

Politecnico di Torino

DIPARTIMENTO DI AUTOMATICA E INFORMATICA

Corso di Laurea Magistrale in Ingegneria Informatica

Tesi di Laurea Magistrale



Progetto e sviluppo di emulatori di Cyber Physical System per Cyber Range ibridi

Candidato

Simone Riggi

Relatore

Prof. Paolo Prinetto

Summary

In the last years, cyber-physical systems (CPS) have attracted considerable attention due to their intrinsic combination of physical and cybernetic aspects, in particular, about the aspect of security. Industrial control systems (ICS) belong to a class of cyber-physical systems in which cyber entities (PLCs, computers) have direct control over control processes and physical actions. Any cyber attacks on these systems could have very serious consequences directly on the population.

Modern industrial information systems make extensive use of an acquisition and supervision system (SCADA), for the management of the various infrastructure components. These structures and their controllers can be the ideal target for cyber attackers, causing serious economic and environmental losses. This is especially the case for services that are critical for a community, such as power plants or water distribution systems.

In this context, the thesis focuses on the importance of developing and implementing a hybrid cyber range, for testing and verifying any vulnerabilities of such systems, in an isolated and secure manner. Particular importance was dedicated to the design of the SCADA network infrastructure, to the logical modeling of the controller and to the creation of a web application that allows the remote monitoring and control of the physical model.

The results achieved show that the use of the Niagara framework considerably simplifies and reduces the development time of a SCADA system, offering numerous advantages both in terms of security and performance.

Sommario

Negli ultimi anni, i sistemi cyber-fisici (CPS) hanno attirato una notevole attenzione a causa della loro intrinseca combinazione di aspetti fisici e cibernetici, in particolare, per quanto riguarda l'aspetto della sicurezza. I sistemi di controllo industriale (ICS), appartengono ad una classe di sistemi cyber-fisici in cui le entità cyber (PLC, computer) hanno il controllo diretto sui processi di controllo e sulle azioni fisiche. Eventuali attacchi informatici su tali sistemi potrebbero causare gravissime conseguenze direttamente sulla popolazione.

I moderni sistemi informativi industriali fanno largo uso di un sistema di acquisizione e supervisione (SCADA), per la gestione delle diversi componenti delle infrastrutture. Queste strutture e i loro controller possono essere l'obiettivo ideale per i cyber attaccanti, causando gravi perdite economiche ed ambientali. Questo è il caso soprattutto di servizi critici per una comunità, come le centrali elettriche o i sistemi di distribuzione idrica.

In questo contesto, la tesi si focalizza sull'importanza di sviluppare e realizzare un cyber range di tipo ibrido, per il testing e la verifica di eventuali vulnerabilità di tali sistemi, in maniera isolata e sicura. Particolare importanza è stata dedicata alla progettazione dell'infrastruttura di rete dello SCADA, alla modellazione logica del controller ed alla realizzazione di un' applicazione web che consente il monitoraggio ed il controllo da remoto del modello fisico.

I risultati raggiunti mostrano che l'utilizzo del framework Niagara semplifica e riduce considerevolmente i tempi di sviluppo di un sistema SCADA, offrendo numerosi vantaggi sia dal punto di vista della sicurezza che delle prestazioni.

Indice

1	Introduzione	7
2	CyberRange	10
2.1	Valutazioni di sicurezza un sistema	11
2.2	Il concetto di cyber range	12
2.3	Progettazione e testing	15
2.3.1	Tetsbed	16
2.3.2	Design tipico per cyber security testbed	17
2.3.3	Tipologia di sfide	18
2.4	Training e valutazione	20
2.5	Cyber Physical Systems	21
2.6	Prospettive future	23
3	EVA	25
3.1	Struttura EVA	25
3.2	Model Design	27
3.3	Componenti EVA	31
3.3.1	JACE 8000 controller	31
3.3.2	Raspberry Pi	32
3.3.3	Sensori	32
3.3.4	Attuatori	34
3.3.5	Modbus	35
3.4	TestBench	35
3.4.1	Interazione con l'ambiente	36
3.4.2	Interazione con lo SCADA <i>User</i>	37
3.4.3	Interazione con il consumatore	43
4	Niagara4	45
4.1	Niagara Framework	45
4.2	Niagara Workbench	46
4.2.1	Visualizzazione dinamica	46

4.2.2	Tagging e templating	46
4.2.3	Sicurezza avanzata	46
4.3	Building blocks	47
4.3.1	Moduli	48
4.3.2	Componenti	49
4.3.3	Presentazione XML	50
4.3.4	Stazioni	51
4.3.5	ORDs	52
4.4	Integrazione dei sistemi di controllo	53
4.5	Networking e protocolli Internet	54
4.6	Baja	55
4.7	APIs	56
5	Data and Control Model	57
5.1	History	58
5.2	Wire Sheet Object Management	58
5.3	Logica di controllo	59
5.3.1	Lake	60
5.3.2	Purifier	62
5.3.3	Clean Water	62
5.3.4	Environment	63
5.3.5	House	65
6	Web Application	68
6.1	Architettura	68
6.1.1	Infrastruttura di rete	69
6.2	Interfacciamento protocolli	74
6.3	Software di sviluppo e applicativo	74
6.3.1	EVA Virtual SCADA	76
6.3.2	SCADA Programmer	76
6.3.3	EVA Manager	77
6.3.4	Testbench Administrator	79
6.3.5	EVA <i>Guest</i>	80
6.3.6	Archiviazione	81
7	Conclusioni e lavori futuri	83

Capitolo 1

Introduzione

Negli ultimi 20 anni, le tecnologie nel campo dell'informazione hanno subito importanti cambiamenti ed innovazioni. Questa ha raggiunto un utilizzo diffuso in tutti gli aspetti e settori della vita quotidiana, sociale, economica, amministrativa e dell'industria, facilitando l'amministrazione di individui, comunità, istituzioni e governi. Tutto ciò implica che l'ordine pubblico e la sicurezza di un Paese possono essere influenzati e dipendenti dal funzionamento di ciascuna di queste aree. I singoli individui stanno diventando dipendenti da tali tecnologie, dando origine e significato alla definizione di *cyber addiction*. I rapidi progressi sul campo della tecnologia Internet ha assottigliato i confini dei paesi, svolgendo un ruolo fondamentale nel concetto di globalizzazione. Soprattutto negli ultimi anni, lo sviluppo ed il crescente uso di tecnologie quali Internet of Things, Cloud Computing e Big Data ha portato con sé diversi vantaggi (es. consentendo un accesso istantaneo a tutti i tipi di informazione), prestando però il fianco a diversi rischi e svantaggi, come le minacce informatiche, che sfruttano le vulnerabilità di questi. Il continuo utilizzo di tali tecnologie, non solo modifica gli schemi comportamentali della vita sociale, ma anche le infrastrutture critiche dei paesi. Il numero di attacchi informatici è in continuo aumento a causa di tre principali fattori:

- Economico: un cyber attacco può causare gravi perdite economiche;
- Tecnologico: un semplice attacco può intaccare l'integrità di un sistema di un'infrastruttura fisica di controllo o contaminare l'ambiente ecologico;
- Strategico: un cyber attacco può causare gravi problemi ad un'intera regione interrompendone i servizi pubblici;

. A tale scopo, ciascun paese, sviluppa delle politiche e misure di sicurezza delle informazioni, tenendo conto dello stato e delle capacità dei sistemi informativi critici e delle proprie competenze tecnologiche. Oggigiorno, la gran parte dei paesi controlla le proprie infrastrutture critiche tramite le funzioni di monitoraggio e di controllo fornite dalle rete ICS/SCADA. In passato tali sistemi erano isolati da altri, mentre nel caso di quelli moderni risultano essere sempre più complessi e digitali, connessi verso l'esterno tramite connessione Internet. Pertanto, un utente malevolo può, piuttosto agevolmente, accedere ed interrompere i sistemi di controllo critici se non vengono prese adeguate precauzioni cautelari. Il campo della sicurezza informatica in relazione a ICS e SCADA risulta essere molto complesso, e l'ignoranza sui possibili attacchi che possono minacciarne l'integrità, potrebbe portare a conseguenze significative [12].

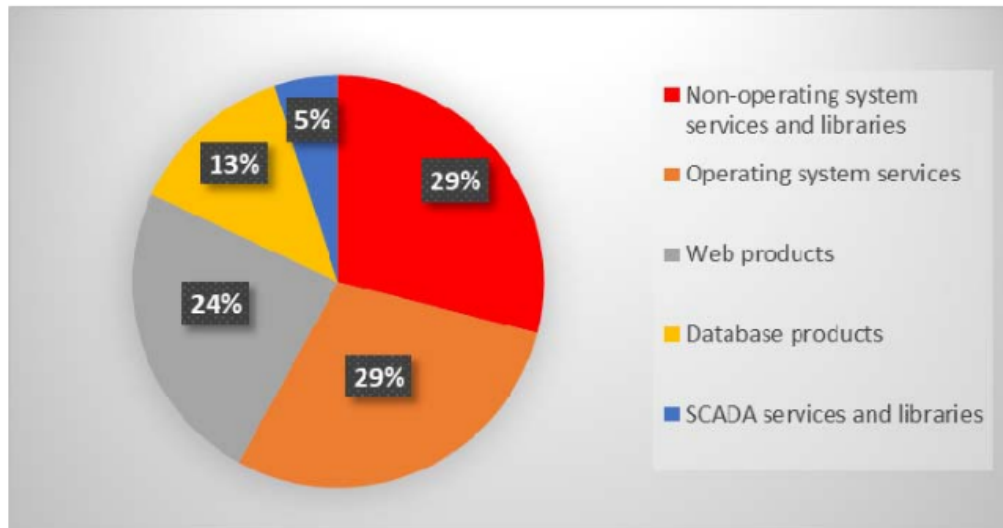


Figura 1.1: Percentuale dei componenti senza patch di sicurezza in un sistema SCADA

Per le ragioni sopracitate, è importante introdurre il concetto di cyber sicurezza in tutti i contesti, anche in quelli considerati "sicuri", proprio perchè solitamente sono i primi punti vulnerabili ad attacchi informatici. Prima di definire un piano di sicurezza di un'infrastruttura, è necessario stabilire quali servizi richiedono protezione e quali componenti di un asset vanno protetti. I principali componenti dell'asset sono i seguenti:

- Dati: ovvero le informazioni scambiate tramite i servizi, non quelle salvate sul disco;

- Risorse ICT: composte da tutti componenti hardware;
- Luogo: dove l'infrastruttura è posizionata;
- Risorse umane: tutte le persone di un'organizzazione rivestono un ruolo chiave nella sicurezza della stessa, dato che, probabilmente, sono in possesso di informazioni sensibili (es. credenziali condivise);

Una volta definiti gli asset, è necessario definire i possibili attacchi che potrebbero minacciarne l'integrità, grazie all'utilizzo di un cyber range. Si tratta un'infrastruttura hardware e software che consente di eseguire un ambiente simulato per l'esecuzione di test e misurazioni applicabili al mondo reale.

Il lavoro di tesi approfondisce e mostra tutti gli aspetti considerati nella progettazione e nello sviluppo di un sistema SCADA virtuale per il controllo ed il monitoraggio di un prototipo di acquedotto. Particolare attenzione è stata posta nella progettazione dell'infrastruttura di rete di sensori, nella logica di controllo del controller e nello sviluppo di un'applicativo web raggiungibile da remoto per la visualizzazione ed il controllo del modello fisico.

La tesi è suddivisa come segue. Nel capitolo 2, si approfondisce il concetto di cyber range ed i tipi di test che vengono effettuati per verificarne il funzionamento e la sicurezza. Proprio come ogni forza militare o di polizia, è infatti necessario un "poligono" per affinare le armi e le tecniche di attacco. Tutte le nuove tecnologie (es. reti domestiche, reti di sensori), che sono state sviluppate e pensate per interagire con la rete, necessitano di un serio banco di prova, per essere testate e valutate prima della loro distribuzione.

Nel capitolo 3, si prosegue con la descrizione della progettazione logica del modello fisico, dei suoi componenti, dell'infrastruttura di rete e dei protocolli utilizzati per la comunicazione.

Nel capitolo 4 e 5, si descrivono le caratteristiche principali del framework Niagara, utilizzato per la realizzazione del sistema SCADA, e della modellazione della logica di controllo del controller.

Nel capitolo 6, si descrive lo sviluppo dell'applicativo web e le sue funzionalità.

Infine, nel capitolo 7, si conclude discutendo gli obiettivi di ricerca ed i possibili sviluppi futuri.

Capitolo 2

CyberRange

La società moderna, dipende ormai, da sistemi informativi e da reti complesse. Le reti collegano utenti, aziende, compagnie, organizzazioni, governi ed i loro dipartimenti. Un'infrastruttura di rete riveste un'importanza critica in servizi come: polizia, protezione civile, servizio sanitario, controllo e distribuzione di acqua, gas, elettricità, telefono, radio e TV, tutti collegati tra loro. L'aggiunta di troppi dispositivi di difesa della rete non è solo costoso, ma aggiunge complessità, rischiando di rendere il sistema inutilizzabile per il suo scopo. Maggiore complessità implica inoltre l'introduzione di possibili errori di configurazione, aumentando il rischio di rendere il sistema meno sicuro di prima. Per il fixing delle vulnerabilità, si fa affidamento alla valutazione e all'applicazione degli aggiornamenti tramite download, sperando che questi non interrompano le funzionalità essenziali dell'intero sistema e che possano ridurre il numero di bug sfruttabili rispetto a prima. Un attaccante solitario, dotato di PC economico e di tool open source, può causare danni ingenti in qualsiasi sistema informativo del mondo. I sistemi attaccati possono diventare instabili o non funzionanti. Le conseguenze possono essere fisiche (sistema fisico compromesso), finanziarie, ambientali (es. sversamenti e perdite di sostanze nocive) e causare perdite ai sistemi a loro volta dipendenti (es. interruzione distribuzione energia elettrica che causa l'indisponibilità di sistemi di telecomunicazione e servizi finanziari). Per rendere tali sistemi più sicuri dagli attacchi di rete, è necessario creare delle difese perimetrali a protezione delle vulnerabilità. E' necessario considerare anche gli attacchi dell'interno, per esempio da utenti insoddisfatti. In passato, i sistemi critici per la sicurezza non erano collegati alla rete. Oggigiorno una connessione ad Internet offre vantaggi significativi, come usare la rete per connettersi su installazioni o sistemi fisici geograficamente distribuiti, consentendo il controllo e il monitoraggio a distanza, senza costi aggiuntivi (es. costi di viaggio), con un feedback immediato sull'output del sistema. Anche i sistemi che ancora

oggi si collegano su reti locali dedicate sono in pericolo; spesso da qualche parte dentro l'infrastruttura di rete esiste una macchina che, per errore, effettua il bidge dell'Internet pubblico con la rete privata. Come nel caso di *Stuxnet*, la connessione fatale può consistere "semplicemente" nell'utilizzo condiviso di due chiavette USB tra due calcolatori su diverse reti. Questo caso, mostra in maniera evidente, che la strategia del "non connettere" nella pratica presenta i suoi limiti; sono dunque necessari altri tipi di accorgimenti per rendere un sistema sicuro. E' possibile proteggere le reti aggiungendo software e dispositivi quali antivirus, firewall e sistemi per la prevenzione di intrusioni e provare a correggere le vulnerabilità note. Questo tipo di soluzione presenta anche degli svantaggi: l'aggiunta di troppi dispositivi di difesa non è solo costoso, ma come detto prima, rischia di rendere il sistema inutilizzabile per il suo scopo, aggiungendo complessità e possibili errori di configurazione. [22]

2.1 Valutazioni di sicurezza un sistema

Per sistemi anche di moderata complessità, una valutazione di tutte le sue caratteristiche di sicurezza, risulta essere sorprendentemente difficile. Nella fattispecie, i motivi di queste difficoltà sono:

- Il comportamento di sistemi complessi può essere influenzato dall'interazione dinamica tra i componenti del sistema stesso ed il loro ambiente. Spesso e volentieri, ciò genera degli effetti collaterali imprevedibili, dipendenze sottili ed accoppiamenti che modificano il comportamento generale in forme impossibili da prevedere mediante analisi statica o tramite il test isolato dei singoli componenti.
- Il comportamento del sistema dipende dalle politiche dei processi, i quali dovrebbero essere schedulati correttamente, ben documentati ed implementati in termini di dipendenze e conseguenze; ma spesso non lo sono. Il cambio delle policy e dei processi, inoltre, avviene in maniera più frequente rispetto all'hardware utilizzato e alla sua configurazione, rendendo più difficile prevedere i problemi.
- I sistemi interagiscono con ambienti esterni, ed eventuali problemi di caricamento e temporizzazione di queste interazioni, possono modificare il modo in cui il sistema si comporta ed espone guasti in cascata;
- Intenzionalmente o non, utenti del sistema, amministratori o semplici manutentori, possono inavvertitamente violare alcune principi impliciti

del design di sistema. Alcuni strumenti specializzati nel log, l'analisi e la riproduzione del traffico di rete, possono aiutare nella simulazione e valutazione di un alto volume di traffico dell'ambiente, in modo da ridistribuire nella maniera migliore il modello dell'ambiente. La funzionalità di *reset*, è una delle caratteristiche fondamentali di un *cyber range*, perchè permette ai test di non essere vincolati da possibili infezioni causate da malware o crash del sistema, che compromettono dati ed integrità.

Solitamente, nella valutazione della sicurezza di un sistema, si ricorre ad audit e test di prenetazione. Entrambi i metodi, hanno però mostrato carenze evidenti, che hanno portato alla definizione del concetto di *cyber range*. [22]

2.2 Il concetto di cyber range

Un cyber range, è un'infrastruttura hardware e software che consente di eseguire un ambiente simulato per l'esecuzione di test e misurazioni applicabili al mondo reale. Senza l'utilizzo di un sistema del genere, sarebbe impossibile eseguire attività di formazione, addestramento, valutazione degli studenti sui sistemi in produzione. Un cyber range può fornire esercizi di formazione avanzati sulla cyber-security per studenti universitari e professionisti, al punto da rivoluzionare l'educazione e l'approccio alla cyber sicurezza. Proprio come ogni forza militare o di polizia, è necessario un "poligono" per affinare le armi e le tecniche di attacco. Infatti, tutte le nuove tecnologie (es. reti domestiche, reti di sensori), che sono state sviluppate e pensate per interagire con la rete, necessitano di un serio banco di prova, per essere testate e valutate prima della loro distribuzione [17]. Tali attività di testing, possono essere estese su un'ampia gamma di ambiti, inclusi i diversi livelli di comunicazione di rete (es. i protocolli di rete), i vari paradigmi computazionali (es. cloud, web, mobile) e i vari domini applicativi (es. sistemi di controllo industriali, sistemi informativi aziendali) [2]. Un cyber range generico utilizza la virtualizzazione della piattaforma per offrire funzionalità di base per l'implementazione di appliance virtuali, configurare topologie di reti flessibili e permettere l'emulazione di diverse caratteristiche della rete. I cyber range, solitamente si servono di cloud-controllers come *OpenStack* e *VMWare vSphere*, come strumenti di configurazione e gestione della rete. Tali strumenti consentono di creare e gestire dei nodi fisici, simulazioni e servizi di rete tra macchine virtuali multiple. Quando un cyber range è utilizzato nei training per la cybersecurity, una LAN collega i diversi nodi tra loro.

Il processo di creazione di un cyber range prevede:

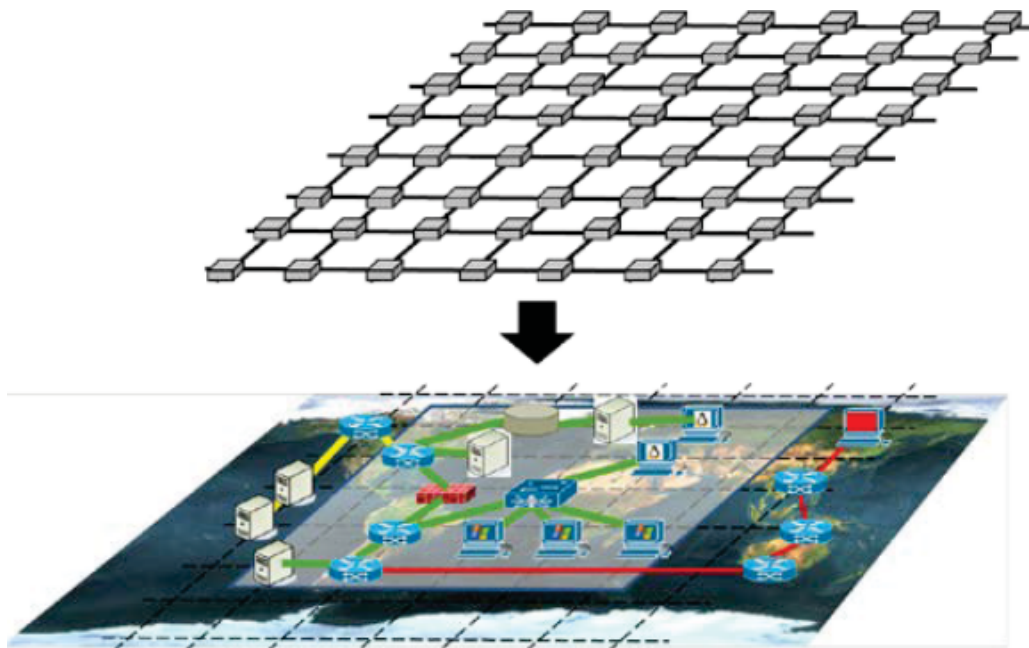


Figura 2.1: Trasformazione di una griglia computazionale(in alto) in un modello rappresentativo di un sistema e del suo ambiente(in basso)

- L'avvio e l'inizializzazione di immagini virtuali;
- L'installazione del contenuto di sicurezza al loro interno;
- La creazione di un servizio di rete;
- La creazione di macchine virtuali multiple;

Un cyber range necessita di monitorare sia macchine virtuali che reti virtuali in real-time. Tramite l'acquisizione del traffico, la sua analisi e visualizzazione, gli utenti possono comprendere i dettagli del processo di formazione ed avere una maggiore consapevolezza dal punto della cybersecurity in quel momento. L'approccio tipico prevede l'utilizzo di tecniche forensi host-based, che consistono in alcuni agenti che incorpora una macchina virtuale. Esistono tre tipologie di cyber range:

- fisici: ricreano fedelmente un'infrastruttura di rete e di calcolo, garantendo il massimo livello di fedeltà; di contro, presentano dei costi piuttosto elevati e scarsa flessibilità nell'utilizzo;
- virtuali: in questo caso tutti i componenti sono simulati; sfruttano la tecnologia sopracitata, ovvero quello della virtualizzazione (es.VMWare

vSphere, OpenStack), per simulare testbed di complessità arbitraria. I cyber range virtuali, possono essere basati su hardware dai costi contenuti (es. di acquisizione e di gestione) e facilmente reperibili in commercio. E' garantito un elevato livello di flessibilità e scalabilità, in quanto gli scenari simulativi possono essere definiti dinamicamente in base alle esigenze simulate; di contro, le prestazioni sono inferiori rispetto alle controparti fisiche. Ciò può essere significativo nell'esperienza simulativa, soprattutto nei casi in cui si va ad eseguire un attacco finalizzato alla compromissione dell'erogazione del servizio (es. gli attacchi DDoS).

- ibridi: coniugano la dinamicità e la flessibilità della virtualizzazione con l'impiego di dispositivi fisici. Ad esempio un cyber range ibrido può utilizzare la piattaforma di virtualizzazione per simulare server e terminali, integrandola con reti di sensori, dispositivi di calcolo fisici, adibiti al controllo di infrastrutture fisiche difficilmente simulabili (es. il sistema di controllo di un acquedotto).

Affinchè sia garantito il necessario livello di realismo negli scenari simulativi implementati, sono necessari dei generatori di traffico (componenti software e hardware) in grado di dare automaticamente origine sia a del traffico benigno che malevolo. Il primo fornisce il contesto su cui si nasconde il secondo. Questi risultano rivestire un'importanza cruciale, in quanto la reazione e la prontezza agli attacchi dipendono dalla capacità con cui si riesce a distinguere il traffico. Un accurato modello di cyber range, può mostrare un comportamento simile a quello di un sistema reale. Come accennato prima, ciò permette di testare scenari ipotetici, ad esempio introducendo deliberatamente guasti o disconnessioni di componenti o aggiungerne di nuovi per analizzare la reazione del sistema nell'ottica di un'espansione futura. Un ottimo test consiste nell'analizzare il comportamento del sistema al variare del flusso di traffico. L'aggiunta di nuovi componenti come workstation, sensori, o l'aggiunta di soglie per la notifica all'interno di un sistema industriale, può causare evidenti cambiamenti nel flusso di traffico critico. Ciò può avvenire in parti inaspettate del sistema (es. un router che tenta di reindirizzare il traffico, causando guasti negli altri router in sovraccarico), rispetto a dove sono stati aggiunti i componenti. In pochi secondi, questo può portare ad un potenziale collasso della maggior parte della rete di sistema. Il cyber range a tale proposito, risulta ideale per lo studio di colli di bottiglia, hotspot e scenari ipotetici, principalmente per due motivi:

- Diagnostica: dispone degli strumenti necessari per eseguire un'approfondita analisi sulle cause e le dinamiche di un collasso;

- Isolamento: il crash del modello di sistema non danneggia il sistema reale;
- Simulazione: è facile simulare l'aggiunta di nuovi componenti ad un sistema per analizzare gli spostamenti del flusso di traffico (es. aggiunta di un percorso di backup per il traffico);

[5] [22]

2.3 Progettazione e testing

In fase di progettazione, l'architettura del cyber range deve essere necessariamente quanto più simile ad un sistema reale. Se così non fosse, i risultati, ottenuti durante la fase di testing, non potrebbero essere considerati attendibili. La prima sfida consiste nel raccogliere quanti più dettagli sull'architettura di un sistema reale. Molto spesso questo risulta essere sorprendentemente difficile, in quanto i sistemi reali tendono a crescere nel tempo e per lunghi periodi: espansione non accompagnata da un'adeguata documentazione. Di conseguenza è richiesto un grosso sforzo per ricreare o convalidare una porzione dell'architettura in maniera ragionevolmente accurata. In alcuni casi, la verifica del sistema reale, è effettuata tramite strumenti di mappatura di rete. La seconda sfida è decidere il livello di virtualizzazione del sistema, valutando vantaggi e svantaggi sull'utilizzo delle macchine virtuali (VM). In molti casi, vengono utilizzate le copie delle VM di un sistema reale, in quanto assicurano il setting di configurazioni, versioni e patch di un sistema reale, ma con la possibilità di effettuarne dei test. Gli svantaggi consistono nel fatto che, spesso e volentieri, le VM contengono dati personali o confidenziali, che devono essere rimossi o sostituiti con informazioni di testing. A volte è più semplice ricreare una VM da zero, cercando di applicare le stesse patch e configurazioni, piuttosto che cercare di risolvere i problemi di una copia. Al momento di decidere il livello di virtualizzazione da adottare, è importante considerare i suoi possibili effetti collaterali:

- Accoppiamenti indesiderati: eseguire più macchine virtuali sui stessi server fisici, potrebbe dare origine inevitabilmente a più LAN virtuali sulla LAN fisica causando alcuni accoppiamenti indesiderati;
- Isolamento: se il carico della CPU di un VM sale, non vale il principio di isolamento, di conseguenza, le prestazioni delle altre VM in esecuzione sulla stessa macchina fisica sarebbero influenzate dall'eccessivo carico elaborativo della CPU;

- Larghezza di banda: l'invio di una grande mole di traffico su una LAN virtuale, ridurrebbe in maniera significativa la banda disponibile per tutte le altre LAN virtuali sullo stesso cavo;

Per quei componenti che non possono essere virtualizzati, è necessario utilizzare dei componenti fisici simili o identici. Tale necessità si manifesta nel caso in cui si intenda misurare parametri come: il carico della CPU o il throughput dei pacchetti. Per questo motivo, l'architettura del cyber range prevede l'integrazione di hardware aggiuntivo ed un'infrastruttura di rete flessibile affinché gli possa essere offerta la connettività necessaria per aggiungerlo al modello. Un cyber range, solitamente ha la capacità di archiviare e salvare dei modelli e le configurazioni, e di ripristinarli quando necessario. All'interno dell'economia dei test, ciò risulta essere davvero utile, dato che, all'avvio di una nuova fase di testing, risulta molto più veloce aggiornare un modello già esistente con le opportune modifiche, piuttosto che ricreare un modello da zero [22].

2.3.1 Tetsbed

I cyber range sono equipaggiati con strumenti di attacco e difesa. E' facile trovare strumenti per la difesa perimetrale o interna come firewall, sistemi anti malware e strumenti di attacco come script per le vulnerabilità, malware o backdoor da iniettare sui bersagli. I cyber range permettono di agli utenti di sperimentare tutti questi tool in un ambiente controllato, senza andare ad impattare sui reali sistemi fisici in produzione [5]. I testbed educativi sono diventati negli anni, una componente fondamentale nell'educazione alla sicurezza informatica. Come descritto nel paragrafo precedente, nuove tecnologie come la virtualizzazione ed il cloud computing hanno facilitato l'implementazione di test simulativi, con annessa rapidità nel rilascio delle risorse e nelle operazioni di reset. Nell'educazione alla sicurezza informatica e negli esercizi, i testbed sono utilizzati per sviluppare le abilità necessarie per raggiungere determinati obiettivi di formazione e per l'esercizio. Ad esempio, per la definizione di un testbed per un sistema SCADA, per insegnare ed allenare sui vari incidenti di sicurezza informatica, o per la sperimentazione su di essa. Nonostante i testbed siano stati ben sviluppati dal punto di vista teorico, in letteratura sono presenti ben poche informazioni sulla progettazione ed implementazione di testbed per il training e l'esercizio sulla cyber security, al fine di comprendere le decisioni architetturali ed i requisiti per lo sviluppo di quest'ultimi. Implementazioni nell'ambito accademico e di ricerca spesso forniscono il codice sorgente, ma solitamente è abbastanza complesso capirne l'impostazione e la documentazione. Inoltre, la configurazione di questi,

può richiedere tempi piuttosto lunghi, soprattutto per i non esperti. Un'altra motivazione è data dal fatto che molti testbed di sicurezza informatica, sono stati sviluppati in ambito militare, spesso per range nazionale. All'interno di un testbed, può essere istanziata una sfida da risolvere da parte dell'utente. Esempi di sfide proposte allo sfidante sono quelle note come *capture the flag* (CTF) o sessioni di training sulla sicurezza IT. Implementare un testbed, a cui possa partecipare sia un principiante che uno studente esperto è davvero difficile. Inoltre, nuove tecnologie già citate come il cloud computing, permettono la manutenzione e l'installazione di testbed, con costi largamente inferiori rispetto a 10 anni fa. Al giorno d'oggi dunque, è molto più facile impostare e rimuovere dei testbed, ma risulta ancora piuttosto complicato crearne di sofisticati ed adeguati.

Ogni testbed di solito ha un design ed un'architettura individuale; ne esistono varie tipologie:

- *Hands-on network security exercises, iCTF*: un testbed per l'educazione alla sicurezza della rete, che fa uso di esercizi per studenti laureati. Questo è stato ulteriormente sviluppato nel corso degli anni e utilizzato per il Concorso UCSB iCTF;
- *DEFCON CTF 3 o Google CTF 4*: testbed utilizzati per proporre sfide in cui l'obiettivo consiste nell'acquisire un determinato segreto;
- *Experimental research in cyber security*: testbed come il DETER, ovvero un progetto sviluppato per la ricerca sperimentale nella sicurezza informatica;
- *Smart grid and SCADA testbed*: testbed che si concentrano sulla verifica della corrente elettrica (es. SCADA). Questi spesso combinano dispositivi fisici con sistemi virtuali al fine di impostare degli esperimenti (es. sistemi di alimentazione virtuale) [6].

2.3.2 Design tipico per cyber security testbed

La figura 2.2 mostra una panoramica sul layout tipico di un testbed per la sicurezza informatica. Oltre l'infrastruttura ICT virtuale, vengono fornite allo sfidante alcune sfide di sicurezza di diverse tipologie. Queste infatti possono avere contenuti diversi (es. web, crittografia) a seconda degli obiettivi della sfida. Solitamente la loro tipologia ed il loro contenuto è stabilito dal personale amministrativo e dai trainer (es. docenti universitari o professionisti nella formazione sulla sicurezza informatica).

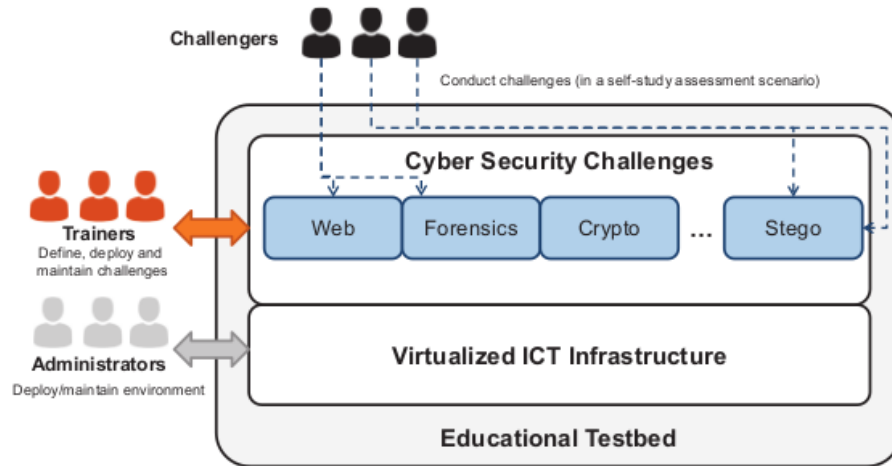


Figura 2.2: Esempio di setup di un testbed sulla cyber security

Gran parte della cyber security consiste nell'analisi dei sistemi software per quanto riguarda le vulnerabilità o l'esame di dati o programmi. L'obiettivo di queste analisi consiste nell'insegnare alcuni principi specifici sulla sicurezza informatica tramite la pratica. Il più delle volte lo sfidante deve ottenere la "bandiera" segreta, ovvero un segreto che rappresenta l'obiettivo della sfida. Questo rappresenta un mezzo davvero formante per i professionisti della sicurezza. In questo modo possono imbattersi in problemi comuni, provando a risolverli ed acquisendo allo stesso tempo una conoscenza su come e perchè si verificano, all'interno di un ambiente simulato e circoscritto, senza compromettere o esaminare la complessità di un sistema reale. [6] [20]

2.3.3 Tipologia di sfide

Esistono diverse tipologie di sfide sulla sicurezza informatica. Di seguito sono descritti i tipi più comuni:

- *Web*: come suggerisce il nome, le sfide riguardano siti ed applicazioni web. All'interno di essi, uno sfidante deve trovare delle vulnerabilità per ottenere la bandiera segreta o accedere al server hosting del servizio stesso. In questo modo, è possibile capire i concetti base e le pratiche più usate all'interno del contesto web;
- *Forensics*: è un tipo di sfida che riguarda l'analisi dei dati. Uno sfidante ha l'obiettivo di trovare alcune informazioni che si nascondono all'in-

terno di registri, nel traffico di rete o in memoria. Le abilità apprese in questa categoria, possono essere utili e messe in pratica in scenari in cui si agisca in risposta ad un incidente;

- *Crypto*: la crittografia rappresenta una parte fondamentale della sicurezza informatica. All'interno di queste tipologie di sfide, lo sfidante impara i concetti fondamentali sui protocolli di crittografia. Sfide di questo genere richiedono solitamente allo sfidante di attaccare e "rompere" implementazioni crittografiche errate, affinché possa decifrare il testo cifrato. In questo modo, lo sfidante acquisisce una consapevolezza maggiore sul complesso mondo della crittografia;
- *Reverse*: per gran parte degli analisti di sicurezza, il *Reverse Engineering* rappresenta un'abilità importante da possedere, soprattutto nell'analisi dei programmi malevoli. Sfide di questo genere richiedono allo sfidante di lavorare su un file binario affinché possa rivelare il flag segreto;
- *Exploitation*; sfida molto simile a quella del Reverse Engineering. Rispetto a quella appena citata, questa sfida pone maggiore attenzione sulla ricerca di vulnerabilità di un certo dato binario o di un'applicazione. Un esempio di questa categoria di sfide è rappresentato dal "set user id" con buffer overflow, dove lo sfidante deve eseguire comandi arbitrari con privilegi elevati. Le abilità acquisite in questo genere di sfide, possono essere riutilizzate in scenari di attacco e di difesa del sistema informatico;
- *Stego*: l'arte di nascondere i dati è nota come stenografia. In alcune situazioni, la presenza, all'interno di un sistema informatico, di un'informazione, anche se cifrata, rappresenta un grosso rischio. Ad esempio, nei regimi soppressivi, gli attivisti potrebbero ottenere delle informazioni sulla loro elettronica, il che è ritenuto illegale dal governo. Questi sarebbero costretti a decifrare tutte le loro informazioni, dato che la cifratura non li proteggerebbe da un'eventuale accusa. Le sfide Stego, richiedono ad uno sfidante di trovare tali dati. Le sfide solitamente utilizzano delle tecniche di stenografia mal applicate. Tecniche di stenografia ben progettate, richiedono risorse enormi per essere violate [6];

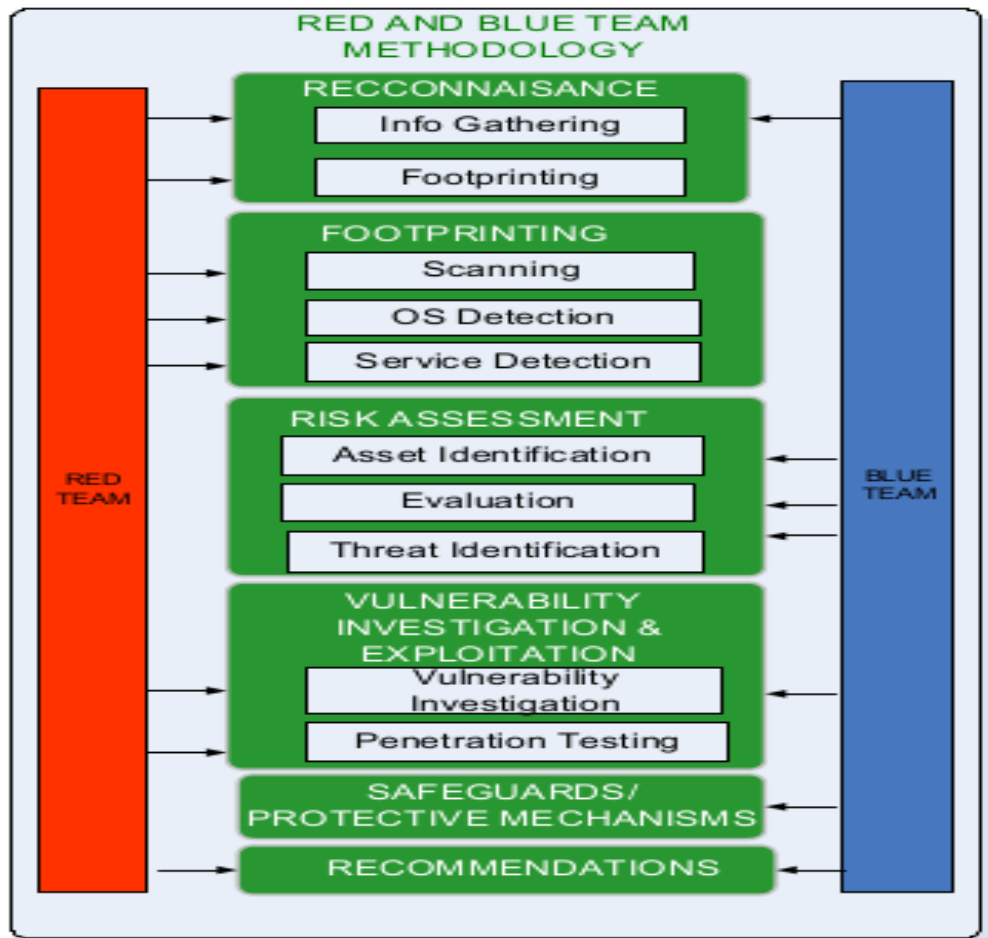


Figura 2.3: Metodologia Red e Blue Team

2.4 Training e valutazione

La valutazione del livello di sicurezza di un'organizzazione è diventata negli anni una priorità fondamentale nella la difesa dei propri sistemi informativi. I dati e le risorse informative sono diventati elementi davvero preziosi di un'azienda, e la protezione di tali risorse è uno degli aspetti principali che deve essere affrontato al suo interno [7]. Uno degli strumenti per eseguire una valutazione sull'attuale livello di sicurezza è quello di utilizzare due squadre: red team e blue team. I compiti e gli obiettivi di tali squadre possono essere visti come *hacking etico*, secondo il quale un'azienda è portata ad assumere un esperto di sicurezza informatica per indagare sulle possibili aree vulnerabili del sistema informativo aziendale. Palmer afferma come le organizzazioni si siano rese conto che uno dei modi migliori per ottimizzare

la sicurezza del proprio sistema è quello di assumere degli esperti di sicurezza informatica che tentano di penetrare i loro sistemi per valutarne la sicurezza e le vulnerabilità [13]. Il *red teaming* è una tecnica comunemente usata in ambito militare per scoprire delle vulnerabilità del sistema o per trovare delle lacune sfruttabili al suo interno, con l'obiettivo principale di ridurre le possibili sorprese, migliorare e garantire la maggior robustezza possibile delle tecniche difensive. [15] E' bene sottolineare come il red team esplori gli aspetti tecnici attraverso i quali il sistema può essere manipolato, mentre la controparte blu, si focalizza sul progettare un approccio manageriale e delle tecniche operative per proteggere il sistema. In sede di simulazione, lo spazio di comportamento è diviso in due categorie: una controllata dalla squadra rossa, che rappresenta l'insieme dei comportamenti malevoli, mentre il blue team rappresenta il gruppo di difensori. Durante la simulazione della squadra rossa, quando si investiga tecnicamente sulle vulnerabilità, è anche possibile formulare possibili soluzioni dal punto di vista del blue team. Per questo motivo, la raccolta di informazioni è un'attività che coinvolge entrambe le squadre per studiare gli obiettivi e l'impatto su un sistema informativo. Ad esempio, le minacce identificate dalla squadra blu potrebbero essere sfruttate dalla squadra rossa durante un test di penetrazione. Partendo da questa consapevolezza, la squadra blu può studiare questi attacchi per prevenirne di futuri. La metodologia combinata della squadra blu e rossa punta a fornire una base di sicurezza che possa offrire informazioni dettagliate del sistema, definendo una panoramica sulle vulnerabilità sfruttabili e sulle strategie difensive. La sicurezza è un delicato equilibrio tra protezione, disponibilità ed accettazione dell'utente. Le squadre rosse e blu contribuiscono alla sicurezza, in quanto rivelano i mezzi tramite i quali i sistemi possono essere penetrati e poi messi in sicurezza [9]. Oltre il red e blue team, è prevista la disposizione di altre tre squadre: la white, green e yellow team. Per quanto riguarda la prima, essa è preposta alla progettazione e costruzione dello scenario utilizzato per la simulazione. La green team invece, ha il compito di gestire l'intera infrastruttura. Infine la yellow team, si occupa di fornire una descrizione di alto livello sullo stato di avanzamento della simulazione [2].

2.5 Cyber Physical Systems

Negli ultimi anni, i sistemi cyber-fisici (CPS) hanno attirato una notevole attenzione a causa della loro intrinseca combinazione di aspetti fisici e cibernetic, in particolare, per quanto riguarda l'aspetto della sicurezza [1] [11]. Una delle principali sfide è quella di ripensare la sicurezza da una nuova prospettiva, dove gli aspetti cyber, umani e fisici siano integrati nel loro insieme.

Nei sistemi CPS, il tempo riveste un'importanza cruciale; per questo motivo, gli aspetti che riguardano la comunicazione in tempo reale sono fondamentali. In particolare, i sistemi di controllo industriale (ICS), appartengono ad una classe di sistemi cyber-fisici in cui le entità cyber (PLC, calcolatori) hanno il controllo diretto sui processi di controllo e sulle azioni fisiche. Eventuali attacchi informatici su tali sistemi potrebbero causare gravissime conseguenze fisiche, direttamente sulla popolazione (es. blackout, disastri nucleari) [1]. Per questo motivo, la sicurezza nei sistemi ICS è di vitale importanza, dato che un'eventuale loro compromissione porterebbe ai problemi citati, causando interruzioni di servizio, ingenti perdite economiche ed infine la messa a repentaglio della popolazione e gli ecosistemi naturali. La comunità di ricerca sul campo della sicurezza informatica, non ha ancora identificato degli strumenti di monitoraggio configurabili che possono essere adattati a diversi scenari di ricerca basati sui CPS e sugli aspetti della comunicazione in real-time. I sistemi CPS presentano configurazioni molto diverse tra loro in base ai loro domini di appartenenza. Spesso sono implementate soluzioni di monitoraggio e di controllo dove i client interrogano i server ogni 10 secondi per visualizzare in grafica i valori aggiornati. E' evidente, che questo tipo di soluzione non rispetta i requisiti necessari affinché sia definita in real-time; inoltre comporta un carico di elaborazione non necessario, che in casi estremi potrebbe portare ad un arresto anomalo del client. La figura 2.4, mostra un esempio di modello di riferimento di un sistema CPS. L'architettura è suddivisa in 3 livelli: rete utente, rete aziendale e rete di produzione [14]. Il livello di rete dell'utente (Livello 3), noto anche come rete di consumatori, è responsabile della raccolta e dell'analisi dei requisiti degli utenti (es. smartphone, Internet app e mobile).

Il livello di rete (Livello 2), ha il compito di svolgere tutti i processi legati alle attività aziendali (es. pianificazione della produzione, gestione operazioni e transazioni finanziarie). Tale livello è coerente con la tecnologia dell'informazione tradizionale, compresa la distribuzione di servizi e sistemi come FTP, siti web, mail server, enterprise resource planning (ERP) e sistemi di automazione degli uffici. Il livello di produzione (Livello 1) include componenti e funzioni dedicate al monitoraggio, controllo, rilevamento e manipolazione dei processi fisici. Tale livello può essere ulteriormente in tre sotto-livelli: LAN di controllo di supervisione (es. human-machine interfaces (HMI), workstation, database), rete di controllo (es. programmable logic controllers (PLC), controller di sistemi di controllo distribuiti e unità terminali remote), rete I/O (es. sensori, attuatori e processo controllato). La rete di produzione è direttamente connessa con la rete aziendale, ma completamente isolata dalla rete di utenti nell'ambiente di automazione industriale tradizionale. Nonostante ciò, la rete di produzione deve essere consistente, combinandosi con la

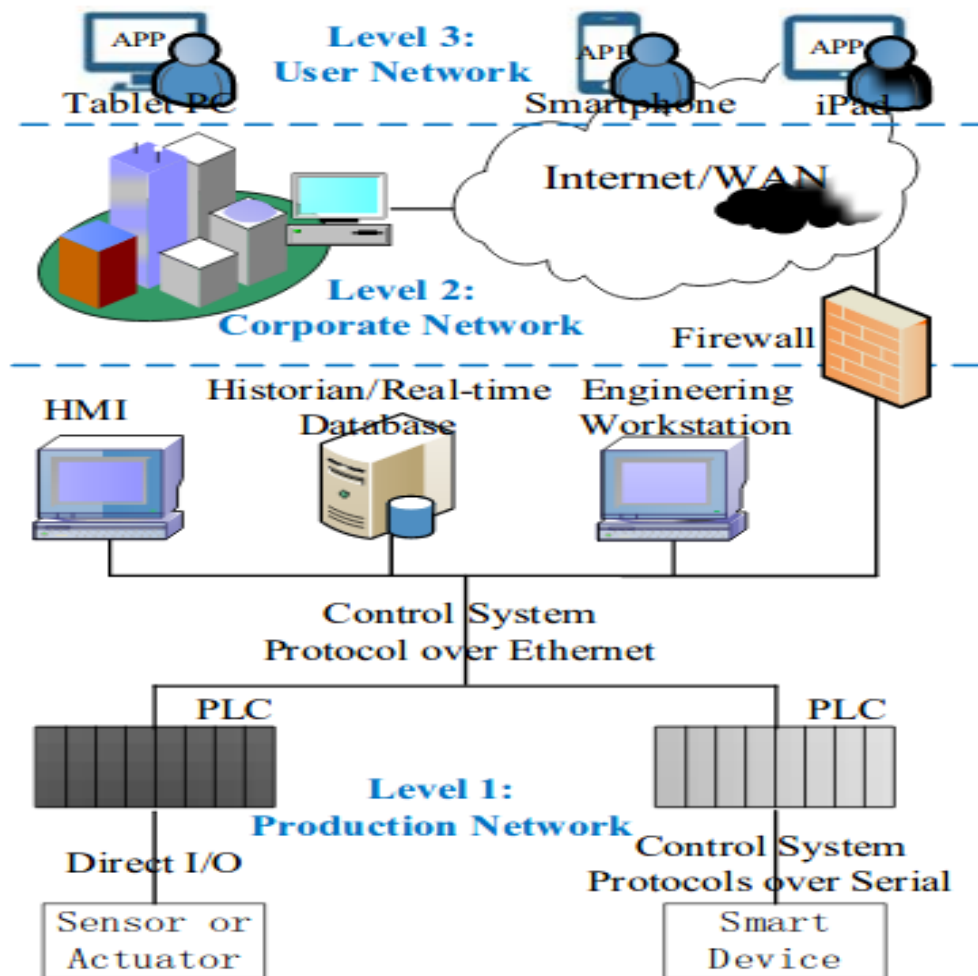


Figura 2.4: Architettura di alto livello di un CPS

rete aziendale e la rete di utenti nell'ambiente industriale. I protocolli del sistema di controllo (es. RS-232, RS-485) devono appartenere a degli standard ed essere compatibili con Internet TCP/IP o un'altra architettura di protocollo uniforme. Inoltre, per caratteristiche, l'introduzione di una rete aperta non fa che introdurre problemi di sicurezza all'interno della rete aziendale, di produzione e di utenti.

2.6 Prospettive future

L'utilizzo di cyber range rappresenta una scelta obbligata per le compagnie interessate alla formazione avanzata del proprio personale nell'ambito della

sicurezza informatica, alla valutazione della capacità di difesa, non ultima, allo sviluppo di soluzioni innovative per la cyber security. La progettazione e la realizzazione di un cyber range, come descritto nei paragrafi precedenti, implica uno sforzo gravoso, a causa della sua complessità e per il numero di risorse coinvolte, economiche e umane. Essa richiede il coinvolgimento di esperti in diversi ambiti della sicurezza informatica, come la realizzazione di modelli di complessità realistica, la generazione di test prodotti da operatori umani e la valutazione tramite punteggio, da assegnare alle singole attività svolte dai partecipanti. Fatta questa premessa, è evidente che la capacità di automatizzare, implementare nuove tecniche e strumenti per realizzare tali attività, rappresentano il prossimo passo importante da effettuare per l'architettura del cyber range della prossima generazione [2].

Capitolo 3

EVA

EVA è un acronimo che sta per "Emulatore Vero Acquedotto", il che rappresenta un modello funzionante di un sistema di approvvigionamento dell'acqua. Questo capitolo mostra la struttura del sistema fisico realizzato e le scelte di design per la realizzazione di un sistema SCADA (Supervisory Control And Data Acquisition) virtualizzato. Partendo da una panoramica generale del sistema fisico, sono poi evidenziate le caratteristiche principali dei componenti utilizzati:

- Controller: JACE 8000, controller IoT embedded e piattaforma server per la connessione multipla di diversi dispositivi e sotto-sistemi;
- Sensori e attuatori: sfruttati a livello di campo e subordinati al controller. Si tratta di componenti hardware compatibili con Raspberry ed opportunamente modificati per questo modello;
- Connessioni e comunicazione: la comunicazione sfrutterà il protocollo Modbus su TCP/IP tramite l'utilizzo di un router per creare una rete locale e coordinata.

Successivamente sono illustrate le principali scelte implementative nella realizzazione del sistema, presentando i dettagli e gli *use cases* dell'applicazione web realizzata ad-hoc per il monitoraggio e la supervisione del sistema fisico. Nel capitolo 5 sarà poi mostrata la logica di controllo installata all'interno del controllore ed i moduli realizzati nel Workbench Niagara, per testarne l'efficienza in fase di simulazione.

3.1 Struttura EVA

Un sistema reale di distribuzione è mostrato nella figura 3.1. Nella prima fase del processo, l'acqua è raccolta da una fonte, che rappresenta un fiume

o un lago; l'acqua è poi pulita all'interno dell'impianto di trattamento nel quale essa viene convertita in acqua potabile tramite l'utilizzo di filtri e la somministrazione di prodotti chimici. Una volta completato il trattamento, viene conservata all'interno di un serbatoio di stoccaggio fino a quando l'utente finale non ne farà utilizzo.

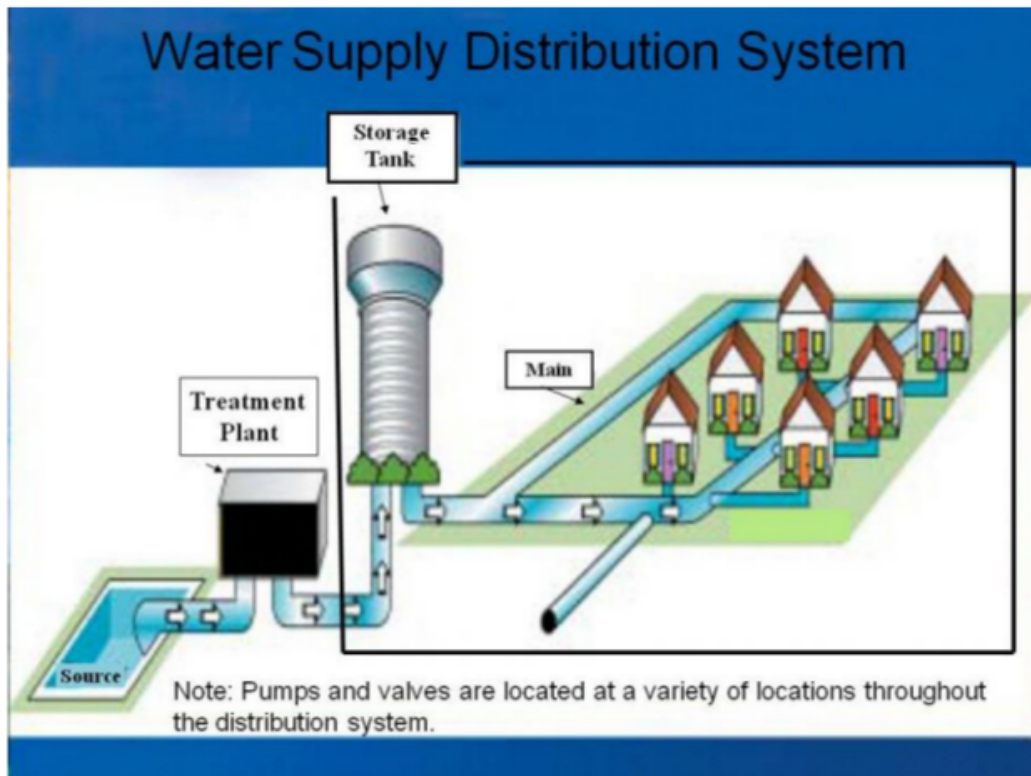


Figura 3.1: Architettura sistema reale di approvvigionamento idrico

Il sistema fisico è stato progettato in accordo con l'architettura della 3.1. Sono stati utilizzati cinque serbatoi per simulare il modello reale, e sono i seguenti:

- *Environment*: rappresenta la fonte naturale contenente l'acqua proveniente da qualsiasi fenomeno ambientale;
- *Lake*: rappresenta la fonte dove, l'acqua proveniente dall'environment, viene raccolta;
- *Purifier*: rappresenta il contenitore dove l'acqua rimane in attesa del trattamento di purificazione;

- *Clean Water*: è il serbatoio dove l'acqua potabile è conservata dopo il trattamento di depurazione;
- *House*: rappresenta l'utilizzo dell'acqua da parte del consumatore, il quale ha accesso al sistema di distribuzione;

3.2 Model Design

Come è possibile notare dalla figura 3.2, il cuore pulsante del sistema fisico si compone di tre settori: lake, purifier e clean water. Ogni serbatoio è connesso all'altro tramite delle tubature. All'interno del modello, l'acqua può fluire soltanto verso un'unica direzione: dal lake, passando dal purifier e verso il serbatoio contenente l'acqua potabile. L'acqua è in grado di raggiungere il serbatoio successivo, grazie all'utilizzo di pompe o dell'effetto di gravità. Il serbatoio clean water, è dotato internamente di due valvole, una elettrica e l'altra manuale, affinché, in situazione critiche, possa essere interrotta l'utenza d'acqua.

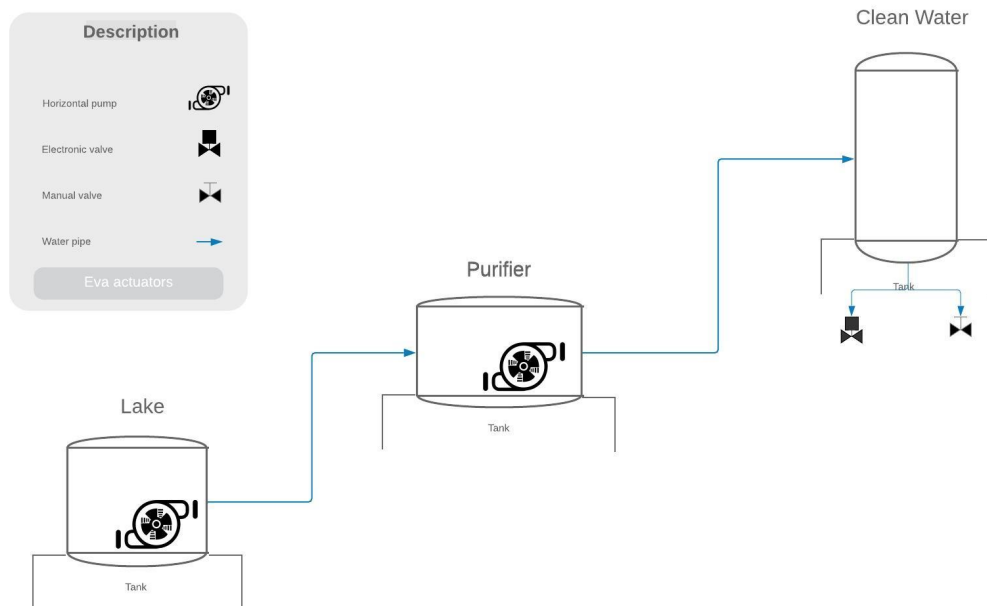


Figura 3.2: Architettura base di EVA

La figura 3.3 mostra che, per controllare specifici parametri del modello funzionante, sono stati introdotti dei sensori di diverso tipo al suo interno: sensori chimici e fisici. I primi rivestono un ruolo critico, affinché si possa

monitorare la composizione dell'acqua, in modo da soddisfare gli standard stabiliti dall'autorità di regolamentazione. I sensori fisici, invece, sono essenziali, dato che senza di essi, sarebbe impossibile controllare, ad esempio, il flusso d'acqua attuale nel modello o la quantità di spazio disponibile all'interno del serbatoio.

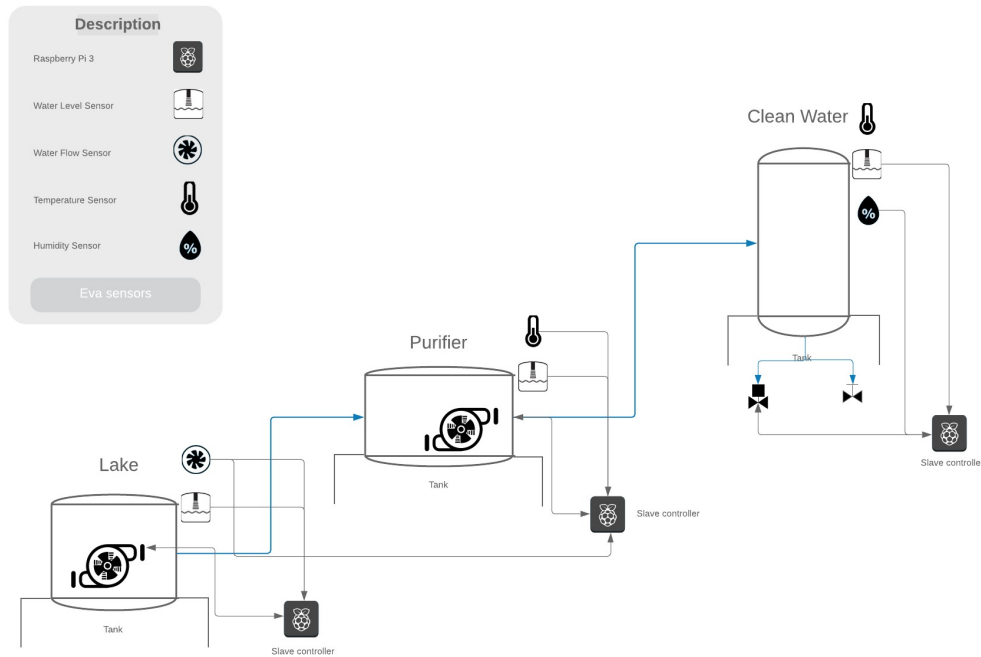


Figura 3.3: Modello con sensori e Raspberry Pis

All'interno del modello è stato necessario includere un set di piccoli controllori in grado di pilotare gli attuatori hardware e leggere i valori dei sensori, seguendo una specifica logica di funzionamento. Inizialmente sono state introdotte 3 Raspberry Pis; questi computer, seppur piccoli, risultano molto performanti, e sono in grado di leggere i dati dai sensori di campo e di attivare o meno gli attuatori, come le pompe e le valvole, agendo direttamente sugli interruttori collegati ad essi. Ogni serbatoio ha richiesto l'utilizzo di uno di questi computer. Le Raspberry Pis sono programmate per eseguire un codice scritto in Python con un differente logica di funzionamento personalizzata rispettivamente per i diversi tipi di sensori.

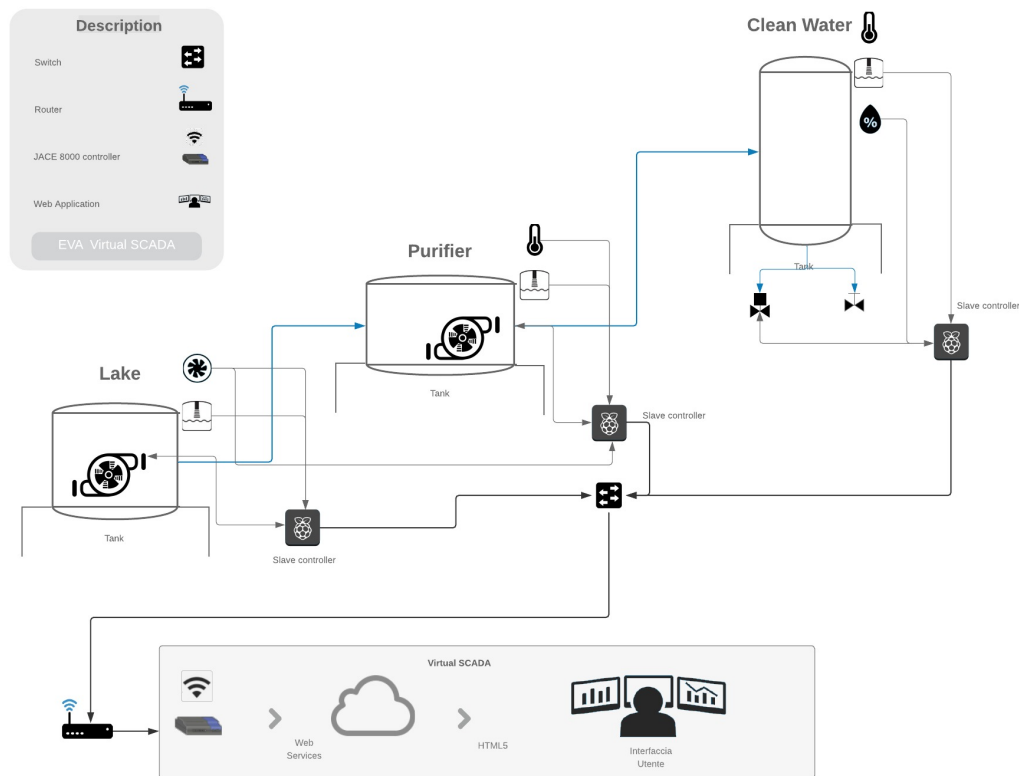


Figura 3.4: EVA Virtual SCADA

Dalla figura 3.4 è possibile osservare la configurazione finale dell'architettura e notare le interconnessioni tra tutti i componenti. E' stata realizzata una LAN con topologia a stella; le Raspberry svolgono il ruolo di slave, aventi il compito di pilotare e leggere i valori dei sensori di campo. Tutte le board sono connesse direttamente alle porte di ingresso di uno switch; quest'ultimo a sua volta è collegato fisicamente ad un router con accesso ad Internet. Il ruolo del master è svolto dal controller JACE 8000. Tramite l'utilizzo di questo particolare device è possibile implementare una Web Application con le medesime funzionalità di uno SCADA, con i vantaggi che una sua virtualizzazione comporta. L'utente, in base ai privilegi di cui dispone, sarà in grado, da remoto, di supervisionare l'intero struttura: data logging, allarmi, scheduling, gestione della rete e manutenzione della logica di controllo dell'intero impianto.

Il controller, come mostrato in figura 3.5, riveste il ruolo di master, ed è posizionato al centro nella topologia logica della rete . Tutte le Raspberry

Pi, ai vertici della topologia adottata, ricoprono il ruolo dello slave. Tutte le board sono connesse tra loro attraverso una connessione WIFI con accesso ad Internet. Lo standard adottato per la condivisione dei dati tra controller e le board è Modbus. Questo protocollo di comunicazione, come descritto nel paragrafo 2.4.4, è al giorno d'oggi uno dei più utilizzati in ambito industriale. E' risultato dunque evidente fare questo tipo di scelta, rispetto a tutti gli altri, affinché si potesse realizzare un sistema simile a quello reale.

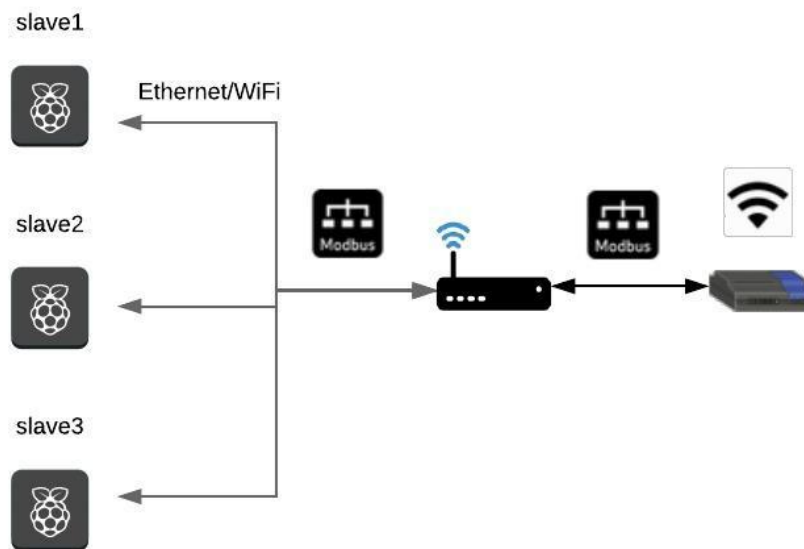


Figura 3.5: Comunicazione Raspberry Pi con Modbus over TCP/IP

Le Raspberry Pi sono fisicamente connesse ai sensori e attuatori. Nella prima fase, i dati sono letti dai primi, per poi essere incapsulati ed inviati al controller attraverso la rete. Il controller colleziona e legge i valori provenienti dai sensori di campo, e stabilisce, in base alla sua logica di controllo, se attivare o meno gli attuatori. Come è evidente, le board non hanno nessun potere decisionale nel processo di attivazione di quest'ultimi, ma bensì il controller, il quale è l'unico che dispone di una panoramica complessiva dell'intero sistema acquedottistico.

3.3 Componenti EVA

In questa sezione, saranno descritti in dettaglio i componenti utilizzati all'interno del modello EVA.

3.3.1 JACE 8000 controller

La JACE 8000, è un controller IoT compatto ed una piattaforma server per la connessione multipla di diversi device e sotto-sistemi. Dotata di una connessione Internet ed una capacità di Web-serving, il controller offre un controllo integrato, funzioni di supervisione, data logging, allarmi, schedulink e network management. Raccoglie e trasmette i dati provenienti dai sensori di campo, permettendone la visualizzazione tramite una grafica ricca tramite Web Browser, via LAN wireless, o da remoto tramite Internet.

HARDWARE SPECIFICATIONS

TI AM3352: 1000MHz ARM® Cortex™-A8
1GB DDR3 SDRAM
Removable micro-SD card with 4GB flash total storage/2GB user storage
Wi-Fi (Client or WAP) IEEE802.11a/b/g/n IEEE802.11n HT20 @ 2.4GHz IEEE802.11n HT20/HT40 @ 5GHz Configurable radio (Off, WAP, or Client) WPA2PSK/WPA2PSK supported
USB type A connector Back-up and restore support
(2) isolated RS-485 with selectable bias and termination
(2) 10/100MB Ethernet ports
Secure boot
24VAC/DC power supply
Runs Niagara 4.1 and later
Real time clock
Batteryless



Figura 3.6: (a) Specifiche hardware JACE 8000. (b) JACE 8000 controller

La JACE 8000 controller, opera con Niagara4, l'ultima versione del Framework Niagara, affinché siano garantite ottime performance. Il Supervisor Niagara può essere utilizzato con tale controller per aggregare informazioni,

includendo dati in tempo reale, archiviazioni e allarmi, per creare una singola applicazione unificata [19].

3.3.2 Raspberry Pi

Una board Raspberry Pi è un computer single-board dotato di una CPU, una RAM ed uno slot per la card SD per la memorizzazione di dati. I componenti hardware dipendono dal modello scelto; in tal caso, si tratta di una Raspberry Pi 3 che offre 1.2 GHz Quad Cortex A53 CPU ed 1GB di RAM. La figura 3.7, mostra lo schema di una board Raspberry; essa dispone di diverse porte di comunicazione(es. HDMI, USB, Ethernet port), WiFi e Bluetooth [10].

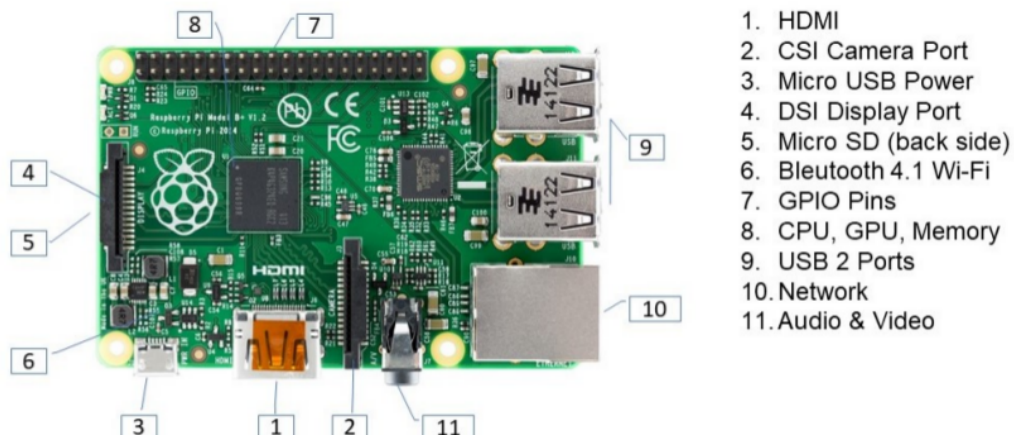


Figura 3.7: Raspberry Pi 3 Schema

3.3.3 Sensori

Nel contesto del sistema di approvvigionamento idrico il trattamento delle acque è un fattore chiave: deve seguire le linee guida mondiali dell'organizzazione della salute in termini di dosaggio corretto dei micronutrienti. La qualità dell'acqua è definita come solidi disciolti totali (TDS) che consiste sia in sali inorganici, ad es. Cloruro di calcio, magnesio e potassio, che materia organica. Quest'ultimi sono parzialmente disciolti in acqua e rappresentano contaminanti e micronutrienti. I sensori ionici aiutano a determinare se l'acqua è potabile, cioè privo di contaminanti e con la percentuale corretta di micronutrienti [3] .

Sensori chimici

- *Temperature Sensor - Waterproof (DS18B20)*: sonda di temperatura sigillata, consente di misurare con precisione le temperature in ambienti umidi con un'interfaccia 1-Wire, che deve essere connessa al microprocessore;
- *Smart Water Ions Sensor Board*: specializzato nella misurazione del dosaggio degli ioni per il controllo della qualità dell'acqua potabile, nel trattamento delle acque e nel monitoraggio dell'acqua agricola;
- *Silver Ion (Ag+) Sensor Probe*;
- *Iodide Ion (I-) Sensor Probe*;
- *Cupric Ion (Cu2+) Sensor Probe*;
- *Chloride Ion (Cl-) Sensor Probe*;
- *Bromide Ion (Br-) Sensor Probe*;
- *Nitrate Ion (NO3-) Sensor Probe*;
- *Fluoroborate Ion (BF4-) Sensor Probe*;
- *Fluoride Ion (F-) Sensor Probe*;
- *Calcium Ion (Ca2+) Sensor Probe*;
- *Conductivity sensor*;
- *Dissolved Oxygen sensor*;
- *Oxidation Reduction Potential sensor*;
- *pH sensor*;

Sensori fisici

- *Sensore Ultrasuoni*: per controllare e coordinare le pompe e le valvole, risulta necessario ricavare il livello dell'acqua registrato all'interno dei serbatoi. Il sensore ad ultrasuoni utilizza le onde ultrasuoni emesse dalla testa di quest'ultimo. Questo riceverà in seguito, l'onda riflessa dall'ostacolo più vicino. Per calcolarne la distanza, vengono considerati due momenti nell'asse temporale: il primo corrisponde al TRIG, ovvero quando l'onda viene emessa, mentre il secondo è definito ECHO, cioè quando l'onda ritorna alla testa del sensore.

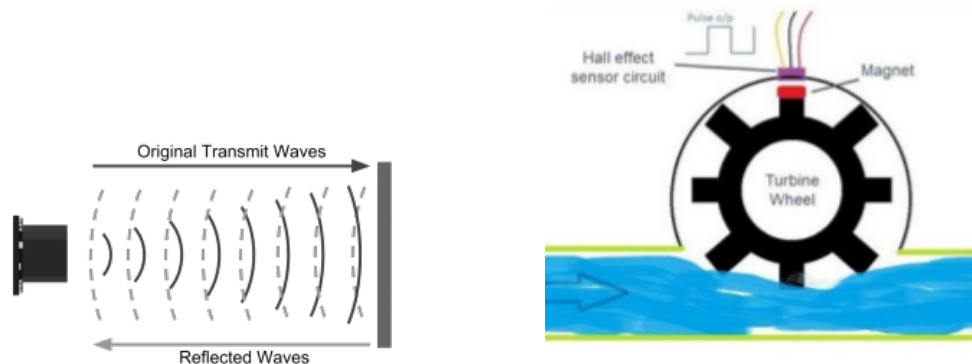


Figura 3.8: (a) Sensore ultrasuoni. (b) Sensore di flusso

- *Sensore di flusso*: nel modello è necessario misurare la quantità d'acqua che scorre da un serbatoio all'altro. La Raspberry è stata programmata affinché calcolasse quanti litri d'acqua sono passati attraverso il tubo. Questo sensore si trova in linea di galleggiamento: utilizza una piccola girandola con un piccolo magnete attaccato che a sua volta genera un effetto Hall misurato da un sensore magnetico, in grado di contare quanti giri ha fatto la girandola. Il movimento del fluido, è misurato considerando che ogni impulso prevede lo spostamento di circa 2.25 mm di acqua corrente.

3.3.4 Attuatori

Il modello proposto, ha diversi serbatoi per diversi scopi. Il "Lake" rappresenta il serbatoio dove l'acqua viene recuperata dal ciclo naturale, per poi essere purificata e trasferita su una cisterna per la sua distribuzione sull'area urbana. Per questo proposito, tramite la rete fisica connessa, l'acqua deve essere trasferita da un serbatoio all'altro tramite l'utilizzo della tubatura. Le cisterne sono però poste a diversi livelli di pendenza e di altezza. Per tale ragione, è indispensabile l'utilizzo delle pompe e delle valvole, pilotate dal controller tramite Raspberry. La lista degli attuatori è pertanto composta da:

- Solenoide Valve: controllata da remoto per l'apertura e chiusura del servizio di erogazione;
- Pompa centrifuga: multistadio ad asse orizzontale;

3.3.5 Modbus

E' stato necessaria la realizzazione di una LAN dedicata, per isolare il modello fisico e permettere la cooperazione tra controller e le Raspberry Pi. Per questo motivo è stato utilizzato un router WiFi ed un IP statico per ogni componente del modello (RPi, controller). In particolare si è optato per l'utilizzo del protocollo Modbus over TCP/IP invece della versione seriale, dato che quest'ultima versione richiede una complessità fisica maggiore rispetto a quella utilizzata [16]. A tale protocollo è assegnato un numero di porta IP (502), comprese le intestazioni IP e TCP, seguite da header e payload specifici di Modbus. L'architettura con Modbus over TCP/IP, garantisce un basso costo e flessibilità in termini di data link e physical layer; possono essere istanziate sia una LAN o una WAN. In entrambi i casi si rende necessario l'utilizzo di un router. Modbus, abbandona il concetto di "client-server" valido per alcuni protocolli, proponendo quello di "master-slaves", presente in altri protocolli utilizzati in ambito industriale come MQTT.

3.4 TestBench

La convalida dell'architettura avviene attraverso un processo per verificare se il modello realizzato soddisfa i requisiti di testing. Tramite questo, è possibile controllarne eventuali incoerenze e complessità. Affinchè potessero essere testate le funzionalità ed il comportamento del prototipo, è stato necessario considerare specifici casi d'uso che permettessero di analizzare la reazione di un sistema chiuso, come quello proposto nella figura 3.4, con l'ambiente esterno. I casi sopracitati, interagiscono con il modello, simulando operazioni di:

- *input*
- *output*

Per quanto riguarda la prima tipologia, si sono considerati due sotto-tipologie di interazione con il sistema:

- *Environmet*: rappresenta l'acqua proveniente da tutti i tipi di sorgente del ciclo naturale;
- *SCADA User*: tutte le operazioni eseguite dall'utente tramite l'utilizzo del *Virtual SCADA*;

L'ultima tipologia, quella degli output, considera anch'essa altre due sotto-tipologie:

- *House*: rappresenta l'utilizzo da parte del consumatore dell'acqua conservata all'interno del serbatoio di acqua pulita *Clean Water*;
- *Fault*: tutti i fattori di perdita d'acqua (evaporazione, perdite) presenti lungo le condotte distributrici che collegano la rete di distribuzione all'impianto idrico a servizio delle singole utenze;

Tramite l'utilizzo del Framework Niagara, è stato possibile simulare i casi appena descritti non solo *fisicamente*, tramite previa installazione dei serbatoi e delle condotte aggiuntive al modello, ma anche *virtualmente*, sfruttando la possibilità, offerta dal framework, di aggiungere dei moduli simulativi all'interno della logica del controller.

3.4.1 Interazione con l'ambiente

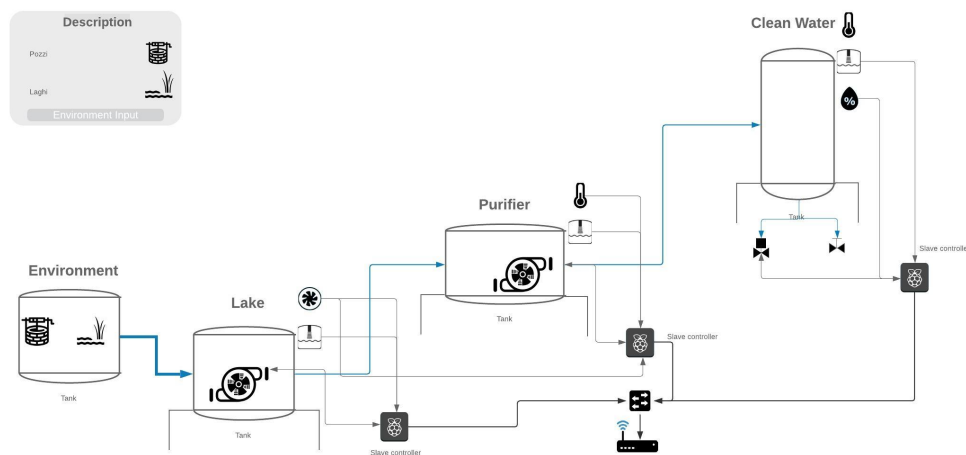


Figura 3.9: Environment input

Come mostrato nella figura 3.9, l'architettura prevede l'installazione di un quarto recipiente denominato "Environment". Questo è fondamentale per simulare l'interazione del modello con l'ambiente esterno. Il serbatoio *Lake*, svolge il ruolo di opera di presa, in corrispondenza della quale avviene la captazione dell'acqua dalle sorgenti naturali [21]. Solitamente tali opere differiscono tra loro a seconda del tipo di acque captate. L'aggiunta di un ulteriore serbatoio come quello di *Environment*, ha lo scopo, per l'appunto, di simulare i diversi tipi di acque provenienti dal ciclo naturale. Nella simulazione proposta sono state considerate principalmente due tipi di sorgenti d'acqua:

- *Acque di superficie*: fiumi, laghi, acqua di transizione, corpo idrico artificiale, ecc;
- *Acque sotterranee*: sorgenti, pozzi, falde, ecc.

Anche gli impianti di trattamento delle acque, affinché siano rese idonee al consumo umano, cambiano in funzione del tipo di acqua in ingresso; solitamente vi è un impianto di potabilizzazione, nel caso di captazione di acque superficiali, e impianti di semplice clorazione nel caso di acque sotterranee [21].

3.4.2 Interazione con lo SCADA *User*

Per il monitoraggio e la supervisione del sistema fisico, l'utente potrà autenticarsi tramite browser o tramite applicazione web, al *Supervisory Control And Data Acquisition* (SCADA) realizzato tramite il Framework Niagara. Ogni utente avrà uno o più privilegi, i quali si contraddistinguono per il tipo di operazioni che quest'ultimo potrà eseguire. In sede di progettazione sono stati definiti principalmente quattro ruoli:

- *SCADA Programmer*: dispone dei privilegi necessari per modificare la logica di controllo del controller di sistema;
- *EVA Manager*: dispone dei privilegi per accedere alla *control room* virtuale, dedicata all'acquisizione dei dati e la fruizione del sistema nel suo complesso;
- *Testbench Administrator*: dispone dei privilegi per la modifica o la creazione dei moduli simulativi e del file di configurazione del testbench;
- *EVA Guest*: dispone dei privilegi di semplice monitoraggio e visualizzazione del sistema;

I ruoli appena descritti sono assegnati manualmente, ad ogni nuovo utente registrato, solamente dall' *EVA Manager*, che sarà l'unico a disporre dei permessi necessari per la modifica del file di configurazione riguardante gli utenti.

SCADA *Programmer*

L'utente con il ruolo di *SCADA Programmer*, è l'unico a disporre dei permessi necessari per la creazione o modifica della logica di controllo del controller. Questo tipo di intervento potrebbe risultare necessario nei casi in cui

si proceda ad un upgrade del modello esistente, mediante l'aggiunta di nuovi serbatoi, sensori e attuatori. Chiaramente, in questo caso, l'utente dovrà necessariamente modificare o ricreare un algoritmo di controllo che tenga conto delle nuove aggiunte. Un ulteriore tipo di intervento potrebbe rendersi necessario nell'eventualità in cui l'utente non ritenga soddisfacente l'algoritmo esistente. La figura 3.10, mostra un'esempio dei casi d'uso e le operazioni che l'utente potrà eseguire una volta completato il login dal WorkBench di Niagara tramite web browser.

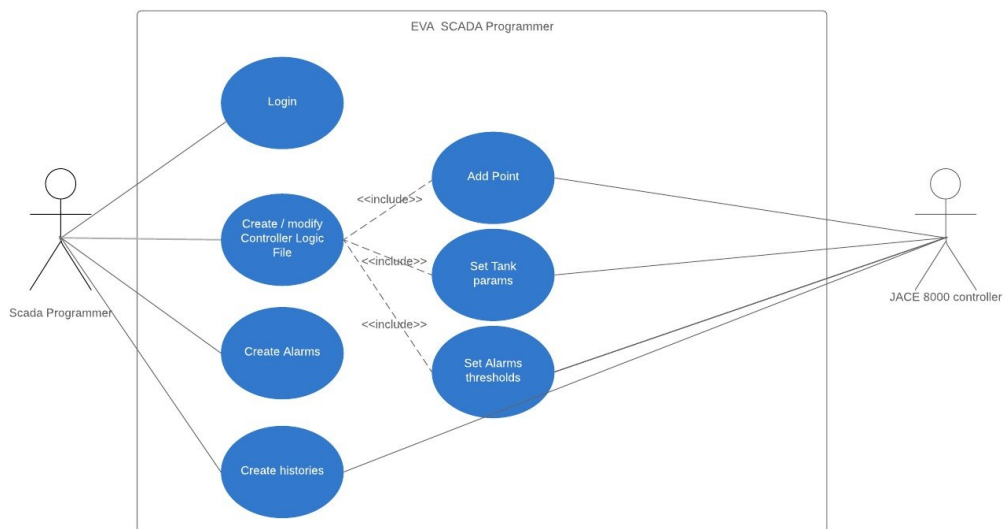


Figura 3.10: UML Use Case SCADA Programmer

Una volta effettuato l'accesso, l'utente sarà in grado di accedere ai file di configurazione del controller. Potrà così creare una nuova logica di funzionamento di quest'ultimo, o più semplicemente modificare quella già esistente. All'interno del file sarà possibile, ad esempio: aggiungere un nuovo sensore o attuatore a livello logico (Point), settare l'altezza dei serbatoi utilizzati ed impostare le soglie secondo le quali, superato un certo range di valori, sia dato origine ad un particolare tipo di allarme. L'utente potrà inoltre definire diverse classi di allarmi, una per ogni sensore e attuatore del modello.

EVA Manager

L'utente con il ruolo di *Manager*, potrà eseguire operazioni di supervisione, acquisizione e telecontrollo tramite interfaccia web mediante l'utilizzo di un web-browser. A questo proposito è stata realizzata un'applicazione web creata *ad-hoc* per il modello proposto.

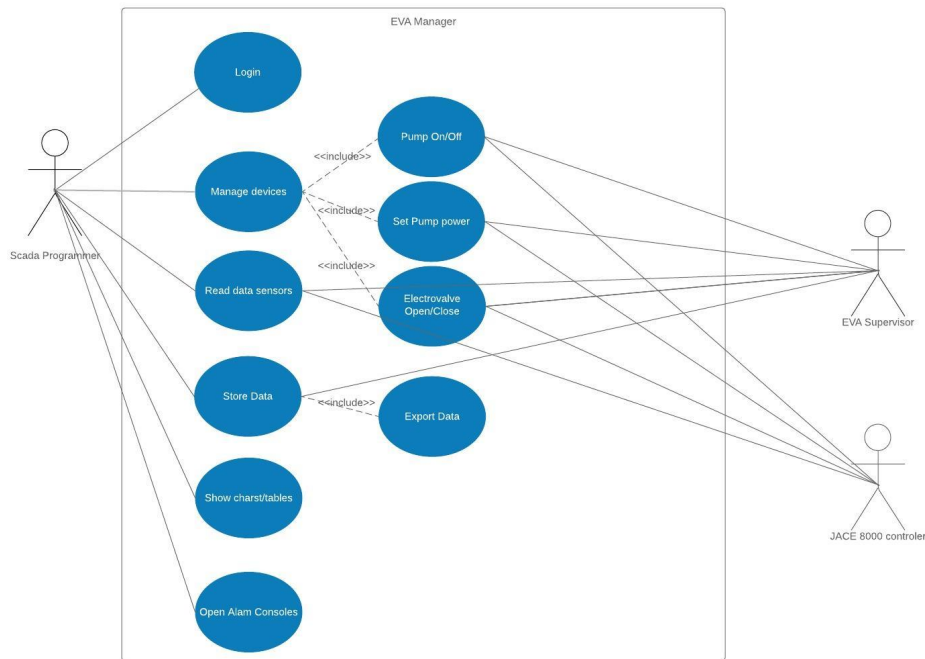


Figura 3.11: UML Use Case EVA Manager

Come è possibile notare dalla figura 3.11, una volta eseguito l'accesso, l'utente sarà in grado di acquisire i dati mediante opportuni driver di comunicazione verso le gli apparati di campo (Raspberry Pi), a loro volta connessi al controller ed ai sensori e attuatori. I dati scambiati sono normalmente grandezze analogiche o digitali, oppure stringhe di testo. Le interrogazioni eseguite dal controller sono sincrone (polling), così da essere in grado di cogliere una qualsiasi variazione di una misura o di uno stato considerato di interesse per il sistema SCADA.

Tramite interfaccia web è possibile inviare comandi al sistema di automazione mediante l'utilizzo di semplici bottoni; sarà dunque possibile azionare o spegnere un dispositivo di pompaggio, dosarne la potenza, aprire e chiudere un'elettrovalvola, per riprendere o sospendere l'erogazione dell'acqua.

Appositi grafici, tabelle, indicatori offrono all'utente una chiara rappresentazione del dato, mostrato sia come grandezza, che sotto forma di animazione grafica (esempio un serbatoio con il liquido colorato che sale e scende in base alla lettura del sensore di livello).

I dati di interesse sono storicizzati sull'archivio locale del server. Tali dati, come già spiegato, saranno consultabili e visualizzabili dall'operatore direttamente sull'applicazione tramite grafici e tabelle, oppure esportati affinché

possano essere eseguite operazioni di analisi.

L'operatore ha a disposizione una console di allarme per ogni settore del modello, la quale funge da "contenitore" per i diversi tipi di allarme (modellizzati in sede di progettazione), che richiedono l'intervento dell'operatore. Ciò scaturirà nell'emissione di un suono, oppure sotto forma di icone lampeggianti, al fine di richiamare l'attenzione dell'operatore per risolvere un'eccezione di sistema non gestibile dalla logica di controllo.

Testbench Administrator

L'utente che riveste questo ruolo, dispone dei permessi necessari per la modifica dei moduli simulativi, per testare, convalidare e verificare la complessità della logica di controllo del modello realizzato.

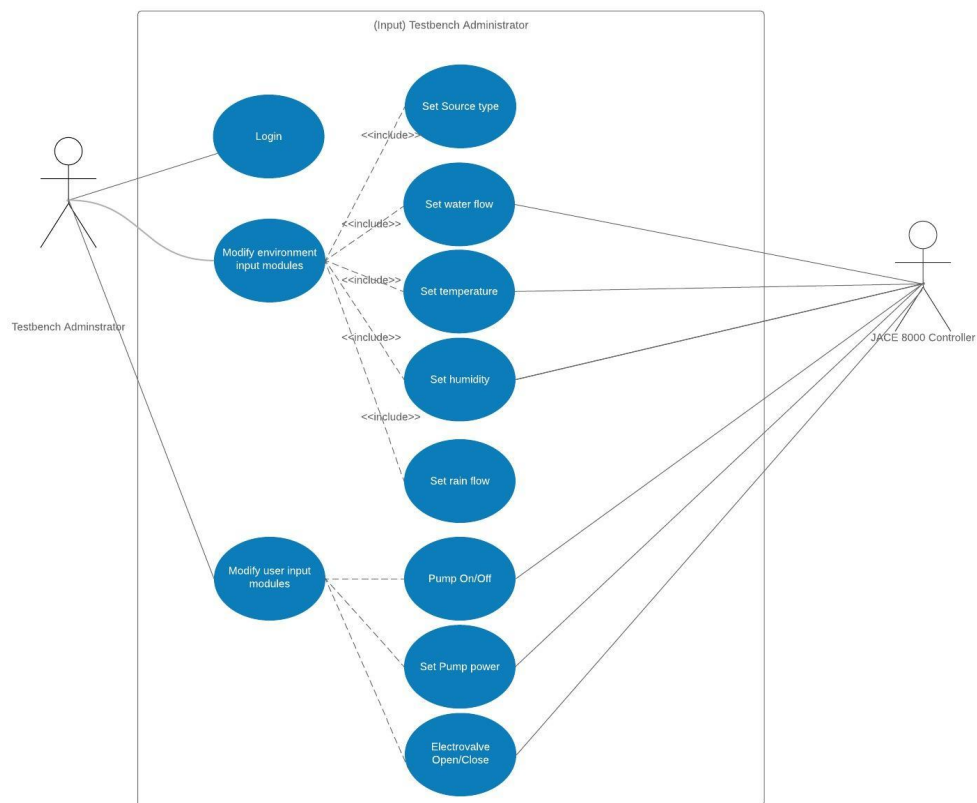


Figura 3.12: UML Use Case (input)Testbench Administrator

Come già spiegato nel paragrafo 1.5, tali moduli, consentono di simulare operazioni di *input* e *output* al sistema. Nel primo caso, come mostrato nella

figura 3.12, l'operatore, una volta terminata la fase di autenticazione all'EVA *Supervisor*, potrà agire su due moduli: *environment* e SCADA *User*.

Per il primo modulo, sarà possibile stabilire il tipo di sorgente naturale: acque superficiali o sotterranee. Nel primo caso, sarà possibile modificare tutti i valori ambientali inseriti in sede di simulazione, riguardanti i flussi d'acqua provenienti dalle superficie, la temperatura atmosferica e l'umidità. Nel caso di fonte sotterranea, sarà modificabile soltanto il flusso d'acqua, dato che parametri come temperatura e umidità perdono di significato.

Il modulo SCADA *User*, invece, risulta fondamentale per la verifica della complessità logica del sistema a fronte di comandi provenienti da un operatore con ruolo di EVA *Administrator*. Tramite esso, è possibile osservare la risposta del sistema a fronte di comandi errati da parte dell'operatore, in modo da facilitare il compito dello SCADA *programmer* nella modellizzazione dei processi di allarme e nella manutenzione della logica di controllo. Un esempio di ciò, potrebbe essere l'attivazione, da parte dell'operatore, di un attuatore, andando a causare una situazione di criticità all'interno del sistema.

I moduli simulativi di output *output* invece, consentono all'*administrator* di impostare i valori di consumo dell'acqua da parte degli occupanti e di simulare l'apertura delle valvole del servizio di erogazione idrico. Proprio il comportamento degli occupanti è tra le principali cause di aggravio dei consumi, poichè hanno il diretto controllo dei dispositivi di regolazione(es. rubinetti) [4].

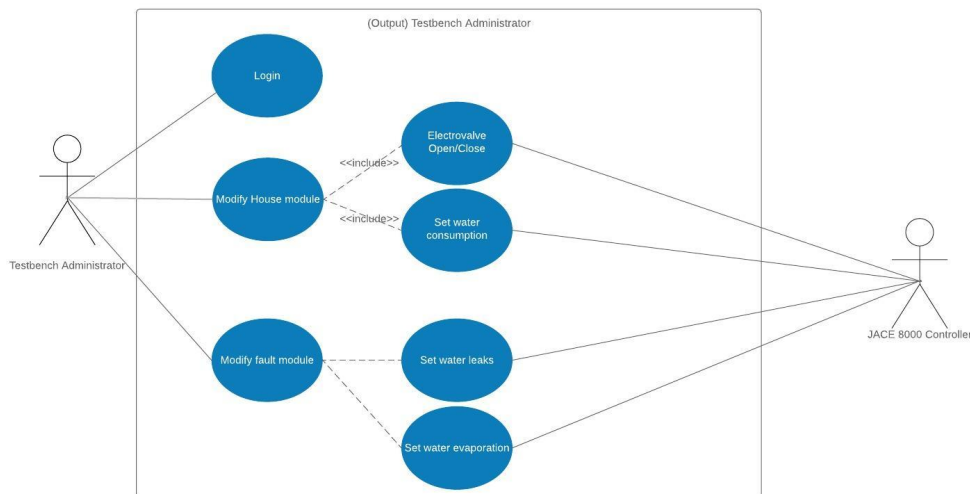


Figura 3.13: UML Use Case (output)Testbench Administrator

Per quanto riguarda il modulo di *Fault*, esso si propone di simulare dei valori di perdita dell'acquedotto. Purtroppo il volume d'acqua non utilizzato, ma disperso, non può essere misurato con una misura diretta, ma solamente attraverso una differenza di volumi d'acqua emuniti, ovvero prelevati dall'ambiente, e quello effettivamente utilizzato (ovvero fatturato con le bollette) [4],

Lungo la rete di un acquedotto, vi sono alcune utenze con riconducibili al consumo della popolazione, quali ad esempio: idranti, fontane pubbliche, scarichi e lavaggi degli impianti dell'acquedotto, ecc.. A queste vanno aggiunte le perdite vere e proprie della rete, ovvero l'acqua prelevata ma non utilizzata dall'utenza e quindi dispersa nel sottosuolo [4].

Possibili cause possono essere:

- rottura delle tubazioni;
- corrosione e/o deterioramento delle tubazioni;
- giunzioni difettose;

L'utente potrà dunque modificare ed inserire dei valori all'interno di tali moduli, al fine di simulare il volume di d'acqua disperso.

Nel caso in cui l'utente simuli in input un volume d'acqua proveniente dalle *acque superficiali* del ciclo naturale, la percentuale di evaporazione dell'acqua stagnante nel serbatoio di *Lake*, sarà considerata come fattore di perdita del modello.

Nell'eventualità in cui il *Testbench Administrator*, non ritenga sufficienti i moduli simulativi presenti nel file di configurazione, potrà aggiungerne di nuovi tramite il Workbench di Niagara accessibile da remoto

EVA Guest

Un account "Ospite" non è in possesso dei privilegi per accedere ai file di configurazione del controller o ulteriori dati privati.

Di fatto, può usufruire di funzionalità limitate che impediscono all'utente di utilizzare strumenti di sistema, apportare modifiche e inviare comandi al sistema fisico.

Come mostrato in figura 3.14, egli potrà solamente visualizzare le misure rilevate dai sensori e la loro rappresentazione, mediante grafici e tabelle, e visualizzare la console di allarme dell'acquedotto.

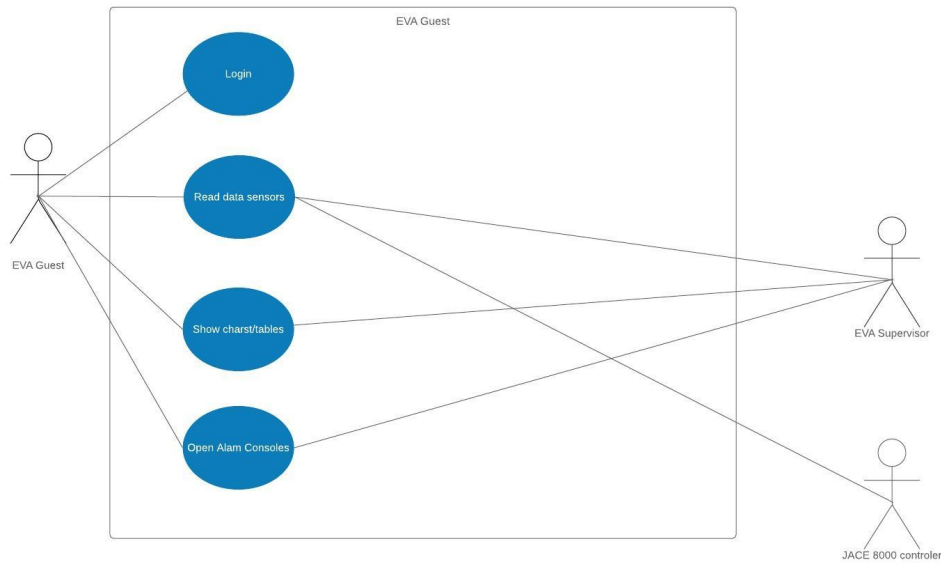


Figura 3.14: UML Use Case EVA Guest

3.4.3 Interazione con il consumatore

Per simulare l'interazione dell'utente consumatore con il modello, sono state introdotte due modalità: con un rubinetto controllato direttamente dall'utente ed una elettrovalvola. Il rubinetto è una semplice valvola che può essere chiusa e aperta manualmente. Per rendere funzionante questo componente, come mostrato in figura 3.15, è stato necessario aggiungere un ulteriore serbatoio, chiamato *House*, collocato ad un'altezza più bassa rispetto alla cisterna contenente l'acqua potabile, così da sfruttare la gravità ed evitare l'utilizzo di un'ulteriore attuatore di pompaggio. Al suo interno, sarà contenuta tutta l'acqua "consumata" dall'utente, proveniente dall'impianto idrico.

Per quanto riguarda l'elettrovalvola, l'utente è in grado di aprirla e chiuderla tramite RFID.

RFID è una tecnologia appartenente alla famiglia AIDC (*Automatic Identification and Data Capture*); un sistema RFID, in particolare, si compone di transponder, detto anche tag, il cui chip contiene i dati, di antenna, per comunicare con il tag e gestita dal controller, e di reader (o controller), che attraverso il segnale radio legge l'identificativo del tag e può scriverne la memoria, trasmettendo poi il segnale all'host.

Nel caso in figura 3.15, il RFID reader è collegato alla board, la quale controlla se il tag ID è autorizzato o meno. In caso di successo, l'elettrovalvola si apre.

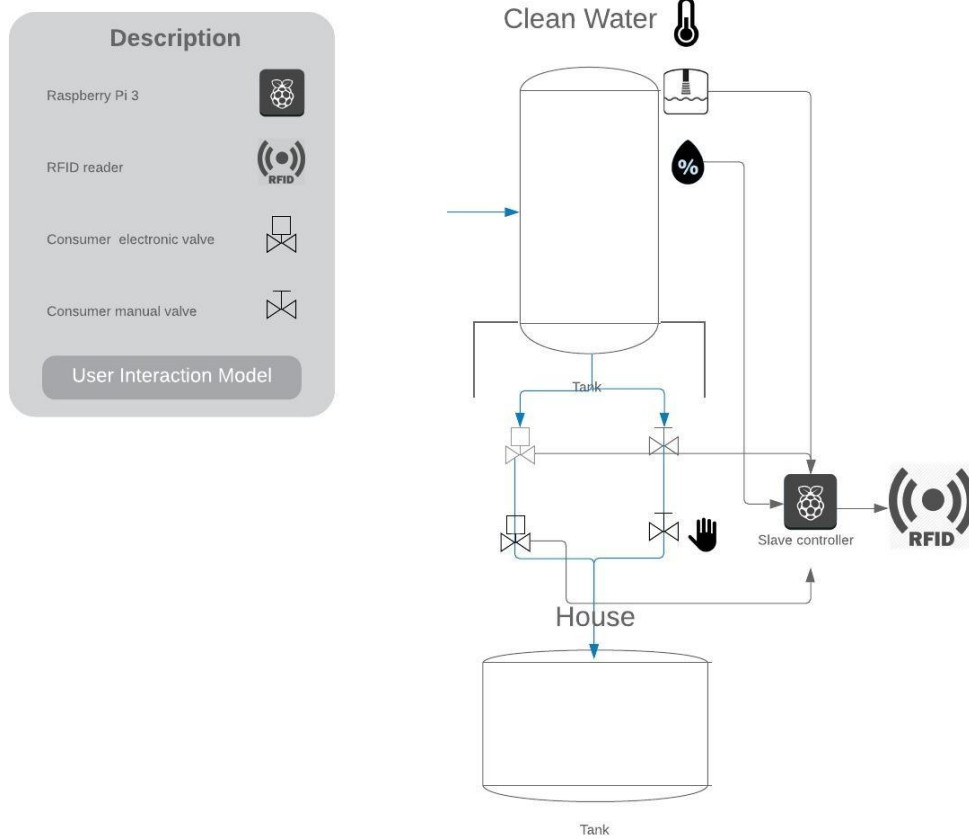


Figura 3.15: Consumer interaction model

Il consumo diretto dall'acqua da parte delle utenze, genera a sua volta, uno svuotamento proporzionale del serbatoio contenente l'acqua potabile (Clean Water), quindi un abbassamento del valore del livello d'acqua di quest'ultimo.

Capitolo 4

Niagara4

4.1 Niagara Framework

Niagara 4 è un framework software aperto creato dall'azienda Tridium, che offre diverse funzionalità ed una varietà di notevoli miglioramenti per facilitare le aziende nello sviluppo di soluzioni che sfruttino appieno l'Internet of Things, fornendo inoltre strumenti di ricerca, navigazione, sicurezza ed una visualizzazione avanzata. Questo framework sta diventando rapidamente il sistema operativo dell'IoT: collega ogni sistema e ne traduce i dati, gestendo e ottimizzando le prestazioni di servizi energetici, automazione, automazione industriale e di applicazioni machine-to-machine. Esempi di dispositivi intelligenti includono: sistemi di monitoraggio e controllo, sensori e sistemi di misurazione. La struttura del framework è caratterizzata da specifiche componenti unite ed assemblate assieme; una struttura di base; nella programmazione orientata agli oggetti, una struttura di progettazione di base riutilizzabile, composta da classi astratte e concrete, che aiuta nello sviluppo di applicazioni. Il framework Niagara è un'infrastruttura software universale che consente alle aziende di creare applicazioni *web enabled* per l'accesso, l'automazione, ed il controllo di dispositivi smart in tempo reale su Internet. Niagara integra diversi sistemi e dispositivi (indipendentemente dal produttore o dal protocollo di comunicazione) in una piattaforma unificata facilmente gestibile in tempo reale utilizzando un web browser che supporti HTML5. Sono inoltre supportate funzioni di ricerca, tagging, relazioni, modelli e gerarchie; il framework include anche un set di strumenti all'avanguardia che consente ai programmatori meno esperti di creare applicazioni efficienti in un ambiente drag-and-drop. Niagara è completamente scalabile, il che significa che può essere eseguito su piattaforme che vanno dai piccoli dispositivi embedded a classi di server enterprise [8].

4.2 Niagara Workbench

Particolare attenzione è posta all'*user experience* (UX) di Niagara. Esso è definito anche come *bajaux* framework, ovvero si basa su tecnologie aperte come ad esempio HTML5, CSS3, JSON e Bajascript V2.0. Un ambiente così composto fornisce in modo efficace un'esperienza migliorata in facilità, utility ed efficienza, nonché la capacità di produrre grafici e dashboard ricche di funzionalità. Niagara4 è progettato per essere reattivo e dinamico al fine di adeguarsi alle mutevoli circostanze di utilizzo e alle variazioni dei singoli sistemi nel tempo. I principali benefici del framework sono riassumibili in: visualizzazione dinamica, tagging e templating, e sicurezza avanzata [18].

4.2.1 Visualizzazione dinamica

L'interfaccia utente *bajaux*, basata su HTML5, vantaggiosa per l'utente finale e fornisce un'esperienza di sviluppo migliore. Gli sviluppatori possono creare facilmente applicazioni dinamiche e interattive, con viste che utilizzano widget *bajaux* che vengono visualizzate su supporti Niagara, come il Workbench o un web browser. La funzionalità di ricerca integrata nel Workbench, permette di localizzare rapidamente i dati per poi passare ad altre viste.

4.2.2 Tagging e templating

La codifica dei metadati a livello di componente consente una serie di modi per trovare i dati, tramite una ricerca e gerarchia di navigazione, oltre a fornire soluzioni per restringere i risultati tramite filtraggio. Allo stesso modo, realizzando template sfruttando il pre-tagged, i dispositivi producono una riutilizzabilità integrata che, ovviamente, si traduce in tempi di integrazioni ridotti. Utilizzando le gerarchie basate su tag, possono essere creati diversi schemi di navigazione distinti per ruolo dell'utente.

4.2.3 Sicurezza avanzata

Gli sviluppatori possono realizzare in tempi abbastanza rapidi, sistemi sicuri, utilizzando il *role-based access control* e caratteristiche di crittografia avanzata. Ulteriori funzionalità riguardante la sicurezza includono l'autenticazione basata sul tipo di connessione, la firma del codice, la quale permette di eseguire un check sui moduli, affinché non siano stati modificati. La gestione della sicurezza è immediata e non richiede allo sviluppatore conoscenze di sicurezza avanzate per sviluppare un sistema sicuro [8].

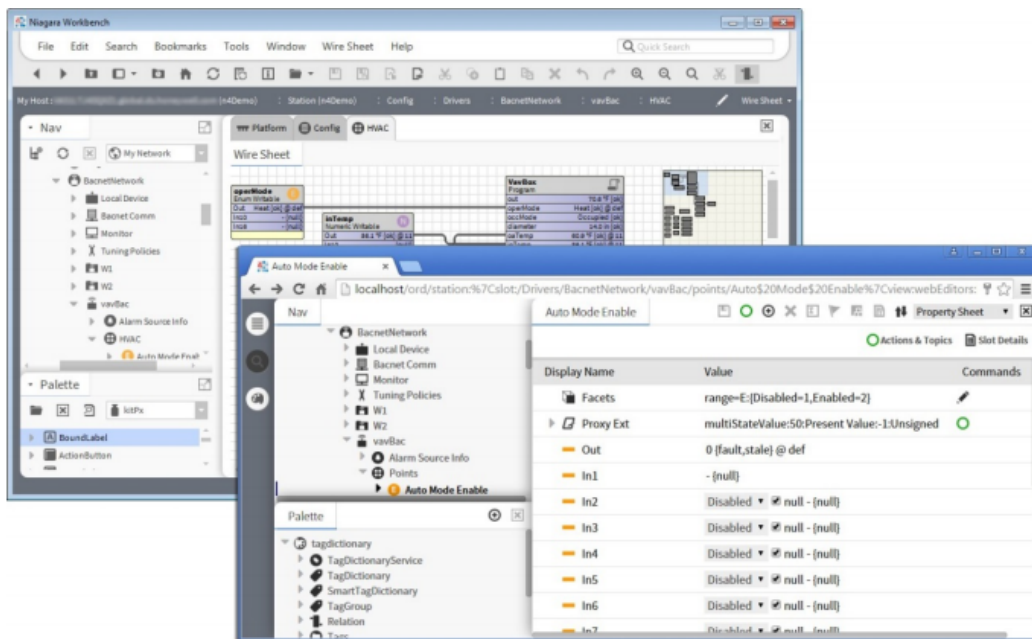


Figura 4.1: Workbench(sinistra) comparata con vista del web browser(destra)

4.3 Building blocks

Questa sezione descrive alcuni blocchi fondamentali che compongono il framework Niagara. Essi sono:

- Moduli: l'unità base del software che comprende Niagara;
- Componenti: i principali elementi costitutivi del framework;
- Presentazione XML (Px): un file formato XML che definisce in che modo Niagara visualizza le informazioni (grafica, testo, allarmi e così via) su diverse applicazioni come: Workbench, Web browser e dispositivi mobili;
- Stazioni: l'unità principale di elaborazione del server nell'architettura Niagara, denominabile come "applicazione". Definita da un singolo file *.bog*, viene avviata come una singola macchina virtuale sul pc host. Solitamente è presente una corrispondenza uno a uno tra le stazioni e gli host (Supervisor o JACE). Tuttavia è possibile eseguire due stazioni nella stessa macchina se esse risultano essere mappate su porte IP differenti;

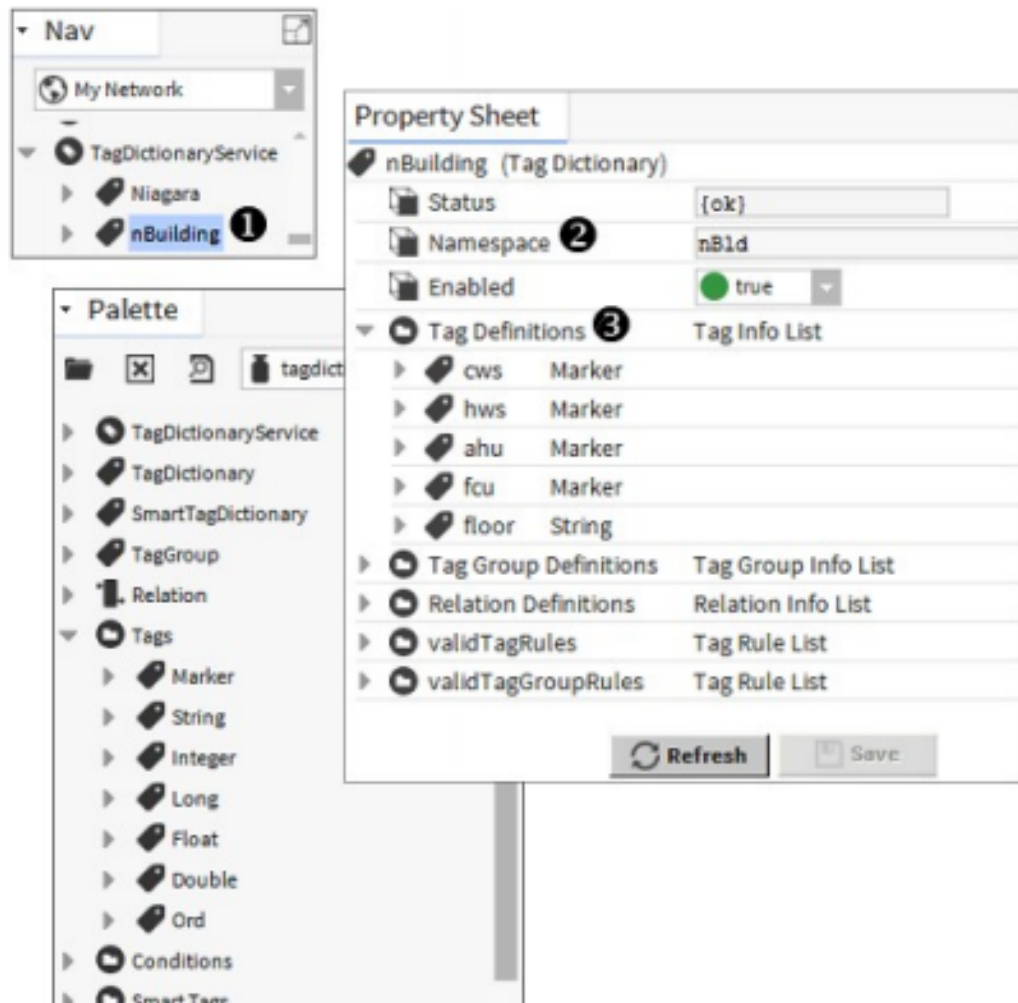


Figura 4.2: Dizionario tag personalizzato

- ORD(Object Resolution Descriptor): è il sistema di identificazione universale Niagara ed è utilizzato in tutto il framework;
- Views: la "visualizzazione" di un componente, come la vista riguardante le Property Sheet, Wire Sheet, e così via [18];

4.3.1 Moduli

Le versioni principali di Niagara sono distribuite insieme ad una serie di moduli di rilascio (con estensione .jar), ma possono essere distribuite come revisioni indipendenti all'interno di una determinata release. La figura seguente elenca una lista parziale dei moduli.

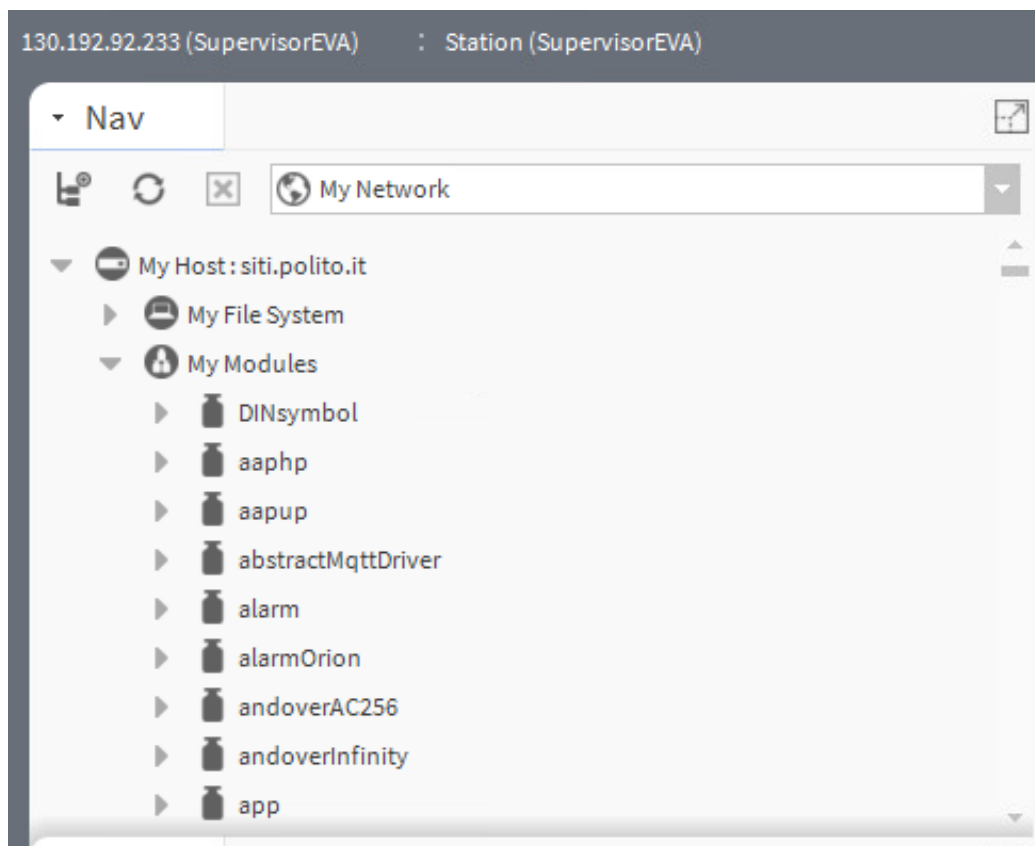
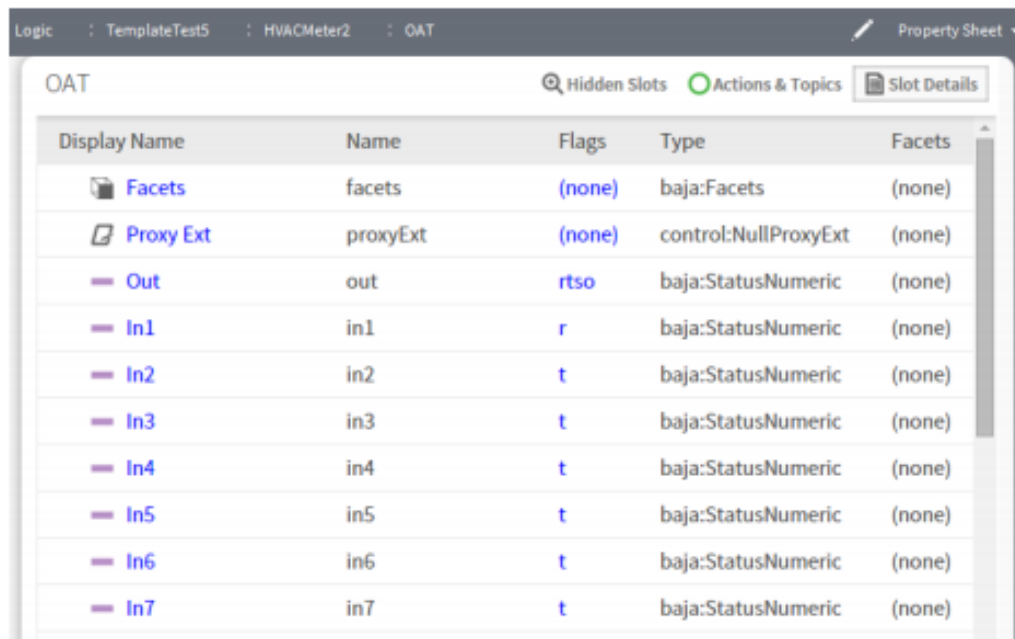


Figura 4.3: Lista di moduli

Tutti i moduli sono composti da un singolo file Java Archive (.jar) conforme alla compressione PKZIP. Questo contiene un file manifest XML, dove al suo interno ogni modulo dichiara le dipendenze da altri moduli e versioni.

4.3.2 Componenti

Come descritto nel paragrafo 3.3, i componenti offrono diversi vantaggi per lo sviluppo di un'applicazione. Essi differiscono dai moduli, in quanto comprendono un'implementazione del framework, mentre i moduli comprendono il software stesso. Come mostrato in figura 4.4, i componenti sono definiti come raccolta di slot. E' possibile vedere tutti gli slot che compongono un componente tramite la vista Property Sheet o la vista AX Slot Sheet.



Display Name	Name	Flags	Type	Facets
Facets	facets	(none)	baja:Facets	(none)
Proxy Ext	proxyExt	(none)	control:NullProxyExt	(none)
Out	out	rtso	baja:StatusNumeric	(none)
In1	in1	r	baja:StatusNumeric	(none)
In2	in2	t	baja:StatusNumeric	(none)
In3	in3	t	baja:StatusNumeric	(none)
In4	in4	t	baja:StatusNumeric	(none)
In5	in5	t	baja:StatusNumeric	(none)
In6	in6	t	baja:StatusNumeric	(none)
In7	in7	t	baja:StatusNumeric	(none)

Figura 4.4: Vista slot di un singolo componente

4.3.3 Presentazione XML

Un file XML è utilizzato per definire una presentazione Niagara. Questo file è chiamato "px", usato per descrivere l'architettura di presentazione. Presentazione è un termine che descrive come Niagara visualizza informazioni (testi, grafici, allarmi, scheduling, e così via). La figura 4.5 mostra un esempio di tale file all'interno dell'editor presente all'interno del Workbench, visualizzato tramite Px Viewer.

L'architettura di presentazione è basata sui seguenti principi di design:

- Il component model è il cuore della filosofia di ogni sottosistema in Niagara. Il framework si propone di costruire basandosi su tale modello. L'architettura di presentazione non fa eccezione. Il component model è utilizzato per offrire una rappresentazione normalizzata della presentazione, il "document object model" (DOM) della presentazione Niagara è sempre un albero "BComponent".
- Le visualizzazioni includono un approccio unificato per renderizzare la grafica, testo e controlli di input tutti all'interno di un singolo component model. Ciò permette a tutte le visualizzazioni di condividere un unico formato di file, proprio come il layout, rendering e le rispettive API di input.

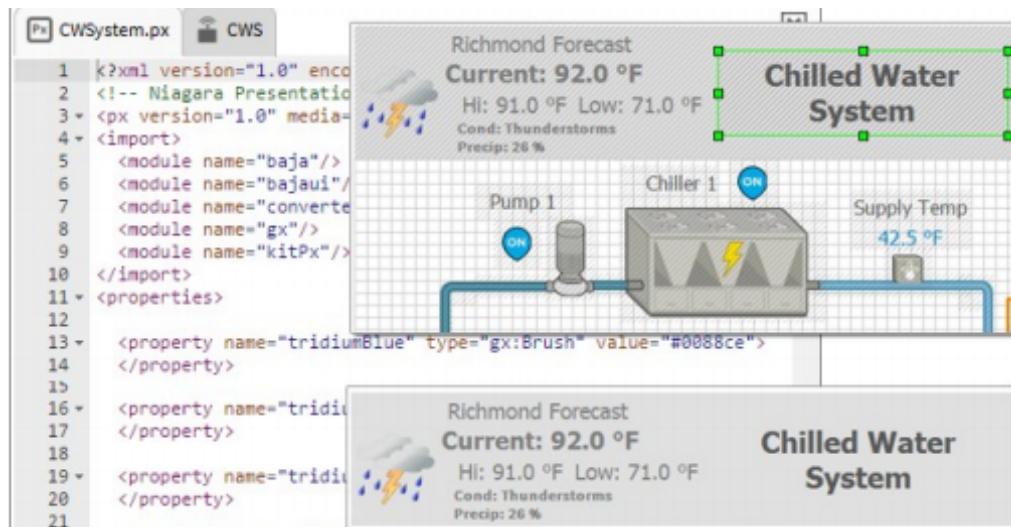


Figura 4.5: File Px all'interno del Px Editor

- L'obiettivo di design di una pagina Px è creare una singola vista che può essere visualizzata su più applicazioni. Ad esempio, dato un file Px, esso può essere automaticamente renderizzato all'interno del Workbench, o come una semplice pagina HTML da browser web, oppure come un file PDF. Tutte le rappresentazione sopracitate sono memorizzate come file Px all'interno di un component model normalizzato.

4.3.4 Stazioni

Una stazione è l'unità principale di elaborazione dei server nell'architettura Niagara.

- Il database di una stazione è definito da un singolo file .bog, ad esempio: *file:!stations/name/config.bog*
- Le stazioni sono avviate a partire dal rispettivo *config.bog* tramite un singolo processo o macchina virtuale, all'interno dell'host.
- Solitamente è presente una corrispondenza one-to-one tra le stazioni e le macchine host (Supervisor o JACE). Tuttavia è possibile avviare due stazioni dallo stesso host nel caso in cui siano mappate su porte IP differenti.

Una stazione esegue dei componenti del framework Niagara e ne permette l'accesso da remoto, tramite web browser, per visualizzare e controllare tali componenti. La parte principale di una stazione include

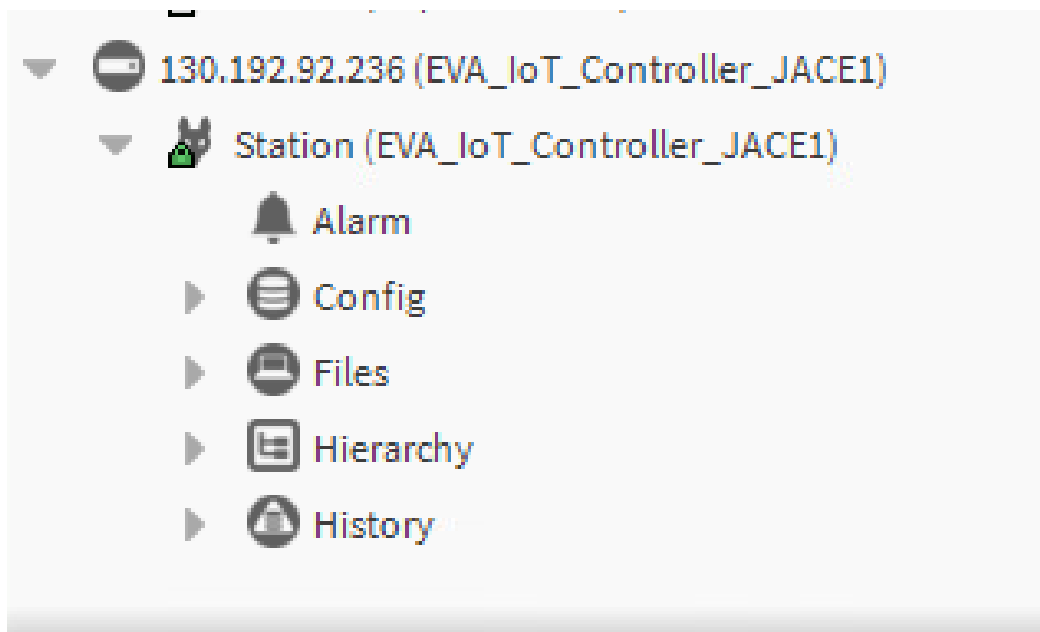


Figura 4.6: Navtree di una stazione

servizi e componenti. Essa rappresenta la combinazione tra web server, database e motore di controllo. Come già detto, essa è in esecuzione all'interno di un PC Web Supervisor o JACE controller.

4.3.5 ORDs

Un ORD è definito come un "Object Resolution Descriptor". All'interno di Niagara esso rappresenta l'identificativo universale di sistema ed è utilizzato all'interno di tutto il framework. L'ORD standardizza e unifica l'accesso a tutte le informazioni. E' stato pensato per combinare diversi nominativi di sistema all'interno di una singola stringa. Esso ha inoltre il vantaggio di essere *parsable* da un host di una API pubblica.

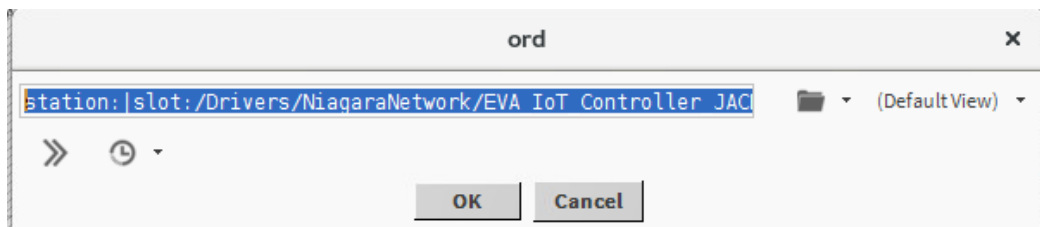


Figura 4.7: ORD e lcoatore grafico di sistema

Un ORD è presente in una o più query. Queste dispongono di uno schema che identifica come eseguire il parse e risolvere un oggetto. Esso può essere relativo o assoluto. La figura 4.7, rappresenta un esempio del primo caso. Un ORD assoluto invece, di solito mostra il formato *host—session—space*

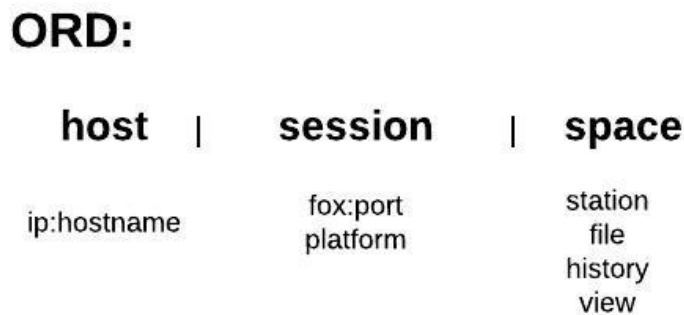


Figura 4.8: Struttura tipica ORD assoluto

- host: identifica una macchina, da un indirizzo IP, come ad esempio "ip:hostname";
- session: identifica il protocollo utilizzato per comunicare con l'host;
- spazio: identifica un particolare tipo di oggetto. Gli spazi comuni sono "module.", "station.", "view.", "spy.", o "history.";

4.4 Integrazione dei sistemi di controllo

Utilizzando il framework Niagara, l'integrazione dei sistemi di controllo permette di eseguire:

- Collegamento dei dispositivi su un supporto di comunicazione comune;
- Modellizzazione di questi in software;
- Programmazione delle applicazioni per utilizzare le informazioni di quei dispositivi;

Prima di poter utilizzare un dispositivo, come un sensore di temperatura o un attuatore, è necessario estrarre le informazioni da tali dispositivi ed inserirli nel software di sistema. Niagara modella tali dispositivi e i loro tipi di dati in software tramite l'utilizzo del modello ad oggetti comune. Questo comporta una semplificazione dei tipi di dati dei dispositivi per renderli più facilmente manipolabili e controllabili tramite software. Il modello ad oggetti di Niagara è simile ad un linguaggio di programmazione, in quanto ci sono alcuni concetti chiave che vengono utilizzati; il vero vantaggio però risiede nelle librerie riutilizzabili delle applicazioni e delle collezioni di oggetti che sono disponibili [8]. Come è evidenziato nella figura 4.9, il *common object model* di Niagara consente al framework di:

- fornire comunicazioni bidirezionali sicure tra dispositivi e Internet;
- inviare informazioni real-time del dispositivo tramite Internet;
- dispositivi di controllo in real-time su Internet;

Gran parte del software Niagara è scritto in Java, il che significa che questo è indipendente dalla piattaforma. Java viene compilato una sola volta ed è possibile eseguirlo su qualsiasi piattaforma grazie ad un livello software esistente tra la macchina e la Java Virtual Machine. Il software Niagara funziona sui controller embedded JACE, citati nel capitolo precedente, utilizzando un sistema operativo QNX ed una IBM J9 Java Virtual Machine(JVM). Niagara utilizza la JVM come ambiente di runtime comune attraverso vari sistemi operativi e piattaforme hardware. La VM è responsabile della definizione di come il software funziona con un determinato set di hardware, come parla alla porta di comunicazione e come interagisce con il sistema operativo [18].

4.5 Networking e protocolli Internet

Niagara è stato progettato per supporre che non ci sarà mai un unico protocollo di rete "standard", un'architettura distribuita o un bus di campo. L'obiettivo è quello di integrarsi in modo pulito con tutte le reti e i protocolli. La struttura standardizza cosa c'è dentro la scatola, ma non come essa comunica con l'esterno. La suite software del framework implementa un adattamento efficiente del modello software con le tecnologie Internet correnti, per fornire un'interoperabilità tra una vasta gamma di prodotti di automazione. Il modello ad oggetti può essere utilizzato per integrare diversi dispositivi fisici, controller e applicazioni di controllo, inclusi oggetti BACnet e punti

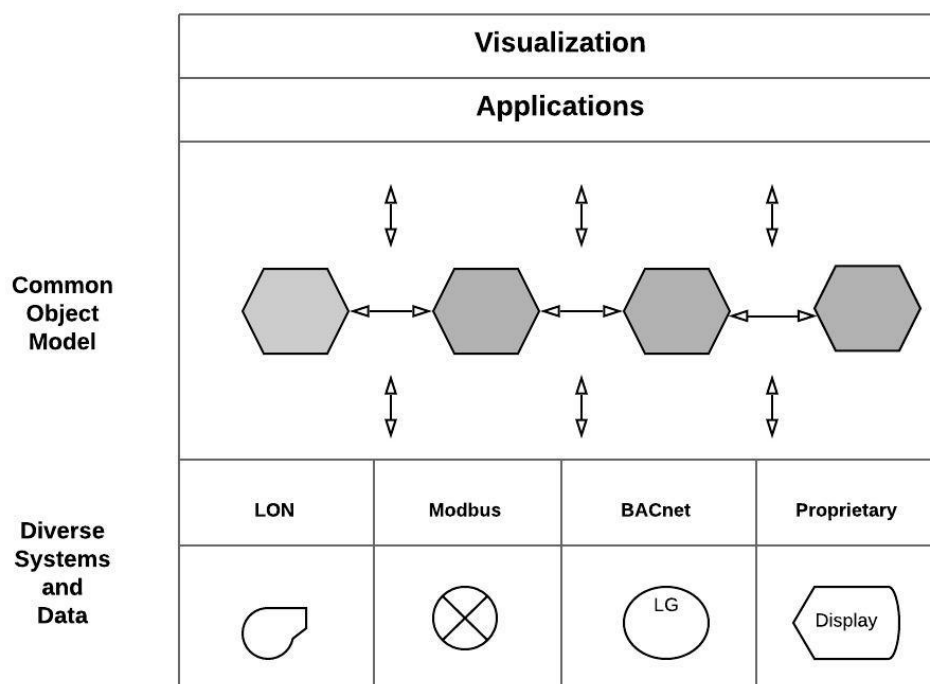


Figura 4.9: Niagara common object model

di controllo legacy. Gli standard del software di livello aziendale includono il protocollo TCP/IP, XML, HTTPS, TLS e altri. Tali standard offrono le basi su cui costruire soluzioni che permettano di condividere le informazioni tra il sistema di controllo e il sistema informativo aziendale. La maggior parte delle funzionalità del framework sono progettate per il doppio utilizzo (programmatori e non). Questa caratteristica è resa possibile tramite un set di API Java a cui gli sviluppatori possono accedere scrivendo del codice Java [18].

4.6 Baja

Il framework core *Baja* (Building Automation Java Architecture) è stato progettato per essere pubblicato come uno standard aperto. Tale standard è stato sviluppato attraverso il Java Community Process. Baja rappresenta una specifica aperta, e Niagara è un'implementazione di tale specifica. Baja non è un insieme di software, ma piuttosto un semplice set di documentazione. La specifica Baja includerà:

- Standard tramite cui i moduli Baja sono confezionati;
- schema XML per il modello del componente;
- Il modello del componente e le sue API;
- Componenti e API del database storico;
- Componenti e API degli allarmi;
- Componenti della logica di controllo e API;
- Componenti di scheduling e API;

Baja è in continua evoluzione e nel tempo, molte altre specifiche per le funzionalità verranno aggiunte al suo interno.

4.7 APIs

Le API (Appliction Programming Interface) definiscono in che modo lo sviluppatore accede alle funzionalità di un software con il framework Niagara. Il Workbench, citato nel paragrafo 3.2, è esso stesso un API Niagara. Utilizzandolo, è possibile creare e modificare la logica di controllo del dispositivo embedded. Molte altre caratteristiche presenti nel framework sono esposte tramite un set di API. Nel mondo Java, le API sono raggruppate in pacchetti, distinti per i nomi dei domini DNS. Il software sviluppato tramite Il Java Community Process, è di solito composto da pacchetti che iniziano con "java" o "javax". Le API sviluppate per Baja sono raggruppate in javax.baja. Queste fanno parte delle specifiche Baja e potrebbero essere implementate da fornitori diversi. Esse garantiscono la neutralità del fornitore e compatibilità. Tutto il software proprietario, al di fuori delle specifiche Baja, è raggruppato sotto i pacchetti di com.tridium. Quest'ultimi contengono codice specifico su come Niagara implementa le API Baja. Se le API di com.tridium sono documentate pubblicamente, allora Tridium incoraggia sviluppatori ad usarle, ma non garantisce la compatibilità con le versioni precedenti. Le API com.tridium non documentate non dovrebbero mai essere utilizzate dagli sviluppatori.

Capitolo 5

Data and Control Model

In Niagara qualsiasi stazione (Supervisor o controller), contiene dei dati in real-time normalizzati e memorizzati all'interno del rispettivo database come *punti*, ovvero uno speciale gruppo di componenti. Ogni punto rappresenta un oggetto di dato.

Ognuno di essi variano per tipo, input e output. I principali ed i più utilizzati in fase di sviluppo sono stati i seguenti:

- control: gruppo di component di tipo base, come numeric, boolean, enum, string;
- logic: gruppo di component che permettono di eseguire operazione logiche tra i vari component (AND, OR, XOR, NOT, ecc.);
- math: gruppo di component per effettuare operazione matematiche tra component e sui loro input e output (ADD, SUBTRACT, DIVIDE, MULTIPLY, SINE, ecc.);
- timer: gruppo di component per operazioni con durata limitata (ONE SHOT, BOOLEAN DELAY);
- select: gruppo di component per operazioni di selezione basate sugli output dei component (BOOELAN SWTICH, SWITCH, SELECT, ecc.);
- constant: gruppo di component per definire costanti di input sui component (NUMERIC CONSTANT, BOOLEAN CONSTANT, ecc.);

5.1 History

In Niagara, un data log viene definito come history. Queste sono collezioni ordinate di record per data e ora. Una singola "history" è una raccolta di valori di dati specifici di un punto specifico all'interno di una determinata stazione locale o remota. Le history sono raggruppate per la loro stazione di origine. Esempi di history utilizzate nello sviluppo sono:

- Intervalli: I punti di controllo che sono configurati per memorizzare dati, hanno delle proprietà tramite le quali possono essere memorizzati in intervalli di tempo specifici; ad esempio, in un acquedotto può essere registrata la variazione di temperatura ogni 15 minuti. In questo caso, la history è memorizzata con il rispettivo timestamp. I dati raccolti possono essere visualizzati in diverse viste, inclusi grafici e tabelle.
- Variazione di valore: i punti di controllo sono configurati in modo da registrare un record ogni volta che viene registrato un cambiamento di valore in quello specifico punto di controllo. Un cambiamento di valore può essere inteso come un cambiamento di stato (ON, OFF), una modifica di un'impostazione (LOW, MEDIUM, HIGH), oppure una qualsiasi variazione, ad esempio la temperatura, che varia di 0,05 gradi.

Tali parametri sono impostati da un tecnico di sistema che "configura" le proprietà delle history all'interno del punto di controllo [8].

5.2 Wire Sheet Object Management

Ogni JACE controller deve necessariamente avere una logica di controllo, definita e progettata dallo sviluppatore, affinché possa essere eseguito l'algoritmo di funzionamento da parte del device. Una volta effettuato l'accesso nella stazione, l'utente dispone dei permessi necessari per creare una cartella dentro la sezione di configurazione del dispositivo, denominata *Config*. Una volta creata, lo sviluppatore avrà la possibilità di istanziare i primi *point component*, che rappresentano una parte di dato. Per mezzo dell'utilizzo di una vista del Workbench, chiamata Wire Sheet, come mostrato in figura 5.1, è possibile mettere in relazione tra loro i vari component.

Per l'utente è abbastanza semplice creare i vari link tra i component della logica di controllo. E' inoltre possibile scegliere a quale input del component di destinazione il component sorgente sarà collegato. Come mostrato nella figura 5.2, quasi tutti dispongono di 16 input, con priorità decrescente.

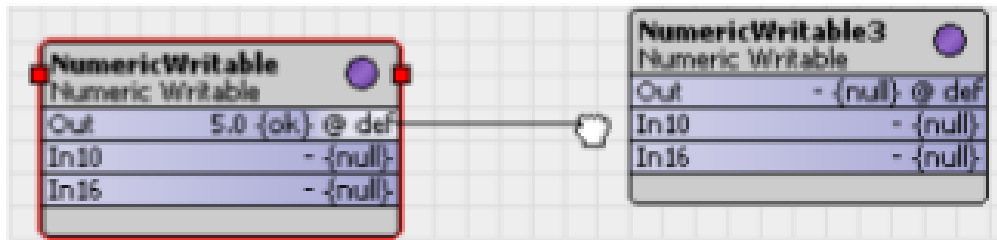


Figura 5.1: Basic object linking

Questa caratteristica è stata un aspetto molto importante in sede di progettazione, dato che è stato possibile eseguire l'override di un determinato valore al verificarsi di un'anomalia, un allarme o un comando eseguito dall'utente, andando dunque a condizionare l'output di uscita di un component critico, come quello che può rappresentare un attuatore o sensore di misurazione.

5.3 Logica di controllo

Come descritto nel capitolo 3, il sistema fisico è stato suddiviso logicamente nelle seguenti sezioni:

- *Environment*: rappresenta la fonte naturale contenente l'acqua proveniente da qualsiasi fenomeno ambientale;
- *Lake*: rappresenta la fonte dove, l'acqua proveniente dall'environment, viene raccolta;
- *Purifier*: rappresenta il contenitore dove l'acqua rimane in attesa del trattamento di purificazione;
- *Clean Water*: è il serbatoio dove l'acqua potabile è conservata dopo il trattamento di depurazione;
- *House*: rappresenta l'utilizzo dell'acqua da parte del consumatore, il quale ha accesso al sistema di distribuzione;

Per quanto riguarda la progettazione e lo sviluppo dell'algoritmo di gestione e monitoraggio da eseguire all'interno del JACE controller, è stato rispettata la suddivisione sopracitata. Come mostrato in figura 5.3, ogni blocco presente sullo sheet rappresenta un container avente al suo interno gli il gruppo di component in relazione tra loro, che vanno a formare l'effettivo

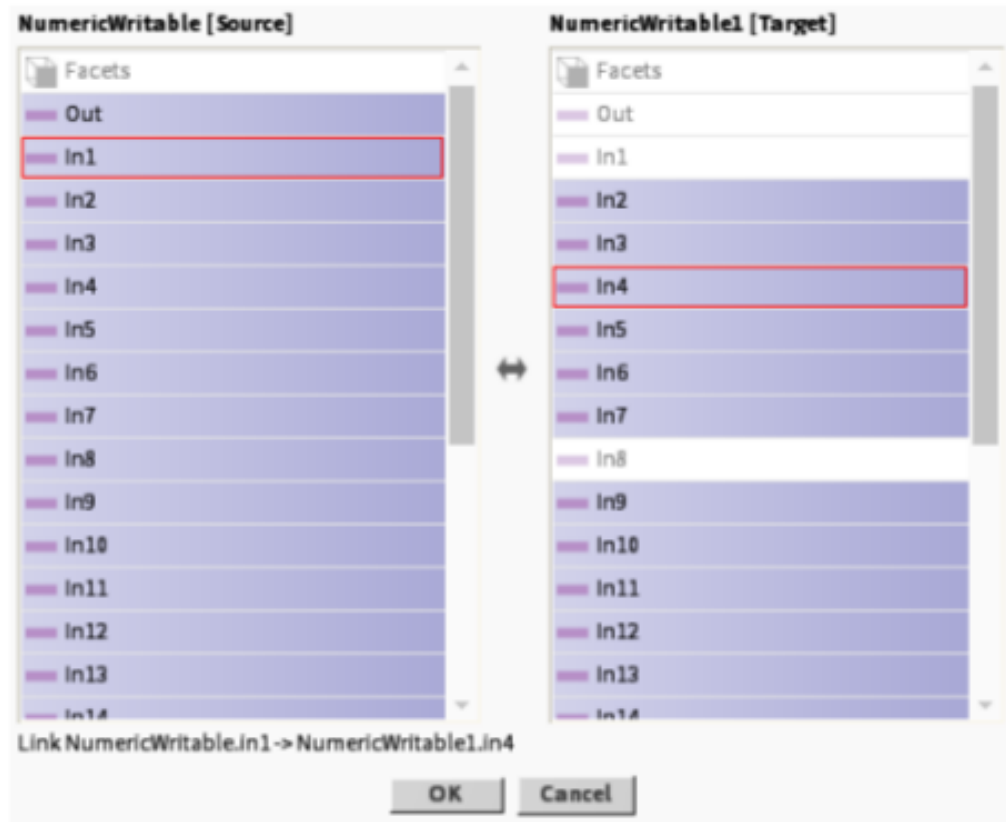


Figura 5.2: Basic object linking

algoritmo di controllo di quel determinato settore. Oltre ai settori elencati precedentemente, sono state modellate 3 logiche aggiuntive per una gestione più complessa degli attuatori dell'acquedotto (EnvLakePump, EnvPurPump) e un monitoraggio più preciso sullo stato attuale dell'interno sistema di controllo (System). I blocchi in alto a destra (CurrentTime, CurrentData), permettono al controller di condividere con il Supervisor Controller il proprio timestamp di sistema.

5.3.1 Lake

La figura 5.4, mostra il gruppo di componenti linkati fra loro, che vanno a comporre l'algoritmo di controllo del settore dell'acquedotto riguardante il Lake. Particolare attenzione è stata posta ai sensori e all'attuatore presenti sul campo. In particolare:

- sensore di misura del flusso di acqua;

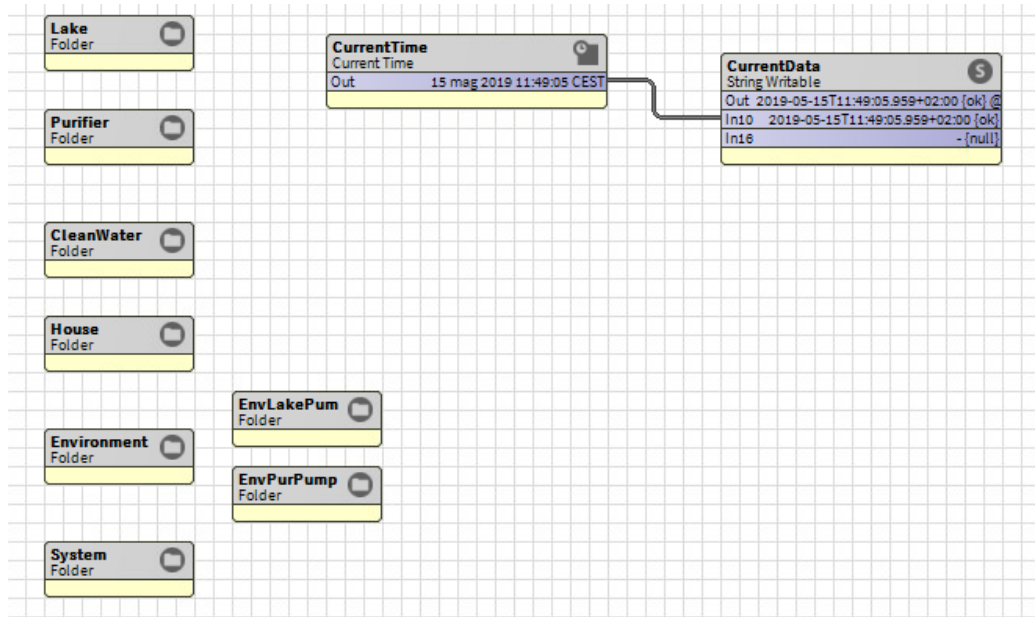


Figura 5.3: Struttura logica di controllo

- sensore di misura del livello dell'acqua all'interno del serbaotio;
- pompa centrifuga;

Il serbatoio del "Lake" è il secondo serbatoio del modello fisico, con un'altezza che si attesta a circa 30cm. Le soglie sono state impostate tramite dei component di tipo constant, aventi un valore fisso ma modificabile dall'esterno. Questi tipi di component fungono da entrypoint dell'algoritmo, affinché possano essere cambiati i parametri e le soglie critiche di quest'ultimo. Come descritto nel capitolo 3, è presente una Raspberry Pi che legge i dati dai sensori ed invia queste informazioni direttamente al JACE controller.

Nel caso in cui, all'interno del serbatoio del "Purifier", siano presenti almeno 2 cm di spazio libero, saranno inviati alla pompa dei comandi di switch-on e switch-off.

All'interno dei component, che rappresentano il valore attuale dei sensori e attuatori, sono state aggiunte due tipi di estensioni:

- alarmExtension: da origine ad una specifica classe di allarme che viene notificata all'utente, nell'evenienza in cui, ad esempio, lo spazio disponibile nel serbatoio è in esaurimento, o nel caso in cui si registrassero dei valori anomali in lettura;

- historyExtension: estensioni orientate allo storage dei dati in ingresso, che viene eseguito costantemente o ad una variazione sensibile del valore in ingresso;

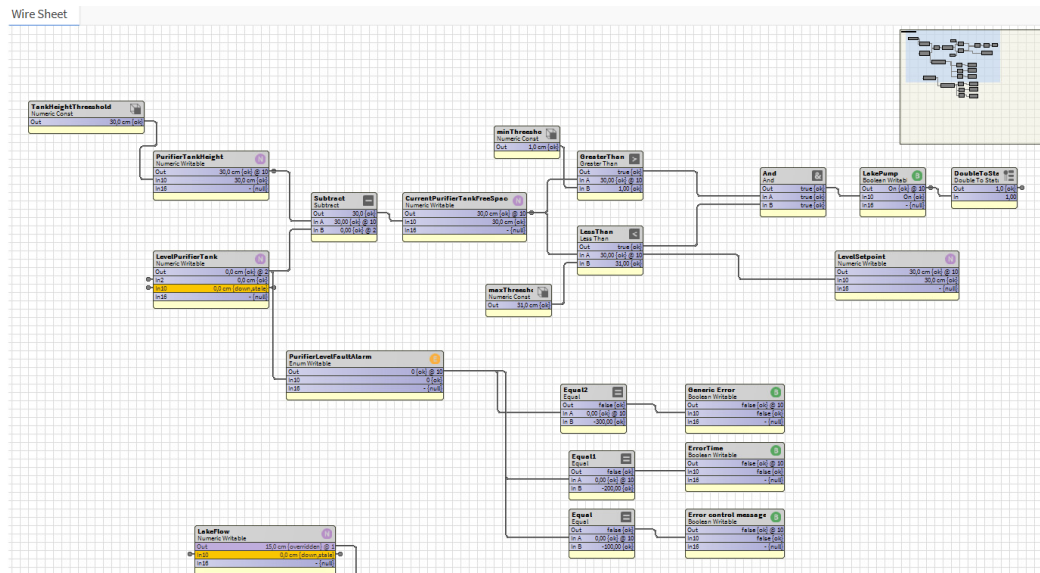


Figura 5.4: Modellazione logica de Lake

5.3.2 Purifier

La modellazione logica riguardante il "Purifier", presenta quasi lo stesso schema del Lake. In aggiunta al sensore di flusso, ultrasuoni ed alla pompa, è presente il sensore di temperatura. Come i precedenti, questo è presente all'interno del serbatoio, i cui dati sono letti ed inviati dalla board Raspberry Pi al controller, assieme a tutti gli altri.

Il sensore di temperatura in caso di errore potrebbe registrare in lettura dei valori anomali (es. -100, -200, -300). In questo caso una specifica classe di allarme viene istanziata e notificata all'utente. Tale allarme "rientrerà" automaticamente, nel momento in cui i valori ritorneranno consistenti.

5.3.3 Clean Water

A differenza dei due precedenti modelli, come mostrato in figura 5.6, la pompa è sostituita da un'elettrovalvola, la cui apertura e chiusura è condizionata dal verificarsi di due specifiche condizioni:

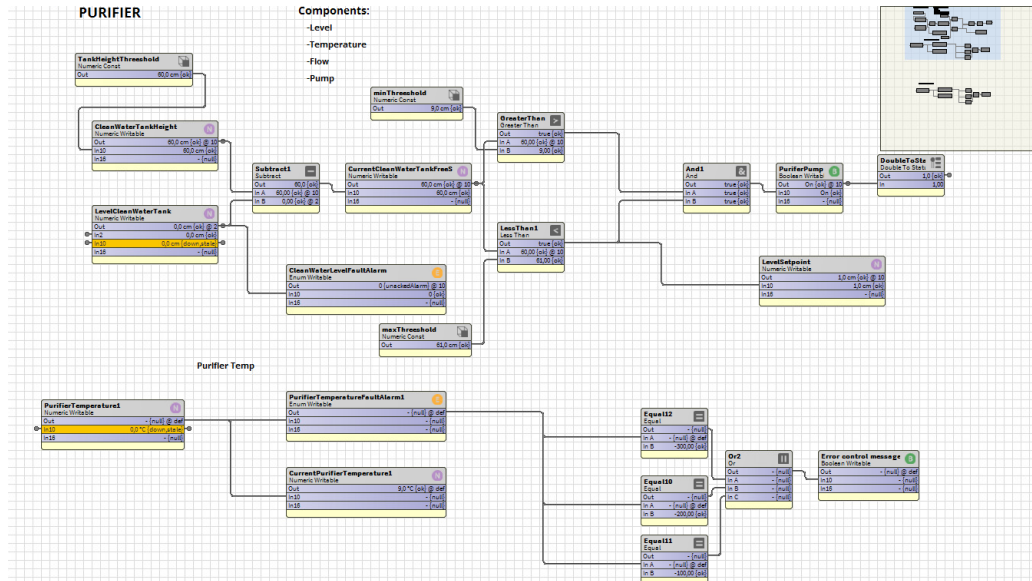


Figura 5.5: Modellazione logica del Purifier

- livello dell'acqua: sia presente una quantità sufficiente d'acqua da rilasciare verso il sistema idrico cittadino;
- RFID: l'utente è in grado di inviare un segnale tramite RFID. Questo rappresenta uno strumento di sicurezza e di conferma aggiuntiva per l'operazione di apertura della valvola. Tramite l'utilizzo di un component One Shot, lo stato di apertura sarà valido soltanto per 10 secondi prima di riprendere le operazioni di controllo sui valori in lettura.

In aggiunta al sensore di temperatura, è presente quello dedicato al rilevamento di umidità nell'aria. Anche tale sensore, in caso di valori anomali, darebbe origine alla classe di allarme dedicata; allarme che verrebbe immediatamente notificato all'utente.

5.3.4 Environment

Rispetto ai modelli precedenti, questo modello contiene la logica di controllo necessaria per simulare l'interazione del modello con l'ambiente esterno.

Nella simulazione proposta, l'utente con privilegi di Testbench user, potrà selezionare due tipi di sorgenti d'acqua:

- *Acque di superficie*: fiumi, laghi, acqua di transizione, corpo idrico artificiale, ecc;



- *Acque sotterranee*: sorgenti, pozzi, falde, ecc.

e dare il via alla simulazione, impostando come true il component booleano denominato *StartSimulation*. E' stato inoltre aggiunto un component di tipo "constant" affinché potesse essere impostato il valore di *Basic Time Unit* (BTU) che specifica con quale frequenza i dati sono generati dai component coinvolti nella simulazione e nella generazione di output.

Avviata la simulazione, è generato un flusso d'acqua proveniente dall'ambiente esterno, verso l'opera di presa dell'acquedotto. La differenza tra i tipi di sorgente, in sede di modellazione, ha riguardato la generazione o meno di valori di temperatura e umidità, per i motivi spiegati nel Capitolo 3. L'opera di presa, Lake, sarà in grado di accumulare una quantità d'acqua minore e uguale alla soglia riguardante l'altezza del suo serbatoio, impostata all'interno del suo modello logico. Nell'istante in cui la pompa del Lake viene attivata, secondo i criteri specificati nel paragrafo 4.3.1, una certa quantità d'acqua, verificato che il serbatoio del Purifier, disponga di spazio sufficiente, viene sottratta al Lake e riversata al suo interno. Secondo criteri simili, la pompa del Purifier, dopo la sua attivazione, trasferirà una quantità d'acqua minore uguale allo spazio libero disponibile all'interno del serbatoio di Clean Water.

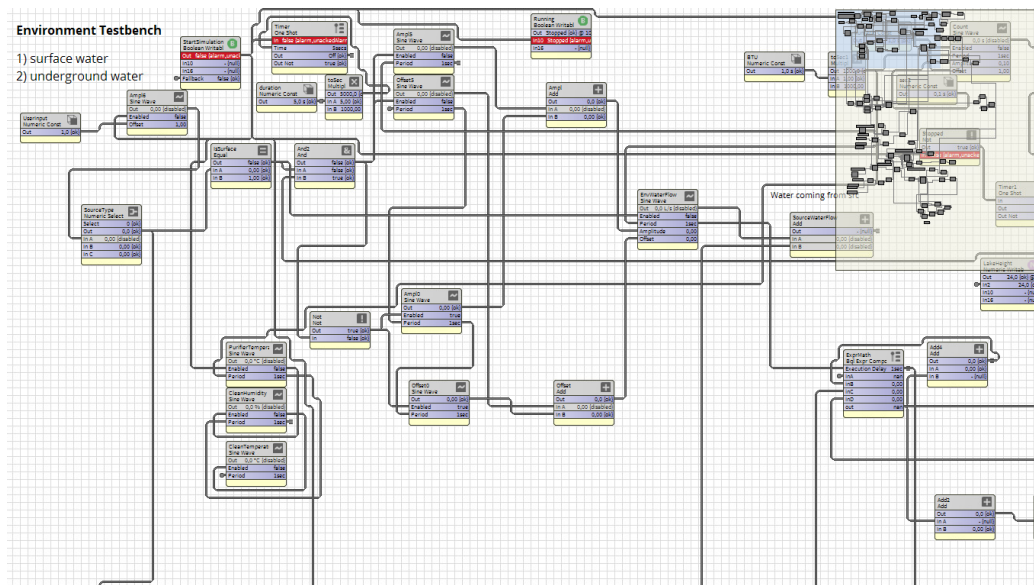


Figura 5.7: Modellazione logica Environment

Modello migliorato delle pompe

I modelli logici *EnvLakePump* e *EnvPurPump*, specificano i criteri di modulazione della potenza delle pompe presenti nel modello fisico. La figura 5.8, mostra un entrypoint, chiamato *MaximumLakePumpFlow*, tramite il quale l'utente o l'operatore può impostare la massima potenza, rappresentata in L/s, di una determinata pompa.

L'output del component *SetLakePumpPower*, è stato generato e calcolato grazie alla logica di controllo dedicata alla pompa, all'interno del modello denominato *EnvLakePump*. La figura ??, mostra una panoramica delle operazioni effettuate nella determinazione della potenza dell'attuatore. Si è tenuto conto dello spazio libero presente nel serbatoio sorgente, e conseguentemente, una politica di attuazione diversa per ogni suo valore. L'idea è quella di non riempire il serbatoio sorgente oltre il suo 90% di spazio totale, affinché possa essere mantenuto un certo margine di sicurezza. Superata questa soglia, la potenza e il numero di litri trasferiti verso la destinazione sarà di conseguenza in aumento, compatibilmente con lo spazio disponibile all'interno del serbatoio di destinazione.

5.3.5 House

Per simulare l'interazione dell'utente consumatore con il modello, sono state introdotte due modalità: un rubinetto, controllato direttamente dall'utente,

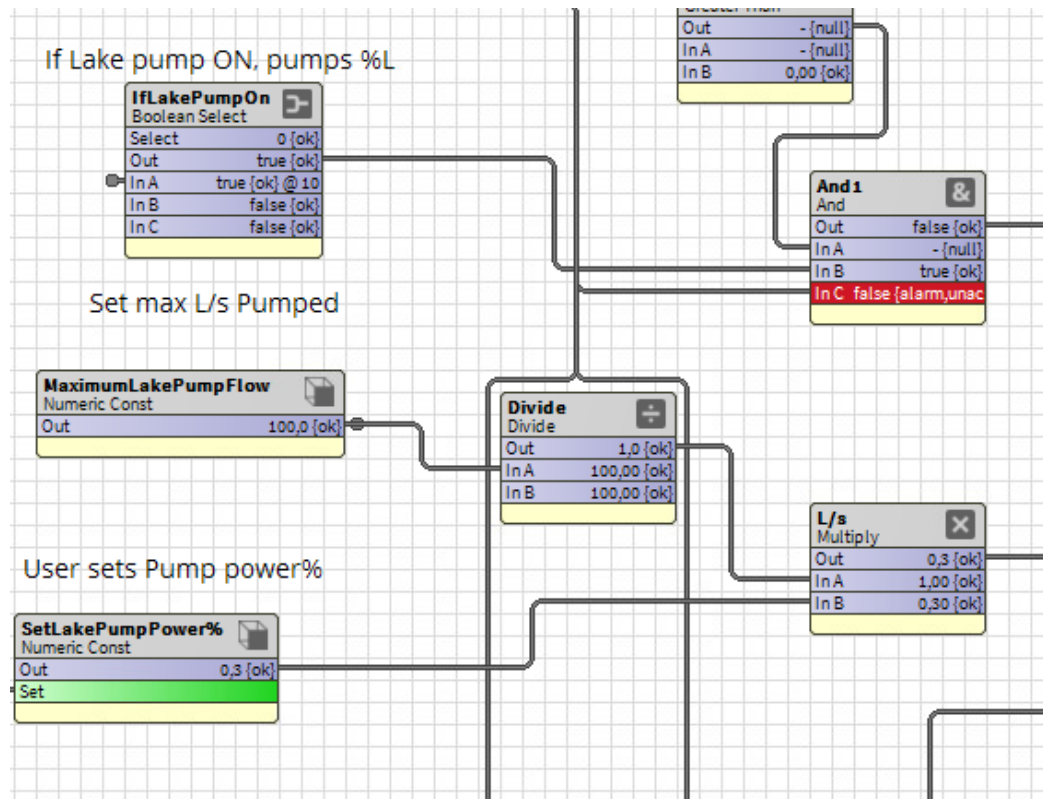


Figura 5.8: Set della potenza di una pompa

ed una elettrovalvola. Entrambe, come mostrate in figura 5.9, "consumano" una certa quantità d'acqua dal serbatoio di acqua potabile (Clean Water). In sede di simulazione, l'operatore può settare una percentuale d'acqua consumata ogni n secondi. Come per i precedenti modelli, si tiene conto del livello di acqua contenuta all'interno del Clean Water, affinché sia maggiore di una determinata soglia di sicurezza.

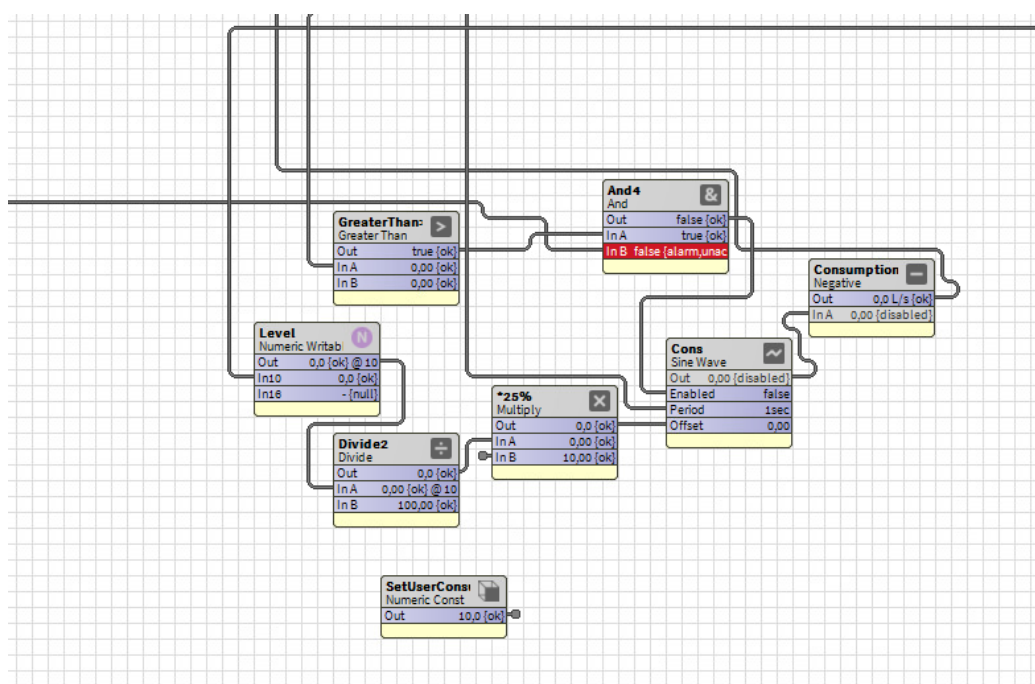


Figura 5.9: Modello logico del consumatore

Capitolo 6

Web Application

6.1 Architettura

L'architettura Niagara, come descritto nel Capitolo 4, consente l'automazione, la supervisione ed il controllo di dispositivi intelligenti di campo attraverso una rete dati. Con tale definizione, ci si riferisce ai tradizionali componenti dei sistemi di controllo (attuatori, sensori e moduli di acquisizione a comando) con CPU integrata e protocollo di comunicazione. In particolare questa architettura consente di operare indistintamente con i protocolli di mercato più diffusi (BACnet, EIB, Modbus, LonWorks, ecc.), oltre che con i protocolli proprietari delle aziende di mercato più affermate.

E' così possibile procedere alla configurazione, al controllo, fino alla rappresentazione grafica di un sistema comunque complesso ottimizzando la scelta dei dispositivi di campo e comunque integrando quelli eventualmente già presenti nel caso si debba anche aggiungere o modificare parti degli impianti automatici. Tutto questo attraverso la rete dati con protocollo TCP/IP e dunque ottimizzando l'utilizzo delle reti LAN, sfruttando così Internet per la connessione da remoto.

Un'altro vantaggio è quello di potersi integrare con i DB gestionali eventualmente presenti: di conseguenza, permettendo l'utilizzo dei dati di campo nella rete gestionale.

Il sistema proposto può essere suddiviso nelle seguenti parti:

- Infrastruttura di rete
- Interfacciamento protocolli
- Software di sviluppo e applicativo

6.1.1 Infrastruttura di rete

La realizzazione del sistema implica la realizzazione o la disponibilità di un'infrastruttura di rete che rappresenti un valore aggiunto, in quanto punto di partenza per la crescita del sistema e base per futuri altri sistemi. Non è stato necessario considerare nessun requisito alla rete dati, in quanto l'utilizzo del JACE controller, consente di trovare la soluzione adeguata secondo il numero di nodi presenti in rete e secondo le architetture disponibili.

Il JACE controller costituisce il supporto hardware su cui viene caricata la piattaforma software Niagara. Questo, integra i dispositivi di campo con la rete dati sia a livello fisico che software, supportando i relativi driver di comunicazione con la piattaforma Java. Al suo interno viene caricata la stazione software, che implementa la logica del sistema per la gestione complessiva dei dispositivi.

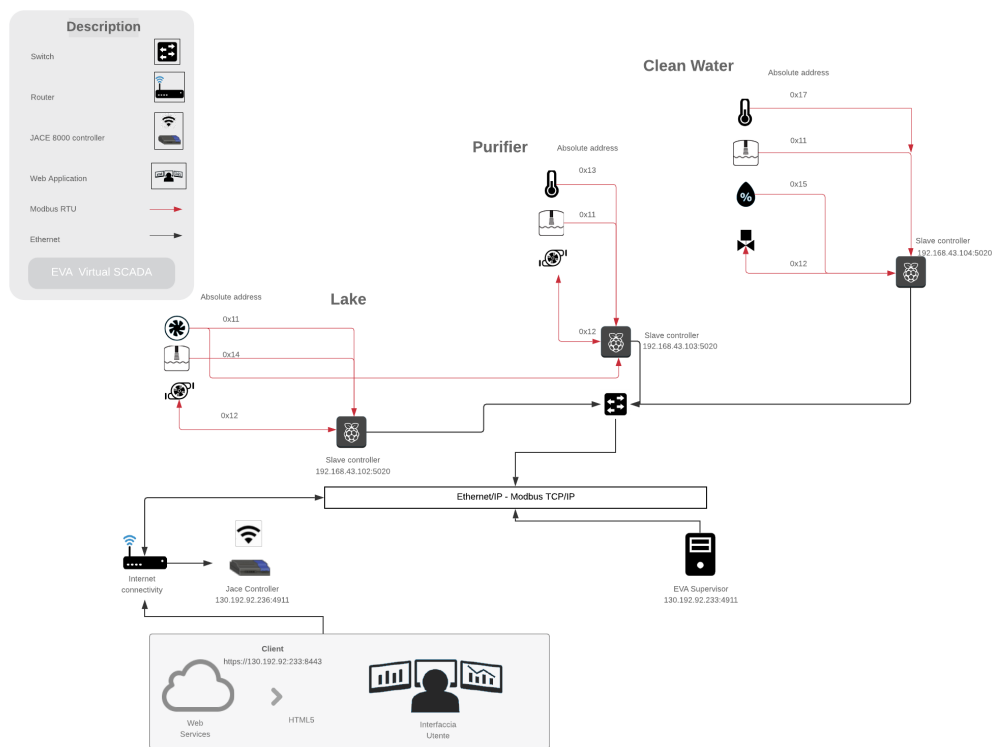


Figura 6.1: Infrastruttura di rete

La figura 6.1 mostra una panoramica sull'infrastruttura di rete del sistema. Tramite e per mezzo di essa si svolgono le seguenti funzioni:

- Gestione dei dispositivi di campo e acquisizione dei relativi dati;

- Comunicazione punto-punto tra gli stessi dispositivi;
- Gestione delle funzioni di controllo della rete di campo locale:
 - Gestione eventi;
 - Logica di controllo;
 - Web Service;
 - Data-logging;
- Server di gestione della rete e per la gestione di allarmi, scheduling, dati, viste, etc.;

Gli elementi principali di tale infrastruttura sono:

- JACE 8000: connessa sia al bus di campo che all'altra LAN;
- Server: su cui è installata una macchina virtuale dove al suo interno è in esecuzione il Web Supervisor per l'archiviazione dei log dei valori misurati dai sensori, dei log di allarme e la gestione delle interfacce web;
- Dorsale: costituita dalla rete LAN su TCP/IP;
- Uno o più PC (opzionali): con web browser, connessi alla rete LAN, per monitorare e visualizzare in tempo reale lo stato del sistema;
- Board Raspberry Pi3: svolgono il ruolo di slave, all'interno di una topologia di rete a stella, il cui centro è occupato dal JACE controller. Tutte le board si trovano ai vertici di tale topologia; queste sono connesse tra loro tramite WIFI e condividono dati e comandi con il master (JACE controller) tramite lo standard Modbus; Tutti gli script delle board sono scritti in Python, includendo un modulo chiamato *PyModbus*, in modo da poter incapsulare le informazioni di sensori e attuatori in pacchetti Modbus;

Le Raspberry Pi sono collegate fisicamente ai sensori e attuatori di campo. Queste collezionano i dati provenienti da quest'ultimi per poi inviarli al controller. Tramite il Workbench Niagara, all'interno della stazione in esecuzione in quel momento, è possibile mappare e creare una corrispondenza tra i sensori e attuatori di campo con quelli "emulati" all'interno della logica di controllo del controller.

Come si nota dalla figura 6.2, è possibile istanziare un nuovo oggetto all'interno della *Modbus Network* a cui si interfaccia il controller. Ognuno di

Database						
Name	Exts	Status	Device Address	Ip Address	Port	Socket Status
 Lake_Tank_Device		{down}	3	192.168.43.102	5020	Open Failed
 Purifier_Tank_Device		{down}	3	192.168.43.103	5020	Open Failed
 CleanWater_Tank_Device		{down}	3	192.168.43.104	5020	Open Failed

Figura 6.2: Mapping delle Raspberry nel controller

essi rappresenta lo slave di campo, il quale rappresenta colui che detiene le informazioni lette da sensori e attuatori. Ad ognuno di essi viene assegnata una configurazione ip ed una porta, affinché possano essere raggiungibili dal controller. All'interno di questi oggetti, come mostrato nella figura 6.3, sono stati creati dei punti di controllo provenienti dalla logica di controllo del controller. Questi sono mappati sugli indirizzi dei registri Modbus TCP/IP con valore esadecimale, in modo da poter leggere i valori di input (provenienti dalla modellazione logica della stazione) e di output (provenienti dai sensori e attuatori di campo). Gli slave, pertanto, non dispongono di nessun potere decisionale; ogni decisione è presa dal controller (master) secondo la propria logica di controllo con input e output i valori acquisiti dai sensori e attuatori di campo.

Database		
Name	Out	Absolute Address
 Level	0,0 cm {down,stale}	hex:11
 Flow	0,0 L/s {down,stale}	hex:14
 Pump	0,0 {down,stale} @ 10	hex:12

Figura 6.3: Mapping dei punti di controllo con i sensori

Dorsale

La dorsale è un elemento fondamentale dell'architettura, necessaria la comunicazione tra le board e le macchine all'interno della LAN e per la connessione

ad Internet. In questo modo, senza ulteriori spese aggiuntive, sono stati sfruttati i vantaggi di una rete IP, prima fra tutte: la velocità di connessione. La rete a banda larga permette l'accesso, la visualizzazione ed il controllo del sistema da remoto.

EVA Supervisor

Il sistema proposto prevede l'utilizzo di un JACE controller, che deve condividere informazioni, archivarle ed avere pagine grafiche in comune, oppure inviare mail di allarme. In tal caso, si è reso necessario l'utilizzo di un Web Supervisor, che consente di realizzare tutte queste funzionalità in modo automatico e centralizzato. E' necessario però far notare che questo componente dell'architettura non è sempre presente, ma viene introdotto soltanto nei casi elencati nel paragrafo successivo, in cui la supervisione si rende necessaria per le funzioni di archivio e concentrazione dei dati.

Il Web Supervisor è un server di rete per le stazioni in esecuzione su qualunque tipo di JACE. E' stato sviluppato per utilizzare al meglio le potenzialità di Internet e consente l'integrazione di protocolli standard come LonWorks e Modbus. Consente inoltre la gestione e configurazione centralizzata della rete fornendo servizi per i database, i messaggi e gli allarmi. Inoltre permette di controllare il sistema con una rappresentazione grafica dell'interfaccia utente analoga a quelli dei più comuni SCADA di mercato [18].

Nello specifico le principali caratteristiche sono le seguenti:

- Supporta un numero illimitato di utenti di rete connessi con Web Browser standard;
- Interfaccia utente dinamica, realizzata con Applet Java;
- Supporta la gestione dei database per l'archiviazione dei dati;
- Permette una sofisticata gestione degli allarmi ed eventi, compresa la trasmissione di mail di notifica;
- Gestisce il backup automatico dei file di configurazione dei JACE in rete;
- Include tool per la creazione di applicativi di controllo dei dispositivi di campo, inclusa la creazione dell'interfaccia grafica;
- Password di protezione e sicurezza utilizzando tecniche standard Java per la cifratura e l'autenticazione;

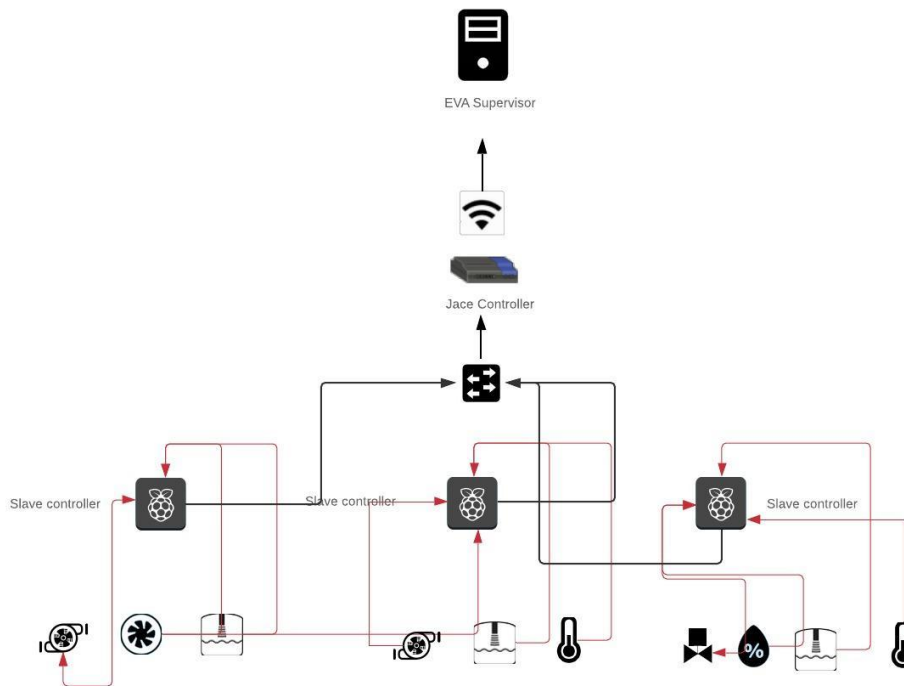


Figura 6.4: Architettura del sistema con JACE controller e Supervisor

Dimensionamento del sistema

La figura 6.1 permette di definire una serie di regole per il dimensionamento del sistema in modo da poter determinare il numero di Supervisor in funzione del numero di JACE controller e delle caratteristiche del sistema.

Caso 1

Si dispone di una sola JACE, al cui interno contiene ed implementa un'interfaccia utente di poche pagine con un archivio dei dati di log limitato. In questa situazione non è richiesto nessun Supervisor, dato che il controller può operare autonomamente data la semplicità del sistema.

Caso 2

Si dispone di una sola JACE, che implementa un'interfaccia utente più complessa ed un archivio dei dati più grande rispetto al Caso 1. In questa situazione, è necessario l'utilizzo di funzionalità avanzate come l'invio di una

mail in caso di allarme, memoria per le pagine grafiche e per l'archivio dei dati. In tal caso è necessaria la presenza di un Supervisor.

Caso 3

Si dispone di più JACE, con un'unica interfaccia utente per sottosistemi gestiti da JACE controller diversi con il rispettivo archivio di dati di log. Per avere un'unica interfaccia utente, le pagine grafiche devono essere caricate sul Supervisor per poi fare riferimento ai dati estratti dai singoli JACE.

L'architettura proposta in figura 6.1, fa riferimento al Caso 2. Nonostante ciò, possibili sviluppi futuri e un'ampliamento del numero di sensori e attuatori utilizzati nel modello, porterebbero allo sviluppo di un sistema le cui caratteristiche rientrerebbero nella terza casistica di quelle descritte.

6.2 Interfacciamento protocolli

Come già specificato, i dispositivi adottati per l'interfacciamento di reti supportate da diversi protocolli, sono i JACE controller, i quali rappresentano una tecnologia aperta a tutti i protocolli standard e proprietari. Questo aspetto è una caratteristica specifica del sistema in quanto gli permette di essere indipendente dai diversi condizionamenti e requisiti di progetto.

Per il sistema proposto ciò significa che:

- All'operatore viene presentata un'unica interfaccia da browser web, abilitata per le applets Java; questa risulta essere intuitiva nell'utilizzo e utilizzabile in maniera concorrente da uno o più utenti;
- Si possono integrare sottosistemi con protocolli diversi, riducendo i costi dell'infrastruttura di rete e di supervisione;
- E' garantito il supporto di database gestionale diffusi sul mercato come Oracle e SQL: grazie alle potenzialità offerte dal linguaggio Java, e possibile scambiare dati con i sistemi standard di archiviazione dei dati;

6.3 Software di sviluppo e applicativo

Il framework Niagara, come spiegato nel Capitolo 4, tra le principali caratteristiche, offre la possibilità di convertire in oggetti Java i dispositivi di campo ed i loro profili, utilizzando la tecnica dei *Java Bean object Modeling*.

A livello di rete, la piattaforma ha permesso:

- Acquisizione e riconoscimento dei dispositivi di campo;
- Configurazione (binding);
- Definizione della logica di controllo;

Identificata come sottorete quella relativa al singolo JACE controller, il database di quest'ultima è residente all'interno del controller in modo da tale da offrire all'utente la possibilità di operare su tutti i dispositivi di campo a prescindere dal protocollo utilizzato.

Le viste grafiche del sistema (tecnicamente chiamata viste "Px"), forniscono dati in tempo reale utilizzando sia immagini che testo. Queste rappresentano l'interfaccia principale del sistema e forniscono collegamenti ad altri elementi grafici, fornendo inoltre speciali viste tramite le quali eseguire comandi o azioni speciali per il controllo dell'intero sistema.

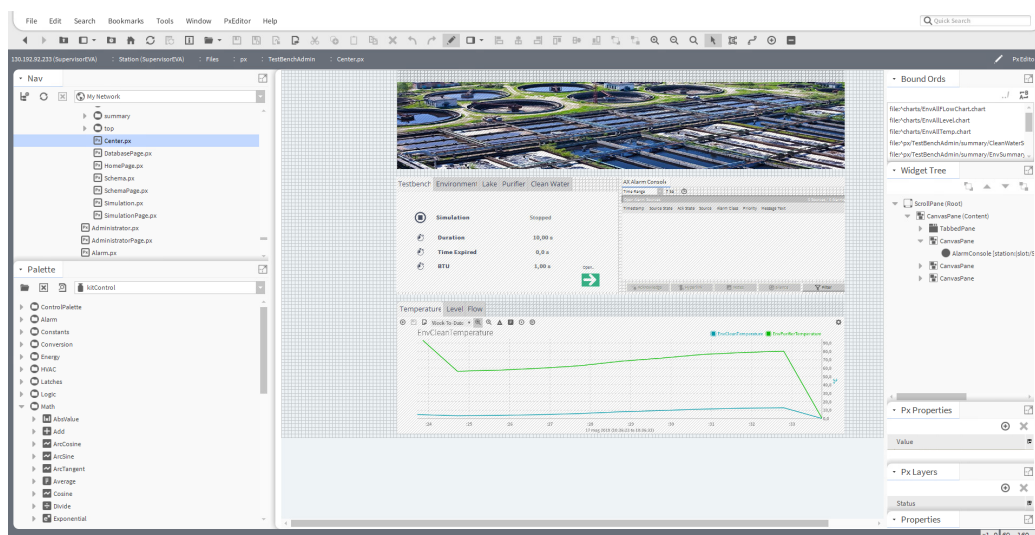


Figura 6.5: Vista Px nel Workbench

Come mostrato nella figura 6.5, ogni vista Px può visualizzare una grafica animata con informazioni di sistema. Per grafica animata s'intende la capacità da parte dell'applicativo di aggiornare i componenti grafici al variare dei valori degli oggetti mappati all'interno della logica di controllo, i quali rappresentano le fonti dei componenti grafici stessi. Una "grafica" può semplicemente essere una singola parola di testo ("ON") o un numero ("24"), oppure un'immagine animata (attuatore lampeggiante). Gran parte delle viste Px sono state assemblate utilizzando una varietà di elementi, chiamata *widget*, che includono elementi di layout grafico, immagini e testo collegati

agli oggetti del modello logico in real-time. I widget forniscono la visualizzazione grafica dei dati, pertanto l'interfaccia utente, è composta da uno o più widget assemblati in un file Px, utilizzati per la visualizzazione tramite web browser.

6.3.1 EVA Virtual SCADA

Per il monitoraggio e la supervisione del sistema fisico, l'utente può autenticarsi tramite browser all'applicativo web sviluppato durante il lavoro di tesi, il quale svolge le funzioni di un sistema SCADA (Supervisory Control And Data Acquisition) creato ad-hoc per l'architettura proposta. Ogni utente disporrà di uno o più privilegi, il che distingue ognuno di essi per il tipo di operazioni che questi potranno eseguire. In fase di sviluppo sono stati definiti principalmente i seguenti ruoli:

- *SCADA Programmer*: dispone dei privilegi necessari per modificare la logica di controllo del controller di sistema;
- *EVA Manager*: dispone dei privilegi per accedere alla *control room* virtuale, dedicata all'acquisizione dei dati e la fruizione del sistema nel suo complesso;
- *Testbench Administrator*: dispone dei privilegi per la modifica o la creazione dei moduli simulativi e del file di configurazione del testbench;
- *EVA Guest*: dispone solamente dei privilegi di monitoraggio e visualizzazione del sistema;

6.3.2 SCADA Programmer

Come mostrato in figura 6.6, l'utente con il ruolo di SCADA Programmer è l'unico a disporre dei permessi necessari per il binding dei punti, la creazione o modifica della modellazione logica del JACE controller tramite il Workbench Niagara.

Dopo aver effettuato l'accesso alla stazione in esecuzione sul controller, l'utente avrà accesso ai file di configurazione di questa. Qualora si renda necessario, disporrà delle funzionalità offerte dal framework per operare un upgrade della logica di controllo (es. nuovi serbatoi, sensori e attuatori) o una sua modifica. Tramite l'utilizzo di una *palette* (in basso a sinistra nella figura), lo *SCADA programmer* può aggiungere nuovi componenti in accordo con le specifiche di intervento. Ognuno di essi variano per tipo, input e output. I principali ed i più utilizzati in fase di sviluppo sono stati i seguenti:

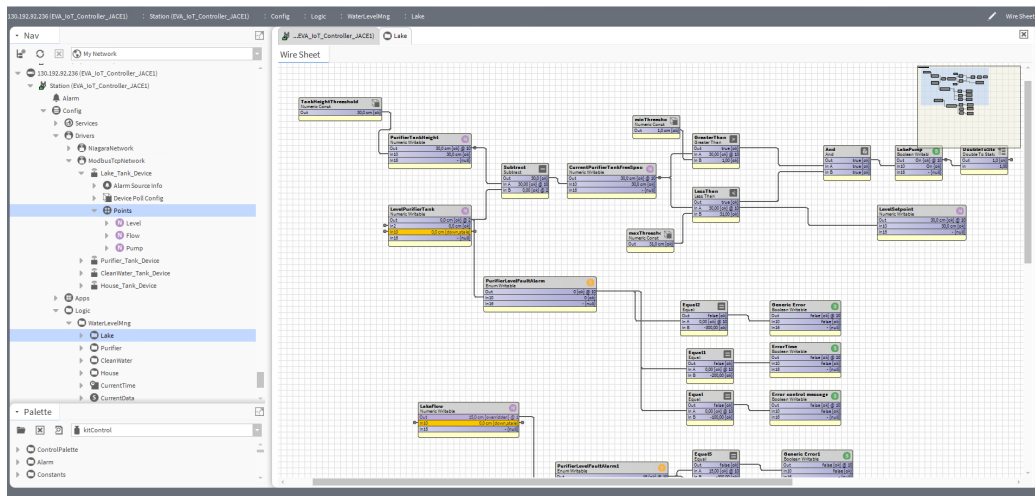


Figura 6.6: Scada Programmer view

- control: gruppo di component di tipo base, come numeric, boolean, enum, string;
- logic: gruppo di component che permettono di eseguire operazione logiche tra i vari component (AND, OR, XOR, NOT, ecc.);
- math: gruppo di component per effettuare operazione matematiche tra component e sui loro input e output (ADD, SUBTRACT, DIVIDE, MULTIPLY, SINE, ecc.);
- timer: gruppo di component per operazioni con durata limitata (ONE SHOT, BOOLEAN DELAY);
- select: gruppo di component per operazioni di selezione basate sugli output dei component (BOOLEAN SWITCH, SWITCH, SELECT, ecc.);
- constant: gruppo di component per definire costanti di input sui component (NUMERIC CONSTANT, BOOLEAN CONSTANT, ecc.);

6.3.3 EVA Manager

L'utente con il ruolo di *Manager*, avrà accesso ad un'interfaccia (figura 6.7) tramite la quale potrà eseguire diversi tipo di operazioni al fine di monitorare e controllare il sistema fisico da remoto tramite web browser.



Figura 6.7: Homepage Eva Manager

Tramite la barra di navigazione a sinistra è possibile accedere ai pannelli di controllo relativi ai tre principali settori dell'acquedotto: Lake, Purifier e Clean Water.

Com'è mostrato nella figura 6.8, l'operatore, tramite un'interfaccia web dinamica, sarà in grado di monitorare i seguenti parametri:

- Lo stato del sistema SCADA (Online/Offline);
- Il timestamp di sistema;
- Il livello dell'acqua all'interno del serbatoio;
- Lo spazio rimanente al suo interno;
- Lo stato ed i relativi comandi degli attuatori (pompa centrifuga, elettrovalvola);
- Una console di notifica degli allarmi;
- I grafici riguardanti i valori misurati;
- I valori in real-time di parametri come: flusso dell'acqua, temperatura e umidità;

Nell'eventualità in cui, nella console degli allarmi, venisse notificata una particolare criticità (es. malfunzionamento di un attuttore, livello dell'acqua

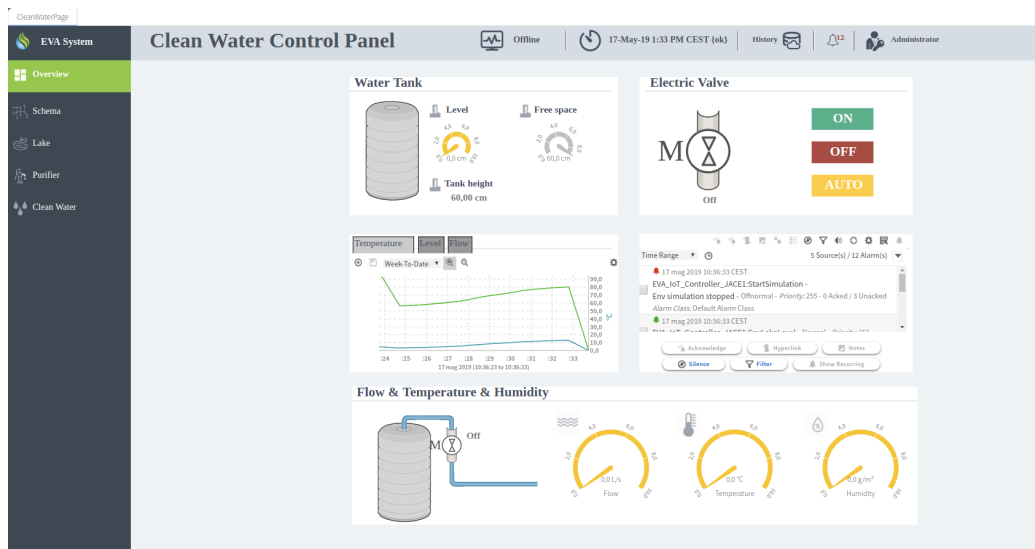


Figura 6.8: Clean Water Panel Control

oltre la soglia di sicurezza). l'operatore può eseguire operazioni di arresto o riavvio degli attuatori.

6.3.4 Testbench Administrator

L'utente che riveste questo ruolo, dispone dei permessi necessari per la modifica dei moduli simulativi, per testare, convalidare e verificare la complessità della logica di controllo del modello realizzato. Spostandosi nella sezione Testbench dell'applicazione (figura 6.9), si ha a disposizione una vista tramite la quale impostare i parametri della simulazione e darne l'avvio. Nello specifico, è possibile settare:

- La fonte d'acqua (superficiale o sotterranea);
- La durata della simulazione;
- La *Basic Time Unit* (BTU);
- La percentuale di acqua consumata dalle utenze;
- La capacità massima delle pompe;

Durante l'esecuzione della simulazione, è possibile monitorare i livelli dei serbatoi di tutti i settori del sistema fisico, inclusi i valori di temperatura, umidità e flusso dell'acqua. Per la visualizzazione delle misure, è possibile

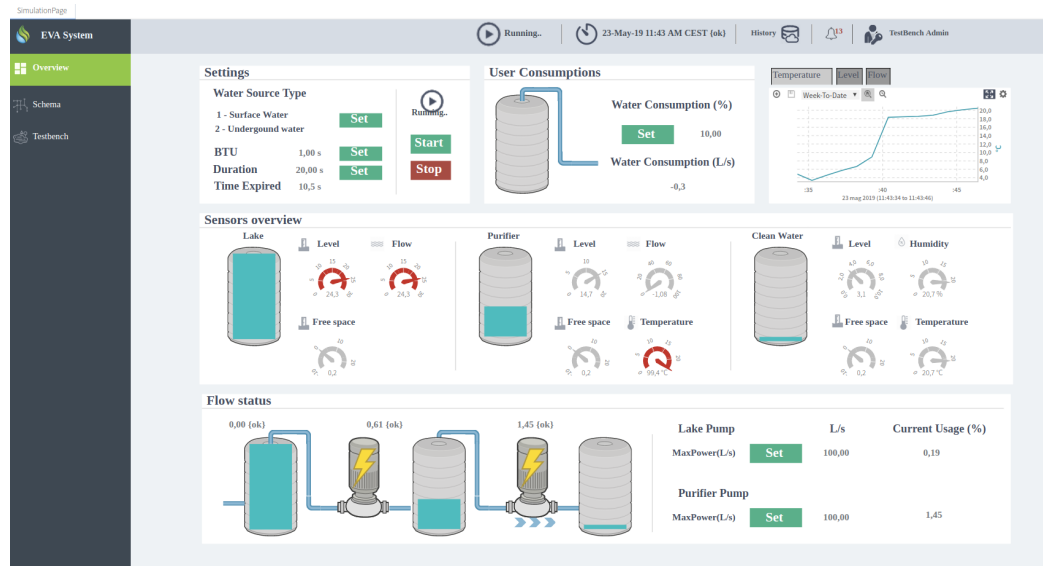


Figura 6.9: Testbench Panel

consultare i grafici presenti nel pannello oppure accedere alla sezione dedicata all'archiviazione, con la loro rappresentazione per mezzo di grafici e tabelle.

6.3.5 EVA Guest

Nella sezione dell'applicazione denominata *Schema*, viene mostrata (figura 5.10(a)) l'architettura del sistema fisico. La vista realizzata risulta essere semplice e dinamica, con l'aggiornamento in tempo reale delle misure e dei componenti grafici corrispondenti.

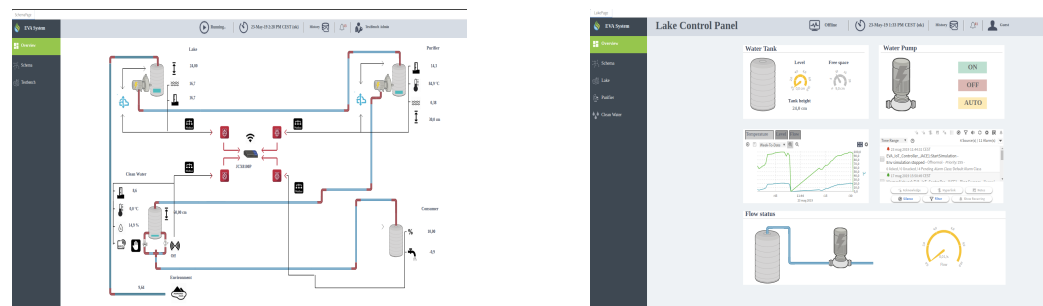


Figura 6.10: (a) Schema Panel. (b) EVA Guest Panel

Un account "Ospite" non possiede dei permessi necessari per accedere ai file di configurazione del controller, o ulteriori dati privati. Di fatto,

può usufruire di funzionalità limitate che impediscono all'utente di utilizzare strumenti di sistema, apportare modifiche e inviare comandi al sistema fisico.

Come mostrato in figura 5.10(b), tutti i comandi operativi degli attuatori sono disabilitati, dato che l'utente non dispone dei permessi necessari per il loro utilizzo. Egli potrà visualizzare tutte le misure in tempo reale provenienti dai componenti del sistema fisico e la console di allarme dell'acquedotto. Le uniche azioni eseguite dall'utente, che generino degli output, sono rappresentate dalle operazioni di esportazione e download degli archivi delle misure.

6.3.6 Archiviazione

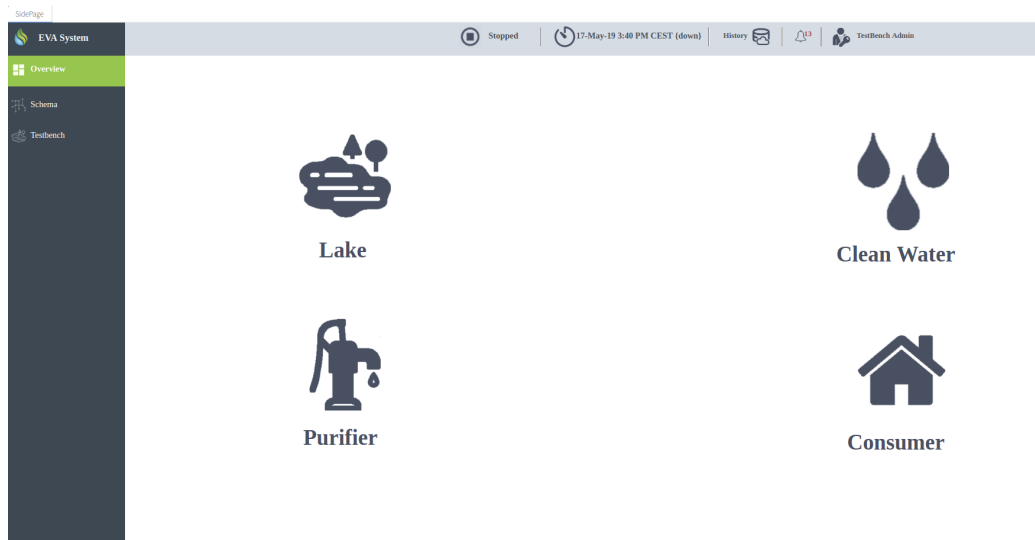


Figura 6.11: History Side Panel Testbench Administrator

Come descritto nel Capitolo 4, le "history" rappresentano una raccolta di valori di dati specifici di un punto specifico (sensore, attuatore) all'interno di una determinata stazione locale o remota. Per quanto riguarda la loro rispettiva visualizzazione, spostandosi nella sezione dell'applicativo denominata *History*, è possibile accedere alla rappresentazione quest'ultimi, mediante l'utilizzo di :

- grafici (barChart, LineChart);
- tabelle;

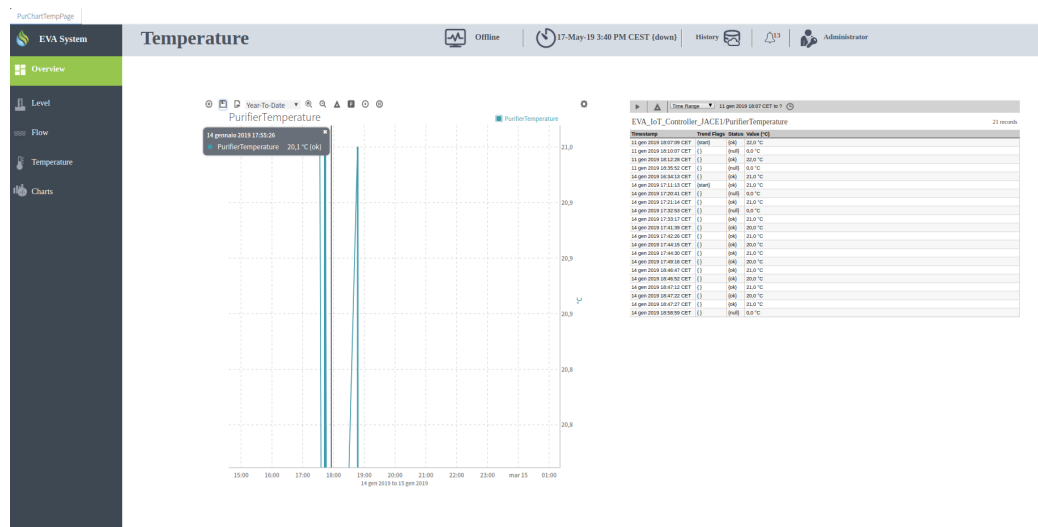


Figura 6.12: Temperature History Panel

Come mostra la figura 6.11, l'operatore potrà scegliere di visualizzare i valori memorizzati di uno specifico settore dell'acquedotto, ovvero:

- Lake;
- Purifier;
- Clean Water;
- Consumer;

In accordo con la scelta effettuata dall'utente, è possibile visualizzare grafici e tabelle per ogni sensore e attuatore contenuto all'interno del sistema fisico, le cui variazioni sono rilevate e memorizzate dal controller. Tramite la vista mostrata nella figura 6.12, possono essere effettuate diverse operazioni sui dati archiviati fino in quel momento, come:

- ricerca per time-range;
- cambio topologia del grafico (barChart, lineChart, shadowChart, discreteLineChart);
- esportazione, stampa e download dei dati (.chart, .csv);

Nella modellazione del sistema, si è scelto di raccogliere e memorizzare i valori di ogni singolo punto soltanto di fronte ad una sua variazione sensibile del suo valore, sfruttando i vantaggi sia in termini di prestazioni che di memoria di storage risparmiata.

Capitolo 7

Conclusioni e lavori futuri

Il modello EVA, prova ad emulare un sistema di controllo industriale che utilizza componenti a basso costo. Questo progetto mira a mostrare e descrivere i passi fondamentali, intrapresi nella progettazione e realizzazione di un sistema SCADA, per il monitoraggio ed il controllo di un emulatore fisico di acquedotto. Mediante l'utilizzo del framework Niagara e del JACE 8000 controller, sono state sfruttate le caratteristiche ed i principali vantaggi ottenuti, in fase di sviluppo, in termini di prestazioni ed integrazione con i dispositivi di campo.

Lo sviluppatore, a fronte delle specifiche richieste dal sistema, è in grado, a seconda delle necessità, di modellare una logica di controllo del controller discretamente complessa e di testarne il funzionamento mediante l'utilizzo del Workbench.

Tramite l'applicazione web sviluppata, ogni utente, dopo previa autenticazione, distinta per ruoli e poteri esecutivi, sarà in grado di eseguire diverse operazioni (monitoraggio, controllo, gestione dei dati, gestione degli allarmi) mediante l'interazione con delle viste dinamiche e animate, che offrono una panoramica in real-time sullo stato del sistema. Ogni operazione di input ha un effetto diretto sul sistema fisico (es. accensione e spegnimento degli attuatori, apertura e chiusura delle elettrovalvole, ecc.).

Dal punto di vista della comunicazione tra controller e dispositivi di campo, possono essere introdotti diversi miglioramenti, come per esempio la cifratura dei pacchetti Modbus TCP/IP, i quali, nell'implementazione attuale, sono trasmessi in chiaro.

Per quanto riguarda l'aspetto grafico dell'applicazione, si può ottenere un notevole upgrade dell'interfaccia, in termini di design e animazioni, mediante l'utilizzo di widget sviluppati da aziende terze, che offrono una grafica più accattivante e snella rispetto a quella offerta dai widget proprietari del framework Niagara.

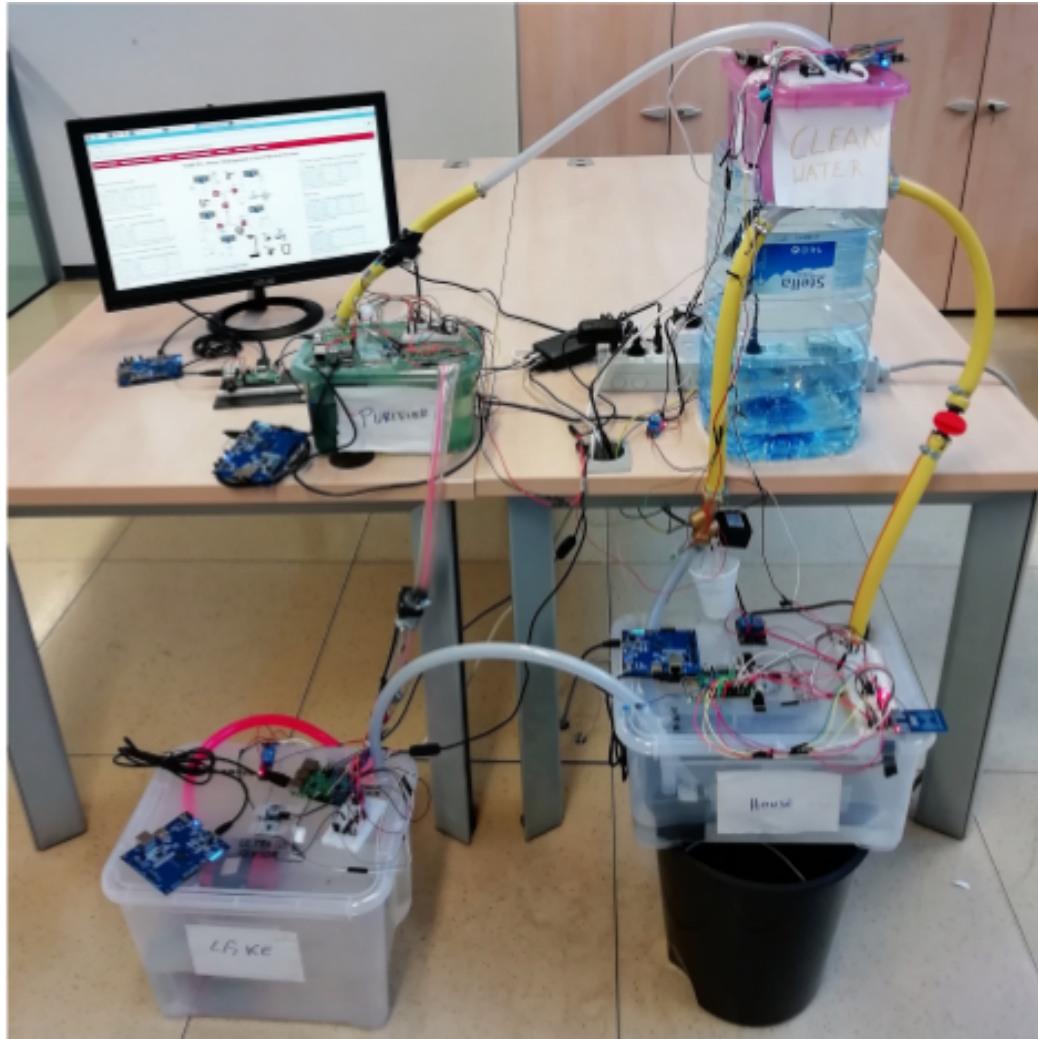


Figura 7.1: Modello fisico EVA

Una delle principali sfide future è ripensare la sicurezza da una nuova prospettiva, dove gli aspetti cyber, umani e fisici siano integrati nel loro insieme. Il sistema realizzato ha l'obiettivo principale di essere inserito all'interno del contesto di Cyber Range, che a sua volta si traduce nella definizione di Cyber Range ibrido. Come spiegato, la tipologia ibrida coniuga la dinamicità e la flessibilità della virtualizzazione con l'impiego di dispositivi fisici. All'interno di tale contesto, sarà possibile testare la resilienza di un'infrastruttura critica, come quella simulata, dal punto di vista della sicurezza informatica.

Bibliografia

- [1] J. Lin A. Humayed, F. Li, and B. Luo. *Cyber-Physical Systems Security-A Survey*. 2017.
- [2] A. Armando. Attacco e difesa in ambiente simulato. cyber range. 2016.
- [3] B.Pandey, G.Airò Farulla, M.Indaco, and P.Prinetto. Survey of computational ability of sensors-actuators in water management cyber-physical system jour of adv research in dynamical and control systems. 2018.
- [4] Cogeme. Acquedotti, lotta alle perdite. <http://www.aob2.it/media/File/UTENTI/ACQUEDOTTO/articolo%20rivista%20perdite%20acquedotto.pdf>, 2004. [Online; accessed 02-March-2019].
- [5] A. Cook, R. Smith, L. Maglaras, and H. Janicke. Measuring the risk of cyber attack in industrial control system. 2016.
- [6] Maximilian Frank, Maria Leitner, and Timea Pahi. Design considerations for cyber security testbeds: A case study on a cyber security testbed for education. 2017.
- [7] H.A. Kruger, L. Drevin, and T. Teyne. Towards a framework for evaluating ict security. 2005.
- [8] LG. Getting Started with Niagara Framework ® Software Manual. https://lghvac.com/resource-service?filename=SOM_Getting_Started_with_Niagara_Framework.pdf/, 2017. [Online; accessed 19-May-2019].
- [9] E. Lo and M. Marchand. *Security audit: A case study*. 2004.
- [10] Raspberry Pi (Trading) Ltd. Raspberry pi3 compute module datasheet. 2016.

- [11] M.Barrere, C.Hankin, A. Barboni, G.Zizzo, F. Boem, S. Maffeis, and T. Parisini. Cps-mt: A real-time cyber-physical system monitoring tool for security research. 2018.
- [12] M.Unver and C.Canbay. Cyber security with national and international aspects. 2010.
- [13] C. Palmer. Ethical hacking. 2001.
- [14] R.Drath and A.Horch. Industrie 4.0: Hit or hype. 2018.
- [15] C. Seng, C. Lian, and T. Su-Han Victor. Automated red teaming: A proposed framework. 2007.
- [16] T.George. Introduction to modbus serial and modbus tcp. 2008.
- [17] Z. Tian, Y. Cui, L. An, S. Su, X. Yin, L. Yin, and X. Cui. A real-time correlation of host-level events in cyber range service for smart campus. 2018.
- [18] Tridium. Niagara AX-3.x Browser Access Guide. <http://www.htccustomer.com/files/Niagara%20AX%20Users%20Guide.pdf>, 2007. [Online; accessed 3-April-2019].
- [19] Tridium. Jace 8000 controller data sheet. 2018.
- [20] G. Vigna. Teaching hands-on network security: Testbeds and live exercises. 2003.
- [21] Wikiwand. Acquedotto. <https://www.wikiwand.com/it/Acquedotto>, 2017. [Online; accessed 20-March-2019].
- [22] H. Winter. System security assessment using a cyber range. 2012.