ALMA MATER STUDIORUM - UNIVERSITÀ DI BOLOGNA

SCUOLA DI INGEGNERIA E ARCHITETTURA

CORSO DI LAUREA MAGISTRALE IN INGEGNERIA INFORMATICA

DISI

Dipartimento di Informatica – Scienza e Ingegneria

TESI DI LAUREA

in

SISTEMI IN TEMPO REALE

PROGETTO DI UN SISTEMA DI REPORTISTICA RICONFIGURABILE PER APPLICAZIONI DI CONTROLLO INDUSTRIALE

Candidato:

Tommaso Dall'Arno

Relatore:

Chiar.mo Prof. Ing. Eugenio Faldella

Correlatore:

Dott. Ing. Enrico Passadore

Anno Accademico 2014/15

III Sessione

Indice

Capitolo 1
Introduzione

Un impianto industriale moderno prevede un apparato di supervisione e di controllo automatico dei processi fisici che lo caratterizzano. L'automatizzazione del controllo avviene perché risulta inefficace o pericoloso un intervento manuale da parte di un operatore direttamente sull'impianto, mentre la supervisione del processo ha lo scopo di fornire le informazioni rilevanti del sistema in esame in modo aggregato e facilmente comprensibile dal personale responsabile del corretto funzionamento del sistema. Per questi motivi, fin dall'avvento dell'elettronica a stato solido negli anni '60, si sono sviluppati in ambito industriale sistemi di supervisione, controllo e acquisizione dati, in inglese SCADA (Supervisory Control And Data Acquisition).

Come il nome stesso suggerisce la funzionalità su cui si basa uno SCADA è l'acquisizione dei dati. Queste informazioni, provenienti dai sensori distribuiti nell'impianto, vengono archiviate ed elaborate in modo da renderle immediatamente fruibili al personale responsabile. Con il termine *monitoraggio* si identifica la direzione in ingresso al sistema di supervisione, mentre con il termine *controllo* si vogliono indicare i segnali di comando inviati al processo supervisionato.

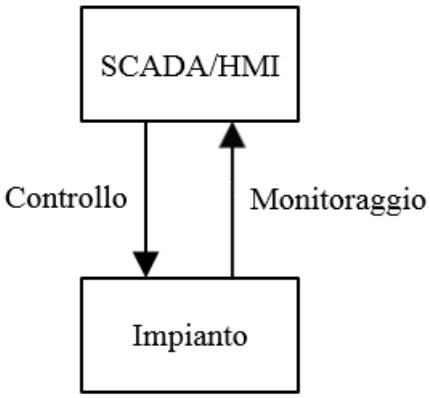

Figura 1. Interazione tra sistema controllato e apparato software

Nel seguito si utilizzeranno i termini impianto, macchina e processo per fare riferimento al sistema fisico supervisionato.

Per entrambe le direzioni è necessario creare una forma di interazione con l'utente: l'operatore deve essere in grado di ottenere informazioni costantemente aggiornate riguardo lo stato operativo di ogni componente dell'impianto e allo stesso tempo deve inviare comandi di controllo e di configurazione per agire sullo stato del sistema. Il componente che implementa questo tipo di interazione è l'interfaccia uomo-macchina, in inglese HMI (Human Machine Interface), tipicamente realizzata con un terminale video touchscreen. Dal momento che una HMI è un componente essenziale per ogni sistema SCADA, si parla spesso di sistemi SCADA/HMI.

L'implementazione del controllo automatico dei processi fisici che avvengono in una macchina automatica è realizzata da attuatori governati da apparati di elaborazione. Tipicamente vengono impiegati apparati denominati PLC (Programmable Logic Controller), che processano i dati sensoriali e generano i segnali di comando per l'attuazione del controllo.

Figura 2. Macchina automatica prodotta da IMA

Sistemi di controllo industriale che comprendono i componenti descritti sopra trovano applicazioni nell'industria manifatturiera, chimica, farmaceutica, in centrali energetiche, in apparati di telecomunicazione, in sistemi di gestione del traffico ferroviario o aereo. Sistemi SCADA/HMI vengono inoltre impiegati per il monitoraggio di sistemi geograficamente distribuiti come gli impianti di trasporto dei fluidi (acquedotti o oleodotti), o le infrastrutture di distribuzione dell'energia elettrica.

1.1 Scopo della tesi

All'interno di un sistema SCADA è presente un elemento software dedicato alla gestione documentale. Tali documenti, chiamati report, hanno l'obiettivo di mostrare in modo strutturato tutte le informazioni legate alla operatività della macchina supervisionata per renderle facilmente comprensibili al personale responsabile del suo funzionamento. Tali report vengono definiti con uno standard che consente di rappresentare le informazioni sul layout e sul contenuto dei report.

Tipicamente i file di definizione dei report vengono prodotti dal progettista in fase di configurazione della macchina e rimangono "cablati" nello SCADA: la modifica di qualsiasi aspetto legato ai report richiede un intervento diretto sui

relativi file di definizione. Questo significa che se l'utilizzatore della macchina automatica volesse personalizzare la struttura dei report dovrebbe rivolgersi al produttore per realizzare le modifiche.

Questo lavoro di tesi riguarda il progetto e la sintesi di uno strumento software che permette di generare, attraverso un'interfaccia grafica, file di definizione del layout e del contenuto dei documenti di report prodotti dal sistema informatico della macchina. L'obiettivo principale è di rendere possibile la personalizzazione dei report ad un livello di astrazione tale per cui non sia necessario interagire con le tecnologie impiegate per l'implementazione dei report stessi. L'automatizzazione del processo di produzione dei file che definiscono i report ha lo scopo di aumentare la flessibilità del sistema di reportistica introducendo un duplice beneficio: non solo verrebbe ridotto il tempo necessario alla configurazione dei report in fase di progettazione, ma si lascerebbe anche la libertà all’utilizzatore del sistema di personalizzare la struttura dei documenti prodotti. Tale sistema viene integrato nell'HMI in modo tale che l'utilizzatore possa personalizzare la struttura dei report agendo direttamente sull'interfaccia presente a bordo macchina.

1.2 La realtà IMA

La tesi è stata eseguita in IMA Industria Macchine Automatiche S.p.A, azienda leader mondiale nel settore dell'automazione per il packaging e la lavorazione di prodotti farmaceutici, cosmetici ed alimentari. In particolare lo sviluppo del progetto descritto in questo testo ha avuto luogo nella divisione IMA Life, specializzata nella progettazione e produzione di macchine automatiche per il settore farmaceutico. IMA è stata fondata nel 1961 a Bologna; acquistata due anni dopo dalla famiglia Vacchi, si è poi espansa fino a consolidare nell'anno fiscale 2014 un fatturato di 854.6 milioni di euro e oggi conta più di 4600 dipendenti, di cui 2300 in paesi esteri. IMA ha 34 centri di produzione dislocati

in Italia, Germania, Francia, Svizzera, Regno Unito, USA, India e Cina. IMA vende i propri prodotti in circa 80 stati. Circa 1000 sono i brevetti posseduti da IMA con cui viene promossa e protetta l'innovazione tecnologica nell'ambito del processamento e packaging farmaceutico e del packaging di thè e caffè.

1.3 Struttura della tesi

Nel Capitolo 1 viene descritto il contesto in cui il lavoro presentato ha avuto luogo e gli obiettivi che si sono voluti raggiungere. In particolare vengono descritti i sistemi di elaborazione dell'informazione tipicamente impiegati in un sistema di automazione industriale con particolare enfasi sull'infrastruttura di monitoraggio e il componente dedicato alla gestione di reportistica.

Il Capitolo 2 contiene una introduzione ai sistemi di controllo e monitoraggio di impianti industriali: vengono analizzati gli elementi principali della struttura fisica di uno SCADA, le componenti software impiegate e le funzionalità tipicamente garantite. Le considerazioni sugli aspetti operativi sono eseguite prendendo come riferimento il software SCADA commerciale iFIX.

Nel Capitolo 3 viene condotta una discussione dettagliata sui sistemi di reportistica, che costituiscono una parte fondamentale di qualsiasi apparato di supervisione industriale. In particolare sono descritte le principali normative che definiscono i requisiti che devono essere soddisfatti dai sistemi di gestione documentale nell'ambito della produzione di beni farmaceutici ed alimentari. Nella seconda parte del capitolo viene illustrata la soluzione implementativa con cui tali report vengono definiti in fase di progetto e sono descritti i meccanismi con cui i report vengono generati. Infine si discutono le potenzialità di un sistema di reportistica riconfigurabile integrato in uno SCADA, il cui progetto è l’argomento principale di questa tesi.

Nel Capitolo 4 l'attenzione si sposta sui passi seguiti per la realizzazione di un framework per la generazione di file di definizione di report tramite interfaccia grafica. Viene inizialmente analizzato il modello di processo di sviluppo software preso come riferimento. Successivamente vengono approfondite le principali fasi del ciclo di vita del progetto: la definizione e l'analisi dei requisiti, la progettazione dell'architettura software e le tecniche adottate per lo sviluppo del codice.

Infine, nel Capitolo 5 viene illustrato il funzionamento del sistema concepito e sono discussi i risultati dei test. Il testo si conclude con alcune considerazioni riguardo gli sviluppi futuri legati alle funzionalità di un apparato software per la gestione documentale nell'ambito del monitoraggio industriale. In particolare si pone l'attenzione sulle tematiche di remotizzazione del monitoraggio industriale e sui benefici che può apportare l'analisi dei dati gestiti da un sistema di reportistica attraverso tecniche di data mining.

Il testo si conclude con l'elenco dei riferimenti bibliografici e sitografici impiegati durante il progetto.

Capitolo 2

Introduzione ai sistemi SCADA/HMI

La funzionalità su cui si basa l'intero funzionamento di un sistema SCADA è l'acquisizione dei dati provenienti dai sistemi di controllo di campo. L'elaborazione di tale informazione ha lo scopo di rappresentarla in modo compatto e facilmente comprensibile dal personale responsabile del sistema sotto controllo.

La grande dimensione e l'eterogeneità dell'informazione raccolta da uno SCADA rende necessaria l'implementazione di funzionalità legate alla generazione di documenti che aggregano dati legati all'operatività della macchina. Tali documenti, definiti report, hanno fondamentale importanza anche per la validazione del processo produttivo e per la messa in vendita dei prodotti del processo industriale. Sia i motivi per cui è necessario l'impiego di uno strumento di produzione di reportistica che i meccanismi che ne definiscono la realizzazione sono di centrale importanza per questo lavoro di tesi e verranno ampiamente discussi nel Capitolo 3.

La storia dei sistemi SCADA inizia con l'avvento dell'elettronica a stato solido, negli anni '60. In quel periodo si concepirono i primi sistemi di supervisione e

controllo di apparati elettromeccanici per la lettura e modifica immediata di valori di correnti e tensioni. I primi processori adatti all'utilizzo real-time vennero prodotti verso la fine degli anni '60. I primi sistemi di interfaccia uomo-macchina vennero implementati con pulsanti per la selezione delle azioni di controllo. Una evoluzione importante avvenne nella seconda metà degli anni '60 con lo sviluppo dei monitor basati sul tubo a raggi catodici. Questi terminali video offrivano per la prima volta la possibilità di mostrare dati legati al sistema supervisionato in formato tabellare. All'inizio degli anni '70 lo sviluppo tecnologico rese disponibili monitor a raggi catodici a colori. La frequenza di aggiornamento dei dati mostrati a video in quel periodo era nell'ordine dei secondi. Anche l'architettura fisica dei sistemi di supervisione industriale ha subito un notevole sviluppo. I primi SCADA furono implementati in modo monolitico fino a quando si affermarono soluzioni distribuite con protocolli standardizzati, in cui ogni nodo è responsabile di un particolare task.

In questo e nei prossimi capitoli saranno illustrate le principali funzionalità di un software SCADA/HMI prendendo come riferimento pratico il software iFIX[1], un sistema di supervisione industriale che offre numerose funzionalità.

2.1 Architettura hardware

La struttura fisica di uno SCADA è tipicamente organizzata in modo distribuito. I sistemi di controllo, tipicamente PLC o softPLC, oltre a fornire i segnali di comando per gli attuatori, hanno anche lo scopo di fornire all'infrastruttura di supervisione i valori delle variabili di interesse per il monitoraggio della macchina. I nodi server possiedono i driver necessari per implementare la comunicazione con i sistemi di campo tramite il protocollo OPC, descritto più in dettaglio nel seguito. A livello server viene anche eseguita una elaborazione preliminare dei dati, acquisiti ad intervalli di tempo regolari e successivamente

1 Il software iFIX è uno SCADA/HMI prodotto da General Electrics.

memorizzati in un database real-time. I nodi client sono invece principalmente dedicati alla visualizzazione a video dei dati. Altre funzionalità legate al livello client riguardano ulteriori elaborazioni come la gestione degli allarmi, cioè agli aspetti strettamente legati al monitoraggio della macchina. La comunicazione tra i nodi server e client è tipicamente basata sul protocollo Ethernet.

La presenza di più nodi server permette di processare i dati in modo distribuito. Questo aspetto aumenta l'affidabilità del sistema riducendo la perdita di prestazioni nel caso in cui un nodo risulti fuori servizio. Per migliorare il livello di disponibilità di un sistema SCADA viene spesso preso in considerazione l'impiego di meccanismi di ridondanza di apparati server. In questo caso un unico nodo possiede al suo interno due istanze funzionalmente identiche che si interfacciano agli stessi sistemi di controllo ma che si alternano automaticamente nel caso in cui una subisca un malfunzionamento. Questo procedimento avviene in modo trasparente agli utenti.

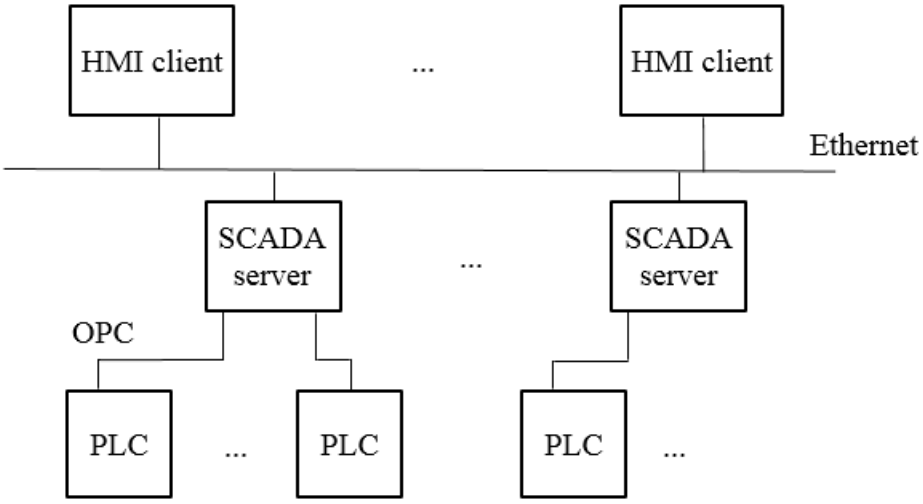

Figura 3. Architettura fisica distribuita

2.1.1 Sensori e PLC

In un apparato industriale il controllo del processo fisico avviene attraverso dispositivi di acquisizione delle grandezze fisiche di interesse e sistemi di attuazione tipicamente elettromeccanici. L'acquisizione dei dati sensoriali e l'implementazione della logica con cui questi vengono elaborati per generare i segnali di comando degli attuatori è affidata a sistemi di elaborazione chiamati PLC (Programmable Logic Controller) o soft–PLC. La caratteristica di funzionamento che entrambi questi sistemi devono garantire è il determinismo del tempo di esecuzione delle singole operazioni eseguite, cioè la caratteristica di funzionamento in tempo reale. La struttura hardware e software di un PLC è dedicata ad applicazioni di controllo automatico e garantisce proprietà di robustezza sia meccanica che rispetto a disturbi elettromagnetici, mentre i soft-PLC si basano su pc e sistemi operativi real-time ed hanno la caratteristica principale di garantire una maggior potenza di calcolo ed essere flessibili.

I PLC furono sviluppati fin dagli anni '70 con l'intento di introdurre soluzioni di controllo flessibili in modo da sostituire le logiche cablate basate su relè precedentemente impiegate. Con i PLC la programmazione sostituì le procedure di cablaggio abbreviando notevolmente i tempi di fermo delle macchine e aumentandone la produttività. L'impiego di PC in ambito industriale avvenne inizialmente per creare interfacce utente per l'interazione con i PLC. L'avvento dei controllori basati su PC è iniziato invece nei primi anni '90 conseguentemente alla riduzione dei costi e all'aumento delle prestazioni dei calcolatori.

I PLC e soft-PLC sono componenti fondamentali in un qualsiasi sistema di controllo industriale. In Figura 4 sono illustrati i principali componenti di elaborazione presenti in un sistema di automazione industriale.

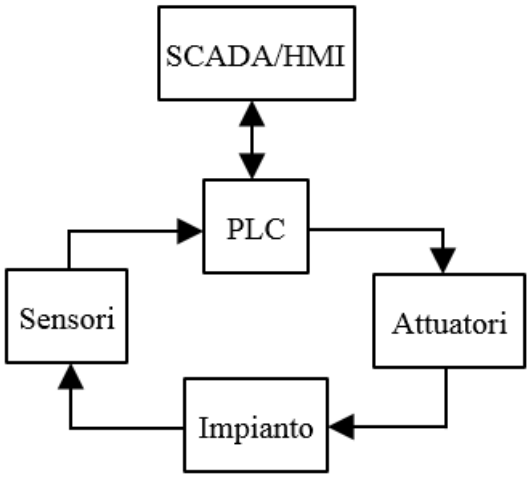

Figura 4. Principali componenti di un apparato di controllo e supervisione

Con l'utilizzo di un PLC tutte le funzioni di elaborazione dei dati sensoriali richieste per il controllo automatico di una macchina o di un processo industriale sono realizzate tramite l'esecuzione periodica di un programma.

La struttura fisica di un PLC è modulare, i componenti principali sono:

- Modulo di alimentazione.
- Modulo processore.
- Moduli di ingresso digitali ed analogiche. Tale componente permette l’acquisizione dei dati sensoriali, dei comandi provenienti dallo SCADA, e dei segnali inviati da altri PLC presenti in rete.
- Moduli di uscita digitali ed analogiche che permettono di modificare lo stato dei sistemi di attuazione del controllo e di inviare i dati relativi al processo allo SCADA.
- Dispositivi dedicati alla comunicazione con altri sistemi di elaborazione all'interno dell'impianto.

Il funzionamento del PLC è di tipo ciclico sequenziale, cioè esso esegue in modo ripetitivo le funzioni programmate. La struttura tipica del programma implementato prevede, oltre alla gestione asincrona di interrupt, quattro fasi distinte:

- Lettura ingressi;
- Elaborazione degli ingressi ed implementazione della strategia di controllo definita dal programma applicativo installato.
- Comunicazione con altri sistemi di elaborazione.
- Scrittura dei valori di uscita.

Ogni ciclo di funzionamento, chiamato ciclo di scansione, prevede l’esecuzione di queste fasi secondo determinate politiche, alcune delle quali sono:

- Sincrona di ingresso e di uscita, in cui l'elaborazione avviene solo dopo che tutti gli ingressi sono stati acquisiti e lo stato dei moduli di output viene aggiornato solo dopo che l'elaborazione di tutte le uscite è terminata;

Figura 5. Funzionamento sequenziale di un PLC

- Sincrona di ingresso e asincrona di uscita. In questo caso l'aggiornamento delle uscite avviene a mano a mano che il programma esegue l'elaborazione dei dati di ingresso.

- Asincrona di ingresso e di uscita: gli ingressi vengono aggiornati ogni volta che il programma incontra un'istruzione che obbliga il PLC ad acquisirne lo stato. Se il valore delle uscite viene modificato durante l'elaborazione, lo stato logico risultante viene immediatamente inviato al corrispondente modulo di uscita.

I principali parametri che definiscono le prestazioni di un PLC sono il tempo di ciclo e il tempo di risposta. Il primo denomina il tempo che intercorre tra due esecuzioni successive dell'applicativo eseguito dal PLC: è un dato fornito dai costruttori e il programmatore deve tenerne conto in particolare per definire la frequenza di campionamento dei dati in ingresso, e quindi l’ampiezza di banda a disposizione). Il tempo di risposta è definito come l'intervallo di tempo massimo che intercorre tra il verificarsi di un evento in ingresso al dispositivo e l'istante in cui avviene la reazione. Nel caso peggiore l'evento si verifica subito dopo che il controllore ha costruito l'immagine degli ingressi. In questo caso

saranno necessari un ciclo di programma per aggiornare l'immagine e un altro ciclo perché l'effetto venga inviato sul campo.

2.1.2 Comunicazione SCADA – PLC

Lo scambio di informazioni tra SCADA e PLC è bidirezionale e necessita di avvenire in modo affidabile e sufficientemente veloce in modo da non compromettere le prestazioni del sistema di supervisione. Come detto le variabili di processo monitorate sono acquisite regolarmente dai sistemi di campo. Per questo motivo il processo di acquisizione dei dati deve rispettare vincoli di real-time. Qualsiasi SCADA contiene una molteplicità di driver per interfacciarsi con PLC di produttori diversi. Il protocollo più in uso per la comunicazione tra SCADA e PLC è chiamato OPC (Open Platform Communication), uno standard dedicato alla definizione di una interfaccia di comunicazione tra applicazioni SCADA/HMI e i sistemi dedicati all'acquisizione dati e controllo real-time in ambito industriale. La specifica OPC è basata sulle tecnologie OLE, COM e DCOM sviluppate da Microsoft per i sistemi operativi Windows.

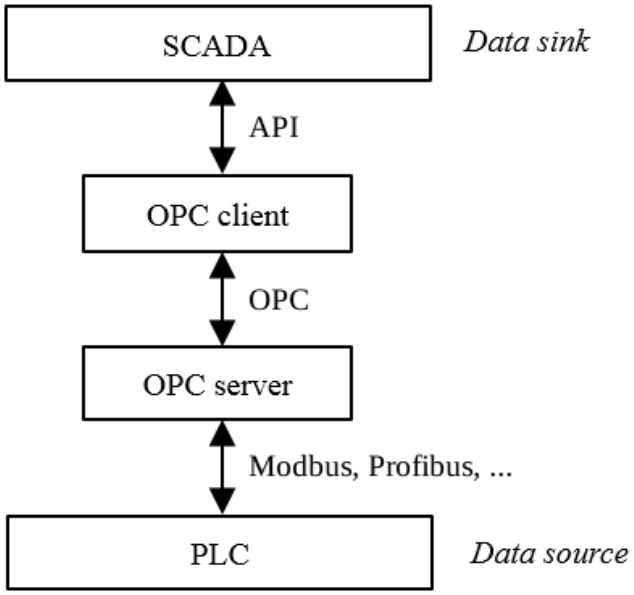

Figura 6. Architettura client-server di OPC

Prima dell'introduzione di OPC, gli sviluppatori di applicazioni software in ambito SCADA erano costretti a sviluppare driver di comunicazione specifici per scambiare dati con ogni distinto sistema di controllo. OPC ha invece messo a disposizione un'interfaccia comune per interagire con differenti prodotti di controllo industriale, indipendentemente dall'hardware e dal software utilizzati nel processo, come mostrato in Figura 6.

La struttura implementata dal protocollo OPC è di tipo client-server. Un qualsiasi processo client basato su OPC può accedere a un server sorgente di dati dotato di interfaccia OPC. In particolare:

- Un server OPC consente ai fornitori hardware di offrire ai propri acquirenti dei servizi che permettono a qualsiasi client di accedere alle loro apparecchiature.

- L'applicazione client controlla i dispositivi e gestisce i relativi utilizzando i metodi standard di accesso ad un oggetto OPC.

OPC segue perciò un'architettura client-server: un server OPC è un'applicazione software che raccoglie le informazioni dai dispositivi (PLC, DCS, ecc.) tramite protocolli nativi (Modbus, Profibus, ecc.), il server fornisce le funzionalità di accesso ai dati collezionati tramite oggetti COM; i client OPC leggono e scrivono i dati sul dispositivo di campo tramite il server OPC.

Le informazioni disponibili tramite un server OPC sono organizzate in gruppi secondo criteri di efficienza e i gruppi possono essere pubblici, cioè disponibili per qualunque client OPC, o locali, accessibili solo dal client che li ha creati.

Uno dei vantaggi più significativi nell'utilizzo di OPC, consiste nel fatto che l'applicazione non necessita di conoscere i dettagli dell'architettura interna del dispositivo con cui scambia i dati, ma esclusivamente i nomi dei gruppi e degli elementi a cui è interessata.

In ogni caso OPC non elimina la necessità dei driver di comunicazione: il costruttore sviluppa il server OPC specifico per il proprio prodotto, utilizzando il protocollo di comunicazione necessario per il funzionamento del suo dispositivo; a questo punto diventa più semplice interfacciare il sistema con altro software compatibile con OPC.

La prima formulazione dello standard nacque dalla collaborazione di alcuni fornitori di prodotti di automazione, riconosciuti a livello mondiale, e Microsoft. Lo standard venne chiamato “OPC Specification” ed era basato sulle tecnologie COM e DCOM di Microsoft. Le specifiche definiscono un insieme di oggetti, interfacce e metodi per semplificare l'interazione delle applicazioni di controllo di processo e di automazione della produzione.

E' importante capire che OPC non è un protocollo di comunicazione paragonabile a Ethernet o TCP/IP, ma rappresenta un livello di astrazione più alto: è costituito da un insieme di API che nascondono la rete di trasporto sottostante e la codifica utilizzata per lo scambio dei dati.

Allo stato attuale l’attenzione dei progettisti dello standard OPC si è focalizzata sulle aree comuni a tutti i venditori. Ulteriori funzionalità saranno definite nelle successive versioni. La versione attuale è focalizzata su tre differenti server:

- Online DataAccess: offre metodi per uno scambio efficiente di dati (read-write) tra un'applicazione ed un device per il controllo di processo.
    - Write sincrona/asincrona.
    - Read polling, report by exception, cache and device read.
- Allarmi e gestione degli eventi: meccanismi che permettano la notifica della occorrenza di eventi specifici e di condizioni d'allarme.

- Historical Data Access: strumenti per la lettura e l'elaborazione dei dati storici per analisi a posteriori.

Gli obiettivi di OPC possono essere sintetizzati nei seguenti punti:

- Flessibilità orientata alla compatibilità di funzionamento con sistemi progettati da diversi produttori.
- Definire funzionalità ad un alto livello di astrazione che prescinda dai dettagli implementativi dei protocolli di più basso livello.
- Efficienza della comunicazione tra sistema di controllo e infrastruttura di supervisione.

Le specifiche includono due tipi di interfacce:

- Custom Interface per l’utilizzo da parte degli sviluppatori dei server e dei client.
- Automation Interface come riferimenti ad un insieme di interfacce di OLE automation per supportare client sviluppati con applicativi business ad alto livello come Excel, Visual Basic, etc

Come già discusso, queste specifiche OPC non danno informazioni sull’implementazione delle interfacce ma solo sulle loro caratteristiche: tutte le funzionalità si basano su Application Programming Interfaces (APIs).

2.2 Architettura software

Le funzioni implementate da uno SCADA/HMI sono relative alla elaborazione dei dati provenienti dai dispositivi di campo e dagli input immessi dagli utilizzatori della macchina automatica. Con lo sviluppo tecnologico le piattaforme software SCADA hanno iniziato ad inglobare database real time

integrati in grado di gestire centinaia di modifiche al secondo mantenedo la conformità alle interfacce database standard come Open Database Connectivity (ODBC) e Object Linking and Embedding for Database (OLE DB).

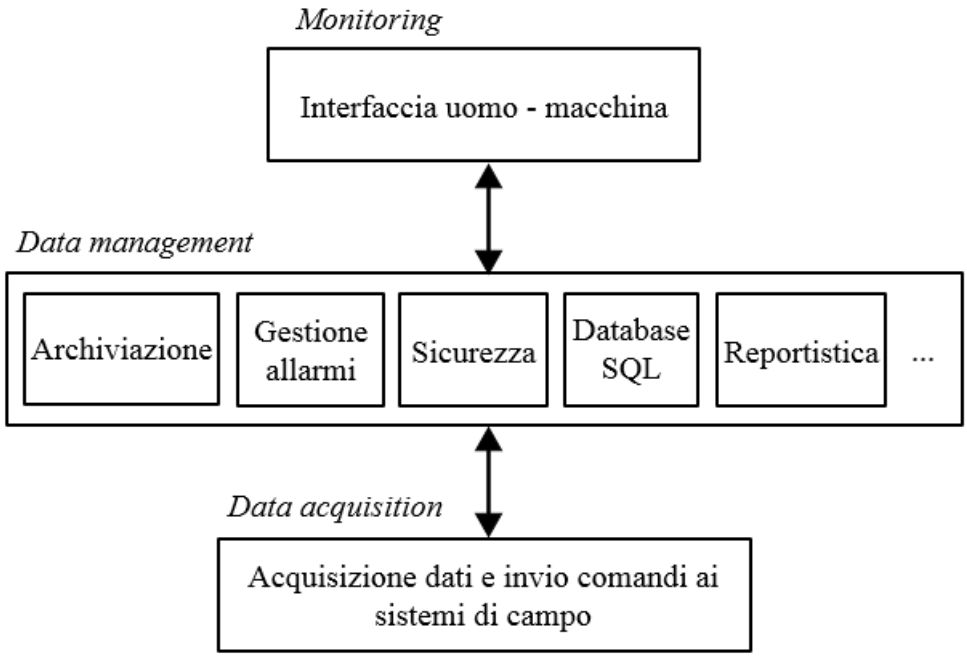

Figura 7. Principali funzionalità di uno SCADA

Le frecce bidirezionali presenti in Figura 7 indicano che il sistema garantisce sia la possibilità di acquisire i segnali provenienti dai sistemi di campo che campionano le grandezze fisiche, che la gestione di comandi e configurazioni inviate ad essi per modificare lo stato della macchina.

Il funzionamento di un sistema SCADA è basato sul processo di acquisizione dei dati provenienti dall'impianto: questo infatti è il legame tra il processo fisico monitorato e il sistema software che elabora ed archivia i dati sensoriali. Questa attività è implementata da un motore di gestione del flusso di dati tra gli apparati di controllo e il sistema di supervisione. Una delle caratteristiche più importante che questo software di acquisizione deve garantire è il determinismo della frequenza di acquisizione dei dati. Per questo motivo l'implementazione

del motore di acquisizione è basata su un database real-time. Nel caso di IFIX, un programma denominato SAC (Scan Alarm and Control) ha il compito di smistare i dati letti tramite il protocollo OPC e di indirizzarli verso un database real time chiamato PDB (Process DataBase) che contiene l'informazione continuamente aggiornata relativa alle caratteristiche fisiche di interesse dell'impianto monitorato. Ogni tag del PDB contiene l'informazione associata ad un particolare aspetto della macchina, come il valore di una grandezza analogica o lo stato di un allarme.

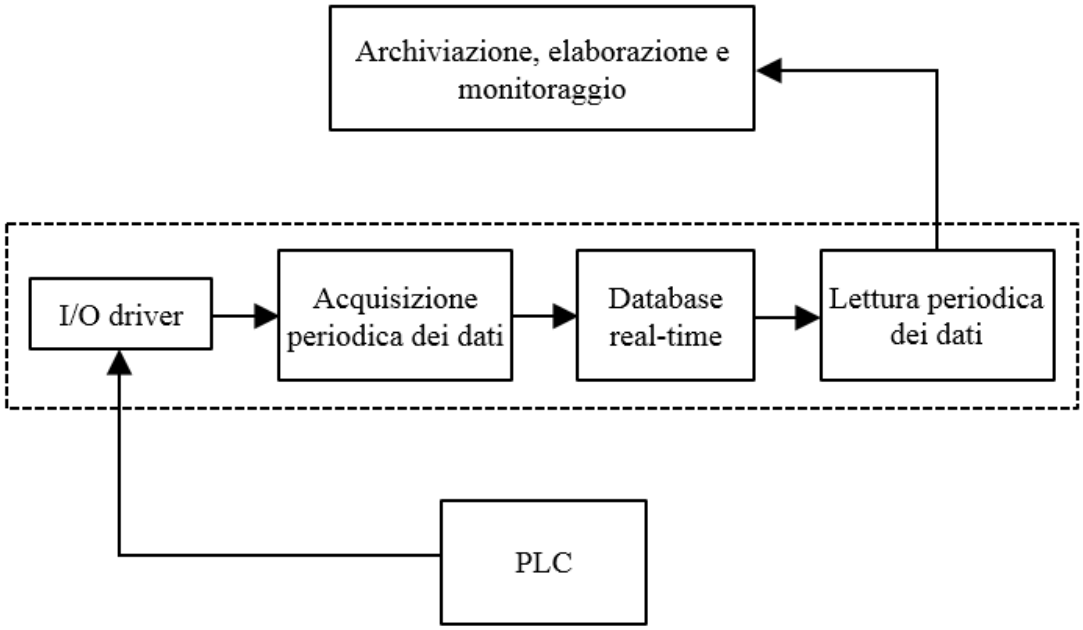

Figura 8. Componenti principali del sistema di acquisizione dei dati di uno SCADA

Il contenuto di tali tag, archiviato in un database SQL, viene inviato ai sistemi HMI per l'aggiornamento dei valori mostrati nel terminale di output.

Come già sottolineato, la maggior parte delle applicazioni eseguite in uno SCADA si basa sull'acquisizione dei dati ad intervalli regolari. Il software di acquisizione presente negli SCADA moderni permette al progettista di definire

la frequenza di campionamento e di elaborazione dei dati acquisiti in modo tale da bilanciare l'utilizzo di risorse di sistema e di migliorare le prestazioni dell'intero sistema. Il periodo di acquisizione minimo che il motore di acquisizione IFIX è in grado di garantire è di 0.05 secondi.

Un'elemento indispensabile per l'analisi dei dati è il database SQL, impiegato per l'archiviazione. Tale componente permette l'implementazione del sistema di gestione documentale attorno al quale questa tesi è incentrata.

Una ulteriore funzionalità tipicamente garantita da un sistema SCADA è la possibilità di introdurre script per ampliare le capacità del sistema. In IFIX ad esempio è possibile personalizzare le funzionalità standard tramite scrip in VBA (Visual Basic for Application). Tramite script è ad esempio possibile accedere agli oggetti grafici che compongono la HMI, eseguire query sul database real-time sul database SQL.

2.2.1 Proprietà di real-time

Con il termine real-time si riferisce alla capacità del sistema di controllo di reagire alle sollecitazioni del processo in tempi deterministici e in genere piccoli rispetto i tempi caratteristici della dinamica del processo. Le funzioni svolte da un sistema di controllo sono tali da rendere questa capacità di reazione un requisito irrinunciabile. Le soluzioni possono contemplare il contenimento dei tempi di reazione per insiemi ridotti di eventi generati dal processo, il sacrificio di funzioni complesse a favore di soluzioni semplici o l'adozione di soluzioni tecnologiche dedicate.

I sistemi di controllo di campo devono garantire determinismo del tempo di esecuzione delle operazioni che eseguono, questo per assicurare che la schedulazione di tali attività non generi ritardi indesiderati nelle operazioni di

controllo e quindi che il funzionamento dell'impianto si mantenga conforme alle aspettative.

È necessario garantire il rispetto di deadline anche per i sistemi di supervisione SCADA – HMI e ciò è motivato dal fatto che l'operatore deve poter essere avvisato tempestivamente su eventuali modifiche inattese dello stato interno della macchina.

L’evoluzione dei sistemi di calcolo avvenuta negli ultimi decenni è stata tale da rendere realistico il desiderio di realizzare un sistema real-time. Oltre allo sviluppo dei sistemi di calcolo, benefici nel mondo dell'automazione sono giunti anche grazie allo sviluppo della capacità dei sistemi di comunicazione a disposizione dei sistemi di controllo. Questi intervengono quando le dimensioni del processo sono tali da rendere necessario l’allestimento di sistemi di comunicazione complessi e di grandi dimensioni (reti geografiche).

Riguardo la valutazione delle prestazioni di un sistema di controllo sono significativi i parametri di affidabilità e di disponibilità. La capacità di operare correttamente con continuità risulta un'ulteriore caratteristica desiderabile rispetto quella di real-time.

2.3 Interazione uomo-macchina

La realizzazione delle funzioni di un sistema di supervisione e controllo comporta sempre la realizzazione di sottosistemi responsabili dell’interazione tra gli operatori e il sistema medesimo denominati interfacce uomo-macchina (HMI o Human-Machine Interface). La complessità dello sviluppo di una HMI è funzione del tipo di interazione richiesta mentre quest’ultima dipende dalle caratteristiche del processo controllato. L’interfaccia uomo-macchina può realizzare molti gradi di interazione comprendendo funzionalità di semplice osservazione dello stato di esercizio del sistema, nel caso di sistemi che

realizzano procedure completamente automatizzate, o funzionalità responsabili della esecuzione di procedure manuali gestite dagli operatori.

In casi analoghi a quello dei sistemi di automazione industriale le procedure automatiche sono responsabili dell'acquisizione dei dati e di un'eventuale primo trattamento degli stessi (un esempio è dato dalle procedure di validazione) mentre l'interfaccia uomo-macchina rende disponibili funzionalità per tipi di analisi dell'informazione altrimenti non realizzabili. Dal momento che sono previste forme di controllo oltre che di supervisione, risulta fondamentale la realizzazione di interfacce di facile utilizzo e di funzionalità accessorie necessarie alla comprensione dello stato di esercizio del sistema comprendenti la gestione della notifica degli allarmi e la visualizzazione di grafici relativi alle grandezze più rappresentative relative all'impianto.

Un ruolo fondamentale nei sistemi HMI è ricoperto dai quadri sinottici, cioè rappresentazioni grafiche schematizzate del sistema supervisionato in cui vengono messi in evidenza i valori delle grandezze fisiche più rilevanti per il monitoraggio del processo. In questo modo l'utente possiede una visualizzazione estremamente sintetica e costantemente aggiornata del funzionamento del processo ed è in grado di percepire eventuali segnalazioni di condizioni di funzionamento anomale relative a singoli componenti o a determinate aree.

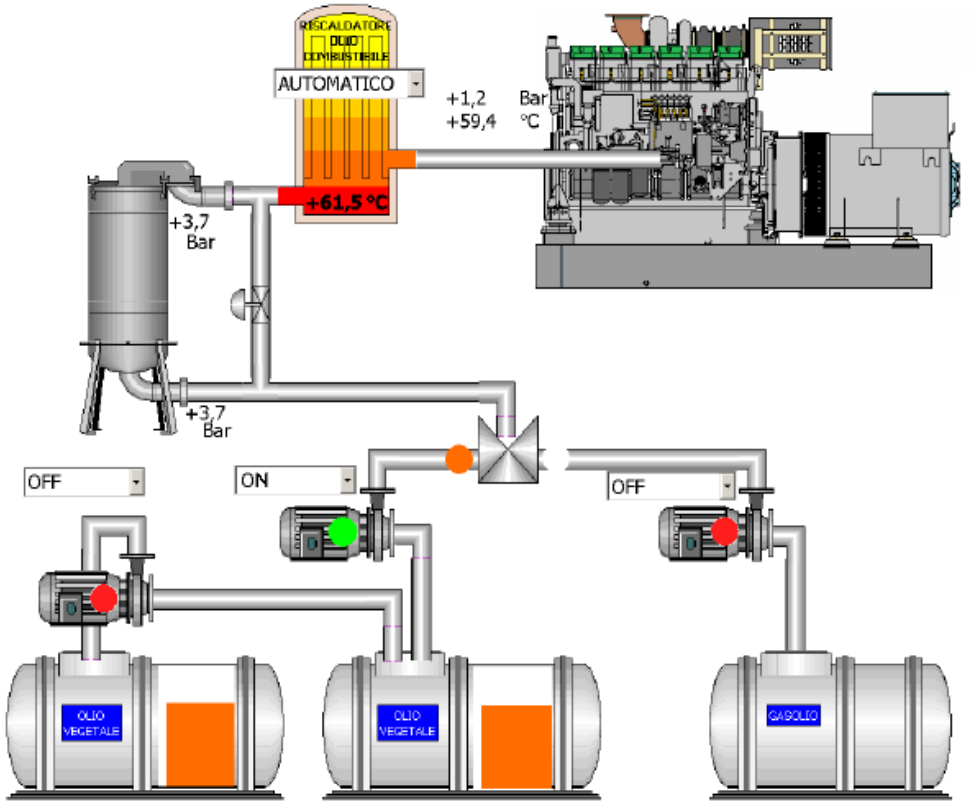

Figura 9. Esempio di quadro sinottico

I principali sviluppi futuri relativi all'interazione uomo-macchina in ambito industriale riguardano la gestione della HMI attraverso reti geografiche come Internet. L'impiego delle tecnologie web nell'ambito dell'automazione industriale rende possibile l'utilizzo di dispositivi mobili per la supervisione di impianti. La remotizzazione del monitoraggio di apparati industriali è un tema centrale nello sviluppo degli SCADA moderni. L'impiego di terminali mobili facilita e rende più efficace il processo di supervisione dell'impianto. Dal punto di vista del progettista, l'utilizzo di tecnologie web per la gestione del lato client in un apparato SCADA permette di fornire assistenza immediata da remoto, con conseguenti benefici in termini economici e di rapidità dell'intervento.

2.4 Gestione degli allarmi

Un obiettivo di un sistema di supervisione e controllo è la notifica immediata della presenza di condizioni di funzionamento anomale dell'impianto controllato. Anche se spesso situazioni particolarmente pericolose sono gestite in modo autonomo direttamente dallo SCADA proprio per garantire un intervento più rapido possibile, l'informazione riguardante gli allarmi deve essere messa in evidenza e in qualsiasi stato si trovi il sistema HMI, in modo tale da permettere un tempestivo intervento dell'operatore e di evitare conseguenze indesiderabili per gli operatori e per il sistema.

La natura particolare degli allarmi è tale che anche il trattamento che essi subiscono prevede una gestione particolare. Ogni volta che un allarme si genera l'operatore è chiamato ad effettuarne un riconoscimento. Riconoscere un allarme significa prendere visione della sua segnalazione e quindi della potenziale situazione di pericolo per il processo. Nel momento in cui la condizione anomala scompare e si ritorna al normale funzionamento il sistema segnala il ritorno da una condizione di allarme. Spesso la criticità delle situazioni che conducono alla generazione di un allarme o di un evento è tale che in ogni pagina che l'operatore ha a disposizione sono riportati gli ultimi allarmi/eventi accaduti. Spesso a chi realizza il sistema si chiede di evidenziare la condizione di allarme con particolari rappresentazioni (ad esempio il valore che lampeggia) oppure di portare automaticamente alla vista dell'operatore la pagina che contiene la lista degli allarmi attivi. Un'altra operazione che il sistema compie sulle informazioni di allarme è l'associazione con il tempo. Ogni volta che un allarme si genera, viene riconosciuto, viene cancellato, il sistema associa all'informazione l'istante temporale in cui ciò è avvenuto. Avere la possibilità di ordinare temporalmente gli allarmi generati dal sistema permette di ricostruire cronologicamente la storia degli accadimenti, cosa che spesso consente di capire l'evoluzione del processo controllato.

Altre funzioni che il sistema SCADA può mettere a disposizione per agevolare le operazioni di trattamento allarmi/eventi sono:

- Associazione di una classe, permettendo così il raggruppamento di allarmi simili che possono per esempio richiedere un uguale trattamento da parte di un operatore;
- Associazione di una priorità, aiutando così un operatore nella scelta di quale allarme trattare in maniera preferenziale;
- Associazione di permessi operatore, impedendo così ad operatori non idonei a gestire allarmi non adeguati alla loro mansione.

Come illustrato negli esempi, queste associazioni hanno quindi lo scopo di consentire una gestione evoluta degli allarmi sia in termini di classificazione degli stessi che di abilitazione degli operatori al trattamento.

2.5 Sicurezza

Il diffondersi delle tecnologie informatiche nell'ambito dell'automazione industriale ha portato a considerare sempre più attentamente i temi della sicurezza tipici del mondo IT, quali la protezione da accessi non autorizzati, la perdita di dati e la caduta di sistemi o di reti dovuti a software malevoli. La gestione del flusso informativo proveniente dall'interfaccia HMI riguarda anche aspetti legati alla garanzia di soddisfacenti livelli di sicurezza, implementati tramite il controllo degli accessi al sistema da parte di operatori, l'associazione di privilegi alle classi di utenti, e la produzione di file di log degli accessi e delle operazioni eseguite da ciascun utente sulla macchina.

Uno tra gli obiettivi di questi tipi di controlli è legato alla garanzia che il processo produttivo venga espletato nelle modalità predefinite, ed eventuali modifiche ai parametri della macchina siano state memorizzate ed archiviate

per un'analisi futura. Obiettivo del sistema di sicurezza non è solo impedire o individuare tentativi di manomissione malevoli, ma anche di evitare conseguenze negative causate da errori umani.

A partire dagli anni 2000 è stato definito lo standard ISA99 per la formalizzazione delle procedure implementative atte a delineare i criteri di realizzazione di un sistema di automazione industriale conforme ai requisiti di sicurezza informatica. Il concetto di sicurezza informatica nell'ambito dell'automazione industriale viene applicato in modo trasversale sia ai sistemi di controllo quali PLC o DCS, sia ai sistemi di supervisione. Le procedure descritte trovano campi applicativi nei sistemi in cui un mancato rispetto di tali linee guida porterebbe a situazioni di pericolo per l'incolumità degli operatori, alla perdita di informazioni riservate o a danni economici.

I principi su cui si basa il funzionamento di un sistema di sicurezza informatica, sia in ambito IT che nel campo dell'automazione risiedono nel garantire riservatezza, integrità e disponibilità dell'informazione. Di seguito vengono analizzati i principali aspetti che differenziano gli approcci adottati per la gestione della sicurezza informatica in campo IT e nell'automazione industriale:

- Rischi. In un'architettura IT la perdita dei dati o delle informazioni può incidere sul ritardo di qualche transazione oppure nel business. Ad esempio una ditta pubblicitaria che perde delle informazioni sui propri clienti può avere un grande danno economico, una banca che lascia trapelare informazioni riservate avrà certamente dei danni d'immagine. In una rete industriale, la sicurezza di un sistema può incidere anche sull'integrità fisica di un asset: un PLC fuori controllo potrebbe rendere l'intero sistema pericoloso per persone, risorse ed ambiente.

- Requisiti di disponibilità. Tipicamente il carico di lavoro in un sistema IT è maggiore durante l'orario di ufficio, inoltre è possibile gestire

eventuali reboot di sistema in caso di anomalie di funzionamento. In un complesso di produzione, invece, le macchine operano con continuità. In questo caso, fermare l'impianto equivarrebbe a fermare la produzione, con gravi conseguenze economiche ed interruzioni di servizio.

- Architettura di rete. Come è stato descritto, un impianto industriale può comprendere apparati di diversa natura collegati in rete. Questa varietà funzionale porta ad avere delle criticità differenti. La sicurezza in un sistema IT ha come scopo principale quello di proteggere i propri server. Nell'industria, invece, funzionalità client e server possono essere implementate sulla stessa macchina, questo introduce una maggiore complessità nell'implementazione delle politiche di sicurezza nelle reti. Inoltre, mentre in campo IT il throughtput non è un parametro critico, in un sistema di controllo non sono accettabili ritardi del funzionamento dei componenti ed è indispensabile garantire le prestazioni.

2.6 Gestione ricette

Nell'ambito dell'impiego di macchine automatiche come quelle prodotte da IMA, vengono utilizzati insiemi di parametri di configurazione necessari per definire le caratteristiche operative legate al tipo di funzionamento richiesto dalla macchina, cioè alle caratteristiche del processo che è necessario eseguire. Il valore di alcuni di questi parametri dipende dal tipo di prodotto processato, altri dipendono dalla macchina impiegata. L'insieme di questi parametri viene comunemente chiamato ricetta e qualunque SCADA offre funzionalità di gestione di ricette.

La cosiddetta ricetta consiste nella composizione di alcuni parametri essenziali da inserire nei sistemi di controllo della macchina per far avviare la produzione di un lotto. Lo SCADA deve permettere di definire quali siano tali parametri e i vincoli o condizioni associati ai loro valori. L'apparato di supervisione deve

inoltre permettere di salvare le ricette in archivio e di richiamarle quando necessario: un operatore può quindi salvare una ricetta per riutilizzarla in un altro momento.

Nell'esempio che segue viene illustrato il funzionamento delle ricette.

Valore:

Prodotto: X
Dipendenza1 a
Dipendenza2 b

In Archivio:

Dipendenza1= a Dipendenza1= c
Parametro1 = a1 Parametro1 = c1
Parametro2 = a2 Parametro2 = c2

Nell'esempio viene mostrato che il Prodotto è stato definito come combinazione della “dipendenza1” e della “dipendenza2”. In questo caso il prodotto “X” è formato da “dipendenza1” con valore “a” e “dipendenza2” con valore “b”. In archivio per ogni valorizzazione delle dipendenze vengono salvati tutti i valori dei parametri dipendenti. Nell'esempio è mostrato l'archivio per la dipendenza1. La dipendenza1 ha due possibili valori: “a” e “c”; per ognuno di essi i parametri che dipendono hanno valori differenti a seconda del valore della dipendenza (per la dipendenza1 = a, parametro1 = a1 e parametro2 = a2; per la dipendenza1 = c, parametro1 = c1 e parametro2 = c2). Se viene definito un nuovo prodotto “Y” che ha la dipendenza1 = c, i parametri parametro1 e parametro2 avranno automaticamente valori “c1” e “c2”.

2.7 Archiviazione

Una funzionalità essenziale in un sistema SCADA è la memorizzazione dei dati relativi a diversi aspetti del funzionamento della macchina automatica. Gli scopi della storicizzazione dei dati sono molteplici: ad esempio è tipicamente necessario analizzare l'andamento di grandezze fisiche da cui dipende la qualità del processo produttivo operato dalla macchina. Ad esempio è necessario poter verificare se la temperatura di un impianto di sterilizzazione sia scesa al di sotto di una soglia minima durante l'operatività dell'impianto. L'attività di un sistema SCADA comporta il trattamento di dati rappresentativi degli stati assunti dal processo nella sua evoluzione e delle condizioni di esercizio delle infrastrutture che costituiscono il sistema di controllo (sensori, attuatori, apparecchiature di acquisizione dati, sistemi di comunicazione, sistema di elaborazione). Le attività di supervisione e controllo fanno uso del database real-time nel quale sono contenuti i dati costantemente aggiornati inviati dai sistemi di campo.

Una volta divenuti inattuali i dati gestiti dal sistema non vengono perduti ma conservati in una base di dati. Una prima ragione per l'archiviazione dei dati è lo studio del processo: con i dati storici si può studiare l'evoluzione del processo nel tempo, creare associazioni causa-effetto, oppure studiare le dinamiche del sistema. Un'altra ragione è la documentazione storica: un caso frequente di archiviazione dettata dalla necessità di creare documentazione sul processo di produzione della macchina è quello nel quale norme, regolamenti o leggi prevedono esplicitamente la conservazione dei dati. Come verrà discusso nel Capitolo 3.1, nel settore farmaceutico esistono normative che regolamentano le modalità con cui vengono generati i documenti relativi ai dati relativi ad aspetti critici del processo.

Un'ultima ragione che giustifica l'archiviazione dei dati è legata al fatto che spesso i dati storici hanno interesse statistico e contabile: si consideri ad esempio una linea automatizzata in cui sono installate più macchine automatiche. Essa viene gestita da un sistema di controllo automatico e da un'attività umana responsabile della predisposizione delle condizioni di esercizio della linea. Un sistema di archiviazione e reperimento dei dati è in grado di fornire un insieme di informazioni fondamentali per l'analisi dell'attività della linea: numero di pezzi prodotti, tempi di funzionamento e di arresto, cause di eventuali interruzioni della produzione, livelli di materia prima contenuta nei serbatoi. La semplice osservazione di questi dati è in grado di fornire elementi di analisi importanti, analisi che diviene veramente efficace se si dispone, oltre che dei dati attuali, delle serie storiche. È evidente che lo studio di questo tipo di dati volto a comprendere le cause e trovare i rimedi alle interruzioni di produzione dovute a guasti è un'attività d'interesse tecnico mentre le analisi dei dati più propriamente legati alle quantità sono d'interesse gestionale.

2.8 Reportistica

Un componente essenziale in un sistema SCADA/HMI è l’apparato dedicato alla gestione documentale. I documenti prodotti, chiamati report, hanno lo scopo di aggregare in modo sintetico e facilmente visualizzabile le informazioni riguardanti il funzionamento del sistema supervisionato. Lo scopo è dare al personale responsabile una visione chiara del funzionamento dell’impianto tramite l'uso di tabelle e grafici. Tipicamente tali documenti presentano i risultati di analisi statistiche eseguite sulle sequenze di dati archiviati in modo da generare informazioni utili per la valutazione dell'operatività dell'impianto. L'importanza di sistemi di questo tipo si spiega anche considerando l'enorme quantità di dati gestiti da uno SCADA e soprattutto la loro eterogeneità: è necessario che i dati acquisiti durante il

funzionamento della macchina siano elaborati in modo da poter estrarre, analizzare e visualizzare le informazioni più rilevanti.

Un primo scopo di questo processo è legato alla possibilità di analizzare grandi sequenze di dati al fine di estrarre informazioni nascoste riguardo caratteristiche operative della macchina. L'impiego di strumenti di aggregazione di dati come grafici e tabelle facilita enormemente l'analisi delle prestazioni dell'impianto. Il tipo di valutazioni effettuate sui dati presentati da un report è tipicamente di tipo statistico.

Il tema dello studio dei dati relativi a uno SCADA è un argomento divenuto di estrema importanza negli ultimi anni. Lo sviluppo di tecniche di data mining avvenuto nel mondo dell'analisi di dati nel campo IT apre la strada a sistemi di estrazione di informazione nell'ambito dei sistemi di automazione industriale. Questo tipo di analisi ha lo scopo di portare alla luce conoscenza non deducibile da una analisi statistica dei dati: un campo applicativo riguarda la possibilità di implementare implementare una ricerca di relazioni causa-effetto per realizzare un'analisi predittiva di guasti con conseguente razionalizzazione dei costi legati alla manutenzione della macchina. Le tecniche di analisi di Big Data

In settori industriali relativi alla lavorazione di beni strettamente legati alla salute delle persone il controllo di qualità dei processi produttivi è una attività regolamentata da normative internazionali. In particolare per il settore farmaceutico, cosmetico ed alimentare esistono regolamentazioni che definiscono le linee guida da seguire durante la progettazione del sistema di supervisione con lo scopo di garantire la validazione della qualità dei processi produttivi. Particolare importanza è rivestita dal sistema di gestione documentale il quale deve sostanzialmente garantire che siano tracciabili tutte gli aspetti relativi al funzionamento dell’impianto che incidono sulla qualità del prodotto.

Questi aspetti e quelli relativi alle normative che regolano la gestione dei report verranno approfonditi nel corso del successivo Capitolo. In seguito saranno inoltre illustrati i metodi di definizione di report e i sistemi per la loro generazione.

Capitolo 3
Reportistica nel contesto della supervisione industriale

Nell'ambito dei sistemi informatici operanti nella supervisione di impianti industriali un ruolo estremamente importante è rivestito dai sistemi di produzione di documenti, detti “report”, che illustrano in modo sintetico informazioni rilevanti estratte dai dati acquisiti dallo SCADA durante il funzionamento dell'impianto. Come anticipato nel Capitolo 2 l'informazione elaborata ha grandi dimensioni ed è estremamente eterogenea: oltre ai valori provenienti dai sistemi di controllo di campo, si devono riportare anche informazioni legate alla sicurezza come audit trail, oppure ad eventuali eventi di allarme notificati.

Con lo sviluppo dei sistemi di monitoraggio elettronici, la produzione di report in modo automatico ha assunto un ruolo sempre più importante per la validazione dei processi industriali, soprattutto nel campo farmaceutico ed alimentare. Tipicamente infatti il sistema di controllo e quello di supervisione funzionano in modo indipendente mentre in ambito farmaceutico ed alimentare lo spegnimento dello SCADA comporta lo shutdown di tutto l’impianto

monitorato. Questo perché esistono normative che obbligano all'utilizzo di sistemi di archiviazione dei dati essenziali per il controllo della qualità dei prodotti lavorati dalla macchina.

Questo Capitolo chiarisce alcuni aspetti legati alla produzione di report in un sistema SCADA/HMI. In un primo momento sono analizzate le regolamentazioni sui sistemi di reportistica, infine si prendono in considerazione gli aspetti operativi legati alla gestione documentale.

3.1 Normative sulla produzione di report

Uno degli obiettivi di un sistema di generazione di report è la validazione del processo monitorato. Questo aspetto diventa estremamente importante nell'ambito della lavorazione nel settore farmaceutico ed alimentare, per il quale dalla qualità del funzionamento del sistema dipende la salute dei consumatori. Per questo motivo esistono normative generate da enti riconosciuti internazionalmente che definiscono le linee guida che devono essere seguite per poter implementare un sistema di reportistica che possa essere considerato idoneo alla valutazione della qualità del processo produttivo.

Lo scopo principale delle normative attualmente in vigore riguarda la possibilità di continuare a godere dei benefici di una gestione automatizzata dei dati gestiti dallo SCADA garantendo allo stesso tempo che il contenuto dei report sia validato: si pongono infatti i requisiti affinché una firma elettronica apposta a un report abbia la stessa validità di una firma manuale. Per questo motivo tali normative riguardano sia i requisiti minimi che devono essere soddisfatti dal sistema di produzione dei report, ma anche aspetti legati alla garanzia che i dati contenuti siano effettivamente certificati, come la firma

elettronica e i sistemi di audit-trail[2]. Nei casi in cui un sistema computerizzato sostituisca un funzionamento manuale, deve essere garantito che non si verifichi una diminuzione della qualità dei prodotti, del controllo di processo o dell'assicurazione della qualità.

In linea di massima la conformità a tali norme riguarda alcune procedure operative come il backup dei dati, gestione della sicurezza, procedure di validazione automatizzata. Ciascun sistema informatico interessato da tali leggi deve essere reso conforme.

3.1.1 CFR 21 parte 11

La normativa attualmente più riconosciuta per la verifica di conformità di sistemi di gestione documentale in ambito farmaceutico, cosmetico ed alimentare è denominata *CFR 21 parte II - "Electronic Records / Electronic Signature"*. Tale regolamentazione è stata prodotta dalla *FDA – Food & Drug Administration*.

CFR21 parte 11 riguarda il campo della produzione alimentare, farmaceutica, e biologica. In generale la normativa richiede l'esecuzione di controlli incentrati sulla qualità del processo produttivo e sulla sicurezza.

Dal momento che gli scopi di un sistema di reportistica sono di diversa natura, ad esso dovranno accedere persone con diversi ruoli. Gli aspetti operativi dell'impianto monitorato saranno considerati da operatori di macchina, l'analisi dei dati sarà condotta da analisti, mentre aspetti legati alla contabilità e logistica potrebbero essere analizzati da dirigenti. Al fine di rendere possibile questa distinzione di attività e allo stesso tempo evitare che vengano eseguite

2 Nell'ambito della sicurezza informatica i record denominati audit trail sono documenti che elencano le attività svolte sul sistema informatico dagli utenti che ne hanno il diritto di accesso ad esso.

operazioni da chi non ne ha diritto è necessario implementare un meccanismo di differenziazione dei permessi.

Considerati gli obiettivi della normativa, un sistema SCADA/HMI conforme con CFR21 parte 11 deve possedere alcuni componenti essenziali:

- Sistema di sicurezza centralizzato. Gli utenti possono accedere al sistema di monitoraggio solo dopo essersi autenticati.

- Differenziazione automatica dei ruoli. Ciascun operatore deve avere accesso alle sole funzionalità legate al suo ruolo. Ad esempio un operatore macchina non potrà avere accesso alle medesime funzionalità analitiche e ai medesimi dati a cui potrà invece avere accesso un'analista.

- Sistema standardizzato di reportistica. La normativa richiede esplicitamente "*La possibilità di generare delle copie accurate e complete dei record in forma elettronica, adatte a scopi di controllo/ispezione, rivisitazione e copia da parte dell'agenzia*". Si rivela quindi indispensabile la disponibilità di strumenti software in grado non solo di creare tabelle e grafici modificabili ed esplorabili, ma anche di contenere informazioni di gestione non necessariamente attinenti alla fase di analisi.

- Tracciabilità di ogni operazione; ciascuna operazione deve essere registrata e deve essere inserita, in forma di report standard, all'interno di un archivio storico consultabile in qualsiasi momento dagli amministratori del sistema o dagli enti certificanti. In altre parole, l'amministratore deve essere in grado di stabilire "chi ha fatto cosa, in quale istante, da quale macchina (e possibilmente per quale motivo)" in ogni momento.

3.1.2 Annex11

Un'altra risorsa per regolare l'utilizzo dei sistemi di gestione dati computerizzati in ambito farmaceutico è stato deliberato dall'Unione Europea con il nome *Annex 11 - "Computerized systems"*. Anche in questo caso l'obiettivo primario è la regolamentazione dell'utilizzo di procedure automatizzate tramite l'accertamento del buon funzionamento di tutti i componenti che supportano l'infrastruttura software. L'obiettivo rimane la salvaguardia della salute dei cittadini.

Uno degli argomenti introdotti in Annex 11 è la valutazione e gestione dei rischi. La gestione del rischio deve essere applicata a tutto il ciclo di vita del sistema informatico, considerando:

- La sicurezza dei pazienti;
- L'integrità dei dati;
- La qualità del prodotto;

I concetti di autenticazione degli accessi e differenziazione dei permessi sono in primo piano e vengono completati dalla definizione esplicita delle figure ai quali il documento fa riferimento: Process Owner, System Owner, Qualified Person, IT.

Annex 11 regolamenta gli aspetti legati a due fasi distinte del ciclo di vita del software: la fase di progetto e la fase operativa. In fase di progetto l'attività regolamentata è la convalida: le infrastruttura software deve essere convalidata. Si tratta di accertare il buon funzionamento di tutti i componenti che supportano una applicazione: la rete aziendale ed i suoi componenti, le apparecchiature di memorizzazione dove sono custoditi i dati ed i Database, i server che eseguono le applicazioni, le stampanti, i sistemi di back-up e/o archiviazione dei dati. Annex 11 esplicita i contenuti attesi per la descrizione del sistema; in particolare l'organizzazione fisica e logica (architettura

hardware e software); il flusso delle informazioni e le eventuali interfacce con altri sistemi o processi; gli eventuali prerequisiti hardware e software e le misure per la sicurezza.

3.2 Definizione e generazione di report

Le informazioni che costituiscono il contenuto di un report sono i risultati di query eseguite sul database SQL dello SCADA. Questi dati popolano elementi come grafici e tabelle che definiscono il layout grafico del report.

La struttura dei report è descritta in modo dichiarativo con il linguaggio RDL (Report Definition Language), uno standard basato su XML per la definizione di report costruiti su un database SQL. I vantaggi di utilizzare una grammatica XML per la definizione dei report sono evidenti: oltre a poter manipolare la struttura del report tramite funzioni standard di analisi sintattica per XML, è possibile anche progettare applicazioni per la creazione da zero di un report, scopo primario del lavoro di questa tesi.

I componenti grafici più comuni impiegati nella definizione di un report sono campi di testo, grafici e tabelle. I campi di testo hanno lo scopo di esplicitare il significato dei dati presentati, mentre tabelle e grafici permettono la visualizzazione immediata di tali informazioni. Questi elementi, denominati *items*, sono definiti nel nodo *<ReportSections>*. In questo nodo sono impostate anche le caratteristiche grafiche come dimensioni dei singoli item, posizione di questi all'interno del report, caratteristiche dello stile come font e colori, margini e padding.

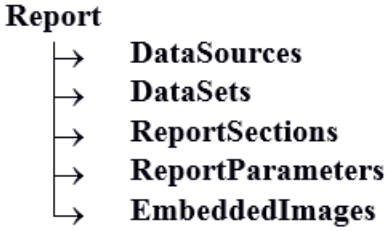

Figura 10. Nodi contenuti nel primo livello di un file RDL

Come mostrato in Figura 10, il primo livello di un file RDL ha una struttura tipicamente composta da cinque nodi:

- <*DataSources*> contiene i riferimenti ai database e alle tabelle dalle quali estrarre i valori richiesti;

- <*DataSets*> è l'elemento che contiene le query e i riferimenti ai parametri utilizzati da queste. L'informazione relativa a ciascuna query utilizzata nel report, ai suoi parametri e al riferimento del database associato è contenuta nei nodi *DataSet*, figli di *DataSets*;

- <*ReportSections*> contiene le definizioni degli elementi grafici, denominati item. Esempi di item sono grafici, immagini o tabelle;

- <*ReportParameters*>. In questo nodo si trovano i parametri utilizzati dalle query. Esistono due tipi di parametri: alcuni sono direttamente impostati dall'utente nel momento in cui si esegue il report, altri sono definiti da query. Questo secondo tipo di parametri deve indicare all'interno del relativo nodo il riferimento alla query necessaria per la sua valorizzazione. Per questo motivo un parametro può contenere il riferimento a un nodo dataset.

- <*EmbeddedImage*> contiene invece la codifica delle immagini eventualmente presenti nel documento.

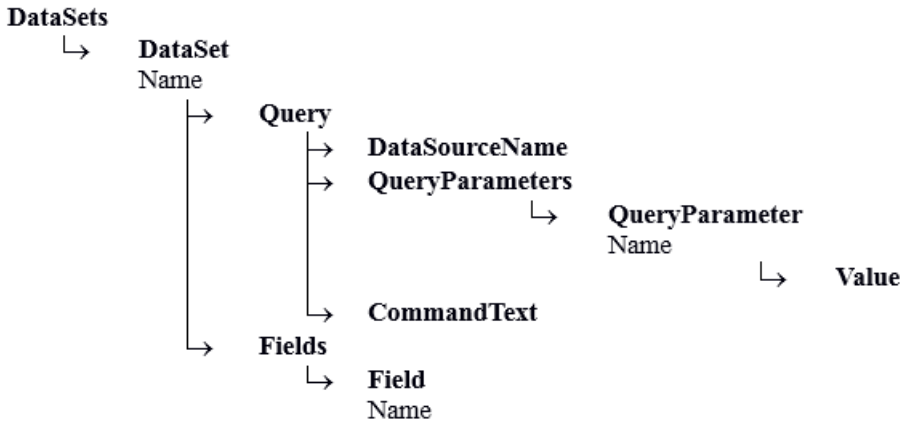

Figura 11. Struttura di un nodo DataSet

L'elemento *DataSets* viene utilizzato per identificare le informazioni necessarie per estrarre informazione dal database. In particolare le query vengono invocate sia nei nodi item che nei nodi parameters. Il nodo figlio *Query* contiene il nome del nodo *DataSource* necessario per identificare il database da cui estrarre i dati.

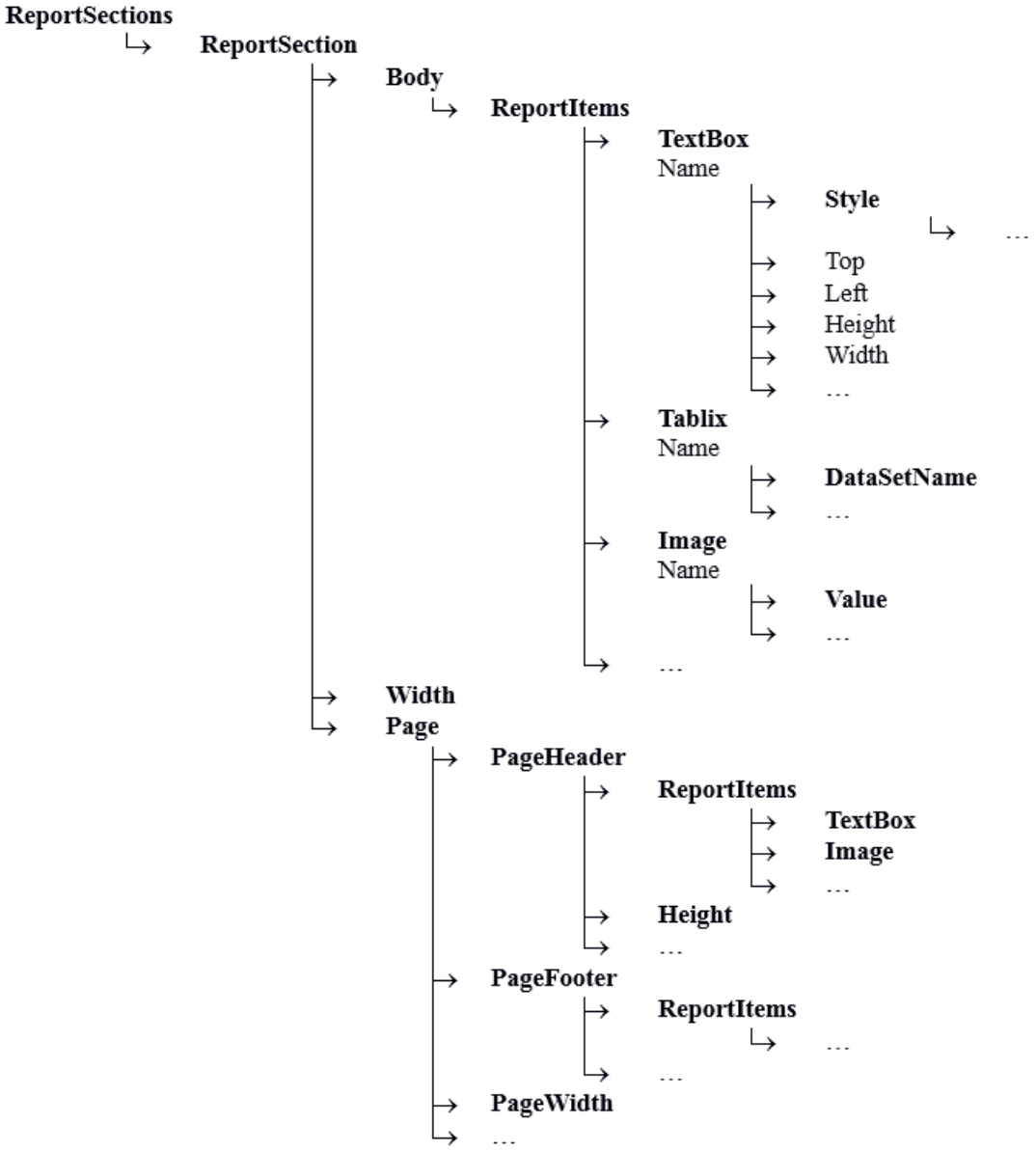

Figura 12. Principali elementi di un nodo ReportSection

ReportSection è il nodo che definisce il layout degli oggetti grafici, le query utilizzate per valorizzarli e l'impaginazione del documento.

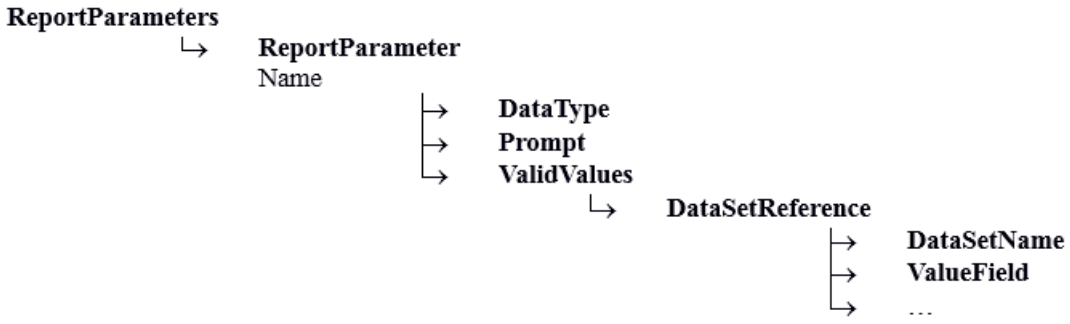

Figura 13. Elementi RDL del nodo ReportParameters

3.2.1 Tecnologie per la generazione di report

L’esecuzione di un file RDL, con lo scopo di generare un documento di report, consiste nella interpretazione del suo contenuto e nell'esportazione, in un determinato formato, del layout. Tale attività è affidata a una applicazione che estrae l’informazione necessaria da un database.

I contenuti mostrati all'interno di un report sono estratti dal database SQL implementato nello SCADA. L'immagine riportata sotto rappresentai componenti impiegati nel processo di generazione del report.

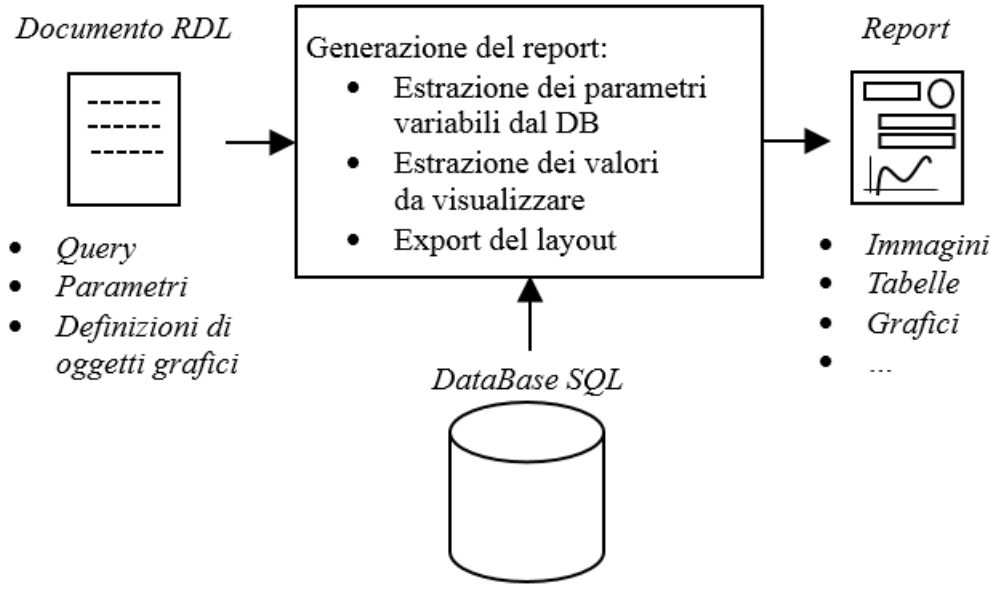

Figura 14. Processo di generazione dei report

Nell'ambito del progetto in esame il database utilizzato per l'archiviazione è "SQLServer" e lo strumento impiegato per la generazione dei report è chiamato "SQL Server Reporting Services". Dal momento che esistono parametri che possono essere definiti direttamente dall'utente, questa applicazione prevede un'interfaccia per l'inserimento di valori di configurazione.

3.3 Reportistica riconfigurabile

Allo stato attuale la definizione dei report eseguiti dal sistema di gestione documentale viene inserita all'interno dello SCADA dal progettista: in fase di configurazione del sistema di reporistica vengono realizzati i file RDL per i report relativi a specifiche aree di interesse. La struttura dei report creati può certamente essere accordata con il cliente, ma per quanto detto risulta "cablata" nello SCADA della macchina automatica. Ciò significa che il personale utilizzatore dell'impianto non può personalizzare i report se non richiedendo un intervento dell'assistenza tecnica o agendo direttamente sui linguaggi utilizzati

per la definizione dei report: RDL e SQL. Nasce quindi la necessità di creare uno strato di astrazione software che permetta di personalizzare la struttura dei report senza agire direttamente sulle tecnologie utilizzate per la loro implementazione.

Il progetto di un framework per la definizione di report tramite interfaccia grafica ha due obiettivi. In primo luogo si intende aumentare la flessibilità del prodotto SCADA/HMI. Si vuole infatti offrire la possibilità agli utilizzatori del sistema di monitoraggio di poter personalizzare la struttura dei report prodotti, sia nella grafica che nei contenuti. In questo modo si creano indubbiamente benefici a vantaggio dell'utilizzatore derivanti dal fatto, ad esempio, che il contenuto informativo richiesto può essere aggregato in un unico documento anche se relativo a diverse aree tematiche.

Un secondo beneficio di questa applicazione stà nella automatizzazione del processo di generazione di report in fase di configurazione dell'apparato di reportistica. L'impiego di un'editor di report a un livello di astrazione superiore rispetto gli editor di RDL rende la configurazione più semplice e veloce. Un aspetto rilevante in questo caso è dato dal fatto che l'impiego di una interfaccia grafica rende possibile la modifica dei report anche a personale non esperto.

Capitolo 4
Realizzazione di un sistema di reportistica riconfigurabile

In questo capitolo vengono descritti i passi percorsi per la sintesi del sistema software dedicato alla gestione dei report tramite interfaccia grafica. Il sistema viene concepito con l'intento di essere integrato in un apparato di monitoraggio di un impianto industriale. Come già anticipato lo scopo di questa tesi è la sintesi di un framework per la realizzazione di un sistema di reportistica riconfigurabile che permetta al progettista software di definire i report in modo automatizzato e al tempo stesso garantisca la possibilità agli utilizzatori della macchina automatica di personalizzare la struttura dei report prodotti, sia nell'aspetto che nei contenuti.

Lo sviluppo dell'applicazione ha seguito le fasi tipiche di un processo software strutturato:

- Definizione delle specifiche ed analisi dei requisiti;
- Progettazione dell'architettura;

- Sviluppo del codice;
- Test e convalida.

Trattandosi di un progetto dai contenuti piuttosto innovativi nell'ambiente SCADA/HMI le fasi del processo software non sono state seguite in modo strettamente sequenziale: invece di adottare una metodologia a cascata è stato impiegato un'approccio di sviluppo a spirale, illustrato in Figura 15. Secondo questo modello viene inizialmente definito un insieme relativamente ristretto di requisiti funzionali. Con il procedere del progetto della struttura software e con lo sviluppo sono stati ricercati ulteriori specifiche riguardanti nuove e sempre più sofisticate funzionalità. Ad esempio, durante lo sviluppo delle funzioni di editing, si è scelto di aumentare il livello di dettaglio a cui rendere possibili le modifiche degli elementi nel report.

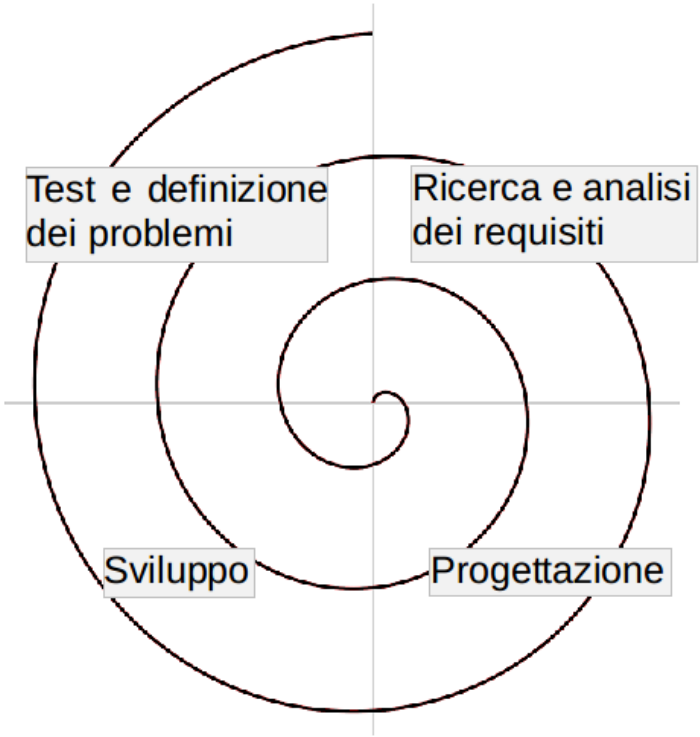

Figura 15. Sviluppo ciclico del progetto

Il principale vantaggio di questo approccio evoluzionistico consiste nella possibilità di sviluppare le specifiche in modo incrementale, a mano a mano che si comprendono meglio le potenzialità del sistema in esame. La prima delle quattro fasi che si susseguono nel processo software consiste nell'analisi di un insieme di requisiti generali che permettono di definire le funzionalità del sistema iniziale, via via perfezionate grazie allo studio di nuovi requisiti. Questo tipo di sviluppo del progetto non ha avuto un impatto negativo sui tempi di realizzazione in quanto è risultato relativamente semplice modificare in modo incrementale la struttura e l'implementazione del programma.

La presenza della fase di test nello sviluppo a spirale rappresenta la necessità di eseguire verifiche periodiche dello stato dello sviluppo dell'applicazione. In questo modo è possibile individuare i requisiti non pienamente soddisfatti o capire meglio quali nuove funzionalità risultano necessarie. Un approccio di questo tipo richiama il processo gestionale denominato “Stage gate”, tipicamente adottato per la supervisione dell'andamento di progetti innovativi.

4.1 Analisi dei requisiti

L'analisi dei requisiti comprende lo studio sia del servizio atteso dall'applicazione, che dei vincoli che riguardano la realizzazione del sistema. Di seguito vengono analizzati i requisiti funzionali, cioè quelli relativi ai servizi che il sistema deve garantire all'utilizzatore e che sono offerti da singoli componenti software. Successivamente sono discussi i requisiti non funzionali, che indicano invece caratteristiche più generali del software e non sono riferiti ad un particolare componente. Tra questi assumono estrema importanza, relativamente all'aspetto interattivo dell'applicazione, i requisiti di usabilità, intesa come facilità e soddisfazione riscontrati dall'utente nell'impiego del software.

Come è già stato discusso alcune di tali specifiche sono state definite come base di partenza per la progettazione software, mentre altre sono state ricercate ed analizzate in seguito, in modo da modificare incrementalmente le funzionalità offerte.

4.1.1 Requisiti funzionali

Le specifiche relative alle funzionalità implementate dal sistema riguardano la realizzazione di un framework per la sintesi e la modifica, tramite interfaccia grafica, di file di definizione di report definiti in linguaggio RDL e basati su database SQL. Il sistema deve operare all'interno di un sistema SCADA/HMI di una macchina automatica IMA e deve presentare quindi un'interfaccia grafica che permetta l'interazione con l'utente. In particolare il sistema deve fornire un editor grafico che permetta all'utente di comporre e personalizzare la struttura grafica e deve contenere un componente per la generazione del codice RDL necessario alla generazione del report. Tra le specifiche operative viene anche inclusa la necessità di aggiungere elementi direttamente definiti dall'utente.

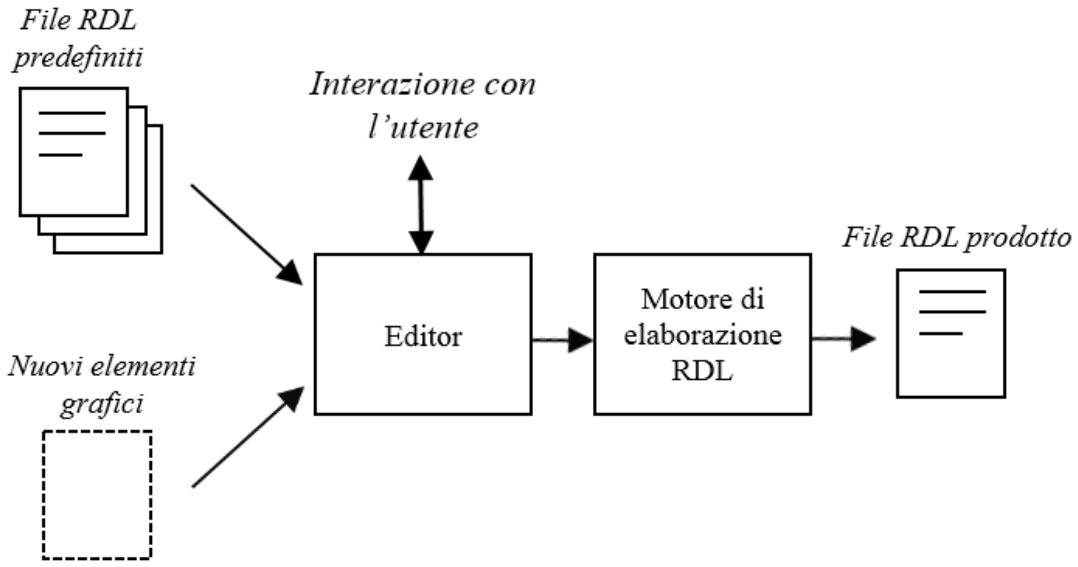

Figura 16. Principali blocchi funzionali

Come detto tali requisiti sono stati definiti in modo da poter implementare una applicazione che apporti due principali benefici: dal punto di vista dello sviluppatore del sistema SCADA/HMI l'impiego di un interfaccia per la generazione dei report riduce il tempo di sviluppo, mentre l'utilizzatore finale del sistema di reportistica acquista la possibilità di personalizzare il layout e il contenuto dei report.

Dall'analisi delle specifiche fin qui descritte è stata delineata l'architettura di massima del sistema software, mostrata in Figura 16. In particolare sono individuati due blocchi funzionali: uno relativo alla fase di editing del report, l'altro associato alla creazione automatica del file RDL con gli item inseriti dall'utente.

Dal momento che si vuole creare un livello di astrazione superiore rispetto ad un editor di RDL, è stato preferito far sì che l'utente possa selezionare elementi composti da uno o più item grafici relativi a una area tematica. Tali insiemi di item sono stati chiamati “sezioni”. Per quanto riguarda l'aspetto grafico, l'utente ha la possibilità di personalizzare il posizionamento delle sezioni, senza avere la possibilità di modificare il layout dei singoli item. Questo rende indubbiamente lo strumento di editing di più immediato utilizzo. L'identificazione delle sezioni è basata sui nomi dei nodi item. Tali nomi vengono infatti ridefiniti aggiungendo un prefisso identificativo della sezione di appartenenza.

Le principali funzionalità implementate dal componente “Editor” illustrato in Figura 16 sono:

- Analisi dei file RDL predefiniti presenti nel sistema. L'analisi condotta in questa fase ha lo scopo di individuare le macchine presenti nella linea, i report associati a ciascuna di queste e di identificare le sezioni

presenti in ciascun report. In questo modo si permettere all'utente di navigare tra le sezioni inseribili nel file di output.

- Costruzione degli item inseriti dall'utente. L'interazione con l'utente prevede che si possano inserire nuove sezioni, cioè sezioni non presenti nei report predefiniti.

- Implementazione dell'interazione con l'utilizzatore. Questo servizio comprende le tipiche operazioni di editing di file grafici: selezione e rimozione di sezioni, inserimento di margini al report, allineamento delle sezioni inserite, … .

4.1.2 Requisiti non funzionali

I principali requisiti relativi a proprietà generali del software riguardano caratteristiche tipicamente richieste nell'ambito dell'ingegneria del software: efficienza, affidabilità, portabilità. Un ulteriore obiettivo richiesto è la facilità d'uso dell'interfaccia utente. Questa caratteristica è estremamente importante nell'ambiente industriale in cui è necessaria una interazione efficace tra l'utente e la macchina.

I requisiti non funzionali tenuti in considerazione sono dunque:

- Efficienza;

- Affidabilità e robustezza;

- Portabilità;

- Manutenibilità ed evolvibilità;

- Usabilità.

Tali specifiche non sono soddisfatte da un particolare componente software, ma sono collegabili all'intero sistema. Questo risulta un'ulteriore motivo per cui è necessario intervallare le fasi di sviluppo e progettazione con fasi di test dell'applicazione. È infatti necessario testare periodicamente sia le funzionalità dei singoli componenti software che le prestazioni dell'intero apparato.

4.1.3 Requisiti di usabilità

Tra le proprietà generali richieste da un sistema software riveste particolare importanza in ambito dei sistemi SCADA/HMI la caratteristica di usabilità dell'interfaccia. Nel contesto dell'automazione industriale il sistema software è lo strumento con cui gli operatori eseguono operazioni sulla macchina e acquisiscono informazioni sul suo stato. Risulta dunque indispensabile che le funzioni eseguibili sull'impianto siano facilmente accessibili tramite un'interfaccia grafica intuitiva e le cui funzioni siano immediatamente comprensibili. In sostanza si richiede che tutte le operazioni messe a disposizione dall'applicazione siano facilmente utilizzabili senza che sia necessario che l'operatore venga istruito per eseguirle.

L'usabilità dell'applicazione riguarda quattro componenti presenti in qualunque contesto lavorativo:

- Gli utenti.
- Un insieme di compiti che devono essere svolti dagli utenti.
- Le funzionalità offerte dal sistema software.
- L'ambiente in cui il sistema viene utilizzato.

L'obiettivo principale è che le funzionalità possano essere sfruttate in modo veloce ed efficace, per migliorare la qualità del servizio offerto e per aumentare

la soddisfazione dell'utilizzatore. È ovviamente richiesto che le funzionalità offerte siano effettivamente correlate alle esigenze degli operatori nel contesto in cui queste vengono sfruttate e in questo senso l'analisi dell'usabilità dell'interfaccia grafica fornisce linee guida per la progettazione dell'intera applicazione.

Un ulteriore aspetto da tenere in considerazione per valutare soluzioni progettuali riguardanti l'usabilità dell'applicazione è l'ambiente in cui questa viene impiegata. In ambito industriale il sistema software è un componente dell'intero impianto e gli operatori impiegano il loro lavoro anche per la gestione di altri apparati. Risulterebbe quindi difficilmente praticabile fornire una formazione approfondita sull'utilizzo dell'applicazione: l'utente deve potersi concentrare esclusivamente sul suo compito e l'interfaccia grafica deve rendere la tecnologia sottostante trasparente all'utilizzatore.

Le soluzioni adottate per soddisfare i requisiti di usabilità a livello di sistema HMI, come l'impiego di monitor touchscreen, sono già state descritte nel Capitolo 2.4. Per quanto riguarda l'applicazione in esame l’aumento della facilità d’uso è ricercata attraverso diverse soluzioni:

- Viene inserita un'area in cui è mostrato il layout del report che si sta costruendo.
- Sono implementati meccanismi di drag & drop per la gestione delle sezioni.
- L'utilizzo di una barra di stato per la visualizzazione di commenti esplicativi sulle funzionalità offerte.

L'utilizzo di drag & drop permette di facilitare la disposizione degli elementi selezionati all'interno dell'anteprima del report.

Per quanto riguarda la progettazione di strumenti di editing di file grafici come quello che si vuole realizzare in questo progetto, un requisito essenziale riguardo l'usabilità, e in particolare la soddisfazione dell'utente, è il concetto denominato "What you see is what you get". Tale approccio al progetto di un sistema software prevede che venga fornita all'utente un'anteprima dell'oggeto grafico che sta realizzando. In questo modo in ogni istante l'utilizzatore possiede un'idea chiara sul layout di ciò che sta costruendo. Ovviamente fornire questa funzionalità è di grande rilevanza per ogni applicazione che incorpori un editor di file che definiscono aspetti di presentazione di dati tramite elementi grafici. In questo senso può essere conveniente implementare a livello di "View Model" un componente. La soddisfazione di questo requisito richiede uno sforzo orientato allo sviluppo del codice necessario all'interpretazione del report parziale da mostrare nell'anteprima dell'editor.

4.2 Progettazione

Il processo di progettazione del sistema ha l'obiettivo di definire i componenti software e le relative funzionalità. In questa fase non vengono tenute in considerazione nel dettaglio le caratteristiche implementative ma solo le funzionalità necessarie per soddisfare i requisiti descritti nel Capitolo 4.1. In particolare vengono affrontati i seguenti passi:

- Suddivisione dei requisiti ed identificazione dei sottosistemi.
- Specifica delle funzionalità di ciascun componente.

Risulta evidente come l'analisi dei requisiti sia collegata alla progettazione dei sottosistemi che compongono l'intero sistema: ogni componente è creato con lo scopo di soddisfare una particolare specifica. La definizione del comportamento e delle interfacce di ogni sottosistema permette di completare la definizione dell'architettura del software.

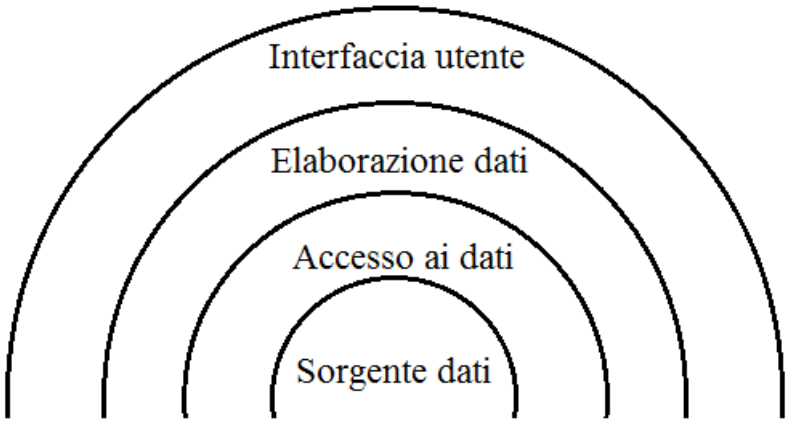

Figura 17. Struttura a livelli del sistema software

4.2.1 Identificazione dei componenti

In fase di identificazione dei sottosistemi si definiscono i blocchi funzionali che cooperano per garantire che il sistema complessivo fornisca i servizi attesi. Risulta inoltre conveniente stabilire le funzioni che ogni componente mostra agli altri in modo tale da semplificare il processo di connessione dei singoli componenti software per implementare cooperazione tra gli stessi. La figura riportata sotto illustra l'architettura stratificata che è stata definita durante la progettazione del sistema.

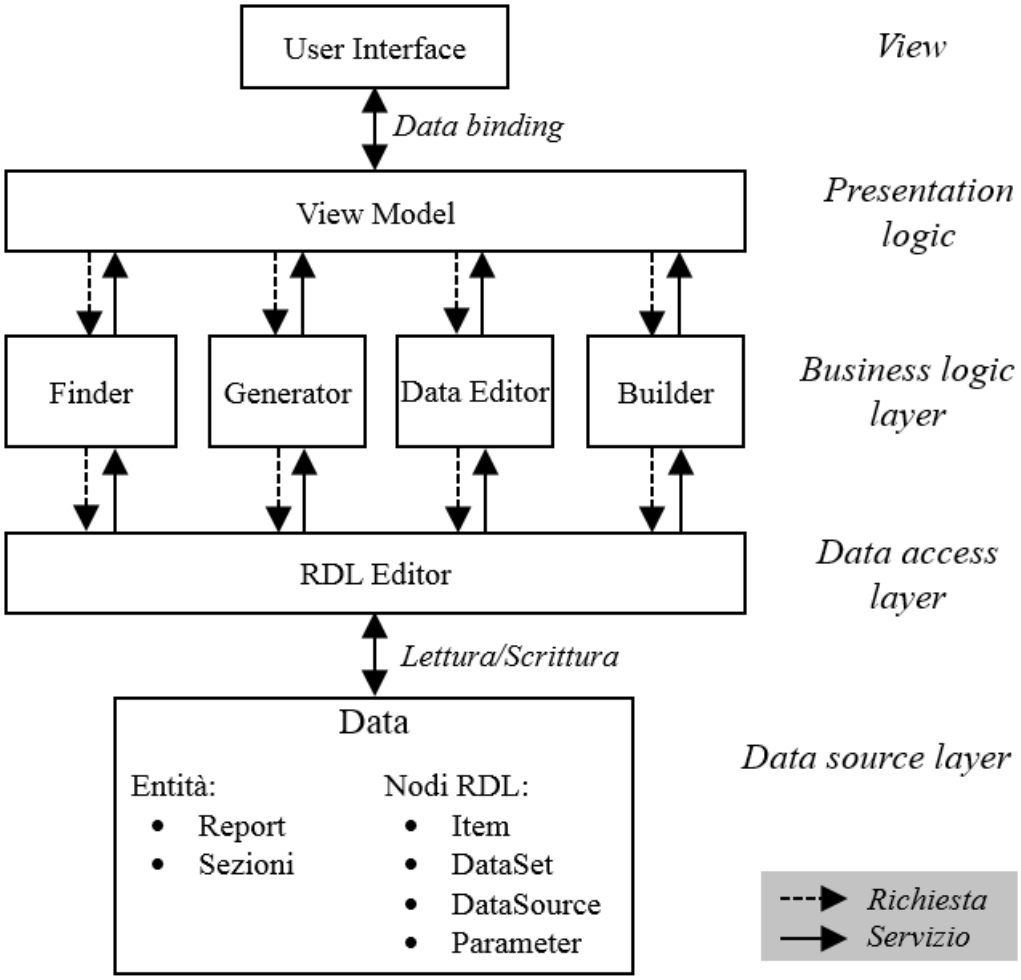

Figura 18. Principali componenti software di IMAReportBuilder.

L'individuazione degli oggetti idonei alla costruzione dei componenti è il passo successivo alla definizione dei sottosistemi eseguita grazie all'analisi preliminare dei requisiti. In questa fase è conveniente impiegare diagrammi delle classi per delineare la struttura interna di ciascuno dei componenti mostrati in Figura 18.

La rappresentazione dei dati gestiti dall'applicazione tiene conto della distinzione tra entità con caratteristiche più concrete come report e sezioni e i nodi RDL. Nella prima categoria ricade anche la rappresentazione del concetto

di macchina, anche se tale entità non possiede caratteristiche definite in RDL. Per quanto riguarda i nodi, questi vengono incapsulati in classi che forniscono metodi di lettura di alcune caratteristiche peculiari del nodo stesso come ad esempio la posizione o le dimensioni.

Il componente Manager permette l'applicazione di funzioni di lettura e scrittura di elementi RDL. In questo si definiscono ad esempio le funzionalità di ricerca di elementi in file RDL e di scrittura o rimozione di nodi.

Lo strato “business logic” definisce gli elementi che implementano le funzionalità su cui si fonda il sistema. Esso è costituito da quattro componenti:

- Finder.
- Generator.
- Data Editor.
- Builder.

Finder è composto dalle classi che implementano l'identificazione delle macchine supervisionate, dei report relativi a ciascuna macchina e all'individuazione delle sezioni contenute in ciascun report.

Il componenete Generator ha il compito di costruire nuove sezioni in base agli input forniti dall'utente. Crea l'elemento selezionato con i valori passati e produce un report fittizio per contenere l’item stesso e tutti i nodi RDL necessari alla sua visualizzazione. Ad esempio, se l'utente esegue l'inserimento di un'immagine, è necessario che oltre all'item associato all'immagine selezionata venga costruito anche l'elemento “EmbeddedImage” che contiene la codifica dell'immagine stessa. La costruzione di nuovi item grafici viene eseguita a partire da un template del relativo nodo RDL, cioè da un nodo in cui

i campi che ne definiscono le proprietà non sono valorizzati: il sottosistema di generazione si occupa di aggiornare il template con i dati inseriti dall'utente.

Le operazioni di editing eseguite dall'utente sull'interfaccia grafica vengono propagate sui dati dal componente Data Editor. Queste funzionalità sono relative al posizionamento delle sezioni all'interno del report o alla personalizzazione della query associata ad un item.

Il sottosistema Builder è dedicato all'implementazione del documento RDL in output. La costruzione del file viene avviata dall'utente tramite un pulsante di avvio e non richiede interazioni con l'utente, essendo completamente automatica. In questa fase il sistema valuta le sezioni che sono state inserite in fase di editing e procede all'inserimento dei nodi item nel file di output. Successivamente vengono inseriti anche i nodi che non rappresentano informazioni grafiche: per ogni item selezionato vengono letti tutti i nodi necessari all'esecuzione del report.

Il componente ViewModel realizza le funzionalità di gestione dell'interfaccia grafica in entrambi i versi: invia i comandi immessi dall'utente e aggiorna i valori visualizzati. L'aggiornamento dei dati nella View viene realizzato con meccanismi di data binding, realizzati sfruttando l'interfaccia "InotifiyPropertyChanged", che ViewModel implementa. Maggiori dettagli riguardanti la gestione dell'interazione con l'utente verranno discussi nel capitolo dedicato alo sviluppo. Tra i dati elaborati dall'applicazione ne esistono alcuni che presentano caratteristiche grafiche mostrate dall'interfaccia e che quindi devono essere gestiti direttamente dal sottosistema ViewModel. Un esempio sono le stesse sezioni, che possono essere selezionate e posizionate nell'area di anteprima tramite drag&drop. Per questo motivo è necessario che anche le classi utilizzate per rappresentare i dati implementino l'interfaccia "INotifyPropertyChanged". Per mantenere separazione tra i componenti che rappresentano i dati e quelli che gestiscono le interazioni con l'utente vengono

create delle classi wrapper che inglobano le classi “Data” (nodi RDL, sezioni, report) ma aggiungono le informazioni grafiche necessarie alla gestione del layout e implementano l’interfaccia di notifica di eventi.

I due diagrammi delle classi illustrati di seguito illustrano l'approccio impiegato per la rappresentazione dei dati. Il primo riguarda i nodi RDL, mentre il secondo le entità macchine, report e sezioni. Il punto di collegamento tra le due rappresentazioni consiste negli oggetti di classe “item”, componenti atomici che costituiscono le sezioni.

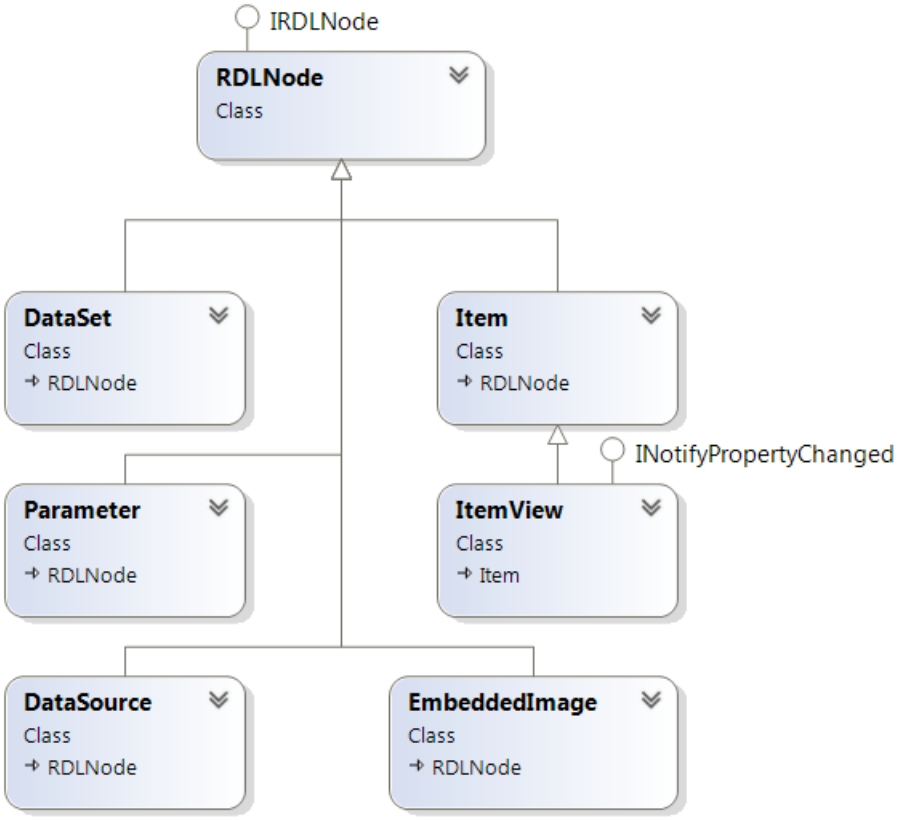

Figura 19. Diagramma delle classi relative alla rappresentazione dei nodi RDL

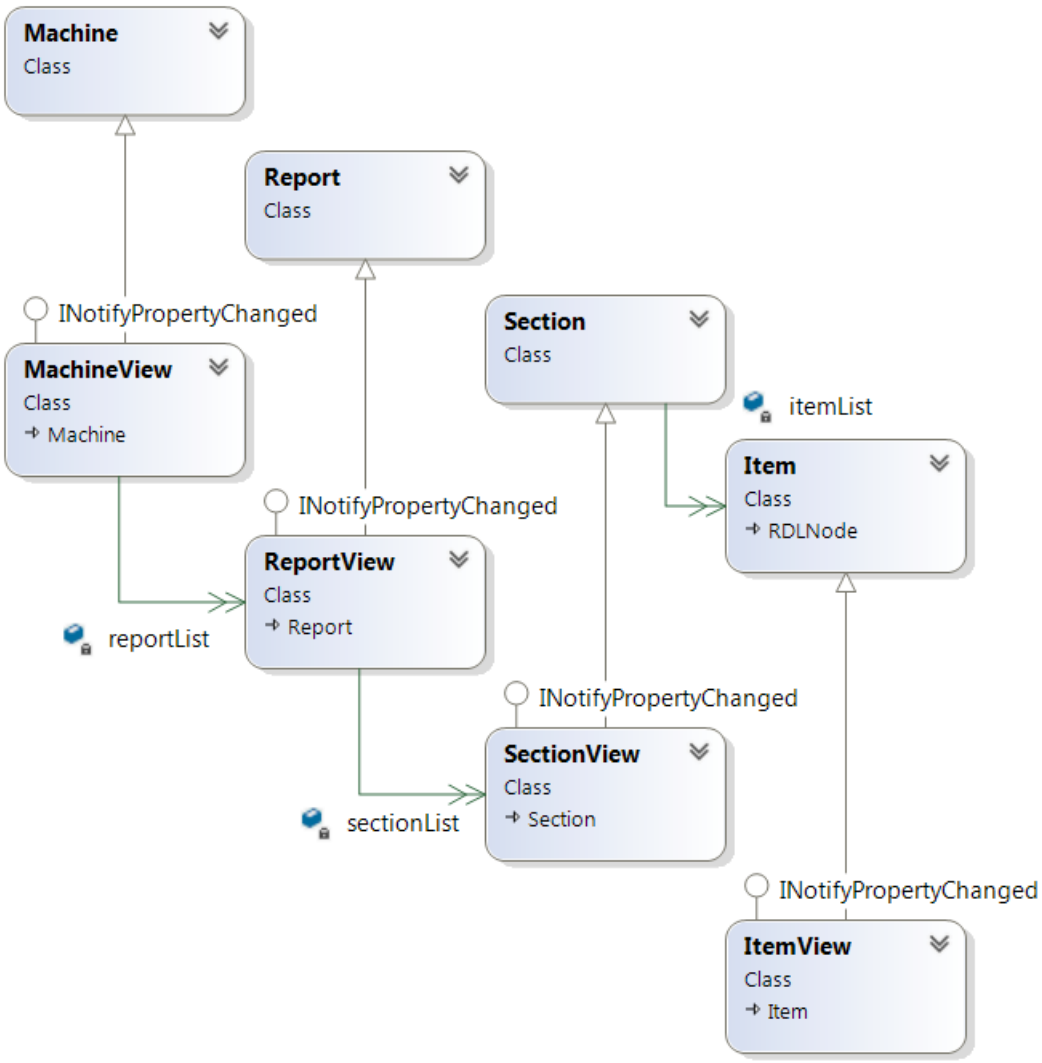

Figura 20. Classi per la rappresentazione delle entità

La procedura automatica di costruzione del file RDL è implementata dal componente software Builder. Questo processo è avviato dall'oggetto ReportBuilder, creato alla pressione del relativo pulsante nell'interfaccia grafica. Le classi BodyBuilder e HeaderBuilder si occupano della ricerca dei nodi necessari alla visualizzazione degli item selezionati dall'utente, rispettivamente contenuti nel body e nell'header del report. Questa ricerca consiste nell'individuazione dei nodi DataSet, DataSource, Parameter e EmbeddedImage relativi agli item inseriti. Oltre ad individuare i DataSet

relativi agli item e i relativi parametri è necessario ricercare anche i DataSet utilizzati per valorizzare i parametri. La classe ReportElements viene utilizzata per mantenere in memoria i nodi da scrivere sul file di output, mentre Report Format rappresenta le informazioni legate al layout del report come il valore delle dimensioni o dei margini.

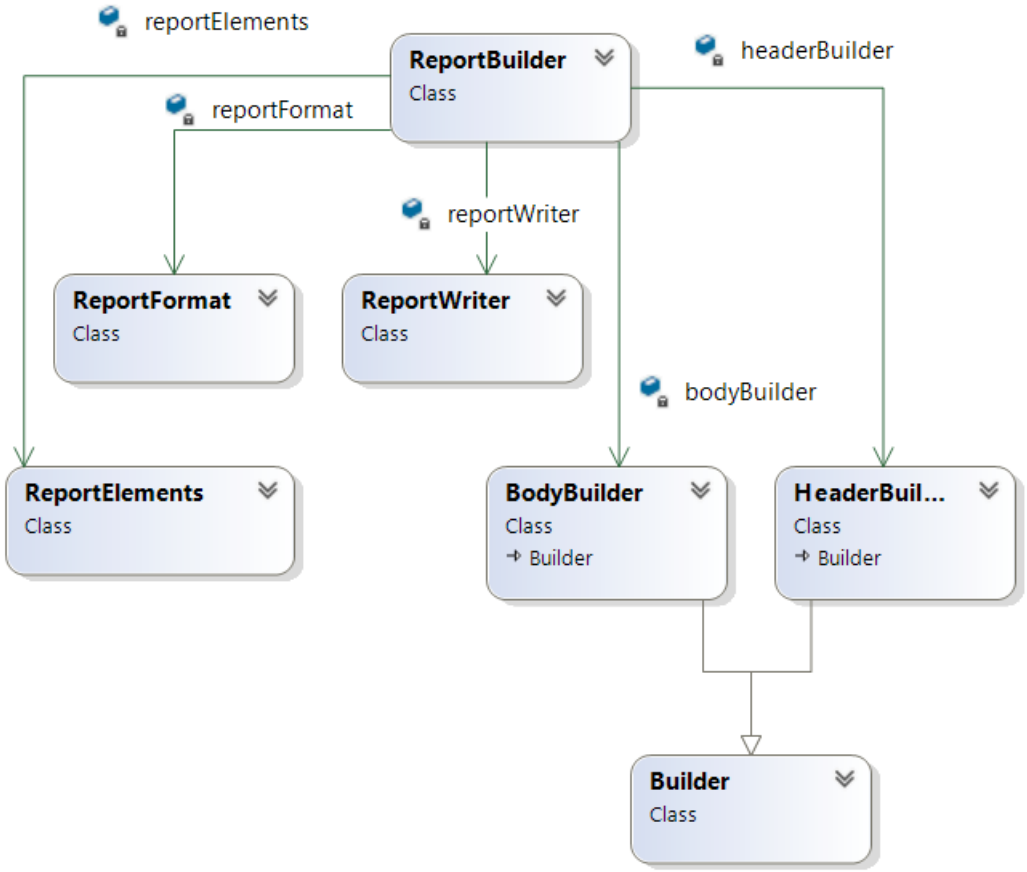

Figura 21. Diagramma delle classi del componente Builder

La classe ReportWriter offre i metodi di scrittura sul report in output dei nodi RDL individuati dagli oggetti di classe Builder.

4.2.2 Struttura dell'interfaccia utente

Per quanto riguarda l'interazione con l'utente viene definita una pagina principale in cui sono proposte le funzioni di editing e una pagina di configurazione in cui è possibile modificare le impostazioni relative al funzionamento dell'applicazione. Nella pagina di configurazione possono essere definiti:

- I path del file system in cui ricercare i report e le immagini, e la locazione in cui salvare il file di output.

- I path RDL utilizzati dai componenti che eseguono funzioni di lettura e scrittura di nodi o documenti RDL. Questi percorsi individuano ad esempio le caratteristiche "pageWidth" e "pageHeight" all'interno di un documento RDL. Tali path sono memorizzati in un file di testo e sono gestiti in lettura da una classe statica denominata "RDLPath".

- I template dei nodi utilizzati per la creazione di nuovi item dal componente Generator.

Nel seguito si considerano gli aspetti legati alla pagina principale dell'applicazione, la cui struttura è mostrata in Figura 22. Le principali funzionalità dell'interfaccia grafica sono:

- Presentazione degli elementi selezionabili dall'utente per permettere la navigazione tra le macchine, i report e le sezioni.

- Riproduzione dell'anteprima del report prodotto. Ciò significa mostrare una rappresentazione grafica di ogni sezione per illustrare il layout del report in fase di editing.

- Mostrare i comandi di editing per la personalizzazione della struttura del documento.

La progettazione dei componenti dedicati alla gestione dell’interfaccia e all’aggiornamento del modello segue il pattern MVVM – Model/View/ViewModel.

Per aumentare l’usabilità si è deciso di integrare in una stessa finestra i comandi relativi alla selezione degli elementi e alla modifica del layout del report generato. In questo modo risulta intuitivo gestire allo stesso tempo il contenuto e il layout del report, evitando di rendere necessario cambiare finestra per passare da una funzionalità all'altra. Come illustrato in Figura 22 i comandi relativi alla selezione del contenuto e quelli associati alla modifica dell’aspetto grafico sono organizzati in aree distinte della finestra principale.

Nella parte superiore della finestra principale viene inserita una barra di stato, cioè un campo di testo dedicato alla visualizzazione di commenti che hanno lo scopo di aggiornare l'utente sullo stato operativo dell'applicazione e di indicare la presenza di eventuali configurazioni errate. Ogni operazione eseguita sull’interfaccia dall’utente ha come effetto la generazione di un messaggio di stato. La buona riuscita dell’esecuzione è accompagnata dalla descrizione di ciò che è stato eseguito dalla logica del programma, mentre eventuali eccezioni prevedono la comparsa di un messaggio che indica il problema riscontrato, in modo tale che l’utente possa essere istruito su quali azioni deve effettuare per rendere eseguibile l’operazione richiesta. Una possibile errore riscontrato dal sistema consiste nel tentativo di avviare la generazione di un report senza aver definito alcun valore nell'header.

La parte centrale dell'interfaccia grafica riguarda il contenuto del report. Nell'area sinistra viene riprodotto una vista ad albero in cui l'utente può navigare tra le sezioni appartenenti ai report individuati. In quest'area è inoltre presente un form per l'inserimento di nuovi oggetti grafici. L'inserimento di una sezione, reso possibile nella vista ad albero, comporta l'introduzione di una rappresentazione della sezione nell'area centrale, chiamata “View”, dedicata

alla visualizzazione dell'anteprima del report. La View permette di fornire all'utilizzatore una rappresentazione grafica del report in fase di costruzione, in accordo al principio “What you see is what you get”. A destra della zona dedicata all'anteprima viene gestita una lista delle sezioni, del titolo e del logo inseriti dall'utente. In quest'area risulta possibile eliminare gli elementi precedentemente selezionati.

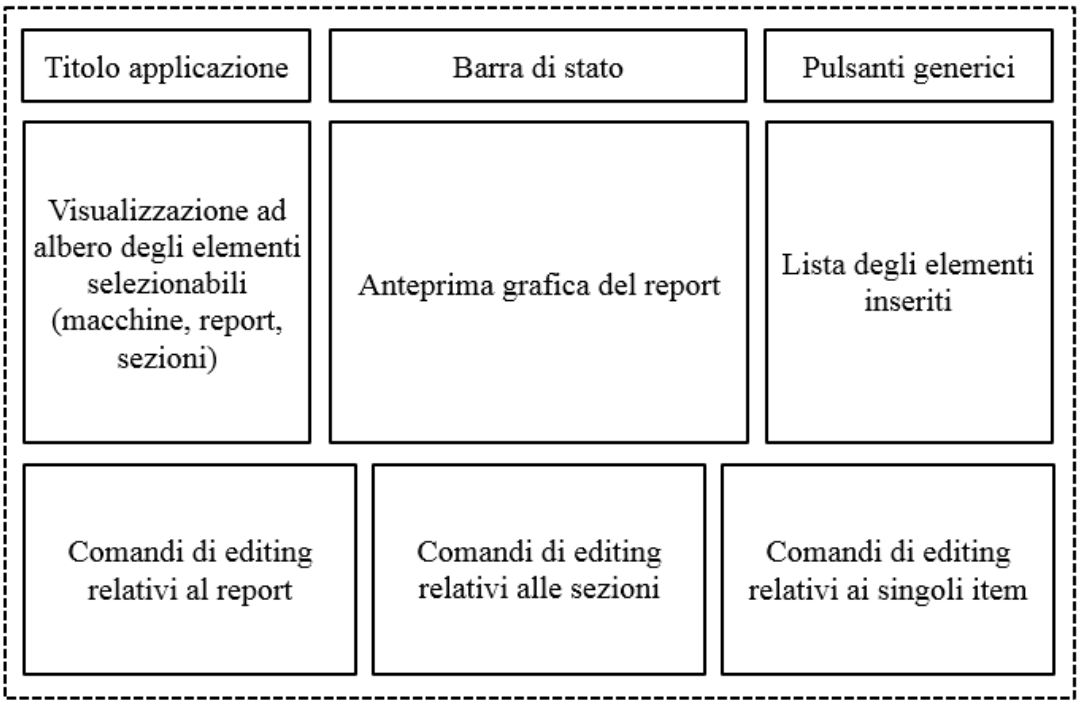

Figura 22. Struttura grafica della finestra principale.

Nella parte inferiore dell'interfaccia grafica vengono gestiti i comandi necessari alla personalizzazione del report. Funzionalità di editing grafico riguardano ad esempio l'allineamento delle sezioni inserite o l'impostazione di margini laterali alla pagina. Sempre nella parte inferiore è possibile visualizzare l'elenco degli item che compongono la sezione selezionata. Per ogni item è possibile visualizzare e modificare il contenuto della relativa query in modo da personalizzare il contenuto del report.

4.3 Sviluppo

Durante la fase di implementazione dell'applicazione è necessario definire le caratteristiche più concrete legate alla programmazione del sistema, all'implementazione che garantisca una facile manutenibilità e riusabilità. Oltre all'implementazione dei sottosistemi, un aspetto importante di questa fase riguarda l'integrazione di questi per comporre un sistema che rispetti i requisiti definiti in precedenza.

L'applicazione è stata sviluppata in ambiente .NET, utilizzando C#, un linguaggio di programmazione Object Oriented che fornisce le principali funzionalità relative ai linguaggi di questo tipo come Incapsulamento, Ereditarietà, Polimorfismo. Altri costrutti messi a disposizione sono tipi di valore nullable, enumerazioni, delegati, espressioni lambda, metodi e tipi generici.

L'utilizzo di C# è stato motivato da due considerazioni: prima di tutto i sistemi SCADA realizzati in IMA sono basati su sistema operativo Windows, per questo motivo il supporto al framework .NET è nativo in essi. Inoltre, dato che gli obbiettivi dell'applicazione in esame sono quelli di manipolare un linguaggio formalmente analogo ad XML, è risultato piuttosto immediato realizzare le funzioni di manipolazione di XML con le numerose classi definite per questo scopo in ambiente C#.

Le librerie WPF (Windows Presentation Foundation) costituiscono lo stato dell'arte della tecnologia di sviluppo di interfacce grafiche in ambiente .NET, mentre il pattern Model/View/ViewModel definisce le linee guida per l'implementazione dell'interazione con l'utente.

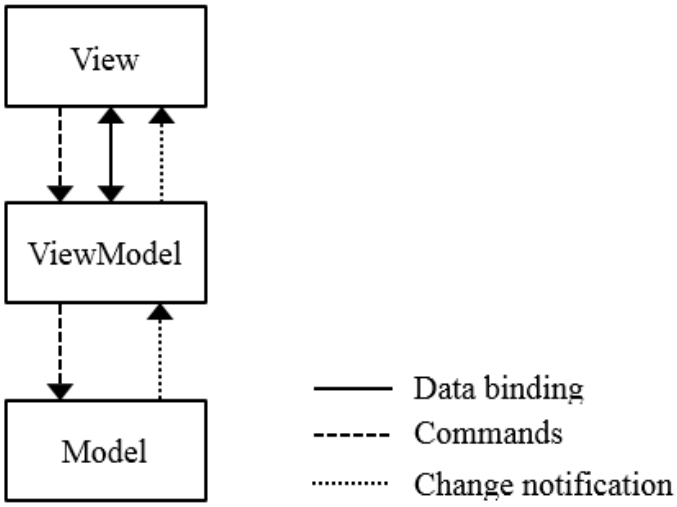

Figura 23. Componenti del pattern Model-View-View Model

4.3.1 Realizzazione del modello e rappresentazione dei dati

Come già discusso, i componenti definiti in fase di progettazione implementano le funzionalità di editing e di generazione del file RDL. Nel seguito vengono descritte le principali soluzioni implementative impiegate per la realizzazione di tali sottosistemi insieme alle difficoltà incontrate nel processo di sviluppo.

Finder, il componente dedicato alla ricerca delle macchine, dei report e delle sezioni, utilizza funzionalità di lettura del file system ed implementa la logica di identificazione delle sezioni. In ciascuno dei report predefiniti l'attributo “Name” di ogni item viene modificato in modo da inserire un prefisso che indichi il nome della sezione a cui appartiene. Questa operazione permette al componente Finder di distinguere le sezioni presenti nel report analizzato.

Il componente software Generator è composto dalle classi che implementano la creazione di nuove sezioni. Come detto tale attività viene gestita dall'utente tramite un form presente nell'interfaccia grafica. Nei campi del form sono definite le caratteristiche della sezione che si vuole inserire nel report. Il sottosistema Generator inserisce tale informazione nei campi vuoti del template relativo alla sezione scelta. L'impiego di un template RDL da valorizzare nel momento in cui è richiesto l'inserimento della sezione permette di rendere la creazione di nuove sezione indipendente dai dettagli del linguaggio RDL: nel caso in cui sia necessario eseguire modifiche sintattiche è sufficiente agire sui documenti di template memorizzati.

Data Editor ha lo scopo di propagare sui dati i comandi scelti dall'utente tramite l'interfaccia. Si tratta di semplici funzionalità di editing grafico come l'allineamento delle sezioni o la modifica di margini nel report.

Il processo di creazione RDL viene implementato dal componente Builder. Questa operazione avviene in modo automatico e come già sottolineato deve implementare la ricerca dei nodi RDL associati a ciascun item inserito dall'utente. Ad esempio un nodo item può contenere il riferimento ad un nodo DataSet che contiene la query necessaria all'estrazione dei dati. Dal momento che un nodo DataSet prevede l'utilizzo di parametri, anch'essi definiti da DataSet, il procedimento di ricerca dei nodi RDL consiste in un ciclo iterativo.

Questo ciclo di identificazione di nodi, rappresentato in Figura 24, termina quando si presenta la condizione in cui tutti i nodi da inserire in un passo sono già stati inseriti.

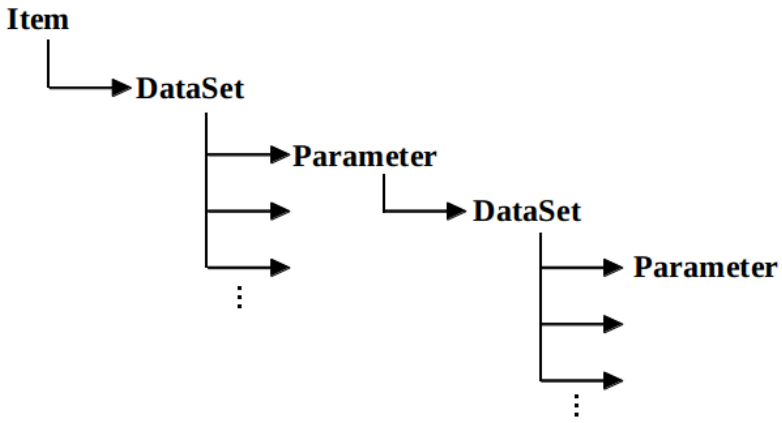

Figura 24. Processo iterativo di ricerca dei nodi RDL

Oltre ai nodi Item, DataSet e Parameter, il sistema si occupa dell'identificazione dei nodi DataSource necessari per l'esecuzione delle query e la codifica delle immagini eventualmente presenti, informazione contenuta nei nodi EmbeddedImage.

4.3.2 Gestione dell’interazione con l’utente

Per l'implementazione dell'interfaccia utente sono stati utilizzati i meccanismi offerti dalla libreria WPF. Questi garantiscono una notevole flessibilità di design delle funzionalità di interazione con l'utente. Le caratteristiche che sono risultate più utili alla progettazione del lavoro di questa tesi sono:

- Disaccoppiamento tra aspetti legati al design della grafica e alla definizione del comportamento dei componenti dell'applicazione. Il layout è definito in modo dichiarativo con il linguaggio XAML (Extensible Application Markup Language), mentre le funzionalità sono programmate con classi C#. Il legame tre la View e il modello è realizzato tramite meccanismi di databiding gestiti dal componente ViewModel.

- Creazione di interfacce grafiche in modo indipendente dalla risoluzione del monitor impiegato. È infatti possibile realizzare dimensionamento automatico dei componenti in base alla risoluzione scelta per lo schermo.

- A ciascun componente grafico viene associato uno stile. Gli stili vengono definiti in XAML in un file dedicato. Questo, oltre ad aumentare la modularità della struttura dell'applicazione, permette la riconfigurazione dinamica dell'aspetto grafico a run time. Nella definizione degli stili si tiene conto anche delle modifiche grafiche subite dai componenti a fronte delle operazioni eseguite dall'utente. Ad esempio le sezioni selezionate assumono un aspetto caratteristico nell'area di anteprima del report, dovendo anche essere messe in primo piano rispetto le altre sezioni presenti. Questo meccanismo viene implementato in XAML tramite la definizione di triggers associati a un componente grafico.

Il linguaggio XAML, una specializzazione di XML, permette di descrivere in modo dichiarativo gerarchie di oggetti grafici, e i relativi stili. Il file MainWindows.xaml rappresenta l'aspetto grafico della finetra principale di IMAReportBuilder.

L'indipendenza dalla risoluzione è ottenuta impiegando dimensioni considerate in DIP (Device Independent Pixels), unità logiche corrispondenti a 1/96 di pollice. L'utilizzo dei DIP permette all'applicazione di autodimensionarsi graficamente in base al DPI (Dot Per Inches) scelto. Il risultato è che l'interfaccia grafica può scalare in modo da adattarsi alla risoluzione scelta. Questo è anche dovuto al fatto che WPF è interamente basato su grafica vettoriale.

Un ulteriore aspetto di WPF che è risultato particolarmente utile è costituito dalla possibilità di definire grafica 2D in modo estremamente agevole tramite l'elemento denominato “Canvas”. A questo componente possono essere aggiunte forme geometriche posizionabili in modo relativo al sistema di riferimento definito dal canvas stesso. Tale elemento è stato utilizzato per la modellazione del report: gli item, graficamente rappresentati come rettangoli, vengono dinamicamente aggiunti al report, cioè all'elemento Canvas, man mano che l'utente li seleziona.

La caratteristica forse più innovativa di WPF è il meccanismi che permettono la realizzazione di Data Binding. Questo meccanismo permette di instaurare un legame tra gli elementi grafici e le proprietà del programma che definisce il comportamento dell'applicazione. Il maggior beneficio derivato da questa tecnica è la netta separazione tra l'interfaccia utente e il resto dell'applicazione. Utilizzando data binding non è necessario gestire eventi tramite metodi ascoltatori ma è sufficiente definire un legame tra elementi grafici e proprietà della logica sottostante. La caratteristica offerta dal databinding di poter legare tra loro oggetti grafici ed entità logiche come variabili e comandi spiana la strada all'utilizzo del pattern MVVM (Model View Viewmodel) per lo sviluppo software. Come il nome stesso suggerisce, secondo questo pattern architetturale l'applicazione si compone di tre livelli logici distinti. Se una proprietà del ViewModel viene modificata, il nuovo valore si propaga alla View in modo automatico attraverso il meccanismo di databinding. Quando l'utente spinge un bottone nella View, un comando del ViewModel esegue l'azione richiesta. È solo il ViewModel che può manipolare i dati.

Il meccanismo del data binding permette di evitare l’utilizzo degli event handlers, aumentando il disaccoppiamento tra le funzionalità definite nel livello business logic e quelle che permettono la gestione dell'interfaccia grafica.

In questo modo si disaccoppiano completamente il funzionamento dell'interfaccia dalla logica operativa: gli oggetti che compongono la View non interagiscono mai direttamente con gli oggetti del modello. Allo stesso tempo sia ViewModel che modello sono completamente slegati dalle classi della View.

Data la capacità di rappresentare interfacce utente in modo dichiarativo, XAML trova naturale impiego nell'ambito del pattern MVVM. Per questo motivo tramite l'utilizzo di XAML si crea una netta distinzione tra l'attività di sviluppo della logica e quella di design dell'interfaccia.

Nel pattern MVVM la classe ViewModel ha il compito di gestire i dati che vengono mostrati nella View. Il ModelView non contiene riferimenti alla View ed è completamente disaccoppiato dall'implementazione della View. ViewModel implementa le proprietà e i comandi ai quali la View si lega.

L'aggiornamento dei dati nell'interfaccia grafica tramite data binding prevede che le classi del ViewModel implementino l'interfaccia “INotifyPropertyChanged”, la quale dichiara il metodo “NotifyPropertyChanged”, che permette di instaurare il legame tra le proprietà definite nel ViewModel e i dati mostrati nella View. Dal momento che i meccanismi di gestione dell'interfaccia prevedono la modifica dei dati, è necessario che anche le classi che rappresentano questi dati implementino l'interfaccia “INotifyPropertyChanged”.

Gli oggetti XAML impiegati per l'implementazione dell'interazione con l'utente sono i Button, TextBlock, TextBox e ComboBox. La realizzazione della vista ad albero delle macchine, dei report e delle sezioni è basata sul componente grafico TreeView, mentre l'oggetto XAML utilizzato per la realizzazione dell'area di anteprima del report è denominato Canvas. Questo controllo grafico è stato ridefinito rispetto all'implementazione originale per poter implementare la gestione del drag&drop sugli elementi definiti in esso. In particolare la classe

“CanvasUserControl”, contenuta nel componente ViewModel, implementa la classe canvas nativa in WPF e aggiunge i metodi necessari alla gestione degli eventi associati al drag&drop, come la selezione di una sezione, il trascinamento in un altra posizione e il rilascio.

Di seguito è rappresentata l'interfaccia realizzata per il sistema software in esame.

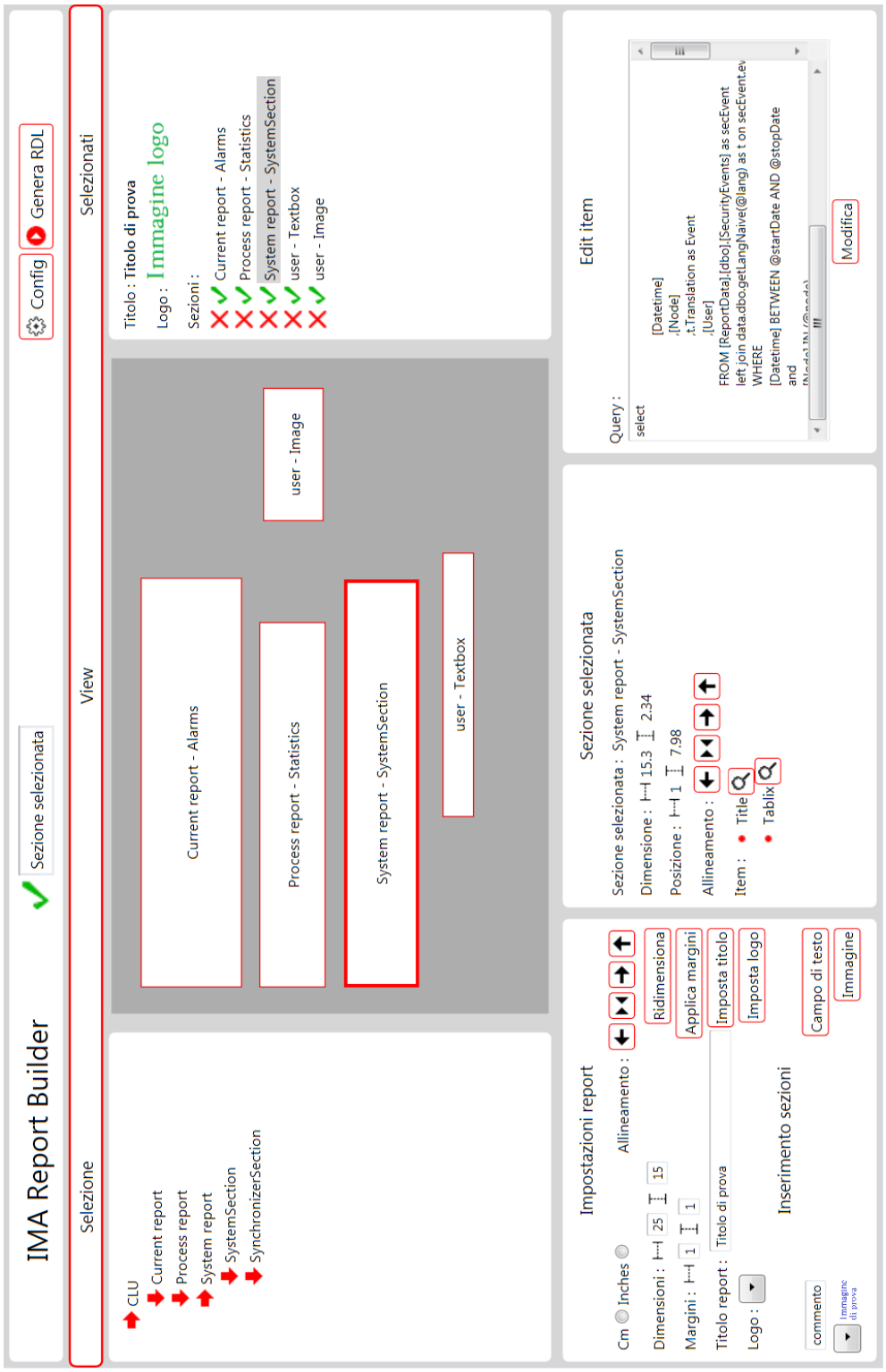

Figura 25. Interfaccia grafica del sistema progettato

Capitolo 5
Conclusioni

Il progetto descritto in questo testo ha permesso di definire la realizzazione di un sistema software, denominato “IMAReportBuilder”, in grado di generare documenti di report personalizzabili in un contesto di supervisione industriale. In particolare tale apparato è stato concepito con lo scopo di presentare una interfaccia grafica che renda immediato l'editing della struttura dei file RDL, impiegati per la creazione dei report. Le principali caratteristiche possedute dall’applicazione sono:

- Astrazione dal linguaggio di definizione dei report. Tramite interfaccia grafica risulta possibile modificare sia il layout che il contenuto del report prodotto agendo su pulsanti e campi di testo.
- Soddisfacimento dei requisiti di usabilità comunemente legati ad editor di file grafici. In particolare sono state implementati:
    - Un'area di anteprima che permette di rappresentare la struttura grafica del report che l'utente sta costruendo. Questa funzionalità

è in accordo con l'approccio “What you see is what you get”, essenziale nella progettazione di editor grafici.

- Meccanismi di drag&drop per la selezione delle sezioni e per il loro posizionamento.

- Una barra di stato per agevolare l'utilizzo dei comandi forniti e per istrure l'utilizzatore sul corretto impiego dell'interfaccia.

- Automatizzazione del processo di generazione RDL. È stato implementato un componente software per la generazione dei nodi necessari al fine della corretta interpretazione del file RDL.

- Semplicità di adattamento a modifiche dello standard di definizione. Si è cercato di rendere il motore di generazione indipendente dai dettagli implementativi dello standard RDL.

Le funzionalità implementate dal sistema in esame riguardano la fase di configurazione dei report e sono sintetizzate in Figura 26.

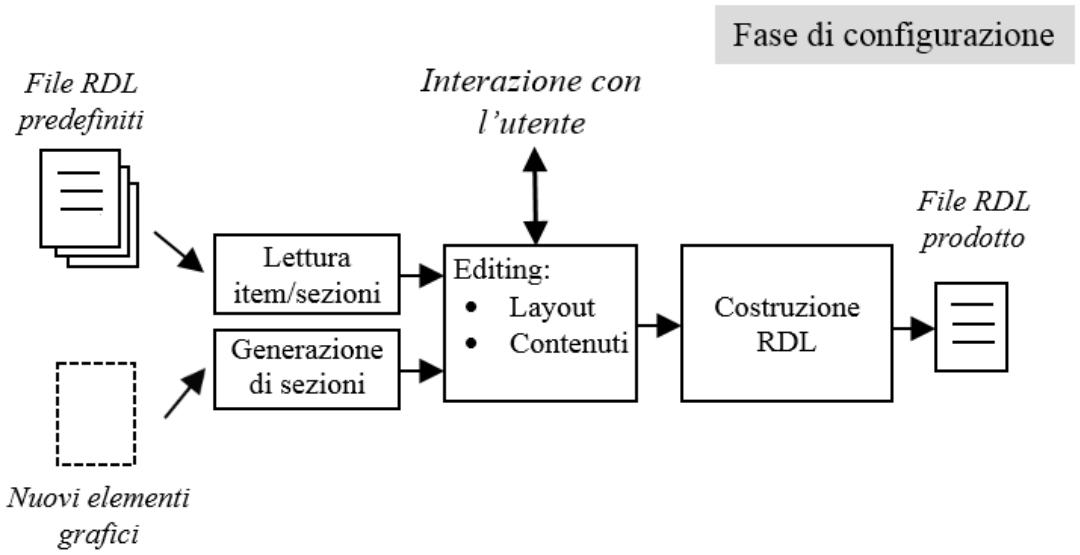

Figura 26. Fase di configurazione di un report

Come discusso nel Capitolo 3, i file RDL vengono interpretati da una applicazione dedicata al fine di produrre il documento di report.

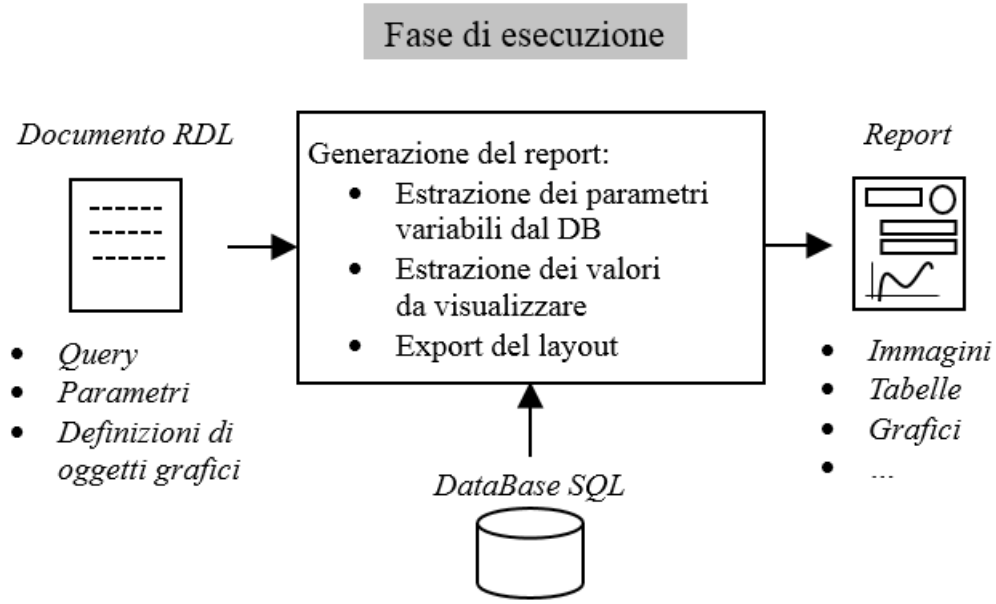

Figura 27. Processo di generazione di un report

I vantaggi introdotti dall’utilizzo dell'applicazione descritta riguardano sia la fase di progetto del sistema di reportistica che il suo utilizzo. I benefici consistono nella:

- Riduzione dei tempi necessari alla definizione dei report in fase di configurazione del sistema SCADA. Questo aspetto è strettamente legato al sottosistema di costruzione RDL implementato.

- Personalizzazione dei report da parte dell'utente della macchina attraverso un'interfaccia grafica.

Di seguito viene mostrata l'interfaccia grafica nel momento in cui si avvia il processo di generazione del file RDL.Viene inoltre illustrato il report generato a partire dal file di output.

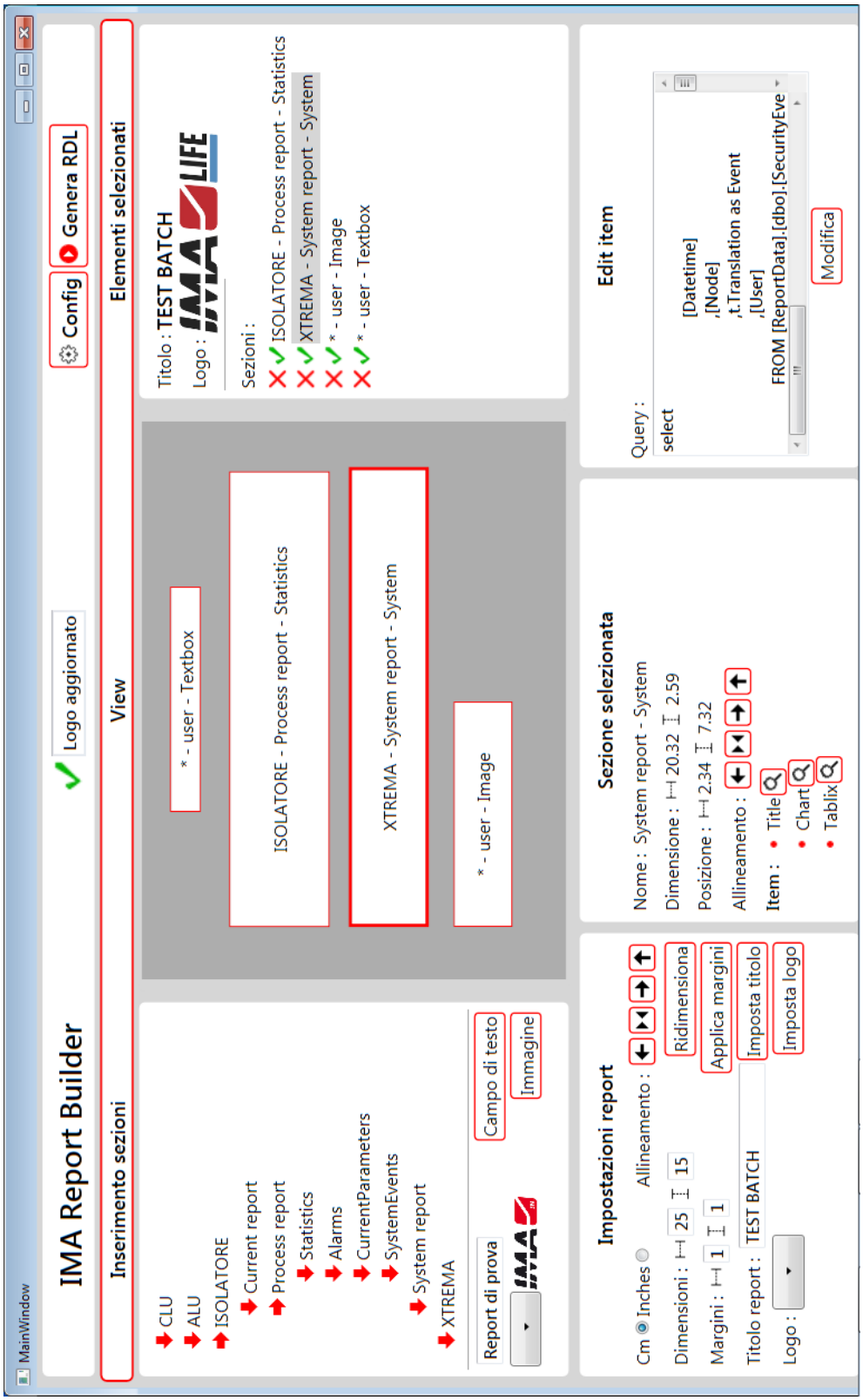

Figura 28. Interfaccia utente

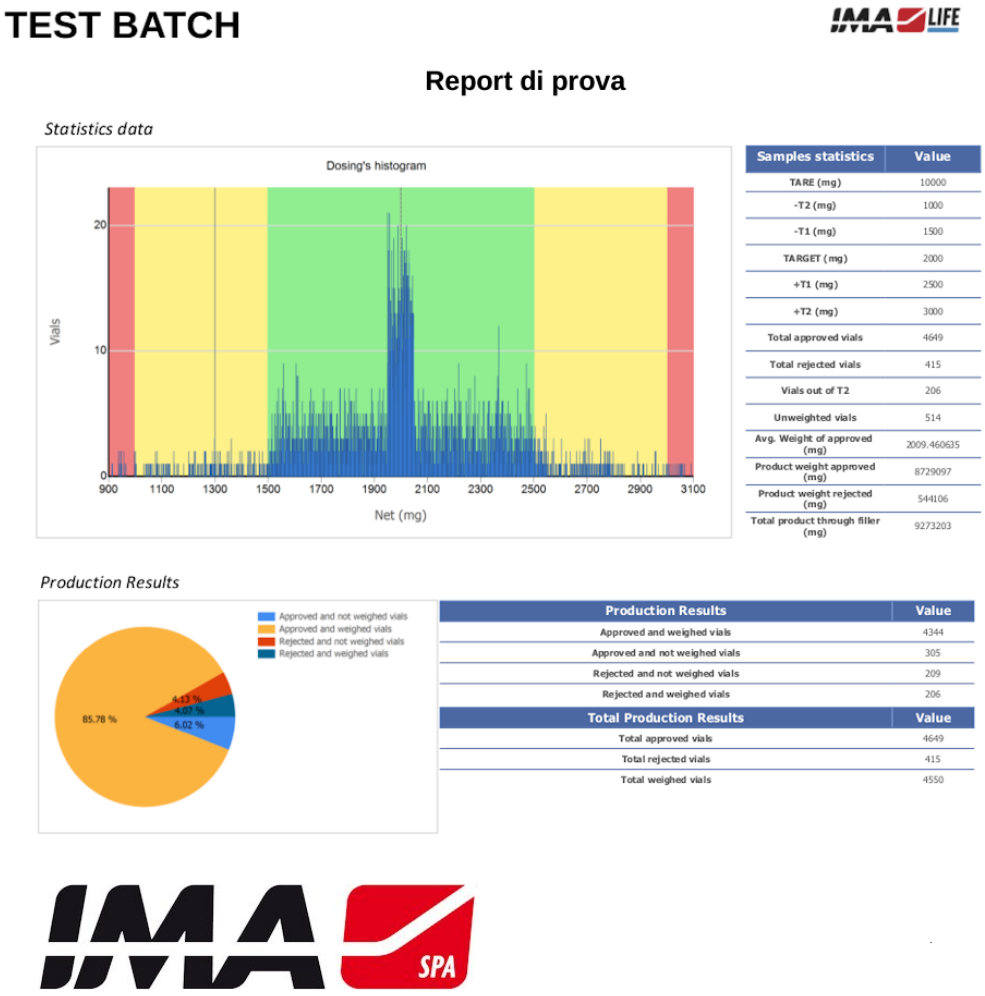

Figura 29. Report generato in fase di test

5.1 Sviluppi futuri

Lo sviluppo delle tecnologie informatiche ha senza dubbi importanti conseguenze anche nel campo dell'automazione. Uno dei temi più discussi nel l'ambito del monitoraggio industriale è l'adozione di infrastrutture che permettano la supervisione da remoto di grandi impianti.

Per quanto riguarda il sistema di reportistica viene sempre più presa in considerazione l'utilizzo di tecniche di data mining per l'analisi dei dati gestiti da uno SCADA. La grande quantità di informazioni gestite in un impianto industriale lascia intuire che sia possibile eseguire analisi più dettagliate rispetto le usuali considerazioni di tipo statistico. Non è infatti sbagliato riferirsi ai dati elaborati da uno SCADA con il nome di “Big Data”: informazione di grandi volumi, estremamente eterogenea ed acquisita in tempo reale ad elevate velocità.

L'impiego di tecniche di data mining ha lo scopo di estrarre informazioni non visibili dai dati archiviati nel database della macchina e presentati nei report, per questo motivo non si intende rimpiazzare il procedimento di analisi statistica dei dati, ma solo integrarne i risultati. In particolare un possibile scenario applicativo riguarda l'analisi dei dati operativi per la definizione di relazione di causa-effetto tra le dinamiche presenti nell'impianto monitorato, con l'obiettivo di effettuare un'analisi predittiva di guasti. L'attenzione sui sistemi inferenziali applicati nel mondo dell'automazione è in continua crescita, spinta dal sempre maggior successo che queste tecniche hanno nel campo dell'IT.

Riferimenti bibliografici e sitografici

[Bimbo e Colaiacovo, 2006] Bimbo S., Colaiacovo E. (2006) “Sistemi SCADA” Ed.Apogeo.

IFIX Training Student Guide Version 5.5 – 10.12

[Sommerville, 2008] Sommerville I. (2008) “Ingegneria del software” Pearson.

[Gamma, Helm, Johnson, Vlissides, 1994] Gamma E., Helm R., Johnson R., Vlissides J. (1994) “Design Patterns, elements of reusable object oriented software” Prentice Hall.

Wikipedia – Enciclopedia libera,
https://it.wikipedia.org/wiki/SCADA

FDA CFR 21 Part11,
https://www.accessdata.fda.gov/scripts/cdrh/cfdocs/cfcfr

OPC Foundation,
https://opcfoundation.org

Documentazione tecnica .NET,
https://msdn.microsoft.com/it-it/library