la radio-protezione.

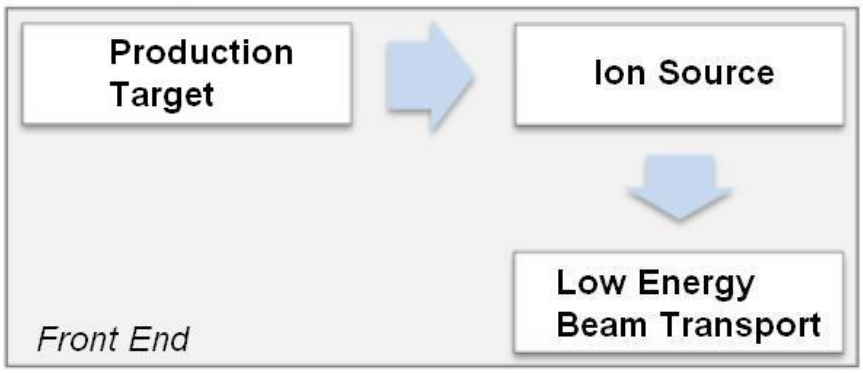

Figura 2.10: Parte della facility ISOL dove avviene la produzione degli isotopi radioattivi (Target di Produzione), la ionizzazione del fascio radioattivo (Sorgente di Ionizzazione) ed il primo step di accelerazione.

Nei Laboratori Nazionali di Legnaro sono in corso attività di ricerca e sviluppo sul target di produzione, sulla sorgente di ionizzazione e sul sistema di post accelerazione, con l'obiettivo di studiare la ionizzazione ed il trasporto di fasci di ioni *stabili*. Considerando però l'impossibilità di poter lavorare con il sistema

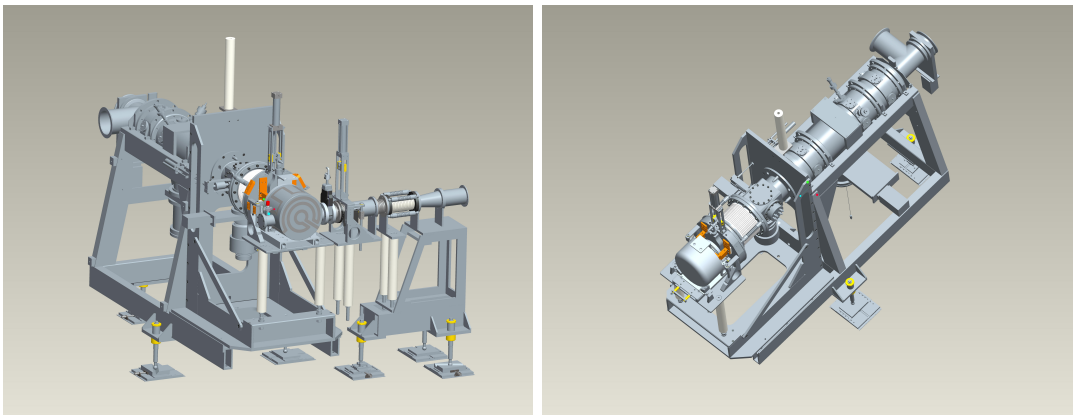

Figura 2.11: Rappresentazione del Front End *on-line* (a sinistra) impiegato all'interno della facility SPES per la produzione dei fasci di ioni radioattivi e del Front End *offline* (a destra) impiegato attualmente per le attività di ricerca e sviluppo.

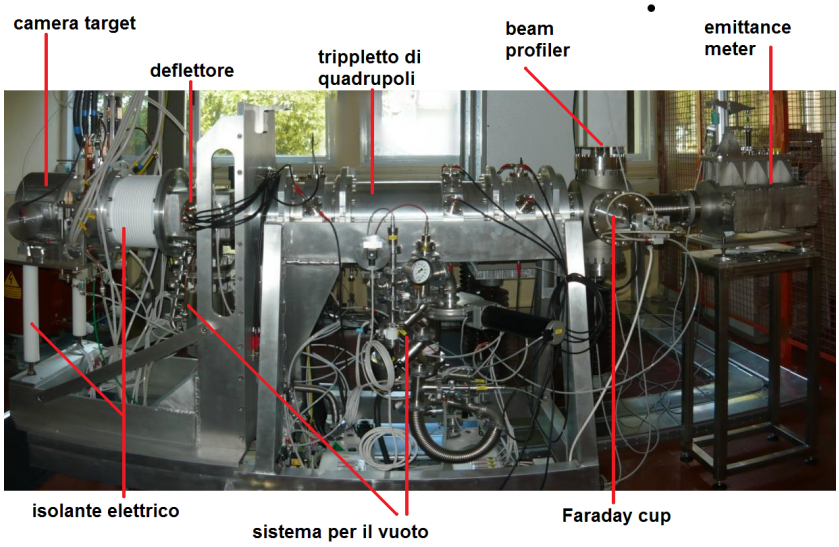

Figura 2.12: Vista del Front End offline attualmente impiegato ai Laboratori Nazionali di Legnaro.

completo, il *test bench* impiegato per queste attività (*front end offline*) risulta essere costituito dal Front End privo della linea protonica, come rappresentato nelle Figure 2.11 e 2.12.

Lo scopo che il sistema offline vuole realizzare è l'esecuzione di test di io-

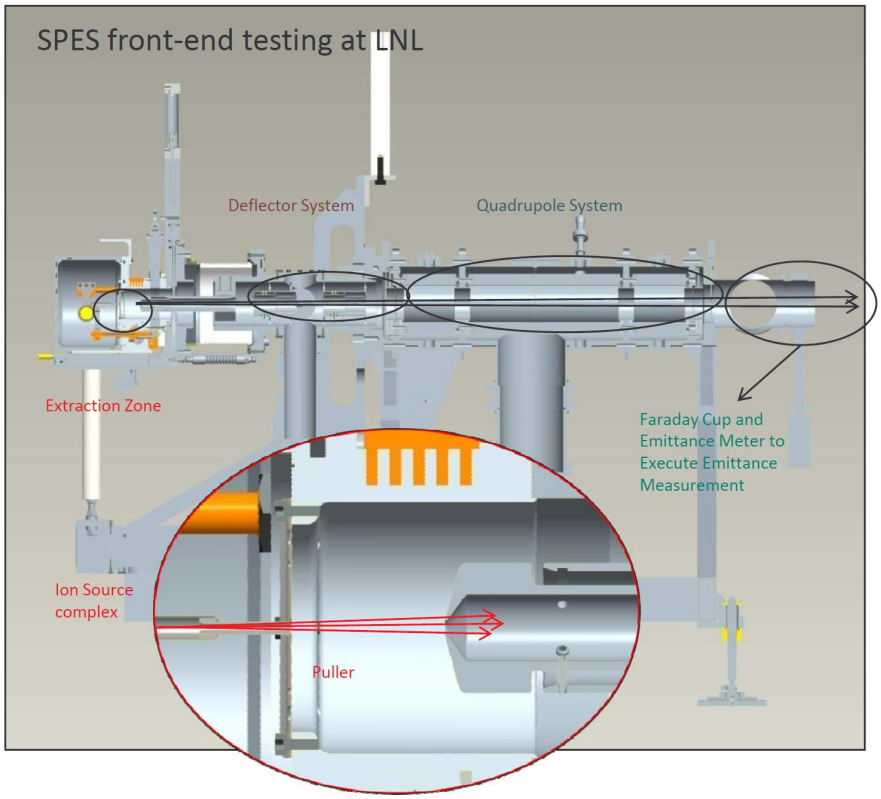

Figura 2.13: Schema del sistema Front End impiegato per i test di ionizzazione con la tecnica *Mass Marker Capillary*.

nizzazione mediante una particolare tecnica chiamata *Mass Marker Capillary*; i *Mass Marker* sono degli elementi chimici di riferimento che possono essere composti da una quantità variabile e ben definita di atomi.
I *tracer* rilasciati dal materiale di riferimento, controllabili attraverso un tubo in tantalio, si comportano come una resistenza elettrica variabile in funzione della temperatura; il principio di impiego dei *Mass Marker* si basa sulla pressione dei vapori all'interno del canale in tantalio mentre tutto il sistema si trova in una condizione di alto vuoto.
Attraverso la *Mass Marker Capillary* è possibile verificare l'efficienza di ionizzazione ed estrazione del dispositivo descritto (Figura 2.13).

Nello svolgimento delle prove di ionizzazione una sorgente calibrata, visibile in Figura 2.14, viene posta all'interno della camera target. Attraverso il circuito di riscaldamento, gli atomi presenti all'interno del provino vengono portati allo stato gassoso. Una volta avvenuto il passaggio di stato, gli atomi liberi vengono convogliati, attraverso il moto Browniano [1] delle particelle, nel tubo di tungsteno e localmente ionizzati; gli ioni vengono quindi accelerati per mezzo della differenza di potenziale tra camera target e blocco Front End, attraversano il sistema di deflessione ed il sistema di focalizzazione del fascio fino ad arrivare ad una *faraday cup* e da un *Beam Profiler*, seguito da un *Misuratore di Emittanza*, entrambi visibili in Figura 2.12.
L'emittanza è un indice qualitativo che permette la valutazione delle dimensioni trasversali e della divergenza del fascio prodotto. Confrontando quindi il numero di atomi introdotti nel sistema con il numero di atomi rilevati alla fine della linea di produzione, per mezzo del misuratore di emittanza, è possibile determinare un indice di efficienza del fascio prodotto nel Front End.

Altro importante obiettivo da conseguire è lo sviluppo dei sistemi necessari per la formazione ed il trasporto del fascio: bisogna infatti poter controllare le caratteristiche del fascio ionizzato in modo tale da minimizzare l'indice di emittanza e massimizzare l'efficienza di trasporto.
I principali dispositivi che permettono il controllo e l'interazione con il fascio ionizzato sono il blocco estrattore, costituito dall'alimentatore di alta tensione, il sistema di deflessione ed il sistema di focalizzazione. In particolar modo l'alimentatore di alta tensione fornisce la differenza di potenziale necessaria ad indurre la prima accelerazione agli ioni in uscita dalla sorgente di ionizzazione; il sistema di deflessione permette di correggere gli errori di disallineamento del fascio indotti dalle alte temperature presenti nella camera target e nel canale di tungsteno (sede della ionizzazione superficiale); il sistema di focalizzazione consente, attraverso l'ausilio di una terna di quadrupoli, di monitorare e agire sulla geometria del raggio.

[1] Il moto browniano è il moto disordinato delle particelle (dell'ordine del micrometro) presenti in fluidi o sospensioni fluide.

Figura 2.14: Rappresentazione del provino impiegato per i test di ionizzazione sul sistema offline.

In un secondo momento, a valle del sistema di focalizzazione, verrà posto un *filtro di Wien* per la purificazione, in massa, del fascio di ioni prodotto.

Presso i LNL, in un laboratorio dedicato, vengono eseguiti i test di ionizzazione, di formazione e di trasporto del fascio (Figura 2.16) [8]. Nell'area di lavoro sono presenti i seguenti dispositivi:

- il sistema Front End;
- il *rack* per la piattaforma di alta tensione;
- il *rack* per i dispositivi posti a massa;
- il trasformatore di isolamento;
- il sistema laser per la fotoionizzazione;
- la griglia di protezione per isolare il sistema Front End;
- il sistema per la produzione del vuoto nel blocco Front End;

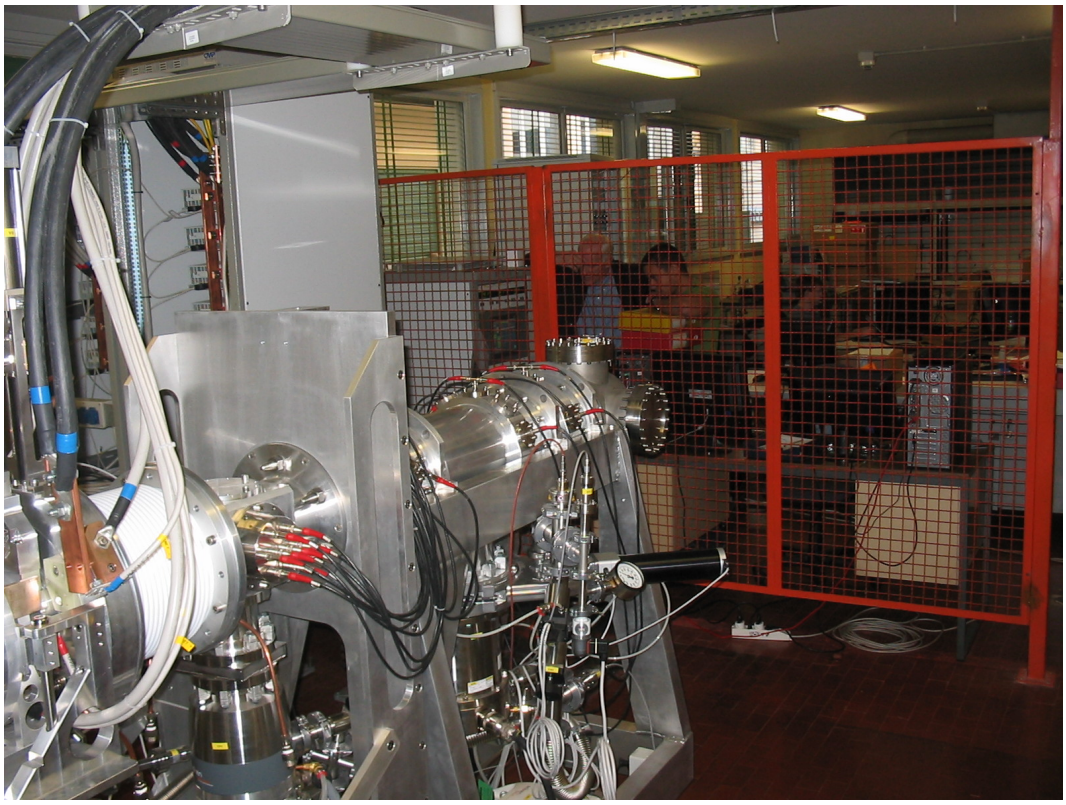

Figura 2.15: Posizioni del deflettore e dei quadrupoli all'interno del blocco Front End.

- il sistema per il raffreddamento della camera target;
- il *rack* ed il sistema di diagnostica del fascio.

Il blocco target (costituito dalla camera target e dall'estrattore) è connesso al resto della struttura attraverso degli elementi isolatori in teflon caricati con fibre di vetro[2] (Figura 2.17); tale isolamento è indispensabile dal momento che il blocco target è soggetto all'azione della differenza di potenziale di $30kV$ impiegata per fornire la prima accelerazione al fascio.

[2]Quando il sistema sarà messo in produzione i supporti di isolamento in teflon saranno sostituiti da supporti in materiale ceramico.

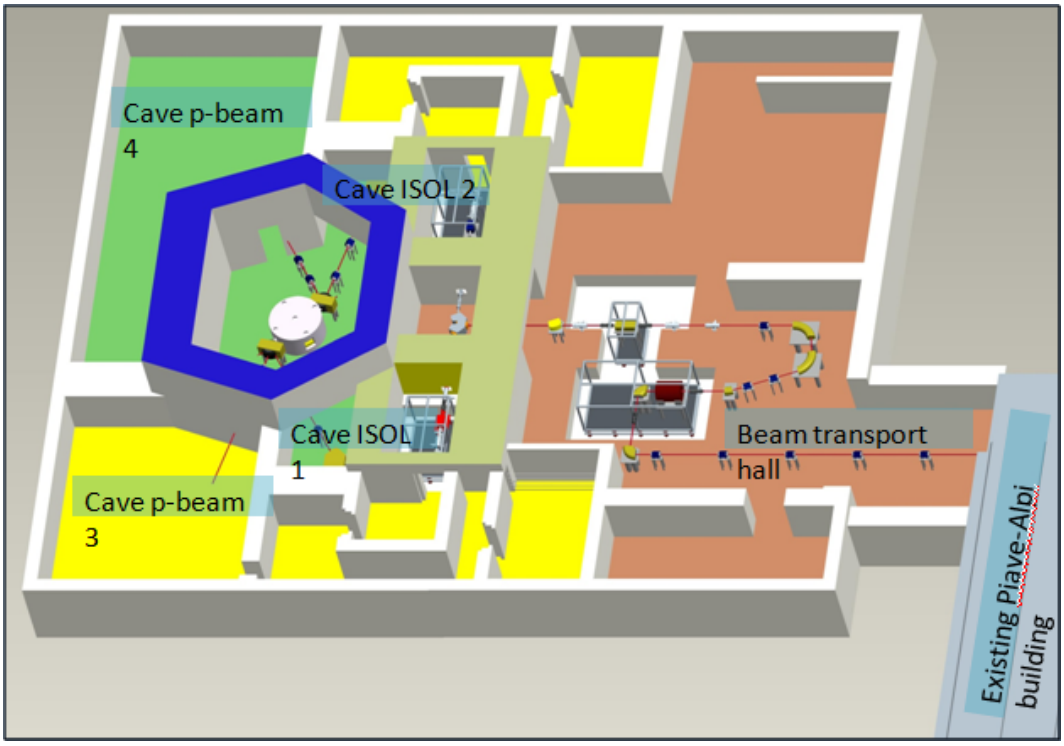

Figura 2.16: Vista del laboratorio Front End impiegato per i test e per le attività di ricerca e sviluppo sul target offline.

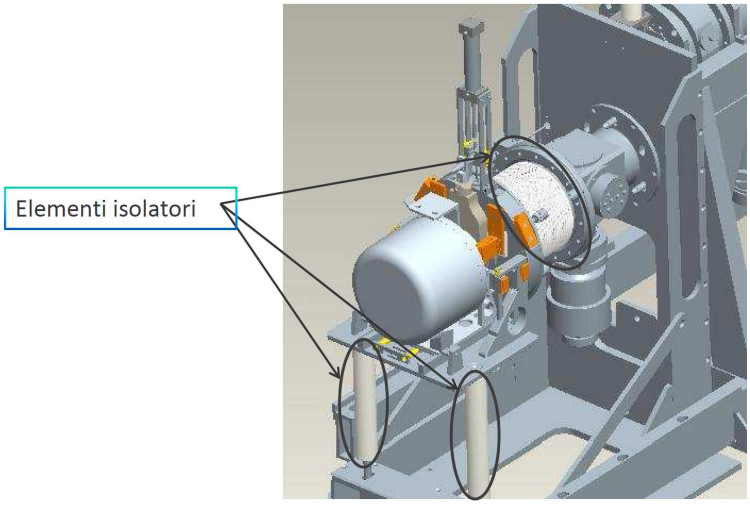

Figura 2.17: Elementi isolatori tra camera target e blocco Front End.

I dispositivi di controllo e di alimentazione del sistema Front End vengono raccolti all'interno di *rack* opportunamente organizzati:

1. una coppia di *rack*, osservabile in Figura 2.18, comprende tutti i dispositivi connessi al target con un potenziale di $30kV$ rispetto a terra. In essi sono presenti tutti gli alimentatori di corrente impiegati per il sistema di riscaldamento della camera target, per l'anodo, per la linea di produzione e la strumentazione per il monitoraggio delle temperature della camera target;

2. una seconda coppia raccoglie tutti i dispositivi connessi alle parti del blocco Front End con potenziale nullo rispetto a terra, come si può vedere in Figura 2.19. In questo gruppo sono organizzati gli alimentatori di alta tensione per i sistemi di deflessione e focalizzazione, l'alimentatore per la piattaforma di alto voltaggio ed i dispositivi *enbedded* utilizzati per la gestione dei controlli sui sistemi appena descritti;

3. un terzo *rack* viene utilizzato per contenere tutti i PLC (*Programmable Logic Controller*) con relativi cablaggi ed interfacce *touchscreen*. Tutto ciò è relativo alla gestione del sistema pneumatico e del sistema di sicurezza del *test bench*;

4. un quarto rack invece è dedicato alla strumentazione per la diagnostica del fascio, e contiene quindi tutti gli strumenti necessari ad acquisire i dati dai dispositivi sul Front End che vanno effettivamente a rilevare proprietà fisiche.

Figura 2.18: Vista dei *rack* contenenti tutti i dispositivi aventi un potenziale di $30kV$.

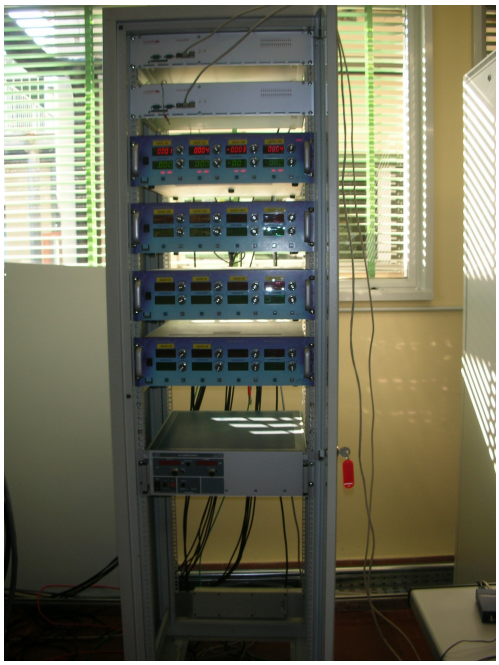

Figura 2.19: Vista del *rack* contenente i sistemi di controllo e di alimentazione per il deflettore, il tripletto di quadrupoli ed il blocco estrattore.

Bisogna isolare i *rack* contenenti i dispositivi che lavorano al potenziale di piattaforma dalle strumentazioni poste a potenziale zero e dalla rete elettrica del laboratorio; per questo si utilizzano delle connessioni in fibra ottica per la trasmissione dei dati (sia analogici che digitali) ed un trasformatore di isolamento in maniera da disaccoppiare la linea di alimentazione della piattaforma ad alta tensione dalla rete elettrica (Figura 2.20).
Osservando la Figura 2.16, si può notare che l'area dove è posto il target *offline* è delimitata da una griglia di protezione atta ad impedire l'accesso al blocco Front End durante lo svolgimento dei test. Nell'area di lavoro sono stati posizionati, oltre al *test bench*, i *rack* contenenti i dispositivi che operano al potenziale di $30kV$ ed il trasformatore di isolamento; fuori dalla zona operativa si trovano invece i rimanenti *rack* comprendenti i dispositivi per la gestione del sistema pneumatico, del sistema di sicurezza, del sistema di deflessione, del sistema di focalizzazione e l'alimentatore che fornisce la differenza di potenziale di $30kV$ al blocco target.

Figura 2.20: Dispositivi impiegati per l'isolamento della piattaforma di alto voltaggio: a sinistra il trasmettitore in fibra ottica per le trasmissioni dei dati, a destra il trasformatore di isolamento impiegato per disaccoppiare la linea di alimentazione della piattaforma di alto voltaggio dalla rete del laboratorio.

2.2.1 Introduzione al sistema di controllo

Come si evince da quanto scritto sinora sulla composizione del Front End per il progetto SPES, si tratta di un impianto che ha bisogno di apparecchiature hardware eterogenee per la sua messa in funzione, quindi si è reso necessario creare un'infrastruttura che permetta la gestione da remoto e in parallelo di tutti i componenti presenti nei rack disposti all'interno del laboratorio. Il modo più efficace per connettere tutte queste apparecchiature è utilizzare una rete ethernet efficientemente configurata, per non creare colli di bottiglia che potrebbero inficiare il corretto funzionamento delle apparecchiature. Creata l'infrastruttura di rete si sono valutati vari middleware per la gestione e condivisione di informazioni di questo sistema distribuito.
I middleware più utilizzati nel campo della ricerca sono:

1. TANGO (TAco Next Generation Objects)
2. TINE (Three-fold Integrated Networking Environment)
3. EPICS (Experimental Physics and Industrial Control System)

Presso i laboratori di Legnaro si è scelto di dare la preferenza ad EPICS: nel prossimo capitolo verranno esplicate le motivazioni che hanno portato a selezionare quest'ultimo rispetto agli altri. Configurata l'infrastruttura di rete ed il middleware, bisogna creare o selezionare gli strumenti che serviranno a creare il livello più alto dell'architettura (ovvero le interfacce utente), che per gli utilizzatori finali sarà il vero e proprio sistema di controllo dell'impianto, e che quindi nasconderà tutti i livelli più bassi tra cui quelli legati all'hardware. A Legnaro si è deciso che il programma selezionato per la creazione delle interfacce sia un mero strumento di visualizzazione grafica, e che quindi il vero sistema di controllo sia fatto al livello più basso(livello middleware). Questa potrebbe sembrare a primo avviso una scelta sbagliata o una complicazione maggiore, ma nel campo della ricerca dove tutto è in continua evoluzione, e si usa il metodo prototipale, fare i controlli a livello più alto comporterebbe che ad ogni modifica o implementazione successiva delle stesse, bisognerebbe rifare un test completo per accertarsi che i controlli esistenti o nuovi funzionino e non vadano in conflitto. Le scelte fatte in

precedenza e attualmente possibili per lo sviluppo delle interfacce sono LabView 2008 con delle patch per il supporto di EPICS, o LabView 2009, o il Control System Studio (CSS), che supportano nativamente EPICS; nel capitolo 4 verrà trattato approfonditamente e spiegato perchè si è deciso di utilizzare CSS. Un altro aspetto molto importante non solo nel campo scientifico ma in tutti gli ambiti dove c'è un massiccio flusso di dati, è la loro conservazione in modo da poter successivamente rielaborarli o visualizzarli, e questo aspetto verrà approfondito nel capitolo 5 di questa tesi.

Capitolo 3

EPICS

A differenza delle applicazioni di automazione industriale che generalmente possono essere ricondotte ad un'unica tipologia di dispositivi di controllo, come ad esempio i PLC (Programmable Logic Controller), il funzionamento di un impianto complesso come un acceleratore di particelle richiede l'impiego di strumenti dalle caratteristiche molto diverse e spesso non gestibili né attraverso un unico modello di interfaccia hardware né mediante un comune protocollo di comunicazione. Pertanto si rende necessaria l'adozione di un software di controllo unico, che permetta l'integrazione di tutte queste scelte tecnologiche.

In commercio sono disponibili diverse soluzioni che rispondono a queste esigenze, ma tra queste spicca il software EPICS, un insieme di strumenti ed applicazioni dedicate allo sviluppo di sistemi di controllo distribuiti. Secondo valutazioni eseguite da enti indipendenti, questo framework, nato da un progetto Open Source e supportato da una nutrita community, risulta essere molto competitivo dal punto di vista di robustezza e velocità di calcolo. Inoltre esso viene attualmente utilizzato all'interno di importanti progetti in numerosi laboratori di ricerca per la fisica nucleare, tra cui spicca il progetto ITER (International Thermonuclear Experimental Reactor). [1]. In base alle caratteristiche tecniche fornite, all'ottimizzazione dei costi che ne deriva dall'usare un software libero e privilegiando la collaborazione tra laboratori (aspetto molto importante in un ambito come quello della

[1]ITER progetto per la produzione di energia attraverso la fusione nucleare a cui stanno lavorando diversi team scientifici e un consorzio a cui partecipano Unione Europea, Russia, Cina, Giappone, Stati Uniti d'America, India e Corea del Sud

ricerca scientifica), i Laboratori di Legnaro hanno scelto di usare EPICS come software di controllo per la facility SPES.

3.1 Cos'è EPICS

Il software EPICS (*Experimental Physics and Industrial Control System*) è un sistema multipiattaforma [2] , costituito da un insieme di *tool*, librerie ed applicazioni Open Source dedicati alla creazione di sistemi di controllo distribuiti Real-Time, che non solo risponde a questa esigenza, ma implementa anche una funzionalità di livello superiore che si potrebbe definire "bus software", ovvero dà la possibilità di accedere al valore di una variabile di processo acquisita in un qualunque sottosistema attraverso una semplice chiamata "per nome".
Questa infrastruttura di rete, costruita sopra i noti protocolli TCP/IP e UDP, si chiama *Channel Access* e costituisce un middleware attraverso cui un processo Server rende disponibili i suoi dati a un numero virtualmente illimitato di processi Client senza che il Client debba conoscere a priori quale sia il Server che gli fornirà l'informazione richiesta.
In figura 3.1 è rappresentato un esempio di possibile architettura di un sistema di controllo basato su EPICS. Le caratteristiche di velocità e robustezza del *Channel Access* sono state riconosciute da valutazioni condotte in modo indipendente da diversi Laboratori e queste, oltre alla già citata indipendenza dalla piattaforma hardware, sono state tra le motivazioni che hanno portato alla decisione di usare EPICS come strumento per lo sviluppo dei controlli di SPES.

Epics non è quindi un software monolitico ma è composto da diversi moduli, che si possono suddividere nelle seguenti tre categorie:

- *EPICS base*:
  rappresenta il *core* dell'ambiente di sviluppo. Esso mette a disposizione i principali componenti e librerie per creare e gestire una rete di controllo

[2] La lista dei sistemi operativi supportati da EPICS è reperibile nel sito `http://www.aps.anl.gov/epics/about.php`

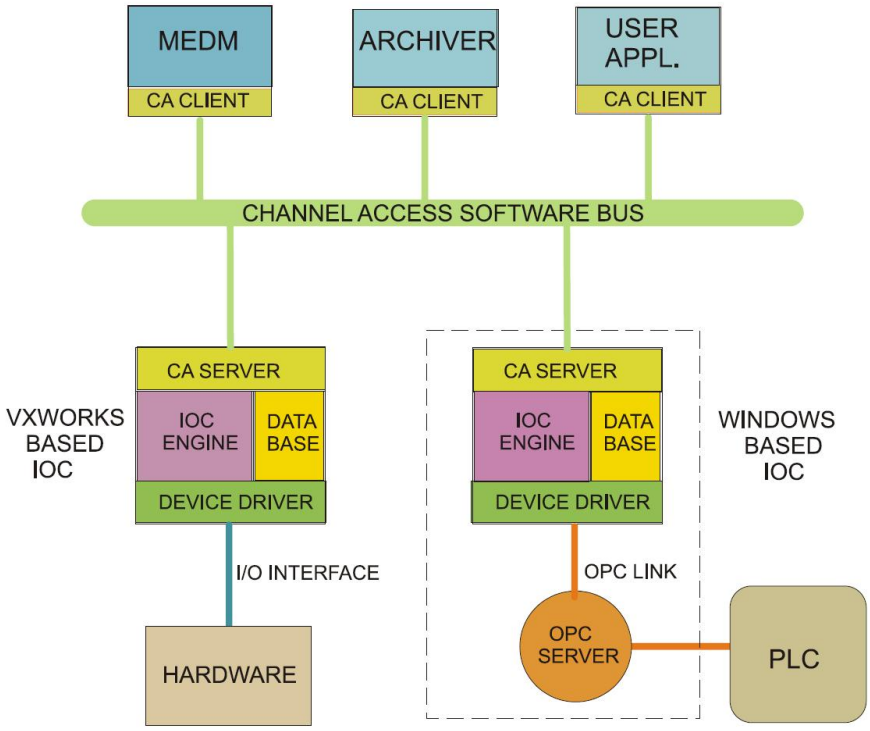

Figura 3.1: Esempio di architettura di un sistema di controllo basato sul software EPICS.

distribuita: le librerie standard per la costruzione degli algoritmi di controllo, le librerie di comunicazione con il sistema operativo, le librerie per la gestione dell'infrastruttura di comunicazione (*Channel Access*);

- *EPICS extensions*:
  in questa categoria vengono inseriti tutti i *tool* EPICS per la gestione degli *host*, un insieme di quest'ultimi viene anche chiamato OPI (Operator Interface), e le applicazioni client dedicate all'*editing* dei software di controllo, nonché alla gestione e alla diagnostica della rete di controllo distribuito;

- *EPICS module*:
  all'aumentare del numero dei supporti scritti dalla comunità EPICS per realizzare le interfacce verso differenti hardware, si è optato per mantenere separati tali moduli dal pacchetto *EPICS base*. In questo modo il singolo

utente può selezionare i supporti necessari in funzione dell'applicazione che deve sviluppare. Gli *EPICS module* vengono a loro volta catalogati come segue:

- *Soft Support*: ovvero i supporti EPICS di tipo software, come ad esempio nuovi *Record type*;
- *Hardware Support*: ovvero i supporti EPICS necessari per comunicare con i dispositivi hardware.

Le applicazioni server, chiamate IOC (*Input/Output Controller*), eseguono gli algoritmi di controllo locali e si interfacciano, attraverso gli appositi *module*, con le eventuali strumentazioni hardware. Le informazioni elaborate dai server vengono poi rese disponibili alle applicazioni client attraverso il protocollo *Channel Access*. Si fa notare che le applicazioni server vengono fornite dal pacchetto *EPICS base*, mentre per poter utilizzare le applicazioni client desiderate è necessario installare le apposite *extension* all'interno dell'ambiente di sviluppo.

3.1.1 Channel Access e IOC

Il Channel Access fornisce un accesso trasparente al IOC databases[3] Come già accennato precedentemente è un'architettura basata sul modello client/server, ogni IOC ha il suo interno il Channel Access server il quale provvederà a stabilire una connessione con un arbitrario numero di client, e ogni OPI e IOC contiene al suo interno il Channel Access client. I servizi client di base messi a dispozione dal Channel Access sono:

- Search: individua gli IOC che contengono le variabili richieste e stabilisce la comunicazione con ciascuna;
- Get: ottiene il valore più tutte le informazioni aggiuntive e facoltative di quella variabile;

[3]Non si deve considerare come il concetto di database classico ma come la locazione di memoria dove risiedono le Process Variable definite all'interno dell'applicazione IOC.

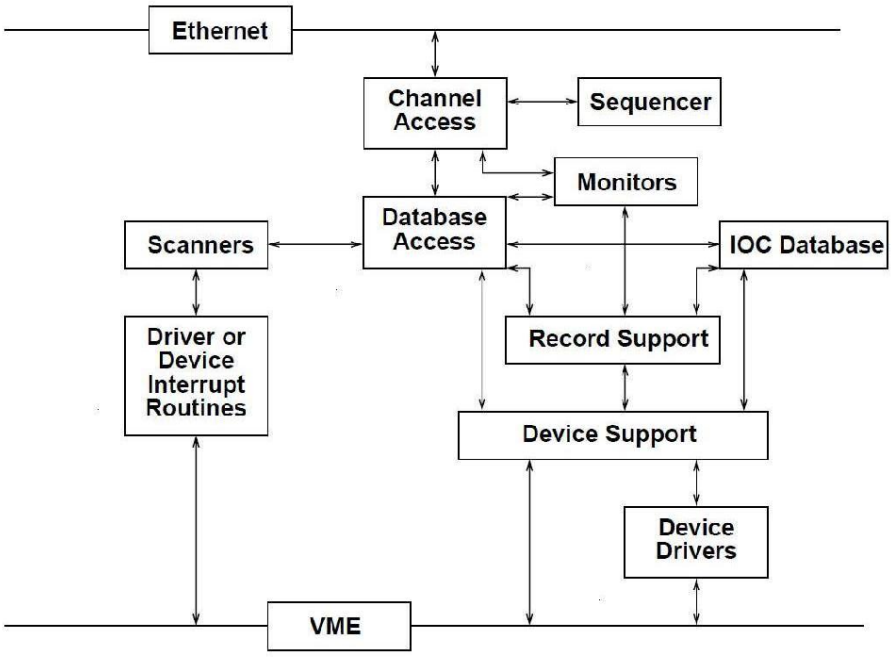

Figura 3.2: Schema degli elementi costituenti un Input/Output Controller di base

- Put: cambia il valore delle variabili richieste con quello passato alla funzione;

- AddEvent: questa è una richiesta per avere informazioni dal server solo quando la Process Variable associata cambia di stato, ovvero cambia un valore interno.

Per maggiori informazioni si rimanda alla guida [9] o all'elaborato di Maurizio Montis [10]. L'Input Output Controller, come si è detto in precedenza, esegue gli algoritmi di controllo, questi ultimi creati tramite l'utilizzo di database al cui interno vengono definiti dei Record i quali identificano univocamente le Process Variable. La sua struttura è composta come mostrato in figura 3.2:

- l'*IOC Database*: è la memoria dove risiedono le *Process Variable* definite all'interno dell'applicazione IOC. Tali variabili di processo saranno poi disponibili per tutti i client ed IOC, presenti nella rete di controllo, che ne fanno richiesta;

- il *Database Access*: è la routine preposta alla gestione degli accessi verso l'IOC Database. Essa però non gestisce gli accessi alla memoria dati eseguiti dal *Record Support* e dal *Device Support*;

- lo *Scanner*: è il meccanismo con cui vengono definite le modalità di aggiornamento dei valori delle *Process Variable*;

- il *Record Support* ed il *Device Support*: sono le *routine* dedicate alla gestione dei Record e dei Support sviluppati da terze parti. In particolare vengono usati quando sono definite rispettivamente nuove tipologie di Record e nuovi moduli per realizzare la comunicazione con i dispositivi hardware;

- il *Device Driver*: gestisce gli accessi verso un dispositivo esterno collegato all'IOC. Il *Device Driver* viene usato quando il metodo di accesso verso un particolare hardware è molto complesso rispetto a quello che un *Device Support* può elaborare;

- il *Channel Access*: in questo caso identifica il software di interfaccia tra il controllore ed il canale di trasmissione;

- il *Monitor*: è una routine utilizzata per supervisionare l'aggiornamento dei valori delle *Process Variable* e definire il meccanismo di *callback* per la trasmissione dei dati. La tecnica denominata *"reporting by exception"*, o *callback*, si basa sulla seguente procedura: ogni volta che un client risulta interessato ad un certo dato situato in un particolare server, tale server esegue la notifica del dato solamente quando questo cambia. Così facendo, non solo si minimizza il traffico sulla rete, ma è anche possibile monitorare la salute del server;

- il *Sequencer*: permette di realizzare una macchina a stati finiti all'interno dell'applicazione IOC.

In EPICS, con il termine *Process Variable* viene indicata l'informazione elementare scambiata tra i *Channel Access server* e i *Channel Access client* presenti nella rete di controllo. Le *Process Variable* rappresentano quindi i parametri di

interesse per poter gestire il sistema.

3.1.2 OPI

Essendo questo un software sviluppato da un'ampia comunità scientifica i mezzi per interfacciarsi ad Epics usando il Channel Access sono numerosi, alcuni ormai anche andati in disuso e non più utilizzati, perchè sostituiti da più nuovi ed efficienti tool. Tutti questi ricadono nelle sessione riguardante le estensioni di EPICS e sono stati sviluppati per cercare di risolvere tutte le esigenze di cui un laboratorio di ricerca ha bisogno o potrebbe aver bisogno. Ci sono strumenti per la creazione dei IOC Databases come VDCT (Visual Database Configuration Tool)[10], strumenti per creare le interfacce operatore come MEDM, strumenti per la gestione degli allarmi ALH (Alarm Handler), strumenti per la memorizzazione delle Process Variable come Channel Archiver, strumenti per la visualizzazione dei dati salvati come Archiver Viewer, per la monitorizzazione dei dati come StripTool e molti altri[4] Un operatore che lavora con Epics sicuramente si è scontrato con questi strumenti, e prima di valutare il funzionamento del Control System Studio, conviene utilizzare tutti questi strumenti per meglio capire ed utilizzare il CSS il cui scopo è appunto racchiudere in un software unico tutte le funzionalità maggiormente usate di EPICS, dando inoltre la possibilità, dato che è Open Source, a tutta la comunità di contribuire allo sviluppo creandosi i propri plug-in per integrare funzionalità ancora non presenti.[5]

[4]Per la lista completa delle extension di Epics si rimanda al seguente link `http://www.aps.anl.gov/epics/extensions/index.php`

[5]`http://cs-studio.sourceforge.net/`

Capitolo 4

Control System Studio

Il Control System Studio[11] è un sistema di controllo che cerca di incorporare molte applicazioni, già esistenti in EPICS, in un unico prodotto che sia facile da gestire ed utilizzare anche da chi ha poca dimestichezza con l'informatica. Tuttora molte applicazioni EPICS sono sviluppate per UNIX e X-Windows (X11), e sono tutte indipendenti l'una dalle altre, rendendo così difficile l'interscambio di dati; e non agevolano l'utente che si trova davanti ad interfacce con diversi stili e logiche. Questo è dovuto al fatto che il sistema di X11 è stato progettato appositamente per non fornire alcun supporto predeterminato per gli elementi dell'interfaccia utente (quali pulsanti, menù, barre del titolo); l'infrastruttura viene quindi creata e fornita dagli ambienti desktop. Per questo motivo le interfacce di X sono variate considerevolmente nel tempo e sono differenti passando da un desktop manager all'altro. Il CSS è creato sull'attuale tecnologia software, ovvero Java ed Eclipse, con speciale attenzione alla interoperabilità. Un punto di forza di questo prodotto è che l'interfaccia grafica è basata appunto su uno dei più moderni ed utilizzati framework per applicazioni grafiche Eclipse [1] RPC base. Questo permette di integrare o cambiare facilmente le applicazioni, in quanto pensate come plug-in, che quindi possono essere selezionate da un utente. Un'altra importante caratteristica è l'accessibilità dei dati a tutte le applicazioni dovuta all'utilizzo di

[1]Eclipse è un ambiente di sviluppo integrato multi-linguaggio e multipiattaforma. Ideato dall'Eclipse Foundation, un consorzio di grandi società quali Ericsson, HP, IBM, Intel, MontaVista Software, QNX, SAP e Serena Software, esso viene sviluppato da una comunità strutturata sullo stile dell'open source.

un livello intermedio che opera da interfaccia tra i tipi CSS-data definiti nel CSS ed i dati definiti nel sistema di controllo; questo livello è chiamato Data Access Layer (DAL) e assicura l'accesso trasparente ad ogni protocollo di controllo di sistema, quindi non è solo una piattaforma per EPICS ma per tutti i protocolli di sistema che implementano DAL, come ad esempio TAco Next Generation Objects (TANGO) [2] e Three-fold Integrated Networking Environment (TINE) [3].

4.1 Architettura del CSS

Il CSS non crea un nuovo ambiente di lavoro, ma usa il Rich Client Platform (RCP) Eclipse. L'Eclipse RCP è scritto in Java e può essere eseguito su tutti i sistemi operativi per i quali è disponibile la Java Virtual Machine (JVM). La piattaforma di sviluppo è incentrata sull'uso di plug-in, che sono dei componenti software ideati per uno specifico scopo; questo permette una più agevole estendibilità del prodotto in esame. E' provvisto di alcuni profili generici per applicazioni, tra cui menù, preferenze, aiuti di sistema. Il runtime di Eclipse è la base per ogni installazione Eclipse RCP. Il runtime del programma controlla e carica tutte le esecuzioni che stanno provvedendo alla funzionalità delle applicazioni. La decisione che ha portato gli sviluppatori del CSS ad utilizzare Eclipse, piuttosto che Netbeans o altri RCP, nasce dal fatto che esiste una comunità sempre più vasta di persone capaci di migliorare ed implementare la struttura di Eclipse, in quanto essa si fonda sulla tecnologia OSGi (conosciuta come Open Service Gateway initiative)[4]

[2]TANGO: sistema di controllo distribuito orientato agli oggetti che utilizza CORBA, per la gestione degli acceleratori, attualmente utilizzato in molti laboratori che utilizzano acceleratori di particelle

[3]TINE: sistema di controllo, che provvede al controllo e all'integrazione degli acceleratori al Deutsches Elektronen Synchrotron-DESY, ad Amburgo, in Germania

[4]L'alleanza di OSGi è un'organizzazione che ha definito:

- la gestione del modello del ciclo di vita del software;
- i moduli (chiamati bundles);
- un service registry;
- un ambiente di esecuzione.

Il CSS consiste di due parti:

- il nucleo centrale: plug-in che definiscono i dati di controllo, la gestione delle istanze, e le User Interface UI;

- le applicazioni: plug-in che implementano i vari sistemi di controllo (ad esempio i widget); nel seguito del capitolo verranno riportate quelle con rilevanza maggiore e che si sono attivate ed utilizzate per i laboratori dell'Istituto Nazionale di Fisica nucleare (INFN) di Legnaro (PD).

Tecnicamente i plug-in del CSS sono plug-in ordinari di Eclipse, infatti se si hanno tutti i plug-in che compongono il nucleo del CSS e quelli che compongono il sistema di controllo vero e proprio, si può benissimo avviare il CSS come un Eclipse IDE[5], ovvero avere nell'ambiente di sviluppo di Eclipse anche tutti gli strumenti del CSS. Ovviamente utilizzarlo in questo modo è improponibile dato che si vuole che il CSS possa essere adottato anche da un utente che non abbia conoscenze informatiche, ma l'esempio serve solo a chiarire che il CSS è un Eclipse IDE minimizzato per eseguire solo gli applicativi del CSS, come si vede in Figura 4.1. Ogni plug-in di CSS, che implementa l'extension point di Eclipse, sarà automaticamente caricato nel menù del CSS, ed quindi sempre automaticamente si avranno a disposizione anche le relative pagine di aiuto, se previste dallo strumento.

Un altro vantaggio da non sottovalutare che ha portato alla scelta di utilizzare Eclipse RCP è che l'avvio delle applicazioni *stand alone* di Java è molto rapido. Inoltre tutti gli applicativi del CSS che si avviano sono gestiti dalla stessa Java Virtual Machine (JVM) di Eclipse runtime, quindi tutti i servizi comuni come la connessione al database sono inizializzati una volta sola all'avvio del sistema di controllo, cosa che non sarebbe possibile se tutti gli applicativi fossero indipendenti, in quanto ognuno di loro avrebbe la propria JVM e la condivisione dei servizi non sarebbe possibile.

[5]Integrated Development Environment, ambiente di sviluppo integrato per la realizzazione di programmi per sistemi informatici

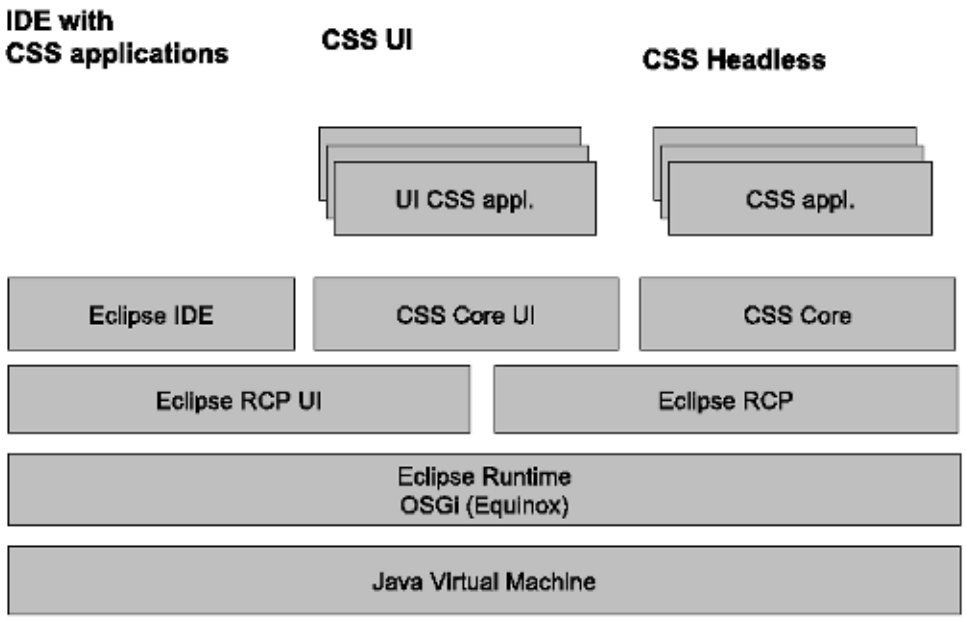

Figura 4.1: Architettura dell'Eclipse IDE e del CSS.

4.2 Data Access Layer (DAL)

Il Data Access Layer (DAL) è un insieme di interfacce API (Application Programming Interface) [6] per un accesso dinamico dei dati ad un sistema di controllo. Queste API disaccoppiano quindi il livello dati dal tipo di Sistema di controllo EPICS, TINE, TANGO, come la figura 4.2 chiarisce.

Sono integrate in un plug-in con il metodo dell'extension point; in questo modo un plug-in con una nuova implementazione per un sistema di controllo può essere installato senza dover ricompilare il nucleo del CSS. Per decidere quale implementazione si deve utilizzare, il nome della variabile di processo (PV, process variable) deve iniziare con un prefisso tipo: <nome del sistema di controllo>:// <nome della PV>, in questo modo un programma all'interno del nucleo del CSS provvederà tramite un parser a catturare il prefisso ed a utilizzare correttamente il DAL corretto. Se il prefisso viene omesso il sistema provvederà ad utilizzare il sistema di controllo di default definito nella configurazione 4.3. Attualmente le

[6] Si tratta di un insieme di procedure disponibili al programmatore, di solito raggruppate a formare un set di strumenti specifici per un determinato compito.

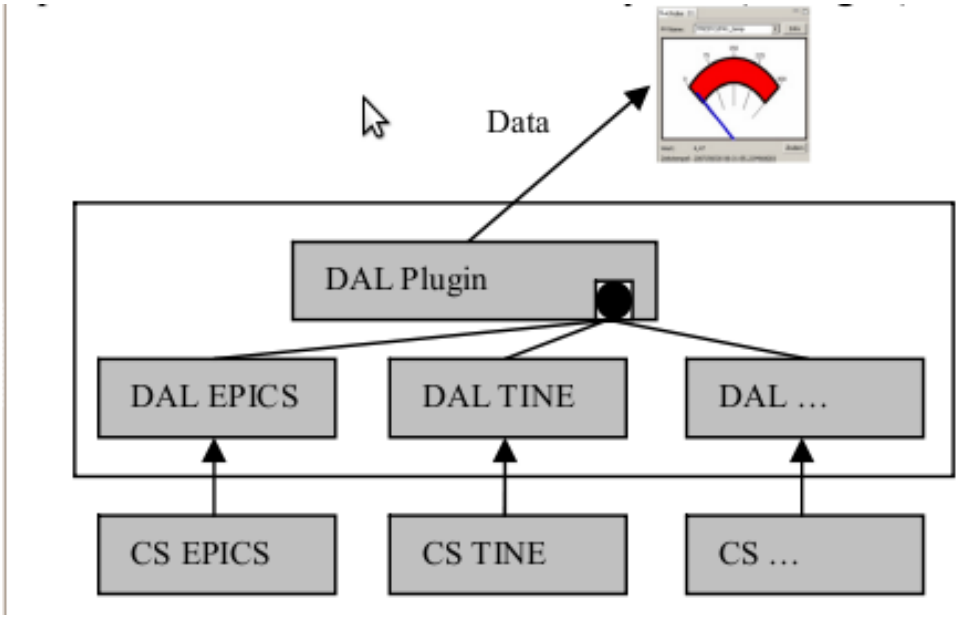

Figura 4.2: Architettura del Data Access Layer.

implementazioni di DAL disponibili sono per EPICS, TANGO e TINE. Nel CSS attualmente utilizzato presso i laboratori di Legnaro è presente solo il plug-in necessario a definire il Data Access Layer di Epics.

4.3 Applicazioni e configurazione del CSS

Sono ad oggi disponibili molte applicazioni per il CSS. Le applicazioni generalmente più utilizzate sono PROBE, EPICS PV Tree, BOY, DATA BROWSER; si tratta di quelle utilizzate all'interno dei Laboratori INFN di Legnaro. Di seguito si riporta in breve una descrizione delle loro caratteristiche e del loro utilizzo. Come si è detto precedentemente, il CSS permette grazie alla sua peculiarità di inglobare al suo interno quegli applicativi che prima erano individuali,e una comunicazione trasparente e semplice tra questi, attraverso dei menu specifici, i quali vengono attivati selezionando la PV desiderata e premendo il tasto destro come si potrà osservare nell'immagine 4.6.

4.3.1 Pannello di configurazione

Prima di iniziare ad utilizzare questo prodotto nella sua totalità si deve approntare una configurazione minima per indicare al prodotto quali sono gli IOCServer presenti nella rete, con quale sistema di controllo ci si deve interfacciare e quindi quale protocollo utilizzare per la comunicazione, e quale database si è scelto di usare. Per ulteriori dettagli sulla configurazione del prodotto si rimanda alla guida online dello stesso. Si fa inoltre notare che per alcuni strumenti come DATA BROWSER, BEAST, Message Log, è necessario installare un database MySQL o Oracle (attualmente gli unici supportati) ed altri software non forniti assieme al pacchetto base; qui si è scelto di riportare la configurazione necessaria dopo che si è fatto quanto riportato in Appendice per i tool sopra indicati. Dal menu principale selezionare CSS → Preferences come si può vedere in Figura 4.3, andare alla voce Control System e selezionare quello desiderato, nel caso attuale EPICS. Fatto ciò si deve andare alla voce EPICS e selezionare i parametri

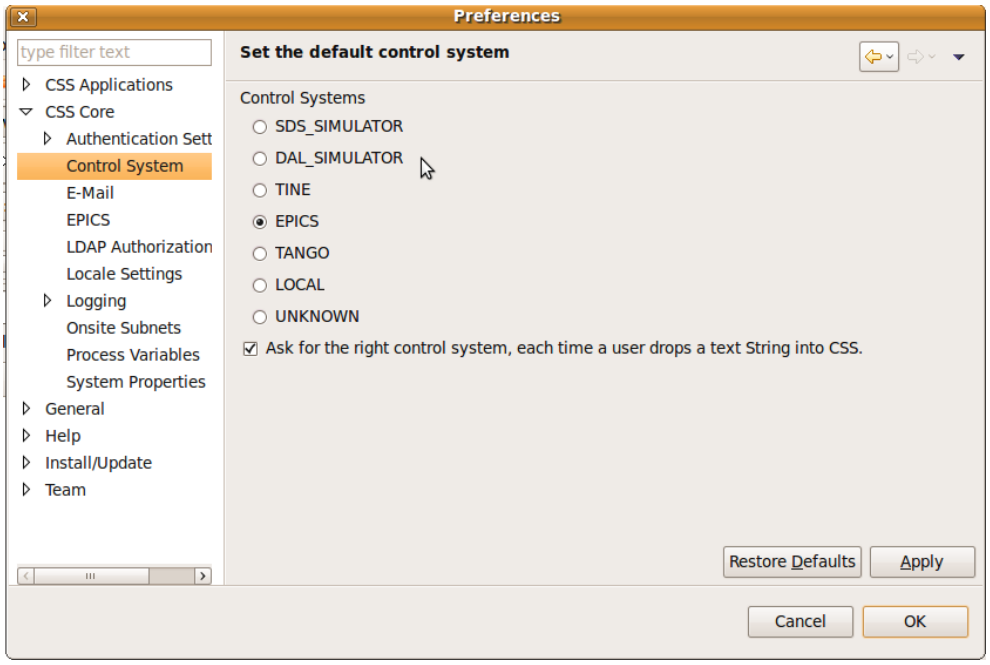

Figura 4.3: Pannello per la scelta del sistema di controllo da utilizzare

come mostrato in figura 4.4; alla voce addr_list inserire tutti gli indirizzi IP delle macchine che contengono uno IOCServer. Apportando solo queste modifiche alla

configurazione si arriverà a poter utilizzare tutti gli strumenti che s'interfacciano con il Channel Access Epics.

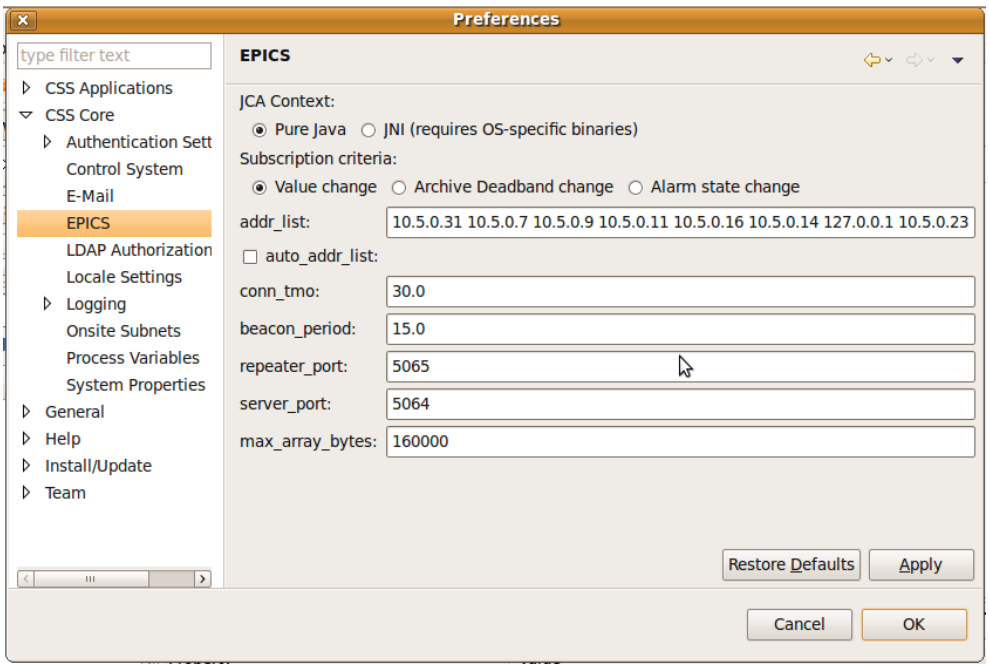

Figura 4.4: Configurazione di Epics

4.3.2 Probe

Il CSS Probe permette di scrivere e leggere il valore corrente di una Process Variable, ed è molto utile per controllare se una PV esiste e se assume valori corretti. Questo tipo di applicazione si apre all'interno del CSS e si adatta automaticamente alla grandezza della finestra, ma come tutti i plug-in Epics essendo drag and drop può essere spostata all'interno dell'interfaccia dell'area di lavoro oppure staccata dall'interfaccia principale per crearne un frame a sé stante.

Utilizzo

Questo tipo di applicazione non ha bisogno di nessun tipo di configurazione preliminare se non quelle dell'applicativo principale; a questo proposito si veda quanto riportato nel precedente paragrafo. Il campo Nome PV, come si vede dall'immagine 4.5 offre, tramite un menu a tendina, una lista con tutti i nomi preceden-

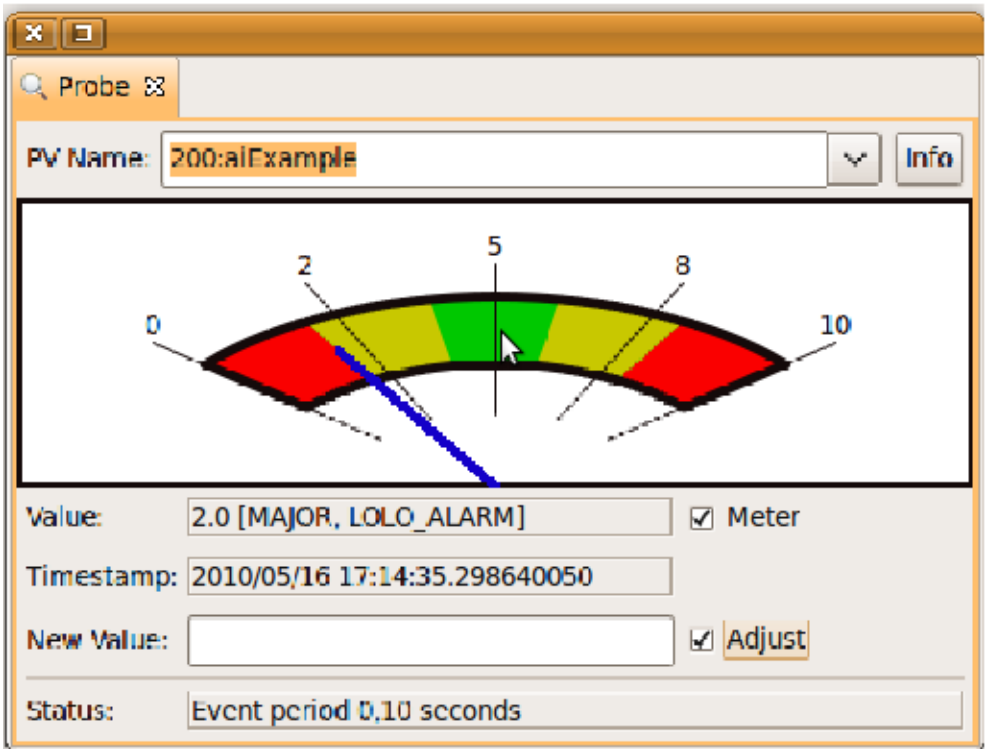

Figura 4.5: Probe.

temente digitati; se all'interno di questa lista non è presente la PV desiderata si può inserire il nome desiderato, seguito dal tasto Invio. Il tool mostra il valore sia graficamente, attraverso un indicatore analogico suddiviso in aree per individuare visivamente quando la variabile e' in allarme; sia tramite il campo Value dove sono riportati il valore, lo stato, ed il livello di allarme; inoltre nel campo Timestamp è possibile leggere quando è avvenuto l'ultimo cambiamento della variabile. Il Bottone "Adjust" apre una finestra di testo dove è possibile inserire il valore che si vuole impostare alla PV. Infine c'è una barra di stato che mostra ogni quanto la variabile viene aggiornata: questo se tutto funziona correttamente, altrimenti mostra i messaggi di errore pervenuti.

4.3.3 EPICS PV Tree

Questa applicazione visualizza i record EPICS in modo gerarchico, mostrando il tipo di record, il loro valore attuale e lo stato degli allarmi, seguendo il flusso di

dati attraverso i link di ingresso dei record EPICS (INP,INPA,DOL,...).

Utilizzo

Questo tipo di applicazione può essere attivata o dal menu generale del CSS (windows → Show View → Others → PV Tree) oppure da un'altra applicazione selezionando la variabile desiderata e premendo il tasto destro del mouse, facendo in questo modo comparire un menu a tendina dove scegliere alla voce CSS un'altra applicazione come illustrato in figura 4.6.

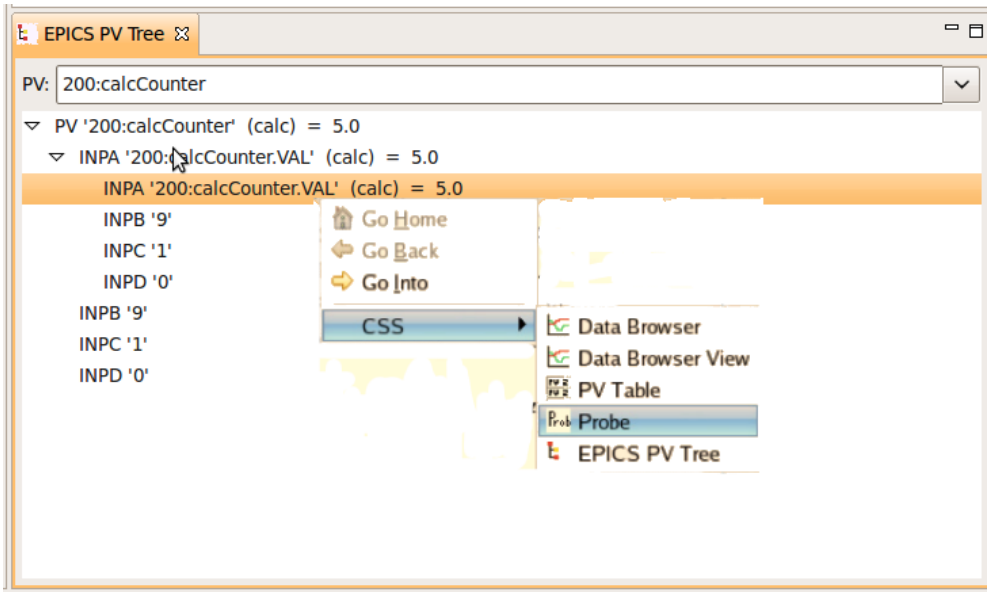

Figura 4.6: Pv Tree con relativo menu

4.3.4 DATA BROWSER

Il Data Browser[12] è un applicativo molto utile che permette attraverso un grafico temporale a scorrimento di visualizzare sia i valori correnti che quelli archiviati dal sistema di controllo. Gli utenti in questo modo hanno un accesso trasparente e semplice ai vari sistemi di memorizzazioni dei dati. Infatti data la continua evoluzione del CSS questo tipo di applicativo è rimasto retro-compatibile nel tempo sia con le vecchie architetture per la memorizzazione dei dati, sia con le innovazioni più recenti che utilizzano i database relazionali MySQL ed Oracle.

Attualmente infatti è previsto che lo strumento possa visualizzare dati memorizzati tramite un applicativo esterno chiamato ArchiveEngine, di cui si tratterà in seguito, obsoleto ma tuttora utilizzato dalla stragrande maggioranza dei laboratori scientifici che utilizza EPICS, grazie alla sua stabilità e velocità nella memorizzazione dei dati e all'indiscutibile vantaggio di essere a costo zero. Questo strumento inoltre offre caratteristiche simili a quelli già esistenti come il tool chiamato Strip Tool o Archive Viewer, semplificandone l'utilizzo. Entrambi questi strumenti acquisiscono i dati dal Channel Access e disegnano in tempo reale i dati pervenuti in un grafico a righe. Sono molto utili per controllare e quindi correggere le applicazioni di controllo, e per monitorare l'andamento dei dati.

Utilizzo

Dato il nome di una Process Variable (PV) il Data Browser mostrerà i valori di questa PV nel corso del tempo come si può vedere in Figura 4.7.

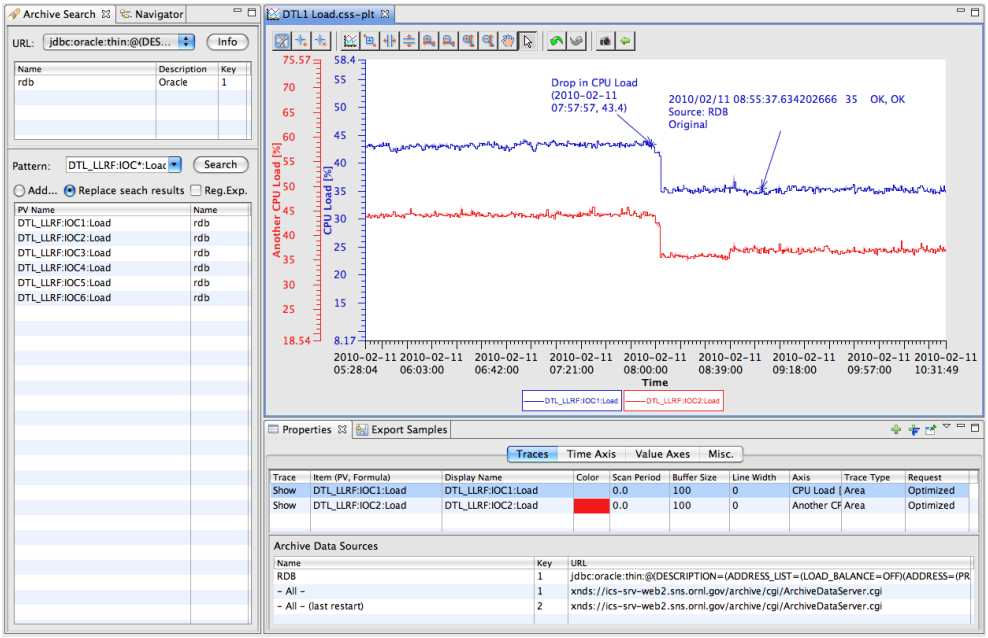

Figura 4.7: CSS Data Browser con la configurazione riportata in basso; a sinistra il tool Archive Search View, per la visualizzazione delle PV memorizzate

Figura 4.8: Barra degli strumenti del Data Browser

I nomi delle PV possono essere inseriti direttamente tramite un apposito menu che si attiva cliccando con il tasto destro del mouse nell'area del grafico, oppure tramite una ricerca nell'archivio selezionato, o ancora tramite drag and drop da un'altra applicazione del CSS con il solito menu che compare selezionando e premendo con il tasto destro del mouse sopra la PV. La barra sovrastante l'area del grafico (Figura 4.8) permette varie funzioni, tra cui vari tipi di zoom, la possibilità di inserire e togliere note direttamente sul grafico, ed un pulsante di navigazione; è predefinito che l'effetto dei pulsanti sia relativo all'intera area del grafico, se non viene selezionato l'asse delle "Y" desiderato. Si rimanda alla guida per le altre funzionalità di tale barra.

Nell'area sotto il grafico, come si può vedere in Figura 4.7, compare un pannello di configurazione suddiviso a sua volta in altri due pannelli: uno utilizzato per definire le proprietà ed uno per definire come si vuole che i dati vengano esportati. Nel primo pannello, quello delle proprietà, si possono definire vari aspetti, come la scala degli assi ed i loro limiti, se utilizzare la funzione di ottimizzazione o meno dei dati: nel primo caso viene fatta una media sugli ultimi 800 campioni e nel secondo caso vengono visualizzati i dati grezzi. Si fa notare inoltre che durante la sua esecuzione questo applicativo colleziona periodicamente i dati reali del sistema di controllo con un periodo configurabile nel suddetto pannello di controllo, mentre un buffer ad anello permette di aggiustare la quantità di campioni da memorizzare in modo da prevenire l'esaurimento della memoria. Altamente innovativa è la possibilità, nel campo dove si aggiungono le PV, di fare anche delle piccole elaborazioni di base, che sono quelle attualmente messe a disposizione dalla libreria java.lang.Math; queste formule posso essere inserite tramite un comodo pannello rappresentato in Figura 4.9.

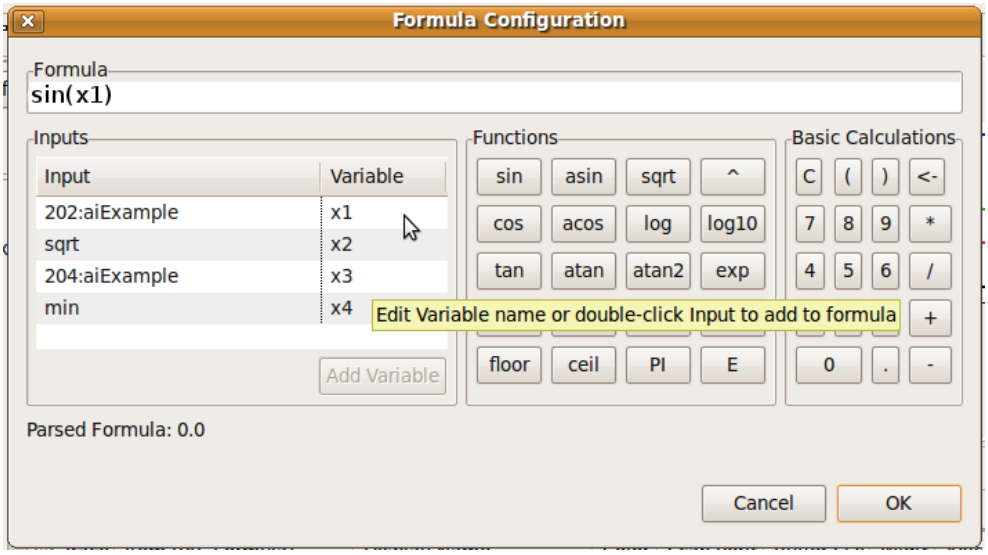

Figura 4.9: Pannello per l'inserimento delle formule

4.3.5 BOY: modulo per la creazione dell'interfaccia operatore

Il CSS BOY è un'interfaccia operatore (OPI) per lo sviluppo dell'ambiente di controllo che l'operatore finale arriverà ad utilizzare. Questo tipo di interfaccia è stato pensato nell'ottica che ogni operatore di un esperimento messo a conoscenza della struttura, e quindi delle variabili EPICS utilizzate per implementare il sistema di controllo vero e proprio, possa crearsi la propria interfaccia in modo semplice ed immediato con gli strumenti (widget) messi a disposizione dal CSS. Un grosso vantaggio dell'utilizzo dell'editor BOY, è che si possono sviluppare potenti OPI in pochi minuti ed una volta creati cominciano a funzionare immediatamente. L'idea principale di questo strumento, accennata sopra, è collegare dei widget per leggere e scrivere le PV, semplicemente specificando al loro interno mediante un pannello delle proprietà la PV da associare; inoltre è possibile aumentare le potenzialità del widget aggiungendo al suo interno più di un JavaScript: in essi è quindi possibile implementare altre funzionalità oppure implementare codice per elaborazioni complesse. Nei Laboratori di Legnaro si è deciso, dato il congruo numero di persone che lavorano al progetto, di suddividere il personale per aree di competenza, creando delle task force. Quindi il compito di elaborare le

interfacce che vanno a gestire tutta la strumentazione descritta nel Capitolo 2 è stato dato in carico al gruppo dei controlli, che ha una maggiore conoscenza della struttura dei database EPICS sviluppati fino ad ora. Si può definire quindi l'OPI come un'interfaccia grafica che rappresenta per l'utente finale il sistema di controllo vero e proprio: infatti grazie ai vari pannelli, che sono allegati in Appendice (come anche la guida su come si utilizzano), l'operatore può controllare tutta la strumentazione necessaria per il suo esperimento da remoto in modo semplice. Si è deciso, all'interno del gruppo di controllo, per facilità di manutenzione delle interfacce, che tutti i controlli complessi o meno siano realizzati ad un livello più basso, e di lasciare al BOY solo il mero compito di visualizzazione, quindi delegare agli script solo funzione di messaggi o grafiche. L'editor di OPI in BOY si può pensare come uno strumento simile a Word che implementa la struttura WYSIWYG (What You See Is What You Get). Infatti ha la maggior parte delle funzioni di un editing moderno che facilitano molto lo sviluppo OPI, come: copia, incolla, annulla, ripristina, drag and drop dei widget, righello, guida, zoom in/out, crea gruppo o link e così via. L'esecuzione delle varie OPI può essere visualizzata in schede di una stessa finestra o in finestre diverse. Tutte le interfacce che vengono sviluppate attraverso questo strumento vengono create all'interno di una cartella di lavoro, come negli attuali ambienti di sviluppo, il che significa che possono essere utilizzate in qualsiasi applicazione di Eclipse RCP o CSS semplicemente copiando il file all'interno della cartella di lavoro. Qui di seguito viene riportata l'area di lavoro che, come si può notare, è suddivisa principalmente in 6 parti:

1. tool per la navigazione delle interfacce OPI realizzate (in basso a sinistra)
2. area di lavoro per la creazione della OPI (centrale)
3. palette: strumento da cui è possibile tramite drag and drop spostare i widget nell'area di lavoro (penultimo a destra)
4. proprietà dove è possibile configurare ogni componente spostato nell'area di lavoro e l'area di lavoro stessa (ultimo a destra)
5. outline per avere una visione completa dell'interfaccia e per spostarsi rapidamente da un punto ad un altro (in basso a destra)

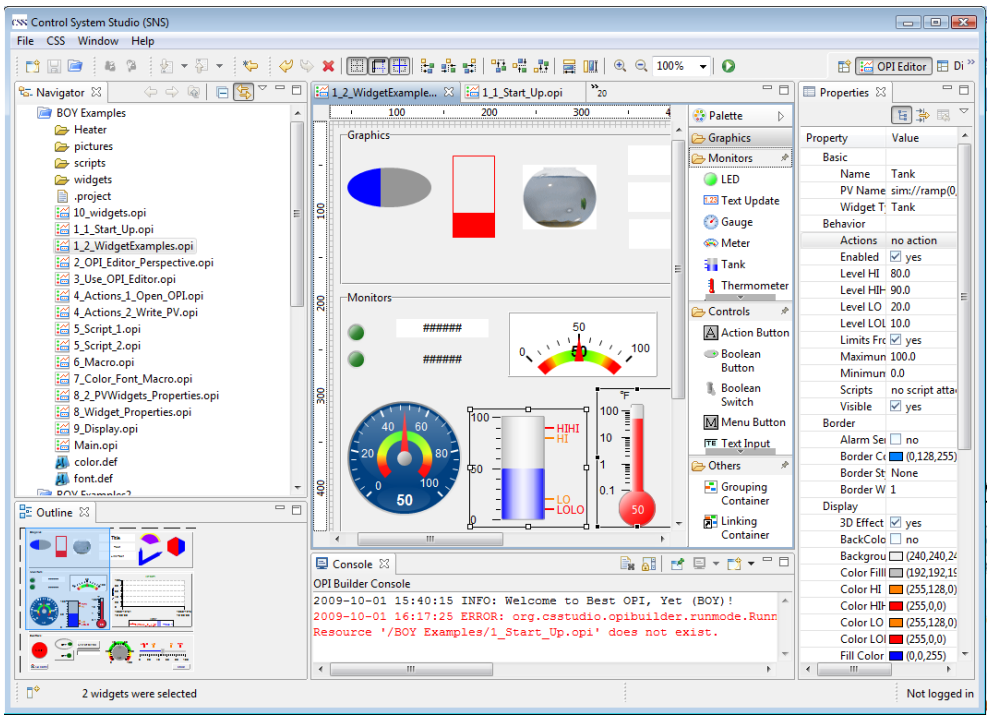

Figura 4.10: Editor degli OPI

6. console dove visualizzare i messaggi di errore o di attenzione (in basso centrale).

Utilizzo

L'utilizzo di questo applicativo è molto semplice e intuitivo, ed è già stato sommariamente spiegato nella presentazione di questo tool; per creare un'interfaccia che controlli ad esempio un alimentatore di alta tensione si devono individuare innanzitutto le variabili Epics che governano tale alimentatore e decidere quali widget utilizzare. Una volta individuate le variabili Epics ed i widget, come primo passaggio basta trascinare i widget scelti nell'area di lavoro, dopo di che andare nelle proprietà di ognuno di essi e nel campo Name inserire il nome della PV che gli si vuole associare. Per vedere in esecuzione poi l'interfaccia appena creata basta premere il pulsante verde che rappresenta il classico simbolo di Play e l'interfaccia andrà in esecuzione.

Nell'ideazione dell'interfaccia si è cercato di puntare molto sull'usabilità sia dal

punto di vista dell'operatore che anche dello sviluppatore. Il CSS infatti permette tramite l'utilizzo di Container di raggruppare widget che monitorano, comandano PV che logicamente sono associate (nel deflettore per esempio sono raggruppate per canale). In questo modo si ha un effetto visivo immediato che permette all'operatore di controllare meglio cosa sta impostando e vedendo, sia allo sviluppatore di muoversi e spostare in un movimento solo tutte le PV, quindi si consiglia vivamente di raggruppare il più possibile i vari widget per funzionalità o per apparato controllato. Un altro widget Container molto utile dal punto di vista dello sviluppatore che si consiglia di utilizzare è il link-container che permette di richiamare al suo interno un'interfaccia OPI precedentemente creata. Ad esempio come si può vedere nella figura 4.11 si è creata un'OPI chiamata Menu.opi e poi utilizzata ovunque nelle varie interfacce, agevolando così l'utente nella navigazione. Inoltre permette allo sviluppatore di cambiare in modo rapido, nel caso fosse necessario, il menu in un unico punto. Lo stesso metodo è stato usato per altre interfacce, ad esempio quelle di monitoraggio del Fascio.

Essendo come più volte indicato il CSS un derivato di Eclipse, la struttura delle cartelle e la selezione dell'area di lavoro è la medesima; si è scelto quindi una suddivisione della stessa ed una nomenclatura simile a quella usata per le classi JAVA, ovvero i nomi delle interfacce iniziano tutte con la lettera maiuscola e se sono nomi composti, ogni nome inizia con la lettera maiuscola, identicamente per i file js. All'interno quindi della workspace esiste una cartella CSS contenente le seguenti sottocartelle:

- ExampleOPI: un insieme di opi e js create dai sviluppatori del CSS che spiegano l'utilizzo dei vari widget tramite esempi (opzionale)

- OPI: contenente tutti i file .opi rappresentanti le interfacce operatore

- JavaScript: contenente tutti i file .js usati dalle varie opi

- IMG: che contiene tutte le immagini utilizzate nelle interfacce.

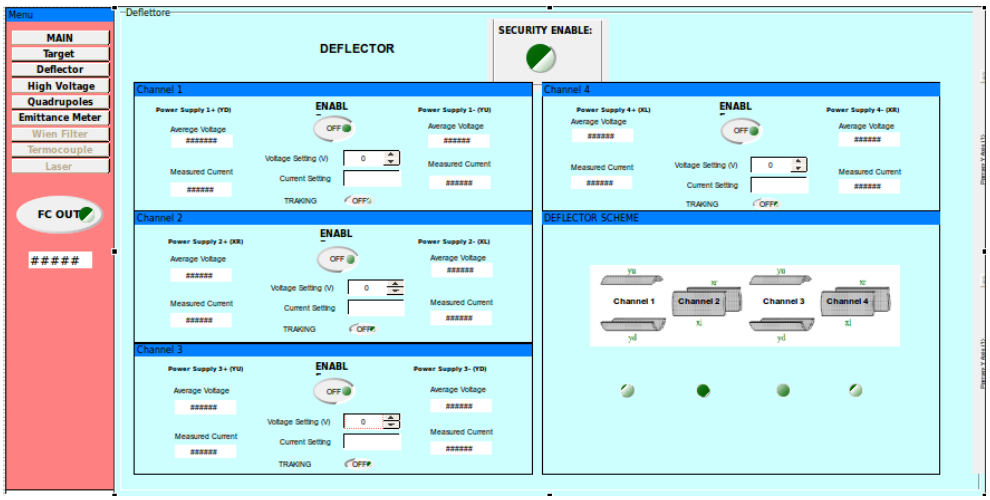

Figura 4.11: Interfaccia del deflettore, a sinistra si può osservare il menu, uguale per tutte le interfacce importato usando il link container

4.3.6 Conclusioni

In questo capitolo è stata presentata una panoramica generale dell'architettura del CSS e dei suoi principali componenti; si è potuto quindi osservare quanto questo strumento sia facile da utilizzare e da configurare se l'architettura di base è già implementata; per tali motivi si può con una certa sicurezza affermare che il CSS è un software che può essere utilizzato al 100% in fase di produzione. Dal punto di vista del programmatore, un software scritto interamente in Java e strutturato come plug-in di Eclipse permette, anche con scarsa documentazione, di capire rapidamente lo schema delle classi, dei package e le dipendenze, e quindi di iniziare in tempi brevi a sviluppare per inserire le proprie modifiche se necessario. Inoltre è bene segnalare che si tratta di un progetto open source sostenuto dalla collaborazione tra diversi laboratori e università, tra cui si citano i tre più importanti:

- DESY: Deutsches Elektronen Synchrotron
- SNS: Spallation Neutron Source
- BNL: Brookhaven National Laboratory

Altro punto di forza sta nel fatto che la comunità che si occupa dello sviluppo di questo software è molto aperta a nuove collaborazioni e quindi ad aiutare nuovi laboratori o università che fossero interessate al progetto. Prima di valutare la possibilità di interfacciarsi ad EPICS tramite CSS, presso i LNL di Padova era in uso LabView Versione 8 per sviluppare le interfacce di controllo del Front End.[7]. Lo svantaggio di questa scelta sta nel fatto che nativamente Labview versione 8 non supporta EPICS; ma questo problema è stato superato grazie al lavoro dei ricercatori dei Laboratori Nazionali di Oak Ridge che hanno sviluppato delle librerie (EpicsToLabview.* e Ca.* [8]), e appositi VI (Virtual Instrument) che hanno permesso di interfacciare Labview ad EPICS. Le interfacce così ottenute sono però risultate instabili, ovvero sono molto soggette alla velocità di risposta della rete, che in caso di malfunzionamenti costringe l'operatore a chiudere l'intera interfaccia Labview e a ripartire. Si fa notare che attualmente la versione 9 di LabView supporta nativamente EPICS e quindi si potrebbe considerare un eventuale confronto tra il CSS e LabView. In questo confronto si dovrebbe dare molto peso all'occupazione di banda della rete ed ai tempi di risposta tra scrittura e lettura di una PV, nonché alla stabilità del software. Si sottolinea anche il fatto che un confronto tra CSS e LabView è da considerarsi proponibile solo se LabView viene usato per implementare semplici interfacce senza alcun tipo di controllo ad alto livello; questa limitazione è perfettamente concorde con la scelta progettuale che demanda tutti i controlli al livello EPICS.

[7] A questo proposito si rimanda al lavoro di Scudellaro

[8] Il .* sta ad identificare l'estensione .dll ,se si usano in un sistema operativo Windows, e .so se si utilizzano in un sistema operativo UNIX

Capitolo 5

Archiviazione e gestione dei dati sperimentali

In tutti gli ambiti dove si opera con piccole o grandi moli di dati si ha la necessità di memorizzarli per un'elaborazione successiva. A differenza poi degli ambiti in cui si opera, bisogna fare uno studio approfondito per adottare la soluzione che ottimizza il più possibile costi e benefici, tenendo però presente nel calcolo non solo lo stato attuale del sistema ma anche stimando in modo non troppo approssimativo quale sarà l'ammontare/frequenza dei dati una volta che il sistema andrà in produzione. Nell'ambito sperimentale queste stime risultano molto difficili, in quanto non si parte da un impianto già in funzione sul cui contorno costruire le architetture informatiche, bensì il sistema di controllo viene creato di volta in volta con il crescere della facility; quindi stimare il numero finale di Process Variable necessarie per il totale controllo dell'impianto risulta quasi impossibile. Attualmente il Front End necessita per il suo funzionamento di circa 600 process Variable, ma questo valore è destinato ad aumentare, in quanto per ora tutta la parte della sicurezza è condotta attraverso PLC e dei programmi appositi che li gestiscono; e si è scelto che Epics possa leggere solo alcuni valori, in modo che l'operatore possa conoscerne lo stato senza dover andare nei diversi pannelli dislocati nel laboratorio a leggere tali valori, che sono di vitale importanza sia dal punto di vista della sicurezza che per il corretto funzionamento dell'esperimento. Se in seguito si decidesse, per esempio, che l'estrattore all'interno della camera

target venga comandato anche dall'interfaccia di controllo, si dovrà aggiungere sicuramente, oltre alla attuale variabile di lettura, una variabile che si interfacci con il PLC in cui si riportano i passi da far fare all'estrattore, due che corrisponderanno ad un incremento positivo e negativo fisso, una che controlli se è arrivata ad uno degli estremi e così via, per un totale di 4 o 5 Process Variable che possono aumentare a seconda di ciò che si vuole ottenere. Si pensi di fare la medesima cosa per ogni dispositivo attualmente comandato da PLC ed il numero delle Process Variable incrementerà notevolmente. Ci sono laboratori come SNS e BNL che attualmente lavorano con un numero di variabili molto elevato dell'ordine di 10000 variabili; si ricorda che una Process Variable non è composta dal solo campo valore ma da un campo che ne definisce lo stato, e uno che ne definisce il livello di allarme, ed inoltre potrebbero essere degli array di valori. Oltre al congruo numero di Process Variable da gestire un altro problema da non sottovalutare è la frequenza di aggiornamento di quest'ultime: infatti potrebbe interessare poter memorizzare variabili che cambiano il proprio valore ogni decimo di secondo. Attualmente ci sono due software che si interfacciano ad Epics attraverso il Channel Access e memorizzano le variabili: entrambi si chiamano Archive Engine; uno è sviluppato in C[1] e per la memorizzazione delle variabili utilizza un file binario; l'altro è sviluppato in Java dal SNS e utilizza per la memorizzazione dei dati o il DBMS MySQL o Oracle. Per quest'ultima soluzione nel progetto SPES si adotta come DBMS MySQL. Attualmente ai laboratori di Legnaro si stanno prendendo in considerazioni entrambi i sistemi per valutarne l'efficienza e la stabilità, ed inoltre si stanno eseguendo dei primi test, che hanno portato a buoni risultati, per integrare un altro tipo di database non relazionale al secondo tool sviluppato dagli SNS. Il database preso in esame si chiama Hypertable e la prima release stabile è uscita nel gennaio 2009 ed è stata prodotta dalla software-house Zvents Inc, i cui sponsor attualmente sono Baidu[2] e Rediff.com[3] I programmi atti alla

[1]Linguaggio di programmazione ideato nei Bell Laboratories della AT&T nel 1972 da Dennis Ritchie come evoluzione del linguaggio B.

[2]Baidu è il principale motore di ricerca in lingua cinese in grado di ricercare siti web, file audio e immagini.

[3]Rediff è uno dei principali provider al mondo di notizie, informazione, comunicazione, intrattenimento e servizio di shopping.

memorizzazione sono stati installati su un Server Rack della Dell con le seguenti caratteristiche:

- processore: Intel Xeon E5520, 2,26 GHz/8 MB, 4C, 80 W
- Memoria: RDIMM da 8 GB
- Storage: Unità SATA 9 cm (3,5) (7.200 rpm) da 250 GB usata per il sistema operativo ed i servizi, tre dischi da 1TB in RAID 5 software, quindi una capienza totale per i dati di 1,8TB (la procedura per creare un Raid 5 via software è stata riportata in appendice)
- sistema operativo CentoOS 5

Si è scelto come sistema operativo CentoOs 5 con interfaccia grafica kde, perchè il kernel si basa sulla più nota distribuzione Red Hat, quindi dà una sicurezza dal punto di vista della stabilità e della reperibilità di pacchetti stabili aggiuntivi necessari per la gestione del server. Il server è poi monitorato da un altro server in cui è installato l'applicativo Nagios[4] che controlla lo stato del RAID e quindi lo stato del server stesso.

5.1 L'ArchiveEngine classico

L'archiver è un tool che si può suddividere nel seguente modo:

- Campionamento: come si può vedere dalla figura 5.1 l'Archive Engine campiona le Process Variable o Channel da ogni Channel Access Server. I dati su come ogni campione deve essere campionato sono reperibili all'interno di un file di configurazione; si può campionare una PV ad ogni cambiamento della stessa, ogni qualvolta supera un limite massimo (questo si configura all'interno del CA), oppure utilizzando un periodo di campionamento.
  La configurazione e il funzionamento di Archive Engine ovviamente richiederà

[4]Nota applicazione open source per il monitoraggio di computer e risorse di rete. La sua funzione base è quella di controllare nodi, reti e servizi specificati, avvertendo quando questi non garantiscono il loro servizio o quando ritornano attivi.

una pianificazione, in quanto solo i dati che si è deciso di campionare ed immagazzinare saranno disponibili per un reperimento ed un'analisi futura. Quindi dovranno essere fatti alcuni compromessi ragionevoli tra il bisogno di memorizzare tutti i cambiamenti minuscoli di tutti i canali, da un lato, ed i vincoli di archiviazione dall'altro.

- Memorizzazione: i dati vengono strutturati tramite degli indici binari e dei file che li contengono. La maggior parte degli utenti finali non si dovrà preoccupare della struttura interna di questi file, e nemmeno dove essi verranno memorizzati. Tutto ciò è possibile perchè degli indici aggiuntivi permetteranno di suddividere l'archivio in più sottoarchivi facendolo però vedere all'utente finale come uno unico. Il tool però non fa nessuna attività di backup quindi sarà a cura degli utenti dover pianificare per ogni locazione i permessi di accesso ed ogni quanto fare una copia di sicurezza.

- Reperimento: questo tool mette a disposizione dei semplici sistemi di reperimento delle informazioni come anche dei semplici tool per delle comparazioni tra canali. Ma l'aspetto più importante è che mette a disposizione delle API in modo che gli utenti si possano sviluppare dei tool di analisi più sofisticati che includono anche un protocollo di rete XML-RPC per l'accesso remoto. Per esempio questo protocollo viene utilizzato dal DataBrowser del CSS.

- Log: in un file di log vengono memorizzati tutti i messaggi di monitoraggio di ArchiveEngine, ovvero messaggi di warning come per esempio bufferOverflow di ProcessVariable, scorretti Time Stamp, oppure messaggi di errore come crash dell'Archive Engine.

Come si è accennato e come si può osservare dalla figura l'ArchiveEngine è configurato tramite un file XML (vedi Figura 5.2 che riporta quali canali memorizzare e come). Per ogni canale è possibile settare varie modalità di campionamento che verranno spiegate dettagliatamente in seguito. Comunque per ogni canale viene memorizzata ogni informazione reperibile dal Channel Access: Valore, Time Stamp, e Stato ma anche informazioni di controllo tipo unità di

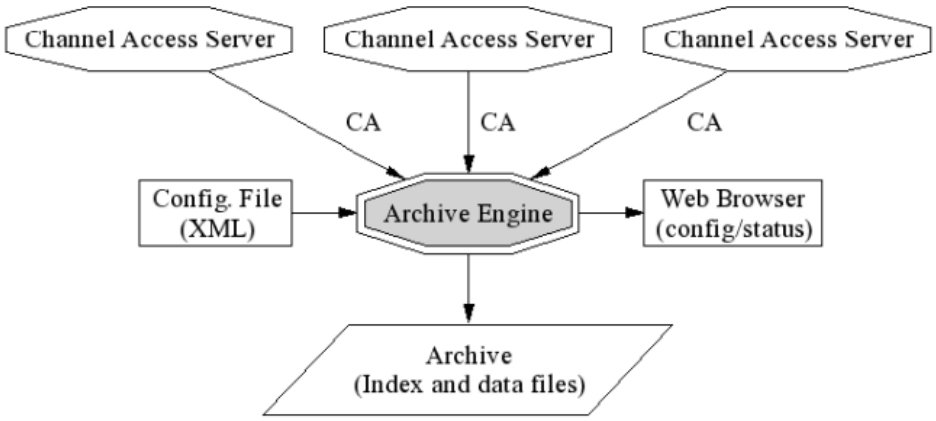

Figura 5.1: Archive Engine

misura, allarmi, limiti ecc. I dati poi vengono salvati in un archivio sotto forma di file locale. Mentre questo servizio è in esecuzione è possibile vedere tramite un web browser lo stato e la configurazione dell'Archive Engine, questo grazie ad un web server interno.

5.1.1 File XML di configurazione

Come si vede dall'immagine 5.2 c'è un intestazione che definisce la versione dell'XML utilizzato, il tipo di codifica utilizzata, e dove si trova il file che contiene il *Document type definition (DTD)* ; per maggiori informazioni riguardanti quest'ultimo si rimanda al manuale[9]. Qui di seguito invece si spiegheranno i significati dei tag utilizzati all'interno del file di configurazione più legato al funzionamento del servizio di memorizzazione (Archive Engine).

Come prima cosa si possono suddividere i tag in tre categorie, quelli globali riguardanti tutti i canali e l'Archive Engine, quelli riguardanti la suddivisione dei canali, ed infine quelli riguardanti il singolo Canale. Prima di iniziare è bene rendere chiara la differenza che c'è tra Process Variable e Canale, in quanto la differenza è minima e dipende dal contesto. Si può dire che la Process Variable è appunto quella variabile che si utilizza nel processo di controllo, quindi ci si riferisce di più al campo valore ed al suo utilizzo nel sistema di controllo, mentre

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE engineconfig SYSTEM "engineconfig.dtd">
<engineconfig>
    <write_period>30</write_period>
    <get_threshold>20</get_threshold>
    <file_size>30</file_size>
    <ignored_future>1.0</ignored_future>
    <buffer_reserve>3</buffer_reserve>
    <max_repeat_count>120</max_repeat_count>
    <group>
        <name>Vacuum</name>
        <channel><name>vac1</name>
                <period>0.1</period><monitor/>
        </channel>
        <channel><name>vac2</name>
                <period>1</period><monitor/><disable/>
        </channel>
        <channel><name>vac3</name>
                <period>2</period><scan/>
        </channel>
    </group>
    <group>
        <name>RF</name>
        <channel><name>rf1</name>
                <period>1</period><monitor/>
        </channel>
        <channel><name>rf2</name>
                <period>1</period><monitor/>
        </channel>
        <channel><name>rf3</name>
                <period>1</period><scan/>
        </channel>
    </group>
</engineconfig>
```

Figura 5.2: Esempio di file di configurazione

si parla di canale quando si vuol far riferimento a quell'insieme di informazioni che rappresentano la Process Variable all'interno del ChannelAccess:

- Time Stamp
- Status/Severity
- Value

- Meta information: Unità, Limiti, Dimensione degli array, enumerazione di stringhe ...

Parametri globali

- write_period: si tratta di un'opzione globale che deve precedere qualsiasi definizione di gruppo o canale. Si definisce l'intervallo di tempo in secondi che il motore per l'archiviazione deve far intercorrere tra le scritture.
  Per esempio il valore predefinito di 30 secondi significa che l'Engine scriverà i valori ogni 30 secondi.

- get_threshold: opzione globale che permette all'Engine di cambiare valore tra l'operazione di "Sample" e di "Sample utilizzando il monitor"; il valore di default è 20 secondi.

- file_size: opzione globale che determina la grandezza massima prima della creazione di un altro file di dati. Il valore di default 100 indica che dopo aver scritto approssimativamente 100 MB di dati, esso crea un nuovo file.

- ignore_future: opzione globale che definisce "quanto in là nel futuro" come "adesso + ignore_future", ed è espresso in ore. Questo serve ad ignorare i campioni che hanno Time Stamp al di là di questo tempo. Questo parametro serve fondamentalmente in due casi: nel caso di un riavvio dell' IOC server in quanto tutti i valori vengono ripristinati con Time Stamp zero; oppure quando per una ragione sconosciuta viene prodotto un Time Stamp improprio come ad esempio "1/02/2035". Il valore di default è 6 ore.

- buffer_reserve: opzione globale che permette di riservare per ogni canale una quantità di memoria che serve ad ammortizzare il tempo di scrittura. La taglia di questo buffer è l'intero superiore ottenuto dal seguente calcolo:

$$buffer\_reserve * \frac{write\_period}{scan\_period}$$

Poiché una scrittura può essere ritardata da altri processi in esecuzione sullo stesso computer, per esempio un'altra scrittura su disco, il buffer è più grande del minimo richiesto: di default questo valore è impostato a 3.

- max_repeat_count: quando un campione è settato per essere memorizzato dopo uno specifico intervallo (scanned mode), cioè il contrario della memorizzazione ad ogni cambiamento del Canale (monitor mode), l'Engine memorizza solo i campioni che dopo l'intervallo di tempo sono cambiati. Questa opzione definisce dopo quante volte che lo stesso valore si ripete deve essere comunque memorizzato. Questa opzione ha il fantastico pregio di limitare lo spazio occupato da una variabile. Il valore predefinito è 120, il che significa che un canale che viene sottoposto a scansione ogni 30 secondi viene scritto una volta ogni ora, anche se il suo valore non fosse cambiato.

Parametri

- group: ogni canale deve appartenere ad un gruppo, quindi una configurazione base si potrebbe pensare come un gruppo principale che contiene tutti i canali. Questo però limiterebbe le potenzialità di questo parametro, che assieme ai parametri disconnect e disable permette in un caso di risparmiare spazio in quanto abilita la registrazione dei canali solo quando effettivamente questi sono attivi, quindi permette di non memorizzare il rumore; ed in secondo luogo permette di memorizzare solo i canali quando questi sono sotto il valore 0 compreso. Il sottoparametro name definisce il nome del gruppo, ovviamente non è permesso avere gruppi con lo stesso nome, mentre è possibile avere lo stesso canale sotto gruppi diversi.

- channel: questo elemento definisce un canale fornendo il nome e le opzioni di campionamento. Il nome si definisce tramite il tag name, le opzioni di campionamento tramite questi elementi obbligatori:

  – period: definisce il periodo di campionamento. Nel caso di campionamento periodico, è il lasso di tempo che intercorre tra un tentativo di campionamento e l'altro. In caso sia abilitata l'opzione di monitor

(opzione successiva) esso è la stima di ogni quanto varia il canale. Il periodo è espresso in unità di secondi. Se un canale è elencato più di una volta, per esempio come parte di gruppi diversi, il canale verrà campionato comunque una sola volta. Il meccanismo di campionamento è determinato in modo da massimizzare la velocità dei dati. Se, ad esempio, il canale X è una volta configurato per essere campionato periodicamente ogni 30 secondi e in un altro gruppo è configurato tramite l'opzione monitor con un periodo stimato di un secondo, il canale sarà campionato con periodicità 1 secondo.

- scan: utilizza il periodo settato nella precedente opzione per effettuare il campionamento memorizzando il valore più recente del canale nell'archiver.
- monitor: il valore settato nel tag periodo viene utilizzato per calcolare il buffer da riservare in memoria per quel canale. In altre parole se la frequenza di aggiornamento di quel canale è superiore al valore stimato inserito nel tag del periodo, il motore può perdere dei valori.

5.1.2 Installazione

L'installazione di questo strumento non è facile poiché implica molte conoscenze di compilazione C e C++ sotto ambiente Unix, in quanto non funziona in ambiente Windows; inoltre necessita di librerie fornite da produttori diversi, che devono essere anch'esse compilate ed installate in uno specifico ordine per evitare errori dovuti a dipendenze. Un altro aspetto che rende questo prodotto difficile da installare è la sua dipendenza dalla versione del compilatore usata, infatti a seconda della sensibilità di quest'ultimo si dovranno fare modifiche ai sorgenti. Per i vari passaggi si rimanda alla guida di installazione del prodotto.[14]

Una volta completata l'installazione si è creato uno script che va in esecuzione ad ogni riavvio della macchina server. In fase di installazione del Raid 5 software si è preparata una posizione logica per i dati */share/archive* dove si devono posizionare tutti i dati relativi alla archiviazione dei Canali Epics. Avendo in un'unica macchina server due strumenti per l'archiviazione dati, si è deciso di sal-

vare i dati ed i relativi indici dell'archive classico a partire dal seguente percorso */share/archive/arch/*.

5.2 L'ArchineEngine con RDB

La principale differenza tra l'Archive Engine trattato precedentemente ed il seguente sta nel fatto che non si utilizzano più file contenenti i dati e file contenenti gli indici, ma si usa un database relazionale (RDB). Attualmente sono supportati MySQL e Oracle, ma la struttura utilizzata permetterebbe di implementare delle classi che supportano altri tipi di database relazionali e non. Un'altra importante differenza che si può vedere dall'immagine 5.3 è che non si utilizza più un file XML contenente i canali da memorizzare e come campionarli, ma è tutto contenuto nel Database. Per il progetto SPES si è scelto di testare questo applicativo utilizzando MySQL, perchè il numero di Process Variable è limitato, la velocità di acquisizione è sufficiente a non aver perdita di informazioni, ed infine perchè è gratuito. Quindi il paragrafo relativo alla configurazione sarà basato su MySQL.[15]

Dal momento che molti laboratori utilizzano ancora il precedente sistema, è stato creato dagli sviluppatori comunque un parse XML che permette di caricare la vecchia configurazione nel nuovo tool in modo semplice ed immediato. Un'altra cosa da non sottovalutare è lo sforzo di adottare un unico linguaggio di programmazione per sviluppare tutti gli applicativi al contorno di EPICS, infatti come detto nell'introduzione di EPICS ci sono varie OPI e altri tool per l'utilizzo di quest'ultimo, quindi se da una parte questo permette di gestire in svariati modi il sistema di controllo, dall'altra un laboratorio che volesse apportare delle modifiche ad alcuni tool dovrebbe poter disporre di operatori che conoscono e sono specializzati in vari linguaggi.

5.2.1 Installazione e Configurazione

Prerequisito necessario per l'utilizzo di questo tool è che nel server sia installato MySQL, e che sia stato configurato in modo che i database necessari per la storicizzazione dei dati siano posizionati non come nella configurazione classica di MySQL, ma nel seguente percorso *share/archive/*, che come detto in precedenza

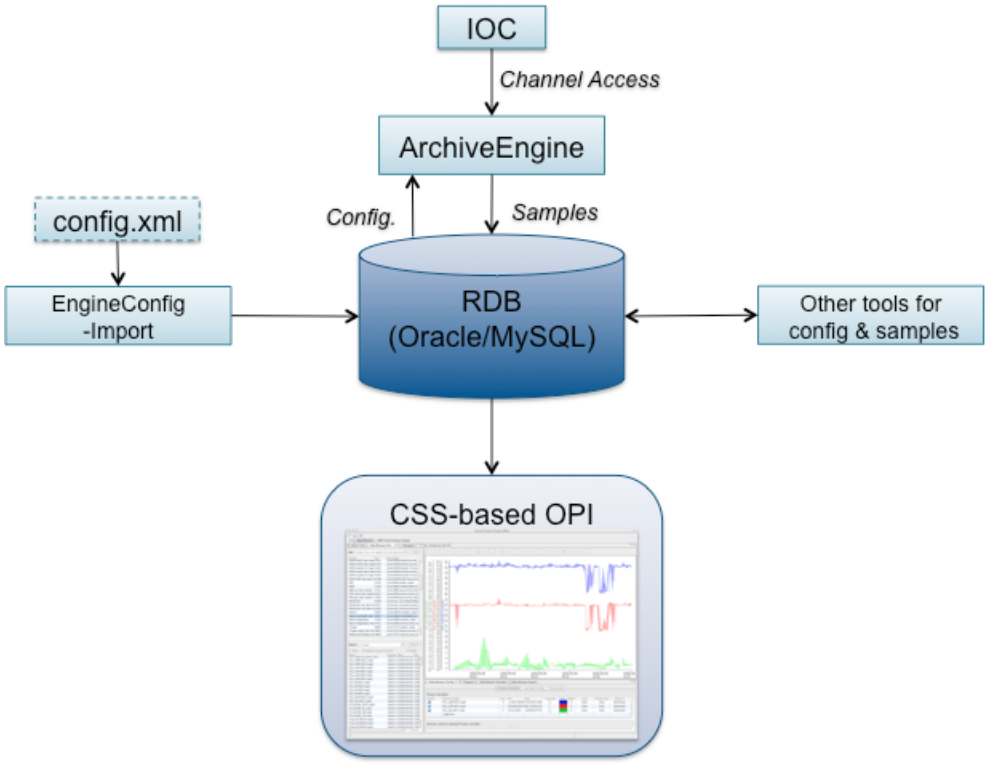

Figura 5.3: Panoramica del Archive Engine con RDB

è la cartella creata nella partizione riservata ai dati, e in RAID 5. Per fare ciò il metodo più rapido è fare l'installazione classica di MySQL, che in un sistema Unix basato sulla distribuzione Red Hat equivale ad eseguire i seguenti due comandi:

```
yum install mysql-server.i386
yum install mysql-client.i386
```

Fatto ciò si devono creare i database come verrà spiegato in seguito, quindi bisognerà spostare i database creati nel percorso *share/archive/*; questo può essere fatto in vari modi, e dipende molto dalla distribuzione Unix utilizzata, nel nostro caso basta procedere nel seguente modo:

1. copiare i database relativi ai dati, che sono archive, ALARM e log;
2. dare gli stessi permessi e controllare che il proprietario del database sia MySQL e che il gruppo di appartenenza sia anche MySQL;
3. eliminare i database MySQL dalla posizione originale;

4. creare al posto dei database appena eliminati dei link virtuali ai database reali.

Di default MySQL non prevede la necessità di fare un backup incrementale dei database, quindi per fare in modo che questo sia possibile bisogna abilitare nel suo file di configurazione la creazione di un file log binario, in questo modo MySQL all'interno di questo file memorizzerà tutte le transizioni eseguite nei vari database, quindi con degli appositi comandi poi si potrà effettuare un backup incrementale. Per il momento è stata prevista tale possibilità ed i file di log vengono salvati in */share/archive/backup/* L'abilitazione di tale file si fa aggiungendo al file *my.conf* la seguente istruzione:

```
log_bin = /share/archive/backup/mysql-bin.log
```

Per utilizzare questo applicativo, c'è bisogno di creare la struttura delle tabelle nel proprio RDB; si fa notare che l'unica differenza tra lo schema utilizzato per MySQL e per Oracle sta nella struttura della tabella sample. Questo è dovuto al fatto che Oracle nella gestione dei Time stamp include anche i nanosecondi mentre invece MySQL no, quindi per ovviare a questo problema nella struttura della tabella che memorizza i campioni è stata aggiunta una colonna "nanoseconds" per salvare quest'informazione. Inoltre il codice è performante sia che si usi MySQL che Oracle, in quanto, tramite il path che viene richiesto in input, l'archive Engine utilizzerà o il dialetto MySQL o il dialetto Oracle se necessario, altrimenti userà l'SQL standard. La struttura delle tabelle si trova all'interno del file "mysql_schema.txt" che è contenuto nel pacchetto org.csstudio.archive.rdb nella directory dbd, quindi una volta installato MySQL basterà effettuare l'accesso alla shell di MySQL e lanciare lo script con il comando *source* passando come parametro il percorso del file. Si deve però precisare che i parametri globali (write_period, max_repeat, buffer_reserve), contenuti nel file XML non vengono importati nel database ma devono essere specificati all'interno di un file che si chiama *plugin_customization.ini*, altrimenti vengono utilizzati quelli di default specificati all'interno del file *preferences.ini*. Questi due file sono dei file che in java vengono chiamati file di properties e sono strutturati come una coppia

di stringhe chiave, valore separati dal simbolo "=" come riportato nel seguente esempio:

```
# Write period in seconds
org.csstudio.archive.engine/write_period=5
# Maximum number of repeat counts for scanned channels
org.csstudio.archive.engine/max_repeats=120
# Write batch size
org.csstudio.archive.engine/batch_size=500
# Buffer reserve (N times what's ideally needed)
org.csstudio.archive.engine/buffer_reserve=1000
```

Come si può vedere non c'è un'esatta corrispondenza tra i parametri globali precedenti e quelli attuali, se non per il parametro write_period che però a differenza di prima non può essere inferiore a 5, e per il parametro max_repeats e buffer_reserve che hanno il medesimo significato. Non esiste ovviamente il parametro file_size, in quanto serviva solo a dimensionare i file, nel sistema precedente è al posto di buffer_reserve; c'è un nuovo parametro che è batch_size che indica dopo ogni quanti campioni si procede alla scrittura, e questo serve ad ottimizzare i tempi di scrittura. Sempre all'interno di questo file devono essere passate le credenziali di accesso al database,

```
# Archive RDB Access
org.csstudio.archive.rdb/url=jdbc:mysql://[host]:[port]/[database]'
     user=[user]&password=[pw]
```

dove al posto di:

- host: va messo l'indirizzo ip del server MySQL, se si decidesse che il Servizio dell'Archive Engine giri in un computer diverso da quello dove è installato, altrimenti localhost.
- port: la porta che si assegna a MySQL di default è 3306
- database: va specificato il nome del database che contiene la struttura dell'archive utilizzando lo script menzionato sopra di essa come archive

- user: l'utente con cui accedere al database
- pw: la password con cui accedere al database

All'interno dell'RDB esiste un ID unico che identifica ogni engine, ma siccome tuttora molti tool utilizzano il nome dell'engine come identificatore si consiglia di mantenere unico anche quest'ultimo, se si avesse la necessità di usarne più di uno. Ogni engine costruirà al suo interno un web server, che mostrerà lo stato del servizio e qualche funzionalità di base per controllare quali canali sono accessibili, quali hanno problemi e se ci sono sovrascritture di dati. Ogni engine viene memorizzato nell'RDB nella forma $http://<host>:<port>/main$.

5.2.2 Come caricare il file di configurazione XML

Nella sezione precedente si è parlato di come si può utilizzare il file di configurazione creato per il precedente engine anche per questo tool, ma non è stato specificato come si deve procedere. Innanzitutto si devono scaricare i sorgenti del prodotto, ed a tal proposito si rimanda all'apposita sezione dell'appendice: Installazione / Configurazione di Eclipse per lo sviluppo del CSS. All'interno della folder org.csstudio.archive.rdb c'è il file *EngineConfigImport.product* che permette di lanciare tramite Eclipse tale programma; se viene eseguito senza nessun parametro di ingresso visualizzerà un messaggio d'errore nella console, come il seguente:

```
Missing option -engine
Options:
  -help                          : show help
  -config my_config.xml          : XML Engine config file
  -rdb_url jdbc:...              : RDB URL
  -rdb_user user                 : RDB User
  -rdb_password password         : RDB Password
  -engine my_engine              : Engine Name
  -export                        : export configuration as XML
  -description 'My Engine'       : Engine Description
```

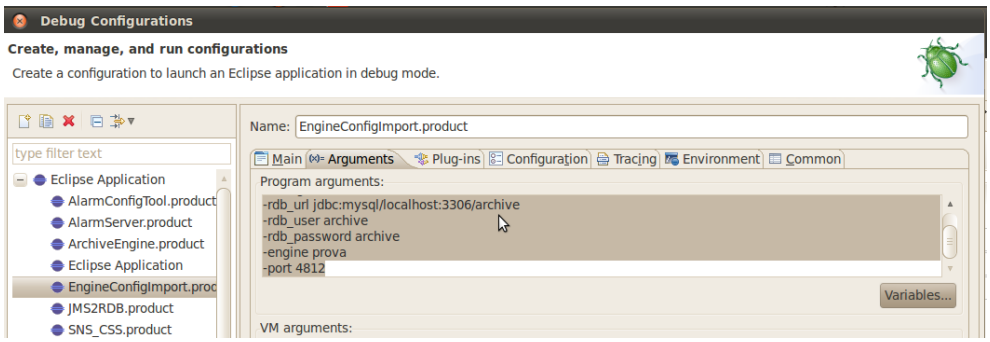

Figura 5.4: Passaggio argomenti tramite Eclipse

```
-host my.host.org              : Engine Host
-port 4812                     : Engine Port
-replace_engine                : Replace existing engine config, or stop?
-steal_channels                : Steal channels that are already in other eng
-delete_config                 : Delete existing engine config
```

Questi sono tutti gli argomenti che si possono passare al programma; se si decide di eseguire questo programma tramite Eclipse questi parametri vanno messi nella finestra di Run Configuration o Debug Configuration (dal menu → Run → Run Configuration); nella finestra che compare si sceglie la sottofinestra indicata con (x)=Arguments e nello spazio program arguments si inseriscono tutti gli argomenti, come si vede in figura: 5.4. Come si può capire dall'uscita mostrata precedentemente non tutti i parametri si possono mettere contemporaneamente perché alcuni escludono gli altri, ed il parametro -engine e -port identificano unicamente l'engine all'interno dell'RDB. Altrimenti si può decidere di eseguire il programma da linea di comando passandogli i medesimi argomenti. Non ci si sofferma troppo sui parametri perchè l'uscita in caso di nessun parametro o con parametro -help produce un output esplicativo.

5.2.3 Come eseguire l'ArchiveEngine

Per le prime fasi di test si può anche decidere di eseguirlo manualmente tramite Eclipse come si è mostrato per il programma EngineConfigImport, ma visto che

questo programma deve restare in esecuzione sempre per poter memorizzare tutte le variabili presenti nella rete EPICS, si è deciso di creare un script da eseguire ad ogni avvio della macchina. Gli argomenti da passare in fase di esecuzione sono simili a quelli precedentemente mostrati:

```
-rdb_url jdbc:...            : RDB URL
-rdb_user user                : RDB User
-rdb_password password        : RDB Password
-engine my_engine             : Engine Name
-port 4812                    : Engine Port
```

Lo script creato per il server e che si riporta qui di seguito è stato testato su CentoOs 5:

```
[fontsize=8]
DIR=/usr/local/css/ArchiveEngine
PROGRAM=ArchiveEngine

LOGFILE=$DIR/$PROGRAM.log
LOGFILE=$DIR/$PROGRAM‘date +’_%Y-%m-%d_%H:%M:%S’‘.log
PIDFILE=$DIR/$PROGRAM.pid
DATA=/tmp/$PROGRAM.$$
OPT="-port 4812 -rdb_url jdbc:mysql://localhost/archive -rdb_user archive
     -rdb_password archive -engine LNL_Engine"
unset DISPLAY
export PATH=$JAVA_HOME/bin:$PATH
case $1 in
start)
    cd $DIR
    nohup ./$PROGRAM $OPT >$LOGFILE 2>&1 &
    echo $! > $PIDFILE
    ;;
stop)
    if [ ! -f $PIDFILE ]
```

```
    then
       exit
    fi
    PID=‘cat $PIDFILE‘
    FOUND=‘ps aux | fgrep $PID | grep -v grep | grep --count $PROGRAM‘
    if [ $FOUND -eq 1 ]
    then
        kill $PID
        rm $PIDFILE
    else
        echo "Cannot locate $PROGRAM under pid $PID"
    fi
    ;;
status)
    if [ ! -f $PIDFILE ]
    then
       echo "No pid"
       exit
    fi
    PID=‘cat $PIDFILE‘
    ps aux | fgrep $PID | grep -v grep
    ;;
*)
    echo $0 "start | stop | status"
esac
```

5.3 L’Archive Engine con Hypertable

Il progetto nasce dall’idea di trovare al posto di MySQL od Oracle, database relazionali molto noti, un’alternativa valida che sia affidabile e molto sicura; infatti a seconda del tipo di attività che un’azienda si trova ad operare si possono o meno tollerare fermi di servizio per manutenzione, per aggiornamento, anche per

eventuali failure di qualche tipo; ci sono però ambiti dove niente di tutto questo è ipotizzabile, e dove serve l'affidabilità di una soluzione "fine nine": 99.999% di uptime, che significano poco più di 5 minuti di stop all'anno.
Per ottenere questa percentuale MySQL, per quanto già il prodotto base sia affidabile, ha bisogno di clusterizzazioni, replicazioni, e quant'altro. E la sua architettura è un'architettura pensata per un sistema stand alone, non è un database distribuito. Oracle invece nativamente mette a disposizione vari tool per il recovery dei dati, ed in più è un database che sin dalle prime versioni ha fatto molta attenzione alla connettibilità utilizzando:

- Processing distribuito: un'applicazione che lavora su un nodo di una rete detto client ma accede a un database che risiede su un nodo diverso detto server. Il kernel di Oracle è installato sul nodo che possiede il database. Molte applicazioni si collegano allo stesso database. La gestione del database è separata dall'esecuzione delle applicazioni.

- Database distribuito: un insieme di database fisici visti come un unico database logico. L'utente non deve sapere dove risiedono fisicamente i dati (location transparency). Ogni database fisico resta comunque autonomo rispetto agli altri (site autonomy).

Oltre all'attenzione all'affidabilità, nell'ambito degli esperimenti scientifici c'è bisogno di velocità in modo da poter memorizzare tutti i valori necessari a comandare e controllare gli apparecchi distribuiti; all' INFN-LNL per il progetto SPES si è deciso di utilizzare EPICS come middleware. Chi ha sviluppato il CSS ha implementato l'archiver in modo che sfrutti il dialetto migliore sia per MySQL che per Oracle, in modo tale da ottimizzare le performance di entrambi. Ma anche in questo caso Oracle è risultato più performante, l'unico svantaggio nel suo utilizzo sta nel fatto che esso è un database i cui costi di licenza sono considerevoli e la configurazione deve essere fatta da personale esperto per aver la maggior efficienza, altrimenti potrebbe risultare anche molto inefficiente, e dare prestazioni molto inferiori a MySQL. Quindi la scelta spesso nei vari laboratori ricade su MySQL o sull'archiver tradizionale. Visti i promettenti risultati di Hypertable condotti dal

gruppo Mauro Giacchini (INFN LNL) e Ralph Lange (BNL)[5] si è deciso di modificare l'attuale Archive Engine in modo che utilizzi Hypertable, ed in un successivo momento di modificare il CSS in modo tale che reperisca i dati storicizzati invece che su MySQL o Oracle su Hypertable e li visualizzi tramite il DataBrowser. Hypertable è un database rilasciato sotto licenza GNU che gestisce applicazioni che hanno un enorme volume di dati, è veloce nell'individuazione degli errori e dei fail-over, e si basa sull'idea di BigTable di Google. Come Hypertable ci sono altri progetti, si pensi al più noto HBase[6]; le motivazioni che hanno spinto a scegliere Hypertable è che dai risultati di Giacchini-Lange a parità di stabilità ed efficienza a livello prestazionale Hypertable è risultato più performante.

5.3.1 Performance

Le Performance sono tanto importanti quanto l'affidabilità: Hypertable permette tramite l'utilizzo di fileSystem distribuiti e soluzioni HW come RAID di ottenere una buona affidabilità con ottime prestazioni. Nel seguente lavoro si è puntato molto sul livello delle prestazioni in ambito di inserimento e reperimento, in quanto molti laboratori hanno necessità di trovare un'alternativa ad Oracle e MySQL, poiché in entrambi in casi non si riesce a storicizzare 10.000 campioni al secondo senza avere perdita di dati. Non avendo la possibilità di disporre di un DBMS Oracle, qui di seguito verranno riportati solo i confronti con MySQL. Come detto in precedenza l'obiettivo del progetto era quello di riuscire a storicizzare 10.000 campioni al secondo, e reperire 1000 PV per 4 canali distinti in meno di un secondo. Per fare ciò è stato creato un database Epics, con 1000 PV che variano ogni decimo di secondo. I dati ottenuti sono stati realizzati su un notebook con le seguenti caratteristiche:

- Sistema operativo: Ubuntu Karmic kola 9.10
- Processore: Intel Pentium M 740, 1,73GHz
- Ram: 2GB

[5] http://www.lnl.infn.it/ epics/joomla/first-test-hypertable.html

[6] HBase è una base di dati distribuita open source modellata sul BigTable di Google e scritta in Java

- Hard Disk: 5600 RPM sata Disk

Le performance ottenute su MySQL ed Hypertable sono riportate nella seguente tabella:

operazione	Hypertable	MySQL
inserimento	- 20 min - 11999173 campioni - 42K campioni/sec	- 20 min - 5725506 campioni - 4,8K campioni/sec
estrazione	- 20 min - 1328395 campioni - 8K campioni/sec	- 20 min - 1253654 campioni - 8K campioni/sec

Molto spesso è interessare vedere anche come risponde l'applicativo alle esigenze di un utente standard, quindi si è provato ad eseguire un'estrazione molto frequente da parte degli operatori, ovvero estrarre 4000 campioni per 4 PV distinte, e i risultati sono stati i seguenti:

n° campioni	time estrazione	campioni al sec
1132	30 ms	
1331	39 ms	
1332	14 ms	53,8 K
1331	16 ms	

5.3.2 Hypertable

Struttura dati

In Hypertable la struttura dati consiste in una tabella multidimensionale contenenti informazioni che possono essere richieste tramite l'uso di una singola chiave. La prima dimensione della tabella è la chiave di riga (row key). La row key è una chiave primaria e definisce l'ordine nel quale i dati della tabella sono fisicamente memorizzati. La seconda dimensione è la column family, questa dimensione presenta molte analogie con le colonne dei database tradizionali. La terza dimensione è il column qualifier, di cui ogni column family può contenerne al suo interno un

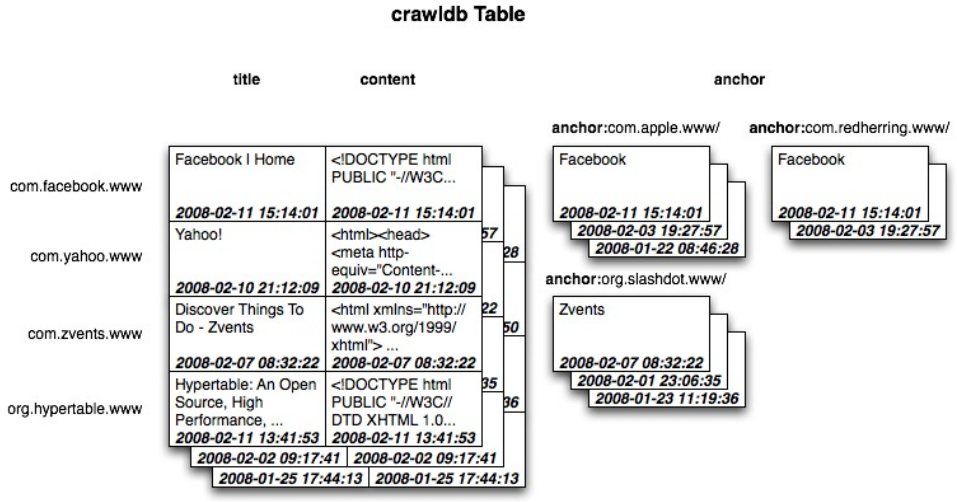

Figura 5.5: Modello dei dati Hypertable

numero infinito. Un esempio di utilizzo nel nostro caso sarebbe quello di utilizzare un'unica column family Valore e al suo interno suddividerla tramite qualificatori con le informazioni contenute nel canale, per esempio: stato, severità, unità di misura, ecc. Questa struttura non è stata adottata perche si è visto tramite i test preliminari eseguiti con le prime versioni di Hypertable che era meglio utilizzare un'unica column family. La quarta ed ultima dimensione è il tempo, questa dimensione consiste in un time stamp che è automaticamente assegnato dal sistema e rappresenta l'ora, con precisione dei nanosecondi, di inserimento della cella. Concettualmente, una tabella di Hypertable può essere pensata come un foglio di calcolo tridimensionale con una versione di time stamp per ogni cella. (Vedi Figura 5.5)

Quanto visto sino a qui è a livello molto concettuale; se si guarda al di sotto di questa astrazione, la tabella multidimensionale delle informazioni è rappresentata come una lista composta da chiave/valore. La chiave è essenzialmente la concatenazione delle quattro dimensioni (row, column family, column qualifier, e time stamp), come si può meglio capire dalla seguente figura 5.6.

Tutti i dati delle tabelle possono essere memorizzati su un file system distribuito o sul file system locale. Quello che si è utilizzato è Hadoop DFS[7], ma

[7]Hadoop è un framework software creato da Apache che supporta dati ad alta intensità

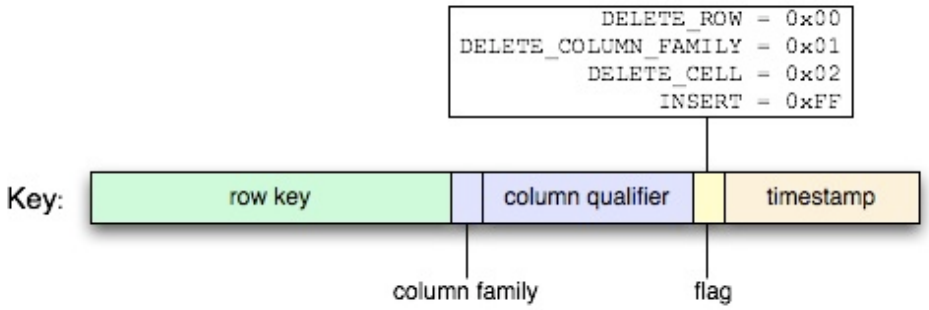

Figura 5.6: Diagramma della chiave Hypertable

è possibile utilizzare anche altri filesystem come KFS[8], oppure creare un nuovo supporto per il sistema distribuito scelto, in quanto Hypertable è strutturato in modo da essere indipendente dal file system utilizzato e mette a disposizione delle classi astratte da cui partire per sviluppare il nuovo connettore.

La coppia di dati chiave/valore è memorizzata in file chiamati CellStore; al più alto livello di astrazione questi file sono delle liste ordinate di coppie chiave valore. Fisicamente queste coppie sono storicizzate come sequenze di blocchi compressi (di approssivativamente 65Kb ognuno). Alla fine della sequenza c'è un indice il quale è una lista di "chiavi" a cui si aggiunge l'offset del corrispettivo blocco. Per ogni blocco nel file ci sarà una voce di indice che contiene l'ultima chiave del blocco assieme all'offset del blocco. Questo indice viene caricato in memoria non appena il sistema legge il file CellStore. Un esempio esplicativo di questo file è riportato in figura 5.7.

Un'altra caratteristica da non sottovalutare attuata dai progettisti di Hypertable per aumentare le performance è quella di utilizzare un sistema ibrido tra un sistema orientato per righe ed uno orientato per colonne, dove per orientato si intende come vengono fisicamente salvati i dati nel disco fisso, ovvero se si predilige salvare tutti i dati di una riga vicini o se si predilige salvare tutti i dati

e applicazioni distribuite. Consente quindi alle applicazioni di lavorare con migliaia di nodi e petabyte di dati. Hadoop è stato ispirato da MapReduce di Google e Google File System (GFS).

[8]Kosmos filesystem: è un filesystem distribuito scritto in C++ ad alte prestazioni per le applicazioni web-scale quali la memorizzazione di dati di log, MapReduce, dati, ecc. Si basa sull'idea di Google Filesystem.

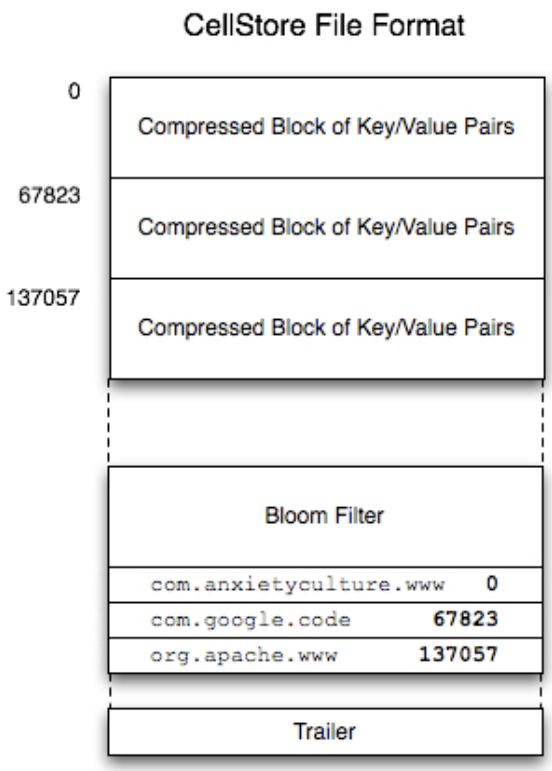

Figura 5.7: Struttura dei file di Hypertable

di una colonna vicini. Per implementare questo sistema in fase di creazione di una tabella è possibile aggiungere al comando di creazione della tabella il comando Access Group che appunto provvede a raggruppare e salvare assieme fisicamente nello Stesso Cell Store le colonne facente parti di un gruppo. Quindi se si vuole ottenere un sistema solamente orientato alle righe basta creare un unico Access Group con tutte le colonne, viceversa per ottenere un sistema orientato sulle colonne basta creare un Access Group per ogni colonna. Di default se non viene specificato nessun Access Group per la tabella si ottiene un sistema orientato per riga, ovvero viene automaticamente creato un Access Group di default con tutte le colonne all'interno. Se si utilizza invece l'Access Group ma non si raggruppano tutte le colonne, le colonne che non vengono raggruppate verranno raggruppate assieme nel gruppo di default.

```
CREATE TABLE nome_tabella (
  colonna1,
  colonna2,
  ...,
  ....,
  colonnaN,
  ACCESS GROUP nome_gruppo (colonna1,colonna2,),
  ACCESS GROUP nome_gruppo2 (colonna3,...,colonna7)
)

Questo comando è uguale al seguente :

CREATE TABLE nome_tabella (
  colonna1,
  colonna2,
  ...,
  ....,
  colonnaN,
  ACCESS GROUP nome_gruppo (colonna1,colonna2,),
  ACCESS GROUP nome_gruppo2 (colonna3,...,colonna7)
  ACCESS GROUP default (colonna7,...,colonnaN)
)
```

Struttura dei processi

Il seguente diagramma 5.8 mostra quali sono i processi presenti in Hypertable e come sono relazionati uno all'altro, in questo paragrafo si andranno a spiegarne le caratteristiche e funzionalità, e la relativa assomiglianza con il più famoso database distribuito, BigTable.[16]

- Hyperspace: questo servizio che è l'equivalente di Chubby in BigTable è un servizio che fornisce un file system per la memorizzazione di piccole quantità di metadati. Un'ulteriore sua funzione è quella di gestire gli accessi in modo

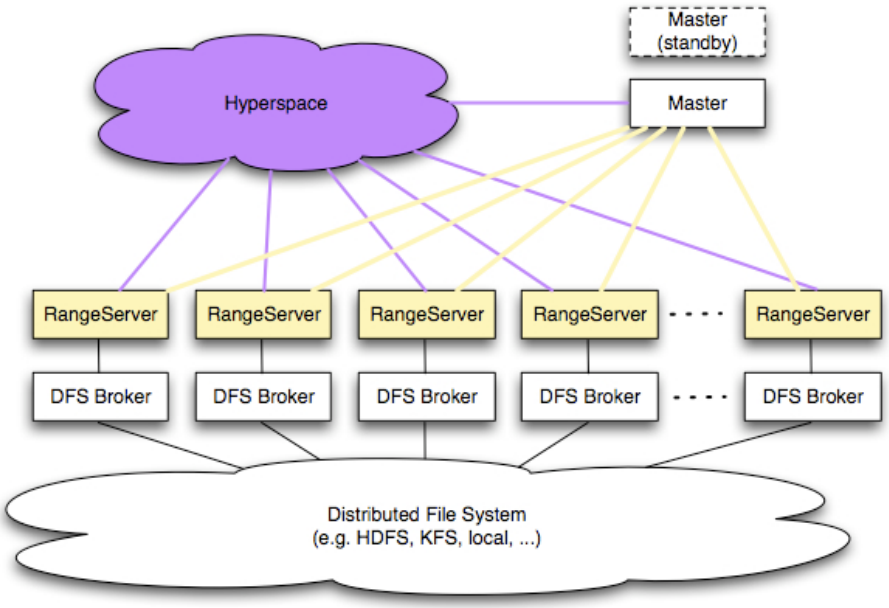

Figura 5.8: Diagramma di tutti i processi presenti in Hypertable

qualifi

condiviso o bloccante per ogni file; attualmente, a differenza della struttura messa in piedi da Google, per Hypertable è un singolo server, ma in un futuro rispecchierà tutte le funzionalità di Chubby.[9]

- Range Server: le tabelle sono suddivise in intervalli di righe contigue, e ognuno di questi blocchi è gestito da un Range Server. Inizialmente ogni tabella è composta da un singolo intervallo che comprende lo spazio dell'intera chiave. Con il riempirsi della tabella, l'insieme di righe finisce per superare una soglia massima di dimensione (di default settata a 200 MB), quindi il blocco verrà suddiviso in due intervalli, utilizzando la chiave della riga centrale come punto di rottura. La prima metà rimarrà assegnata al Range Server di prima, ed alla seconda ne verrà assegnato uno nuovo; la procedura si itera ad ogni superamento della soglia. Di conseguenza ogni Range Server gestisce tutte le letture e scritture dei dati della parte di tabella a cui è assegnato, tutte le modifiche ai dati o i nuovi inserimenti vengono

[9]Per maggiori informazioni http://labs.google.com/papers/chubby.html

fatti in un'area di memoria riservata (tenendo traccia dei commit in un file di Log), queste aree vengono chiamate CellCache. Periodicamente poi queste aree vengono salvate sul disco, in un file appositamente formattato chiamato CellStore, di cui abbiamo già spiegato la struttura. Per il reperimento di tutti i dati, il Range Server farà un merge della struttura dati in memoria e di quella storicizzata mediante un Merge Scan come riportato in figura 5.9.

- Master: gestisce tutte le operazioni strutturali, come la creazione e l'eliminazione di tabelle. Nessun dato passa attraverso questo servizio, in modo tale che se per brevi periodi di tempo dovesse non essere attivo o reperibile, i client (Range Server) non ne vengono influenzati. Il master è inoltre responsabile per la rilevazione di errori del Range Server, della riassegnazione degli intervalli se necessario, e del bilanciamento del carico per ogni Range server.

- DFS Broker: Hypertable come più volte sottolineato è progettato per girare sopra vari tipi di file System. Per fare ciò è stata creata un'interfaccia astratta chiama per l'appunto DFS Broker. Questo processo non fa altro che tradurre le varie chiamate standard al file di sistema nelle chiamate specifiche per il sistema selezionato (HDFS, ovvero hadoop, KFS, e Unix/local).

Installazione e Configurazione

Gli sviluppatori di Hypertable mettono a disposizione dei pacchetti autoinstallanti per le varie distribuzioni Unix in commercio, quindi a seconda del sistema operativo basta lanciare l'apposito comando che nel caso in questione è:

```
rpm -ivh hypertable-<versione>.rpm
```

Per le prime fasi di test non sono state necessarie configurazioni particolari, anche perchè si è deciso provvisoriamente di testare le performance utilizzando come file system il file system della macchina in cui è stato installata Hypertable, e non si è sfruttata la possibilità di rendere Hypertable distribuito su più macchine, per

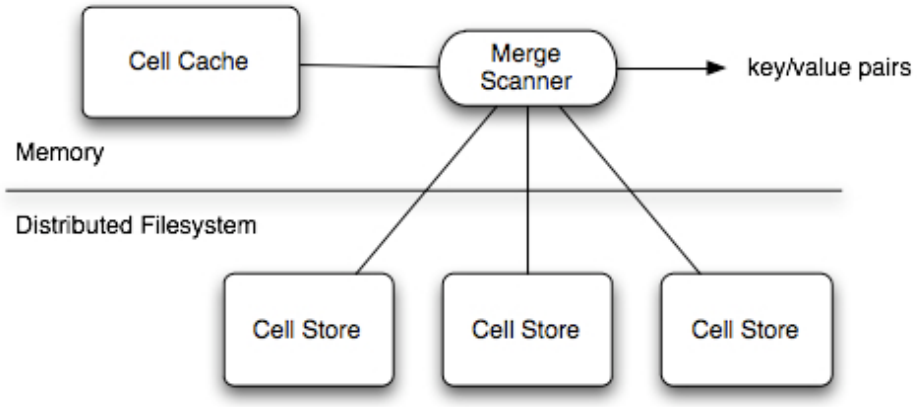

Figura 5.9: Funzionamento Merge Scan

evitare di perdere l'obiettivo finale, ovvero quello di aggiungere il supporto al CSS di questo database ed ottenere le stesse prestazioni rilevate da Giacchini-Lange nei loro test. Una volta installato il prodotto si deve procedere con l'avvio dei servizi, ed il comando è il seguente:

```
/opt/hypertable/current/bin/start-all-servers.sh local
```

dove il parametro *local* serve a indicare che il file System che si vuole utilizzare è quello della macchina, gli altri attualmente supportati sono KFS e hadoop. Fatti partire i servizi necessari al funzionamento del prodotto, si deve accedere alla shell di Hypertable per creare la struttura della tabella; i comandi da eseguire sono i seguenti:

```
/opt/hypertable/current/bin/ht shell (per accedere alla shell)

Welcome to the hypertable command interpreter.
For information about Hypertable, visit http://www.hypertable.org/

Type 'help' for a list of commands, or 'help shell' for a
list of shell meta commands.

hypertable> create table archive(pv); (per creare la tabella)
```

```
hypertable> exit (per uscire dalla shell)
```

Si fa notare che questo prodotto non ha adottato lo standard query language, ma ne ha creato uno ad hoc per migliorarne le prestazioni, inoltre non tutte le query che si possono fare con un un normale rdb sono accettate, ad esempio non è possibile ordinare e raggruppare. La struttura con un'unica colonna è stata scelta in quanto nelle prime versioni utilizzate di Hypertable è risultato più performante in fase di inserzione e di reperimento adottare questo sistema. In una fase successiva si potrebbe pensare di adottare una struttura più complessa e studiarne le prestazioni. In fase di creazione della tabella con più colonne è possibile, per ottenere migliori prestazioni in caso si utilizzino file system distribuiti, l'opzione ACCESS GROUP <nomeGruppo> (nomecolonna1,nomecolonna2) che permette di memorizzare tutte le colonne dello stesso gruppo fisicamente vicine nel disco. Nel caso in cui si cambia la struttura della tabella o il suo nome, si deve automaticamente andare a cambiare il codice java per la connessione con Hypertable in modo tale che i dati vengano scritti nelle apposite colonne. Altrimenti il programma scriverà i dati tutti nella colonna PV e se essa non esiste lancerà un errore. Sono state già prodotte delle classi di prova per gestire più colonne che si possono trovare nel pacchetto org.csstudio.archive.htrdb.

5.4 Reperimento dei dati sperimentali

Il reperimento dei dati sperimentali in Hypertable allo stato attuale dei test risulta trasparente all'utente finale, in quanto per ora tutta la configurazione è rimasta su MySQL, quindi l'utente utilizza il DataBrowser come se stesse utilizzando MySQL. La procedura da seguire per fare in modo che ciò accada è sostituire l'attuale plug-in utilizzato per la connessione con Oracle e MySQL con quello creato per accedere a Hypertable, e ricreare l'eseguibile del CSS. Il metodo per fare questa sostituzione è riportato in Appendice nella sezione Installazione/Configurazione di Eclipse per lo sviluppo del CSS.

5.5 Gestione degli Allarmi

In un progetto sperimentale come SPES o in un ambiente industriale potrebbe esserci la necessità di dover avvisare l'operatore che la soglia massima stimata per una determinata apparecchiatura sta per essere raggiunta, oppure che il sistema automaticamente intraprenda un'azione a seconda del livello di allarme. Non serve quindi solo storicizzare quello che è successo ma bisogna anche fare in modo che queste informazioni siano utili. Tutti i widget del CSS permettono tramite il loro bordo di colori diversi a seconda dell'intensità dell'allarme di avvisare visivamente l'operatore ed in caso tramite degli script associati a questo il programmatore può decidere di far intraprendere altre azioni, ma come si è detto si preferisce lasciare questi tipi di controllo a livello del database Epics.
Molto spesso però ci sono apparecchiature che non possono avere sempre un operatore presente che ne controlli lo stato, ma al massimo possono avere un operatore disponibile pronto ad intervenire quando gli arriva notifica di un malfunzionamento. Come più volte si è detto il CSS nasce dall'idea di unificare in un unico prodotto tutti i tool già presenti in EPICS, ecco quindi che BEAST (Best Ever Alarm System Toolkit) va a sostituire il tool più rappresentativo, ALH (Alarm Handler), tra quelli che gestiscono gli allarmi in Epics. Questo sistema si basa, come il già citato MEDM per creare le interfacce, sulle librerie motif il cui supporto sta sparendo per gli ambienti Unix. Comunque sempre per questioni di affidabilità si sono provati entrambi gli strumenti, sia ALH che BEAST; la documentazione relativa ad ALH è molto chiara e esplicativa, sia dal punto di vista dell'utente che dell'amministratore che deve fornire un tool configurato all'utente; non è così per BEAST che risulta molto più intuitivo da parte dell'utilizzatore, quindi non ha bisogno di un manuale per interagire con i tool di cui è composto, "Alarm Tree" ed "Alarm Table". Invece è carente nei riguardi dell'amministratore che deve fare uno sforzo maggiore per capire quali sono i pacchetti necessari per il corretto funzionamento. Un altro punto a sfavore per BEAST come si vedrà nel successivo capitolo sta nel fatto che per funzionare ha bisogno di più programmi per la gestione dei messaggi a differenza di ALH che è un sistema stand alone che si interfaccia direttamente al Channel Access e memorizza lo stato degli allarmi

su un file di testo ed esegue delle operazioni salvate su un file XML di configurazione. La prima funzione è disponibile solo se ALH è compilato con il supporto CMLOG. Dopo aver configurato entrambi i sistemi, e visto il loro funzionamento, si è scartato ALH per due ragioni principali: il problema del supporto della grafica in versioni future di Linux, e la gestione del file di memorizzazione delle variabili, non condivisibile con altre installazioni di ALH. Inoltre BEAST mette a disposizione più funzionalità ed è integrato con tutti gli altri tool del CSS, quindi l'operatore può portare la variabile di interesse da una schermata ad un'altra con dei semplici passaggi del mouse. Ma proprio come ALH, anch'esso non memorizza tutti i cambiamenti di stato della variabile ma solo l'ultimo cambiamento, per avere uno storico degli allarmi bisogna anche utilizzare il pacchetto JMS2RDB che è un servizio che memorizza tutti i messaggi derivanti dal Channel Access tra cui anche gli allarmi.

5.5.1 BEAST - Best Ever Alarm System Toolkit

Questo tool è formato da un Alarm server che riceve gli allarmi dall' IOC, tramite il Channel Access. Come si può vedere dalla figura 5.10 a seconda della configurazione è possibile ottenere un messaggio sonoro, quindi cogliere o ignorare l'allarme in corso. L'interfaccia permette agli utenti di vedere gli allarmi correnti, quindi verificarli, e intraprendere un'operazione relativa a quell'allarme; oltre a questo può navigare nello storico degli allarmi o cambiarne la configurazione.[17] Nell'ambito del Progetto SPES questo tool non è stato utilizzato nel pieno della sua potenzialità, ma si è preferita una configurazione parziale, che comprende:

- JMS server, per l'interfaccia grafica, e Alarm server per la comunicazione;
- RDB, dove è stata memorizzata la configurazione dell'Alarm server;
- CSS, con le interfacce grafiche per gli allarmi, Alarm tree e Alarm table;
- Alarm server;
- JMS Monitoring plug-in per una visione grezza dei messaggi catturati dal tool, per il debug del setup;

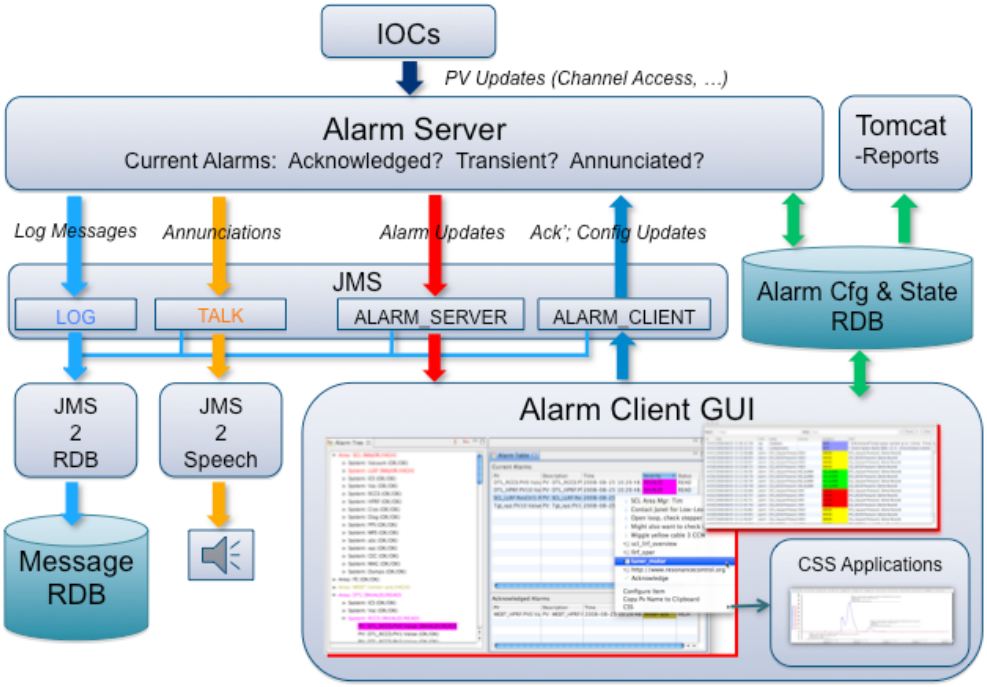

Figura 5.10: Struttura di Best Ever Alarm System Toolkit

- AlarmConfigTool, per l'esportazione e l'importazione della configurazione degli allarmi come XML e per convertire degli esistenti file di configurazione fatti per ALH;
- JMS2RDB, per un sistema di monitoraggio dell'attività degli allarmi.

Installazione

Si è deciso per il momento di utilizzare sempre la stessa macchina usata per l'Archive Engine anche per storicizzare i log e gli allarmi. Come si è accennato prima nel database relativo agli allarmi si memorizzano solo gli ultimi due stati dell'allarme che quindi ha un'occupazione di pochi MB a seconda del numero di canali monitorati, mentre più preoccupante è la quantità di dati rappresentati dai messaggi di log che potrebbero arrivare anche a GB a seconda di che messaggi si vuole tenere traccia; non si è potuto fare una stima reale che quantificasse realmente l'occupazione, in quanto il sistema è stato in funzione per poche ore, e non si è trovata quindi ancora una configurazione appropriata. Similarmente

all'Archive Engine all'interno del pacchetto org.csstudio.alarm nella cartella "db-d" si può trovare lo script MySQL per creare la struttura necessaria all'alarm server per funzionare. Se si vuole in un futuro avere anche uno storico degli allarmi si può procedere anche all'installazione del servizio JMS2RDB altrimenti questa fase può essere saltata, anche in questo caso la struttura del database si trova all'interno del relativo pacchetto, org.cssstudio.sns.JMS2RDB nella cartella "dbd"; si fa inoltre notare che per il funzionamento di quest'ultimo servizio si ha bisogno anche dell'installazione di un messenger broker(JMS Server)[10]. Quindi si caricano le strutture nel DBMS con il semplice comando *source <path script>* da dare all'interno della "shell" di MySQL. Una volta creata la struttura nel proprio DBMS si può procedere con la creazione degli eseguibili dei suddetti pacchetti; per capire come si creano si rimanda all'Appendice *Come creare l'eseguibile del CSS*, la proceduta è la medesima. Si è deciso di installare tutti questi server nella cartella usr/local/css/ in modo tale da avere un'unica posizione con tutti i file di log dei server per una più facile manutenzione. Per installare il messenger broker ActiveQM basta scaricare il file e scompattarlo nella directory menzionata sopra e generare vari script batch per l'avvio automatico dei servizi in caso di riavvio della macchina.

Configurazione

La documentazione, come accennato, è poca e non molto chiara soprattutto per la parte di configurazione ed installazione. La configurazione come per l'Archive Engine RDB deve essere salvata sul database, e si può procedere in tre modi:

- inserendo tramite insert nel db le varie pv;
- creare un file xml e poi importarlo nella base di dati tramite AlarmConfigTool;
- usare AlarmConfigTool per convertire il file di configurazione creato per ALH in un XML adatto ad essere importato.

[10]Nel caso specifico si è usato ActiveQM che implementa totalmente il java messenger service JMS

Il primo modo è più adatto quando si ha già il sistema configurato, e si ha la sola necessità di aggiungere al monitoraggio una nuova PV al suo gruppo di appartenenza; altrimenti si pensi a quanto sarebbe scomodo scrivere per ogni nuova PV una query di inserimento come nell'esempio precedente, facendo attenzione a mantenere una struttura gerarchica. Nell'esempio infatti si è creato un gruppo principale che è LNL_ALARM a cui si sono aggiunti due sottogruppi Magnet e Oven i quali hanno al loro interno le PV, si pensi fare la stessa cosa quando si vogliono monitorare 200 o più variabili.

```
INSERT INTO
  ALARM_TREE(COMPONENT_ID, PARENT_CMPNT_, NAME, CONFIG_TIME)
  VALUES(1, NULL, 'LNL_ALARM', NOW());

INSERT INTO
  ALARM_TREE(COMPONENT_ID, PARENT_CMPNT_, NAME, CONFIG_TIME)
  VALUES(2, 1 , 'Magnet', NOW());

INSERT INTO
  ALARM_TREE(COMPONENT_ID, PARENT_CMPNT_, NAME, CONFIG_TIME)
  VALUES(3, 1 , 'Oven', NOW());

INSERT INTO
  ALARM_TREE(COMPONENT_ID, PARENT_CMPNT_, NAME, CONFIG_TIME)
  VALUES(4, 2 , 'TaBtraMagn_Hcps:Volt', NOW());

INSERT INTO
  ALARM_TREE(COMPONENT_ID, PARENT_CMPNT_, NAME, CONFIG_TIME)
  VALUES(5, 3 , 'TaIoniOven_Hcps:Curr', NOW());
```

Con il secondo metodo, riuscire a dare la struttura specificata sopra è molto più semplice ed immediata, soprattutto con i tool moderni per la gestione degli XML, l'unica difficoltà è sapere qual è la struttura da passare all'AlarmConfigurationTool; per scoprirla si è utilizzato il terzo passo, ovvero dato che lo scopo del

progetto era testare anche le performance di ALH, si è sfruttato il fatto di essersi creati il file di configurazione, che è stato convertito attraverso l'AlarmConfigurationTool. Aver costruito il file di configurazione per ALH, seguendone l'accurata guida, ha reso molto più semplice la comprensione dell'applicativo BEAST che invece ne era sprovvisto.

```
 GROUP    NULL LNL_ALARM
$GUIDANCE
Line 1 of guidance
$END
GROUP LNL_ALARM Magnet
CHANNEL Magnet TaBtraMagn_Hcps:Volt
CHANNEL Magnet TaBtraMagn_Hcps:Curr
GROUP LNL_ALARM Oven
CHANNEL Oven TaIoniOven_Hcps:Curr
CHANNEL Oven TaIoniOven_Hcps:Volt
```

La struttura risultante è la seguente, eseguendo il tool citato :

```
<!-- Generated from ALH config file
 -   File name: '/home/loris/Scrivania/AllarmiRudolf.alhConfig'
 -->


<config name="LNL_ALARM">
    <component name="Magnet">
        <pv name="TaBtraMagn_Hcps:Volt">
            <description></description>
            <latching>true</latching>
        </pv>
        <pv name="TaBtraMagn_Hcps:Curr">
            <description></description>
            <latching>true</latching>
        </pv>
    </component>
```

```
    <component name="Oven">
        <pv name="TaIoniOven_Hcps:Curr">
            <description></description>
            <latching>true</latching>
        </pv>
        <pv name="TaIoniOven_Hcps:Volt">
            <description></description>
            <latching>true</latching>
        </pv>
    </component>
</config>
```

Conclusioni

Nell'ambito del progetto SPES sono stati testati inizialmente molti sistemi di interconnessione con EPICS, middleware utilizzato per la creazione del sistema di controllo e comunicazione con gli eterogenei sistemi hardware presenti. Si è passati da varie versioni di LabView[13] a delle interfacce con MEDM[10] fino al CSS presentato in questo elaborato. Una parte del mio lavoro presso INFN-LNL è stata dedicata a testare questo nuovo prodotto, uscito da poco e open source, quindi a configurarlo in modo che comunicasse con i vari server IOC, fino a svilupparci delle interfacce operatore semplici ed efficaci, non macchinose da comprendere, controllando man mano il comportamento e l'effettiva efficienza.
Il prodotto, una volta configurato in modo ottimale, è riuscito a rispondere a pieno alle esigenze del laboratorio, infatti ogni interfaccia abilita in lettura/scrittura le sole PV visualizzate, senza quindi provocare sovraccarico di rete con mille richieste di lettura a PV non interessanti in quel momento per l'operatore. Nelle fasi di test si è anche creata un'interfaccia unica con tutte le variabili utilizzate in SPES, circa trecento, con ottimi risultati. Un ulteriore punto di forza del prodotto, che ho potuto osservare, sta nel supporto dato dai tre principali laboratori: DESY, SNS, BNL che hanno maggiormente sviluppato questo pacchetto.
Questo nuovo prodotto ha la principale e voluta caratteristica di essere basato su un potente IDE come Eclipse, infatti il CSS è un insieme di plugin per quest'ultimo. La parte centrale del lavoro si concentra sulla ricerca di una base di dati in grado di sostituirsi a MySQL e ORACLE. L'alternativa proposta è stata quella di utilizzare Hypertable, un database non relazionale ma che ha una struttura dati pensata per la memorizzazione e per il reperimento di enormi moli di dati in modo rapido. Gli obiettivi erano di riuscire a storicizzare 10000 campioni al sec-

ondo e reperire 1000 campioni per 4 PV in meno di un secondo, e come si è visto nel capitolo relativo, si è riusciti a superare di gran lunga tale obiettivo. Questi risultati hanno interessato anche la comunità internazionale di EPICS presente al convegno tenutosi in Francia ad Aix en Provence dal 2 al 4 giugno 2011, organizzato per l'occasione dal Progetto Internazionale ITER[11]. Il contributo presentato da me e Mauro Giacchini in quell'occasione è reperibile all'indirizzo http://www-arch.iter.org/sites/epics2010/slides/. Lo stesso progetto è stato successivamente ripresentato da Giacchini al convegno tenutosi a Long Island, New York (presso BNL) dall'11 al 15 ottobre 2010. Attualmente l'HyperArchiver è in uso presso i laboratori di Legnaro ed è sotto test presso i Laboratori ESS Bilbao. [12].

[11]ITER: http://www.iter.org/.

[12]ESS Bilbao: http://essbilbao.org:8080/ESSBilbao/en

Appendice A

Documentazione varia prodotta per i Laboratori di Legnaro

Tutti i documenti operativi, relativi al mio lavoro e di tutti gli studenti che prima di me si sono occupati sino ad ora della parte relativa alla realizzazione del sistema di controllo del Progetto SPES, si trovano per una maggiore fruibilità sul server web interno del Progetto SPES, (www.epicsdev1.lnl.infn.it/wiki). Questi documenti sono stati redatti in lingua inglese perchè l'obiettivo è che essi possano essere facilmente resi disponibili anche alla comunità EPICS, se necessario, e perchè all'interno del progetto SPES lavorano persone di diverse nazionalità. Inoltre avere un sito web dove si riportano tutti i manuali operativi favorisce una maggior condivisione, distribuzione delle conoscenze; e in un contesto lavorativo dove il cambio di personale è notevole è importante per la trasmissione delle competenze acquisite.

A.1 Manuale d'uso interfacce

Per avviare l'interfaccia del Front End fare doppio click sull'icona rappresentata in figura A.1 che si trova sul desktop della postazione di controllo.

1. In alto a sinistra si trovano i pulsanti per accedere ai vari pannelli:
   (a) "Target" (per la gestione degli alimentatori HeatLine-OvenMagnet-Anode)

Figura A.1: Icona programma CSS

(b) "Deflector" (per la gestione degli alimentatori che comandano la direzione del fascio)

(c) "Quadrupoles" (per la gestione degli alimentatori che comandano il fuoco del fascio)

(d) "High Voltage" (per l'impostazione dell'alta tensione nel Front End)

(e) "Termocouples"

2. Prima di iniziare qualsiasi esperimento o test controllare i livelli di vuoto e la posizione dell'estrattore:
   Sul pannello denominato MAIN leggere i corrispondenti valori in basso a sinistra nel pannello riportato in figura A.2

3. Impostare il valore di alta tensione:

   (a) Sul pannello MAIN premere il pulsante High Voltage

   (b) impostare lo switch in posizione ON

   (c) nel campo Voltage Setting inserire il valore di tensione (doppio click con il mouse sul campo corrispondente digitare la cifra e premere INVIO oppure usare le frecce a lato del campo che hanno un passo di 10 V in questo caso l'invio non serve)

4. Interfaccia Target si presenta a schede come riportato in figura A.3:
   per attivare Heat Line Magnet Oven Anode cliccare con il mouse sopra il nome del generatore che si vuole comandare.

5. Attivare Heat:

   (a) premere target sul MAIN e selezionare la scheda Heat

   (b) cliccare sul pulsante ENABLE in basso a destra

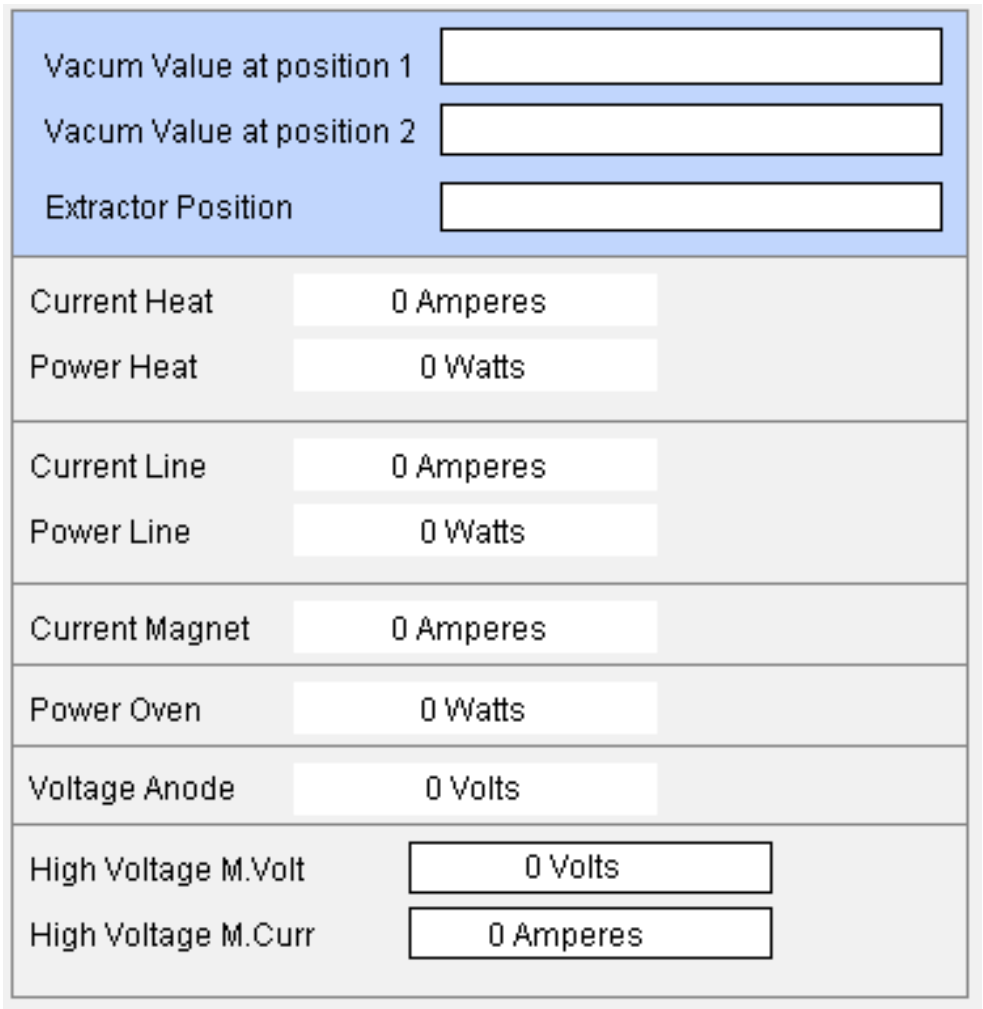

Figura A.2: Pannello con i parametri di Vuoto

(c) nel campo Maximum Voltage inserire il massimo valore di tensione, premere INVIO e cliccare sul pulsante SET a fianco (max 10 Volt dopo di che il campo si azzera)

(d) nel campo Time Rise inserire l'intervallo di tempo (in secondi) in cui avverrà la graduale variazione di corrente dal valore attuale al valore voluto e premere INVIO oppure usare le frecce che hanno un passo di 30 sec

(e) nel campo Set Current inserire il valore di corrente desiderato e premere INVIO oppure usare le frecce (hanno passo 1 A) max 1.250 A per valori superiori viene inserito sempre 1.250

Figura A.3: L'immagine rappresenta la parte superiore del pannello Target, suddivisa in pannelli

(f) cliccare il pulsante SET a fianco del campo Set Current. Durante la fase di set compare la scritta SET IN PROGRESS in Giallo

(g) attendere che la corrente si porti al valore desiderato, la scritta SET IN PROGRESS si spegne

6. Attivare Line:

(a) premere target sul MAIN e selezionare la scheda Line

(b) cliccare sul pulsante ENABLE in basso a destra

(c) nel campo Maximum Voltage inserire il massimo valore di tensione, premere INVIO e cliccare sul pulsante SET a fianco (max 30 Volt dopo di che il campo si azzera)

(d) nel campo Time Rise inserire l'intervallo di tempo (in secondi) in cui avverrà la graduale variazione di corrente dal valore attuale al valore voluto e premere INVIO oppure usare le frecce che hanno un passo di 30 sec

(e) nel campo Set Current inserire il valore di corrente desiderato e premere INVIO oppure usare le frecce (hanno passo 1 A)

(f) cliccare il pulsante SET a fianco del campo Set Current. Durante la fase di set compare la scritta SET IN PROGRESS in Giallo

(g) attendere che la corrente si porti al valore desiderato, la scritta SET IN PROGRESS si spegne

7. Attivare Magnet:
   (a) premere target sul MAIN e selezionare la scheda Magnet
   (b) cliccare sul pulsante ENABLE in alto a sinistra
   (c) nel campo Maximum Voltage inserire il massimo valore di tensione, premere INVIO e cliccare sul pulsante SET a fianco (max 300 V)
   (d) nel campo Set Current inserire il valore di corrente desiderato e premere INVIO oppure usare le frecce max 5 A
   (e) cliccare il pulsante SET a fianco del campo Set Current
   (f) attendere che la tensione si porti al valore desiderato
8. Attivare l'Oven:
   (a) premere target sul MAIN e selezionare la scheda Oven
   (b) cliccare sul pulsante ENABLE in basso a destra
   (c) nel campo Maximum Voltage inserire il massimo valore di tensione, premere INVIO e cliccare sul pulsante SET a fianco
   (d) nel campo Set Current inserire il valore di corrente desiderato e premere INVIO oppure usare le frecce (hanno passo 1 A) max 50 A
   (e) cliccare il pulsante SET a fianco del campo Set Current. Durante la fase di set compare la scritta SET IN PROGRESS in Giallo
   (f) attendere che la corrente si porti al valore desiderato, la scritta SET IN PROGRESS si spegne
9. Attivare Anode:
   (a) premere target sul MAIN e selezionare la scheda Anode
   (b) cliccare sul pulsante ENABLE in alto a sinistra
   (c) nel campo Maximum Voltage inserire il massimo valore di tensione, premere INVIO e cliccare sul pulsante SET a fianco (max 30 V)
   (d) nel campo Set Current inserire il valore di corrente desiderato e premere INVIO oppure usare le frecce max 110 A

(e) cliccare il pulsante SET a fianco del campo Set Current

(f) attendere che la corrente si porti al valore desiderato

10. Inserire la Faraday Cup e il B-Profile:

(a) sul MAIN selezionare uno dei pulsanti TARGET, QUADRUPOLES, DEFLECTORS o HIGH VOLTAGE a piacimento, per ognuna di questi pannelli comparirà la schermata per il controllo e monitoraggio della Faraday Cup e per il B-Profiler

(b) nello schermo di destra in alto a destra cliccare il pulsante ovale FC OUT per inserire la Faraday Cup

(c) sempre nello schermo di destra in alto a sinistra cliccare il pulsante rettangolare SET BP IN per inserire il B-Profile

11. Impostare la tensione dei quadrupoli:

(a) sul MAIN selezionare QUADRUPOLES

(b) cliccare il pulsante ovale ENABLE del quadrupolo che si desidera attivare

(c) nel campo Voltage Setting inserire il valore di tensione che si intende applicare al quadrupolo e premere INVIO

(d) cliccare sul pulsante di attivazione del Tracking per permettere al sistema di avere il minimo scarto tra i due valori di tensione misurati sul quadrupolo in esame; una volta che i due valori differiscono per meno di 4 Volt, disattivare il Tracking per non avere oscillazioni di tensione

12. Impostare la tensione dei deflettori:

(a) sul MAIN selezionare DEFLECTORS

(b) cliccare il pulsante ovale ENABLE del deflettore che si desidera attivare

(c) nel campo Voltage Setting inserire il valore di tensione che si intende applicare al canale deflettore (doppio click con il mouse sul campo corrispondente digitare la cifra e premere INVIO oppure usare le frecce

a lato del campo che hanno un passo di 10 V in questo caso l'invio non serve)

(d) cliccare sul pulsante di attivazione del Tracking per permettere al sistema di avere il minimo scarto tra i due valori di tensione misurati sul deflettore in esame; una volta che i due valori differiscono per meno di 4 Volt, disattivare il Tracking per non avere oscillazioni di tensione; è importante ricordare che i deflettori vanno sempre attivati in coppia con valori uguali di tensione

A.2 Installazione / Configurazione di Eclipse per lo sviluppo del CSS

Qui di seguito viene riportata una breve guida su che versione di Eclipse bisogna installare per poter iniziare a crearsi l'ambiente di lavoro atto a sviluppare nuovi plug-in per il CSS o per creare una versione del CSS con i soli moduli necessari per il Progetto SPES. La specificità della guida è stata redatta in base alle conoscenze riscontrate all'interno dei Laboratori, non è una guida all'utilizzo di Eclipse per la quale si rimanda ai manuali in commercio.

A.2.1 Installazione

1. Installare JDK 6 o superiore
2. Scaricare l'ultima versione di Eclipse per RCP/Plug-in Developers (attualmente la dimensione è di 182MB)
3. Estrarre il contenuto compresso dove si preferisce.
4. Scaricare la versione del Delta Pack facendo molta attenzione che corrisponda esattamente a quella dell'Eclipse scaricato in precedenza (questo pacchetto è necessario se si vuole creare la propria versione eseguibile del CSS o di un proprio programma creato in java). Per sapere quale versione si è scaricato fare i seguenti passaggi: andare sul Menu Help → About Eclipse→About Eclipse → premere il pulsante che ha come tooltip Eclipse.org

5. Estrarre il file compresso nella cartella dove si è estratto precedentemente Eclipse facendo attenzione a non sovrasrcivere nessun file.

A.2.2 Configurazione

1. Aprire Eclipse e scegli la cartella di lavoro.
2. Scaricare i sorgenti del CSS dal seguente sito http://ics-web.sns.ornl.gov/css/products.html
3. Dal menu File → Import → oppure tasto destro del mouse selezionare la voce import, a questo punto compare l'immagine A.4

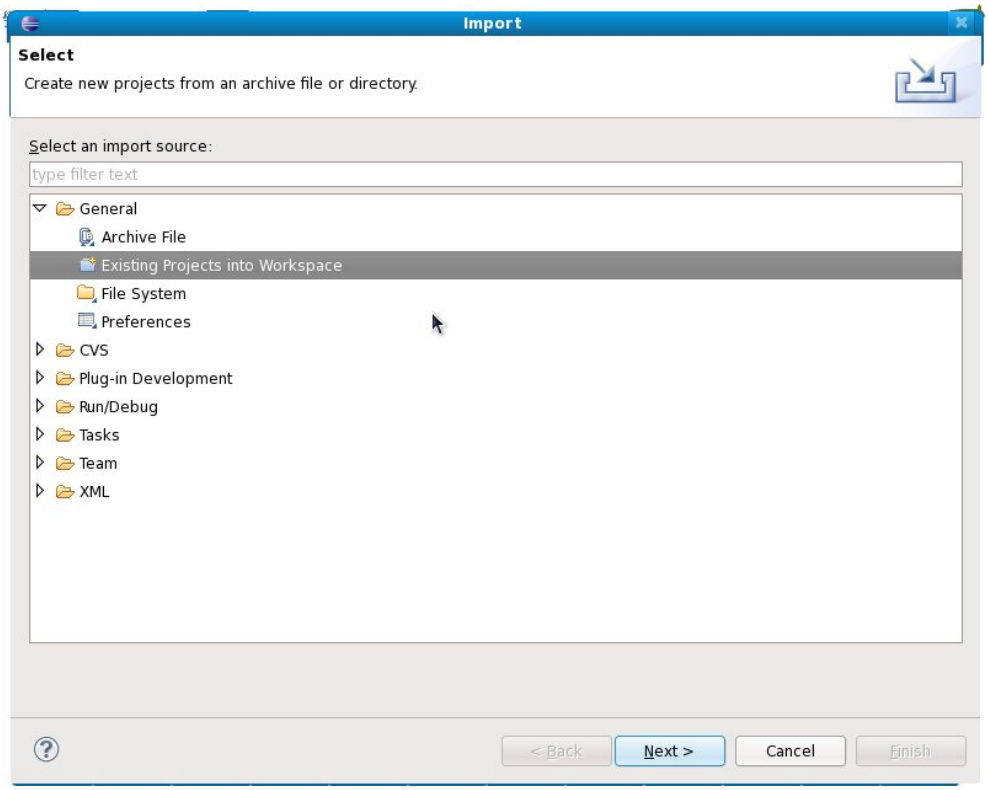

Figura A.4: Pannello per importare i sorgenti

4. Selezionare la voce "archive file", premere "Browse" infine scegliere il file compresso precedentemente scaricato
5. premere fine
6. se compare una finestra di sovrascrittura premere no

A.2.3 Eseguire il CSS nell'ambiente di lavoro

1. aprire il "package" org.csstudio.sns.product
2. aprire il file SNS_CSS.product e premere il pulsante play sulla finestra chiamata "Overview" vedi Figura A.5

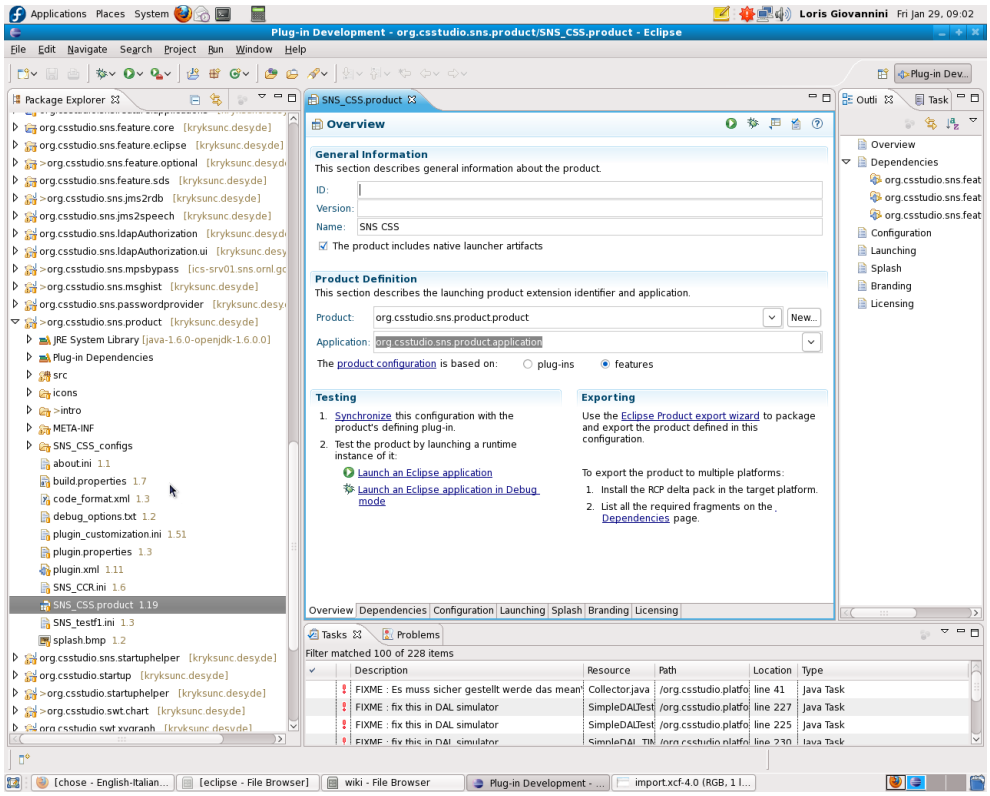

Figura A.5: Immagine della schermata relativa al file SNS_CSS.product

A.2.4 Aggiungere plug-in al CSS

Si possono trovare molti plug-in da aggiungere alla configurazione di base del prodotto; qui di seguito viene riportata la procedura ed un elenco dei plug-in che servono al Progetto SPES.

1. Menu → Run → Run Configurations....
2. Selezionare la voce Plug-ins Tab e selezionare:

   (a) org.cssstudio.alarm.[All] (escluso the annunciator))

   (b) org.cssstudio.archive.channelarchive.

   (c) org.cssstudio.debugging.jmsmonitor

   (d) org.csstudio.debugging.rdbshell

3. premere il pulsante "Apply"

4. premere il pulsante "run"

In questo modo però i plug-in aggiunti saranno attivi solo nella versione lanciata tramite Eclipse; se si vuole produrre un eseguibile con i plug-in sopra menzionati si devono seguire le seguenti istruzioni:

1. Aprire SNS_CSS.product (in Eclipse)

2. Aprire la sottofinestra chiamata "Dependencies"

3. All'interno della finestra che si apre aprire org.csstudio.sns.feature.core (numero versione es 1.0.xx)

4. La finestra principale è suddivisa in due parti una intitolata "Plug-ins and Frangmentation" e l'altra "Plug-ins Details"; sotto al titolo "Plug-ins and Frangmentation" compare una frame interno con una lista di plug-in, a fianco a questo c'è un pulsante con scritto "Add", premerlo e nella finestra che compare inserire in alto i nomi dei plug-in che si vogliono installare.

A.2.5 Creare il file eseguibile del CSS

Una volta fatti tutti i punti precedenti, si può procedere con la creazione di un proprio file eseguibile del CSS. La prima volta che si utilizzerà la funzione di export bisognerà innanzittutto eseguire questi passi preliminari, se si è deciso di scompattare il file Delta-pack in una directory diversa da quella scelta per installare Eclipse:

1. Menu → Windows→Preferences

2. Nel menu ad albero che compare alla sinistra scegliere "Plugin-in Devlopment" → "Target Platform"

3. Selezionare la scritta "Running Platform (Active)"

4. Premere il pulsante "Edit"

5. Nella finestra intitolata "Location" premere il pulsante Add, compare la seguente schermata A.6

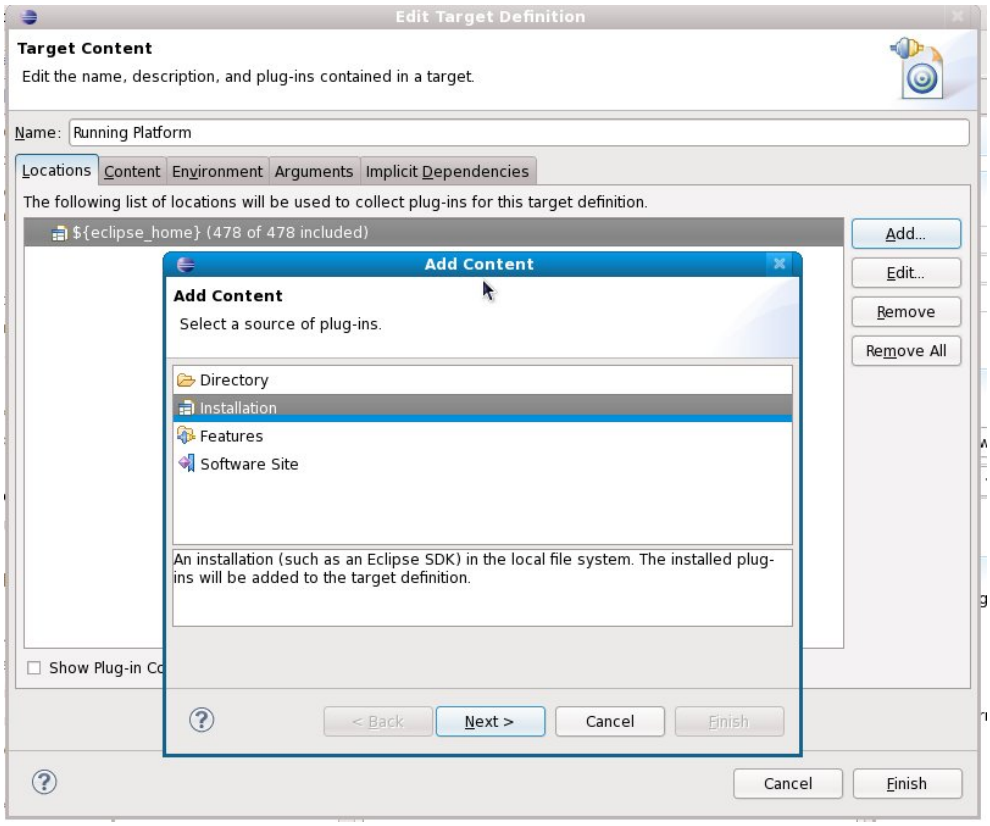

Figura A.6: Finestra aggiungi contenuto

6. selezionare la voce "Installation" e premere il pulsante "Next"

7. nella nuova finestra premere il pulsante "Browser" e selezionare la cartella dove è stato scaricato il Delta-Pack

8. premere il pulsante fine poi il pulsante applica ed infine il pulsante "OK"

Se questi passi sono stati già fatti basta aprire SNS_CSS.product e premere il pulsante in alto a destra con etichetta "Export" (il secondo dopo il tasto play) come si vede in figura A.5 e nella nuova finestra che compare

1. selezionare la cartella di destinazione
2. spuntare la casella con la voce "Export multiple platform"
3. premere il pulsante successivo e selezionare le piattaforme per cui si vuole creare l'eseguibile (es. linux (gtk/x86) e win32 (win32/x86))
4. premere il pulsante fine.

Appendice B

Codice Prodotto

B.1 Archive Engine e Hypertable

In questo parte dell'appendice si riportano i metodi delle classi da me modificate e le nuove classi create per fare in modo che l'Archive Engine supporti anche Hypertable. La parte più difficile, a causa della scarsa documentazione, è stata la comprensione dell'architettura e quindi le classi da andare a modificare. Qui di seguito (figura B.1 e figura B.2) vengono riportati due diagrammi delle classi semplificati che aiutano a comprendere l'architettura generale.

Si riporta solo il metodo modificato, con le istruzioni necessarie all'apertura della connessione all'interno della classe RDB-Archive.

Listing B.1: Codice RDB-Archive

```
1  private HTArchive ht_client;
2
3  /** Connect to RDB */
4  @SuppressWarnings("nls")
5  private void connect() throws Exception
6  {
7
8    // Create new connection
9    CentralLogger.getInstance().getLogger(this).
10   debug("Connecting to '" + url + "' " + (use_staging ? "(stage)" : "(main)"));
11   rdb = RDBUtil.connect(url, user, password, false);
```

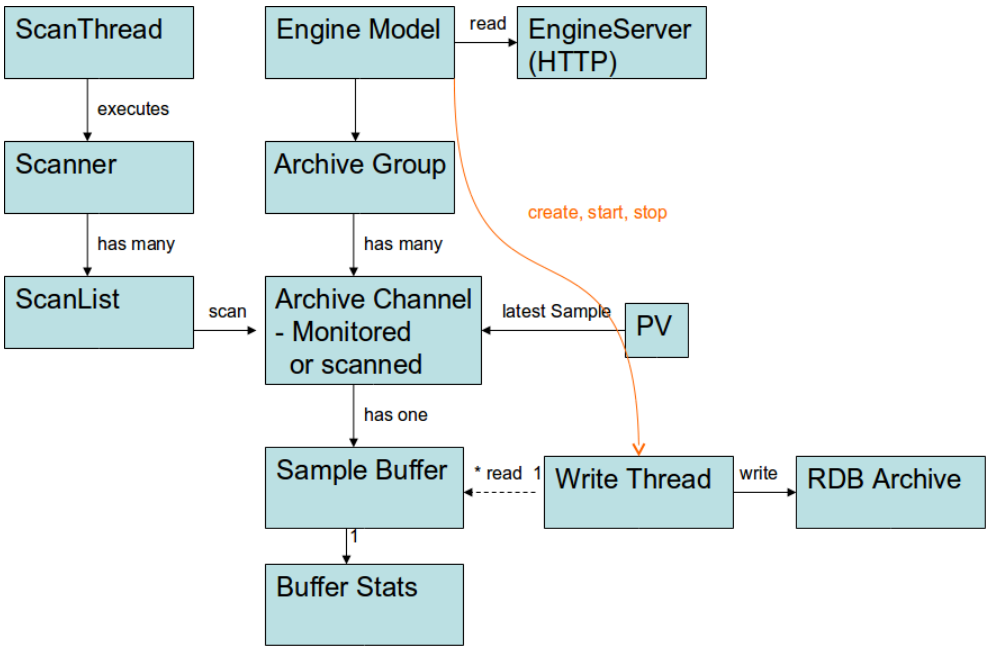

Figura B.1: Diagramma delle classi Semplificato che illustra in breve la connessione delle principali classi dell'Engine

```
12  sql = new SQL(rdb.getDialect(), use_staging);
13  channels = new ChannelCache(this);
14  severities = new SeverityCache(rdb, sql);
15  stati = new StatusCache(rdb, sql);
16
17  // TODO Remove Oracle test code
18  if (false)
19  {
20    System.out.println("Enabling Oracle trace");
21    final Statement stmt = rdb.getConnection().createStatement();
22    stmt.execute("alter session set tracefile_identifier='KayTest_max'");
23    stmt.execute("ALTER SESSION SET events " +
24    "'10046 trace name context forever, level 12'");
25  }
26
27  if (hypertable && ht_client==null){
28    ht_client = new HTArchive("localhost", 38080);
```

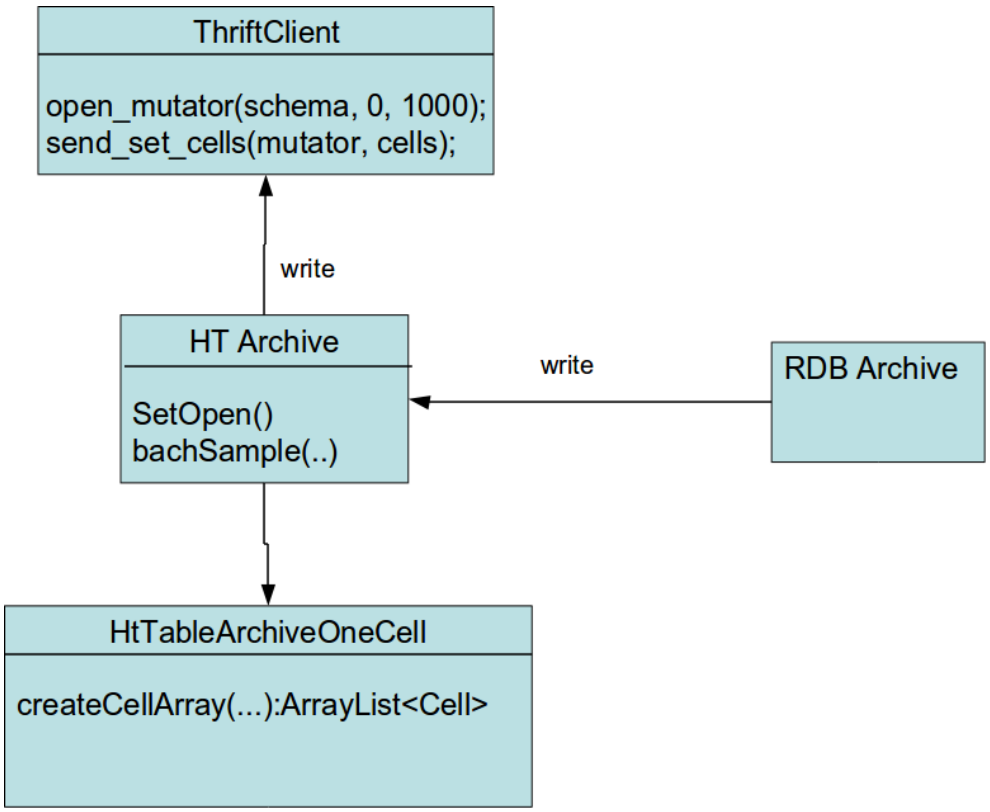

Figura B.2: Diagramma delle classi, dettaglio classi coinvolte nella scrittura dei dati

```
29
30      if (!ht_client.isOpen()){
31        ht_client.setOpen(true);
32      }
33    }
34
35    // In case of a re-connect after error, forget samples that caused error
36    if (debug_batch)
37    {
38      batched_channel.clear();
39      batched_samples.clear();
40    }
41
42  }
```

```
43
44 /** Add a sample to the archive.
45 * <p>
46 * For performance reasons, this call actually only adds
47 * the sample to a 'batch'.
48 * Need to follow up with 'commitBatch()' when done.
49 * @param channelConfig Channel to which this sample belongs
50 * @param sample
51 * @throws Exception on error
52 * @see #commitBatch()
53 */
54 public void batchSample(final ChannelConfig channelConfig,
55 final IValue sample) throws Exception
56 {
57   // Need to write meta data?
58   if (channelConfig.getMetaData() == null)
59     writeMetaData(channelConfig, sample);
60   final Timestamp stamp = TimeWarp.getSQLTimestamp(sample.getTime());
61   final Severity severity =
62   severities.findOrCreate(sample.getSeverity().toString());
63   final Status status = stati.findOrCreate(sample.getStatus());
64   String group = findGroup(channelConfig.getGroupId()).getName();
65   //Manage hypertable
66   //File dati = new File ("dati");
67   //FileOutputStream strem = new FileOutputStream(dati);
68
69   if (campioniHt <=10000){
70     control++;
71     if (ht_client!=null){
72       timeHt = System.currentTimeMillis();
73       ht_client.batchSamples(channelConfig, stamp, severity, status, sample,
74                              this.url,group,4);
75       sommaHt+=(System.currentTimeMillis()-timeHt);
76       campioniHt++;
```

```
77
78          if (campioniHt==10000){
79          //strem.write((”10000 campioni scritti in
                tempo:”+sommaHt+”\n”).getBytes());
80          System.out.println( campioniHt + ”campioni preparati e scritti in tempo ht: ”
81                                      +sommaHt);
82          campioniHt=0;
83          sommaHt=0;
84        }
85      }
86      }
87      //end
88      if(campioniSql<=10000){
89        timeSql=System.currentTimeMillis();
90        control−−;
91        //System.out.println(”control=” +control);
92        if (sample instanceof IDoubleValue)
93        {
94          final double dbl[] = ((IDoubleValue)sample).getValues();
95          batchDoubleSamples(channelConfig, stamp, severity, status, dbl);
96        }
97        else if (sample instanceof ILongValue)
98        {
99          final long num = ((ILongValue)sample).getValue();
100         batchLongSamples(channelConfig, stamp, severity, status, num);
101       }
102       else if (sample instanceof IEnumeratedValue)
103       { // Enum handled just like (long) integer
104         final long num = ((IEnumeratedValue)sample).getValue();
105         batchLongSamples(channelConfig, stamp, severity, status, num);
106       }
107       else
108       { // Handle string and possible other types as strings
109         final String txt = sample.format();
```

```
110         batchTextSamples(channelConfig, stamp, severity, status, txt);
111       }
112       sommaSql+=(System.currentTimeMillis()−timeSql);
113       campioniSql++;
114     }
115     if (campioniSql==10000){
116       campioniSql=0;
117       //strem.write((”10000 campioni scritti SU RDB in
              tempo:”+sommaSql+”\n”).getBytes());
118       System.out.println(”10000 campioni preparati in tempo: ” + sommaSql);
119       sommaSql=0;
120     }
121     if (debug_batch)
122     {
123       batched_channel.add(channelConfig);
124       batched_samples.add(sample);
125     }
126 }
```

Listing B.2: Codice HTArchive

```
1  package org.csstudio.archive.rdb;
2
3  import java.sql.Timestamp;
4  import java.util.ArrayList;
5  import org.apache.thrift.TException;
6  import org.apache.thrift.transport.TTransportException;
7  import org.csstudio.archive.rdb.ChannelConfig;
8  import org.csstudio.archive.rdb.Severity;
9  import org.csstudio.archive.rdb.Status;
10 import org.csstudio.platform.data.IValue;
11 import org.hypertable.thrift.ThriftClient;
12 import org.hypertable.thriftgen.Cell;
13 import org.hypertable.thriftgen.ClientException;
14
```

```
15 public class HTArchive {
16 private long mutator;
17 private boolean open = false;
18 private boolean openMutator =false;
19 private HtTableArchive archive = new HtTableArchive();
20 private HtTableArchive2 archiver2 = new HtTableArchive2();
21 private HtTableArchiveQual archiver3 = new HtTableArchiveQual();
22 private HtTableArchiveOneCell archiver4 = new HtTableArchiveOneCell();
23 private ThriftClient client = null;
24 private final String schema = "archive";
25
26 public HTArchive(String host, int port) {
27     try {
28             this.client = ThriftClient.create(host, port);
29             setOpen(true);
30     } catch (TTransportException e) {
31             // TODO Auto-generated catch block
32             setOpen(false);
33             e.printStackTrace();
34     } catch (TException e) {
35             setOpen(false);
36             // TODO Auto-generated catch block
37             e.printStackTrace();
38     }
39 }
40 /**
41 * call the metod to create the hypertable cells
42 * @param channel
43 * @param stamp
44 * @param severity
45 * @param status
46 * @param sample
47 * @param url
48 * @param group
```

```
49 * @parma int call the different schema.
50 * @throws TException
51 * @throws ClientException
52 */
53 public void batchSamples(final ChannelConfig channel,
54             final Timestamp stamp, final Severity severity,
55             final Status status, final IValue sample, final String url,
56             final String group, int i) {
57 ArrayList<Cell> cells=null;
58   if (i==1){
59     cells= archive.createCellArray(channel, stamp, severity,status, sample, url,
           group);
60   try {
61             client.send_set_cells(mutator, cells);
62   } catch (TException e) {
63             e.printStackTrace();
64   }
65   }
66 String insertsql=null;
67   if (i==2){
68             insertsql = archiver2.createSQL(channel, stamp, severity,
69                       status, sample, url, group);
70             try {
71                       try {
72                                 client.hql_query(insertsql);
73                       } catch (TException e) {
74                                 // TODO Auto-generated catch block
75                                 e.printStackTrace();
76                       }
77             } catch (ClientException e) {
78                       // TODO Auto-generated catch block
79                       e.printStackTrace();
80             }
81   }
```

```
82    if (i==3){
83            cells = archiver3.createCellArray(channel, stamp, severity,status, sample,
                  url, group);
84
85                  try {
86                          client.send_set_cells(mutator, cells);
87                  } catch (TException e) {
88                          // TODO Auto-generated catch block
89                          e.printStackTrace();
90                  }
91    }
92            if (i==4){
93                  try {
94                          cells = archiver4.createCellArray(channel, stamp,
                                severity,status, sample, url, group);
95                          client.send_set_cells(mutator, cells);
96                          } catch (Exception e) {
97                                  // TODO Auto-generated catch block
98                                  e.printStackTrace();
99                          }
100           }
101   }
102
103   public void close() {
104     // scrive una colonna nel database
105     if (isOpen())
106             try {
107                     if (mutator!=0)
108                       client.close_mutator(mutator, true);
109                     client.close();
110                     setOpen(false);
111             } catch (ClientException e) {
112                     // TODO Auto-generated catch block
113                     e.printStackTrace();
```

```
114             } catch (TException e) {
115                     // TODO Auto-generated catch block
116                     e.printStackTrace();
117             }
118 }
119
120 public void setOpen(boolean open) {
121   try {
122       if (open)
123           client.open();
124           //automatic flush after 1000 ms
125           mutator = client.open_mutator(schema, 0, 1000);
126       this.open = open;
127       this.openMutator=open;
128   } catch (ClientException e) {
129       // TODO Auto-generated catch block
130       this.open = false;
131       this.openMutator = false;
132       e.printStackTrace();
133   } catch (TException e) {
134       // TODO Auto-generated catch block
135       if(e instanceof TTransportException)
136               if(((TTransportException) e).getType()==2){
137         this.open = true;
138         if (!openMutator){
139           try {
140                   mutator = client.open_mutator(schema, 0, 1000);
141           } catch (ClientException e1) {
142                   // TODO Auto-generated catch block
143                   e1.printStackTrace();
144           } catch (TException e1) {
145                   // TODO Auto-generated catch block
146                   e1.printStackTrace();
147           }
```

```
148                 this.openMutator = true;
149         }
150           }else{
151                     this.open = false;
152                       this.openMutator = false;
153               }
154     }
155 }
156
157 public boolean isOpen() {
158
159         return open;
160 }
161
162 public void flush(){
163 if(mutator!=0)
164           try {
165               //long timeTempflush=System.currentTimeMillis();
166               client.send_flush_mutator(mutator);
167               //long timesendFlush=System.currentTimeMillis() - timeTempflush;
168               //System.out.println("flush send time " + timesendFlush);
169               //timeTempflush=System.currentTimeMillis();
170               client.flush_mutator(mutator);
171               //timesendFlush=System.currentTimeMillis() - timeTempflush;
172               //System.out.println("flush mutator " + timesendFlush);
173           } catch (ClientException e) {
174               // TODO Auto-generated catch block
175               e.printStackTrace();
176           } catch (TException e) {
177                 // TODO Auto-generated catch block
178                 e.printStackTrace();
179           }
180 }
181 }
```

Listing B.3: Codice HtTableArchiveOneCell

```
1  package org.csstudio.archive.rdb;
2
3  import java.sql.Timestamp;
4  import java.util.ArrayList;
5  import java.util.HashMap;
6  import java.util.List;
7
8  import org.csstudio.platform.data.IDoubleValue;
9  import org.csstudio.platform.data.IEnumeratedValue;
10 import org.csstudio.platform.data.ILongValue;
11 import org.csstudio.platform.data.IValue;
12 import org.hypertable.thriftgen.Cell;
13 import org.hypertable.thriftgen.Key;
14
15 /**
16 * This is a prototipe
17 * CREATE TABLE "archive" (
18 * "pv",
19 * ACCESS GROUP variable ("pv")
20 * );
21 * @author loris
22 *
23 */
24 public class HtTableArchiveOneCell {
25
26   private Cell pv=null;
27
28   private String[] p={"timeStamp","group","val","severity","satus","engine",
       "sempleMode",
29                 "samplePeriod","tipoValue","separator"};
30
31   // private ArrayList<String> cellnames =new ArrayList<String>(p);
32   private String rowkey;
```

```
33  private ArrayList<Cell> cells;
34
35      /**
36       * Constructor for table archive and set parameter
37       * @param channel
38       * @param stamp
39       * @param severity
40       * @param status
41       * @param sample
42       * @param url
43       * @param group
44       */
45      public ArrayList<Cell> createCellArray(final ChannelConfig channel,
46          final Timestamp stamp, final Severity severity,
47          final Status status,final IValue sample, final String url,final String
                vargroup)throws Exception {
48              cells = new ArrayList<Cell>();
49
50                      short flag = 255;
51
52                      long ts = stamp.getTime();
53
54                      rowkey=channel.getName()+"."+ts;
55                      //the separator must initialized before all
56              this.pv =new Cell(new Key(rowkey,"pv",null,flag));
57
58              final String sepc ="#c#";
59              final String sepv ="#v#";
60              HashMap<String, String> valtip= getValue(sample);
61
62              final String spv = channel.getName();
63              final String staus = status.getName();
64              final String timestamp = ""+stamp.getTime();
65              final String seve =severity.getName().toString();
```

```
66              final String val = valtip.get("value");
67              final String sempleMode = channel.getSampleMode().getName();
68              final String samplePeriod =
                    ((Double)channel.getSamplePeriod()).toString();
69              final String tipoValue = valtip.get("tipo");
70                      final String svalue= spv + sepc + staus + sepc + timestamp
                            + sepc + seve + sepc + vargroup + sepc + url + sepc+
71                                  sempleMode + sepc + samplePeriod + sepc
                                        + tipoValue + sepc + val;
72
73                      this.pv.value=svalue.getBytes();
74                      //System.out.println(pv.toString());
75              cells.add(this.pv);
76
77                          return cells;
78      }
79      /**
80        * in this moment doesn't manage the array
81        */
82      public HashMap<String, String> getValue(final IValue sample) throws
            Exception{
83              String value="";
84              String tipo="";
85              if (sample instanceof IDoubleValue)
86      {
87          final double dbl[] = ((IDoubleValue)sample).getValues();
88
89          for (int i=0;i<dbl.length && !Double.isNaN(dbl[i]);i++){
90              value =value + Double.toString(dbl[i]) +
                    (i==dbl.length-1?"":"#v#");
91          }
92          if (dbl.length>1){
93              tipo="array"; }
94          else{
```

```
95              tipo = "double";}
96
97          }
98          else if (sample instanceof ILongValue)
99          {
100             final long num = ((ILongValue)sample).getValue();
101             value = Long.toString(num);
102             tipo = "long";
103         }
104         else if(sample instanceof IEnumeratedValue)
105         { // Enum handled just like (long) integer
106             final long num = ((IEnumeratedValue)sample).getValue();
107             value = Long.toString(num);
108             tipo="long";
109         }
110                 else{
111           // Handle string and possible other types as strings
112             value = sample.format();
113                     tipo= "string";
114                 }
115             HashMap<String, String> map = new HashMap<String, String>();
116             map.put("tipo", tipo);
117             map.put("value", value);
118                 return map;
119         }
120
121         /**
122           * return all column with value of table archive
123           * @return cells of archive Table
124           */
125         public List<Cell> getCells() {
126                 return cells;
127         }
128
```

```
129         public String getText (byte[] arr)
130         {
131           String s = new String(arr);
132           return s;
133         }
134 }
```

B.2 Reperimento dei dati da Hypertable DataBrowser

Per il reperimento dei dati si è proceduto a modificare il costruttore della calsse RDBArchiveReader in modo che oltre ad aprire la connsessione con MySQL apra la connessione anche con Hypertable.

Listing B.4: codice ArchiveReader

```
1  public class RDBArchiveReader implements ArchiveReader
2  {
3  /** Initialize
4        * @param url Database URL
5        * @param user .. user
6        * @param password .. password
7        * @param stored_procedure Stored procedure or "" for client-side optimization
8        * @throws Exception
9        * @throws Exception on error
10       */
11      public RDBArchiveReader(final String url, final String user,
12              final String password, final String stored_procedure) throws Exception
13      {
14          this.url = url;
15          this.user = user;
16          this.password = (password == null) ? 0 : password.length();
17          timeout = Preferences.getTimeoutSecs();
18          rdb = RDBUtil.connect(url, user, password, false);
```

```
19          String local=”localhost”;
20          client_ht =ThriftClient.create(local, 38080);
21           //client_ht.hql_query2(”select * from archive2 KEYS_ONLY LIMIT =1”);
22           // Ignore the stored procedure for MySQL
23          if (rdb.getDialect() == Dialect.MySQL)
24              this.stored_procedure = ””;
25          else
26              this.stored_procedure = stored_procedure;
27          String schema = Preferences.getSchema();
28          if (schema.length() > 0)
29              schema = schema + ”.”;
30          sql = new SQL(rdb.getDialect(), schema);
31                              stati = getStatusValues();
32                              severities =getSeverityValues();
33
34      }
```

Metodo per decodificare i valori inseriti in Hypertable:

Listing B.5: Codice AbstractRDBValueIterator

```
1  abstract public class AbstractRDBValueIterator implements ValueIterator {
2  ....
3  ....
4   protected IValue decodeSampleTableValue3(final List<String> cells) // throws
5    // Exception
6    {
7              //int size = cells.size();// numero colonne dello schema
8              //I have only one cell in the schema
9    //the value are in 3 position of List
10             final String[] valued = cells.get(3).split(”#c#”);
11             final Timestamp stamp = new Timestamp(new
                   Long(valued[2]).longValue());
12             //get the value as saved into cell_value
13             final ITimestamp time= TimeWarp.getCSSTimestamp(stamp);
14               //TimeWarp.getCSSTimestamp(stamp);
```

```
15          //time = getTimestamp(stamp, );
16          final String status = valued[1];
17          final ISeverity severity = filterSeverity(new Severity(valued[3]),valued[1]);
18          final String tipoValue = valued[8];
19
20          final String[] values = valued[9].split("#v#");
21          final String value=values[0];
22          // Is it an error to have enumeration strings for double
23          // samples?
24          // In here, we handle it by returning enumeration samples,
25          // because the meta data would be wrong for double values.
26          if (tipoValue.equals("double")) {
27                  if (meta instanceof IEnumeratedMetaData)
28                  return ValueFactory.createEnumeratedValue(time, severity,
                        status,
29                          (IEnumeratedMetaData) meta, IValue.Quality.Original,
30                          new int[] { (int) new Double(value).doubleValue() });
31          // Double data. Get array elements – if any.
32
33          //final String[] doublestr = value.split("#v#");
34          double data[] = new double[values.length];
35          for (int n = 0; n < values.length; n++)
36                  data[n] = new Double(values[n]).doubleValue();
37
38          if (meta instanceof INumericMetaData)
39                  return ValueFactory.createDoubleValue(time, severity, status,
40                          (INumericMetaData) meta, IValue.Quality.Original, data);
41          // Make some meta data up
42          return ValueFactory.createDoubleValue(time, severity, status,
43                          getDefaultNumericMeta(), IValue.Quality.Original, data);
44  }
45          // Enumerated integer?ITTimeWarp
46          if (tipoValue.equals("long")) {
47                  final int num = new Integer(value).intValue();
```

```
48              if (meta instanceof IEnumeratedMetaData)
49                      return ValueFactory.createEnumeratedValue(time, severity,
                            status,
50                              (IEnumeratedMetaData) meta, IValue.Quality.Original,
51                              new int[] { num });
52              // Fall back to plain (long) integer
53              if (meta instanceof INumericMetaData)
54                      return ValueFactory.createLongValue(time, severity, status,
55                              (INumericMetaData) meta, IValue.Quality.Original,
56                              new long[] { num });
57              // Numeric, but not meta data
58              return ValueFactory.createLongValue(time, severity, status,
59                              getDefaultNumericMeta(), IValue.Quality.Original,
60                              new long[] { num });
61            }
62              // Default to string
63                  final String txt = new String(value);
64                  return ValueFactory.createStringValue(time, severity, status,
65                              IValue.Quality.Original, new String[] { txt });
66              }
67  ...
68  ...
69  ...
70  }
```

Classe utilizzata per la lettura dei dati.

Listing B.6: codice RawSampleIterator

```
1  package org.csstudio.archivereader.htrdb;
2
3  import java.sql.Timestamp;
4  import java.util.ArrayList;
5  import java.util.Iterator;
6  import java.util.List;
7
8
9  import org.csstudio.platform.data.ITimestamp;
10 import org.csstudio.platform.data.IValue;
11 import org.csstudio.platform.utility.rdb.TimeWarp;
12 import org.hypertable.thriftgen.RowInterval;
13 import org.hypertable.thriftgen.ScanSpec;
14
15
16 /** Value Iterator that reads from the SAMPLE table.
17  * @author Kay Kasemir
18  */
19 public class RawSampleIterator extends AbstractRDBValueIterator
20 {
21
22     /** Result of <code>sel_samples</code> */
23     private Iterator<List<String>> iteratorcell = null;
24     private int conta=0;
25
26     /** 'Current' value that <code>next()</code> will return,
27      * or <code>null</code>
28      */
29     private IValue value = null;
30
31     /** Initialize
32      * @param reader RDBArchiveReader
```

```
33      * @param channel_id ID of channel
34      * @param start Start time
35      * @param end End time
36      * @throws Exception on error
37      */
38     public RawSampleIterator(final RDBArchiveReader reader,
39             final String name, final int channel_id,final ITimestamp start,
40             final ITimestamp end) throws Exception
41     {
42         super(reader, name, channel_id);
43         try
44         {
45             determineInitialSample(start, end, name );
46         }
47         catch (Exception ex)
48         {
49             if (! RDBArchiveReader.isCancellation(ex))
50                 throw ex;
51             // Else: Not a real error; return empty iterator
52             value = null;
53         }
54     }
55
56     /** Get the samples: <code>result_set</code> will have the samples,
57      * <code>value</code> will contain the first sample
58      * @param start Start time
59      * @param end End time
60      * @throws Exception on error, including cancellation
61      */
62     private void determineInitialSample(final ITimestamp start, final ITimestamp
        end, final String name) throws Exception
63     {
64
65         final Timestamp start_stamp = TimeWarp.getSQLTimestamp(start);
```

```
66          final Timestamp end_stamp = TimeWarp.getSQLTimestamp(end);
67          /**
68           * hypertable don't have the management of timezone so I must
69           * convert the start and the end time in GMT
70           */
71
72          int
                offset=start.toCalendar().getTimeZone().getOffset(start_stamp.getTime());
73          Timestamp GMTstart_stamp = new
                Timestamp(start_stamp.getTime()-offset);
74          Timestamp GMTend_stamp = new
                Timestamp(end_stamp.getTime()-offset);
75
76          /**
77           * hypertable don't have the management of timezone so I must
78           * convert the start and the end time in GMT.
79           * In Hypertable in the predicate Timestamp the time end must be > then
                start time
80           */
81          long t = System.currentTimeMillis();
82          ScanSpec scan = new ScanSpec();
83          RowInterval intervalrow = new RowInterval();
84                  intervalrow.setStart_row(name+".+start_stamp.getTime());
85              intervalrow.setStart_inclusive(true);
86                  intervalrow.setEnd_row(name+"~");
87              intervalrow.setEnd_inclusive(false);
88          //intervalrow.setStart_inclusive(false);
89          ArrayList<RowInterval> lista = new ArrayList<RowInterval>();
90          lista.add(intervalrow);
91          scan.setRow_intervals(lista);
92          scan.setStart_time(start_stamp.getTime()*1000000);
93          scan.setEnd_time(end_stamp.getTime()*1000000);
94          scan.setRow_limit(10000);
95          long scanner = reader.getClient_ht().open_scanner("archive", scan, false);
```

```
96          List<List<String>>listcel =
                reader.getClient_ht().next_cells_as_arrays(scanner);
97          reader.getClient_ht().close_scanner(scanner);
98          iteratorcell=listcel.iterator();
99          long f = System.currentTimeMillis()−t;
100         System.out.println("extration semple "+ f);
101         // Get first sample
102         if (iteratorcell.hasNext()){
103                 conta=1;
104                 value =decodeSampleTableValue3(iteratorcell.next());
105         }else
106                 close();
107
108     }
109
110     /** {@inheritDoc} */
111     public boolean hasNext()
112     {
113         return value != null;
114     }
115
116     /** {@inheritDoc} */
117     @SuppressWarnings("nls")
118     public IValue next() throws Exception
119     {
120         // This should not happen...
121         if (iteratorcell == null)
122             throw new Exception("RawSampleIterator.next(" + channel_id + ")
                    called after end");
123
124         // Remember value to return...
125         final IValue result = value;
126         // ... and prepare next value
127         try
```

```
128         {
129                 if (iteratorcell.hasNext()){
130                         value = decodeSampleTableValue3(iteratorcell.next());
131                     conta++;
132                 }else{
133                         close();
134                     System.out.println("totale campioni estratto " +conta);
135                 }
136         }
137         catch (Exception ex)
138         {
139             close();
140             if (! RDBArchiveReader.isCancellation(ex))
141                 throw ex;
142             // Else: Not a real error; return empty iterator
143         }
144         return result;
145     }
146
147     /** Release all database resources.
148      * OK to call more than once.
149      */
150     @Override
151     public void close()
152     {
153         super.close();
154         value = null;
155     }
156 }
```

Bibliografia

[1] M. Manzolaro. *Analisi termica e strutturale del bersaglio diretto per la produzione di fasci radioattivi per il progetto SPES*. Tesi di Laurea Specialistica, Università degli Studi di Padova, a.a.2006-07.

[2] G. Diavola. *Fasci di ioni radioattivi ai laboratori nazionali del sud dell'INFN: il progetto EXCYT*. Il Saggiatore, 1999.

[3] M. Lindroos. *Of ISOL-type radioactive beam facilities*. EPAC, 2004.

[4] http://www.ganil.fr/eurisol/EURISOLlinks.html.

[5] M. Deicher. Radioactive isotopes in solid state physics. *Europhysics News*, Vol 33, No 3, 2002.

[6] A. Andrighetto, L. Biasetto, M. Manzolaro, P. Benetti, S. Carturan, P. Colombo, F. Gramegna, G. Meneghetti, B. Monelli, G. Prete, et al. The SPES Project At LNL. In *AIP Conference Proceedings*, volume 1099, page 728, 2009.

[7] G. Meneghetti, M. Manzolaro, and A. Andrighetto. Design of the SPES Target Heating System: theoretical analyses and comparison with experimental data. In *TCN CAE 2008 International Conference on Simulation Based Engineering and Sciences, October 16–17, 2008, Venice, Italy.*

[8] J. Vasquez. *Activity Report january-may 2009*. Laboratori Nazionali di Legnaro, 2009.

[9] M. Kraimer, J. Anderson, A. Johnson, E. Norum, J. Hill, R. Lange, and B. Franksen. EPICS InputIOutput Controller Application Developer's Guide. Release 3.14. 8, 2005. Argonne National Laboratory.

[10] M. Montis. *Utilizzo di software EPICS per lo sviluppo del sistema di controllo dell'apparato di produzione di ioni del progetto SPES.* Tesi di Laurea Specialistica, Università degli Studi di Padova, a.a.2009-10.

[11] K. Kasemir. Control System Studio Applications. ICALEPCS, 2007.

[12] K. Kasemir. Control System Studio (CSS) Data Browser. ICALEPCS, 2007.

[13] C. Scudellaro. *Utilizzo di software LabView per la realizzazione dell'interfaccia operatore dell'apparato Front End del progetto SPES.* Tesi di Laurea Triennale, Università degli Studi di Padova, a.a.2009-10.

[14] *Channel Archive Manual.* http://ics-web.sns.ornl.gov/kasemir/archiver/manual.pdf.

[15] K. Kasemir. *RDB Channel Archiver.* https://ics-web.sns.ornl.gov/css/docs/RDBChannelArchiver.doc.

[16] http://code.google.com/p/hypertable/wiki/ArchitecturalOverview.

[17] K. Kasemir, X. Chen, and E. Danilova. The Best Ever Alarm System Toolkit. *ICALEPCS09, Kobe Japen*, 2009.

[18] http://www.lnl.infn.it/ spes/tdr2008/tech_design08_index.htm.

[19] M. Libralato. *Studio elettro-termo-strutturale del sistema di estrazione e ionizzazione del progetto SPES.* Tesi di Laurea Specialistica, Università degli Studi di Padova, a.a.2008-09.

[20] A. Covello G. Prete. *SPES Techical Design Report.* Istituto Nazionale di Fisica Nucleare, Laboratori Nazionali di Legnaro, Aprile 2008.

[21] G. Jelicic. *Modellazione solida e test della Camera Target del progetto SPES.* Tesi di Laurea Triennale, Università degli Studi di Padova, a.a.2007-08.

[22] C. Gobbi. *Modellazione solida, progettazione e misure termiche su sistemi meccanici appartenenti alla facility SPES.* Tesi di Laurea Triennale, Università degli Studi di Padova, a.a.2008-09.

[23] http://www.aps.anl.gov/epics/.

[24] M. Giacchini R.Izsàk J. Vasquez E. Bindi M. Montis A. Andrighetto G. Bassato P. Benetti L. Biasetto M. Manzolaro B. Monelli R. Oboe G. Prete D. Scarpa L. Costa. Control system development for the spes project. In *Annual report 2008*, pages 205–206, Laboratori Nazionali di Legnaro, Luglio 2009.

[25] http://www.lnl.infn.it/annrep.html.

[26] L. Costa M. Giacchini J. Vasquez A. Andrighetto G. Prete Minglong Li G. Bassato. Safety requirements in spes control system: preliminary design. Ottobre 2009.

[27] http://www.aps.anl.gov/epics/wiki/index.php/rrm
_3-14.

[28] C. Negus. *Linux Bible.* Wiley Publishing, Inc., 10475 Crosspoint Boulevard Indianapolis, IN 46256, 2008 edition edition, 2008.

[29] R. MARTIN. Kraimer EPICS input/output controller (IOC) application developer's guider, 2003.

[30] Z. Kakucs, P. Duval, and M. Clausen. An EPICS to TINE translator. *Arxiv preprint physics/0111064*, 2001.

Ultima verifica delle pagine web in data 20 marzo 2011

Ringraziamenti

Ringrazio in primis me stesso, per non aver mai mollato neanche nei momenti più duri.
Ringrazio tutti i colleghi del progetto SPES per il fantastico ambiente che mi ha permesso di imparare molte cose, non solo nell'ambito informatico, in particolare: Alberto Andrighetto per i suoi preziosi consigli e le innumerevoli panoramiche generali sul Progetto SPES; Giorgio Bassato per le sue dritte quando rimanevo bloccato nelle compilazioni C, C++; Maurizio Montis, per avermi fatto da mentore in EPICS e Linux per le prime fasi del mio progetto; Nicola Conforto per i momenti goliardici dopo i pasticcini con i colleghi; Jesus Vasquez e Jacobo Montano per i loro suggerimenti sulle interfacce grafiche.
Ringrazio Mauro Giacchini per aver creduto in questo lavoro, tanto da convincermi a presentarlo allo Spring Epics Meeting (Aix en Provence, 31 maggio-2 giugno 2010); ringrazio un po' meno il malessere che gli ha impedito di darmi il suo supporto lasciandomi solo al battesimo del fuoco di fronte alla platea di un convegno internazionale.
Un ringraziamento a tutte le persone, che in silenzio mi hanno saputo sostenere pregando per me in tutti questi anni di università e non solo; tra queste un pensiero speciale va a mia nonna Stella.
Un ringraziamento ai miei genitori, i quali mi hanno sempre sostenuto, anche se non sempre hanno condiviso le mie scelte, ma credo che sia nel DNA dei genitori non condividere a pieno le scelte dei figli.
Ringrazio mia sorella Lorena e mio cognato Emanuele, per la bellissima nipotina con cui gioco tornando bambino anch'io, e per il loro sostegno morale e non solo.
Un ringraziamento speciale a Valentina Sartori, per aver letto questa tesi da cima a fondo pur non padroneggiando l'argomento, ed averla più e più volte corretta; per essere stata sveglia con me a preparare il mio ultimo esame, spronandomi nei momenti di bisogno e stando in silenzio quando proprio non ne potevo più.

Elenco delle figure