Politecnico di Milano
Scuola di Ingegneria Industriale e dell'Informazione
Corso di Laurea Magistrale in Ingegneria Informatica
Dipartimento di Elettronica, Informazione e Bioingegneria

Un framework teorico e tecnologico per sistemi ciberfisici intelligenti

Relatore: Prof. Manuel ROVERI
Correlatore: Prof. Cesare ALIPPI

Tesi di laurea di:
Matteo CESANA Matr. 817976

Anno Accademico 2014–2015

Ai miei genitori, che mi hanno accompagnato e grazie ai quali ho potuto portare a termine questo percorso, ed alla zia Ivana e tutti quelli che mi hanno sostenuto con il loro interesse in questi anni.

Sommario

I sistemi ciberfisici sono dei sistemi informatici capaci di operare in modo attivo ed interagire con l'ambiente in cui operano al fine di fornire un'applicazione o un servizio. Questi sistemi, che generalmente operano in condizioni reali (o anche ostili), possono soffrire di guasti o malfunzionamenti nel sistema (ad es. nell'hardware/software, nei sensori o negli attuatori) o le condizioni ambientali in cui il sistema opera possono cambiare andando ad influenzare le prestazioni dell'applicazione (o servizio) considerata.

Per questo motivo, negli ultimi anni, si stanno sviluppando dei sistemi ciberfisci dotati di meccanismi di adattatività che permettono al sistema stesso di adattarsi a nuove condizioni di funzionamento.

In questa tesi viene presentato un framework teorico e tecnologico per la progettazione di sistemi ciberfisici intelligenti capaci di operare in un ambiente che cambia e adattandosi a nuove condizioni di funzionamento (andando a gestire guasti o malfunzionamenti nel sistema o cambiamenti nell'ambiente in cui il sistema opera). Per raggiugere questo scopo è necessario dotare il sistema di meccanismi intelligenti sia a livello di singolo dispositivo che a livello di rete di dispositivi.

A livello di singolo dispositivo (che è tipicamente dotato di una serie di sensori e capacità comunicativa) i meccanismi di intelligenza hanno come scopo quello di modellare il fenomeno fisico sotto osservazione ed andare ad identificare cambiamenti nei dati acquisisti. Per fare questo ci si appoggerà a tecniche di identificazione di sistemi, meccanismi di analisi del residuo, change detection test (che sono tecniche statistiche per rilevare variazioni nel comportamento statistico di un processo generante i dati) ed analisi statistiche (per ridurre l'occorrenza di falsi positivi).

Quando viene rilevata una variazione da parte di un dispositivo, si attiva un livello di intelligenza di rete (gestito da un coordinatore di rete) che ha come scopo quello di sfruttare le relazioni spaziali e temporali nei dati acquisiti per validare il cambiamento a livello di rete e distinguere tra cambiamento nell'ambiente o guasto in un singolo dispositivo. Successivamente, dopo aver identificato la causa del cambiamento, si attiva una fase di adattamento al cambiamento sfruttando informazioni riguardanti il cambiamento (ad es. una stima dell'istante di tempo in cui il cambiamento è iniziato).

Oltre alla formulazione teorica, in questa tesi viene proposto un framework tecnologico per sistemi ciberfisici intelligenti basato su quanto proposto che si appoggia a schede Nucleo ST con le relative schede di espansione per i dispositivi del sistema e che saranno connessi tramite Bluetooth Low Energy ad uno smartphone Android nel ruolo di coordinatore della rete. Il sistema realizzato è stato testato con efficacia in un'ampia campagna sperimentale.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduzione

L'evoluzione tecnologica che il mondo sta vivendo negli ultimi anni è accompagnata da nuovi scenari e possibilità di innovazione ancora da esplorare. Secondo le previsioni dello studio Cisco Visual Networking Index (VNI) Mobile in Italia, nel 2019 ogni italiano avrà in media 3 dispositivi mobile, inoltre una grande crescita è prevista per i dispositivi che implementano connessioni M2M (Machine-to-Machine), cioè che consentono la comunicazione tra dispositivi diversi. La tendenza sarà quella di avere sempre più dispositivi connessi in rete e, conseguentemente, una maggiore quantità di informazioni da elaborare.

Negli ultimi anni si è sentito sempre più spesso parlare di Internet of Things (IoT): quest'estensione dell'utilizzo della rete consente agli oggetti di comunicare scambiando informazioni sia fra loro che con l'utente. Su questa idea negli ultimi anni si sono apertiscenari applicativi in diversi settori: si possono portare esempi di IoT già dalla vita comune di ogni giorno, quali la domotica, ovvero la gestione dei dispositivi nella casa intelligente, che può regolare la temperatura interna in base alla presenza di persone, rilevare eventuali fughe di gas, incendi o allagamenti e comunicare con il telefono del proprietario per eventuali notifiche, oppure i dispositivi di Health Care, impiegati nella diagnosi medica, oppure ancora impiegati per la supervisione di mari o foreste.

Lo scenario in cui si pone questa tesi è quello dei Cyber-physical Systems. Questi sistemi hanno l'obiettivo di acquisire dati dall'ambiente fisico in cui sono immersi attraverso dispositivi di controllo, ed integrare le informazioni raccolte per creare una rappresentazione informatica del contesto reale in cui operano. Per svolgere questo compito il sistema è tipicamente dotato di una rete di sensori wireless, wired o ibrida, per interfacciare il sistema all'ambiente, ed una parte di intelligenza, che elabora le informazioni ottenute per descrivere i fenomeni fisici monitorati. Essendo questi sistemi dotati di dispositivi pervasivi in grado di comunicare tra loro, si può capire che l'ossatura che li compone è un sistema IoT, a cui vengono aggiunte delle funzionalità specifiche che sappiano coordinare e rendere intelligente la soluzione. Lo

sviluppo tecnologico nella realizzazione di sistemi microelettromeccanici (MEMS) e nella comunicazione radio a corto raggio ha permesso la realizzazione di queste reti le cui unità sono dispositivi a basso costo e basso consumo energetico. In una rete di sensori tradizionale le comunicazioni sono perlopiù asimmetriche, quindi i nodi hanno il compito di acquisire le misure ambientali ed inviare i dati a dispositivi più potenti con il compito di immagazzinare le informazione e compiere l'analisi di questi dati. In un CPS ogni dispositivo è dotato di microprocessore che permette di compiere una prima elaborazione dell'informazione a livello locale. Ogni nodo è connesso con il sistema, permettendogli di comunicare ed interagire con le altre unità, permettendo lo scambio delle informazioni necessarie all'analisi. Questo scenario applicativo va visualizzato in un contesto dove i nodi sono dotati di energia limitata, ed essere in grado di evitare gli sprechi significa aumentare la vita utile del dispositivo. A questo scopo si progetta il sistema in modo di ridurre le comunicazioni, che consumano energia, trasmettendo dei dati sintetici che rappresentano il comportamento rilevato dal nodo invece che ogni sua misura.

1.1 Cyber-Phisical Systems

Con il termine Cyber-Phisical Systems si vuole definire quei sistemi il cui scopo è quello di monitorare ed interagire con il sistema fisico in cui operano. Un CPS è composto da una rete di dispositivi, ciascuno dotato di capacità computazionale e di controllo, con il compito di acquisire dati dall'ambiente in cui si trova, di elaborare informazioni provenienti dai vari nodi e di costruire una rappresentazione informatica del comportamento dei fenomeni fisici monitorati. Il sistema è formato da una componente sensoristica, necessaria per acquisire i dati ambientali, ed una componente di intelligenza, capace di comprendere e valutare le informazioni disponibili.

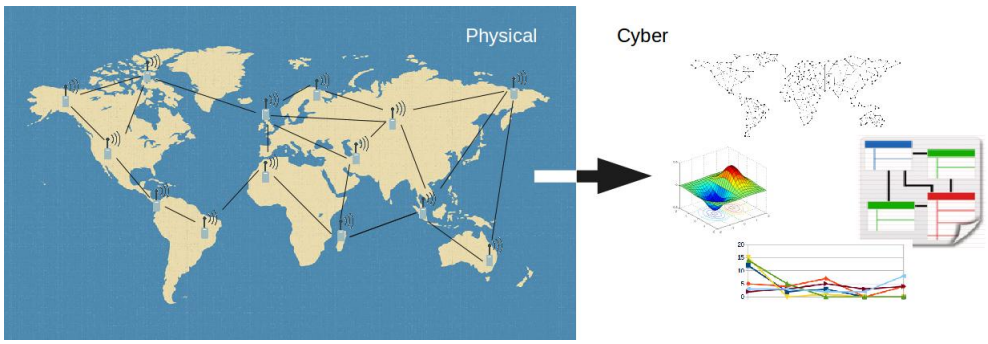

Figure 1.1: Il sistema ciberfisico viene integrato nell'ambiente, permettendo elaborazioni informatiche dei dati rilevati rappresentanti l'ambiente.

I singoli dispositivi che formeranno i nodi della rete devono esser dotati di capacità di comunicazione, preferibilmente wireless, in modo da poter comunicare e permettere all'intero sistema di sfruttare i dati locali. Avere a disposizione una mag-

giore quantità di dati provenienti da diversi nodi significa poter costruire un modello più accurato.

I CPSs possono includere anche degli attuatori, che andranno ad agire in modo da modificare i valori dei fenomeni fisici qualora si verifichi la necessità.

1.1.1 Caratteristiche

I sistemi adottati per interagire con il mondo fisico molto spesso sono dei piccoli sistemi embedded creati ad hoc per la specifica applicazione. L'obiettivo è quello di sfruttare al massimo le capacità del sistema, adottando degli algoritmi che ottengano la miglior efficienza, considerando in fase di progettazione la limitatezza delle risorse a disposizione.

Questi particolari sistemi hanno caratteristiche che li contraddistingono nel panorama dalle applicazioni informatiche:

- il loro scopo è quello di integrare il processo fisico con appositi componenti informatici progettati per adattarsi al fenomeno fisico e simularne digitalmente le caratteristiche;
- le risorse informatiche sono limitate in base alle caratteristiche fisiche delle risorse hardware impiegate per lo sviluppo embedded dei dispositivi;
- si tratta di un sistema distribuito, dove elementi potenzialmente eterogenei possono comunicare tramite le tecnologie di comunicazione (WLAN, Bluetooth, ...);
- il sistema deve essere in grado di riadattarsi e riconfigurarsi qualora se ne verifichi la necessità;
- generalmente la struttura di ogni componente si sviluppa in tre parti: logica applicativa, connettività e sensorialità;
- il sistema deve monitorare periodicamente il fenomeno fisico, comunicando all'utente lo stato dell'ambiente e permettendogli potenzialmente di interagire con esso.

Dal 2006 applicazioni che sfruttano questo tipo di sistemi hanno attratto l'interesse della comunità scientifica e sono stati individuati possibili utilizzi in svariati campi.

Come già accennato, una parte di un CPS è costituita da dispositivi distribuiti che comunicano tra loro. Questo concetto ha preso negli ultimi anni il nome di Internet of Things, e viene considerato un' estensione del concetto di rete. Ogni dispositivo può comunicare con gli altri e mettere a disposizione della collettività le proprie informazioni. L'idea è simile a quella che sta alla base delle relazioni umane. Ogni persona è contraddistinta da un nome e si relaziona all'ambiente che

gli sta intorno tramite i cinque sensi. Essa sa elaborare e trovare il significato delle informazioni che acquisisce ed è in grado di comunicare con le altre persone con un linguaggio comune. Allo stesso modo, ogni dispositivo è contraddistinto da un nome o indirizzo, e si relaziona all'ambiente che gli sta intorno tramite i propri sensori. Questo è programmato per elaborare e trovare il significato delle informazioni che acquisisce, ed è in grado di comunicare con gli altri dispositivi tramite tecnologie di comunicazione comuni.

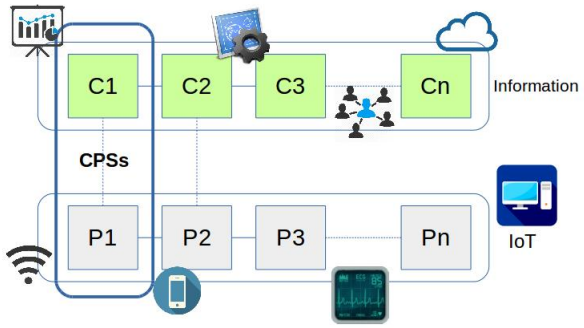

Figure 1.2: Struttura di un'applicazione CPS: interazione tra risorse fisiche capaci di comunicare tra loro (P1, P2,..., Pn) e sistemi informatici (C1, C2,...,Cn).

Un CPS utilizza quindi l'idea di IoT per stabilire le connessioni funzionali tra i diversi dispositivi di cui è composto e studia le relazioni tra entità fisiche e informatiche, integrando le funzionalità hardware, middleware e logica computazionale.

L'Internet of Things non va considerato come il caso generale dei CPSs, ma uno strumento che questi utilizzano per implementare la rete fisica con tutti i suoi componenti e gestirne le comunicazioni.

1.1.1.1 Adattativo e reattivo rispetto l'ambiente

Il sistema ciberfisico progettato deve avere costantemente sotto controllo l'attuale situazione dei fenomeni fisici monitorati. Una caratteristica fondamentale di questi sistemi è la loro capacità nel rilevare eventi che introducono un cambiamento ed essere in grado di reagire ad essi ri-adattandosi alle nuove condizioni ambientali. Va considerato che ogni dispositivo della rete è isolato dagli altri, quindi l'unica informazione che può sfruttare sono i dati da esso prodotti. Effettuando misurazioni per periodi lunghi, è possibile ricavare una quantità di informazioni che permettono di identificare il pattern in grado di descrivere il comportamento naturale del fenomeno fisico assunto finora e che si prospetta essere costante. Qualora il nodo, monitorando i valori, recepisca le misure come non più concordi al comportamento previsto, si deduce che il modello usato non sia più valido per le condizioni ambientali a causa di un cambiamento che lo ha interessato: il sistema, allora, dovrà sostituire il modello

obsoleto con uno nuovo ed eseguire le future analisi sulla base di questo.

È possibile che utilizzando una rete di sensori diversi di essi siano utilizzati per monitorare caratteristiche ambientali simili oppure in relazione fra loro: ad esempio utilizzando controllori ridondanti o sensori di temperatura distanti fra loro, ma in un medesimo ambiente, si ipotizza che i risultati mostrino una qualche somiglianza. Le relazioni, però, non sono esclusivamente fra sensori fra loro omogenei: infatti pensando a un giroscopio e a un accelerometro, entrambi posti a monitorare i movimenti di una porta, a una variazione angolare corrisponderà la variazione di velocità causata dall'apertura o chiusura.

Successivamente alla rilevazione di un cambiamento da parte di un nodo è utile compiere una valutazione globale, chiedendo conferma alle altre unità con esso relazionate, che potrebbero essere state interessate dal cambiamento senza però averlo rilevato. Per poter compiere questo passo è fondamentale garantire la sincronizzazione di tutti i nodi: infatti se davvero i parametri dei diversi sensori fossero correlati, alla variazione di uno in un determinato momento dovrebbe corrispondere la variazione, più o meno accentuata, anche del secondo. Questa verifica permette di controllare la coerenza tra le informazioni rilevate dai diversi nodi, che a volte viene meno: potrebbe accadere che un sensore si sia guastato, e quindi il suo comportamento non sia più rappresentativo della reale misura ambientale o analogamente un evento eccezionale potrebbe interessare solo un dispositivo che risulterebbe non più conforme; inoltre potrebbero essere cambiate le dipendenze fra sensori oltre che i loro modelli. Diventa possibile trarre le giuste conclusioni grazie all'interoperabilità tra i diversi nodi, che scambiandosi informazioni possono avere una visione più elevata del singolo sensore.

1.1.1.2 Resilient and scalable

In un CPS, data la moltitudine di elementi che lo possono comporre, si potrebbe incorrere nell'impossibilità di accedere ad un nodo. Le cause possono essere molteplici: potrebbe essere un nodo considerato guasto dalla rete, e quindi le informazioni non sarebbero utilizzabili; oppure per un esaurimento della batteria il dispositivo potrebbe spegnersi; inoltre potrebbero esserci problemi di connessione, per cui la trasmissione risulterebbe momentaneamente sospesa. Diventa quindi importante garantire al sistema un'elasticità per poter usare il massimo delle informazioni disponibili, garantendo che la momentanea indisposizione di alcune di esse non ne infici il funzionamento. Inoltre la rete deve essere progettata per essere all'uopo espandibile o contraibile, aggiungendo o eliminando nodi: quindi l'intero sistema deve sapersi riadattare agilmente anche modicando la sua struttura. In caso di reti molto popolate c'è il rischio che si crei un overload di comunicazioni, dove la grande mole di informazioni trasmesse possa influire negativamente sul funzionamento del CPS. Per evitare che la gestione di tante informazioni necessarie al processo di analisi diventi

un fattore nocivo, è utile trasmettere un'informazione sintetica che sia frutto di una prima analisi a livello di nodo.

1.1.2 Progettazione

La progettazione di un CPS tipicamente viene compiuta su più livelli: ai più alti si definiscono le astrazioni che dovrà soddisfare il sistema, mentre ai più bassi si assegnano i compiti ai diversi componenti. Ad un primo livello vengono stabiliti i requisiti che deve soddisfare il sistema, determinando gli eventi che dovranno essere gestiti dai diversi dispositivi in base allo scopo specifico. Viene poi calcolato un modello analitico in grado di riprodurre il comportamento del particolare fenomeno fisico, utilizzando metodi numerici e di machine learning. Il sistema progettato deve saper compiere un'analisi real-time, orchestrando i vari componenti in modo da garantirne la coerenza dei dati. Va anche messo l'accento sulla sicurezza e la disponibilità del sistema che, essendo composto da una moltitudine di elementi, risulta suscettibile a guasti, che devono essere individuati ed isolati per non inficiare le caratteristiche del CPS funzionante.

1.1.3 Applicazioni classiche

Il concetto di CPS può essere applicato in svariati campi, ovunque ci sia la necessità di interagire con il mondo e voler monitorare un fenomeno fisico distribuito. Esso trova applicazioni nel controllo e sicurezza del traffico, controllo dei processi produttivi, strumentazione medica, domotica, reti energetiche, monitoraggio ambientale ed altri.

Nel campo dei servizi sanitari, viene utilizzato per il monitoraggio dei parametri vitali di un paziente o per rilevare i parametri che deve mantenere costanti una sala operatoria: questi vengono rilevati da appositi dispositivi hardware che, in modo più o meno decentralizzato, eseguono un'analisi per verificare l'assenza di criticità mediche.

Nel settore della distribuzione elettrica sono stati progettati sistemi per il miglioramento dell'efficiente distribuzione elettrica in base alla necessità, che prendono il nome di Electric Power Grid: delle centraline distribuite nella rete elettrica misurano il consumo in una determinata zona nei diversi orari, ottenendo informazioni che permettono di ottimizzare la distribuzione elettrica senza causare inutili sovraccarichi.

I CPSs soddisfano i requisiti per la progettazione di veicoli intelligenti, che sappiano identificare la propria posizione nello spazio e muoversi per arrivare alla destinazione impostata, valutando e gestendo la presenza di ostacoli fisici.

Questo approccio viene impiegato nel controllo ambientale, come le Wireless Sensor Network, ad esempio per verificare la stabilità di sistemi rocciosi in prossimità di

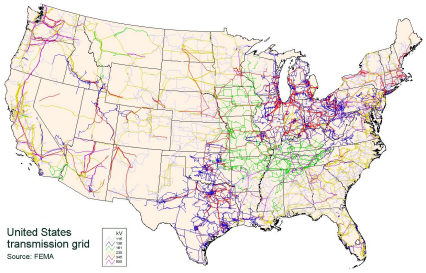

Figure 1.3: Esempio su vasta scala di Electric Power Grid.

zone abitative, monitorando i diversi parametri che ne verificano la stabilità, oppure per il controllo delle acque dove i parametri devono rispettare certi vincoli per il mantenimento dell'ecosistema.

1.2 Soluzione proposta

Lo scopo della tesi è quello di progettare un Sistema intelligente in grado di apprendere il comportamento dell'ambiente. In questo lavoro quando si parla di intelligenza si intende nell'accezione di capacità del sistema di adattarsi ad un cambiamento. Questo particolare tipo di sistema prende il nome di Self-adaptive System. Diversamente dai sistemi ciberfisici adattativi presenti in letteratura, il sistema proposto in questa tesi non è fornito di informazioni a priori sul fenomeno fisico che dovrà monitorare, ma si appoggerà ad una fase di training per acquisire le informazioni necessarie per impararne autonomamente il comportamento. Inoltre i componenti del sistema saranno dotati di meccanismi intelligenti che permetteranno di rilevare un cambiamento del fenomeno fisico percepito dai sensori, e sfruttando l'interoperabilità tra i dispositivi determinare la presenza di guasti all'interno della rete.

Per creare un sistema accurato atto allo scopo di individuare il comportamento ambientale verrà applicata una logica a due livelli: una prima a livello di unità eseguendo l'analisi locale, e la seconda a livello globale, dove un dispositivo più potente ha il compito di controllare ed aggregare le informazioni dei diversi dispositivi della rete. Avendo una visione più ampia sull'intero sistema è possibile individuare dispositivi dal comportamento difettoso, permettendo l'individuazione di guasti nella rete.

Nel sistema proposto i singoli nodi hanno il compito di monitorare l'ambiente e, sfruttando l'interoperabilità prodotta dalle informazioni condivise, riuscire ad inferire il modello dell'ambiente. I sistemi comprendono una componente hardware ed una software, in grado di acquisire informazioni dall'ambiente ed elaborarle al fine di descrivere l'andamento temporale del fenomeno fisico controllato. Ogni unità sarà predisposta per monitorare uno o più sensori che funzionano in parallelo. Si vuole dotare questo sistema di intelligenza, quindi instaurare meccanismi di apprendimento

del normale comportamento del fenomeno, per poter eseguire a run-time l'analisi sui valori che il fenomeno sta assumendo, per capire se sta seguendo l'andamento atteso oppure un evento sconosciuto ha causato un cambiamento rispetto alle regole note. In tal caso il sistema deve capire quando è incominciato il cambiamento, e sapersi adattare al nuovo comportamento, imparando le nuove regole che descrivono l'andamento del fenomeno fisico.

In una prima fase del lavoro viene proposta una soluzione teorica che soddisfa i requisiti di un sistema autoadattativo, fornendo gli strumenti metodologici necessari alla caratterizzazione dei comportamenti ambientali, la ricerca di cambiamenti ed il successivo riaddestramento. Vengono messe in evidenza le differenze tra le risorse impiegate per l'analisi locale e quelle necessarie a garantire il coordinamento dell'intero sistema.

Successivamente viene illustrato come è stata implementata una soluzione che utilizza i meccanismi descritti nella progettazione teorica per un sistema autoadattativo.

I nodi della rete sono stati sviluppati con schede Nucleo STM32 e relative schede di espansione fornite dalla STMicrelectronics: questo ha permesso di poter implementare la rete partendo da un solido background, riutilizzando librerie già implementate per la comunicazione e l'acquisizione dei dati su dispositivi general purpose semplici da programmare.

Come coordinatore, a cui tutti i nodi faranno riferimento, si è scelto di utilizzare uno smartphone, che si possa interfacciare all'utente ed aggreghi le informazioni provenienti dalla rete. Questa scelta trova le sue motivazioni in base al sempre più diffuso utilizzo di dispositivi mobile al buon rapporto tra semplicità di utilizzo e prestazioni.

La connessioni tra i nodi sono state implementate utilizzando la tecnologia Bluetooth Low Energy, spesso impiegata per la realizzazione di IoT data la sua elevata efficienza energetica rispetto ad altre tecnologie wireless.

Come già detto, la rete viene pensata per compiere test a livello gerarchico, come proposto in [1]. Si vuole quindi dividere la computazione su due livelli e distribuire l'intelligenza tra i componenti della rete. In questo modo si solleva il dispositivo centrale dal compito di accollarsi l'intera gestione ed analisi dei dati, ma viene sfruttata la capacità di elaborazione di ogni microcontrollore, aumentando la capacità elaborativa dell'intero sistema. Ogni dispositivo avrà bisogno di una tempo iniziale per addestrarsi ad imparare il normale comportamento dei parametri monitorati dai sensori. In questa fase l'ambiente deve essere stazionario, quindi si ipotizza che i suoi valori stiano seguire il corretto andamento naturale del fenomeno fisico. Successivamente a questa fase il dispositivo è pronto per iniziare a monitorare l'ambiente, considerato ora in condizione non-stazionaria, dove si vuole analizzare a run-time se il suo comportamento è compatibile con quello stimato nel modello.

Il primo livello di Change Detection avviene localmente sul dispositivo. Per questa fase viene utilizzato il Change Detection basato sull'ICI rule, come esposto in [1], la cui efficienza ed efficacia sono dimostrati dai dati sperimentali riportati in questo articolo. Questo test viene effettuato su finestre di campioni, verificando che i dati acquisiti siano caratterizzati dalla stessa distribuzione statistica dei valori di riferimento e indicando l'assenza di cambiamento.

Volendo limitare i falsi positivi a cui sono soggetti i CDT si compie una seconda fase di validazione del cambiamento impiegando test di ipotesi che verifichino la decisione. Nel caso siano note le relazioni tra più sensori, è possibile sfruttare un test multivariato che valuti il cambiamento su più sensori contemporaneamente, utilizzando il test multivariato T-quadrato di Hotelling sull'esempio di [4]. I diversi sensori coinvolti potrebbero percepire il cambiamento in modo più o meno rapido. Inoltre va considerato che ogni misura è soggetta ad una componente di rumore che ne modifica il valore. Per permettere al sistema di tollerare questo margine di errore strutturale presente in ogni dato, il cambiamento potrebbe non essere rilevato immediatamente a livello locale da tutti i sensori coinvolti. Come verifica, quando viene individuato un cambiamento da un sensore, ci si aspetta che siano coinvolti anche quelli a lui correlati e, eseguendo un test multivariato, può essere rilevato un'anomalia nei trend delle variabili coinvolte che confermi o meno l'ipotesi di cambiamento. Questa validazione del cambiamento locale non solo ha lo scopo di confermare l'ipotesi di cambamento, ma serve anche a individuare e reagire a dei guasti che hanno colpito i nodi della rete.

In aggiunta a questo meccanismo di validazione ne viene adottato uno a livello globale, facendo comunicare il dispositivo che rileva un cambiamento con quelli a lui correlati, che possono confermare il cambiamento eseguendo i test di ipotesi localmente, oppure negare un mutamento ambientale.

Va ricordato infatti che in normali condizioni operative i dispositivi sono soggetti ad una percentuale strutturale di guasti. Il framework proposto deve consentire di implementare ad esempio applicazioni fault-tolerant, che aggregando e studiando le informazioni sappiano classificare il comportamento anomalo di un dispositivo e segnalarne il guasto.

Il dispositivo mobile non è soltanto un connettore tra i diversi nodi, ma ha il ruolo di analizzare i dati per avere una visione completa dell'ambiente monitorato: grazie a queste conoscenze globali sa come trattare le informazioni che riceve da ogni specifico sensore, evitando di inviare richieste di verifica ai dispositivi non coinvolti. I nodi interessati eseguiranno i dovuti controlli, comunicando l'esito al coordinatore e riadattandosi in caso di conferma di cambiamento.

Una volta implementata la soluzione tecnologica sono state svolte varie prove sperimentali per verificare il funzionamento del sistema. È stato simulato da software un cambiamento modificando il valore acquisito da un sensore, in modo da

conoscere l'istante esatto e l'entità del cambiamento. Sono state eseguite due tipologie di prove: la prima riguardante un singolo dispositivo che acquisisce temperatura, umidità e pressione atmosferica, al quale sono stati introdotti tre tipi di guasto con diversi gradi di confidenza dell'ICI based CDT sul sensore della temperatura. Mentre il secondo test ha visto protagonista un sistema funzionante composto da due dispositivi programmati per monitorare la temperatura: in un primo caso è stato simulato un cambiamento ambientale dove entrambi i dispositivi rilevano una variazione, mentre nel secondo si è voluto rappresentare un guasto modificando l'acquisizione di uno dei due nodi. I dati raccolti confermano la capacità del sistema proposto di saper rilevare i cambiamenti nell'ambiente in relazione al grado di confidenza adottato e sapersi riadattare alle nuove condizioni.

Il lavoro prodotto da questa tesi si limita all'acquisizione e rappresentazione di un fenomeno fisico mediante un sistema informatico, per capirne l'andamento nel tempo e monitorarne il corretto funzionamento. Sviluppi e miglioramenti futuri potrebbero riguardare la struttura adottata, ponendo l'attenzione sulla limitazione del numero massimo di dispositivi connettibili utilizzando la struttura proposta. Infatti nelle specifiche del Bluetooth 4.0 si afferma che ogni dispositivo può avere un massimo di sette connessioni attive, limitando il numero massimo di nodi della rete. Una soluzione potrebbe prevedere l'utilizzo di nodi ponte che aggreghino i nodi termiali della rete e l'analisi finale eseguita da un server, con potenza elaborativa maggiore. Inoltre i comportamenti dei diversi parametri vengono ricavati con l'utilizzo di semplici modelli lineari. Si potrebbe sostituire questa elaborazione con approcci più sofisticati che migliorino la precisione delle previsioni. Infine le unità sono identificati solamente da un nome identificativo e si potrebbe apportare una maggiore sicurezza all'interno della rete tramite algoritmi di identificazione ed autenticazione per evitare intrusioni e manomissioni da parte di terzi.

1.3 Struttura della tesi

La tesi è strutturata nel modo seguente: nella prima sezione si fornirà un inquadramento sull'argomento relativo ai Cyber-Physical Systems. Nel secondo capitolo vengono esposti gli studi presenti in letteratura sui Self-adaptive Systems. Il terzo capitolo comprende l'idea che sta alla base del sistema proposto, fornendone le caratteristiche generali. Nel quarto capitolo viene illustrato il ruolo del singolo dispositivo a livello teorico, con i requisiti che deve soddisfate. Nel capitolo successivo si specifica come i dispositivi possano interagire per arrivare a permettere un'intelligenza a livello globale. Nel sesto capitolo si illustra una possibile implementazione del sistema descritto basta su schede ST Nucleo, con i risultati dei test sperimentali condotti.

Chapter 2

Analisi della Letteratura

2.1 Self-Adaptive Systems

Negli ultimi anni si sta registrando un costante aumento dei sistemi di informazione pervasivi. Questi dispositivi hanno il compito di operare in qualsiasi contesto in rete ad alte velocità di trasmissione, integrando un numero elevato di dispositivi compatibili ed eterogenei, dai sensori embedded ai Cloud servers, e stream di dati, dai dati web alle misure dei sensori in tempo reale, provenienti da un ambiente in continuo cambiamento e con risorse disponibili non stabili. Questi sistemi comprendono un'ampia gamma di applicazioni e possono avere dimensioni variabili, dal monitoraggio di piccole aree a spazi molto estesi, come Smart Cities o Smart Grid elettriche. Una strategia proposta per affrontare questo tipo di nuove sfide è l'auto-adattamento. Un Self-Adaptive System (SAS) è un sistema che si propone di adattare sé stesso in funzione dei cambiamenti dell'ambiente in cui opera. Questi sistemi sono caratterizzati dall'abilità di gestire autonomamente una o più caratteristica del proprio sistema, che prendono il nome di proprietà self-*, come self-configuration, self-healing in presenza di guasti, self-optimization e self-protection contro le minacce. Il sistema deve essere consapevole di sé stesso, cioé deve poter monitorare le proprie risorse, il proprio stato ed il proprio comportamento. I suoi comportamenti e le decisioni prese sono strettamente influenzati dal contesto in cui opera. Secondo Dey [5], per 'contesto' si può definire "ogni informazione che può essere utilizzata per caratterizzare la situazione di un'entità. Un'entità è una persona, luogo od oggetto considerata rilevante per l'interazione tra utente e applicazione, inclusi utente ed applicazione stessi".

Il sistema utilizza dei sensori per acquisire informazioni a proposito del suo contesto ed elaborare le informazioni. Nel corso degli ultimi anni i SAS sono stati oggetto di studi e ricerche, e nella letterature sono state formulate diverse tassonomie per classificarle. Tra le più significative si può notare quella proposta da Rorh et al., che propone una classificazione dei sistemi di ricerca auto-adattativi nel 2006 [18].

In questo lavoro vengono descritti i sistemi self-adaptive rispetto le dimensioni origine, attivazione, system layer, controllo distribuito e operazioni. Nel 2009 Salehie e Tahvildari [19] presentarono un lavoro di illustrazione panormanica software self-adaptive ed i possibili sviluppi della ricerca con inclusa una loro tassonomia per questo tipo di sistemi. Handte et al. [11] classificano i componenti adattativi di un'applicazione pervasiva nelle dimensioni di tempo, livello, controllo e tecnica. Salehie e Tahvildari [19] introducono le 5W + 1H questions, a cui bisogna rispondere in fase di progettazione di un sistema auto-adattativo in relazione ai requisiti specifici. Le domande sono:

1. When to adapt? Quando applicare l'adattamento?
2. Why do we have to adapt? Perché abbiamo bisogno di adattarci?
3. Where do we have to implement change? Dove dobbiamo implementare l'adattamento?
4. What kind of change is needed? Che tipo di cambiamento è necessario?
5. Who has to perform the adaptation? Chi deve eseguire l'adattamento?
6. How is the adaptation performed? Come deve essere gestito l'adattamento?

Per quanto riguarda il chi dovrà apportare il cambiamento, la natura di un SAS presuppone che non ci dovrebbe essere intervento da parte dell'utente. La risposta alla domanda "Who?" viene considerata il sistema stesso, che ha il ruolo di adattarsi alle nuove situazioni quando un comportamento si manifesta.

Altri autori hanno proposto tassonomie basate su queste domande. Di seguito esporrò quella proposta da Krupitzer et al. [13].

2.2 Tassonomia

2.2.1 When to adapt?

L'aspetto del tempo è relativo alla domanda When?. L'approccio tradizionalmente adottato [11] è quello dell'adattamento reattivo, cioè dare il via alla fase di adattamento quando un evento ne produce il bisogno, come ad esempio un cambiamento o una diminuzione delle risorse. In contrapposizione Rorh et al. [18] propongono due metodologie per pianificare il cambiamento prima che questo avvenga: l'adattamento predittivo e quello proattivo. Nel predittivo il sistema identifica il cambiamento e la necessità di adattamento prima che si verifichi, mentre il proattivo lo innesca ancora prima che il cambiamento avvenga. Il tempo viene quindi diviso in prima e dopo la necessità di adattamento. Dal punto di vista dell'utente adattamenti proattivi sono preferibili perché consentono la continuità del lavoro senza tempi di interruzione. D'altro canto gli algoritmi predittivi necessari ai proattivi sono affetti

da complicanze. Tipicamente questi algoritmi hanno una complessità implementativa elevata, sono molto dipendenti dal contesto in cui vengono applicati, e previsioni errate potrebbero creare risultati subottimi o adattamenti che risultano dannosi al sistema. Per questi motivi in alcuni casi si preferisce l'approccio reattivo. Studiando l'adattamento reattivo e proattivo applicati nel ciclo MAPE (illustrato nel paragrafo 2.3.1) si può notare che svolgono attività simili nelle fasi di monitoring, planning ed execution. Nella fase di analysis il sistema reattivo controlla che i dati monitorati stiano seguando un normale pattern, mentre in quello proattivo vengono utilizzati per capire il comportamento del sistema o lo stato dell'ambiente. I due approcci possono essere combinati in modo da usare il proattivo come principale e il reattivo in caso il primo fallisca.

Un altro aspetto che riguarda il tempo è la frequenza di monitoraggio dell'ambiente. Può essere continuo, ed in tal caso verranno eseguiti continuamente controlli sull'ambiente e le risorse. Altrimenti può essere un monitoraggio adattativo, che va a monitorare soltanto determinati parametri, ed in casi critici intensifica il processo di monitoraggio.

2.2.2 Why?

Nei sistemi analizzati l'adattamento avviene come reazione ad un cambiamento. Questo dovrà avvenire solo in presenza di fenomeni per cui il sistema è stato progettato. Si vogliono dunque identificare le ragioni del cambiamento, e rispondere alla domanda Why?, che in un sistema SAS possono riguardare uno o più dei seguenti elementi:

- un cambiamento nelle risorse tecniche, causate da un guasto o dall'introduzione di nuove reti o connessioni;
- un cambiamento nell'ambiente, cioè un cambiamento dello stato del contesto;
- un cambiamento riguardante l'utente, ovvero una variazione delle preferenze su fenomeni che si vuole decidere di monitorare.

È fondamentale capire ed identificare quali debbano essere le ragioni su cui basare l'adattamento e stabilire un adeguato processo di monitoraggio, senza il quale non potrebbe venire rilevato il cambiamento, ed il processo di adattamento stesso non potrebbe iniziare.

2.2.3 Where?

L'adattamento potrà essere implementato a diversi livelli del sistema. Si deve rispondere alla domanda Where?, cioé scegliere a che livello del SAS eseguire il cambiamento. In generale un SAS è composto da due categorie di elementi: la logica di adattamento e gli elementi gestiti [24].

Gli elementi gestiti sono composti da diversi livelli. Le risorse tecniche sono componenti hardware, controllati da un sistema software (sistema operativo), ed in caso di sistemi distribuiti da un middleware. L'applicazione può essere organizzata centralmente su singolo dispositivo, oppure distribuita in tante applicazioni che vengono eseguite su diverse unità. In questo caso verrà considerata come un'unica applicazione anche se in realtà è un insieme di applicazioni che interagiscono. Durante il funzionamento si potrebbero creare dei problemi di interferenza e dipendenze nell'utilizzo delle risorse: nasce quindi la necessità di una comunicazione e coordinazione tra tutti i componenti. La comunicazione deve essere vista sotto due prospettive: l'infrastruttura di rete, composta da schede di rete, antenna, routers e tutti i componenti fisici necessari per costruire una connessione, ed i pattern di comuicazione, cioè la logica con cui si decide di comunicare ed utilizzare l'infrastruttura, ad esempio la logica di comunicazione potrebbe essere event-based o publish/subscribe.

Un ulteriore livello a cui può agire l'adattamento è il contesto, dove l'ambiente è monitorato ed in caso di necessità vengono utilizzati degli attuatori controllati per riadattare il contesto in cui l'applicazione opera.

L'applicazione può essere progettata per adattarsi in uno o più di questi livelli. Deve essere effettuato uno studio per determinare gli obiettivi applicativi e trovare la soluzione che ottimizzi alcuni criteri, come costi, sicurezza, rapidità, etc.

2.2.4 How?

Oltre ad identificare i livelli a cui applicare l'adattamento sarà necessario definire le precise azioni da effettuare ai singoli livelli, rispondendo alla domanda "What kind of change?".

McKinley [16] distingue due diversi approcci per scrivere software adattativo: parameter adaptation e compositional adaptation. Il primo approccio consiste nel modificare il comportamento del sistema modificando i parametri di sistema. Questo approccio è semplice da adottare, ma può comportare una notevole complessità nel caso in cui i parametri siano tra loro dipendenti. Come parametro si può considerare anche il particolare algoritmo utilizzato e, quando per un componente siano stati implementati diversi, è possibile passare da uno all'altro a run-time in base alle esigenze. Non è invece contemplata la possibilità di aggiungere nuovi algoritmi o nuovi componenti al sistema durante l'esecuzione.

Il compositional adaptation permette di cambare dinamicamente algoritmo o componenti a run-time: utilizzando questo approccio diventa possibile sostituire o riparare durante l'esecuzione i componenti difettosi per evitare la perdita di performance da parte del sistema.

Una terza tecnica di adattamento non considerata da McKinley è il context-adaptation: in questo caso non è il sistema in prima persona a modificarsi, ma

è il contesto in cui l'applicazione opera ad essere adattato mediante attuatori per mantenere i valori dei parametri monitorati entro certi limiti.

2.3 Adaptation control

In un sistema ciberfisico la parte software è responsabile di gestire la logica di adattamento. Essa ha il compito di controllare il comportamento del sistema e dell'ambiente, ricevere istruzioni e comunicare con l'utente, ed all'occorrenza è responsabile di prendere decisione sull'adattamento. L'ultima domanda a cui rispondere è "How to adapt?". I due approcci proposti in letteratura per l'implementazione della logica di adattamento sono l'internal approach, che intreccia la componente di logica di adattameto con le risorse di sistema, e l'external, che mantiene ben divise le due parti per favorire la manutenzione e possibili modifiche future.

Per decidere come effettuare il cambiamento si deve stabilire la metrica che regoli il controllo del SAS, che dovrà prendere decisioni sul comportamento da adottare. In letterature sono presenti diverse tipologie di metriche utilizzabili a questo scopo, come modelli, regole, obiettivi o funzioni di utilità. In base all'analisi del problema si potrà scegliere il metodo che meglio si applica al problema specifico.

Altro aspetto è il grado di decentramento della logica di adattamento: un SAS può essere centralizzato se esiste un'unità centrale col ruolo di gestire tutte le risorse e prendere decisione sull'adattamento. È preferibile utilizzare questo approccio quando le risorse da gestire sono poche, dato che, quando il numero di queste aumenta, si potrebbero creare delle inefficienze. Converrà quindi dividere la logica di adattamento tra i diversi componenti per distribuire la complessità del problema ed aumentare l'efficienza del sistema, facendo attenzione al coordinamento delle diverse attività.

2.3.1 Logica di adattamento

Un Self-adaptive System, come già accennato, è composto da una logica di adattamento e dalle risorse gestite. Sia le risorse gestite che la parte di logica di adattamento possono a loro volta essere composte da più elementi. Nella logica di adattamento si possono distinguere le quattro funzionalità MAPE (Monitoring, Analysis, Planning e Executing), che possono sfruttare una base dei concetti o di conoscenza quando presente. Inoltre possono essere implementati metodi di apprendimento. Questi componenti di logica prendono forma nel software che gira sui diversi dispositivi, creando un legame forte tra logica e risorse gestite.

Considerando le dimensioni illustrate precedentemente su cui si basano le scelte di progettazione dell'adattamento, il tempo agisce sulla decisione di quali algoritmi utilizzare in fase di analisi, scegliendo tra un'analisi proattiva, per determinare in

anticipo l'adattamento, oppure un'azione reattiva al cambiamento. Inoltre il monitoraggio può essere adottato continuo sia in caso di adattamento reattivo o proattivo.

La ragione di adattamento può influenzare le fasi di monitoraggio, analisi e pianificazione, dato che in base al problema specifico si dovranno stabilire quali aspetti monitorare, come eseguire l'analisi e su quali politiche eseguire l'adattamento.

La fase di monitoraggio è influenzata anche dal livello scelto, in cui si va a stabilire quali elementi delle risorse gestite siano significativi in questa fase. Di conseguenza anche pianificazione ed esecuzione saranno interessati da questa dimensione.

La dimensione delle tecniche scelte riguarda la pianificazione, dove si determinano gli algoritmi da utilizzare, e la fase di esecuzione, dove viene controllata la loro esecuzione.

La quinta dimensione Adaptation Control descrive la struttura della logica, e quindi non è relazionabile con le quattro fasi del processo MAPE.

In tabella 2.1 vengono riassunte le fasi con le relative dimensioni.

	Time	**Reason**	**Level**	**Technique**
Monitoring	Continuous	What to monitor	Indication of the levels	-
Analyzing	Algorithms depend on reactive or proactive dimension	Where to analyze	-	-
Planning	-	What should be influenced by planning	Adaptation plans address these levels	Plans for performing the techniques
Executing	-	-	Execution pf the change on the levels	Execution of the techniques on different elements

Table 2.1: Dimensioni MAPE rispetto le fasi di logica di adattamento.

2.3.2 Approcci implementativi

I principali approcci implementativi presenti nella letteratura sono due: internal ed external aproach [8]. Nel primo caso la logica dei sensori, attuatori ed adattamento non è nettamente distinta dalle risorse gestite. Questo approccio comporta alcuni lati negativi, come la scarsa scalabilità, la manutenzione difficile ed una complicata gestione del testing della logica di adattamento. Questo tipo di approccio è prevalentemente applicabile ad adattamenti locali non particolarmente elaborati.

L'external approach divide nettamente la parte di logica di adattamento dalle risorse gestite. Questa soluzione risolve i problemi descritti per il central approach,

aumentando la scalabilità del sistema e permettendo in fase di manutenzione di agire soltanto sulla parte interessata. Inoltre consente un riutilizzo degli algoritmi o dei processi utilizzati dalla logica di adattamento. Nella maggior parte dei casi studiati in letteratura l'external approach produce risultati migliori rispetto all'internal.

2.3.3 Criteri di decisione

In base allo specifico problema ci saranno necessità diverse in base alle quali stabilire se reagire ad un dato evento ed in che modo pianificare le azioni necessarie. Le metriche che stabiliscono quando pianificare l'adattamento possono essere stabilite in base a modelli, regole o politiche, obiettivi od utilità. A seconda del tipo di informazione che vogliamo analizzare il sistema può essere disegnato ponendo il focus su particolari aspetti.

Nell'approccio model-based viene utilizzato un modello per rappresentare la situazione attuale del sistema e del contesto che, in seguito, viene confronta con quella desiderata. Questo conmprende informazioni riguardanti gli obiettivi, l'architettura del sistema, lo stato dell'ambiente o gli eventi che accadono. Sulla base dell'analisi del modello viene stabilito come pianificare l'adattamento.

In caso di approccio rule-based, o policy-based, viene stabilito come il sistema debba reagire nelle differenti situazioni che si registrano e come adattarsi. Queste regole solitamente vengono stabilite a design-time, e possono essere considerate come approcci non dimanici.

Gli approcci goal-base sono volti a soddisfare gli obiettivi per cui il sistema è predisposto. Durante la pianificazione il sistema deve riuscire a pianificare nel modo migliore l'adattamento, in modo da raggiungere tutti gli obiettivi che gli sono stati posti. In questo caso gli obiettivi influenzano direttamente la fase di pianificazione.

In alternativa è possibile applicare una funzione di utilità al sistema, che ne valuti le prestazioni. In questo caso il SAS dovrà essere in grado di implementare la strategia di adattamento che massimizzi tale funzione. La difficoltà di questo approccio sta nella complessità dell'individuare la funzione obiettivo adeguata ed elaborare la soluzione considerando l'incertezza presente nel calcolo dei costi e dell'utilità.

2.3.4 Grado di decentralizzazione

Solitamente i SASs sono sistemi composti da diversi elementi con la capacità di comunicare tra loro. È importante stabilire come la logica di adattamento vada centralizzata o decentralizzata, e come le funzionalità MAPE debbano essere distribuite nell'architettura. Generalmente quando si tratta il caso di struttura di controllo, viene considerato più appropriato l'approccio top-down per la gestione dell'auto-adattamento. Per questo genere di applicazioni tipicamente si predilige sviluppare un'unità centrale col compito di controllare e coordinare l'intero sistema. In modo

opposto, in caso di auto-organizzazione è preferibile un approccio bottom-up, dove le distinte unità si organizzano e coordinano autonomamente. Nel primo caso il sistema adottato utilizza il self-adaptation centralizzato, mentre nel secondo si sfrutta il self-adaptation decentralizzato. Nel caso sia necessario distribuire la logica di adattamento, i progettatori del sistema devono stabilirne le regole di interazione tra i diversi componenti.

In caso di logica di adattamento centralizzata, l'intelligenza viene implementata in un sottosistema che ha il compito di monitorare il contesto e le risorse e controllare l'adattamento. Questo tipo di architettura è meglio applicabile a sistemi con una bassa quantità di informazione da trattare ed un numero limitato di dispositivi connessi: all'aumento di questi, infatti, si potrebbe verificare la saturazione della capacità massima elaborativa del sistema di controllo centrale che farà da collo di bottiglia per l'intero sistema, limitandone le performance.

Un approccio agli antipodi del precedente è la totale decentralizzazione, dove ogni dispositivo periferico contiene tutta la propria logica di adattamento con tutte le funzionalità MAPE: in questo caso le varie parti di logica devono comunicare per integrare le informazioni e raggiungere gli obiettivi di sistema. Nel modello di riferimento proposto da IBM per Autonomic Computing [12], esistono elementi autonomi che oltre ad orchestrare la coordinazione introducono un approccio ibrido. Come distribuire le funzionalità MAPE tra i diversi livelli e ad i diversi dispositivi del sistema viene descritto da Weynes et al. [24], dove vengono proposti diversi pattern: Coordinated Control Pattern, Information Sharing Pattern, Master/Slave Pattern, Regional Planning Pattern e Hierarchical Control Pattern. I primi due menzionati cono completamente decentralizzati con interazioni, mentre i restanti tre hanno elementi centralizzati, quindi seguono un approccio ibrido.

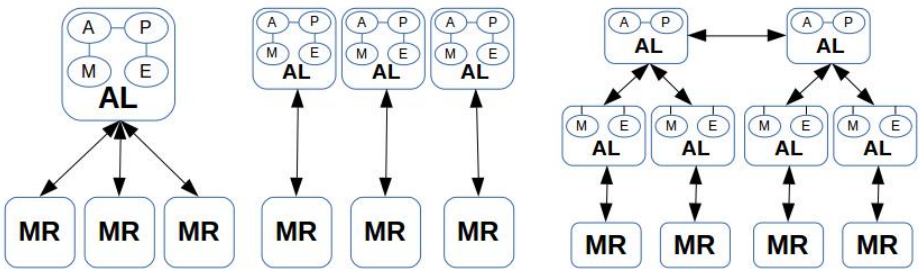

Figure 2.1: Esempi di strutture di logica di adattamento (AL = Adaptation Logic, MR = Managed Resource, M,A,P,E = funzionalità MAPE).

Per disegnare un buon SAS è necessario considerare le implicazioni relative alla distribuzione del sistema. È importante scegliere il pattern adeguato in base al

numero di sotto-sistemi, alla coordinazione necessaria per attuare le decisioni di adattamento ed al tipo di obiettivi che si vuole raggiungere.

2.4 Main Approach

Finora sono state esposte le caratteristiche dei SASs progettati secondo le funzionalità MAPE, che è l'approccio più diffuso. Di seguito vengono esposte altre tecniche proposte per gestire e migliorare l'auto-adattamento di questi sistemi.

2.4.1 Control loops

Caratteristica comune ad ogni sistema proposto al controllo dell'ambiente è il funzionamento ciclico, in cui periodicamente vengono acquisiti i nuovi dati ed in base all'analisi compiuta prese decisioni sul comportamento da assumere. Per quanto riguarda le iterazioni di controllo che ciclicamente vengono effettuate, vanno valutati diversi aspetti in fase di progettazione: sarà necessario sviluppare architetture di riferimento per i cicli di controllo, creare un catalogo delle strutture di controllo ricorrentie middleware di supporto, sviluppare tecniche di verifica e validazione del comportamento del ciclo e la sua integrazione con i comandi dell'utente.

In letteratura vengono distinti due tipologie di cicli: i cicli chiusi ed i cicli aperti. Un ciclo chiuso utilizza le informazioni sulle variabili di output monitorate per modificare le variabili di processo, mentre un ciclo aperto le modifica senza considerare l'output. I cicli chiusi possono a loro volta essere divisi in sistemi di controllo feedback, che reagiscono al cambiamento delle variabili di output, e sistemi di controllo feedforward, con l'obiettivo di anticipare gli effetti del cambiamento delle variabili di output.

I cicli di controllo feedback possono essere adattativi, modificando il controllore in risposta ai cambiamenti recepiti. I due schemi standard per questo tipo di cicli sono Model Identification Adaptive Control (MIAC) e Model Reference Adaptive Control (MRAC).

Kephart e Chess [12] introducono una struttura di controllo chiamata ciclo MAPE-K, dove le funzionalità MAPE vengono estese con una base di conoscenza condivisa. Questo approccio è diventato uno dei più diffusi nella progettazione di SASs. Viene proposto un controllore di adattamento esterno, quindi con ciclo di controllo separato dalle risorse gestite. Il tipo di adattamento è reattivo e vengono utilizzati i cicli di controllo feedback per implementare un ciclo MAPE-K.

2.4.2 External control mechanisms

Garlan et al. [9] utilizzano questo approccio dividendo le operazioni di individuazione ed applicazione del cambiamento in moduli separabili, che possono essere

analizzati, modificati, rimossi o riutilizzati da sistemi differenti. Il lavoro proposto utilizza un approccio architecture-based, per poter riutilizzare infrastrutture e meccanismi per specifici sistemi.

Schmeck, Müller-Schloer, Cakar, Mnif, e Richter [20] propongono un sistema ibrido, che utilizza sia meccanismi di controllo interni che esterni per monitorare i parametri dell'ambiente ed impostare alcuni attributi del sistema e le caratteristiche dell'ambiente. Il meccanismo di controllo è centralizzato, ma può essere sostituito da uno decentralizzato o con struttura multi-level. Il sistema può essere controllato da meccanismi interni ed esterni gestiti dal sistema, e viene aggiunta una struttura di osservazione/controllo che permette all'utente di decidere direttamente quali meccanismi adottare, limitando l'autonomia del sistema rispetto alle necessità del supervisore.

2.4.3 Component based software engineering (CBSE)

Mishra and Misra [17] utilizzano un approccio component based per disegnare un modello di SAS che autonomamente esegue l'integrazione di nuovi componenti a run-time. L'applicazione è progettata per la sostituzione di un componente con uno differente scelto da un elenco - noto a priori all'applicazione - di possibili alternativi, che sia quello con il maggior grado di somiglianza. La valutazione del componente è fatta con metodi numerici, che assegnano un punteggio ad ogni pezzo in base a quanto questo sia adatto a sostituire quello guasto.

2.4.4 Model driven

L'utilizzo di modelli che aiutino lo studio della miglior soluzione è ampiamente diffuso per i sistemi auto-adattativi, con un utilizzo in a diversi livelli in svariati settori.

Con un innovativo progetto basato sui modelli adattativi e le relative trasformazioni model-to-code, Geihs et al. [10] sviluppano una parte di middleware per un'applicazione self-adaptive operante in un determinato contesto. In questo lavoro viene esposto un progetto astratto di modello in grado di generare autonomaticamente il codice, permettendo alta flessibilità per lo sviluppo di applicazioni adattative.

Vogel, Neumann, Hildebrandt, Giese, and Becker [22] propongono un approccio model-driven per un sistema self-adaptive per l'auto-monitoraggio dell'architettura, che confronti il sistema in esecuzione con dei modelli delle diverse attività di sistema.

In caso di controllo intelligente eterogeneo, il modello viene determinato su più livelli, intrecciando informazioni provenienti da diverse fonti di conoscenza. La conoscenza acquisita è fondamentale per l'efficienza delle operazioni, e può essere utilizzata dal sistema per verificare una serie di regole o di principi.

2.4.5 Nature-inspired engineering

Un ambito di ricerca relativamente moderno, che sta attirando le attenzioni da diversi settori, è il prendere ispirazione dai comportamenti esistenti in natura per sviluppare applicazioni. In letteratura è presente una linea di ricerca che comprende diversi esempi di soluzioni software ed hardware che prendono spunto da sistemi naturali o biologici, oppure su come costruire sistemi con software auto-adattativo sulla base delle proprietà e del comportamento di sistemi di riferimento naturali e biologici.

2.4.6 Multiagent systems

Lo scopo di un approccio multiagente in un SAS è quello di disaccoppiare gli elementi che devono interagire per avere una buona percezione del contesto e soddisfare requisiti di robustezza in caso di guasto o eventi inattesi. Weyns e Georgeff [23] illustrano come, con l'ausilio di agenti disaccoppiati, si possa dare al sistema la flessibilità necessaria all'adattamento e il riutilizzo dei componenti. Gli agenti sono un insieme di oggetti con il compito di eseguire dei lavori, che nel contesto possono dover interagire con un proprio sottoinsieme ed essere isolati rispetto altri. Nelle strategie cooperative sono stati ottenuti buoni risultati in termini di obiettivi raggiunti e capacità di apprendimento combinando modelli a stati con funzioni di guadagno.

2.4.7 Feedback systems

Spesso un sistema che lavora a comportamento automatico ha al suo interno alcuni elementi a controllo feedback. Un motivo del diffuso utilizzo è dato dal dover garantire che l'output misurato mantenga un certo valore o comportamento anche in presenza di interferenze. Il concetto di feedback è quindi fondamentale per un SAS. Negli ultimi decenni sono stati sviluppati metodi di controllo sempre più improntati su questo approccio ed ai modelli dinamici. In Magee & Kramer, 1996 [14], viene analizzata la struttura dinamica nelle architetture software, formando le fondamenta per successivi lavori di ricerca. I feedback ciclici hanno avuto un ruolo fondamentale nei processi di gestione necessari per l'evoluzione del programma.

L'approccio MAPE appartiene alla famiglia dei feedback systems, ed è quello che ha avuto maggiore successo nell'ambito dei SASs.

2.4.8 Learning approach

Questo tipo di approccio si propone di ottimizzare continuamente la sua organizzazione ed i suoi parametri allo scopo di diventare più efficiente in termini di costi e di performance. In Unity [21] l'auto-organizzazione viene effettuata con l'aiuto di un arbitro centralizzato che calcoli le allocazioni ottime delle risorse sulla base

di funzioni di utilità, ridefinite con l'apprendimento di nuove informazioni durante il corso dell'esecuzione. Fisch et al. [7] hanno proposto un approccio collaborativo basato sullo scambio della conoscenza. Questa è utilizzata per generare delle regole, ed ogni volta che un agente scopre una nuova regola la propaga ed anche gli altri agenti ne possono usufruire. L'approccio collaborative reinforcement learning viene proposto anche in Dowling et al. [6] per un sistema di componenti decentralizzati e coordinati auto-adattativi.

Prothmann et al. utilizzano il padadigma di evolutionary programming per l'apprendimento. In un contesto di controllo viene integrata un'architettura di osservatore/controllore capace di attivare o disabilitare l'apprendimento. Un classificatore responsabile dell'apprendimento online è combinato con un algoritmo di apprendimento evolutistico quando offline.

In letteratura sono anche presenti casi in cui l'apprendimento agisce sulla modifica di parametri del sistema o di algoritmi utilizzati.

Chapter 3

Il sistema proposto: idea generale

In questo capitolo si vuole dare un descrizione generale del sistema proposto, che verrà dotato di dispositivi intelligenti che gli permettano di imparare il comportamento del fenomeno fisico monitorato senza avere informazioni pregresse che ne descrivano le caratteristiche. Il sistema deve essere autonomo, in grado di rilevare i cambiamenti a livello di sensore e, sfruttando le informazioni acquisite, di determinare se l'ambiente abbia subito una modifica, rilevata coerentemente da tutto il sistema, oppure un guasto determina un comportamento anomalo.

3.1 L'architettura del sistema

Data la natura del problema di dover monitorare un ambiente da molteplici posizioni sorge la necessità di avere distribuiti sul territorio diverse unità volte al controllo dei parametri interessati. Questi dispositivi devono essere in grado di acquisire le misurazioni ambientali tramite i sensori di cui è equipaggiato, elaborare queste informazioni ed all'uopo comunicare con gli altri.

Ogni singola unità dovrà essere composta da tre moduli:

- un modulo principale, che svolge i calcoli e le analisi sui dati raccolti e regola il funzionamento del dispositivo;
- un modulo sensoristico, che integrato al primo lo interfacci all'ambiente permettendogli di acquisire i dati attraverso i diversi sensori con cui è equipaggiato;
- un modulo di connettività, che permetta all'unità di poter scambiare dati o informazioni col resto del sistema.

Le singole unità così composte saranno i mattoncini su cui basare il funzionamento dell'intero sistema. Questi dispositivi devono essere in grado di integrarsi all'ambiente ed impararne il comportamento per saperne rilevare le anomalie.

L'applicazione che si vuole progettare non è strettamente legata all'architettura della rete, ma, come noto, le diverse topologie di rete applicabili hanno diverse caratteristiche, e dovrà essere valutata in fase di design quale di queste soddisfa maggiormente i requisiti applicativi. La rete potrà quindi assumere una struttura a stella, a mash, totalmente connessa, ed altre topologie di rete esistenti. Deciso quale struttura assegnare alla rete ed individuati i compiti elaborativi da distribuire ad ogni componente, va stabilito che genere di informazioni trasmettere. Il sistema che si vuole progettare deve risultare robusto ed allo stesso tempo elastico. Infatti il numero dei nodi operativi potrebbe variare nel tempo: si potrebbe verificare la necessità di estendere la rete introducendo nuovi sensori, oppure di dovere far fronte alla momentanea inagibilità di uno di essi. La rete deve essere dinamica e sapersi riadattare in relazione ai nodi disponibili. In questo scenario, dove la scalabilità ha un ruolo fondamentale, va progettata la soluzione che garantisca il miglior funzionamento anche nel caso pessimo, cioè quando la rete sarà molto popolosa e sarà presente una grande mole di informazione da elaborare.

Una volta deciso come impostare la struttura del sistema, vanno ben distinti i compiti elaborativi che andranno svolti a livello di singolo dispositivo da quelli che necessitano invece di una visione globale con una maggiore quantità di informazioni.

I nodi non avranno soltanto il compito di campionare dati dall'ambiente ed inviarli ad un elaboratore, ma si vuole sfruttare la loro potenza elaborativa rendendoli sensori intelligenti. Ogni dispositivo sarà programmato per acquisire dati e su questi compiere un'analisi attraverso test statistici con l'obiettivo di verificare la presenza di cambiamenti nella porzione di ambiente da lui monitorata. Lo scopo è quello di avere una rete formata da tanti sensori intelligenti coordinati tra loro, che siano in grado di compiere autonomamente calcoli ed analisi sui dati ambientali, diminuendo il numero delle comunicazioni necessarie per la trasmissione di tutti i dati, e grazie all'interazione con gli altri nodi sfruttare l'interoperabilità del sistema per creare un'intelligenza globale che conosca ed utilizzi le relazioni tra essi per lo studio del sistema distribuito.

Per ragioni legate al problema, discusse al paragrafo 5.1, è stato scelto di creare una rete che implementi una topologia a stella. Nonostante questa scelta le premesse fatte e la logica applicativa descritta in seguito valgono per qualsiasi architettura adottata.

Dopo aver esposto la struttura che assume il sistema, e fatta la distinzione tra singola unità ed intero sistema, verranno illustrate le capacità che dovranno avere i dispositivi per garantire un'intelligenza di adattamento e che comportamento deve assumere l'intero sistema.

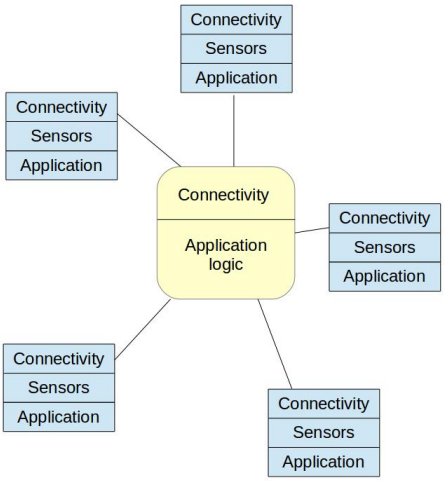

Figure 3.1: La rete di sensori con topologia a stella.

3.2 Funzionamento del sistema

Il primo passo per il rilevamento di un cambiamento deve essere compiuto dal singolo dispositivo progettato al controllo dei parametri fisici. Questo deve essere in grado di acquisire i dati, capirne le caratteristiche e sapere distinguere il comportamento corretto del fenomeno fisico rispetto l'introduzione di eventi che ne causano una variazione.

Lo schema generale che deve seguire l'applicazione è rappresentato in Figura 3.2. Partendo dal basso, la figura illustra come viene identificato e creato un modello che rappresenti il comportamento del parametro monitorato a partire da dati precedentemente acquisiti. Successivamente è possibile valutare quanto l'approssimazione dell'ambiente che abbiamo precedentemente calcolato sia fedele rispetto ai suoi valori reali. Una volta che si sono calcolati un numero di residui di riferimento sufficientemente grande da essere rappresentativo, il sistema ha acquisito la conoscenza relativa al comportamento atteso, sia dell'ambiente che del sistema informatico stesso, e viene creata una base dei concetti contenente le regole conosciute dal sistema che identificano il pattern attribuito al comportamento ambientale.

Finita la fase di apprendimento il sistema è operativo. In questa fase ogni unità del sistema acquisisce ciclicamente dati dai sensori ed esegue l'analisi su essi, verificando che le il comportamento previsto dalle regole acquisite rispecchi il reale andamento acquisito. Dopo aver campionato e salvato i nuovi valori sensoriali acquisiti ed aver calcolato il residuo rispetto al modello, i dispositivi eseguono un'analisi sui dati, per verificare che l'ambiente si stia comportando come previsto dal sistema, oppure abbia cambiato il proprio comportamento, quindi il modello conosciuto risulta obsoleto e deve essere aggiornato.

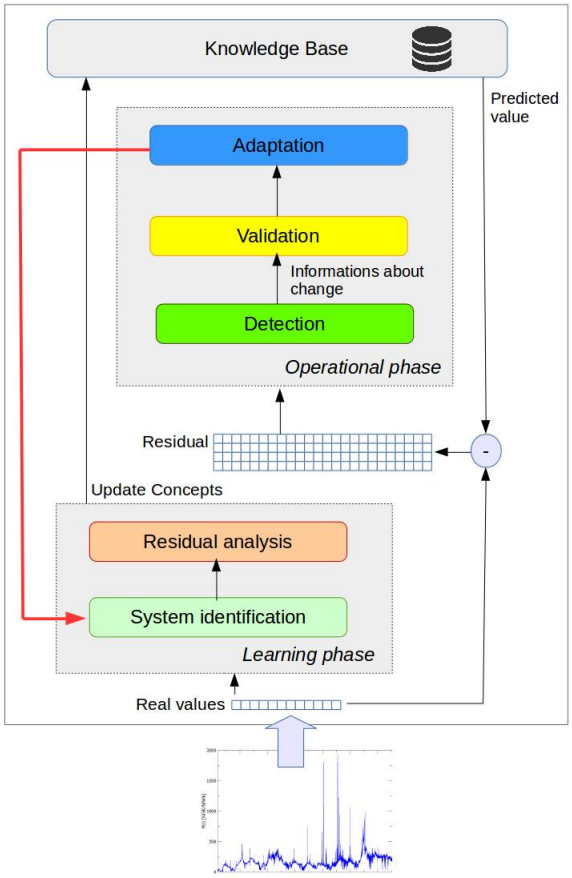

Figure 3.2: Schema del funzionamento del sistema: processo di addestramento ed analisi operativa del sistema.

A questo scopo esistono diversi tipi di test che si possono implementare, come ad esempio considerare un cambiamento quando il valore previsto e quello reale si discostano eccessivamente, oppure analisi più accurate, come i Change Detection Test, che identificano se i valori assunti da una variabili seguano un comportamento atteso, oppure se all'interno di questi valori si possa identificare un istante in cui considerare una variazione rispetto ai dati precedenti. Tipicamente i problemi relativi ai Change Detection Test sono i falsi positivi, falsi negativi ed il ritardo di individuazione di cambiamento. Infatti a causa del rumore che modifica il valore dei dati il test potrebbe fallire. Inoltre potrebbe accadere che al verificarsi di un cambiamento, questo non venga subito rilevato, ma siano necessarie ulteriori misurazioni per verificare che la sua distribuzione statistica abbia caratteristiche diverse da quelle di riferimento.

In caso il test che si è scelto di applicare sul dispositivo dia esito positivo si procede con una validazione del risultato, atta allo scopo di mitigare la possibilità di rilevare e comunicare un falso positivo. Viene eseguito un test di validazione del cambiamento che dia conferma del risultato elaborato, eseguendo ad esempio un test

di ipotesi.

Per l'analisi di cambiamento non si è usato direttamente il test di ipotesi poiché questo tipo di test è soggetto ad una percentuale di errore strutturale, ed eseguendolo continuamente si sarebbe incappati inevitabilmente in decisioni errate. Utilizzandolo invece una volta sola come conferma del CDT il suo risultato sarà esatto con una probabilità molto alta.

Quando il test eseguito sull'ambiente rileva che i concetti appresi non sono più rappresentativi del fenomeno fisico, il sistema dovrà re-impararne il comportamento, rieseguendo la fase di campionamento di una quantità di valori rappresentativa, calcolandone il nuovo modello ed aggiornando la base dei concetti con le nuove nozioni acquisite.

A livello di sistema si avranno tante unità che svolgono la procedura sopra esposta. L'idea è quella di farle interagire in modo da poter sfruttare una maggior quantità di dati, integrando le informazioni così da generare un'intelligenza globale a livello di rete per una visione di tutto l'insieme.

A questo punto i dispositivi di monitoraggio sono attivi e stanno operando per il controllo di eventuali cambiamenti. Per poterli connettere e capirne le possibili relazioni che esistono tra i diversi dispositivi in modo da permettere la creazione di un'intelligenza di più alto livello sono necessarie 3 fasi: connessione dei diversi nodi, creazione del grafo delle dipendenze per tutti i sensori presenti nel sistema e sincronizzazione di tutte le unità.

La fase di connessione è strettamente legata alla tecnologia che si è scelto di utilizzare. Ogni dispositivo deve essere visibile agli altri e deve poter comunicare potenzialmente con tutti. Nella topologia a stella adottata, il coordinatore centrale si attiva in un momento successivo rispetto ai nodi, quando questi stanno già monitorando l'ambiente a regime. Eseguendo una scansione ricerca i dispositivi raggiungibili nel suo raggio di connessione, ed individuati i nodi della rete stabilisce la connessione con loro. Finita questa fase tutti i dispositivi sono connessi con il coordinatore, che può fungere da ponte tra tutte le coppie di dispositivi, che risultano quindi concettualmente tutti collegati tra loro.

Come già detto, il vantaggio che si vuole avere dal sistema ciberfisico è quello di poter sfruttare la maggior quantità di informazioni possibile. Diventa cruciale sapere quali dati possano essere considerati in relazione tra loro e se esistano dipendenze più o meno forti tra i diversi sensori. Se due sensori sono impiegati per monitorare uno stesso parametro, ci si aspetta che i dati che questi elaborano siano coerenti, e quindi il cambiamento di uno possa essere verificato dal cambiamento dell'altro. Questa fase è molto legata alla topologia di rete che si è scelto di adottare. Il sistema deve essere in grado di confrontare ogni coppia di sensori presenti nella rete per capire i collegamenti rilevanti, e notificare ad ogni dispositivo le eventuali correlazioni tra i propri sensori, cioè i dati da lui più facilmente usufruibili. Nella soluzione proposta

il dispositivo centrale raccoglie i dati di tutti i sensori, e li confronta a coppie per capirne le relazioni.

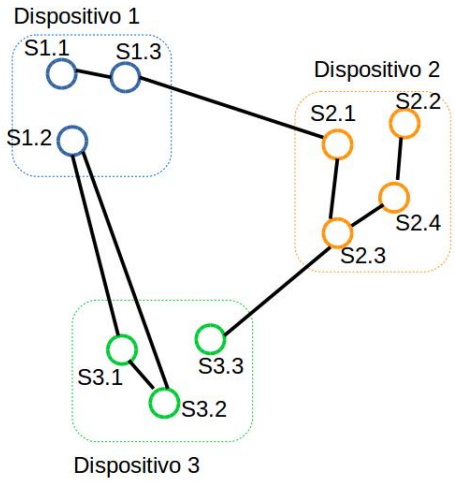

Figure 3.3: Esempio di grafo delle dipendenze tra sensori locali e tra diverse board.

L'ultimo problema che si deve affrontare è quello della sincronizzazione, fondamentale per far lavorare all'unisono i diversi dispositivi, inizialmente privi di una concezione temporale globale. I dispositivi possono essere eterogenei, misurare parametri diversi e quindi avere frequenze di funzionamento diverse. Sarà necessario fornire a tutti i nodi delle informazioni che gli permettano di conoscere il tempo di sistema ed essere tra loro sincronizzati.

Questo passaggio è fondamentale, perché quando un dispositivo rileva un cambiamento e ne chiede una validazione globale, richiede attraverso il coordinatore di interrogare i dispositivi che potrebbero essere coinvolti nel cambiamento, i quali acquisendo dei segnali con caratteristiche diverse potrebbe non aver ancora riconosciuto il cambiamento. Fornendo nell'istante di connessione la nozione di tempo iniziale si stabilisce una base comune a tutti i dispositivi su cui poter calcolare l'istante di tempo a cui sia avvenuto un cambiamento.

Una volta che tutti i dispositivi sono stati connessi e si è creata la base dei concetti, che questa volta sarà quella riguardante il comportamento globale del sistema, il sistema è inizializzato e potrà iniziare a sfruttare le informazioni provenienti da tutti i dispositivi. L'idea è quella di aggiungere una validazione a livello globale, che sfrutti le relazioni precedentemente calcolate. Infatti se due sensori monitorano parametri che sono in relazione tra loro, ci si aspetta che al variare di uno corrisponda una variazione anche dell'altro. Ad esempio avendo due dispositivi posti a monitorare la temperatura di un ambiente ci si aspetta che abbiano comportamento simile. Con l'aggiunta di questo livello, ogni volta che un dispositivo rileva un cambi-

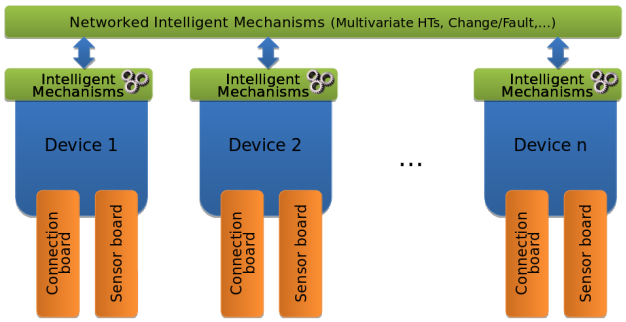

Figure 3.4: Schema di funzionamento del sistema: i dispositivi con i rispettivi moduli acquisiscono i dati e ne estraggono informazioni, che poi condividono ad un livello superiore allo scopo di trasmetterla all'utente o attivare attuatori controllati.

amento questo chiede conferma a quelli che gli sono risultati correlati nel momento di creazione del grafo delle dipendenze. Questi possono confermare o negare il cambiamento, ed integrando le diverse risposte si possono ricavare informazioni riguardanti all'intero sistema oltre che sulla singola unità. Ad esempio si può analizzare se il sistema stia evolvendo correttamente come atteso, oppure se il comportamento anomalo di un solo sensore possa essere considerato come un guasto oppure come evento riguardante solo la porzione di ambiente da lui monitorata, oppure ancora se l'ambiente abbia modificato le regole che si erano imparate, quindi il grafo delle dipendenze non è più valido e si dovrà procedere con un ricalcolo delle dipendenze

3.3 Esempio applicativo

Per chiarezza si consideri l'esempio riportato in figura 5.6. L'ambiente viene monitorato da tre dispositivi con i loro sensori utilizzando la topologia a stella. Al momento della connessione al coordinatore il sistema stabilisce le relazioni presenti tra i sensori della rete.

Dal dispositivo 1 viene notificato il cambiamento locale del parametro monitorato dal sensore 3 al tempo T*. Queste informazioni vengono inviate al coordinatore, che ha la visione completa del grafo delle dipendenze tra i sensori di tutti i dispositivi presenti, quindi può sapere quali altri sensori dovrebbero rilevare un cambiamento in relazione al sensore S1.3. Dal grafo emerge che questo sensore è correlato al sensore S2.1. Il dispositivo notifica allora al dispositivo 2 che all'istante di tempo T* al fenomeno fisico monitorato dal sensore 1 potrebbe essere in corso un cambiamento che il sensore non è stato ancora in grado di rilevare. Il dispositivo 2 riceve il messaggio ed esegue un test multivariato sui data-trend dei sensori a questo correlato, 3 e 4. Il test restituisce un esito alla conferma di cambiamento, che viene trasmesso al coordinatore. Questo era l'unico dispositivo interessato dal cambiamento, quindi non dovrà attendere risposta da altri. Sulla base di questa risposta il dispositivo centrale

può compiere un'analisi globale e notificare ai dispositivi 1 e 2, che erano quelli interessati dal cambiamento, eventuali decisioni. In caso di cambiamento seguirà la fase di training a partire dall'istante T*, come descritto precedentemente.

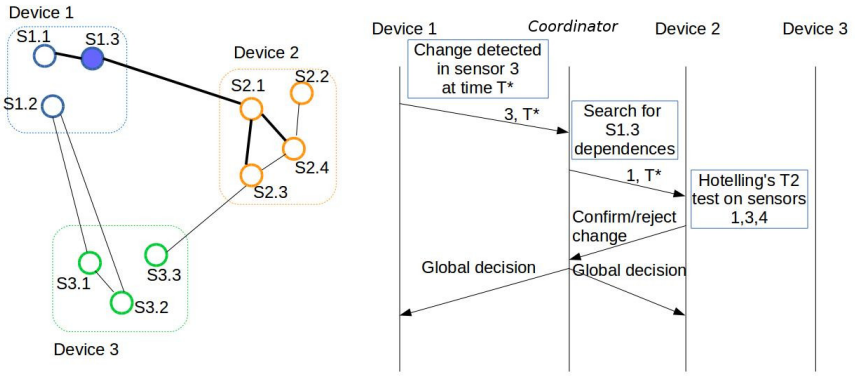

Figure 3.5: Esempio di funzionamento della rete in caso di cambiamento locale.

Chapter 4

Intelligenza a livello di dispositivo

Le singole unità progettate saranno i mattoncini che andranno a contribuire al funzionamento dell'intero sistema. Come già detto, si vuole dotare questi dispositivi di un'intelligenza a livello locale. Il loro scopo sarà quello di poter campionare i dati dall'ambiente, costruire un modello del fenomeno fisico e compiere un'analisi sui dati disponibili. Il comportamento che si vuole impartire a questi dispositivi è descritto in Figura 4.1.

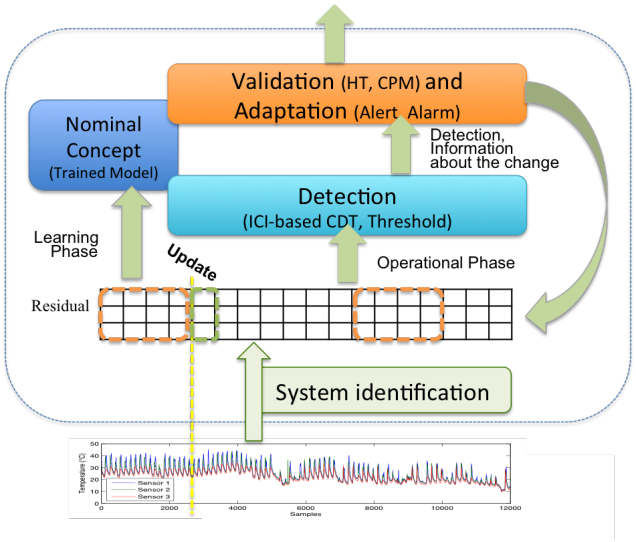

Figure 4.1: Comportamento di ogni dispositivo: iniziale acquisizione dei dati, a cui segue un'analisi per l'identificazione del modello e l'aggiornamento dei concetti conosciuti. In caso di rilevazione di un evento si procede con la validazione e l'eventuale riaddestramento.

In questo capitolo verranno esposte le soluzioni ideate per ciascuna fase. Inizialmente il dispositivo acquisisce per ogni sensore s_i un insieme di dimensione N di misure di riferimento $X_i = \{x_i(1), x_i(2), ..., x_i(N)\}$ del fenomeno in condizioni stazionarie. Da questi valori è in grado di identificare un modello che ne descriva il comportamento e permetta sulla base degli ultimi valori misurati di effettuare una

stima sul valore futuro $\hat{x}_i(n+1) = f(x_i(n), x_i(n-1), ..., x_i(n-k))$. Successivamente si analizza i residui relativi al modello acquisito, definiti come $r_i(t) = x_i(t) - \hat{x}_i(t)$, e da questi configurare il Change Detection Test ed i test di ipotesi per la validazione del cambiamento rilevato. Durante l'esecuzione l'unità dovrà acquisire nuovi campioni ed eseguire il CDT per verificare la validità del proprio modello rispetto al reale comportamento dell'ambiente. Infine, quando un cambiamento viene rilevato e validato dai test, il dispositivo deve essere in grado di riconfigurarsi per adattarsi alle nuove condizioni ambientali successive al cambiamento.

4.1 System identification e modello predittivo

La prima conoscenza che il dispositivo deve acquisire riguarda l'andamento del fenomeno fisico nel tempo, per poter prevedere come questo evolverà sulla base di un insieme di valori precedentemente acquisiti che siano caratteristici del comportamento generale. Ogni microcontrollore deve elaborare un modello corrispondente ad ogni fenomeno fisico che i suoi sensori stanno monitorando. In letteratura sono presenti molteplici approcci atti a questo scopo. Per scegliere l'algoritmo da utilizzare bisogna considerare l'ambito applicativo nel quale deve operare: esso deve saper effettuare una previsione sul prossimo valore del parametro ma senza richiedere un'elevata complessità computazionale. In questa soluzione si è scelto di utilizzare un modello lineare autoregressivo, utile a rappresentare processi che variano nel tempo, ipotizzando che l'elemento successivo della serie dipenda linearmente dagli elementi precedenti. In alternativa si sarebbero potuti usare allo stesso scopo altri tipi di modelli di tipo lineare, come i modelli ARMA (autoregressive moving average) oppure modelli OE (Output-Error). Si potrebbero scegliere anche algoritmi più complessi che sfruttino un'identificazione del modello non lineare, ad esempio utilizzando Artificial Neural Network.

4.1.1 Modello Autoregressivo del fenomeno fisico

Il modello AR(p), dove p ne rappresenta l'ordine, si presenta così:

$$z_t = \phi_1 z_{t-1} + \phi_2 z_{t-2} + ... + \phi_p z_{t-p} + \varepsilon$$

dove i parametri $\phi_1, \phi_2, ..., \phi_p$ rappresentano i coefficienti lineari della variabile causale z_t rispetto ai suoi valori passati, ed ε è il rumore del segnale, che segue una distribuzione gaussiana. Lo scopo di questo modello è quello di fornire una previsione del valore corrente della variabile z_t sulla base dei p valori precedenti della serie.

L'algoritmo Il metodo utilizzato per determinare il modello è quello dei minimi quadrati (Least Squares) che permette di trovare una funzione, rappresentata da una

curva di regressione lineare, che si avvicini il più possibile ad un insieme di dati, che in questo caso saranno le misure precedenti. In particolare la funzione trovata deve essere quella che minimizza la somma dei quadrati delle distanze tra i valori osservati e quelli della curva che rappresenta il modello.

L'algoritmo utilizzato dal microcontrollore dovrà permettergli di calcolare i valori $\phi_1, \phi_2, ..., \phi_p$ sulla base delle ultime n misurazioni acquisite. Questi valori devono essere tali per cui il valore della sommatoria $S = \sum_{i=1}^{n} r_i^2$ sia minimo. Il modello sarà tanto più preciso quanto maggiore risulterà il numero p dei coefficienti sufficienti ad approssimare il fenomeno, in quanto esso potrà adattarsi meglio all'andamento temporale del segnale e al crescere del numero n di valori a disposizione.

4.2 Change Detection

Dopo aver individuato il modello adatto a rappresentare l'andamento temporale del fenomeno nel tempo, si vuole procedere istruendo il dispositivo a rilevare i cambiamenti ambientali. In questo caso si vuole monitorare che il comportamento dei fenomeni fisici evolva come atteso. Finché il fenomeno monitorato mantiene il suo valore 'vicino' a quello del modello significa che siamo in grado di prevedere correttamente come questo sta evolvendo nel tempo. Per fare questo vengono acquisiti i valori $x_i(t)$ rilevati dal sensore s_i e confrontati con il valore $\hat{x}_i(t)$ previsto dal modello, calcolando l'insieme dei residui di riferimento $R_i^{ref} = \{r_i(1), r_i(2), ..., r_i(N)\}$. Questo insieme fornisce la misura di quanto il modello descriva adeguatamente la caratteristica ambientale monitorata dal sensore, le cui caratteristiche devono rimanere costanti durante il controllo. Per questa ragione il dispositivo esegue un Change Detection Test sul residuo r_i, verificando che il valore previsto sia adeguatamente simile a quello reale. Nello specifico si è scelto di utilizzare il Change Detection Test basato sull'Intersection of Confidence Interval (ICI), che garantisce buone prestazioni con risultati affidabili, come conferma la campagna sperimentale su un suo impiego in [4]. In alternativa è possibile utilizzare altri tipi di Change Detection Test presenti in letteratura.

Questo tipo di test ha bisogno di un periodo di apprendimento, ricevendo dei campioni di riferimento per valutare quanto e come il modello si discosti dal valore reale. In fase di esecuzione il test partirà da questi dati per verificare che l'errore stia mantenendo la propria conformità.

4.2.1 ICI based Change Detection Test

Ipotizzando che il trend di misurazioni sia stato acquisito in condizioni stazionarie, e con esso sia stato calcolato il relativo modello AR, siamo ora interessati a monitorare il segnale per rilevare eventuali cambiamenti nell'ambiente. Sfruttando il

modello acquisito l'idea è quella di monitorare l'errore, cioè la differenza tra valore atteso e valore reale, e non direttamente il segnale. Così facendo non verrà notificato un cambiamento ogni qual volta si modifichi il valore di un parametro misurato, bensì ogni volta che questo si allontanerà dal modello che lo descrive. Quando questo accade significa che il modello non è più valido per l'ambiente in cui opera, e si dovrà valutare il cambiamento e la rielaborazione di un nuovo modello che descriva l'ambiente in seguito al cambiamento.

Anche in questo caso la letteratura propone numerose soluzioni al problema. La soluzione proposta è un'applicazione di Change Detection basata sulla Intersection of Confidence Interval rule per monitorare l'evoluzione del processo.

L'ICI-CDT opera su una sequenza di valori $z \subseteq \mathbb{R}$ soggetti a rumore, campionati ad una determinata frequenza e che seguono una distribuzione Gaussiana con media $\mu(t)$ e deviazione standard σ:

$$z(t) \sim N(\mu(t), \sigma^2) \qquad , t \subseteq W$$

dove t rappresenta l'istante di tempo e W la griglia dei campionamenti uniformemente spaziati.

Per ogni $t \subseteq W$ l'ICI rule identifica una finestra ottima$U_{i^+}(t)$ e sostituisce a $z(t)$ con una stima $\hat{\mu}$ di μ ottenuta ottenuti tramite una regolarizzazione dei valori appartenenti alla finestra ottima $\{U_i(t) \quad i = 1, 2, ..., L\}$.

Sia $\{\hat{\mu}_i(t) \quad i = 1, 2, ..., L\}$ la sezuenza delle stime di $\hat{\mu}(t)$ valutate su $U_i(t)$ e $\{\sigma_i(t) \quad i = 1, 2, ..., L\}$ le relative deviazioni standard.

Sia I_i l'intervallo di confidenza dell'elemento μ_i definito come

$$I_i = [\mu_i(t) - \Gamma\sigma(t), \mu_i(t) + \Gamma\sigma(t)]$$

dove $\Gamma > 0$ è un parametro fissato. L'algoritmo rileverà la finestra $U_{i^+}(t)$ come quella corrispondente al maggiore indice j per cui l'intersezione $\bigcap_{i=1}^{j} I_i \neq \varnothing$.

L'ICI rule seleziona la finestra adattativa tra aventi estremo sinistro $t = 1$. Allora l'insieme delle finestre per analizzare le osservazioni all'istante T è $\{[1, T_0], ..., [1, T]\}$.

Infine per ogni finestra viene stimata la caratteristica con il polinomio di regressione di ordine 0. Questa idea è motivata dal fatto che la distribuzione considerata in un processo stazionario sarà caratterizzata da media e deviazione standard uniche. Quando questa ipotesi di stazionarietà cade, le finestre selezionate dall'ICI-rule possono essere distribuite come Gaussiane differenti. Sia $F^k(t) \qquad t = 1, ..., T_0$ la sequenza dei valori assunti dalla caratteristica F^k all'interno del training set. Allora $\hat{\mu}^k_{T_0} = \sum_{t=1}^{T_0} \frac{F^k(t)}{T_0}$ e $\hat{\sigma}^k_{T_0} = \sqrt{\sum_{t=1}^{T_0} \frac{\left(F^k(t) - \hat{\mu}^k_{T_0}\right)^2}{T_0 - 1}}$

rappresentano una stima del valore attesimo della k-esima caratteristica e la sua deviazione standard mentre X è nello stato stazionario iniziale. Con questi valori

può essere stimato il primo intervallo di confidenza $I^k_{T_0}$.

Durante la fase operazionale, per ogni $t > T$ il test calola le caratteristiche appartenenti all'intervallo $[0, T]$. Il test agisce verificando se questi valori possono essere stati generati dalla stessa distribuzione Gaussiana.

Infine l'ICI Change Detection Test rileva il cambiamento quando $\bigcap_{t=T_0}^{T} I^k_t, T > T_0$ è vuoto.

4.2.1.1 Algoritmo

Per l'esecuzione del test l'insieme dei dati osservati $X(t)$ viene segmentato in sottosequenze disgiunte di lunghezza v: $Y_s = \{X(t), (v-1)s \leq t < vs\}$, dove $s \subseteq \mathbb{N}$ è l'indice della sottosequenza. Questo test valuta le caratteristiche solo su queste sottosequenze, cioè con un'accuratezza di $\pm v/2$. Viene applicata l'ICI rule per i sottoinsiemi $\{[1, S_0], ..., [1, T/v]\}$, dove $S_0 = T_0/v$.

Il primo parametro calcolato è la media di ogni sottoinsieme Y_s:

$$M(s) = \frac{1}{v} \sum_{t=(v-1)s}^{vs} X(t)$$

La seconda caratteristica è la deviazione standard di ogni sottosequenza, quindi

$$S(s) = \sum_{t=(v-1)s}^{vs} (X(t) - M(s))^2$$

Va notato che $S(s)$ non è distribuita come Gaussiana quando i campioni $X(t)$ provengono da una distribuzione arbitraria. Per ovviare a questo problema viene effettuata una trasformazione che deriva una caratteristica relativa alla varianza di $X(t)$ e che segue una distribuzione Gaussiana. Questa trasformazione ha forma

$T(S(s)) = \left(\frac{S(s)}{v-1}\right)^{h_0}$, dove h_0 viene ricavato dalle osservazioni del Training Set, come illustrato in [15].

Chiamiamo questa seconda caratteristica $V(s)$.

Il seguente algoritmo descrive i passi necessari per l'esecuzione del test. Le linee 1-11 rappresentano la fase di Training, dove vengono stimati tutti i parametri necessari al test. Le linee 14-21 nel loop rappresentano la fase di esecuzione, che viene eseguita ogni volta che sono disponibili le v osservazioni successive.

4.3 Validazione

Per mitigare eventuali falsi positivi rilevati dal CDT che andrebbero ad incidere sulle prestazioni del sistema, si vuole validare a livello locale il cambiamento. Dovendo validare un cambiamento che ha riguardato il comportamento del fenomeno

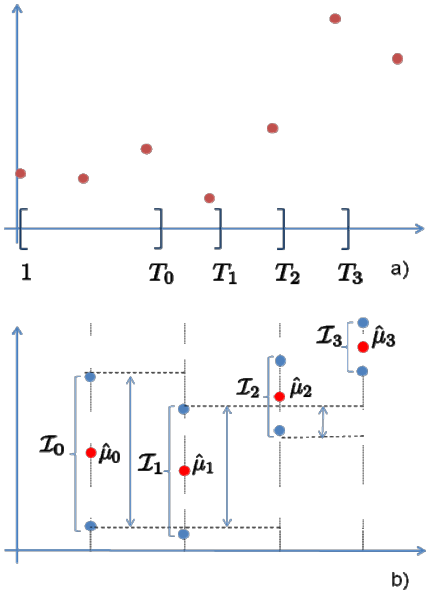

Figure 4.2: Esempio di applicazione di ICI rule su segnale campionato e suddiviso in intervalli.

fisico valgono ancora le considerazioni illustrate per il CDT. Ha senso per questo eseguire un'analisi sul residuo, che fornisce la misura dell'adeguatezza del modello. In fase di training si calcolano le caratteristiche del campione di errori di riferimento R_i^{ref}, ricavandone media e varianza. Per compiere questa analisi esistono diversi tipi di test che si potrebbero utilizzare. Per questo scopo è stato utilizzato il paired t-test, che esegue una valutazione su cambiamenti di valori attesi per variabili con distribuzione gaussiana, che permette di decidere in relazione alla rumorosità del sensore se effettivamente il valore acquisito si è discostato da quello previsto.

4.3.1 Test di validazione: paired t-test

Una volta rilevato il cambiamento si dovrà validare questa scelta. Il sistema ha riscontrato una variazione dei residui con il Change Detection Test, che afferma che i valori campionati si stiano discostando dal valore previsto dal modello. Va vericato che effettivamente le caratteristiche attuali del segnale si stiano allontanando da quelle di riferimento. Il test che si può effettuare per verificare se due insiemi di dati che seguono una distribuzione normale siano significativamente differenti è il Paired t-test.

In questo test di ipotesi viene considerato l'insieme del residuo di riferimento R_i^{ref} di dimensione n_1, con valore medio $\bar{R}_1$ e varianza $S^2_{R_1}$. Il CDT identifica un insieme R_2 contenente n_2 valori di residuo in cui si è rilevato un cambiamento della distribuzione. L'insieme analizzato è caratterizzato da valore medio $\bar{R}_2$ e varianza $S^2_{R_2}$. Si vuole verificare che i valori della del secondo insieme siano compatibili con quelli dell'insieme di riferimento. L'ipotesi nulla che si vuole verificare con il

test di ipotesi è $\bar{R}_1 = \bar{R}_2$. Il test può avere formule diverse a seconda del tipo di campioni che si stanno esaminando. Nel nostro caso il test deve essere applicato tra un intervallo di riferimento e l'intervallo degli ultimi campioni successivi all'istante di cambiamento. I due intervalli avranno quindi dimensione e varianza potenzialmente diversi. Il valore del paired t-test relativo sarà:

$$t = \frac{\overline{R}_1 - \overline{R}_2}{s_{R_1R_2}\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}$$

dove $s_{R_1R_2} = \sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}$ è la deviazione standard dei due insiemi.

Confrontando il valore ottenuto con il valore della t-student per $n_1 + n_2 - 2$ gradi di libertà al livello di significatività α si può decidere se accettare o rifiutare l'ipotesi nulla in favore dell'ipotesi alternativa.

In caso di rifiuto dell'ipotesi nulla sarà validato il cambiamento.

4.4 Funzionamento del dispositivo

Il singolo dispositivo è progettato per essere anche autonomamente in grado di compiere il processo di adattamento senza interagire con gli altri. Partendo dal ruolo che il dispositivo assume come descritto nel capitolo precedente, si vanno ad identificare le fasi di funzionamento che integrino le soluzioni sopra esposte.

4.4.1 Attivazione e training dei dispositivi

Le unità sono disegnate per essere operative a partire dalla loro accensione. Una volta posizionate nel luogo di cui si desidera monitorarne il comportamento possono essere accese. Questi dispositivi sono modellati come una macchina a stati. Inizialmente sono in stato di training, e in questa prima fase devono configurarsi per essere in grado di monitorare correttamente i parametri fisici per cui sono progettati. L'analisi si divide in due parti: la prima relativa allo studio del comportamento del fenomeno fisico e la seconda che ne valuta le caratteristiche.

4.4.1.1 Modello del fenomeno fisico

Nella prima fase di training è necessario addestrare il dispositivo al normale comportamento che seguono i fenomeni fisici che deve monitorare.

Ogni sensore ha associato un buffer circolare per memorizzare le ultime misure effettuate. In questa fase i sensori campionano ad una determinata frequenza i valori x_i dei diversi sensori s_i, fino a riempire i buffer oppure finché non è stato acquisito un numero sufficientemente grande di campioni per poterne rappresentare il normale comportamento del fenomeno fisico. Si suppone che in questa fase l'ambiente sia

Algorithm 4.1 Processo di inizializzazione dei nodi.

1. $dataTrend_j = \{x(t), t = 1, ..., N\}$, $error_j = \{x(t), t = 1, ..., N\}$;
2. for $i = 0$ to N\{
   (a) for each sensor j\{
      i. Acquire $dataTrend_j(i)$;
   (b) }
3. }
4. for each sensor j calculate $ARmodel_j(dataTrand_j)$;
5. for $i = 0$ to N\{
   (a) for each sensor j\{
      i. Acquire $dataTrend_j$;
      ii. $error_j(i) = dataTrend_j(i) - ARprediction(dataTrend_j, i)$;
   (b) }
6. }
7. for each sensor j {
   (a) $\mu_j = \sum_{i=0}^{N} \frac{error_j(i)}{N}$;
   (b) $\sigma_j^2 = \sum_{i=0}^{N} \frac{\left(error_j(i) - \mu_j\right)^2}{N-1}$;
8. }
9. Configure ICI based CDT and HP with μ_j and σ_j^2.

stazionario e non intervengano elementi a modificarne il comportamento caratteristico. Da questi dati raccolti il dispositivo deve costruire il modello che rappresenta l'andamento dei suoi parametri monitorati. Il sistema calcola per ogni sensore il modello AR sulla base dei dati collezionati nel buffer. Questo modello calcolato permette di fare una previsione del prossimo valore misurato $\hat{x}_i$ sulla base degli n precedenti, diventando più accurato al crescere di n, assumendo che il fenomeno fisico non modifichi il proprio comportamento.

4.4.1.2 Calcolo delle caratteristiche dei residui

In una seconda fase del training va valutata la bontà del modello e la rumorosità del sensore. Vengono eseguite ancora una serie di misurazioni dei parametri monitorati, e confrontati con i valori predetti dal modello calcolato nella fase precedente. Ogni parametro avrà anche un buffer addetto a salvare gli errori commessi ad ogni misura, costruendo per ogni sensore s_i l'insieme di riferimento R_i^{ref}. Finite

queste misurazioni verranno calcolate media e varianza caratteristiche dell'errore commesso. Questi dati serviranno come riferimento per eseguire il Change Detection Test ed i test di ipotesi, che dovranno tenere conto dell'errore strutturale dovuto all'imprecisione del modello ed al rumore.

Raccolti questi dati sull'errore di riferimento, viene configurato l' ICI based CDT sulla base dei residui calcolati, la dimensione W della finestra su cui eseguire i futuri test ed il parametro dell'intervallo di confidenza Γ da applicare al test. Vengono anche calcolate le caratteristiche di media e varianza utili all'esecuzione dei test di ipotesi.

Da questo momento il dispositivo è in stato ready, e può iniziare a svolgere il compito di monitoraggio dei sensori.

In questa fase non esiste ancora la rete, ma ogni singolo nodo ha studiato la propria porzione di ambiente autonomamente, e finché non verrà contattato dal dispositivo Android continuerà a monitorare i parametri senza poter comunicare e sfruttare la sinergia delle informazioni di tutta la rete.

4.5 Fase operativa

Finita l'inizializzazione di ogni dispositivo si potrà cominciare la fase di controllo dei fenomeni fisici impostati. Monitorare significa raccogliere i dati dall'ambiente e compiere un'analisi per verificare se qualche evento stia modificando il normale comportamento. Questa routine verrà ripetuta ciclicamente alla frequenza di campionamento preimpostata.

4.5.1 Esecuzione CDT

Il primo passo per individuare un cambiamento viene ovviamente compiuto a livello di singolo dispositivo con l'esecuzione del Change Detection Test. Questa fase è cruciale per il funzionamento del sistema, perché viene determinata la sensibilità del sistema. Si è scelto di utilizzare l'algoritmo CDT basato su ICI, che monitora il buffer dell'errore in cerca di anomalie. L'analisi non viene eseguita per ogni dato ricevuto, ma su finestre di dati. Il dispositivo per un determinato numero di cicli acquisirà valori $x_i(t)$ dall'ambiente senza eseguire su di essi un'analisi. Raggiunto il numero W di campioni il dispositivo esegue il CDT sulla finestra di dati raccolti. L'algoritmo analizza questi dati e decide se questi potrebbero appartenere ad una distribuzione con le stesse caratteristiche di R_i^{ref}, e ricalcola l'intervallo di riferimentoin funzione del parametro Γ. Se una volta ricalcolati i margini dell'intervallo quello sinistro risulta maggiore di quello destro allora l'intervallo è vuoto, e l'algoritmo sostiene di aver individuato un cambiamento nei dati che sta analizzando. Qualora invece l'intervallo sia non vuoto, i dati contenuti nella finestra analizzata hanno caratteristiche simili a quelli di riferimento, quindi non si ha motivo di sospettare cambiamento. In tal

caso il dispositivo continua la sua normale routine, raccogliendo la prossima finestra di dati da monitorare con il CDT, e così via finché l'intervallo ha dimensione non nulla.

Utilizzando questo tipo di test, che risulta particolarmente efficace, si ha lo svantaggio di poter eseguire il test soltanto quando sono stati raccolti un numero sufficiente di campioni per riempire la finestra. Questo comporta un ritardo strutturale nell'individuazione di cambiamenti. Perciò l'approccio proposto non sarebbe adatto a casi particolari di rischio, dove c'è bisogno di rilevare prontamente un cambiamento per evitare danni indesiderati.

4.5.2 Validazione locale

Come già detto, in caso non venga rilevato alcun cambiamento il dispositivo non compie nessuna azione e continua l'attività di monitoraggio dell'ambiente.

Consideriamo invece il caso in cui il CDT dia esito positivo. Il test sta ipotizzando che sia avvenuto un cambiamento nell'ambiente, e si vuole verificare questa affermazione per evitare falsi positivi. Il test impostato per la validazione è il paired t-test, che compara la media degli ultimi campioni sospetti di cambiamento con quella calcolata in fase di inizializzazione. In caso che il test rifiuti l'ipotesi di cambiamento, il dispositivo non esegue nessuna azione e continua ad eseguire il prossimo ciclo di controllo; altrimenti, in caso di conferma, il sistema valida la decisione di cambiamento ed esegue un riadattamento alle nuove condizioni ambientali.

4.5.3 Re-training

Quando il dispositivo capisce di essere di fronte ad un cambiamento ambientale la cui base dei concetti non ne sa descrivere il comportamento, deve riconfigurarsi in modo da costruire i nuovi modelli che descrivano l'evoluzione dell'andamento per tutti i parametri per cui è stato confermato il cambiamento. Esistono metodi che permettono di analizzare la serie dei valori assunti da una variabile ed identificare l'istante T^* in cui si verifica una variazione. In [2] viene proposto un algoritmo di Refinement Procedure per l'individuazione di T^*, mentre in [3] è utilizzato il Change Point Method a questo scopo. Alternativamente è possibile, come nella soluzione proposta, considerare iniziato il cambiamento da W campioni precedenti, dove W è la dimensione del campione esaminato dal CDT. Significa che in questo istante di tempo il dispositivo ha rilevato un evento che ha modificato i valori dei fenomeni fisici monitorati e dovrà riconfigurare i sensori considerando le nuove condizioni proprio a partire da T^*, cioè da quando il fenomeno fisico ha assunto un comportamento diverso da quello assunto precedentemente.

Per la costruzione del modello c'è bisogno di un numero rilevante di valori, e qualora quelli successivi a T^* non siano sufficienti al calcolo del modello bisogna

impostare il dispositivo in stato di reading, cioè il dispositivo smette di monitorare l'ambiente finché non ha elaborato il nuovo modello valido. In questo stato esegue soltanto la funzione di lettura, senza compiere il successivo controllo.

Una volta raggiunta la dimensione minima per l'elaborazione del modello AR l'unità passa in stato di training, e con i valori acquisiti calcola i parametri del nuovo modello per i sensori che sono stati interessati dal cambiamento. Successivamente esegue la dimensionazione di media e varianza dell'errore, calolando la differenza tra i valori raccolti ed il valore atteso utilizzando il nuovo modello. Infine viene riconfigurato l'ICI based CDT ed i test di ipotesi come in fase di inizializzazione. Finita la riconfigurazione il dispositivo può tornare in stato ready, riprendendo il suo lavoro di raccolta dati e analisi in cerca di successivi cambiamenti.

4.5.4 La struttura dati

Ogni unità sensoriale è predisposta per monitorare più fenomeni fisici. Dovrà quindi gestire una lista di sensori, ognuno con le proprie diverse caratteristiche. Dopo aver discusso i compiti che dovrà adempiere con le risorse necessarie per le analisi sui dati, riporto di seguito le informazioni essenziali che dovrà contenere la struttura dati che descrive i sensori:

- un buffer $trend[N]$, con gli ultimi valori letti dal sensore;
- un buffer $error[N]$, con gli ultimi scarti calcolati tra il modello ed il valore atteso;
- una struttura dati con dei parametri che rappresenti il modello, dipendente dal tipo di modello utilizzato;
- i valori μ e σ^2, media e varianza dell'errore, necessari a stimare i valori dei test di ipotesi;
- la struttura dati utilizzata per il calcolo del CDT;
- una variabile di stato del sensore, che può valere 'training', 'ready' o 'reading'.

Algorithm 4.2 ICI based - Change Detection Test.

1. Compute {M(s), s=1,...,S_0}.
2. $\hat{\mu}^M_{S_0} = \sum_{s=1}^{S_0} \frac{M(s)}{S_0}$.
3. $\hat{\sigma}^M_{S_0} = \sqrt{\sum_{s=1}^{S_0} \frac{\left(M(s)-\hat{\mu}^M_{S_0}\right)^2}{S_0-1}}$.
4. Define $\mathrm{I}^M_{S_0} = \left[\hat{\mu}^M_{S_0} - \hat{\sigma}^M_{S_0}, \hat{\mu}^M_{S_0} + \hat{\sigma}^M_{S_0}\right]$.
5. Compute the firts six comulations of X from TS.
6. Compute h_0 as described in [cit.], and define T.
7. Compute {S(s), s=1,...,S_0}.
8. Compute {V(s)=$T\left(S\left(s\right)\right)$, s=1,...,$S_0$}.
9. $\hat{\mu}^V_{S_0} = \sum_{s=1}^{S_0} \frac{V(s)}{S_0}$.
10. $\hat{\sigma}^V_{S_0} = \sqrt{\sum_{s=1}^{S_0} \frac{\left(V(s)-\hat{\mu}^V_{S_0}\right)^2}{S_0-1}}$.
11. Define $I^V_{S_0} = \left[\hat{\mu}^V_{S_0} - \hat{\sigma}^V_{S_0}, \hat{\mu}^V_{S_0} + \hat{\sigma}^V_{S_0}\right]$.
12. Set $s = S_0$
13. while ($I^M_{S_0} \neq \emptyset$ *and* $I^V_{S_0} \neq \emptyset$){
    (a) Set $s = S_0 + 1$
    (b) Wait for v observations, until $Y(s)$ is populated
    (c) Compute $M(s)$ e $V(s)$
    (d) $\hat{\mu}^M_s = \frac{(s-1)\cdot\hat{\mu}^M_{s-1}+M(s)}{s}$
    (e) $\hat{\sigma}^M_s = \frac{\hat{\sigma}^M_{S_0}}{\sqrt{s}}$
    (f) $\hat{\mu}^V_s = \frac{(s-1)\cdot\hat{\mu}^V_{s-1}+V(s)}{s}$
    (g) $\hat{\sigma}^V_s = \frac{\hat{\sigma}^V_{S_0}}{\sqrt{s}}$
    (h) $I^M_s = \left[\hat{\mu}^M_s - \hat{\sigma}^M_s, \hat{\mu}^M_s + \hat{\sigma}^M_s\right] \cap I^M_{s-1}$
    (i) $I^V_s = \left[\hat{\mu}^V_s - \hat{\sigma}^V_s, \hat{\mu}^V_s + \hat{\sigma}^V_s\right] \cap I^V_{s-1}$
14. }
15. Detect a change in $[(s-1)\,v, sv]$

Chapter 5

Intelligenza a livello di coordinatore e rete

Per rendere il sistema adattativo bisogna essere in grado di capire cosa stia succedendo nell'ambiente. La variazione rilevata a livello di dispositivo non permette di distinguire tra cambiamento nell'ambiente e guasto al sensore. Per disambuiguare questa situazione è possibile sfruttare il fatto che le differenti unità forniscono differenti viste dello stesso fenomeno fisico. Per questa ragione si vuole effettuare un'analisi a livello di rete dei sensori che operano nello stesso ambiente.

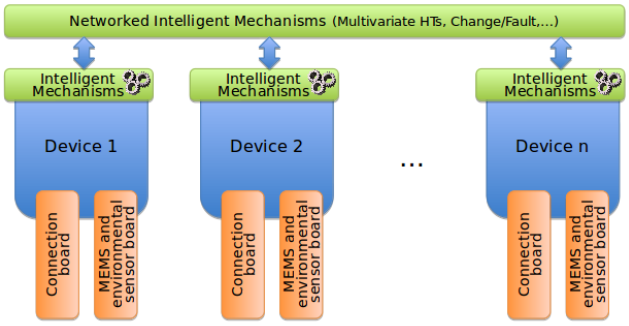

Figure 5.1: La condivisione delle informazioni fornite da ciascun nodo consente la creazione di un'intelligenza a livello globale, ampliando possibili analisi sul sistema ciberfisico.

I dispositivi saranno utilizzati per rilevare le informazioni relative ad una porzione di ambiente e, grazie alla capacità dei nodi di comunicare, sarà possibile creare una conoscenza che integri quelle dei singoli dispositivi, permettendo così la creazione di un'intelligenza che ne sfrutti la sinergia e che sia capace di distinguere i guasti all'interno della rete da particolari eventi che modificano l'ambiente monitorato sulla base delle singole analisi sui sensori.

5.1 Topologia della rete e distribuzione delle funzionalità

Nella costruzione del sistema Cyber-fisico distribuito va determinato che tipo di architettura adottare per meglio garantire le funzionalità che forniscono intelligenza al dispositivo ed alla rete. L'idea di voler sfruttare le relazioni tra dati di unità differenti implica che ogni coppia di nodi debba essere concettualmente in comunicazione diretta. Un'architettura con connessione totale sembrerebbe la più appropriata per questo scopo, permettendo al dispositivo che rileva il cambiamento di interrogare direttamente i nodi con cui è in relazione.

D'altro canto questo approccio distribuito aumenterebbe la complessità della soluzione tecnologica implementata, richiedendo una meticolosa pianificazione della sincronizzazione e coordinazione tra dispositivi. Per soddisfare in modo semplice questi requisiti, si è scelta una topologia a stella, con un dispositivo centrale che collega tutti i dispositivi dispersi nell'ambiente. In questo modo si potrà centralizzare il calcolo delle relazioni, fornendo al nodo centrale visione su tutta la rete. Per comunicare con gli altri nodi, ogni unità dovrà passare per questo nodo, che orchestrerà le comunicazioni tra ogni coppia di unità. Inoltre, facendo passare tutte le comunicazioni tramite un solo dispositivo, questo potrà avere una visione aggiornata sullo stato della rete, aggregare i dati e fornire informazioni all'utente senza necessitare di elevata complessità.

La struttura della rete mette in evidenza che il collo di bottiglia del sistema potrebbe essere il coordinatore a cui sono collegate tutte le periferiche: questo deve ricevere dai nodi connessi delle informazioni sintetiche, che siano un indice del proprio stato. Per arrivare a questo valore ogni periferica dovrà elaborare le informazioni a propria disposizione. Viene applicata quindi un'intelligenza distribuita tra i diversi nodi, ognuno in grado di riconoscere il proprio stato con una fase di elaborazione locale e di comunicarlo al supernodo. Questo sfrutterà i dati a lui recapitati per prendere decisioni e inviarle ai nodi periferici.

Questa divisione a due livelli gerarchici della capacità computazionale del sistema porta con sé alcuni vantaggi: ogni nodo non è semplicemente un sensore responsabile di campionare un fenomeno fisico, ma è un sistema capace di valutare se l'informazione campionata sia rilevante ed in grado di elaborare un valore sintetico rappresentante l'informazione. Questo processo sarebbe potuto essere delegato anche al dispositivo centrale, che con i dati ricevuti dal sensore avrebbe eseguito gli stessi test e raggiunto le stesse decisioni. Va però ricordato che nel caso pessimo, quindi in un contesto con numerosi nodi, questo funzionamento avrebbe inficiato l'efficienza del sistema, comportando un overload informativo del supernodo centrale, che avrebbe avuto una grossa mole di dati da analizzare e, conseguentemente, maggiore complessità e possibili inefficienze. Inoltre ogni nodo è dotato di potenziale capacità elaborativa che non sarebbe stata utilizzata. La trasmissione di indici sintetici per

trasmettere l'informazione consente di sfruttare appieno i nodi periferici e di semplificare la complessità del nodo centrale. Altro vantaggio è la possibilità di trasmettere un numero limitato di dati, allo scopo di non sovraccaricare la rete. Il dispositivo centrale è dotato di un'unica antenna con cui gestisce le connessioni, e sovraccaricare questa rete potrebbe significare perdere dei pacchetti. Inoltre trasmettere significa consumare energia, e per un sistema alimentato a batteria ridurre i consumi implica la garanzia di una maggiore vita utile al dispositivo, una qualità fondamentale per questo tipo di applicazioni.

Come già detto il collo di bottiglia del sistema è il nodo centrale che collega tutti i nodi periferici, ed in assenza di esso non si avrà più una rete, ma tante unità indipendenti. Il sistema deve garantire il proprio funzionamento anche in questa evenienza. Ovviamente l'ambiente non sarà più monitorabile dall'utente, poiché questa situazione sarà caratterizzata dall'assenza di un collettore e visualizzatore di tutte le informazioni; ciononostante i dispositivi devono continuare ad operare autonomamente, anche senza la possibilità di avere informazioni riguardanti la rete ma solo con le proprie informazioni, in attesa che la connessione venga ristabilita e la rete riprenda il suo corretto funzionamento.

5.2 Change Detection Test gerarchico

Come esposto nel capitolo precedente, ogni dispositivo esegue localmente un Change Detection Test per la ricerca di cambiamento ed un test per validarne la decisione. Questo permette a livello di nodo di avere un buon livello di certezza su quello che sta accadendo nella porzione di ambiente monitorata da quel nodo. Quello che si vorrebbe fare è sfruttare l'informazione elaborata da questo nodo per effettuare dei controlli su più vasta scala, a livello globale, in modo da verificare che il cambiamento sia sentito da tutti i sensori che ne sono coinvolti.

Quando un nodo in fase operativa rileva e valida un cambiamento, ci si aspetta che anche i sensori a lui correlati la rilevino. Si vuole andare a sfruttare le informazioni di quei sensori che in fase di inizializzazione sono risultati correlati ad esso e che non hanno ancora rilevato il cambiamento. Dal primo sensore deve partire una richiesta diretta ai dispositivi a lui correlati per chiedere se anche loro abbiano rilevato un cambiamento ad un determinato istante di tempo. Ricevuta la richiesta viene eseguito il test di ipotesi sulle variabili interessate, e viene inviata la risposta al mittente informandolo se dall'analisi dei dati sia stato scelto di confermare o negare il cambiamento. Il secondo dispositivo potrebbe non aver ancora rilevato il cambiamento per la diversa rumorosità dei sensori oppure perché il CDT non è ancora stato eseguito a causa della diversa frequenza di campionamento.

Si è così aggiunto un livello di validazione ulteriore. La prima fase è avvenuta a livello locale, con CDT e test di ipotesi, mentre la seconda fase è a livello globale,

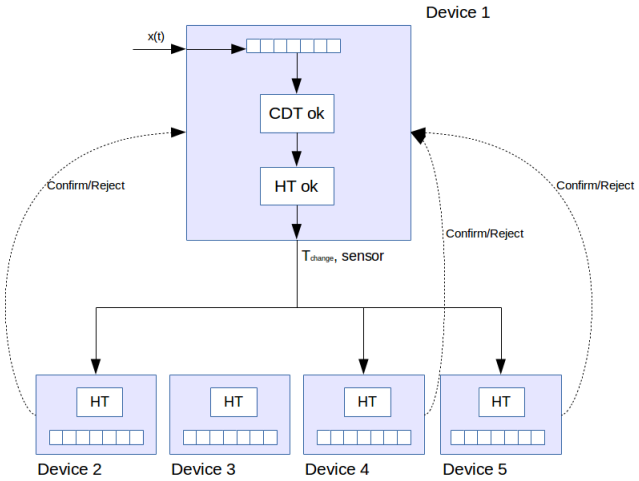

Figure 5.2: Funzionamento del Change Detection Test gerarchico. Nel caso particolare Device 1 rileva un cambiamento e chiede conferma ai Device 2,4 e 5, a lui correlati, comunicandogli istante di cambiamento e sensore interessato. Questi rispondono con una conferma o negazione del cambiamento dopo il relativo test di ipotesi.

con i test di ipotesi eseguiti dai dispositivi interessati, sfruttando l'interoperabilità dei dati dell'intera rete.

Questo ulteriore passaggio è utile per diminuire ulteriormente la percentuale di falsi positivi, ma potrebbe risultarlo anche per rilevare anomalie in determinati sensori, come guasti o eventi eccezionali che riguardano ed influenzano soltanto un determinato sensore.

Quando il dispositivo avrà ricevuto conferma della sua ipotesi di cambiamento sarà possibile avviare una procedura di re-training dall'istante del cambiamento individuato, ricalcolando il modello che descrive l'andamento dei parametri interessati, riconfigurando l'ICI based CDT ed i parametri necessari ai test di validazione.

5.3 Inizializzazione della rete

Dopo che le unità del sistema avranno finito la loro fase di inizializzazione, queste saranno pronte per andare a costituire i nodi della rete. Il coordinatore scansionerà il territorio e selezionerà i dispositivi interessati. Dati i molti fattori che possono influenzare la connessione, come distanza, barriere fisiche o interferenze, alcuni dispositivi potrebbero non essere momentaneamente rilevati. In tal caso si può ripetere la scansione per ricercare anche i dispositivi mancanti.

Per creare una rete attiva e abilitata al controllo dell'ambiente sono necessarie 3 fasi: creazione delle connessioni, calcolo del grafo delle dipendenze e sincronizzazione.

5.3.1 Connessione

Il coordinare avvia questa fase scansionando i dispositivi visibili nel suo raggio di azione ed individua quelli che devono far parte della rete in base al nome. Una volta individuati e selezionati i nodi della rete l'utente dà l'istruzione allo smartphone di formare la rete, inviando ad ognuno una richiesta di connessione. Questa viene ricevuta da ogni nodo ed accettata. Ora le connessioni sono stabilite, ed è possibile inviare e ricevere dati da tutti i nodi.

5.3.2 Creazione del grafo delle dipendenze

La seconda cosa che deve fare il coordinatore è creare il grafo delle dipendenze. Alcuni parametri controllati dai sensori potrebbero essere in relazione sia a livello locale che a livello globale. Ad esempio, dati due diversi dispositivi A e B posizionati in una camera, entrambi progettati per monitorare temperatura, umidità ed accelerazione, la temperatura misurata dal sensore A sarà in qualche modo in relazione a quella del sensore B, mentre le accelerazioni misurate dai due sensori probabilmente non saranno in alcuna relazione. Se invece il sensore B fosse posizionato all'aperto, la sua temperatura seguirebbe un certo andamento in base all'orario, mentre quella di A rimarrebbe sempre più o meno costante. In questo secondo caso i due sensori non saranno in alcuna relazione. Allo stesso modo possono esistere relazioni tra sensori di diverso tipo, ad esempio temperatura ed umidità spesso sono spesso correlate. È interessante sfruttare queste informazioni per creare una struttura di validazione del cambiamento, che dovrà essere visto anche dai sensori che sono correlati a quello che l'ha notato per primo. Per poter costruire questo grafo bisogna però possedere tutti i trend di misurazioni effettuate su ogni sensore di ciascun dispositivo.

Appena stabilita la connessione inizia questa fase di trasferimento dei trend campionati. Con un primo invio si trasmettono i codici che indicano a quali sensori sono riferiti ed in quale ordine, e successivamente trasmette di seguito tutti i valori che ha campionato seguendo l'ordine stabilito. Ad esempio, se il dispositivo sta monitorando temperatura ed umidità, al primo invio trasmetterà i codici dei due sensori, prima quello di temperatura poi quello di umidità, mentre negli invii successivi trasmetterà messaggi dove sarà concatenato prima il valore di temperatura e poi quello di umidità.

Ad ogni ricezione il coordinatore deve collezionare i dati ricevuti in apposite strutture. Una volta ricevuti i dati da ogni dispositivo della rete, potrà proseguire a calcolare le correlazioni tra i sensori. Per la costruzione del grafo delle dipendenze è necessario confrontare i trend per ogni coppia di sensori. Sia $G = (V, E)$ il grafo delle dipendenze, dove $V = \{s_1, ..., s_n\}$ è l'insieme formato da tutti i sensori della rete, ed $E = \{e_{i,j} | s_i \, ed \, s_j \, risultano \, correlati\}$ è l'insieme degli archi che unisce i sensori tra cui è presente una relazione. Per determinare il grado di correlazione viene utilizzato

l'indice di correlazione di Pearson. Questo riceve in ingresso gli array con i trend di dati in esame, e restituisce un indice compreso tra -1 e 1 in base alla loro correlazione. Si può impostare una soglia sopra la quale si considerano i due sensori correlati e si aggiunge il corrispondente arco $e_{i,j}$ nel grafo, mentre al di sotto si considerano non correlati. Ripetendo questo procedimento per tutte le combinazioni di coppie di sensori si otterrà il grafo completo delle dipendenze.

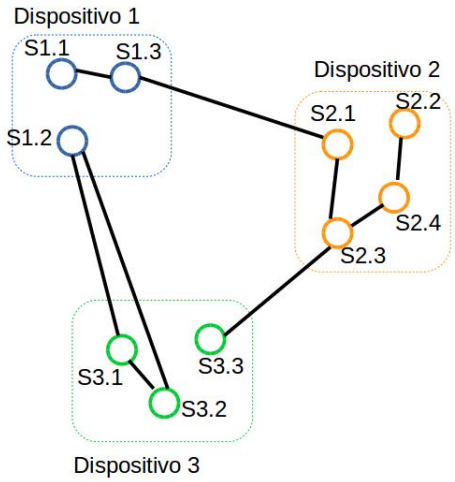

Figure 5.3: Esempio di grafo delle dipendenze tra sensori locali e tra diverse board.

Come già accennato, queste dipendenze possono essere sfruttate anche a livello locale per effettuare dei test multivariati. Lo smartphone dovrà quindi comunicare ad ogni dispositivo le proprie dipendenze interne.

5.3.2.1 Indice di correlazione di Pearson

Una volta che il dispositivo ha rilevato e validato localmente il cambiamento, si vuole sfruttare le informazioni conosciute dagli altri dispositivi per validare anche globalmente il cambiamento. Si dovranno considerare solo le informazioni di quei sensori che risultino relazionati con quello che ha rilevato il cambiamento: ciò significa che i due diversi sensori stanno monitorando due fenomeni fisici che si influenzano reciprocamente ed al variare di uno varierà anche l'altro. Per scoprire queste relazioni si può utilizzare l'indice di correlazione di Pearson. Questo metodo statistico esprime un'eventuale relazione di linearità tra due variabili.

Date due variabili statistiche X e Y, l'indice di correlazione di Pearson è definito come la loro covarianza divisa per il prodotto delle deviazioni standard delle due variabili:

$$\rho_{XY} = \frac{\sigma_{XY}}{\sigma_X \sigma_Y}$$

.

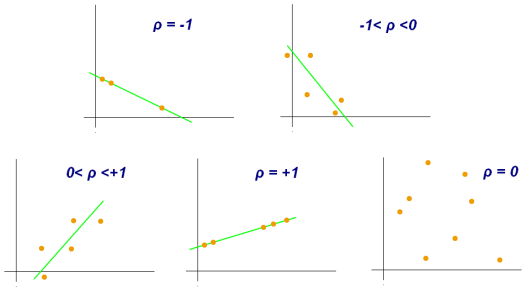

Figure 5.4: Esempi di correlazioni di Pearson.

Il coefficiente assume sempre valori compresi tra -1 e 1. Se assume un valore vicino allo 0 si dice che le variabili sono incorrelate, mentre più si avvicina agli estremi e più ci sarà correlazione fra loro. Se il valore è positivo saranno direttamente correlate, altrimenti si diranno inversamente correlate.

5.3.3 Sincronizzazione

L'ultima cosa che resta da fare è inviare l'orario per sincronizzare tutto il sistema. Questo è necessario perché quando un nodo notifica un cambiamento fa anche ua stima di quando questo ha avuto inizio. Se non si tratta di un falso positivo, questo cambiamento dovrebbe coinvolgere anche altri sensori che sono correlati ad esso, e per la diversa dinamica non hanno ancora rilevato il cambiamento che è già in atto. Sincronizzando questi dispositivi è possibile verificare se anche altri sensori hanno iniziato a notare un cambiamento, magari senza superare la soglia di guardia, ma comunque avvicinandosi.

Ogni sensore lavorerà ad una certa frequenza prestabilita, e ad ogni campionamento corrisponde un time-stamp. Il coordinatore dopo aver inviato le informazioni sulle correlazioni locali invia l'orario attuale in secondi, quindi $t_0 = ore \cdot 3600 + minuti \cdot 60 + secondi$. I dispositivi lo ricevono e lo impostano come orario iniziale di sistema da cui partire a contare. Quando dovranno rilevare il tempo di cambiamento lo calcoleranno come $t_{change} = t_0 + t_s \cdot timestamp$, dove t_s è l'intervallo che intercorre tra due campionamenti, e timestamp è riferito a quello in cui viene rilevato il cambiamento. Questo tempo viene comunicato allo smartphone, che lo comunica a tutti i dispositivi che sa avere almeno una relazione con il primo dispositivo.

Da questo momento la rete è funzionante ed ogni dispositivo è inizializzato con i modelli AR relativi ai comportamenti dei fenomeni da controllare e con le caratteristiche dei residui, necessarie al rilevamento di cambiamenti. Conclusa anche la fase di connessione ogni dispositivo entra a far parte della rete ed all'occorrenza potrà

interagire con gli altri dispositivi per verificare le decisioni sullo stato dell'ambiente monitorato.

5.4 Validazione globale

Avendo calcolato il grafo delle dipendenze è ora possibile sfruttare una maggiore quantità di informazioni per descrivere il comportamento dell'ambiente. È noto che due sensori risultati in relazione stanno monitorando due fenomeni che mantengono un comportamento simile, e nel caso si presenti un evento inaspettato che modifica l'ambiente questo deve coinvolgerli entrambi.

Consideriamo il caso di un dispositivo regolarmente attivo e connesso alla rete che può sfruttare il meccanismo di validazione locale

Una volta confermato il cambiamento a livello locale si vuole compiere un test a livello più ampio che confermi che il cambiamento è in atto anche nell'ambiente controllato da altri sensori. Il singolo sensore presenta come limitazione quella di non poter verificare la propria correttezza non potendo confrontare il suo comportamento con qualcuno che fornisce una versione corretta del comportamento.

Per questo motivo si ha la necessità di confrontarne il comportamento con altri che potrebbero essere stati interessati dal cambiamento. Si potranno ora sfruttare le correlazioni trovate in fase di connessione. Ogni dispositivo ha inviato i dati campionati da ogni sensore, e il coordinatore ha saputo individuare le correlazioni presenti tra essi. Il dispositivo centrale ha una visione globale della rete e può sapere che se un nodo ha individuato un cambiamento; questo, a sua volta, probabilmente coinvolgerà altri determinati dispositivi che agiscono nello stesso ambiente.

Questa visione ad ampio raggio è utile per dare un'ulteriore verifica della coerenza dei dati con le decisioni prese. Inoltre permette di individuare l'eventuale comportamento anomalo di alcuni nodi. L'individuazione di guasti esula dallo scopo della tesi, ma queste informazioni che vengono ricavate sono un indice da cui si potrà eseguire test di Fault Detection. In ogni modo sapere che anche i sensori correlati hanno percepito (o stanno cominciando a percepire) il cambiamento è una prova che il cambiamento dell'ambiente è confermato dai diversi dispositivi che lo controllano.

Nel caso in cui un dispositivo ha rilevato un cambiamento locale, la ricerca globale di conferma deve passare per tutti i dispositivi che potrebbero essere stati coinvolti dal cambiamento. Quindi il dispositivo che ha rilevato un cambiamento comunica allo smartphone Android quale sensore ha rilevato il cambiamento ed in quale istante di tempo. Questo ricerca nel grafo globale delle dipendenze con quali sensori di quali dispositivi siano correlati ad esso e ad ogni dispositivo coinvolto vengono inoltrati il codice dei propri sensori correlati a quello da cui è partita la notifica e l'istante di tempo del cambiamento.

Ora i dispositivi devono compiere la validazione locale del cambiamento. La dif-

ferenza rispetto al caso di singolo nodo è che quando il coordinatore ha calcolato il grafo delle dipendenze ha informato ogni nodo delle relazioni interne tra i propri sensori. Qualora ce ne siano diventa possibile eseguire un test multivariato per verificare il cambiamento su più sensori contemporaneamente, che, essendo in relazione, potrebbero esserne tutti coinvolti. Sulla base delle relazioni note sul sensore il dispositivo decide quale test di ipotesi applicare. In caso questo non abbia relazioni con altri verrà eseguito il paired t-test per verificare la variazione nella distribuzione dei dati acquisiti, mentre in caso sia correlato ad altri si eseguirà il test multivariato T^2 di Hotelling che verifica il cambiamento anche per gli altri sensori. Dal test selezionato verrà formulata una risposta di conferma o negazione del cambiamento da fornire al coordinatore.

Listing 5.1: Porzione di codice relativa alla validazione locale di un dispositivo coinvolto nel cambiamento.

```
bool searchChange(Sensor *s)
{
  int n = countDependencies();
  bool result;
  if(n>1)
    result = hotellingT2(s);
  else
    result = t_test(s);
  return result;
}
```

Il dispositivo centrale sulla base di tutte le risposte ricevute sarà in grado di valutarle e potrà verificare se il cambiamento è stato sentito anche dagli altri sensori interessati dal mutamento del fenomeno fisico indagato. Se altri sensori confermano il cambiamento viene valutato sia avvenuto un cambiamento a livello ambientale; mentre in caso di nessuna conferma si evidenzia un comportamento anomalo di un singolo sensore, che verrà considerato guasto.

Una volta che i dispositivi avranno verificato il cambiamento verrà eseguita la fase di re-training sui sensori coinvolti, come descritto nel capitolo precedente.

5.4.1 Test multivariato: Hotelling's T-squared

Il test T-quadrato di Hotelling è un test multivariato utilizzato per verificare la correlazione tra diversi trend di dati. Il test di ipotesi può essere formulato come:

Siano μ il vettore contenente le medie dei parametri stimati, e

$\mu_0 = \begin{bmatrix} \mu_1 \\ \mu_2 \\ . \\ . \\ \mu_n \end{bmatrix}$ il vettore colonna dei valori medi attesi.

Si possono formulare le seguenti ipotesi:

$$H_0 : \mu = \mu_0$$
$$H_1 : \mu \neq \mu_0$$

dove H_0 è l'ipotesi nulla che indica l'assenza di cambiamento, ed H_1 è l'ipotesi alternativa.

La statistica T-quadrato di Hotelling è definita come segue:

Siano $x_1, x_2, ..., x_n$ px1 vettori colonna di numeri reali corrispondenti alle misurazioni i.i.d. effettuate e

$\bar{x} = (x_1 + x_2 + ... + x_n)/n$ le loro medie,

$S = \frac{1}{n-1} \sum_{i=0}^{n} (x_i - \bar{x})(x_i - \bar{x})'$ la matrice non negativa della loro varianza e

La statistica T-quadrato di Hotelling è data da

$$T^2 = (\overline{x} - \mu_0)'(\frac{1}{n}S)^{-1}(\overline{x} - \mu_0)$$

Il valore di T^2 indica la distanza di $\overline{x}$ da μ_0. Se questo valore è elevato significa che la distanza sarà elevata, quindi si potrà rifiutare H_0 e si accetta l'ipotesi alternativa di cambiamento. Per valutare l'intervallo entro il quale si accetta l'ipotesi nulla, si sfrutta la proprietà del test T^2 di Hotelling per cui

$$T^2 \sim \frac{np - p}{n - p} F_{p,n-p}(\alpha)$$

dove $F_{p,n-p}(\alpha)$ denota la variabile casuale di Fisher-Snedecor con p ed n-p gradi di libertà e livello di significatività α a cui si vuole svolgere il test di ipotesi. potrà essere rifiutata H_0 a favore di H_1 se $T^2 \geq \frac{np-p}{n-p} F_{p,n-p}(\alpha)$.

I test multivariati permettono di osservare simultaneamente il comportamento di più variabili, ottenendo risultati che permettono di valutare il cambiamento sulle variabili coinvolte nel loro insieme.

Chapter 6

Un sistema ciberfisico intelligente basato su ST Nucleo: sviluppo e testing

Dopo aver presentato il sistema ciberfisico ed esposto le le soluzioni metodologiche impiegate per la realizzazione, si vuole proporre una possibile implementazione del sistema progettato allo scopo di valutarne il funzionamento in un contesto reale. Di seguito verranno descritte in una prima parte le risorse tecnologiche utilizzate per l'implementazione, successivamente una descrizione di come queste sono state utilizzate dare vita al sistema; infine sono esposti i risultati emersi dalla fase di testing.

6.1 Risorse tecnologiche impiegate

6.1.1 Le schede NUCLEO ST

Per l'implementazione dei dispositivi da distribuire nell'ambiente sono state utilizzate le schede Nucleo ST, integrate con alcune schede di espansione.

Ogni unità utilizzata per monitorare l'ambiente deve essere composta da 3 diversi moduli:

- il modulo base è il microcontrollore vero e proprio, programmato per gestire gli altri due moduli, permettendo l'acquisizione dei dati, l'analisi e la connessione con la rete;
- un modulo sensoristico, che permette al nodo di campionare i parametri fisici;
- un modulo di connettività, che consente al nodo di connettersi alla rete e comunicare le sue informazioni.

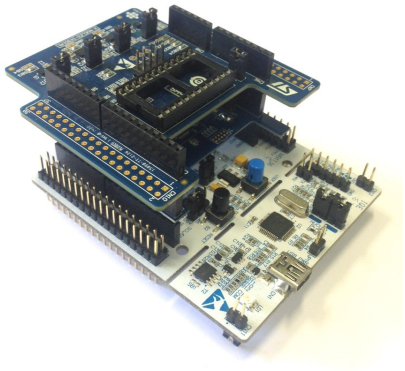

Figure 6.1: Un esempio di nodo della rete: la scheda Nucleo STM32 con le schede di espansione con sensori e trasmettitore Bluetooth.

Per l'implementazione di questi 3 livelli sono stati utilizzati componenti prodotti dalla STMicroelectronics, di cui descriverò brevemente le caratteristiche principali.

6.1.1.1 NUCLEO STM32 L0-F4

Lo sviluppo dell'applicazione è stato svolto sulle board Nucleo STM32. Sono state utilizzate board di due diverse serie, L0 ed F4, che differiscono per potenza e prestazioni ma che vengono programmate nel medesimo modo. Queste schede hanno integrato il ST-LINK/V2-1 debugger/programmer, che agevola la loro programmazione. Sarà sufficiente semplicemente collegare il dispositivo al PC tramite USB, e copiarci all'interno un file binario contenente l'applicazione compilata per l'installazione e l esecuzione. Sono presenti due pulsanti, uno di Reset per far ripartire l'esecuzione del programma installato ed uno User per interagire con l'utente. La piedinatura della scheda è stata progettata compatibile a quella di Arduino Uno per agevolare la programmaizione e l'utilizzo di altri componenti aggiuntivi.

Per generare il file binario con il programma è stato utilizzato il tool online MBED, che fornisce supporto agli sviluppatori di microcontrollori ARM. Vengono fornite una serie di librerie, utili ad esempio per la gestione del Bluetooth e dei sensori, che semplificano la programmazione in C/C++ del dispositivo. Selezionando da un menu lo specifico dispositivo per cui si vuole creare l'applicazione MBED, si creerà un file binario compilato in base alle specifiche hardware del particolare dispositivo. Scaricato questo file e collegata la scheda STM32 al PC via USB, si deve copiare direttamente il file all'interno del dispositivo collegato, che verrà installato dall' ST-LINK/V2-1 integrato. Finita la fase di installazione ha inizio l'esecuzione del programma scritto.

Le board della serie L0 vengono definite Ultra Low Power. Sono caratterizzate da un'efficientissima gestione dell'energia, e sono state pensate per quelle applicazioni

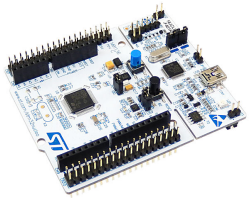

Figure 6.2: NUCLEO STM32

che debbano avere un'autonomia nell'ordine della decina di anni, anche lavorando ad alte temperature.(A discapito di questo le prestazioni: il processore utilizzato è l' ARM Cortex-M0 con frequenza in Run-mode di 32MHz, ed hanno in dotazione 64KB di memoria Flash ed 8KB di memoria RAM.)

La serie F4 è definita invece High Performance, data la frequenza di esecuzione fino a 100MHz ed altre caratteristiche che ne ottimizzano le prestazioni. (Quando si trova in Stop-mode è in grado di ridurre al minimo i consumi energetici, migliorando anche l'efficienza elettrica. Per questa board viene utilizzito un processore ARM Cortex-M4, con fino a 512KB di memoria Flash e 128 KB di memoria SRAM.)

6.1.1.2 X-NUCLEO-IKS01A1

Questa expansion board è progettata per essere compatibile con Arduino UNO, e può essere collegata alla scheda sopra descritta in modo Plug and Play, incastrando i suoi piedini negli appositi spazi sulla scheda inferiore. Questa scheda è una piattaforma multisensoriale MEMS, ed è dotata dei seguenti sensori:

- LSM6DS0: MEMS 3D accelerometro e giroscopio;
- LIS3MDL: MEMS 3D magnetometro;
- LPS25HB: MEMS sensore di pressione atmosferica;
- HTS221: sensore capacitivo digitale per temperatura ed umidità;

Per essa è implementata una libreria che ne facilita l'uso ad alto livello, mettendo a disposizione del programmatore delle funzioni che permettano di leggerne direttamente i valori dei parametri.

6.1.1.3 X-NUCLEO-IDB04A1

Anche questa expansion board ha la piedinatura uguale alla precedente, così da essere utilizzabile come Plug and Play. In questo modo i 3 moduli necessari per formare un nodo operativo della rete di sensori si possono assemblare semplicemente incastrare una sopra l'altra le 3 diverse schede.

Questo modulo si occupa della trasmissione Bluetooth. È caratterizzato da un basso consumo energetico (7.3 mA RX e 8.3 mA TX a +0 dBm).

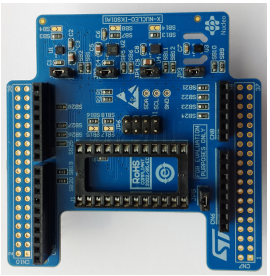

Figure 6.3: X-NUCLEO-IKS01A1

Per farla funzionare correttamente bisogna creare un ponte tra i piedini D3 e D13. Ogni dispositivo sarà riconoscibile all'interno della rete grazie ad un nome e ad un indirizzo MAC. Questo indirizzo dovrà essere univoco all'interno della rete.

La libreria utilizzata per la connessione BLE sfrutta un servizio UART, che mette a disposizione del dispositivo un servizio, con ID prefissato, con due caratteristiche, una di lettura e una di scrittura, anche loro con ID prefissati. Il dispositivo per comunicare al supernodo modificherà la sua caratteristica di scrittura, mentre una modifica della caratteristica di lettura corrisponderà ad un'istruzione ricevuta dal coordinatore.

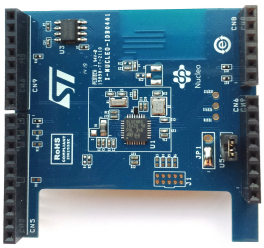

Figure 6.4: X-NUCLEO-IDB04A1

6.1.2 Android

Il dispositivo che ricopre il ruolo di supernodo deve avere una capacità elaborativa superiore ai nodi periferici, deve avere una certa autonomia energetica e deve potersi interfacciare con l'utente per notificargli lo stato della rete. In questo contesto la scelta è ricaduta sull'utilizzo di un dispositivo mobile, che meglio interpreta queste caratteristiche. Nello specifico si è scelto di utilizzare Android, a causa della maggiore diffusione (nel luglio 2014 questo sistema operativo copriva l'84% del mercato dei dispositivi mobile, in vantaggio rispetto a iOS e Windows Phone), per la semplice programmabilità e la vasta documentazione disponibile.

Sono partito a sviluppare da un'applicazione già esistente messa a disposizione dalla community. Questa era composta da due Activity: la prima eseguiva una scansione dei dispositivi Bluetooth Low Energy che lo smartphone è in grado di

trovare nelle vicinanze e ne mostra un elenco, identificandoli con nome e relativo indirizzo MAC. Selezionando un elemento si apre una seconda Activity, che mostra servizi e relative caratteristiche posseduti dal dispositivo BLE selezionato.

Da questa base ho sviluppato un'applicazione che gestisse la connessione di tutta la rete.

La prima Activity non si doveva limitare a mostrare un elenco dei dispositivi visibili, ma consentire anche all'utente di selezionare tutti e soli quei dispositivi che fanno parte della rete. La scelta è aiutata, e verranno selezionati di default i dispositivi denominati 'ST_NET_X', dove X varia per ogni nodo. Selezionati tutti i nodi della rete si può passare alla seconda Activity premendo 'Next', che riceverà dalla prima tramite Intent gli indirizzi MAC dei dispositivi selezionati.

Questa parte dell'applicazione avrà la responsabilità di creare la rete e gestirne tutte le prassi. Per la gestione della connessione l'Activity si appoggia ad un Service. Questo oggetto Android ha la caratteristica di essere unico nel sistema, quindi non sarà possibile associare ad ogni dispositivo BLE un proprio Service, ma le connessioni con tutti i dispositivi vengono gestite da uno solo.

Dallo smartphone devono partire le richieste di connessione verso i vari dispositivi ST per la creazione della rete. Ad esso viene naturalmente affidato il ruolo di GATT Client, che contatterà tutti i dispositivi di cui ha selezionato l'indirizzo nell'Activity precedente per stabilire la connessione. Per ogni indirizzo selezionato, viene creato un oggetto BluetoothDevice che rappresenti il dispositivo corrispondente. Questo crea la richiesta di connessione verso il dispositivo, e gli viene associato un oggetto di tipo BluetoothGattCallback, con un valore identificativo, con il compito di stare in ascolto dei cambiamenti di stato della connessione e dei valori delle caratteristiche per quel nodo.

Viene poi creato un oggetto di tipo BluetoothGatt, con il compito di scrittura della caratteristica GATT contenuta nel dispositivo per le comunicazioni in ingresso. Il servizio salva tutti questi oggetti in un'apposita lista.

Quando si riceveranno dei dati, viene rilevato l'evento di cambiamento di una caratteristica dall'apposito oggetto BluetoothGattCallback, che interpreta di quale tipo di dato si tratti in base al numero di byte trasmessi ed ad un byte identificativo. Questa fase di ricezione dei byte comunicati dai nodi della rete è stata eseguita dall'apposito Service, che è indipendente dall'activity che gestisce la logica della rete. Ad ogni ricezione di comunicazione, il Service analizza i byte ricevuti e crea una stringa contenente un prefisso indicante il tipo di informazione ricevuto, seguito dall'informazione vera e propria. Questa stringa e l'ID del dispositivo che ha inviato il dato verranno trasmessi tramite un Intent all'Activity, la quale utilizzerà un oggetto di tipo BroadcastReceiver per gestire nel modo adeguato l'informazione ricevuta.

Quando si deve comunicare con i nodi della rete, ci si servirà degli oggetti BluetoothGatt corrispondenti a ciascun nodo, indicando quale caratteristica di quale

servizio si vuole andare a modificare. Alcune informazioni verranno trasmesse a tutti i nodi, altre andranno comunicate ad una specifica periferica, andando a selezionare l'oggetto BluetoothGatt responsabile di comunicare con essa all'interno della lista.

6.1.3 Bluetooth Low Energy

Bluetooth Low Energy, o Bluetooth Smart, è una nuova tecnologia di comunicazione wireless introdotta da Nokia nel 2006. Si è da subito distinta per la sua efficienza nelle comunicazioni a corto raggio, trovando diffusi utilizzi in applicazioni di healthcare, fitness, domotica ed altri settori. Attualmente i maggiori Sistemi Operativi per Smartphone e PC supportano nativamente questa tecnologia, e si stima che entro il 2018 più del 90 percento degli Smartphone con Bluetooth abilitato supporteranno BLE.

Per le sue caratteristiche e per l'interesse a studiare questa tecnologia per la quale si prevede un massiccio utilizzo in molteplici campi in un immediato futuro, si è scelto di utilizzare questa tecnologia per implementare la rete.

6.1.3.1 Il protocollo

Il trasferimento avviene tramite il profilo generico GATT (Generic Attribute Profile), che si basa sul protocollo ATT(Attribute Protocol).

Il protocollo ATT definisce i ruoli di client e server. Il server contiene degli attributi, definiti da tre parametri:

- Type, che ne definisce il significato (temperatura, stato della batteria, informazioni sul dispositivo);
- Handle, un valore che identifica ogni attributo;
- Value, che contiene il valore dell'attributo.

Questi dati sono accessibili dal client attraverso il protocollo, nelle modalità di lettura, scrittura o entrambe.

Il profilo GATT permette tramite questo protocollo la creazione di profili che organizzano la divisione degli attributi in servizi e caratteristiche per facilitarne l'utilizzo. I servizi contengono un insieme di caratteristiche, che a loro volta conterranno un valore accessibile dal client ed altri valori numerici che descrivono la caratteristica.

6.1.3.2 Funzionamento

Il funzionamento di un dispositivo BLE piò essere descritto come una macchina a 5 stati.

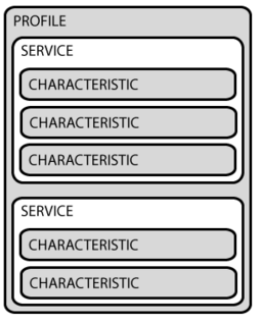

Figure 6.5: Struttura di un profilo GATT.

La periferica una volta iniziato il processo di advertising e impostato lo stato a "discoverable", diventa visibile ad altri dispositivi che supportano Bluetooth 4.0, ed è possibile connettersi ad essa. Dal momento della connessione la periferica che ha richiesto la connessione potrà accedere alle caratteristiche GATT. Ogni dispositivo BLE può essere connesso con una sola periferica per volta, quindi dal momento della creazione della connessione la periferica non sarà visibile da altri, fin quando la connessione non sarà terminata. I servizi e le caratteristiche BLE vengono identificati tramite un ID a 128 bit. Alcuni servizi tra i più comuni (Healt Care Service, Battery Supply, etc.) hanno associato un valore ID secondo lo standard proposto da Bluetooth SIG, oppure possono essere personalizzati dagli sviluppatori per lo specifico progetto.

La periferica, in quanto contiene al suo interno i dati dei servizi e caratteristiche associati ad essa, sarà chiamata Server GATT, mentre il dispositivo, come smartphone o tablet, che richiede la connessione sarà il Client GATT.

Vengono anche distinti i ruoli di central e peripheral. Il dispositivo definito central è quello che scansiona e cerca advertisement, mentre quello peripheral è chi si occupa di generare gli advertisement.

Il protocollo Security Manager dello standard Bluetooth Low Energy utilizza l'algoritmo per la criptazione autenticata CBC-MAC (Cipher Block Chaining Message Authentication Code), che implementa l'algoritmo di cifratura a blocchi AES (Advanced Encryption Standard).

Il numero massimo di dispositivi collegabili ad unico dispositivo Bluetooth 4.0 è 7, e questo pone un limite nell'estensione massima nella rete così come è progettata.

6.2 Implementazione

Il sistema che si vuole sviluppare soddisfare i requisiti del il sistema ciberfisico descritto nei capitoli precedenti, implementando una soluzione specifica per le risorse tecnologiche impiegate.

6.2.1 Attivazione e training dei dispositivi

Le schede Nucleo ST devono essere programmate per acquisire dati dall'ambiente allo scopo di impararne il comportamento e saperne controllare la stabilità. Come descritto nel capitolo 4, il dispositivo campiona i valori dei sensori stabiliti necessari per il calcolo dei parametri del modello AR corrispondente. Successivamente stima i residui di riferimento, su cui calcola media e varianza necessari per i test di ipotesi, e configura l'ICI based CDT, configurando il test per ogni sensore, specificando i valori di riferimento, la dimensione della finestra su cui verrà eseguito il test, ed il parametro Γ relativo all'intervallo di confidenza.

6.2.2 Connessione

Una volta che i dispositivi Nucleo ST avranno finito la loro fase di inizializzazione saranno pronti per andare a costituire i nodi della rete, ed imposteranno il loro indirizzo Bluetooth visibile. Lo smartphone avviando l'applicazione responsabile al loro controllo scansionerà il territorio e selezionerà i dispositivi interessati. Dati i fattori che possono influenzare la visibilità dei dispositivi, è possibile ripetere la scansione per la ricerca dei dispositivi mancanti.

Come descritto nell'approccio metodologico, ora bisogna dare forma alle tre fasi necessarie per la costituzione della rete: creazione delle connessioni, calcolo del grafo delle dipendenze e sincronizzazione. I dispositivi lato client e lato server devono avere uno standard comune per comunicare e conoscere il significato dei byte ricevuti. Nell'applicazione sviluppata il riconoscimento avviene grazie alla lunghezza del messaggio o a dei byte identificativi. Le librerie utilizzate per la connessione dalle board Nucleo permettono di inviare fino ad un massimo di 20 byte per invio.

Lo smartphone avvia questa fase scansionando i dispositivi Bluetooth visibili nel suo raggio di azione, ed individua quelli che devono far parte della rete in base al nome. Una volta individuati e selezionati i nodi della rete l'utente dà l'istruzione allo smartphone di formare la rete, inviando ad ognuno una richiesta di connessione. Questa viene ricevuta da ogni STM32 ed accettata. Da questo momento il nodo non è più visibile da altri dispositivi Bluetooth, ed è connesso alla rete. Il dispositivo Android lo inserisce nella liste dei BluetoothGatt connessi alla rete ed istanzia un oggetto in ascolto delle comunicazioni provenienti da esso. In caso l'utente abbia selezionato oggetti Bluetooth estranei ai dispositivi programmati, questi non saranno accettati dal coordinatore in quanto in fase di connessione in quanto privi dei servizi impostati per la comunicazione e riconosciuti dallo smartphone.

Il secondo passo è la creazione del grafo delle dipendenze per individuare le possibili relazioni all'interno della rete.

Quando i dispositivi ricevono la richiesta di connessione dallo smartphone, come descritto in precedenza, dovranno inviare i trend di dati acquisiti dai sensori. In

un primo invio informerà lo smartphone sulla tipologia di fenomeno monitorato, inviando un array di byte contenente i valori identificativi di quei parametri in formato intero, concatenati nell'ordine in cui verranno successivamente inviati i dati. Quindi se, ad esempio, il dispositivo sta monitorando 3 sensori, invierà un messaggio di 12 byte, dove i primi 4 byte rappresentano l'identificativo del primo sensore, i successivi 4 quello del secondo, e gli ultimi il terzo. Lo smartphone saprà che la ricezione di un messaggio con lunghezza multipla di 4 è il primo messaggio della trasmissione dati, e predisporrà le strutture per riceve le successive comunicazioni. Successivamente i dispositivi invieranno tutti i dati contenuti dal buffer. Con la stessa logica di prima, si crea un array di byte contenente i valori dei diversi sensori in formato float, quindi ancora 4 byte ciascuno, nello stesso ordine in cui sono stati inviati gli id precedenti. I dispositivi immagazzineranno questi dati in diverse liste, per poi eseguire l'analisi sulle correlazioni. I nodi notificano la fine dei dati con l'invio di un byte contenente un valore speciale, che informa il coordinatore che i dati sono terminati e che deve comunicare l'ora di inizio connessione per il coordinamento e le correlazioni calcolate.

In un primo messaggio lo smartphone dovrà quindi comunicare ad ogni dispositivo le proprie dipendenze interne. Per fare questo si utilizza una trasmissione di 8 byte. Di questi byte l'i-esimo conterrà il valore corrispondente al suo i-esimo sensore, che sarà qui caratterizzato dalla posizione occupata. L'idea è quella di comunicare per ogni sensore i un valore binario dove il j-esimo bit indica la relazione tra i sensori i e j. In caso di relazione varrà 1, altrimenti 0.

Lo smartphone ha il compito di comunicare ad ogni unità le proprie dipendenze interne, calcolando per ogni sensore i l'array di byte che rappresenta le dipendenze come

$$byte_i = \sum_{j=0}^{n} (2^j \cdot \chi(s_i, s_j))$$

dove $\chi(s_i, s_j)$ è la funzione caratteristica che indica se i sensori i e j sono correlati tra loro, e vale

$$\chi(s_i, s_j) = \begin{cases} 1 & se\ |Pearson(s_i, s_j)| \geq soglia \\ 0 & altrimenti \end{cases}.$$

Quando il dispositivo riceverà questi byte riconoscerà dalla lunghezza che tipo di informazione è contenuta nel messaggio e in modo inverso scomporrà il valore di ogni byte per calcolare e salvare le dipendenze interne per ogni paramentro misurato.

In Figura 6.6 è visibile un esempio di grafo delle dipendenze creato in un sistema composto da quattro unità. Per ogni coppia di sensori per cui è stato calcolato un indice di Pearson superiore alla soglia stabilita è stato aggiunto un arco. Una volta che per ogni dispositivo ne sono note le relazioni, lo smartphone elabora i valori da inviare secondo la logica spiegata prima. Ad esempio, per Device 1 esistono relazioni tra i sensori 1 e 2 e tra 2 e 3. Ovviamente ogni sensore risulta anche in relazione

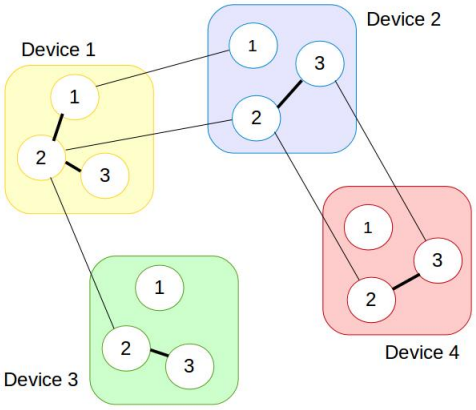

Figure 6.6: Esempio di grafo delle dipendenze in un sistema composto da quattro dispositivi, ciascuno dotato di tre sensori.

con se stesso. La matrice di correlazione relativa sarà $C_{i,j}^{Device1} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}$, dove l'elemento di posizione $c_{i,j}$ indica la relazione tra i sensori i e j. Dalle relazioni espresse in matrice si può facilmente calcolare i valori corrispondenti alle relazioni, contenuti nell'array $b_{Device1} = [3, 7, 6]$. Quando Device 1 riceverà questi valori scomporrà i valori relativi ad ogni sensore per ricavarne le relazioni.

L'ultimo passo che resta da fare è inviare un tempo comune per sincronizzare tutto il sistema. L'invio deve essere la rappresentazione dell'orario attuale in secondi, quindi un valore intero che necisiterebbe 4 byte. Per disambiguare dal caso di invio del trend di dati di un dispositivo con un solo sensore si aggiunge un quinto byte, contenente un valore speciale per indicare che i byte precedenti rappresentano l'orario iniziale da impostare.

Terminata la fase di sincronizzazione i nodi sono connessi ed operativi con la rete, ed abilitati allo scambio di informazioni.

6.2.3 Esecuzione

Terminata la configurazione l'STM32 è pronto per monitorare i fenomeni fisici che gli sono stati impostati. Ciclicamente acquisisce le nuove misure ed ogni volta che ha un numero sufficiente di nuovi campioni esegue il Change Detection Test, aggiornando la struttura dati che ne memorizza la dimensione dell'insieme.

Quando l'insieme dell'ICI based CDT è vuoto il dispositivo rileva il cambiamento, che viene validato dal test di ipotesi; se il dispositivo è già stato connesso potrà sfruttare il test T^2 di Hotelling verificando anche altri sensori in relazione, altrimenti eseguirà il paired t-test.

In caso di dispositivo connesso, il nodo dovrà informare il coordinatore per inoltrare la richiesta a tutti quelli con cui è in relazione. Le informazioni da inviare sono il tempo T^* a cui ha rilevato il cambiamento ed il sensore s coinvolto. Il messaggio sarà costituito da 5 byte, di cui i primi 4 per l'istante di tempo di cambiamento, e l'ultimo contenente l'identificativo di s. Lo smartphone, ricevendo queste informazioni, interpreta il messaggio ed in base al nodo che ha inviato il messaggio e ad s identifica le relazioni coinvolte, e provvede ad inoltrare un messaggio ai dispositivi interessati nello stesso formato del messaggio appena ricevuto, ma sostituendo al quinto byte l'identificativo del sensore coinvolto relativo al dispositivo in questione.

Ricevuti questi messaggi i dispositivi eseguono i test di ipotesi sul sensore interessato, e forniscono una risposta allo smartphone contenente l'esito dell'analisi.

6.2.4 Re-training

I dispositivi Nucleo sono progettati per essere riadattati una volta che la validazione locale abbia dato esito positivo, sia esso sia stato causato dal CDT oppure da richiesta di conferma da parte di un altro nodo. Ogni dispositivo colleziona un numero di campioni sufficiente per stimare un modello AR, ricalcolare i residui di riferimento e riconfigurare l'ICI based CDT ed i test di ipotesi, per poi riprendere la normale esecuzione di controllo. Ovviamento questa fase coinvolge soltanto i sensori che hanno riscontrato un cambiamento, mentre rimangono invariati i restanti.

6.3 Analisi sperimentale

Una volta progettato ed implementato questo sistema, si prosegue con un'analisi sperimentale per verificarne l'effettivo raggiungimento degli obiettivi. Nei test verranno introdotti dei cambiamenti simulati a livello software, in modo da essere pianificati e sapere con certezza la dinamica che il sistema dovrebbe rilevare. L'ICI based CDT verrà eseguito ogni volta che si avrà acquisito il numero sufficiente di campioni su cui eseguire il test. Per questa ragione il ritardo di rilevazione si misurerà in numero di campioni esaminati prima di avere esito positivo.

6.3.1 Analisi su singola unità

In una prima parte della sperimentazione si vuole simulare un comportamento anomalo riguardante un singolo sensore su singolo dispositivo, dove il segnale non segue più la dinamica attesa a causa di tre diversi tipi di guasti, studiando le caratteristiche dei risultati ottenuti. Per questa prima prova si utilizza un solo dispositivo. Va considerato che l'ICI based CDT viene eseguito su una finestra di dati, che sarà la risoluzione massima anche in termini di tempo. Per questa ragione il ritardo verrà

misurato in numero di campioni su cui è stato eseguito il CDT prima di individuare la variazione.

Trattandosi di un sistema reale, che quindi non può essere simulato da software, la campagna sperimentale svolta ha compreso un numero limitato di esecuzioni. Il dispositivo è stato programmato per acquisire i valori di temperatura, umidità ed pressione atmosferica in un ambiente a temperatura costante. Il dispositivo viene fatto funzionare alla frequenza di campionamento di 4 Hz. Quando viene attivato esegue la fase di addestramento, dove nella prima fase, che dura 20 secondi, acquisisce 80 campioni per ogni sensore, che utilizzerà per la stimare i relativi modelli AR di grado 3. Nella seconda parte dell'addestramento per altri 20 secondi acquisisce altri 80 valori, che verranno confrontati con quelli previsti dal modello precedentemente stimato per il calcolo dei residui di riferimento. A questo punto si possono calcolare i parametri necessari ai test di ipotesi e viene configurato l'ICI based CDT, specificando l'array dei residui di riferimento ed il parametro relativo all'intervallo di confidenza Γ e la dimensione W della finestra di valori su cui eseguire il test, nel nostro caso fissata a 20. Finita la fase di training il dispositivo può iniziare il controllo ciclico dell'ambiente, eseguendo il CDT ogni 5 secondi, tempo necessario per aver acquisito 20 campioni.

Le sperimentazioni per ogni tipologia di errore hanno previsto 20 esecuzioni ciascuna, e per un'analisi più accurata ogni tipologia di cambiamento introdotto è stata eseguita due volte, utilizzando diversi valori per il coefficiente Γ dell'ICI based CDT, che rappresenta l'intervallo di confidenza, influendo su ritardi e falsi positivi. Questo parametro varrà nel primo caso 1.5 e nel secondo 3.

Il primo cambiamento simulato è stato di tipo additivo, applicando un aumento a gradino sul parametro temperatura, lasciando gli altri immutati. Si esegue il training del dispositivo in condizioni stazionarie costanti, e si inizia l'esecuzione normale di controllo, acquisendo per il sensore s i valori per i primi 60 valori, senza provocare cambiamento. Dal campione acquisito al tempo $\bar{t} = 65$ si inizia a sommare un valore di offset al valore misurato. Il comportamento del sensore risulta dunque:

$$temp(t) = \begin{cases} getTemperature() & se\ t < \bar{t} \\ getTemperature() + 0.1 & se\ t \geq \bar{t} \end{cases}$$

dove getTemperature() indica la funzione utilizzata per acquisire la temperatura.

Un secondo cambiamento introdotto è quello di tipo moltiplicativo. Come nel caso precedente il dispositivo esegue il training ed inizia la propria regolare attività, ma questa volta dalla quarta finestra di valori il dispositivo assumerà valori amplificati attorno al valor medio del sensore, quindi

$$temp(t) = \begin{cases} getTemperature() & se\ t < \bar{t} \\ getTemperature() + n\left(getTemperature() - \overline{temp}\right) & se\ t \geq \bar{t} \end{cases}$$

dove $\overline{temp}$ rappresenta il valore medio assunto dalla temperatura.

Il test viene eseguito due volte: una con $n = 5$ e $\Gamma = 1.5$, e la seconda con $n = 10$

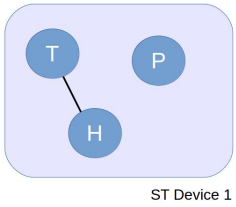

Figure 6.7: Grafo delle dipendenze osservato durante molte esecuzioni in un singolo dispositivo: si è verificato che temperatura e pressione spesso sono risultate in relazione tra loro, ma slegati dalla pressione.

e $\Gamma = 3$.

Il terzo caso proposto è quello noto come stuck-at, con cui si vuole simulare il caso di sensore che si blocca acquisendo sempre lo stesso valore, in particolare gli verrà fatto assumere il valor medio

$$temp(t) = \begin{cases} getTemperature() & se\ t < \bar{t} \\ \overline{temp} & se\ t \geq \bar{t} \end{cases}.$$

Tipologia errore	Γ	**FP**	**DD = 20**	**DD = 40**	**DD = 60**	**DD = 80**	**DD = 100+**
Scalino +0.1	1.5	3	12	2	3	-	-
Scalino +0.1	3	0	5	4	7	-	4
Errore x 5	1.5	4	15	-	1	-	-
Errore x 10	3	1	14	5	-	-	-
Segnale piatto	1.5	3	2	3	5	3	4
Segnale piatto	3	2	3	3	2	2	8

Table 6.1: Risultati statistici in caso di guasto su singolo dispositivo. FP = False Positive, DD = Detection Delay.

Come atteso, con un valore Γ inferiore si diminuisce l'intervallo di confidenza, diminuendo la larghezza degli insiemi ed aumentando di conseguenza la probabilità di avere intersezione vuota, quindi affermare un cambiamento. Complessivamente su un singolo dispositivo si è verificato che il valore di $\Gamma = 1.5$ è soggetto ad un numero elevato di Falsi positivi (10 su 60), e questo valore scende utilizzando nel CDT un intervallo di confidenza doppio (3 su 60). Come atteso, a fronte dell'elevata percentuale di falsi positivi, si nota in tutti i casi studiati una maggiore rapidità nell'individuare i cambiamenti. Dai dati rilevati sul singolo dispositivo si può inoltre notare che il caso di errore moltiplicativo viene rilevato più facilmente, mentre in caso di stuck-at richiede mediamente tre finestre di dati per essere individuato, dato che il valore medio campionato rimane nell'intervallo considerato dall'ICI, che viene assottigliato lentamente.

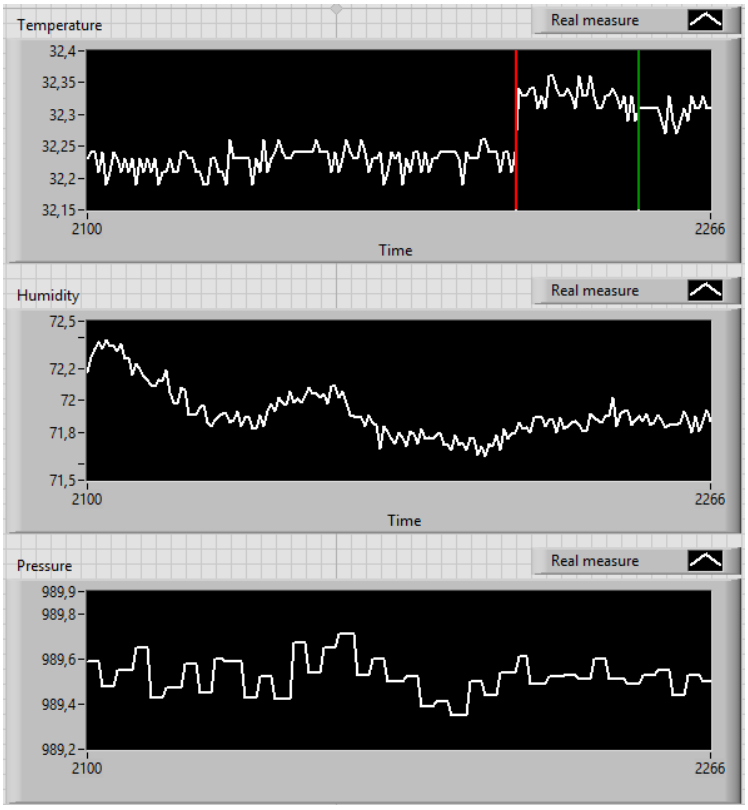

Figure 6.8: Grafici acquisiti durante l'esecuzione del cambiamento additivo sulla temperatura e visualizzati mediante NI LabVIEW: sono presenti i tre sensori monitorati: temperatura, umidità e pressione atmosferica. In rosso è indicato l'istante di cambiamento, mentre in verde l'istante in cui viene eseguito il Change Detection Test.

6.3.2 Analisi su Sistema ciberfisico

Nella seconda parte della fase sperimentale si vuole analizzare un sistema ciberfisico funzionante, quindi che attui lo scambio di comunicazioni quando richiesto. Nel caso specifico sono stati utilizzati due dispositivi programmati con le stesse caratteristiche del caso precedente, ma questa volta monitorando soltanto la temperatura. Avendo a disposizione soltanto una scheda di espansione MEMS, un dispositivo ha monitorato realmente l'ambiente, mentre sull'altro è stato simulato un comportamento costante del valore acquisito, con una piccola componente di rumore casuale.

L'analisi vuole studiare due casi:

- guasto di un sensore: verrà programmato un dispositivo per acquisire normalmente i valori letti dai sensori, mentre il secondo si comporterà come nei casi esposti nella sezione precedente, mutando il proprio comportamento ad un certo istante e rilevando un cambiamento che non deve essere confermato globalmente;
- cambiamento ambientale: in questo caso la modifica di un fenomeno fisico

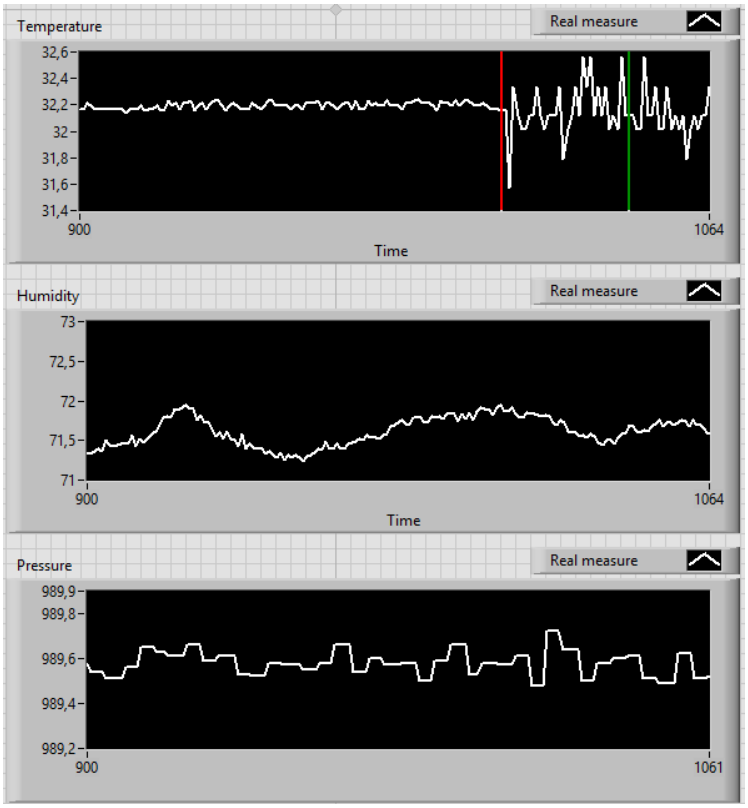

Figure 6.9: Esempio di esecuzione con cambiamento moltiplicativo.

influenzerà i sensori di tutti i dispositivi. Per simulare il cambiamento ambientale vengono programmati i dispositivi per introdurre contemporaneamente uno specifico cambiamento letto dal sensore.

Per entrambi i casi sono stati introdotti i cambiamenti di tipo a gradino e moltiplicativo, come descritti al paragrafo precedente.

Tipologia errore	Γ	**FP (No confirm)**	**DD = 20**	**DD = 40**	**DD = 60**	**DD = 0 (No confirm)**
Scalino +0.1	1.5	4	13	2	-	1
Scalino +0.1	3	1	11	7	1	-
Errore x 5	1.5	1	14	3	-	2
Errore x 10	3	0	15	3	-	2

Table 6.2: Test effettuati sul sistema ciberfisico composto da due nodi in caso di cambiamento ambientale. FP = False Positive, DD = Detection Delay, No confirm = risultato non confermato dall'altro dispositivo.

Nel caso in cui il cambiamento rilevato debba essere visto da entrambi i dispositivi, tutti i casi di falso positivo rilevati dal dispositivo A non sono stati confermati dal B. Avendo due nodi predisposti per introdurre una variazione nel valore di temperatura acquisito allo stesso istante, il cambiamento viene relevato da almeno uno

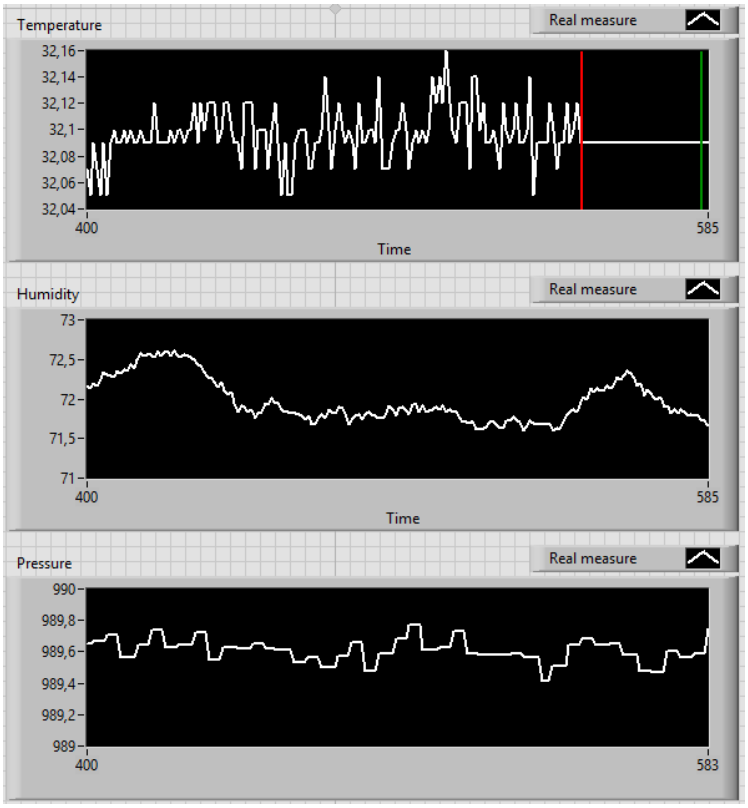

Figure 6.10: Esempio di esecuzione con cambiamento stuck-at.

mediamente più velocemente che utilizzando un singolo dispositivo. Su 80 esecuzioni di questi test in 74 casi è stato rilevato il cambiamento, ed in 73 volte confermato dal test di ipotesi sul secondo.

Nel caso di cambiamento riguardante il solo dispositivo A, i falsi positivi come nel caso precedente non hanno trovato riscontro, così come non sono stati confermati i cambiamenti rilevati da A e non confermati da B.

Inoltre, come visibile dai grafici in Figura 6.13, utilizzando due dispositivi predisposti al monitoraggio di uno stesso fenomeno fisico che subisce una variazione, il tempo di rilevazione medio risulta inferiore che nel caso di cambiamento visibile ad un solo dispositivo. Questo avviene perché c'è una doppia probabilità di individuarlo e ricevere conferma dal secondo. Da questa immagine è anche visibile come al crescere dei campioni successivi al cambiamento il sistema è sempre capace di rilevare correttamente il cambiamento, facendo tendere a zero la probabilità di cambiamento non rilevato.

Complessivamente si può dire che risultati migliori si sono avuti utilizzando come parametro dell'intervallo di confidenza del CDT un valore Γ maggiore di 1.5, in modo da diminuire la percentuale elevata di falsi positivi. I test di ipotesi utilizzati si sono dimostrati affidabili, confermando o rifiutando le ipotesi adeguatamente nella quasi totalità dei casi studiati. Nel complesso il sistema si è dimostrato capace di

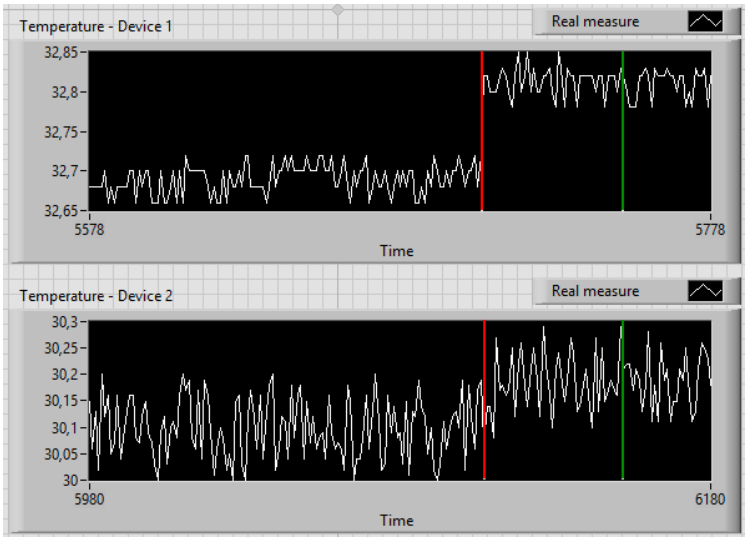

Figure 6.11: Esempio di valori acquisiti dai due dispositivi in caso di cambiamento ambientale additivo. Device1 è il dispositivo reale, mentre Device 2 quello che simula l'acquisizione da sensore. Il cambiamento viene confermato da entrambi i sensori.

Tipologia errore	Γ	**FP**	**DD = 20**	**DD = 40**	**DD = 60**	**DD =80+**
Scalino +0.1	1.5	1	15	4	-	-
Scalino +0.1	3	-	8	6	2	4
Errore x 5	1.5	2	16	2	-	-
Errore x 10	3	-	15	3	1	1

Table 6.3: Test effettuati sul sistema ciberfisico composto da due nodi in caso di cambiamento relativo ad un solo dispositivo. FP = False Positive, DD = Detection Delay. L'ipotesi di cambiamento non è stata mai confermata dall'altro dispositivo.

imparare autonomamente le caratteristiche dei fenomeni fisici da monitorare, capace di rispondere alle variazioni che gli sono state applicate permettendo di specificare la natura del cambiamento rilevato, confermando un cambiamento ambientale oppure osservando un comportamento anomalo del sensore. L'alta percentuale di risposte corrette ricevute dai test di ipotesi dei dispositivi correlati a quello che rileva il cambiamento fa pensare che, aumentando il numero dei nodi della rete e potendo sfruttare un maggior numero di relazioni, un eventuale classificatore implementato su questo framework avrà a disposizione un maggior numero di informazioni attendibili su cui basare la propria analisi.

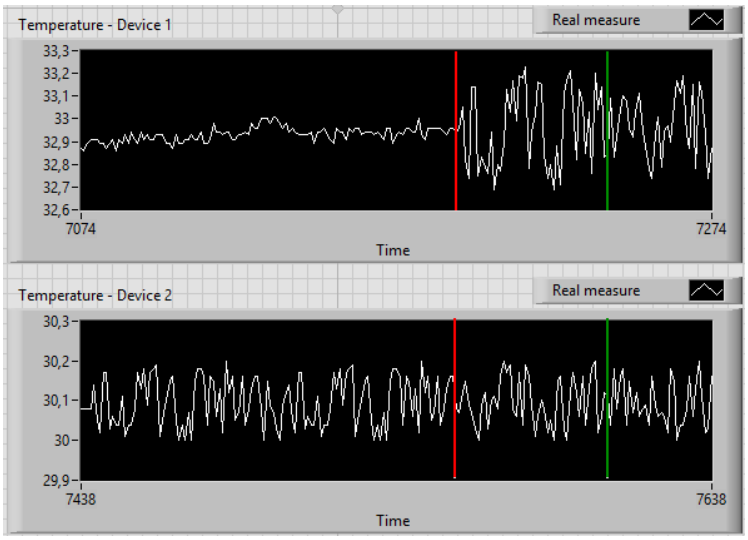

Figure 6.12: Esempio di esecuzione con cambiamento moltiplicativo riguardante solamente Device1. Il cambiamento rilevato non viene confermato da Device2, il cui comportamento è paragonabile a quello assunto in precedenza.

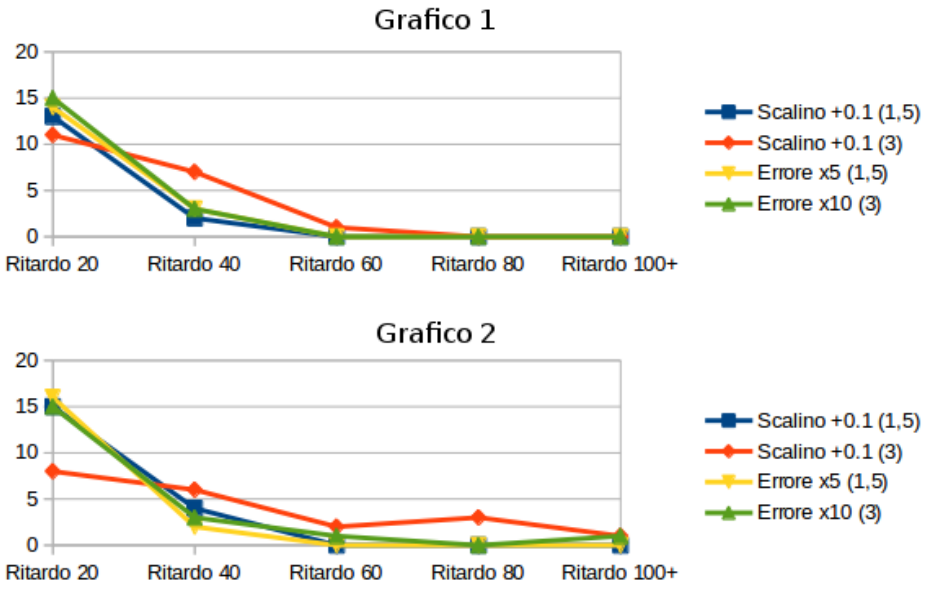

Figure 6.13: Confronto tra le due simulazioni a livello di rete: in Grafico 1 il test in caso di cambiamento ambientale; in Grafico 2 il test su dispositivi con comportamenti differenti. Si può vedere la rapidità di rilevamento del sistema, leggermente migliore nel caso di cambiamento visto da entrambi i dispositivi.

Conclusioni

Il lavoro sviluppato in questa tesi si pone nell'ambito dei Cyber-physical Systems, cioè un insieme di dispositivi progettati per essere integrati nell'ambiente per monitorarne, attraverso sistemi informatici, il comportamento. In letteratura sono presenti esempi di sistemi autoadattativi che possono essere applicati a fenomeni fisici. In questo lavoro è stato orientato verso la progettazione di un sistema dotato di meccanismi intelligenti, con la capacità di imparare autonomamente il comportamento di fenomeni fisici non noti a priori. Inoltre sfruttando le informazioni note riguardanti la rete si è in grado di distinguere comportamenti dovuti ad una variazione delle condizioni ambientali rispetto a quelli dovuti da comportamenti anomali assunti da determinati sensori, che non rispecchiano il comportamento rilevato dagli altri sensori posti a monitorare le caratteristiche dello stesso fenomeno fisico potrebbero essere soggetti a guasto.

È stato trattato inizialmente il problema a livello teorico, formulando una possibile soluzione che sappia fornire al sistema l'intelligenza per svolgere i propri compiti. È stato posto l'accento sulla differenza tra i dispositivi, dotati di un intelligenza a livello di rete, e l'intero sistema, che può sfruttare l'unione delle informazioni provenienti dalle diverse unità, con un'intelligenza a livello di rete. I singoli nodi sono dei microcontrollori dotati di modulo connettivo e modulo sensoristico, che possono ricevere dati dall'ambiente, elaborarli e comunicare informazioni. Sono stati programmati per elaborare i modelli relativi ai fenomeni fisici monitorati e verificare i cambiamenti che li interessano con l'ausilio dell' ICI based Change Detection Test e la successiva validazione attuata mediante test di ipotesi. I dispositivi sono connessi a stella, con un coordinatore che gestisce tutte le comunicazioni del sistema, che in caso un dispositivo dichiari di aver rilevato un cambiamento procede a verificarlo anche a livello di rete innescando controlli su sensori di altri nodi che sono in relazione con il primo.

Successivamente alla formulazione metodologica della soluzione è stata proposta una possibile implementazione basata su dispositivi ST Nucleo ed uno smartphone Android nel ruolo di coordinatore. Verificando il funzionamento del sistema con delle sperimentazioni su casi reali che ne testassero i meccanismi, è emerso che dimensionando adeguatamente il parametro Γ dell' ICI based CDT il sistema risponde

positivamente in tempi ragionevoli, individuando i cambiamenti ove introdotti, e si è visto che sfruttando un numero maggiore di dispositivi vengono ridotti i tempi di rilevazione.

Questo sistema non è stato progettato allo scopo di essere fine a se stesso, ma vuole costituire un framework da cui partire per lo sviluppo di applicazioni ciber-fisiche che necessitino un'integrazione con fenomeni fisici, con la capacità di rilevare cambiamenti inattesi ed all'occorrenza riadattarsi alle nuove condizioni. Questa situazione è verificata in una grande quantità di campi applicativi, basti ricordare gli utilizzi di CPSs descritti in introduzione. Le applicazioni che si basano su questo framework potranno implementare soluzioni specifiche, che sappiano aggregare le informazioni provenienti dai diversi nodi per generare una base di conoscenza ulteriore a quella già introdotta. In altre parole è possibile creare un ulteriore livello di intelligenza, specifica per l'ambito applicativo a partire dalla struttura comune proposta dal framework.

La struttura proposta risente di alcune limitazioni tecnologiche, come la limitatezza dei dispositivi controllabili mediante smartphone. Le specifiche del Bluetooth Low Energy affermano che non è garantito il funzionamento in caso di un numero di connessioni maggiore di sette. Inoltre il raggio di azione è limitato, e spostamenti dello smartphone potrebbero causare momentanee inagibilità da parte di alcuni dispositivi della rete. A partire da queste osservazioni, si potrebbe rielaborare la struttura della rete, predisponendo dispositivi responsabili di aggregare le periferiche connesse e fare da ponte con il coordinatore centrale, utilizzando anche tecnologie che permettano comunicazioni a lunga distanza in modo da abbattere le limitazioni spaziali del territorio monitorato.

Altri miglioramenti possono riguardare anche gli algoritmi utilizzati, ad esempio sostituendo il modello autoregressivo utilizzato per descrivere il comportamento dei fenomeni fisici con tecniche analitiche più sofisticate, come ad esempio le Artificial Neural Network. Il sistema potrebbe essere adattativo anche a livello energetico, attuando sistemi che solo in caso di sospettato evento inatteso aumenti la propria frequenza di campionamento, altrimenti il sistema mantiene un ritmo basso in modo da risparmiare risorse.

Bibliography

[1] Cesare Alippi, Giacomo Boracchi, and Manuel Roveri. A hierarchical, nonparametric, sequential change-detection test. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pages 2889–2896. IEEE, 2011.

[2] Cesare Alippi, Giacomo Boracchi, and Manuel Roveri. A just-in-time adaptive classification system based on the intersection of confidence intervals rule. *Neural Networks*, 24(8):791–800, 2011.

[3] Cesare Alippi, Giacomo Boracchi, and Manuel Roveri. Ensembles of change-point methods to estimate the change point in residual sequences. *Soft Computing*, 17(11):1971–1981, 2013.

[4] Cesare Alippi, Stavros Ntalampiras, and Manuel Roveri. A cognitive fault diagnosis system for distributed sensor networks. *Neural Networks and Learning Systems, IEEE Transactions on*, 24(8):1213–1226, 2013.

[5] Anind K Dey. Understanding and using context. *Personal and ubiquitous computing*, 5(1):4–7, 2001.

[6] Jim Dowling and Vinny Cahill. Self-managed decentralised systems using k-components and collaborative reinforcement learning. In *Proceedings of the 1st ACM SIGSOFT workshop on Self-managed systems*, pages 39–43. ACM, 2004.

[7] Dominik Fisch, Edgar Kalkowski, and Bernhard Sick. Collaborative learning by knowledge exchange. In *Organic Computing-A Paradigm Shift for Complex Systems*, pages 267–280. Springer, 2011.

[8] Jacqueline Floch, Svein Hallsteinsen, Erlend Stav, Frank Eliassen, Ketil Lund, and Eli Gjorven. Using architecture models for runtime adaptability. *Software, IEEE*, 23(2):62–70, 2006.

[9] David Garlan, Shang-Wen Cheng, An-Cheng Huang, Bradley Schmerl, and Peter Steenkiste. Rainbow: Architecture-based self-adaptation with reusable infrastructure. *Computer*, 37(10):46–54, 2004.

[10] Kurt Geihs, Roland Reichle, Michael Wagner, and Mohammad Ullah Khan. Modeling of context-aware self-adaptive applications in ubiquitous and service-oriented environments. In *Software engineering for self-adaptive systems*, pages 146–163. Springer, 2009.

[11] Marcus Handte, Gregor Schiele, Verena Matjuntke, Christian Becker, and Pedro José Marrón. 3pc: System support for adaptive peer-to-peer pervasive computing. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 7(1):10, 2012.

[12] Jeffrey O Kephart and David M Chess. The vision of autonomic computing. *Computer*, 36(1):41–50, 2003.

[13] Christian Krupitzer, Felix Maximilian Roth, Sebastian VanSyckel, Gregor Schiele, and Christian Becker. A survey on engineering approaches for self-adaptive systems. *Pervasive and Mobile Computing*, 17:184–206, 2015.

[14] Jeff Magee and Jeff Kramer. Dynamic structure in software architectures. *ACM SIGSOFT Software Engineering Notes*, 21(6):3–14, 1996.

[15] BFJ Manly. Exponential data transformations. *The Statistician*, pages 37–42, 1976.

[16] Philip K McKinley, Seyed Masoud Sadjadi, Eric P Kasten, and Betty HC Cheng. Composing adaptive software. *Computer*, (7):56–64, 2004.

[17] Arun Mishra and AK Misra. Component assessment and proactive model for support of dynamic integration in self adaptive system. *ACM SIGSOFT Software Engineering Notes*, 34(4):1–9, 2009.

[18] Matthias Rohr, Simon Giesecke, Marcel Hiel, Willem-Jan van den Heuvel, Hans Weigand, and Wilhelm Hasselbring. A classification scheme for self-adaptation research. 2006.

[19] Mazeiar Salehie and Ladan Tahvildari. Self-adaptive software: Landscape and research challenges. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 4(2):14, 2009.

[20] Hartmut Schmeck, Christian Müller-Schloer, Emre Çakar, Moez Mnif, and Urban Richter. Adaptivity and self-organization in organic computing systems. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 5(3):10, 2010.

[21] Gerald Tesauro, David M Chess, William E Walsh, Rajarshi Das, Alla Segal, Ian Whalley, Jeffrey O Kephart, and Steve R White. A multi-agent systems

approach to autonomic computing. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 464–471. IEEE Computer Society, 2004.

[22] Thomas Vogel, Stefan Neumann, Stephan Hildebrandt, Holger Giese, and Basil Becker. Model-driven architectural monitoring and adaptation for autonomic systems. In *Proceedings of the 6th international conference on Autonomic computing*, pages 67–68. ACM, 2009.

[23] Danny Weyns and Michael Georgeff. Self-adaptation using multiagent systems. *Software, IEEE*, 27(1):86–91, 2010.

[24] Danny Weyns, Bradley Schmerl, Vincenzo Grassi, Sam Malek, Raffaela Mirandola, Christian Prehofer, Jochen Wuttke, Jesper Andersson, Holger Giese, and Karl M Göschka. On patterns for decentralized control in self-adaptive systems. In *Software Engineering for Self-Adaptive Systems II*, pages 76–107. Springer, 2013.