Università degli Studi di Padova

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE
Corso di Laurea Magistrale in Ingegneria dell'Automazione

Valutazione sperimentale di reti e protocolli orientati alla Safety

Il protocollo SafetyNET p

Laureando:
Riccardo Ghirardello

Matricola 607377-IAM

Relatore:
Prof. Stefano Vitturi

Correlatori:
Prof. Adriano Valenzano
Ing. Marco Cereia

Anno Accademico 2010-2011

a Carlotta,

"Siamo angeli con un'ala sola,
solo restando abbracciati
possiamo volare."

Luciano De Crescenzo

Ringraziamenti

Intendo porgere i miei più sentiti ringraziamenti a tutte le persone che hanno, in vari modi, reso possibile tutto questo.

Al **prof. Stefano Vitturi** un sentito grazie per avermi permesso di svolgere lo stage presso la sede torinese dell'istituto IEIIT del CNR e per la grande disponibilità offerta durante il periodo di tesi.

Ringrazio sinceramente il **prof. Adriano Valenzano**, il **prof. Gianluca Cena** e il **dir. Riccardo Tascone**.

Un ringraziamento particolare è rivolto all'**ing. Marco Cereia** che mi ha seguito durante tutte le analisi sperimentali effettuate e la stesura dell'elaborato. Il risultato complessivo è stato in parte ottenuto anche grazie ai frequenti confronti avuti, decisamente costruttivi.

Un grazie è rivolto anche a tutte le persone conosciute presso la sede dell'IEIIT di Torino che sono state molto disponibili e gentili nei miei confronti, senza farmi assolutamente pesare la condizione di estraneo sia rispetto al Politecnico di Torino che al Piemonte e alla città di Torino. In particolare, il **prof. Claudio Zunino**, il **prof. Luca Durante**, il **prof. Ivan Cibrario Bertolotti**, **Stefano Scanzio**, **Manuel Cheminod** e **Tingting Hu**.

Tornando all'ambito patavino, un grazie particolare al **prof. Luca Schenato**, per aver contribuito alla pubblicazione dell'articolo scientifico, e ai compagni di corso, **Fabio Paggiaro** e **Roberto Guiotto**, con cui esso è stato realizzato, per l'enorme simpatia e per la collaborazione. Grazie anche a **Federico Cerruti**, **Giacomo Marangoni**, **Chiara Masiero** per il cameratismo e, in particolare, **Edoardo D'elia** per i tanti passaggi in auto offerti.

Passando ai ringraziamenti che coinvolgono le persone a me care, desidero ringraziare per prima la mia fidanzata **Carlotta**. Spiegare quanto la ami e quanto le sia grato è veramente difficile in poche parole, per cui la ringrazio semplicemente per essere entrata nella mia vita e per aver realizzato un mio primario obiettivo andando ad abitare insieme. Stare con te è la cosa più bella che ci possa essere. Grazie anche per le correzioni ortografiche e di forma che, come sempre, ti offri di

dare a tutti gli elaborati importanti che devo produrre. La ringrazio, infine, per la decisione, presa insieme, di adottare un piccolo cane fulvo, con le orecchie lunghe e pelose e la coda nera, di nome **Simba**. L'arrivo di questa piccola bestiolina mi ha davvero tranquillizzato, facendomi sfogare tanta ansia per la stesura della tesi, durante le molte passeggiate e riempiendomi di gioia per i pianti e le feste che fa quando torno a casa.

Un grazie di cuore ai **miei genitori**, **Emiliana** e **Valter**, i quali, con moltissimi sacrifici, mi hanno permesso di giungere fino a dove sono arrivato: sia in termini di percorso di studi che in termini geografici, per essermi stanziato a Torino. Li ringrazio per essermi stati sempre vicini, avermi sempre spronato a non mollare mai e per aver creduto e investito in me tempo e denaro. Spero vivamente di essere d'ora in poi più indipendente.

Un grazie speciale a mia **sorella Elena** per l'affetto e per farmi sentire importante, vedendomi sempre come un punto di riferimento a cui chiedere aiuto. Sono orgoglioso di lei perchè ora, dopo gli esami di maturità, la vedo cresciuta e oramai adulta.

Ringrazio mia **nonna Rina** perchè è sempre stata orgogliosa di me e sempre lo sarà, per il sostegno economico e per l'affetto incondizionato che mi ha sempre dimostrato.

Ringrazio i **miei nonni**, **Giovanna** e **Giovanni**, per le mance elargite durante tutti questi anni di studi in cui ero sempre in rosso e per l'esempio del loro amore, da più di 50 anni, che mi ha permesso di capire che la cosa più importante la vita è la famiglia.

Ringrazio **Mauro**, **Lucia** per aver concepito una figlia meravigliosa come **Carlotta**. A loro e anche a **Maria**, ringrazio per tutto l'aiuto che ci hanno fornito in questi anni e, soprattutto, per esserci sempre vicini in qualsiasi momento (e per i numerosi, e spero che non ce ne siano altri a breve, traslochi). Grazie, infine, per avermi accettato come una persona di famiglia. Sapete che su di me potete contare sempre.

Un pensiero particolare va anche alle persone care che, purtroppo, non ci sono più e che, secondo me, sarebbero sicuramente state presenti in questo giorno così importante per la mia vota: un saluto a **nonno Bruno** e allo **zio Gianni** o, scherzosamente, *franz l'hom del jazz*.

A tutte le persone care dico grazie di tutto e alla prossima...no, non sarà un'altra laurea ma il mio matrimonio con Carlotta.

Riccardo Ghirardello

Indice

CAPITOLO

1

Introduzione

Le reti industriali di comunicazione permettono il trasferimento delle informazioni, necessarie alla realizzazione del processo produttivo, tra gli apparati costituenti un sistema di automazione. Come avvenuto nel mondo *consumer*, queste reti sono andate incontro ad evoluzione, seppure con un ritmo di sviluppo inferiore; si è così passati dalle reti analogiche a quelle digitali e, in quest'ultimo decennio, la tecnologia più consolidata riguarda l'Ethernet via cavo. Essa garantisce la realizzazione di comunicazioni *real-time* ad alta velocità, con una larghezza di banda adatta alla trasmissione di un'elevata mole di dati. Inoltre, al fine di rispondere alla sempre più crescente esigenza di sicurezza degli impianti in relazione al personale, all'ambiente esterno e interno all'azienda, sono state implementate alcune soluzioni Ethernet che consentono la coesistenza, in un unico canale, sia dei normali dati di processo che di quelli relativi alla *safety*.

I messaggi *safe* si caratterizzano per una struttura contenente le informazioni necessarie a consentire una verifica di integrità, sia del dato inviato sia del canale di comunicazione, rilevando la presenza di eventuali errori di trasmissione. In caso di errore, il sistema, o una sua parte, viene portato in uno *stato safe*, in cui gli azionamenti sono disabilitati, e ciò necessariamente influirà sulla *avaiability* del processo produttivo. E' fondamentale, per tanto, che lo stato di sicurezza venga a realizzarsi soltanto al verificarsi di un effettivo errore; per questo motivo è fondamentale valutare l'adeguatezza delle misure di *detection*, per capire se tali meccanismi sono sufficientemente robusti a discriminare un effettivo errore da uno simulato artificialmente in una comunicazione che si svolge correttamente.

Il protocollo industriale oggetto di studio in questa tesi è SafetyNET p, nel formato di comunicazione RTFN. I dispositivi utilizzati per le verifiche sperimentali, che implementano tale protocollo, appartengono alla famiglia dei sistemi PSS4000 della ditta tedesca di automazione Pilz GmbH & Co.KG.

Si è appurato, preliminarmente, che il protocollo rispettasse le caratteristiche descritte nello standard, andando a studiare le specifiche implementazioni adottate da Pilz GmbH & Co.KG nella gestione delle tempistiche e nell'assegnazione dei codici identificativi PID alle diverse tipologie di messaggi in relazione alle informazioni in essi contenute.

Successivamente, è stata studiata la possibilità di bloccare singole celle produttive, o addirittura, l'intero processo produttivo mediante un attacco esterno alla rete, facente leva sui meccanismi di *detection* dei dati *safe*. In particolare, si è cercato di simulare degli errori di comunicazione tramite l'inserimento, nella rete di messaggi anomali che presentassero una struttura in grado di stimolare opportunamente i meccanismi di rilevazione degli errori, ponendosi fra i dispositivi della rete utilizzando l'approccio "*man in the middle*". Una volta verificata l'effettiva possibilità di indurre errori di comunicazione, è stato dimostrato come tutto questo sia possibile anche in un'ottica di attaccante esterno alla rete. Dato che il protocollo di comunicazione, basato sullo *standard Ethernet*, permette l'interconnessione della rete di processo con quella aziendale, connessa ad Internet, la possibilità di indurre errori sul sistema anche dall'esterno rappresenta un problema importante dal punto di vista economico perchè con l'utilizzo di specifici messaggi può essere possibile compromettere l'*avaiability* dell'impianto produttivo e, dunque, causare una perdita ingente di denaro dovuta alla inattività dell'intero processo.

Per dare un fondamento alle analisi effettuate sono state studiate delle modalità tramite cui venire a conoscenza, dall'esterno della rete, di tutti gli elementi necessari per indurre il blocco di parti del sistema di controllo, senza possedere alcuna informazione sul processo realizzato dai dispositivi. Sulla base di queste modalità e sulla conoscenza del protocollo SafetyNET p è stato possibile descrivere una sequenza di operazioni, chiamata *modello di attacco*, da compiere per disabilitare tutti i dispositivi che trattano dati *safe*. Questo algoritmo può essere realizzato automaticamente da un programma oppure da un operatore, in quanto non necessita di operazioni complesse.

Grazie ad esso, si è dimostrato come i meccanismi di *detection*, posti nella struttura dei messaggi *safe*, siano certamente adeguati per il rilevamento di errori di comunicazione e per impedire il verificarsi di situazioni pericolose. Nello stesso momento, però, essi possono essere un ostacolo all'*avaiability* del sistema nel caso in cui vengano indotti fraudolentemente in un sistema che, in realtà, sta funzionando correttamente e secondo le norme di sicurezza.

In questo contesto è stata analizzata la possibilità di interferire dall'esterno con il funzionamento del sistema mentre si lascia a sviluppi futuri il compito di progettare degli interventi atti a limitare o eliminare tale possibilità.

1.1 Organizzazione della Tesi

Nel Capitolo 2 viene fornito un quadro di riferimento sullo stato dell'arte delle reti di comunicazioni industriali relativamente all'evoluzione storica e alle tecnologie attualmente utilizzate, presentando nel successivo Capitolo 3 le caratteristiche generali di uno specifico protocollo basato sulla *safety*, ossia SafetyNET p.

Si entrerà più nello specifico con il Capitolo 4 che presenta la struttura dei mes-

saggi SafetyNET p più utilizzati nella tipologia di comunicazione RTFN, descritti nello standard del protocollo definito dalle norme IEC 61784-2ed2.0.

Prima di introdurre le analisi effettuate, che rappresentano la parte sperimentale della tesi, vengono introdotti nel Capitolo 5 gli strumenti hardware e software, costituenti l'apparato sperimentale, necessari per il loro svolgimento.

Nel Capitolo 6 vengono presentate le analisi effettuate sul protocollo di comunicazione SafetyNET p per testare l'implementazione nei dispositivi PSSuniversal, appartenenti al sistema PSS4000 di Pilz GmbH & Co.KG, in relazione con le specifiche definite nello standard.

Con le conoscenze acquisite dalle analisi, nel Capitolo 7 è stato possibile sperimentare i meccanismi di *detection*, implementati dal *black channel*, iniettando nella rete del traffico anomalo per simulare degli errori di comunicazione e verificare come i dispositivi si comportino in seguito ad essi.

Nella conclusione della tesi, Capitolo 8, sono state riportate delle considerazioni generali sul lavoro sperimentale svolto e proposti eventuali sviluppi futuri.

CAPITOLO

2

Stato dell'arte

Questo capitolo introduttivo ha lo scopo di fornire un quadro di riferimento relativo allo stato dell'arte delle reti di comunicazioni industriali: l'evoluzione storica, le tecnologie attualmente predominanti e i possibili sviluppi futuri. E', infatti, importante fornire delle conoscenze preliminari al fine di comprendere al meglio l'argomento specifico trattato in questa tesi, ossia l'analisi del protocollo di comunicazione industriale SafetyNET p basato su Ethernet.

2.1 Evoluzione delle reti di comunicazione

In analogia con le continue evoluzioni tecnologiche avvenute in ambito di *networking* nel mondo commerciale, anche per il mondo industriale (controllo di processo e automazione di impianti) si sta assistendo ad una progressiva trasformazione che, a passi molto lenti rispetto all'altra, sta cambiando l'intero panorama industriale. Non si parla soltanto di innovazioni a livello di macchinari ma, soprattutto, a livello di dispositivi di controllo e sistemi di comunicazione.

I fattori che hanno rallentato questa inevitabile evoluzione sono molteplici: la compatibilità con i sistemi proprietari già presenti nell'impianto, lo *switching cost* non irrisorio che comporta un fattore di *lock-in*, per cui ciascuna fabbrica si "affeziona" ad un produttore di sistemi di comunicazioni industriali, e il fatto che prima di adottare un nuovo standard di comunicazione è necessario che la tecnologia con cui è implementato sia ben consolidata.

Per questi motivi, il modo più semplice in cui un nuovo sistema di comunicazione prende piede nel mercato è inserirlo in impianti di nuova costruzione. Questi, ovviamente, sono numericamente inferiori rispetto a tutti quegli impianti esistenti che potrebbero adottare le evoluzioni della tecnologia. Si comprende, quindi, per-

ché il tempo di cui necessita una tecnologia per affermarsi nel campo industriale, è immensamente maggiore rispetto a quello necessaro in ambito commerciale.

All'interno di un sistema di automazione industriale, le reti di comunicazione collegano e permettono lo scambio di dati tra i *Controllori a Logica Programmabile* (*PLC*) e i componenti che costituiscono il processo produttivo (come sensori, attuatori e motori elettrici). L' efficienza della catena di automazione dipende fortemente dall'efficienza della rete di controllo. Per questa ragione, negli anni sono state elaborate diverse soluzioni che permettessero di soddisfare le più esigenti richieste industriali.

Le grandi tipologie di rete, ognuna con i propri specifici protocolli, che si sono susseguite nell'evoluzione delle comunicazioni industriali sono state:

- *Collegamento 4-20mA*;
- *FieldBus*;
- *Ethernet industriale*;
- *Wireless*.

2.1.1 Comunicazione basata su *collegamenti 4-20mA*

Inizialmente tutti i dispositivi erano collegati ad una workstation o ad un PLC che controllava solo piccole parti del processo. Il collegamento tra il PLC e sensori/attuatori era realizzato mediante connessioni di tipo punto-punto, con un doppino intrecciato schermato, in un'architettura di controllo centralizzata.

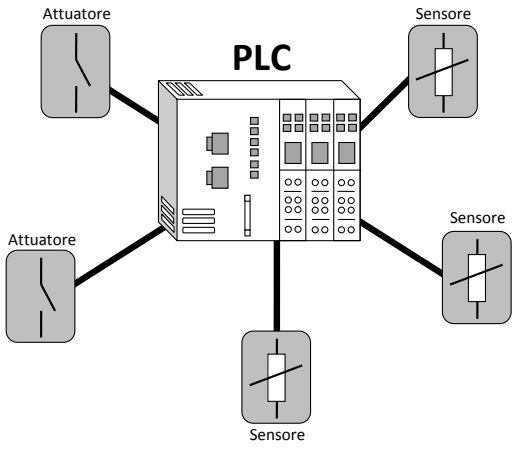


Figura 2.1: principio della connessione 4-20mA

Il collegamento utilizzava un generatore di corrente costante per attenuare i disturbi esterni. In trasmissione, al valore logico basso (bit 0) corrispondeva una corrente di 4 mA mentre al valore logico alto (bit 1) corrispondeva una corrente di 20 mA. Per tale motivo, questa tipologia di comunicazione è detta *collegamento 4-20mA*.

Gli svantaggi di questo sistema di comunicazione industriale analogico, riguardano il cablaggio, molto complesso ed ingombrante, e l'elevato numero di collegamenti in un impianto, che aumenta notevolmente il costo di realizzazione e manutenzione.

2.1.2 Comunicazione basata su *FieldBus*

Dalle problematiche riscontrate con l'approccio del collegamento 4-20mA e dal completamento dello standard ISA S50.02, nasce l'idea di connettere tutti i dispositivi in un unico Bus, detto *FieldBus* (*Bus di campo*).

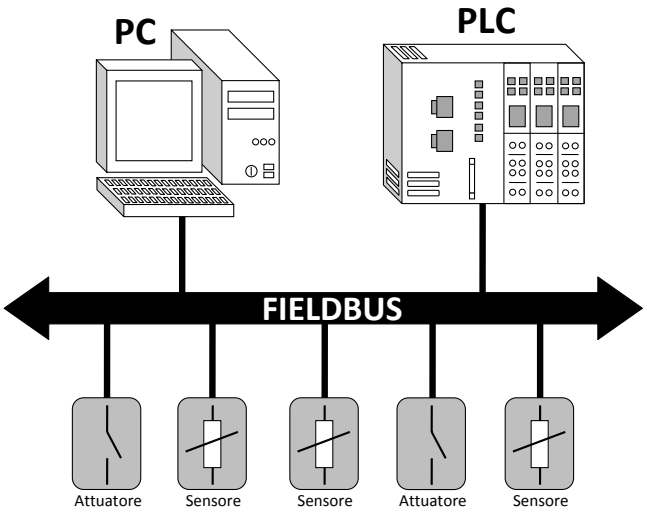


Figura 2.2: principio del FieldBus

I FieldBus sono sistemi di comunicazione industriali digitali utilizzati per collegare strumenti e macchine in un impianto di produzione, all'interno di una architettura di controllo distribuita. In questo modo, il sistema risulta affidabile e può essere facilmente controllato.

All'evoluzione della rete di comunicazione, segue anche l'evoluzione dei dispositivi (sensori e attuatori) che, per essere in grado di comunicare con il sistema di controllo, richiedono un microprocessore che svolga tutte le funzioni di interfacciamento necessarie per un corretto dialogo fra tutti i componenti connessi alla stessa rete.

L'obiettivo dei FieldBus è stato quello di spostare l'interfaccia I/O dal PLC ad un'unità I/O separata, montata vicino al macchinario su cui agisce. Data la separazione tra logica e azionamento, è stato necessario fornire tutti i dispositivi di microprocessore creando così una vera e propria distribuzione dell'intelligenza all'interno della rete. In questo modo, sono possibili elaborazioni locali dei dati sempre più complesse.

A differenza della connessione cablata 4-20 mA, a senso unico dal dispositivo di campo al sistema si elaborazione (PLC), il FieldBus fornisce un bus di campo ad alta velocità e a due vie di collegamento dati, in grado di trasmettere grandi quantità di informazioni. Le informazioni aggiuntive che vengono trasmesse sono i dati

relativi allo stato dei dispositivi, che possono essere utilizzati dal sistema informatico per la diagnostica remota. Questa diagnostica può essere finalizzata ad eseguire manutenzioni predittive e, eventualmente, ad attivare procedure di calibrazione remota sui dispositivi. La manutenzione predittiva è senz'altro un notevole vantaggio offerto da un sistema di comunicazione basato su FieldBus. Utilizzando i dati forniti dai dispositivi attraverso il bus di campo, è possibile prevedere diverse tipologie di problematiche prima che accadano. La manutenzione può quindi essere mirata ed effettuata quando effettivamente necessaria, evitando così manutenzioni periodiche (note come *manutenzioni preventive o calendariali*), non necessarie per la maggior parte dei dispositivi, e *manutenzioni reattive*, cioè effettuate dopo l'effettivo danneggiamento di un componente di rete. Il principale vantaggio della manutenzione predittiva è quello di evitare tempi di inattività del sistema a cui viene applicata, dovuti all'usura o al guasto dei dispositivi che lo compongono. Se un dispositivo di campo viene ritenuto potenzialmente non affidabile, (spesso) è possibile riparalo o sostituirlo prima che blocchi l'intero processo produttivo.

Questo approccio di rete, rispetto al caso precedente, fornisce comunicazioni veloci e deterministiche e consente la condivisione delle risorse della rete e operazioni di diagnostica remota. Inoltre, permette di ridurre la lunghezza e il numero dei cavi elettrici utilizzati in fase di realizzazione dell'impianto, riducendo i costi relativi al cablaggio sia in termini di manodopera che in termini di materiali. Per contro i prezzi dei dispositivi di campo è maggiore, dato che sono provvisti di logica.

Queste sono le caratteristiche generali delle reti FieldBus ma va precisato che nel corso degli anni sono stati realizzati diversi standard, aperti o proprietari, basati su questa tecnologia.

La maggior parte dei protocolli FieldBus furono sviluppati negli anni '80 e conquistarono definitivamente il mercato durante la seconda metà degli anni '90. Tra quelli più utilizzati vi sono Profibus (*Pro*cess *Fi*eld *Bus*, Siemens, 1989), LonWorks (1999), Interbus (1987) e SafetyBUS p (Pilz GmbH & Co. KG, 1999). Nonostante tutti questi standard si appoggino su FieldBus, essi non sono facilmente intercambiabili. Le differenze riguardano il modo in cui sono stati progettati i diversi protocolli di comunicazione e la metodologia di trasferimento dati; è comunque possibile collegarli insieme ma occorre utilizzare appositi *gateway*. La presenza sul mercato di molteplici e differenti tecnologie basate sul FieldBus ha impedito la creazione di un unico standard internazionale. Ogni protocollo FieldBus offre caratteristiche e prestazioni specifiche; per questa ragione un confronto generale delle prestazioni fra gli standard esistenti è difficile.

Le principali tecnologie di trasmissione utilizzate per i FieldBus sono l'RS232 e l'RS485, cablaggi molto ingombranti e costosi, con una velocità di trasmissione che può variare dai 9600 bit/s ai 12 Mbit/s; essa è selezionata all'atto dell'inizializzazione del sistema ed è la stessa per tutti i dispositivi. E' possibile connettere alcune decine di stazioni per ogni segmento (ad esempio fino a 32 stazioni per segmento nel sistema Profibus) e, in caso di necessità, possono essere collegati più segmenti per mezzo di appositi ripetitori. La lunghezza massima raggiungibile dalla linea dipende dalla velocità di trasmissione che si vuole ottenere.

Ad ogni modo, le soluzioni FieldBus proprietarie sono attualmente utilizzate per applicazioni specifiche e sono presenti ancora in molti impianti.

2.1.3 Comunicazione basata su *Ethernet*

Con l'aumento dei livelli di complessità dei processi industriali, la maggior parte delle organizzazioni di protocolli FieldBus sta lottando per soddisfare le esigenze di interoperabilità delle piattaforme di rete e le prestazioni del sistema.

La decentralizzazione dell'intelligenza richiede un'ampia capacità di calcolo e lo scambio di una maggiore quantità di dati, tra cui sempre più informazioni diagnostiche che permettono alla rete di fare un salto di qualità. Ciò richiede una banda trasmissiva maggiore, ma un FieldBus, tipicamente dedicato alla gestione di semplici sensori ed attuatori, non è in grado di trasportare questa elevata quantità di informazioni. Per tali motivi, negli ultimi anni, i produttori di componenti per l'automazione di processi hanno concentrato la loro attenzione sulla tecnologia Ethernet, al fine di utilizzarla come protocollo a livello *data link* (riferito al *modello ISO/OSI*[1]) per connettere i dispositivi.

In ambito *domestico* e *commerciale* sono installati in tutto il mondo miliardi di dispositivi basati su Ethernet ed è per questo che è diventato uno standard mondiale di fatto. La sua diffusione capillare permette ai produttori di hardware Ethernet di fornire dispositivi con prestazioni sempre più elevate a prezzi competitivi e ad alti livelli di interoperabilità.

Prima di passare a protocolli basati su Ethernet anche in campo aziendale, è stato necessario attendere un'evoluzione dell'Ethernet commerciale che lo rendesse adatto alle esigenze dei processi industriali, dove si deve avere una elevata capacità di trasmissione dati, evitare collisioni di messaggi, adeguarsi alle condizioni lavorative in ambiente industriale e soprattutto fornire un comportamento *real-time*, ovvero quando è indispensabile reagire ad un evento entro una determinata scadenza. Le caratteristiche real-time, in particolare sono richieste da tutti quei sistemi per cui non reagire in un certo intervallo di tempo prestabilito potrebbe non soddisfare le specifiche imposte dall'impianto di produzione, o causare danni al sistema o agli operatori.

Un certo numero di innovazioni prestazionali, come il raggiungimento della velocità di trasmissione dati di 100 Mbit/s, con lo sviluppo dello standard *IEEE 802.3y 100Base-T2 Fast Ethernet*, e il passaggio dalla comunicazione *half-duplex* a quella *full-duplex*, hanno reso l'Ethernet commerciale adeguato alle esigenze dei processi industriali.

La struttura di un frame Ethernet contiene un elevato numero di byte di *overhead*[2] che riduce l'efficienza complessiva della trasmissione. Per questo motivo, soltanto attraverso il raggiungimento della velocità di trasmissione di 100 Mbit/s, è stato possibile ottenere delle velocità di trasmissione dati nettamente migliori ripetto a quelle ottenibili con i FieldBus.

Su questa comune base Ethernet ogni organizzazione, per preservare gli investi-

[1] L'*Open Systems Interconnection*, meglio conosciuto come *modello ISO/OSI*, è uno standard per reti di calcolatori stabilito nel 1978 dall'*International Organization for Standardization*, il principale ente di standardizzazione internazionale, che stabilisce per la struttura logica di rete un'architettura a strati composta da una pila di protocolli suddivisa in 7 livelli, i quali insieme espletano in maniera logico-gerarchica tutte le funzionalità della rete.

[2] Il termine *overhead*, nell'ambito delle reti di comunicazione, si riferisce a quella parte di banda di trasmissione che viene utilizzata per spedire, anziché l'informazione utile, dati aggiuntivi necessari per i protocolli di trasmissione stessa.

menti in software e nelle attrezzature di più basso livello di controllo, ha collocato il proprio protocollo sviluppato per FieldBus. Così, ogni protocollo FieldBus si è evoluto nel suo corrispondente basato su Ethernet (vedi Tabella 2.1[3]).

Protocolli basati su FieldBus	**Protocolli basati su Ethernet**
Profibus	Profinet
LonWorks	Lon over Ethernet
DeviceNet / ControlNet	EtherNet/IP
SafetyBUS p	SafetyNET p

Tabella 2.1: evoluzione dei protocolli nel passaggio da FieldBus a Ethernet

La differenza tra gli approcci disponibili risiede nell'architettura complessiva dei sistemi di comunicazione, nei protocolli delle applicazioni industriali del livello 7 del modello ISO/OSI, nella modellazione degli oggetti, nel modello di progettazione per la configurazione del sistema e anche nella modifica dello *standard Ethernet*. Utilizzare protocolli industriali basati su Ethernet non significa che questi siano compatibili con tutti gli altri protocolli presenti nel mercato mondiale. Infatti, tutti i protocolli Ethernet industriali in commercio possono essere divisi in due categorie differenti: i protocolli basati sullo *standard Ethernet* e quelli basati su soluzioni proprietarie dedicate. La differenza fra queste due tipologie si esprime a livello dell'implementazione degli strati del modello ISO/OSI, come è possibile vedere in Figura 2.3.

Con il termine *standard Ethernet* ci si riferisce solo ai primi 4 livelli del modello ISO/OSI, che compongono le 2 parti: *IEEE 802.3*, alla base, e, sopra, il protocollo *TCP/IP* e *UDP/IP*.

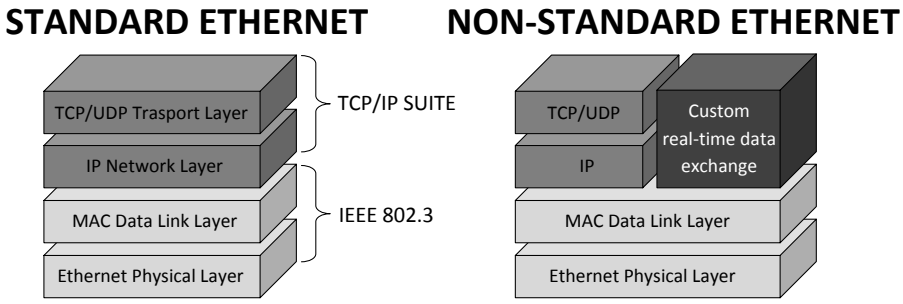


Figura 2.3: Standard e non Standard stack Ethernet

La prima parte di una rete industriale basata sullo *standard Ethernet* è l'*IEEE 802.3*, che fa riferimento a un gruppo di norme che definiscono collettivamente i primi 2 livelli del modello ISO/OSI. Il livello 1, chiamato *physical layer*, specifica i requisiti per tutte le connessioni compatibili e i dispositivi di infrastruttura. Il livello 2, chiamato *data link layer*, specifica il protocollo *MAC* (*Media Acces Control*). Nella sua totalità, l'IEEE 802.3 definisce un formato dei dati per lo spostamento dei pacchetti tra i dispositivi Ethernet.

[3]Nel protocollo EtherNet/IP la sigla IP è l'acronimo di Industrial Protocol.

La seconda parte è costituita dai protocolli *TCP/IP* e *UDP/IP*, e comprende i livelli 3 e 4 del modello ISO/OSI. Lo scopo del protocollo IP è quello di specificare il formato dei pacchetti Ethernet, insieme con il metodo di indirizzamento. I protocolli di trasporto come *TCP* (*Trasmission Control Protocol*) e *UDP* (*User Datagram Protocol*) consentono a due dispositivi sulla rete di scambiarsi dati.

TCP fornisce una comunicazione affidabile perchè tramite meccanismi di *acknowledgement* e di ritrasmissione su *timeout* riesce a garantire la consegna dei dati tra due dispositivi. TCP controlla l'arrivo a destinazione di tutti i pacchetti e può richiedere la ritrasmissione di quelli non ricevuti oppure danneggiati durante la trasmissione. A causa della conferma di ricezione dei messaggi e della loro ritrasmissione in caso di mancato arrivo, TCP non garantisce un determinismo nella consegna dei dati e perciò non è ideale per le applicazioni real-time. Per questo è utilizzato spesso per il trasferimento di dati di diagnostica che non sono ciclici e *time critical*.

UDP, invece, è un protocollo che non utilizza meccanismi per la conferma di ricezione dei messaggi e non garantisce la consegna o il corretto ordine di arrivo dei pacchetti. Per una comunicazione real-time, questo non costituisce un problema, dato che la ritrasmissione di un pacchetto perso non farebbe altro che peggiorare il determinismo richiesto. Al contrario, il fatto che snellisca sensibilmente la comunicazione lo rende adatto per l'utilizzo in comunicazioni real-time.

TCP è un protocollo orientato alla connessione; pertanto, per stabilire, mantenere e chiudere una connessione, è necessario inviare pacchetti di servizio, i quali aumentano l'*overhead* di comunicazione. Al contrario, UDP invia solo i datagrammi richiesti dal livello applicativo.

Molti fornitori di reti di automazione personalizzano uno o più dei suddetti livelli che costituiscono lo *standard Ethernet*, al fine di fornire prestazioni elevate. Qualsiasi soluzione che altera lo strato IEEE 802.3, modifica o ignora il protocollo TCP/IP e UDP/IP non si può ritenere basata sullo *standard Ethernet* .

Attualmente sono disponibili sul mercato mondiale più di una ventina di protocolli di rete industriale Ethernet. Proprio per questo motivo sembra altamente improbabile, in un prossimo futuro, l'esistenza di un unico protocollo applicativo standard basato su Ethernet per l'automazione industriale. Un freno a questa possibile unificazione di standard è dovuto dal fatto che ciascuna azienda lavora per sviluppare i propri prodotti di comunicazione intorno ad un certo protocollo e fatica, poi, ad abbandonare tale tecnologia dopo anni di sviluppo e investimenti. In ogni caso, il fatto che tutte le reti industriali basate su protocolli Ethernet utilizzino la stessa tecnologia di trasmissione rappresenta un vantaggio considerevole rispetto al proliferare di tecnologie di trasmissione tipica dei FieldBus.

Le aziende stanno sempre più espandendo la loro attività a livello globale per affrontare nuove opportunità e ridurre i costi operativi. Il raggiungimento degli obiettivi della globalizzazione e dell'efficienza del processo produttivo richiedono il miglioramento delle interconnessioni tra l'impianto industriale e i sistemi aziendali per la visibilità in tempo reale delle informazioni di processo e per operare una collaborazione efficace. I produttori devono essere in grado di bilanciare la produzione con la necessità di ottimizzare l'utilizzo dei materiali e delle risorse,

pur continuando a garantire e soddisfare i requisiti di qualità dei loro prodotti e la puntualità nella consegna degli stessi.

La rete aziendale, che supporta le tradizionali funzioni amministrative, come la contabilità, gli acquisti, le vendite e le risorse umane, è solitamente basata sulla rete Ethernet commerciale ed utilizza connettività Intranet e Internet. La rete di processo all'inizio non era basata su Ethernet e per tanto necessitava di un *gateway* per tradurre protocolli di applicazioni specifiche in protocolli compatibili alla rete commerciale. Questa traduzione permetteva il passaggio di informazioni tra la rete di processo e l'infrastruttura di rete aziendale ma aveva funzionalità e larghezza di banda limitate. Ora, invece di utilizzare architetture composte da più reti separate, Ethernet permette di unire la parte amministrativa e la parte produttiva di una azienda in un'unica infrastruttura di rete. In particolare si sfruttano a pieno i vantaggi di semplicità di diagnostica e accessibilità rapida alle informazioni offerti da questa tipologia di rete. In questa maniera, dagli uffici commerciali aziendali è possibile, ad esempio, monitorare la situazione del magazzino delle materie prime per ottimizzare il processo di fornitura dei prodotti in esaurimento ma non ancora terminati. Per tanto le aziende riescono a migliorare i processi produttivi, aumentare la produttività e ridurre i costi operativi. E' possibile creare agevolmente di un'unica rete aziendale con tutti i protocolli Ethernet industriali basati sullo *standard Ethernet*.

In commercio, tra tutti i protocolli presenti, solo meno di una decina sono sviluppati sullo *standard Ethernet* mentre gli altri su sue modifiche. Per questo, le soluzioni Ethernet modificate, che alterano questi requisiti per potersi connettere con la rete di gestione aziendale, devono utilizzare dei gateway che ne permettano l'interfacciamento. Ad esempio, le reti proprietarie che non utilizzano i protocolli TCP/IP limitano all'utente finale la possibilità di utilizzare alcuni prodotti dell'*Ethernet commerciale*, come router e alcune tipologie di *firewall*[4]. Questa limitazione annulla uno dei vantaggi fondamentali dello *standard Ethernet* rappresentato dalla facilità, rispetto ai FieldBus, di interfacciare la rete di produzione con quella di gestione aziendale senza l'ausilio di ulteriori dispositivi hardware. Reti come EtherCAT non implementano lo *standard Ethernet* e per questo necessitano dell'utilizzo di hardware proprietari nei dispositivi finali, mentre protocolli come Profinet, Ethernet/IP e SafetyNET p (la comunicazione RTFN nello specifico) lo implementano e dunque consentono l'utilizzo dei generici componenti di rete commerciali.

Un altro vantaggio dell'utilizzo dello *standard Ethernet* è la sua indiscussa compatibilità con le applicazioni commerciali, quali e-mail, browser web, programmi di trasferimento file e strumenti di gestione della rete.

L'integrazione resa possibile da Ethernet fra la rete commerciale e gli impianti di produzione comporta la necessità di proteggere anche quest'ultimi, focalizzando l'attenzione sugli aspetti legati alla *security*, cioè alla protezione delle macchine e dei dati da intrusioni sui sistemi.

In quest'epoca moderna, una delle minacce più gravi, a livello internazionale,

[4]Un *firewall* regola il traffico di rete tra le varie sottoreti. Controlla anche tutti gli aspetti del traffico, anche ispezionando il contenuto dei dati di un pacchetto. I firewall sono applicati nei punti di rischio più importanti nelle reti, per esempio nei punti in cui la rete di processo si interfacciano con la rete aziendale.

per le aziende è quella dello *spionaggio industriale* operato ai danni dei concorrenti. Svolgere l'attività di spionaggio industriale significa spiare le aziende al fine di conoscerne le scelte strategiche e gli sviluppi dei prodotti, giungendo anche ad ottenerne illegalmente i segreti. Scoprire il prodotto di un concorrente è uno dei modi più rapidi ed economici di guadagnare terreno sul mercato nazionale e/o internazionale ed è per questo che molte aziende controllano strettamente le rivali.

La sicurezza non coinvolge solo la riservatezza di informazioni (come formule, progetti e strategie aziendali) ma, data la natura dei moderni impianti industriali che possono essere connessi alla rete aziendale, anche la compromissione della *avaiability* del processo produttivo in atto all'interno dell'azienda. Con il termine *avaiability* si indica il grado con cui il sistema rimane nello stato operativo in un determinato periodo temporale. Viene abitualmente calcolata come rapporto tra la porzione di tempo in cui il sistema resta attivo e la base temporale di analisi.

Così facendo, possono essere creati notevoli danni: dai problemi di produzione o di qualità, ai rischi per l'incolumità degli operatori. Serve, dunque, adottare le misure di security più adatte, a seconda del grado di protezione che si desidera per la rete, come il controllo dell'accesso alla rete e il monitoraggio del traffico per rilevare, ed evitare, minacce e violazioni.

Riassumendo, l'utilizzo di Ethernet come supporto di trasmissione dati in reti industriali garantisce:

- maggiore larghezza di banda, e quindi pacchetti di dati più estesi per la comunicazione con dispositivi industriali sempre più avanzati. Per cui è molto più semplice realizzare strategie di manutenzione predittiva;
- comunicazioni in tempo reale e più veloci;
- connessione di un maggior numero di dispositivi che permette di coprire aree più estese;
- estensione notevole della rete in termini di distanze percorribili con diverse topologie possibili;
- semplificazione dell'installazione e della manutenzione;
- riduzione del costo dei dispositivi;
- possibilità di connessione della rete industriale con la rete locale degli uffici per poter gestire produttività, ordini e magazzino direttamente a livello amministrativo.

La rete basata su Ethernet mira a soddisfare le nuove esigenze di comunicazione industriali con distribuzione dell'intelligenza. In questo approccio, l'applicazione risiede su diversi dispositivi decentralizzati che si connettono via Ethernet.

Il mercato globale per i dispositivi Ethernet è cresciuto notevolmente nel corso degli ultimi anni. Secondo un recente studio svolto dalla ARC Advisory Group il tasso di crescita delle reti Ethernet potrebbe superare il 20% nel 2011, mantenendo un trend positivo fino a oltre il 2013. La tecnologia Ethernet gode di un tale livello di accettazione poichè è facile da capire, distribuire, gestire e mantenere: è a

basso costo, flessibile e supporta un'ampia gamma di topologie di rete. Rispetto alle tradizionali soluzioni basate su FieldBus, che hanno velocità di trasmissione dati tra i 500 Kbit/s e i 12 Mbit/s, la tecnologia Ethernet è in grado di fornire prestazioni notevolmente superiori, con una velocità pari a 100Mbit/s.

	SafetyBUS p	**SafetyNETp**
Dispositivi max in una rete	64	$>$ 64000 Standard 512 Safe
Tempi di reazione massimi	$20ms$	$62.5\mu s$ con RTFL $1ms$ con RTFN
Velocità trasmissione rete	$20 \div 500$Kbit/s	100Mbit/s

Tabella 2.2: confronto delle prestazioni di *SafetyBUS p* e *SafetyNET p*.

2.1.4 Comunicazione *Wireless*

Si stanno affacciando nel panorama delle reti industriali anche le tecnologie *wireless* per la comunicazione di dati senza fili che, come Ethernet, provengono dal mondo delle comunicazioni commerciali.

Su questa tecnologia aleggia ancora un pò di confusione per quanto riguarda la scelta dell'approccio migliore da utilizzare, dato che sono moltissime le varianti di protocolli di comunicazione senza fili. Si arriva, addirittura, a coinvolgere le comunicazioni GSM/Gprs per poter effettuare monitoraggio e assistenza remota anche tramite smartphone.

Tuttavia esistono nel mondo moltissime applicazioni industriali che necessiterebbero di comunicazioni via etere, data l'impossibilità di inserire soluzioni cablate per la presenza di condizioni ambientali estreme.

Il vantaggio di tale soluzione è la diminuzione del costo di messa in opera, l'abolizione dei costi derivanti dal cablaggio della rete e la possibilità di comunicazione con dispositivi posti in zone non raggiungibili con soluzioni via cavo.

Per contro, dal punto di vista economico, un sistema *wireless* è meno conveniente di un sistema cablato per l'alto costo dei singoli dispositivi che lo compongono; inoltre, dato che è effettuata via etere, la comunicazione può essere intercettata ed è, per tanto, necessario risolvere specifiche problematiche relative alla sicurezza dei dati trasmessi. Attualmente la risoluzione di tali problematiche è ancora agli albori perchè gli attuali metodi di cifratura garantiscono si la creazione di comunicazioni sicure ma rendono la comunicazione *wireless* inadatta per applicazioni real-time. Questo perchè le tecniche di cifratura dei messaggi richiedono tempi molto più lunghi di elaborazione, sia in fase di codifica che di decodifica, che impediscono ai dispositivi di intervenire entro certe tempistiche. La comunicazione senza cavi, infatti, non è ancora considerata sufficientemente affidabile per un impiego di tipo industriale o per il controllo di specifiche apparecchiature, nelle quali l'integrità del segnale e il determinismo rappresentano caratteristiche essenziali.

Per questi motivi, ad oggi, le soluzioni cablate possiedono maggior peso nella scelta della tipologia delle rete industriali installate ma, ci si può aspettare, in un futuro, la coesistenza di soluzioni cablate e soluzioni *wireless*, per garantire un più elevato grado di sicurezza dei dati trasmessi e, nel contempo, per poter raggiungere le zone più impervie.

2.2 *Safety*

Al termine italiano *sicurezza* si riferiscono due distinti concetti che, nella lingua inglese, sono espressi con differenti parole: *security* e *safety*.

Il termine *security* corrisponde alla sicurezza delle informazioni contro accessi illeciti mentre il termine *safety* riguarda l'incolumità delle persone, degli oggetti fisici e dell'ambiente circostante. Lo scopo della *safety* è di evitare qualsiasi rischio, diretto o indiretto, di danni alla salute delle persone in conseguenza a guasti dei dispositivi fisici o a perturbazioni del sistema. Nessuno di questi due aspetti della sicurezza può essere valutato senza considerare i sistemi nel loro complesso e l'ambiente con cui interagiscono.

Per i pericoli a cui è esposto, il mondo industriale si è fatto sempre più interessato al problema della sicurezza dei dati sensibili, delle reti di campo, degli apparati di fabbrica nel loro insieme, e del personale umano impiegato a livello del processo produttivo. L'obiettivo è quello di porre tutto il sistema di produzione in uno *stato di safety*, ossia nella condizione in cui il sistema è libero da pericoli e minacce inaccettabili per l'ambiente, i beni e le persone, attribuibili a guasti e malfunzionamenti di hardware e software. Tale sicurezza è mantenuta sia attraverso normative stringenti in fase di progettazione e produzione dei dispositivi di campo, sia mediante specifici protocolli di comunicazione che permettono di dare priorità ai messaggi riguardanti la sicurezza.

La normativa IEC EN 61508 della *Commissione Elettrotecnica Internazionale* (*IEC*) specifica 4 livelli di prestazioni di sicurezza per la riduzione del rischio, chiamati livelli di integrità di sicurezza (*SIL*, *Safety Integrity Level*). Il livello di sicurezza SIL 1 è il più basso e SIL 4 il livello più affidabile. Il SIL è determinato sulla base di una serie di fattori quantitativi, in combinazione con fattori qualitativi, quali i processi di sviluppo e di gestione della sicurezza del ciclo di vita del sistema. Il SIL di un sistema è definito in base a due criteri: l'integrità hardware di sicurezza e l'integrità sistematica della sicurezza. Un sistema deve soddisfare i requisiti di entrambe le categorie per ottenere un dato SIL. In particolare, per l'integrità hardware, il sistema deve possedere determinati valori di massima probabilità di guasto pericoloso (*PFD*, *Probability of Failure on Demand*) per ottenere il corrispondente SIL, come riportato nella Tabella 2.3.

SIL	**PFD**
1	$10^{-5} \div 10^{-6}$
2	$10^{-6} \div 10^{-7}$
3	$10^{-7} \div 10^{-8}$
4	$10^{-8} \div 10^{-9}$

Tabella 2.3: correlazione tra SIL e probabilità di guasto (PFD)

Dal punto di vista strutturale le reti di comunicazione industriale, grazie all'ausilio di protocolli Ethernet, hanno subito una trasformazione. In passato a livello di campo erano presenti due tipologie di reti distinte: la rete di controllo del processo, che trasmetteva i dati necessari alla realizzazione del processo produttivo, e la rete di sicurezza, che implementava tutte quelle funzioni necessarie a tutelare gli operatori, come pulsanti di emergenza, controllo accessi in aree pericolose e azionamento

allarmi. Ora, grazie all'utilizzo di protocolli per la *safety* basati su Ethernet, come Profisafe, Safety over EtherCAT, Ethernet/IP Safety e SafetyNET p, è possibile convogliare sulla rete sia i dati di processo che i dati relativi alla *safety*. Questa integrazione si basa sul concetto di *black channel*.

	Consecutive number (sign of life)	**Time-out** (with receipt)	**Codename** (for sender and receiver)	**Data integrity** (CRC)
Unintended repetition	●			
Loss	●	●		
Insertion	●	●	●	
Incorrect sequence	●			
Corruption				●
Unacceptable delay		●		
Addressing			●	
Masquerade		●	●	●

Tabella 2.4: tipici errori di trasmissione e meccanismi per il loro rilevamento

Il *black channel* consente la comunicazione di dati per la *safety* su canali di trasmissione convenzionali (non ridondati). Tra il *transport layer* e l'*application layer* viene inserito un *livello di safety* che si occupa di effettuare la trasmissione dei dati *safe*, incapsulandoli in una struttura che contiene informazioni necessarie a consentire una verifica di integrità sia del dato inviato che del canale di comunicazione. Il vantaggio di questa tecnica è che la modalità di trasmissione all'interno della rete è sempre la stessa e le informazioni riguardanti la *safety* vengono viste dai dispositivi *non safe* come normali dati che attraversano la rete. Al contrario, i dispositivi *safe* sono in grado di interpretare le informazioni aggiuntive trasmesse insieme al dato per verificare che durante la trasmissione non ci siano stati errori, ritardi o altri malfunzionamenti che abbiano compromesso la validità del dato stesso. I meccanismi di controllo implementati dal *black channel* hanno lo scopo di consentire il rilevamento di anomalie durante la trasmissione, quali la ripetizione o la perdita di un messaggio valido (*Uninted repetition*, *Loss*), l'inserimento di un falso messaggio nella rete (*Insertion*), la ricezione di messaggi in sequenza errata o corrotti (*Incorrect sequence*, *Corruption*), ritardi nella ricezione dei messaggi (*Unacceptable delay*), la ricezione di un messaggio destinato ad un altro nodo della rete (*Addressing*) e il riconoscimento di un messaggio *standard* che presenti la struttura di un messaggio *safe* (*Masquerade*). E' necessario attuare almeno una misura di difesa contro ogni possibile errore, come si può notare in Tabella 2.4. Queste contromisure difensive includono numeri di sequenza, conferma della ricezione, *timout* entro cui considerare attendibile il messaggio, metodologie per garantire l'integrità dei dati di sicurezza, come l'utilizzo di *checksum CRC* (*Cyclic Redundancy Check*), e la ridondanza, che consiste nella duplicazione dei dati trasmessi al fine di aumentarne l'affidabilità.

A seconda della politica di sicurezza attuata, con approccio *failsafe* o *fault-tollerance*, il sistema si comporta in maniera differente in caso di rilevamento di errori.

Un sistema *failsafe*, in caso di fallimento, si comporta in modo tale da non procurare danni ad altri dispositivi o situazioni di pericolo per il personale e l'ambiente. Per questo motivo, i dispositivi si portano in uno stato di sicurezza in cui

tutte le uscite di controllo adibite alla sicurezza vengono disabilitate. I segnali di uscita vengono annullati perchè nell'ambito dell'automazione industriale i segnali di allarme sono normalmente chiusi, quindi attivi a livello logico basso, mentre gli azionamenti sono normalmente aperti, quindi attivi a livello logico alto. Così, con una sola operazione si disabilitano tutti gli azionamenti e si abilitano gli eventuali allarmi per segnalare la situazione di pericolo. Se, ad esempio, un operatore varca una barriera fotoelettrica che immette in una cella di produzione in cui è in funzione un robot, il sistema di controllo riconosce una situazione di possibile pericolo e mette in sicurezza l'area, disabilitando il robot, le cui variabili di comando sono state definite *safe*. Acora più importante è il caso in cui, la barriera fotoelettrica si guasti e non riesca a rilevare la presenza dell'operatore nella cella di lavoro. Definendo le variabili di controllo come *safe*, a differenza di quelle *standard*, il sistema viene messo in sicurezza anche in caso di guasto delle apparecchiature. Nel caso specifico, il sistema di controllo non ha più informazioni da parte della barriera fotoelettrica e non sapendo il motivo della mancata comunicazione, che può essere dovuta ad un guasto o alla perdita di connessione fra i dispositivi, mette in sicurezza l'intera cella lavorativa. Questo comportamento di sicurezza è di primaria importanza anche se potrebbe compromette l'*avaiability* del sistema.

Diversamente, un sistema *fault-tollerant* (*tollerante agli errori*), in caso di guasti hardware o errori software, continua a produrre, anche se a capacità limitata, perchè consente ad alcuni dei suoi componenti di operare correttamente. La qualità del funzionamento diminuisce in proporzione alla gravità del fallimento, rispetto ad un sistema *failsafe* in cui anche un piccolo errore può causare il collasso totale del sistema. Questi sistemi sono più complessi nella realizzazione e permettono, in caso di fallimento, di stabilire delle precedenze operative per garantire la messa in sicurezza del processo in atto e di procedere, eventualmente, con il processo produttivo a prestazioni ridotte. Si immagini un'industria di panificazione, controllata da un sistema *fault-tollerant*, in cui uno dei forni della catena di automazione viene danneggiato. In tale condizione, è possibile continuare a produrre, anche se con una frequenza operativa inferiore, utilizzando quei circuiti di produzione che non coinvolgono la parte guasta. Oppure, si pensi ad un'industria petrolchimica in cui si realizza un guasto, ad alto potenziale di rischio, al macchinario di raffinazione del petrolio. Il sistema di controllo non deve chiudere improvvisamente i condotti che portano il greggio a tale macchinario; l'arresto del flusso deve essere graduale per evitare il danneggiamento delle tubature. Quindi, il sistema di controllo deve permettere al processo produttivo di passare per stati dinamici intermedi prima di arrivare a quello statico di sicurezza, in cui tutte le attività potenzialmente pericolose sono sospese.

Questi due differenti approcci, *failsafe* e *fault-tollerance*, stabiliscono delle diverse relazioni tra il concetto di *avaiability* e il verificarsi di errori.

Nel primo caso, la presenza di qualsiasi errore compromette l'*avaiability* di tutto il sistema, dato che questo viene completamente messo in uno stato di sicurezza, mentre nel secondo, la maggior parte degli errori che si possono verificare non compromettono l'*avaiability* di tutto il sistema ma solo di alcune sue sottoparti. L'importanza dell'*avaiability* di un impianto per una azienda dipende dal valore intrinseco e dalla frequenza di produzione dei pezzi.

Se si considera l'ottimizzazione del processo, l'arresto di sicurezza viene visto

come un impedimento alla produzione; la sicurezza deve essere garantita ma ciò non deve influenzare in maniera preponderante l'*avaiability* complessiva del sistema. Si tratta, quindi, di bilanciare le caratteristiche di *safety* e di *avaiability*, così da evitare, il più possibile, che il sistema venga posto in sicurezza davanti a falsi allarmi. Nei casi in cui l'errore effettivamente si verifichi, è necessario trovare la causa del fermo dell'impianto nel più breve tempo possibile, eliminarla e riattivare il processo produttivo.

CAPITOLO

3

Il protocollo SafetyNET p®

In questo capitolo sono presentate le caratteristiche generali del protocollo di comunicazione SafetyNET p, che forniscono tutti gli elementi basilari di SafetyNET p, in maniera da poter affrontare i successivi capitoli di analisi del protocollo, entrando più nello specifico.

Figura 3.1: Logo dello protocollo SafetyNET p.

3.1 Panoramica generale

Le attuali esigenze di una rete di comunicazione industriale sono quelli di integrare le funzionalità relative alla *safety* e di ridurre i tempi di ciclo dei dispositivi. Per questo, i diffusi FieldBus di comunicazione sono ormai diventati un fattore limitante e, dunque, per questo sono richiesti nuovi sistemi di comunicazione basati su Ethernet, che offrono maggiori prestazioni.

Sulla base di questi requisiti è stato specificato e sviluppato da Pilz GmbH & Co. KG un protocollo di comunicazione industriale, basato su Ethernet, chiamato SafetyNET p. SafetyNET p possiede elevate capacità *real-time*, con tempi ciclo che raggiungono il livello di μs, e integra tutte le funzioni necessarie per rendere *safe* una comunicazione. E' in grado di combinare le funzioni relative alla *safety*

con quelle di controllo della parte di automazione *standard*, senza che possano interagire pericolosamente tra loro ma coesistendo in maniera "intelligente".

Per quanto riguarda la trasmissione di dati e di informazioni legati alla sicurezza di uomini e macchine (*safety*), è stata utilizzata tutta l'esperienza e la tecnologia legata alla sicurezza proveniente dal protocollo, basato su FieldBus, SafetyBus p.

Dopo la sua introduzione nel 1999 il protocollo di sicurezza SafetyBUS p si è affermato nei settori più svariati (industria automobilistica, impianti di risalita e aeroporti) e in centinaia di applicazioni, raggiungendo oltre 650 mila nodi installati in tutto il mondo. A distanza di circa sette anni, grazie al salto tecnologico compiuto nell'ambito dello scambio dati con l'introduzione di Ethernet, è stato creato un nuovo sistema di comunicazione industriale: SafetyNET p.

SafetyNET p è il logico sviluppo di SafetyBUS p, con la velocità e la funzionalità di Ethernet. Mentre la larghezza di banda era molto bassa nei tradizionali sistemi FieldBus, rendendo questa soluzione meno attraente, oggi la situazione con Ethernet è fondamentalmente diversa. Grazie alla maggiore larghezza di banda, le operazioni di diagnostica e di assistenza remota possono essere realizzate senza danneggiare il tempo di ciclo del processo.In ogni caso, SafetyNET p è assolutamente compatibile con le applicazioni realizzate con SafetyBus p. In termini pratici, SafetyBUS p è collegato mediante un *gateway* o attraverso un PLC di sicurezza che possiede entrambe le interfacce di rete. Questa caratteristica permetterà, in futuro, che tutti i dispositivi SafetyBUS p possano essere utilizzati all'interno delle reti SafetyNET pper poter proteggere gli investimenti dei sistemi esistenti.

SafetyNET p possiede una larghezza di banda di 100 Mbit/s e, come supporto fisico, possono essere utilizzati tutti i comuni tipi di connessione, come quelle elettriche (*100BASE-TX*) e in fibra ottica (*100BASE-FX*). Inoltre, l'organismo *IANA*[1] (Internet Assigned Numbers Authority) ha assegnato a SafetyNET p l'*EtherType*[2] 0x9C40; ciò ne assicura, a livello mondiale, l'univoca identificabilità all'interno di reti Ethernet.

A differenza di tutti gli altri approcci esistenti in commercio, SafetyNET p fino dalla fase di progettazione è stato sviluppato per la comunicazione di tutti i dati di processo: sia quelli relativi alla *safety*, *dati safe*, che quelli classici di rete di automazione, non relativi alla sicurezza, detti *dati standard*. Il principio di comunicazione è certificato SIL3 secondo la norma IEC 61508, in modo che possa essere utilizzato in applicazioni *safe* per garantire la salvaguardia del personale operativo, delle macchine e dell'ambiente.

SafetyNET p opera in conformità con il principio di *black channel*, che permette di convogliare sullo stesso canale trasmissivo sia dati *safe* che *standard*. Questo significa che, ad eccezione dei dispositivi di sicurezza, tutti gli altri componenti della rete sono considerati come *non safe*.

Tutti i documenti necessari per l'attuazione di SafetyNET p, come le caratteristiche tecniche e l'implementazione hardware e software del protocollo di comunicazione, possono essere reperiti tramite l'organizzazione che sostiene i produttori durante il processo di realizzazione e i test di conformità dei prodotti SafetyNET p,

[1] Lo *IANA* (*Internet Assigned Numbers Authority*) è un organismo che ha responsabilità nell'assegnazione degli indirizzi IP.

[2] L'*EtherType* è un campo, composto da 2 Byte, presente in un frame Ethernet. Esso è usato per indicare che protocollo è incapsulato nel PayLoad di un frame Ethernet.

ovvero Safety Network International e.V..

La tecnologia dell'automazione si sta allontanando dai sistemi di controllo centralizzato; la tendenza è di spostarsi verso sistemi distribuiti con dispositivi più sofisticati che incorporano una quantità sempre più crescente di pre-elaborazione a bordo. Poichè ogni dispositivo è dotato di più logica a bordo, nei sistemi di controllo decentralizzato la funzione di controllo di processo viene distribuita su un elevato numero di CPU. Un vantaggio di una struttura decentralizzata può essere visto nella reazione a eventi locali, che nella maggior parte dei casi è notevolmente più veloce. I requisiti dei sistemi di comunicazione si sono, di conseguenza, allontanati dal trasferimento dati in configurazione *master/slave*, passando a reti in grado di trasferire i comandi e le richieste tra i dispositivi "intelligenti".

In SafetyNET p non vi è un controllo centralizzato, in quanto non vi è una comunicazione *master/slave*. Questo è il motivo per cui SafetyNET p è un sistema di comunicazione ottimale per sistemi di controllo distribuiti ed è l'ideale per la realizzazione di impianti modulari. E' possibile realizzare sistemi di automazione con, teoricamente, più di 65 000 dispositivi per segmento, con una estensione possibile di svariati chilometri.

SafetyNET p è stato progettato esclusivamente come rete di comunicazione basata sul concetto *publish/subscribe* e funziona senza necessità di un sistema di controllo centrale. Il modello di messaggistica *publish/subscribe* è utilizzato per il trasferimento ciclico sia dei dati *standard* che *safe*. Il mittente di un messaggio, detto *publisher*, non programma i messaggi da inviare direttamente ai ricevitori, detti *subscriber*. I messaggi pubblicati sono caratterizzati in classi e i *publisher* non sono a conoscenza di quanti e quali *subscriber* possano essere interessati ai loro dati. I *subscriber* possono esprimere interesse per una o più classi, ricevendo, tra tutti i messaggi che circolano nella rete, solamente quelli richiesti. Il meccanismo di sottoscrizione consente, dunque, ai *subscriber* di precisare, nel modo più specifico possibile, a quali messaggi sono interessati. Per esempio, un *subscriber* potrebbe "abbonarsi" solo alla ricezione di messaggi pubblicati da determinati *publisher*, oppure aventi certe caratteristiche. Questo modello di messaggistica è la soluzione ideale per distribuire la tecnologia di controllo, dato che i dispositivo sono in grado di comunicare direttamente tra loro senza dover passare la comunicazione attraverso una nodo centrale, detto *master*. Il disaccoppiamento dei *publisher* e dei *subscriber* consente una maggiore scalabilità e una più dinamicità della topologia di rete.

Il protocollo di comunicazione industriale SafetyNET p è stato inserito all'interno delle norme IEC 61158 e IEC 61784-2 come standard internazionale.

Nello specifico, la norma IEC 61784-2, che valuta la comunicazione in tempo reale di un sistema di comunicazione basato su Ethernet, è stata aggiornata nella versione IEC 61784-2ed2.0 per poter includere il *CPF* (*Communication Profile Family*) 18 che descrive il protocollo SafetyNET p.

Al fine di soddisfare le diverse esigenze industriali, all'interno del protocollo di comunicazione SafetyNET p vengono distinte due tipologie di comunicazione, compatibili tra loro, che possono essere utilizzate separatamente o in combinazione:

1. **Comunicazione RTFL (Real-Time Frame Line)**:

   Il formato RTFL è ottimizzato per comunicazioni estremamente veloci all'interno delle celle di produzione, che per lo più sono realizzate in una struttura lineare, in cui le applicazioni sono altamente dinamiche. Ad esempio, compiti di controllo del movimento e circuiti di regolazione.

   Per garantire queste elevate prestazioni, RTFL si è dovuto svincolare dallo *standard Ethernet*: l'*application layer* si appoggia direttamente sui primi 2 livelli, *physical* e *data link layer*, del modello ISO/OSI (Figura 3.2).

   In questa maniera, permette prestazioni *real-time* con tempi ciclo fino a $62.5\mu s$, che sono circa 20 volte più veloci rispetto alla maggior parte dei protocolli di comunicazione industriali, basati su Ethernet, presenti sul mercato.

2. **Comunicazione RTFN (Real-Time Frame Network)**:

   L'interconnessione delle singole celle di produzione e la comunicazione della rete di fabbrica con quella di amministrazione aziendale, sono assolutamente essenziali per una gestione globale del sistema produttivo.

   In SafetyNET p tutte queste funzionalità vengono permesse e realizzate tramite il formato di comunicazione RTFN.

   Questa versione *real-time* si basa sullo *standard Ethernet* ed offre, quindi, la possibilità di comunicare le informazioni attraverso qualsiasi rete Ethernet-commerciale e con i protocolli di comunicazione industriali basati anch'essi sullo *standard Ethernet*. Questo permette l'utilizzo, senza restrizioni, di tutti i componenti di rete Ethernet commerciale, sia passivi, come cavi e connettori, che attivi, come switch e router.

   A causa di possibili collegamenti con questi dispositivi di rete attivi, può essere raggiunto un tempo di ciclo del sistema fino a $1ms$.

   Per questo motivo, RTFN è usato per la comunicazione in cui si ha la necessità di trasmettere grandi quantità di dati, con ridotte esigenze in tempo reale.

   Inoltre, nella comunicazione RTFN, per permettere l'integrazione con RTFL, è possibile anche non utilizzare il protocollo TCP/IP o UDP/IP e appoggiare, pertanto, direttamente sui livelli inferiori del modello ISO/OSI (Figura 3.2).

All'interno della norma IEC 61784-2 ed2.0 sono state elencate entrambe queste due tipologie di comunicazioni: il *CP* (*Communication Profile*) 18/1 si riferisce a RTFL e il CP 18/2 a RTFN.

La comunicazione orientata alla sicurezza è ovviamente supportato da entrambe le varianti RTFL e RTFN di SafetyNET p. Il livello di applicazione delle tipologie di comunicazione RTFL e RTFN è identico ed è implementato come software sul microprocessore dei dispositivi. Nei dispositivi, per una corretta elaborazione del protocollo *safety*, è spesso richiesta una architettura ridondante ed è per questo che, in quelli che implementano SafetyNET p, sono presenti due microprocessori distiniti. Le due tipologie di comunicazione, come rappresentato in Figura 3.2, si differenziano solo per il livello di trasporto. RTFL richiede un supporto hardware

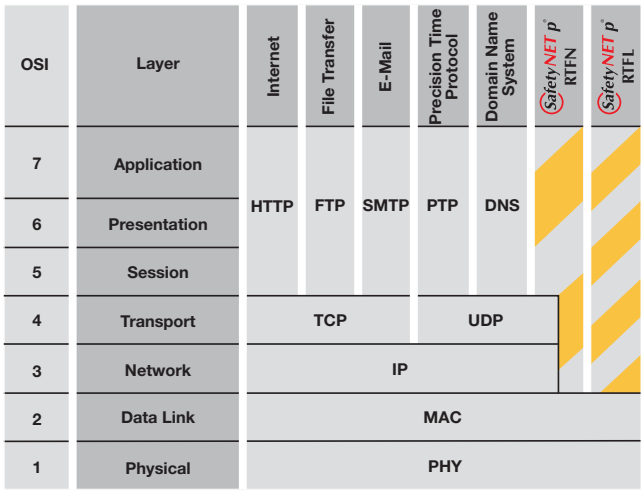


Figura 3.2: Tabella del modello OSI/ISO per la comunicazione RTFL e RTFN

aggiuntivo, un chip FPGA, mentre RTFN non richiede supporti hardware speciali e può essere implementato sotto forma di driver su una qualsiasi scheda Ethernet commerciale. Dato la differenza tra le due tipologie di rete intercorre a livello di trasporto, è necessario dunque un *gateway* per permette la comunicazione tra RTFL e RTFN.

Quindi, RTFL e RTFN non sono soltanto differenti per le prestazioni di tempo ciclo raggiungibili ma anche per l'implementazione di alcuni livelli del modello ISO/OSI. Tuttavia, le strutture dei dati dell'*application layer* sono le stesse e dunque, dal punto di vista dell'utente, entrambi i metodi sono equivalenti.

La sincronizzazione esatta dei dispositivi di rete è particolarmente importante, soprattutto se il processo di produzione richiede azioni con un alto livello di simultaneità.

I sistemi di comunicazione basati su FieldBus utilizzano semplici telegrammi di dati per la sincronizzazione tra tutti i dispositivi della rete. Questo metodo, però, non riesce a considerare il tempo di trasmissione attraverso la rete, che può dipendere dai dispositivi presenti nella rete, né le potenziali collisioni o i tempi di attesa per accedere al mezzo di trasporto stesso. Come risultato, è impossibile una sincronizzazione precisa tra i dispositivi del bus.

Le applicazioni con cicli di processo rapido, in particolare le applicazioni *real-time* distribuite su più dispositivi, richiedono, però, meccanismi più precisi. Questi requisiti più elevati possono essere realizzati utilizzando la sincronizzazione degli eventi a livello di rete. Per questo, è necessario introdurre meccanismi che, nell'ambito della rete, attivino lo svolgimento delle relative azioni al momento rilevante. Per stabilire correttamente questi istanti, ogni dispositivo della rete ha bisogno di un *clock* preciso che sia sincronizzato con gli altri. La precisione della sincronizzazione degli orologi (*clock synchronization*) determina la precisione del sistema complessivo. Questa precisione può essere misurata mediante la differenza tra i *clock* distribuiti, indicata con il termine *jitter*[3]. In pratica, possedere un basso

[3]Con il termine *jitter* si indica la variazione di una o più caratteristiche di un segnale come,

jitter significa processi di produzione più precisi e veloci.

In SafetyNET p sono utilizzati due meccanismi per la sincronizzazione del *clock*: il *Precision Time Protocol* (*PTP*) e il *Precise Clock Synchronisation* (*PCS*).

Il *Precision Time Protocol* (*PTP*), standardizzato dalla norma IEEE 1588, viene utilizzato per sincronizzare i dispositivi SafetyNET p del formato RTFN perchè ha il vantaggio che è supportato da molti componenti *standard Ethernet*, come gli switch e i router.

Il *Precise Clock Synchronisation* (*PCS*), che è ottimizzato per l'utilizzo in RTFL, possiede un *master clock* che informa tutti gli altri *clock* della rete RTFL, detti *clock slave*, del suo valore ad intervalli temporali determinati. Questi regolano il loro valore se incongruente con quello del *master clock*. A sua volta, il *master clock* può essere sincronizzato con l'ora universale degli orologi atomici.

Con la tipologia RTFL del protocollo di comunicazione SafetyNET p, sono possibili *jitter* inferiori ai $100ns$.

Nel caso in cui in una rete di comunicazione industriale SafetyNET p siano utilizzate entrambe le tipologie RTFL e RTFN, è compito del *master clock* RTFL sincronizzare i *clock* tra le sottoreti.

3.1.1 Comunicazione RTFL

Nella tipologia di comunicazione *RTFL* (*Real-Time Frame Line*) i dati vengono trasmessi ciclicamente da un dispositivo all'altro. È irrilevante se i dispositivi sono collegati realmente in una topologia lineare o se sono posizionati in vari rami della rete; l'importante è che ogni dispositivo abbia un proprio predecessore e successore all'interno della rete. Nella descrizione presentata qui di seguito, tale sequenza è chiamata "*linea logica*".

I dispositivi connessi tramite questo collegamento lineare sono distinti in *Root Device* (*RD*) e *Ordinary Device* (*OD*). Una rete RTFL deve contenere un solo RD. Questo dispositivo organizza e controlla il traffico dei dati, creando il *frame* Ethernet che viene trasmesso nella rete. L'RD legge la topologia di rete e configura la linea logica: crea un elenco che associa gli identificativi di tutti i dispositivi OD nella rete con i loro MAC e altre informazioni che sono necessarie per la gestione della rete. Ai dispositivi RD, inoltre, è possibile assegnare manualmente o automaticamente, tramite un server DHCP[4], un indirizzo IP.

Tutti i dispositivi OD della *linea logica* hanno uguali diritti. La loro identificazione all'interno di una rete può essere effettuata in base alla posizione fisica oppure tramite l'associazione di un nome, che può essere immagazzinato in maniera non volatile su una SD card. Quest'ultima opzione è sempre necessaria quando la posizione fisica non può essere individuata chiaramente, come in una struttura ad albero.

---

ad esempio, l'ampiezza, la frequenza, la fase. Le cause che portano alla comparsa di *jitter* devono essere tenute in forte considerazione nella progettazione dei sistemi e componenti elettronici in cui l'integrità dei segnali è un vincolo stringente.

[4]In telecomunicazioni il *Dynamic Host Configuration Protocol* (*DHCP*) (protocollo di configurazione IP dinamica) è un protocollo di rete di livello applicativo che permette ai dispositivi di una rete locale di ricevere dinamicamente ad ogni richiesta di accesso la configurazione IP necessaria per poter operare sulla rete.

I dati tra queste due classi di dispositivi vengono scambiati in conformità con il principio *publisher/subscriber* visto pocanzi. All'interno della rete, il trasferimento dei dati viene iniziato dal RD ad ogni ciclo. Questo dispositivo crea un *frame* Ethernet RTFL e lo invia al suo successore, considerato il primo OD della linea logica. A sua volta, il primo OD scrive i dati che vuole pubblicare nel *frame* Ethernet e lo invia al seguente OD, suo successore. Questo inoltro del *frame* viene effettuato per tutti gli OD della catena ed è possibile perchè i dispositivi all'interno della *linea logica* sono indirizzati tramite il loro *MAC address* ed ognuno di essi conosce quelli relativi sia al suo successore che al suo predecessore. Quando il *frame* Ethernet RTFL ha raggiunto l'OD che vi ha aggiunto i dati che vuole pubblicare, esso prende il messaggio completamente riempito dai vari dispositivi e lo invia in direzione inversa lungo la stessa *linea logica*. Tutti gli altri OD possono prelevare dal *frame* i dati che sono per loro rilevanti.

In questo modo, per ogni abbonato, è possibile scambiare dati con ogni altro dispositivo, dato che, in ogni ciclo, il *frame* viene trasmesso due volte ad ogni dispositivo. Nel primo passaggio, ogni dispositivo può pubblicare (*publish*) i dati e, durante il secondo passaggio, può ricevere quelli che ha sottoscritto (*subscribe*).

In una rete con una struttura lineare, il predecessore e successore di ogni dispositivo sono unici. Siccome ogni OD ha due interfacce Ethernet, la struttura lineare è facile da creare senza componenti di rete aggiuntivi. Per accelerare e rendere deterministico il processo di propagazione del *frame* Ethernet tra i dispositivi è stato utilizzato il meccanismo detto *Cut-Through*. Questo tipo di meccanismo richiede un supporto hardware aggiuntivo: un *FPGA* (*Field Programmable Gate Array*). Il *frame* ricevuto da un dispositivo a una delle sue porte Ethernet può essere letto o scritto mentre viene trasmesso all'altra porta prima di aver ricevuto completamente il messaggio a lui destinato. Questo riduce il tempo di ritardo, introdotti dal passaggio del *frame* da un dispositivi all'altro, che rappresenta uno degli svantaggi delle strutture lineari.

In questa maniera, con la tipologia di comunicazione RTFL possono essere realizzati tempi di ciclo fino a 62,5μs con un jitter inferiore a 100ns. E', inoltre, possibile realizzare topologie fisiche diverse da una *linea logica* ma i periodi di latenza degli *switch* ne aumenta il tempo di ciclo e il jitter.

Per questo, la comunicazione RTFL è utilizzata per applicazioni di unità che sono controllate e sincronizzate tramite la rete, in cui una comunicazione più veloce può accelerare il processo di produzione e portare ad un prodotto di qualità superiore. Ad esempio, le macchine da stampa in cui gli assi possono essere sincronizzati, tramite la rete, in cicli di processo più brevi permettono l'aumento della velocità di produzione e, di conseguenza, l'aumento del volume prodotto, senza ridurre la qualità di stampa.

In una comunicazione RTFL vengono utilizzati i seguenti canali di comunicazione:

- *Cyclic Data Channel* (*CDC*): per i dati di processo ciclico;
- *Message Channel* (*MSC*): per i dati di processo non ciclici;
- *Standard Ethernet Frame* (*SEF*): per la comunicazione di dati TCP/IP.

Da questo elenco si nota come anche i dati TCP/IP possono essere trasmessi in una linea RTFL, senza influenzare le caratteristiche di *real-time* del sistema. I normali dispositivi basati su TCP/IP vengono collegati al primo o all'ultimo *subscriber* OD della *linea logica*. Se il tempo di ciclo dei dati di processo è superiore a $500ms$, i pacchetti TCP/IP vengono trasmessi in un unico frame durante i vuoti fra i messaggi RTFL; se invece è inferiore a $500ms$, i dati TCP/IP vengono segmentati e inviati accodandoli in ogni frame RTFL.

3.1.2 Comunicazione RTFN

RTFN è utilizzato dove i requisiti di *real-time* non sono elevati e nelle reti in cui sono utilizzati l'Ethernet commerciale e i protocolli industriali basati sullo *standard Ethernet*. RTFN utilizza il protocollo UDP/IP e, per questo, i dati possono anche essere instradati oltre i limiti della rete. Questo permette alla rete di gestione aziendale e a quella di controllo del processo di essere collegate, senza la presenza di un *gateway*. E' il caso in cui si voglia connettere il PC dell'ufficio per accedere alla rete di automazione per operazioni di diagnostica o per monitorare il processo produttivo. L'implementazione dello *standard Ethernet*, tuttavia, riduce le caratteristiche *real-time*: i tempi di ciclo non sono inferiori a $1ms$.

In una comunicazione RTFN vengono utilizzati i seguenti canali di comunicazione:

- *Cyclic Data Channel* (*CDC*): per i dati di processo ciclico;
- *Message Channel* (*MSC*) per i dati di processo non ciclici

Possono essere stabiliti dei collegamenti punto-punto tra i dispositivi basati su *frame* MAC o *frame Ethernet* UDP/IP-TCP/IP, a seconda del canale di comunicazione.

I *frame* MAC vengono utilizzati nel CDC per la trasmissione di dati con particolari requisiti *time-critical*. La comunicazione si basa solo sui primi due livelli del modello OSI/ISO e, per questo, richiede meno elaborazione. Questo è il mezzo più rapido di comunicazione in RTFN.

Invece, i *frame Ethernet* UDP/IP sono meno performanti e questo influisce sul comportamento *real-time*. Per questo motivo, UDP/IP viene utilizzato principalmente quando è richiesta la capacità di *routing* del *frame* UDP/IP. I *router* possono essere utilizzati nella versione RTFN senza alcuna restrizione. In questo modo, è possibile la segmentazione delle reti più complesse e SafetyNET p può essere collegata ad ogni dispositivo al di fuori dei confini della rete per, ad esempio, rendere possibile l'esecuzione di operazioni di manutenzione remota. Il protocollo di trasporto UDP ha la stessa funzione del TCP ma rispetto ad essa permette comunicazioni più veloci. Tuttavia, dato che è un protocollo senza connessione, non ha alcun mezzo per garantire che il pacchetto sia arrivato a destinazione. Questo deve essere eseguita dagli strati di livello superiore.

Infine, i *frame Ethernet* TCP/IP non permettono un comportamento *real-time* e sono utilizzati per tutti quei servizi che trattano dati che non sono *time-critical*. La comunicazione TCP/IP viene utilizzata non per scambiare dati di processo fra i dispositivi ma per permettere un accesso remoto per funzioni di assistenza, in cui

è molto utile l'affidabilità della trasmissione dei dati e l'automatica ripetizione dei telegrammi.

A differenza del formato RTFL, in RTFN non vengono distinti i dispositivi in RD e OD e non è richiesto nessun hardware particolare per implementarlo.

I dispositivi di una rete RTFN sono identificati per mezzo di un indirizzo IP. E' possibile utilizzare la versione IPv4, in cui l'indirizzo IP è lungo 4 byte, oppure la versione IPv6, che dispone di un indirizzo di 16 byte, esteso per ampliare lo spazio di indirizzamento.

In questa tesi verrà considerato solo il modello di comunicazione RTFN, in quanto le differenze tra RTFN e RTFL si sviluppano principalmente nel *transport layer*. Dal momento che si è interessati ai meccanismi di protezione attuati a livelli più elevati, tutte le considerazioni che verranno fatte per la tipologia di comunicazione RTFN potranno essere applicate anche al modello di comunicazione RTFL.

3.1.3 Application layer

Come si è visto in precedenza, entrambe le tipologie di comunicazione RTFL e RTFN differiscono soltanto a livello di trasporto, mentre il livello di applicazione resta il medesimo.

Tutte le funzioni di un dispositivo SafetyNET p sono rese disponibili attraverso il livello di applicazione, che rappresenta l'interfaccia tra l'utente e il sistema di comunicazione.

Il livello di applicazione di sicurezza permette di trasmettere i dati *safe* sullo stesso sistema trasmissivo utilizzato per le informazioni *standard*. Tutti i meccanismi di sicurezza che consentono la comunicazione *safe* sono incapsulati all'interno del livello applicativo, quindi con l'eccezione dei dispositivi adibiti alla *safety*, tutti gli altri dispositivi e i componenti della rete, come switch, router e cavi, sono considerati non legati alla *safety*, in conformità con il principio di *black-channel*.

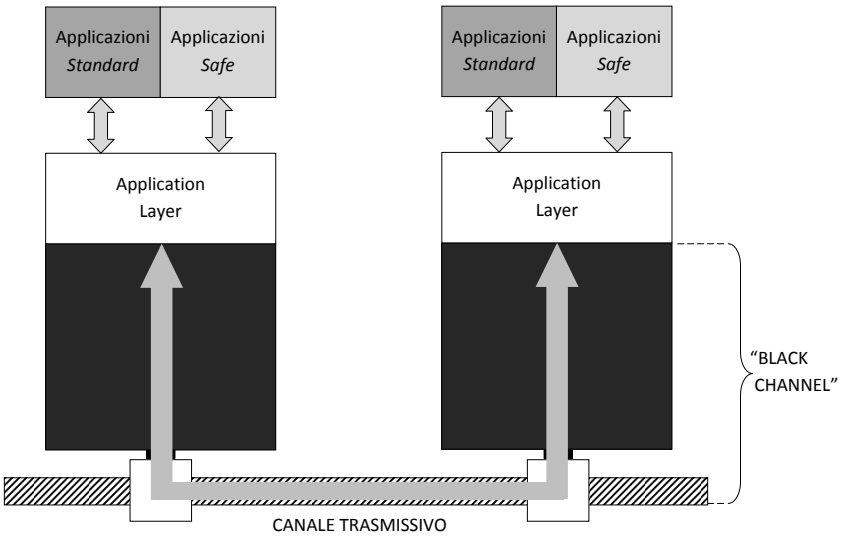


Figura 3.3: principio del "*black channel*"

Il principio di *black-channel* viene implementato, nel modello OSI/ISO, tra il *transport layer* e l'*application layer*. Questo concetto è la base per l'interoperabilità con protocolli di trasporto arbitrario. Per la funzionalità di sicurezza del protocollo SafetyNET p, non importa quale protocollo *transport layer* sia utilizzato per il trasporto dei *frame* di sicurezza perché tutti i meccanismi di sicurezza sono integrati esclusivamente sull'*application layer* e la sua funzionalità è indipendente dai livelli sottostanti. Questo significa che, dal punto di vista del sistema che viene utilizzato come canale di comunicazione, i dati *safe* sono contenuti in pacchetti dentro a dei classici *frame* Ethernet che si integrano senza problemi con il resto del traffico dati della rete.

SafetyNET p utilizza a livello di applicazione il popolare *CANopen*[5], lo standard per il *networking* di sistemi di controllo nel settore industriale, senza la limitazione degli 8 byte.

La decisione di utilizzare CANopen nel protocollo di comunicazione industriale SafetyNET p è stata presa a causa della sua diffusione e della vasta gamma di profili dei dispositivi disponibili. Il livello di applicazione contiene lo standard per la comunicazione e i dati relativi alle prestazioni tecniche e funzionali. Gli elementi essenziali sono i profili dei dispositivi, disponibili per tutti i comuni tipi di dispositivi che si trovano in automazione, che definiscono le funzioni, i parametri e gli oggetti delle varie applicazioni.

L'elemento centrale dell'*application layer* CANopen è l'*object directory*, che opera come collegamento tra l'applicazione e il sistema di comunicazione. In sostanza l'*object directory* specifica la comunicazione, i dati, i parametri e le funzioni dei dispositivi. I dati delle applicazioni e le funzioni applicative sono scritte in *application object*, conservate nella *object directory* e accessibili come elementi all'interno di una tabella.

Come *application object*, in SafetyNET p, si distingue per la comunicazione *standard* tra *PDO* (*Process Data Objects*) e *SDO* (*Service Data Objects*), invece per la comunicazione *safe* tra *SPDO* (*Safe Process Data Objects*) e *SSDO* (*Safe Service Data Objects*) .

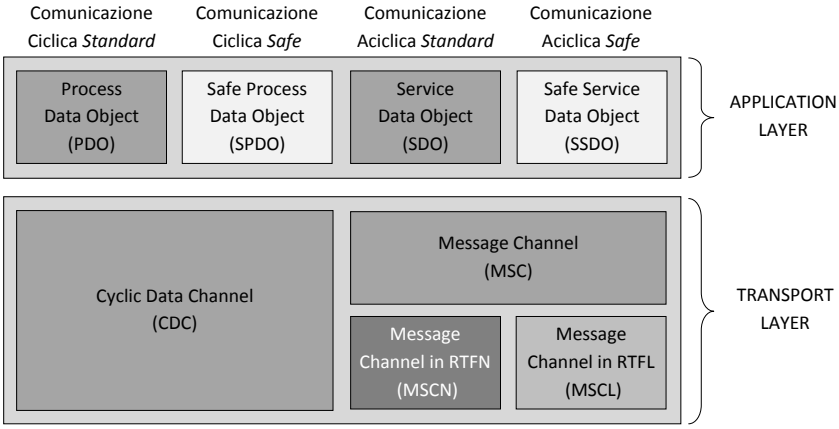


Figura 3.4: canale di comunicazione e *application object*

[5] *CANopen* è uno standard aperto che è stato gestito dall'organizzazione *CiA* (*CAN in Automation*) dal 1995 e è standardizzato come la norma europea EN 50.325-4.

I dati ciclici di processo, che devono essere aggiornati regolarmente in tempo reale, vengono trasmessi come PDO nel canale di comunicazione CDC. I dati aciclici, che invece possiedono requisiti temporali meno stringenti, come parametri di configurazione del dispositivo o dati diagnostici, vengono trasferiti sotto forma di SDO nel canale di comunicazione MSC.

Gli SPDO sono anch'essi memorizzati nella *object directory* ma con una garanzia aggiuntiva. L'accesso in scrittura agli oggetti legati alla sicurezza è consentito solo tramite SSDO. L'accesso in sola lettura ai dati relativi alla sicurezza, invece, è consentito sia tramite SDO che SSDO.

3.1.4 Comunicazione *safe*

Il concetto di *safety*, inteso come la sicurezza di persone, macchine e dell'ambiente, è diventato un fattore molto importante per la corretta progettazione di macchine e impianti nelle realizzazioni industriali più moderne. In un processo industriale ci si concentra sui rischi che sono connessi con l'uso della tecnologia, la gestione degli impianti e dei materiali pericolosi. Per veicolare dati riguardanti la sicurezza funzionale di un macchinario è necessario utilizzare un profilo *safety* per garantire la conformità alle norme vigenti in materia.

SafetyNET p nasce appositamente come protocollo Ethernet industriale real-time orientato alla *safety*, capace di coniugare elevate prestazioni e i requisiti necessari per la sicurezza funzionale, che generalmente impattano sulle prestazioni dei sistemi dal punto di vista sia hardware che software. La caratteristica fondamentale che contraddistingue la soluzione è il fatto di aver pensato, progettato e integrato la sicurezza fin dalle fasi iniziali di sviluppo di SafetyNET p.

In SafetyNET p, la comunicazione *safe* è stata sviluppata in conformità alle norme internazionali pertinenti. Infatti, è conforme al livello di sicurezza SIL3 previsto dalla norma EN IEC 61508. Inoltre, istituti di controllo indipendenti come *TÜV*[6] *BG-PRÜEFZERT* hanno confermato la *safety* e l'idoneità di SafetyNET p per applicazioni di sicurezza a *Cat.4* di *EN 954-1* e al *Performance Level "E"* della norma *EN ISO 13849*.

Il livello di applicazione di sicurezza di SafetyNET p consente il trasferimento di dati relativi alla sicurezza, detti *dati safe*, tramite lo stesso cavo di rete che viene utilizzato per comunicare i dati non relativi alla sicurezza, detti *dati standard*. La combinazione di comunicazione *safe* e *standard* nello stesso supporto fisico ha un certo numero di vantaggi, in cui il più importante è la capacità di semplificare e ridurre i costi di cablaggio. Per contro, questo meccanismo di convogliare le due tipologie di dati sul medesimo cavo introduce delle problematiche, dal momento che i requisiti delle comunicazioni *safe* e *standard* sono chiaramente differenti. Infatti, nel primo caso, l'obiettivo principale è quello di portare il sistema in uno *stato di sicurezza* (detto *safe-state*), entro un massimo tempo prestabilito e, in

[6]Il *TÜV* (*Technischer Überwachungs-Verein*, in italiano *Associazione di Controllo Tecnico*) è un gruppo di società di certificazione tedesche in ambito di sistemi di gestione della sicurezza alimentare e ambientale e per la qualità del sistema di gestione aziendale. Il gruppo *TÜV SÜD*, fondato nel 1866 in Baviera, la cui base principale è rimasta a Monaco di Baviera, è cresciuto progressivamente a livello internazionale al punto di essere attualmente presente in 60 Paesi (con oltre 600 uffici in Europa, America, Asia e 14500 dipendenti).

teoria, con assoluta certezza non appena viene rilevato un guasto in una parte di apparecchiatura. Nell'altro caso, invece, l'obiettivo è il miglioramento dei tipici indici prestazionali, come i tempi di ciclo, la velocità di trasmissione e l'*avaiability*.

In SafetyNET p si sfrutta il principio del *black channel*, che è un tecnica che consente la comunicazione sicura su canali di trasmissione convenzionali, intrinsecamente non sicuri. In pratica, si aggiunge un livello di sicurezza sopra al livello *transport layer* del modello di comunicazione ISO/OSI e al di sotto dell'*application layer*, che attua la trasmissione dei dati *safe* e controlla anche l'integrità del canale di comunicazione, in modo da assicurare che i requisiti di sicurezza siano soddisfatti. L'utilizzo del principio del *black channel* ha due principali vantaggi: in primo luogo, possono essere adottate le convenzionali ed economiche tecnologie di trasmissione utilizzate nelle reti industriali e, dall'altro, possono essere collegati alla stessa rete sia dispositivi *safe*, per un massimo di 512 in un segmento, che *standard*. Inoltre, l'utilizzo di un'unica rete mette a disposizione dei dispositivi *safe* anche un numero di funzioni non relative alla sicurezza, tipiche dei dispositivi *standard*. Per esempio, consentendo funzioni di diagnostica remota sofisticate permette di migliorare l'*avaiability* del sistema e, indirettamente, di aumentare l'efficacia globale della funzioni connesse alla sicurezza.

Nonostante in tipiche condizioni di funzionamento il *Bit Error Rate*[7] su reti cablate possa essere molto basso, gli errori di trasmissione non possono essere completamente eliminati nel mondo reale. Errori sui dati ricevuti possono dipendere da cause temporanee o permanenti. Per esempio, gli errori transitori potrebbero essere causati da interferenze elettromagnetiche, mentre quelli permanenti potrebbero dipendere da eventi di guasto fisico come un cavo rotto.

Ai fini della sicurezza delle macchine, l'elaborazione e la comunicazione di dati di processo relativi alla sicurezza, *safe*, richiedono l'attuazione di misure speciali rispetto alla gestione di quelli *standard*. Tali meccanismi di sicurezza sono attuati all'interno del protocollo SafetyNET p, al fine di superare i più svariati e diffusi errori di comunicazione.

L'approccio del *black channel* delega all'*application layer* la gestione di una serie di meccanismi che possono essere utilizzati per rilevare diversi tipi di errori di comunicazione. Nel caso specifico del protocollo SafetyNET p, sono considerati e gestiti i seguenti errori sui messaggi: corruzione, ripetizione non intenzionale, errata sequenza di consegna, perdita, ritardi eccessivi nella ricezione, inserimento, mascheramento degli errori (un messaggio *standard* ne simula uno *safe*), indirizzamento non valido ed errori di memoria negli switch.

Le contromisure di sicurezza per far fronte a tali errori, poste in atto a livello applicativo per le comunicazioni di dati *safe*, sono le seguenti:

- *Sequence number*: ad ogni singolo messaggio di una connessione viene assegnato un numero di sequenza progressivamente crescente per rilevare, ad esempio, le duplicazioni o le perdite;

[7] In un sistema di trasmissione digitale il *bit error rate* (*BER*), è il rapporto tra i bit non ricevuti correttamente e i bit trasmessi. Il BER è un parametro molto importante perché dà misura della qualità dell'intero sistema di comunicazione. Il BER evidenzia quanto della originaria trasmissione viene perso o giunge distorto all'apparecchio ricevente a causa, ad esempio, di disturbi e rumore nel canale di trasmissione, di problemi degli impianti o di malformazioni originarie del flusso dati.

- *Time-out*: per ogni messaggio è previsto un tempo massimo prestabilito entro cui deve essere ricevuto, in modo da affrontare i problemi di comunicazione;
- *Connection ID*: un codice identificatore (*IDentifier*) univoco è assegnato ai messaggi di ogni connessione, al fine di distinguere, in modo inequivocabile, i vari mittenti e pacchetti;
- *Data integrity assurance*: l'integrità dei dati è garantita contro la corruzione attraverso, ad esempio, la ridondanza e l'utilizzo di *checksum* CRC;
- *Differential data integrity assurance*: per il trattamento dell'indesiderata interferenza tra messaggi *safe* e *standard*.

Communication errors	**Safety measures**				
	Sequence number	**Time-out**	**Connection ID**	**Data integrity assurance**	**Differential data integrity assurance**
Corruption	–	–	–	×	–
Unintended repetition	×	–	–	–	–
Incorrect sequence	×	–	–	–	–
Loss	×	×	–	–	–
Unacceptable delay	–	×	–	–	–
Insertion	×	–	×	–	–
Masquerade	×	–	×	–	×
Addressing	×	–	×	–	–
Revolving memory failures within switches	×	–	×	×	×

Figura 3.5: misure di individuazione degli errori di comunicazione in SafetyNET p

La Figura 3.5 illustra quali tipi di errori sono rilevati dalle diverse misure di sicurezza sopra elencate. Chiaramente, queste contromisure sono piuttosto generiche nella loro natura e, in una soluzione reale, sono possibili diverse implementazioni per realizzarle. Nella Sezione 4, in cui verranno analizzate le strutture dei messaggi SafetyNET p più importanti, verrà descritta in dettaglio la modalità in cui queste vengono implementate all'interno della struttura di un messaggio *safe*.

3.1.5 Topologie di rete

Le esigenze particolari dell'industria di processo sono in parte legate alla struttura degli impianti e alla quantità dei segnali da scambiare. La progettazione degli impianti richiede sicuramente strutture e topologie flessibili, che possono essere ben rappresentate da architetture ad albero. I rami di queste strutture ad albero possono facilmente essere realizzati tramite switch commerciali.

SafetyNET p offre topologie e strutture flessibili e risulta, quindi, ideale per le tipologie più diverse di applicazione. Il supporto per una vasta gamma di topologie è un fattore significante per garantire che il cablaggio venga realizzato in maniera efficiente in base alla struttura fisica dell'impianto. I costi relativi ai lavori di cablaggio vengono così ridotti.

In SafetyNET p possono essere implementate tipologie di rete *lineare*, a *stella*, ad *albero* e ad *anello*, oltre a *connessioni dinamiche*. Inoltre, grazie a meccanismi automatici, in SafetyNET p, risulta possibile operare anche la scansione della topologia dell'intera architettura (*scan topology*). Questo semplifica di molto il riconoscimento della struttura di rete e fornisce agli utenti, oltre alla diagnostica e alla rilevazione dello stato dei dispositivi, informazioni ulteriori relative allo stato dei collegamenti di comunicazione.

Lineare

La topologia lineare viene realizzata collegando i dispositivi in linea, senza creare ramificazioni. RTFL è ottimizzata per essere utilizzata in una topologia lineare, creando una linea veloce con tempi di ciclo fino a 62,5 μs. Questa topologia può anche essere implementata nel formato RTFN ma le prestazioni saranno inferiori rispetto al formato di comunicazione precedente.

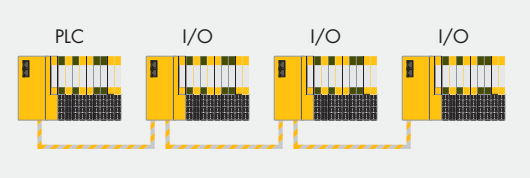


Figura 3.6: topologia Lineare

Stella

La topologia a stella si costiruisce quando più dispositivi sono collegati a un unico punto della rete. RTFN è la tipologia di comunicazione preferibile per topologie a stella. Tuttavia, configurazioni a stella possono essere realizzate anche per RTFL. In questo caso, è importante notare che il tempo di scansione e il jitter saranno aumentati a causa dell'introduzione di tempi di latenza addizionali, dovuti all'uso di uno switch.

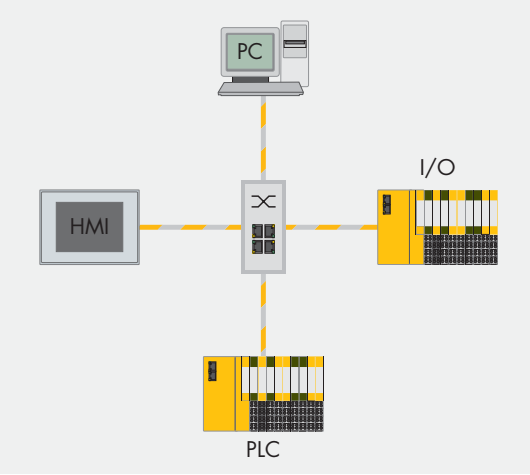


Figura 3.7: topologia a Stella

Albero

La topologia ad albero è la combinazione di topologie di rete a stella e lineari.

In SafetyNET p, questa topologia è utilizzata tipicamente per collegare, tramite RTFN, alcuni segmenti di rete RTFL.

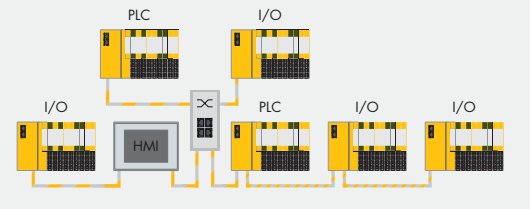


Figura 3.8: topologia ad Albero

Anello

La topologia ad anello consente di aumentare l'*avaiability* di una rete di comunicazione, aumentando la ridondanza dei collegamenti.

La maggiore *avaiability* della tipologia ad anello diventa chiara quando all'interno della rete, si verifica un guasto o si toglie un collegamento dell'anello: in questo caso, il cambiamento delle connessioni non altera il funzionamento della rete perchè tutti i dispositivi sono ancora accessibili.

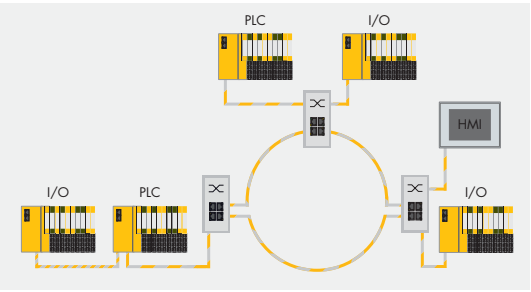


Figura 3.9: topologia ad Anello

Struttura dinamica

In una rete SafetyNET p è, inoltre, possibile aggiungere o rimuovere i dispositivi durante il funzionamento; ciò viene detto *collegamento a caldo* e risulta molto utile in caso di sostituzione dei dispositivi o dell'utilizzo di apparecchi mobili.

CAPITOLO

4

Struttura dei messaggi RTFN

In questo capitolo viene presentata la struttura dei messaggi più utilizzati della tipologia di comunicazione RTFN del protocollo SafetyNET p. Si mira a fornire un riferimento generico a cui si farà riferimento nei capitoli di analisi.

Come già accennato in precedenza, in questa tesi verrà considerato solo il modello di comunicazione RTFN del protocollo SafetyNET p, in quanto le differenze tra RTFN e RTFL si espletano nel *transport layer*. Dal momento che si è interessati ai meccanismi di protezione attuati a livelli più elevati, le considerazioni che verranno fatte per la tipologia di comunicazione RTFN potranno essere applicate anche al modello di comunicazione RTFL.

RTFN utilizza il modello di comunicazione *publisher-subscriber*. I dati pubblicati (*publish*) da un dispositivo sono distinti in classi, ciascuna delle quali può essere sottoscritta (*subscribe*), e quindi ricevuta, da qualsiasi nodo della rete. In questo modello di comunicazione sono utilizzati messaggi appartenenti a due differenti modelli di trasferimento dati: *confermato* e *non confermato*. I servizi *confermati* sono utilizzati dai *subscriber* per richiedere la sottoscrizione a certe tipologie di dati pubblicati dal *publisher*, la cui risposta, inviata dal *publisher* al *subscriber*, indica l'esito della sottoscrizione. I servizi *non confermati* sono utilizzati, invece, dal *publisher* per distribuire i propri dati ciclici ai *subscriber*.

Questi messaggi vengono trasmessi mediante le due tipologie di canale di comunicazione, disponibili per il formato RTFN, introdotti nella Sezione 3.1.2: *Ciclic Data Channel* (*CDC*) e *Message Channel* (*MSC*). Il CDC per la trasmissione di dati *ciclici*, e l'MSC per quelli *aciclici*. Per ciascuno di questi canali esistono specifici messaggi, appartenenti al modello *confermato* o *non confermato*, che sono stati codificati per assolvere a molte delle necessità di comunicazione e per la trasmissione dei dati di processo o di servizio.

Il CDC è utilizzato per effettuare comunicazioni cicliche *punto-punto* tra i dispositivi. La comunicazione ciclica storicamente è stata creata per rispondere alle esigenze industriali, in cui vengono ripetute periodicamente le stesse procedure in maniera continuativa. In questa trasmissione non vengono utilizzati *acknowledgement*, per non alterare il determinismo richiesto e per snellire la comunicazione al fine di preservare le caratteristiche real-time del sistema. Possono essere creati molteplici collegamenti di comunicazione unidirezionale fra i dispositivi, per ognuno dei quali può essere configurato un differente tempo ciclo che specifica il periodo con cui i messaggi CDC devono essere inviati dai dispositivi. In ciascun collegamento di comunicazione possono essere inviati dati di processo *standard* e *safe*, anche accodati all'interno dello stesso messaggio ciclico, che sono comunemente associati all'attuazione o alla lettura di segnali I/O. La struttura dei messaggi contenenti dati *safe* si differenzia da quella degli *standard* per la conformazione ridondante, in accordo con la norma di sicurezza SIL3, e per la presenza nella trasmissione di messaggi confermati aggiuntivi, necessari per monitorare il corretto svolgimento della comunicazione. Queste ulteriori procedure vengono applicate perchè, a differenza dei dati di processo *standard*, deve essere garantito il rilevamento di eventuali errori di comunicazione che porta il dispositivo in uno stato sicuro, anche detto *safe-state*.

L'MSC, invece, è utilizzato per le comunicazioni acicliche *punto-punto* tra dispositivi che non necessitano di comunicazioni real-time. Un protocollo specifico chiamato MSC-MTP (*MSC Message Transfer Protocol*) viene usato per trasmettere i dati in questo canale di comunicazione. In pratica, ciascun dispositivo scrive il proprio messaggio MSC-MTP che viene inserito, eventualmente accodato a quelli provenienti da altri, all'interno di un messaggio più grande chiamato *MSC frame*.

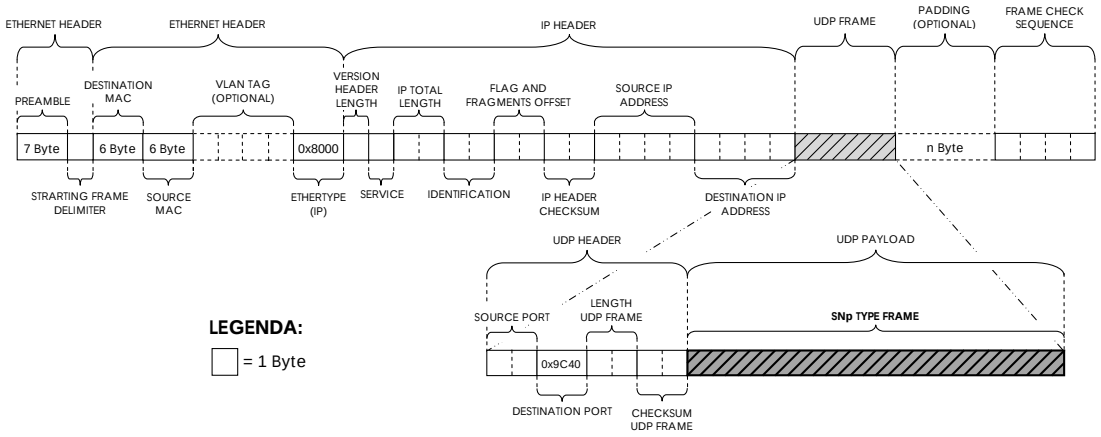


Figura 4.1: incapsulamento del frame SafetyNET p in un frame UDP Ethernet

La comunicazione RTFN può essere effettuata tramite UDP/IP o direttamente a livello MAC. Nell'ambito di questa tesi verrà utilizzato soltanto il protocollo UDP/IP per la trasmissione dei dati, perchè l'unico supportato dai dispositivi a disposizione per le analisi.

La struttura di un frame Ethernet UDP per la trasmissione di messaggi del protocollo SafetyNET p è rappresentato in Figura 4.1, dove si può notare che

all'interno dell'*UDP header* il valore della porta di destinazione assume il valore 0x9C40, che è univocamente assegnato a SafetyNET p dall'organismo IANA.

La voce *SNpTYPE frame*, che coincide con il *payload* di un frame UDP standard, è il campo in cui sono contenuti i byte che costituiscono i vari messaggi, chiamati in generale *PDU* (*Protocol Data Unit*), relativi al protocollo di comunicazione industriale SafetyNET p.

La struttura generale di una PDU SafetyNET p è costituita da una intestazione di 1 byte, detta *header*, e un campo di dimensione variabile tra 0 e 1499 byte, detto *payload*.

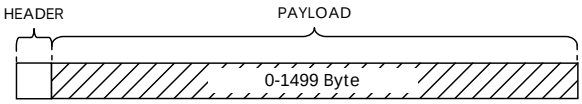


Figura 4.2: struttura del *SNp TYPE frame*

L'*header* contiene un codice per identificare la tipologia di messaggio contenuto nel campo *payload*. Grazie ad esso, il dispositivo SafetyNET p che riceve il frame è in grado di applicare il corretto algoritmo di decodifica del messaggio contenuto nel *payload*. I valori associati al campo *header* permettono di discriminare fra diverse tipologie di messaggi SafetyNET p, non solo limitati al differente canale utilizzato per la trasmissione dei dati, CDC o MSC, ma anche alle differenti funzioni per la gestione della comunicazione.

Sono messe a disposizione diverse tipologie di PDU, per poter svolgere tutte le funzioni necessarie in un protocollo di comunicazione industriale real-time orientato alla *safety*. Esse sono distinte in:

- **Communication Management PDU**: per la gestione della comunicazione;
- **CDC PDU**: per la trasmissione di dati ciclici;
- **MSC PDU**: per trasmissione di dati aciclici.

I differenti *data object* dell'*application layer*, PDO, SDO, SPDO e SSDO, vengono estratti e mappati nelle varie tipologie di PDU mediante apposite interfacce, chiamate *protocol machine*.

Verranno ora elencate le tipologie di PDU più comuni, appartenenti a questa classificazione, che vengono normalmente utilizzate in una comunicazione basata sul protocollo SafetyNET p.

4.1 Communication Managment PDU

Le PDU che appartengono a questa categoria forniscono degli strumenti che permettono di gestire la comunicazione RTFN e interrogare i dispositivi della rete. In particolare, vengono distinte due tipologie:

- **RTFN scan network read PDU**: per effettuare la scansione dei dispositivi connessi alla rete;

- **RTFN connection managment PDU**: per gestire la comunicazione CDC tra i dispositivi.

4.1.1 RTFN scan network read PDU

Le PDU *RTFN scan network read* (*RTFNSNR*) permettono di richiedere e ottenere informazioni riguardanti la topologia della rete e l'hardware ad essa collegato. Questa funzionalità è necessaria per effettuare la diagnosi della rete.

In base al valore assunto dall'*header* è possibile distinguere due tipi di messaggi disponibili:

- **RTFNSNR request**;
- **RTFNSNR response**.

Ogni dispositivo RTFN che riceve una richiesta *RTFNSNR request* risponde in *multicast* con un messaggio *RTFNSNR response*, per comunicare a tutti i dispositivi della rete la propria configurazione.

Questo frame assumerà un ruolo rilevante per il raggiungimento dell'obiettivo di questa tesi, perchè grazie ad esso è possibile venire a conoscenza delle coppie di indirizzi MAC e IP associati a ciascun dispositivo della rete.

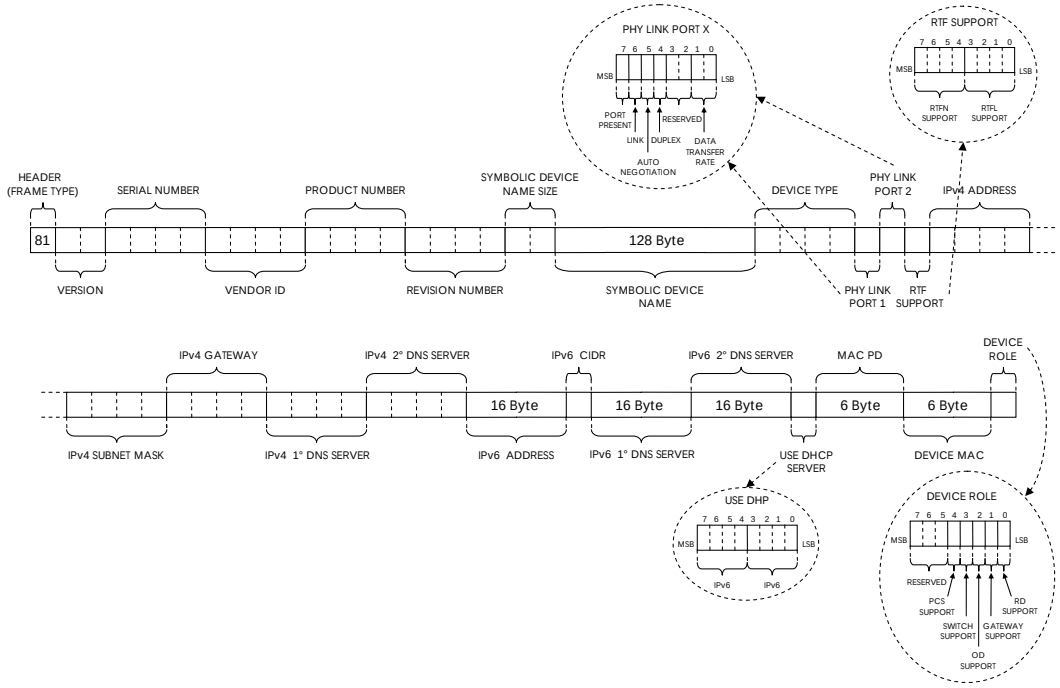


Figura 4.3: struttura di un messaggio *RTFNSNR response*

RTFNSNR request

Il messaggio denominato *RTFNSNR request* permette di richiedere la scansione della topologia della rete. La struttura di questo messaggio è la più semplice possibile dal punto di vista di byte utilizzati: l'*header* assume il valore esadecimale `0x80` e il *payload*, invece, viene lasciato vuoto.

RTFNSNR response

Il messaggio di risposta *RTFNSNR response* possiede, invece, una struttura molto più voluminosa in termini di byte e viene differenziato dal precedente tramite l'*header*, che assume il valore `0x81`.

Con questo messaggio vengono forniti tutti i dati relativi alla configurazione del dispositivo a cui è stata fatta la richiesta di scansione della topologia di rete. Tra le varie informazioni che vengono trasmesse, ci sono il serial number del dispositivo, la sua velocità di trasmissione dati, il tipo di protocollo RTF supportato e soprattutto i suoi indirizzi MAC e IP.

4.1.2 RTFN connection managment PDU

Le PDU *RTFN connection managment* (*RTFNCM*) sono necessarie per la gestione e il corretto funzionamento della comunicazione CDC, che nel caso specifico di RTFN viene chiamata CDCN. A tale scopo, vengono distinti i messaggi:

- **CDCN subscribe**: che utilizzano il modello di comunicazione dati *confermato*, e quindi sono costituiti da una *request* e un *acknowledge*;
- **CDCN still alive**

CDCN subscribe

La coppia di messaggi confermati **CDCN subscribe request** e **CDCN subscribe acknowledge** sono necessari rispettivamente per richiedere la sottoscrizione ai dati, di qualsiasi tipologia, pubblicati da un dispositivo e per confermare l'accettazione di una sottoscrizione.

La struttura di questi due messaggi è la medesima ed essi si differenziano solo per il valore del campo *header*: `0x40` per la *request* e `0x41` per la *response*.

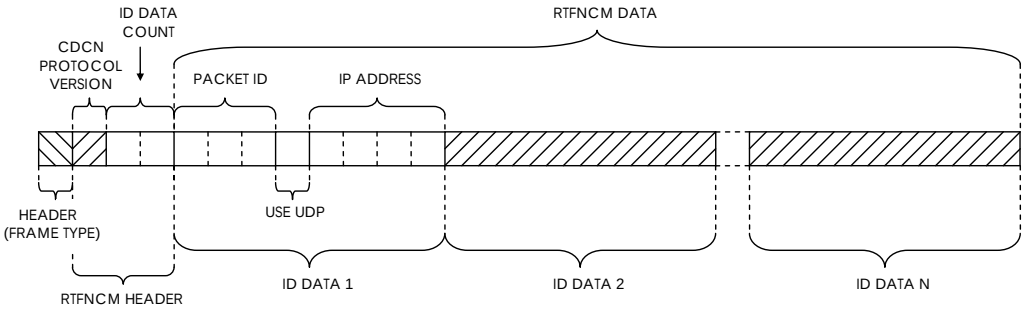


Figura 4.4: struttura dei messaggio *CDCN subscribe request* e *acknowledge*

Questi messaggi servono per creare l'effettivo invio dei dati tra i dispositivi, cioè per creare il canale di comunicazione dati ciclico CDCN.

Come si può notare dalla Figura 4.4, all'interno della struttura di questi frame *CDCN subscribe request* e *CDCN subscribe acknowledge* possono essere accodati più *ID data*, che contengono i 4 byte costituenti l'indirizzo IP del *subscriber*, un

byte per indicare l'utilizzo del protocollo di trasmissione UDP e i 3 byte del *PID* (*Packet IDentifier*).

Il PID è un codice univoco che viene associato a ciascun pacchetto dati ciclici per identificarne univocamente il tipo di informazione trasportata. Ad esempio, ogni dato è associato ad una variabile del programma di controllo e ognuna di esse, che deve essere trasmessa ad un nodo remoto, è associata univocamente ad un PID. In Figura 4.5 è rappresentata una possibile associazione dei PID ai messaggi trasmessi in una comunicazione tra due dispositivi PSSu in cui uno invia all'altro tre tipi di dati riferiti al valore di altrettante variabili. Queste variabili possono poi essere utilizzate per elaborazioni successive, se il dispositivo ricevente è un dispositivo PSSu PLC, oppure semplicemente mappate sulle uscite dei suoi modulo I/O.

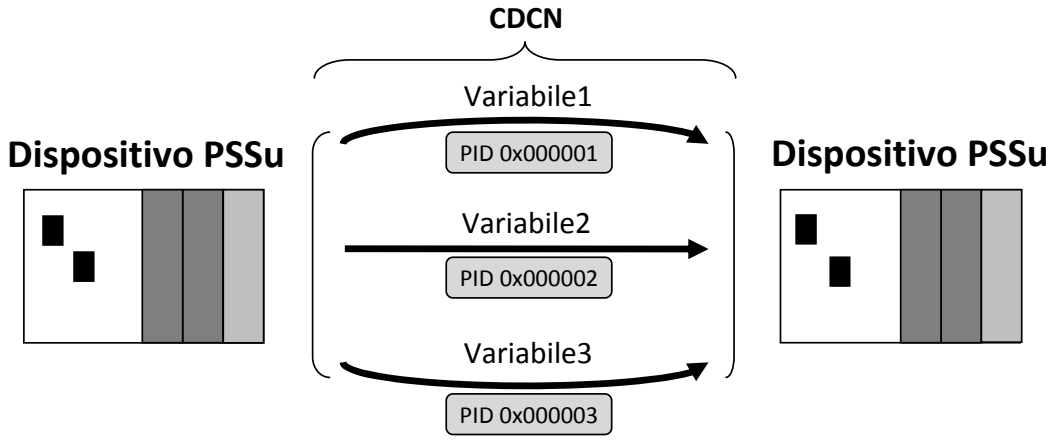


Figura 4.5: esempio di associazione dei PID

Il messaggio *CDCN subscribe request* è utilizzato da un dispositivo *subscriber* per comunicare al *publisher* la tipologia dei messaggi a cui richiede sottoscriversi. Per indicare il tipo di messaggi a cui è interessato ad abbonarsi, il *subscriber* accoda nella PDU i PID ad essi relativi.

In seguito alla ricezione del frame *CDCN subscribe acknowledge*, in cui il dispositivo *publisher* conferma al *subscriber* i PID dei dati da lui generati che accetta di sottoscrivere, comincia l'invio ciclico dei dati richiesti ed inizia così la comunicazione da *publisher* a *subscriber*.

Nel caso in cui in una rete ci siano più dispositivi che necessitano di scambiarsi dati, deve essere attuata questa comunicazione confermata *CDCN subscribe* da parte di ciascun dispositivo. Ad esempio, se due dispositivi devono scambiarsi dati nei due sensi, questa richiesta di sottoscrizione dovrà essere effettuata distintamente da ciascuno di essi nei confronti dell'altro, per un totale di due comunicazioni CDCN subscribe.

CDCN connection still alive

Il frame *CDCN connection still alive* rappresenta il messaggio, scambiato ciclicamente fra i dispositivi facenti parte di una comunicazione, utilizzato per dare informazione del proprio stato di funzionamento. Questa PDU deve solo mettere a conoscenza il destinatario che il mittente è, letteralmente, "ancora vivo" e che

quindi è abilitato alla comunicazione. La sua struttura è costituita dall'*header* che assume il valore esadecimale 0x43 e il *payload* che, invece, viene lasciato vuoto.

Per una comunicazione *standard*, dove la trasmissione dei dati ciclici non è confermata, se il flusso di dati fra due dispositivi avviene in un solo senso, per il mittente non è possibile discriminare lo stato del destinatario e per questo è stato introdotto il frame *CDCN connection still alive*.

Come si vedrà nella Sezione 4.3.2, quando vengono trattati messaggi contenenti dati *safe*, il protocollo utilizza un servizio aggiuntivo che, basato su una comunicazione confermata, permette un più dettagliato monitoraggio dello stato dei dispositivi.

4.2 MSC PDU

Nelle PDU che appartengono a questa categoria vengono mappati tutti i dati acicli che non necessitano di elevate prestazione *real-time*, come ad esempio dati per il monitoraggio dello stato dei dispositivi, per la loro configurazione e per la diagnostica in caso di errore.

La struttura generale di una MSC PDU SafetyNET p è formato da un *header* che assume il valore esadecimale 0x70 e da un campo di dimensione variabile chiamato *MSC data*.

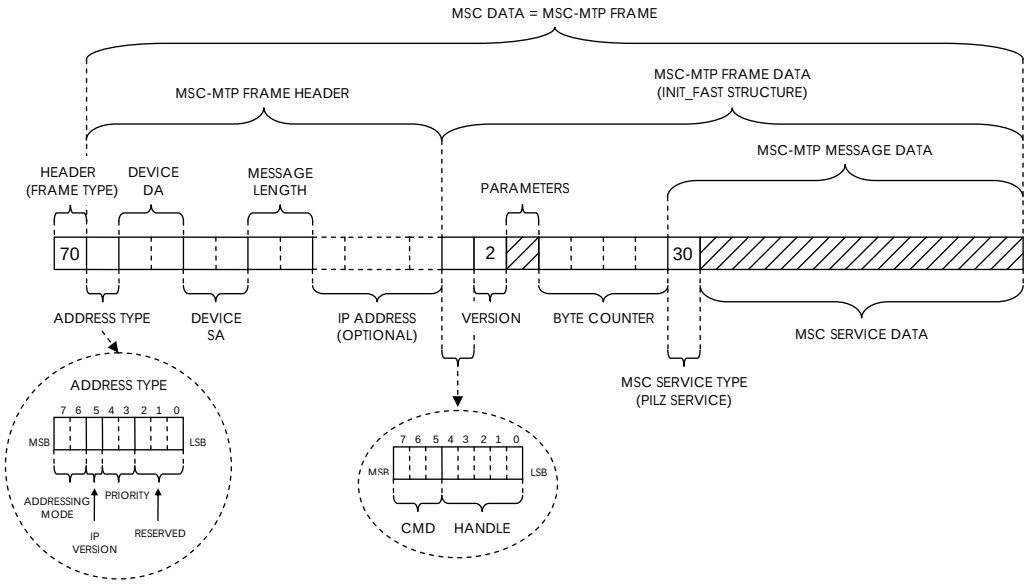


Figura 4.6: struttura dei messaggi *MSC*

All'interno dell' *MSC data* è inserito l'*MSC-MTP frame*, il campo in cui vengono trasmessi i messaggi aciclici mediante l'utilizzo del protocollo *MSC-MTP*.

L'intestazione *MSC-MTP frame header* contiene i campi relativi all'indirizzamento del messaggio trasmesso all'interno dell'*MSC-MTP frame data*, permettendo di impostarne anche la priorità tramite i bit *priority* del campo *address type*.

L'intestazione dell'*MSC-MTP frame data* è costituita dai campi *CMD*, *handle*, *version*, *parameters* e *byte counter*. In genere, il protocollo MSC-MTP invia,

nel campo *MSC-MTP frame data*, un messaggio di inizializzazione al dispositivo di destinazione proponendo l'apertura di una sessione di comunicazione, che viene identificata mediante il campo *handle*, costituito da 5 bit. Il dispositivo di destinazione risponde dando la confermando alla comunicazione e imponendo la dimensione massima del segmento (*Maximum Segment Size*,*MSS*) *MSC-MTP frame data* che può essere trasferito. Per verificare che sia stata rispettata la MSS viene controllato il numero dei byte da trasmettere nell'*MSC-MTP frame data*, riportato nel *byte counter*. Nel caso in cui la dimensione dei dati da trasmettere sia superiore all'MSS, il protocollo MSC-MTP mette a disposizione delle procedure di segmentazione, per rispettare il limite imposto sul volume di byte inviabili in un unico frame. Il campo *version* indica la versione dell'implementazione del protocollo SafetyNET p con cui deve essere decodificato il messaggio. Quando assume il valore esadecimale 0x02, rispetto alla versione precedente, nella decodifica viene inserito un ulteriore campo *parameters*, utilizzato per specificare eventuali parametri. Infine, il campo *CMD* è utilizzato, in analogia con il *PID* della comunicazione ciclica, per identificare le differenti tipologia dei messaggi *MSC-MTP frame data* disponibili.

La sezione *MSC-MTP message data*, dell'*MSC-MTP frame data*, contiene il campo *MSC service type*, che identifica univocamente la tipologia dei dati, e l'*MSC service data*, che riporta il messaggio effettivamente trasmesso mediante il protocollo di comunicazione MSC-MTP. All'interno di questo campo ogni produttore di dispositivi SafetyNET p può inserire i dati aciclici, necessari per le proprie funzioni di diagnostica e di monitoraggio, creandosi un'apposita struttura di codifica delle informazioni. Per i dispositivi utilizzati durante l'analisi sperimentale, in questo campo sono contenuti i *Pilz service message* che hanno la struttura rappresentata in Figura 4.7, con l'*MSC service type* che assume il valore esadecimale 0x30. Viene tralasciata una descrizione dettagliata dei frame, in quanto l'MSC non influenza i meccanismi che si vogliono analizzare in questa tesi, tutti implementati a livello CDC.

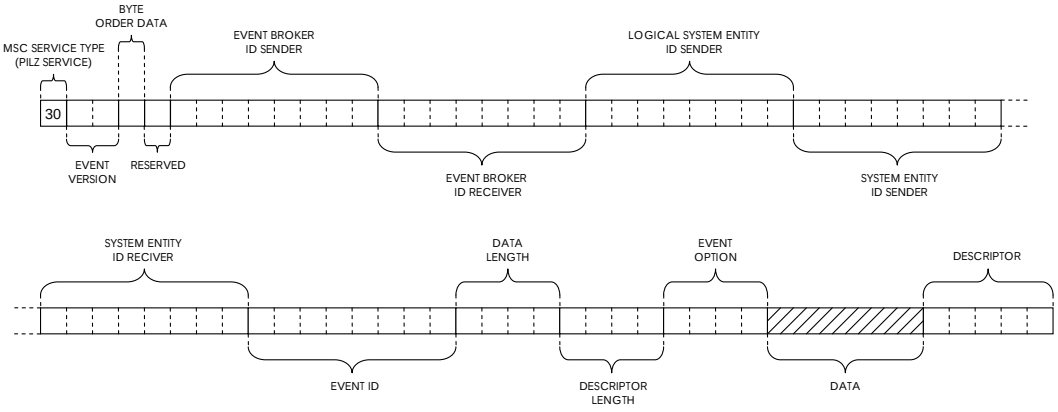


Figura 4.7: struttura dell'*MSC-MTP message data* specifico per i *Pilz service*

I messaggi MSC sono confermati. Il dispositivo mittente si aspetta un'*acknowledge* ai segmenti inviati, entro un timeout prestabilito e dopo una quantità preconfigurata di dati ciclici di processo inviati. Se questa conferma non perviene, i pacchetti

non confermati vengono nuovamente trasmessi al destinatario per un numero massimo di tentativi di invio che può essere preimpostato. La trasmissione termina dopo l'invio dell'ultimo segmento e la ricezione della rispettiva conferma di ricezione, o dall'interruzione della comunicazione che può essere inviata, in qualsiasi momento, dal destinatario. Se il trasferimento non riesce, per problemi di timeout o per l'interruzione della comunicazione, viene segnalato da un appropriato messaggio di errore.

4.3 CDCN PDU

Le *CDCN PDU* vengono utilizzate per trasmettere ciclicamente i dati di processo. A queste vengono affiancate le *RTFNCM PDU*, necessarie per garantire la corretta gestione della comunicazione del canale CDC.

Come si può notare in Figura 4.8, la sua struttura è costituita dall'*header* che assume il valore esadecimale 0x60, da un *CDCN header* che contiene i campi *CDCN protocol version*, per specificare la versione del protocollo da utilizzare per la decodifica, e *size*, che riporta la dimensione in byte dell'intera *CDCN PDU*, e da un *CDC payload*.

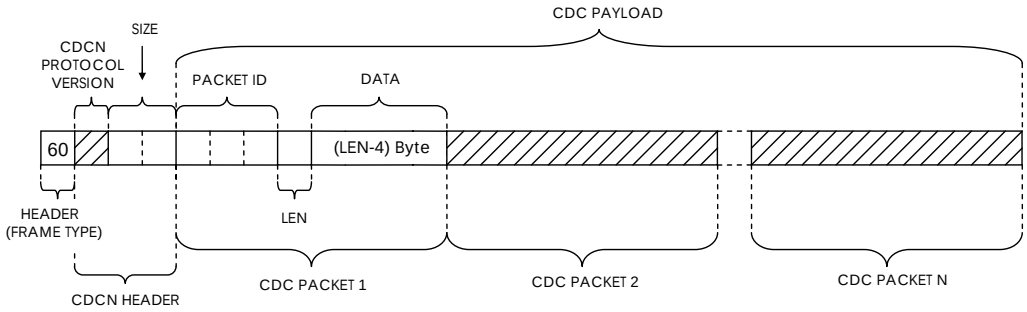


Figura 4.8: struttura del messaggio CDCN

Nel *CDC payload* possono essere accodati più *CDC packet* all'interno dello stesso messaggio. Ciascuno di questi *CDC packet* è composto dai 3 byte che costituiscono il *PID*, un byte per la lunghezza *len* di tale pacchetto e il campo *data* in cui sono contenuti i [len-4] byte che rappresentano la codifica del dato ciclico trasmesso.

All'interno del campo *data* di ciascun *CDC packet* vengono trasmessi i dati di processo, chiamati a livello applicativo *data object*, che si differenziano in *standard*, *PDO* (*Process Data Object*) , e *safe*, *SPDO* (*Safe Process Data Object*). A seconda del tipo di dati da trasferire, *standard* o *safe*, il pacchetto ciclico si differenzia per la struttura del campo *data*.

La mappatura delle *PDO* e *SPDO* viene effettuata tramite l'interfaccia fra l'*application layer* e il sistema di comunicazione, *protocol machine*. Viene utilizzato un *object dictionary*, in cui all'interno di questa tabella vengono inserite le codifiche delle varie tipologie di oggetti utilizzati a livello applicativo. Ciascun dispositivo, ogni volta che deve effettuare delle procedure di manipolazione logica delle variabili, lavora a livello applicativo con gli oggetti e poi, tramite il codice

posseduto da ciascuna oggetto, viene selezionata la rispettiva riga della tabella del *object dictionary*, in cui è specificata la codifica che deve essere effettuata, a livello di trasmissione, per il dato specifico.

4.3.1 PDO

I PDO vengono mappati in un *CDC packet* che mantiene la struttura sopra definita: il campo *PID* per identificare univocamente la tipologia dei dati da trasmettere, il *len* per riportare la lunghezza del pacchetto da decodificare, e il *data* contente la codifica dei dati di processo *standard*.

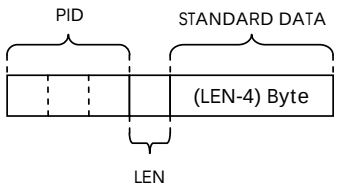


Figura 4.9: struttura del campo *CDC packet* per la trasmissione di dati *standard*

I dati di processo *standard* vengono codificati direttamente nel campo *data* del *CDC packet*, senza che il meccanismo del *black channel* ne effettui alcuna elaborazione.

4.3.2 SPDO

Nel caso si comunicazioni *safe*, i dati di processo sono trasmessi tramite il medesimo canale di trasmissione CDC per mezzo di *SPDO*, che possiedono una struttura più complessa rispetto alle *PDO* della comunicazione *standard*, come si può notare in Figura 4.10.

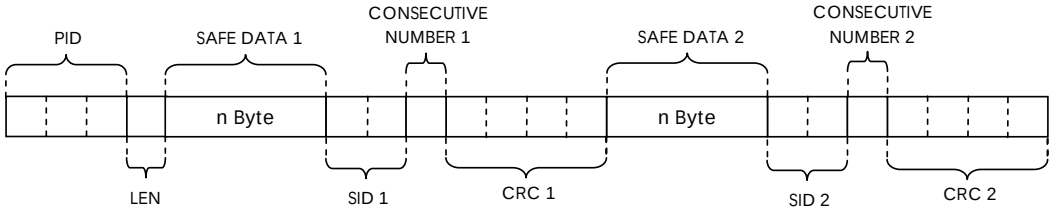


Figura 4.10: struttura del campo *CDC packet* per la trasmissione di dati *safe*

Le SPDO contengono, oltre ai campi *PID* e *len*, i dati di processo *safe*, all'interno del campo *safe data*, e l'implementazione delle contromisure per il rilevamento gli errori di comunicazione menzionati in Fiugura 3.5 di pag 31.

Nello specifico, il campo *SID* di 2 byte, accoppiato con il PID costituisce il codice univoco (*connection ID*) per identificare il mittente del messaggio. Il *consecutive number* di un byte, permette di monitorare la sequenza dei pacchetti inviata mentre il *CRC* di 4 byte, che contiene il *checksum* calcolato mediante il polinomio 0x20044009 sui campi *PID*, *len*, *safe data*, *SID* e *consecutive number*, permette di

verificare l'integrità del pacchetto (*data integrity assurance*). Infine, la *different integrity assurance* è garantita dalla ridondanza (duplice copia) dei campi *safe data*, *SID*, *consecutive number* e *CRC*.

SHB PDU

Dato che le comunicazioni *safe* devono soddisfare determinati requisiti di sicurezza, viene effettuato il monitoraggio della trasmissione nell'*application layer*.

Il protocollo di rete SafetyNET p deve garantire funzionalità real-time, e per questo anche la comunicazione ciclica di dati di processo *safe* è costituita da uno scambio dati non confermato. Dal lato destinatario, è normalmente utilizzato un timeout, entro cui ci si aspetta di ricevere i dati di processo, che permette di individuare eventuali errori di comunicazione. Ma, dato che le comunicazioni *safe* devono soddisfare determinati requisiti relativi alla *safety*, a causa dello scambio di dati non confermato è stato necessario introdurre un meccanismo aggiuntivo che consentisse di rilevare il fallimento di un dispositivo e che, inoltre, permettesse di rilevare un aumento del ritardo nella consegna delle SPDO oltre il tempo di attesa del destinatario. Questo monitoraggio viene implementato tramite un meccanismo, chiamato *Safe HeartBeat* (*SHB*), che consiste nello scambio ciclico di messaggi confermati fra i dispositivi che effettuano una comunicazione *safe*.

La comunicazione SHB utilizza un tempo ciclo per la trasmissione dei messaggi che è indipendente da quello utilizzato per la comunicazione di dati di processo *safe*.

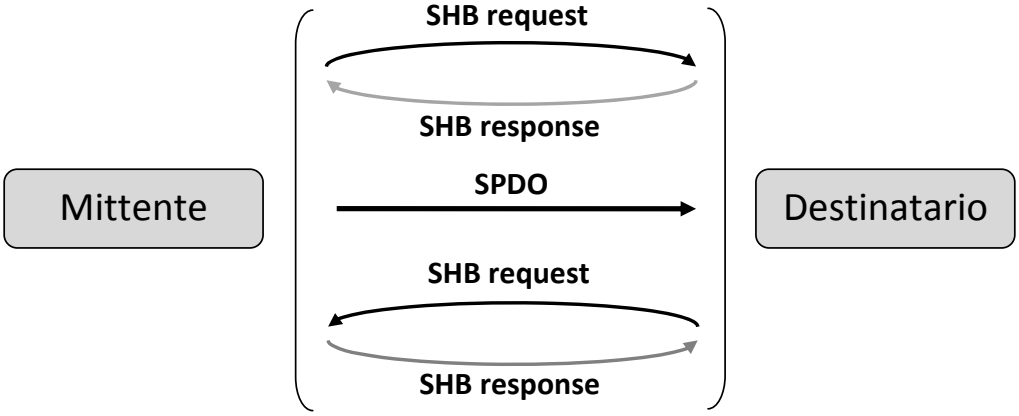


Figura 4.11: modello di funzionamento dell'SHB

La Figura 4.11 descrive il modello di comunicazione SHB per una trasmissione *safe* fra due dispositivi, in cui è stato ipotizzato che il dispositivo di sinistra invii ciclicamente dei dati *safe* a quello di destra. In questa configurazione si vede che i messaggi SHB vengono inoltrati da ciascun dispositivo per sapere lo stato di funzionamento dell'altro. Essendo SHB una comunicazione confermata, ogni *SHB request* viene seguita da un *SHB response*, la cui ricezione conferma al dispositivo richiedente la correttezza della comunicazione e lo stato del destinatario.

Nello specifico, il messaggio **SHB request** possiede la stessa struttura di un

CDC packet relativo a dati di processo *safe*, in cui però il campo *safe data* viene sostituito dal *fail-safe AL state* e *fail-safe AP state*.

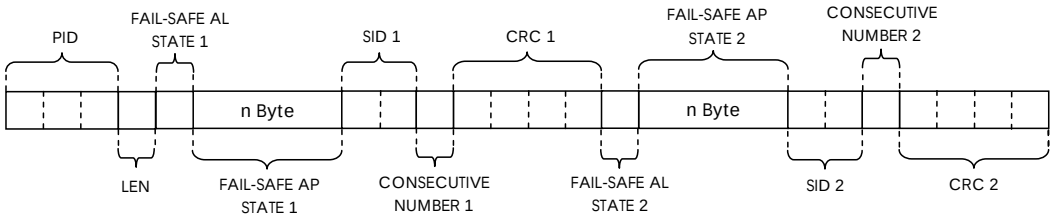


Figura 4.12: struttura del messaggio *SHB request*

Il campo *fail-safe AL state* contiene la codifica dell'informazione sullo stato dell'*Application Layer* (*AL*) relativo alla *safety*, eseguita secondo la Tabella 4.1.

Valore	Descrizione
0x00	FS AL è nello stato START-UP
0x04	FS AL è nello stato STOPPED
0x05	FS AL è nello stato OPERATIONAL
0x7F	FS AL è nello stato PRE-OPERATIONAL

Tabella 4.1: byte per la codifica del *fail-safe AP state*

Gli stati dell'*application layer* dei dispositivi SafetyNET p sono raffigurati in Figura 4.13.

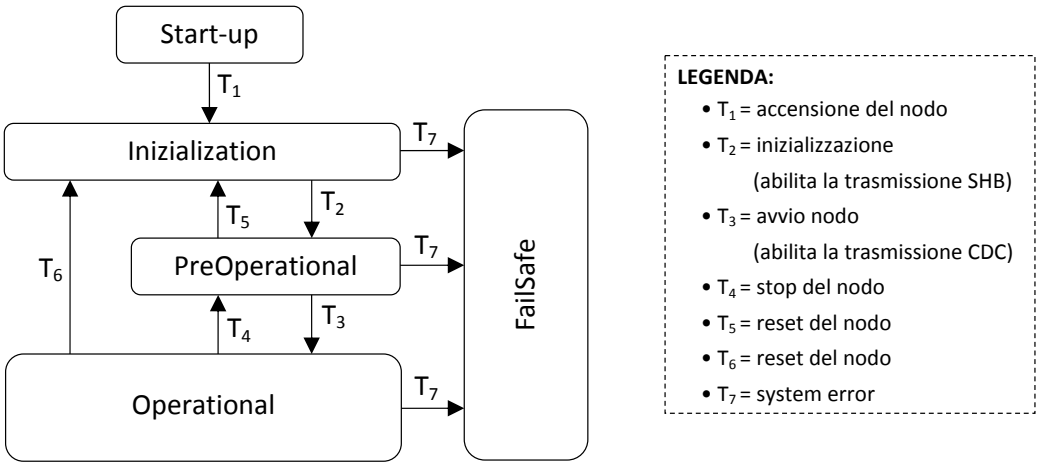


Figura 4.13: macchina a stati dell'*application layer* dei dispositivi SafetyNET p

Il campo *fail-safe AP state* contiene la codifica dell'informazione sullo stato dell'applicazione di processo (*AP*, *Application Process*) *safe* e può essere costituito al massimo da 116 byte. Il contenuto e la codifica di questo campo dipendono solamente dalle scelte progettuali dei produttori di dispositivi SafetyNET p, come Pilz, e quindi non sono descritti nello standard di tale protocollo.

Per quanto riguarda il messaggio **SHB response**, siccome svolge solo la funzione di *acknowledge*, non necessita di portare alcuna informazione al destinatario e per questo elimina completamente i campi *fail-safe AL state* e *fail-safe AP state*. La struttura viene mantenuta ridondata e con i meccanismi di rilevazione degli errori perchè si tratta di una comunicazione che comunque deve garantire i requisiti relativi alla *safety*.

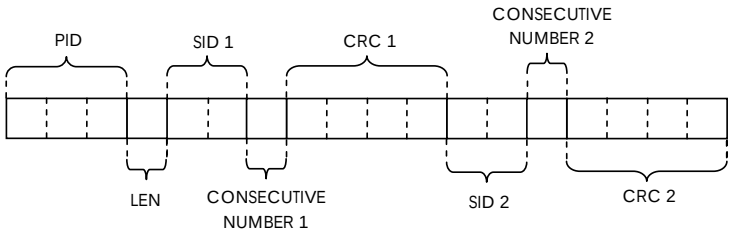


Figura 4.14: struttura del messaggio *SHB response*

Oltre a permettere un monitoraggio reciproco sullo stato dei nodi della rete, il meccanismo di *Safe HeartBeat* consente al servizio *Safe Delay Monitoring* (*SDM*) di monitorare il ritardo dei pacchetti durante una comunicazione *safe*. Il servizio controlla che il tempo tra l'invio di un *SHB request* e la ricezione di un *SHB response* non superi un ritardo massimo prestabilito e configurabile, come mostrato in Figura 4.15.

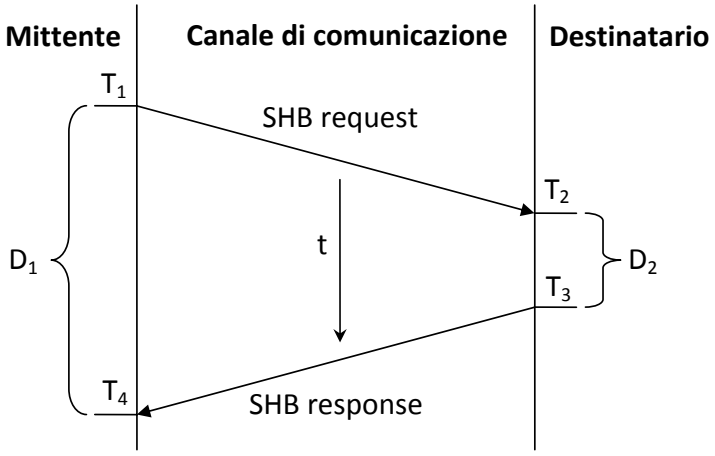


Figura 4.15: principio del meccanismo SDM

Al dispositivo mittente sono noti solamente i tempi di invio T_1 e di ricezione T_4, e quindi la differenza temporale D_1. Sulla base di queste informazioni, il mittente della richiesta può determinare una stima del ritardo nella ricezione del pacchetto da parte del destinatario. Tale ritardo viene confrontato con il valore di soglia impostato: se è rilevato un aumento del ritardo, che supera la soglia configurata, verrà applicata una opportuna reazione.

La procedura di SDM deve essere eseguita da tutti i dispositivi coinvolti in una comunicazione *safe* per determinare il ritardo effettivo nella consegna dei messag-

gi SHB, che permette di stabilire la validità delle informazioni ricevute. Questo meccanismo realizza la contromisura *Time-out*, che fa parte delle misure di individuazione degli errori riportate in Figura 3.5 di pag. 31, utilizzato per rilevare ritardi sulla rete (*Unacceptable delay*).

CAPITOLO

5

Strumenti di Analisi

In questo capitolo sono presentati tutti gli strumenti, sia hardware che software, utilizzati per lo svolgimento delle analisi del protocollo SafetyNET p. Esso permette di fornire le nozioni riguardanti tutti gli elementi di base che compongono l'apparato sperimentale, così da poter affrontare i successivi capitoli di analisi del protocollo specificando solamente le configurazioni utilizzate in ciascun caso.

Al fine di osservare il flusso dei messaggi scambiati nel corso di una comunicazione basata sul protocollo SafetyNET p, sono necessari degli strumenti specifici: una *rete di test* SafetyNET p e degli strumenti, sia hardware che software, che possono essere inseriti in una categoria chiamata *strumenti per l'analisi della rete*.

La *rete di test* è costituita dal sistema fisico PSS4000, prodotto da Pilz GmbH & Co.KG[1]. Essa rappresenta la struttura centrale perchè realizza l'implementazione del protocollo SafetyNET p, descritto nelle sue caratteristiche generiche nel precedente Capitolo 3. Con i dispositivi di questo sistema è stato possibile realizzare diverse configurazioni, necessarie per le varie sperimentazioni. Nello specifico, i vari dispositivi del sistema PSS4000 avuti a disposizione sono distinguibili in:

- *PLC*, denominati *PSSU H PLC1 FS SN SD*,
- *dispositivi I/O*, denominati *PSSU H FS SN SD*,

ciascuno dei quali supporta il modello di comunicazione RTFN e può essere dotato di diversi moduli di uscita ed ingresso digitale, sia per i segnali *standard* che *safe*.

[1]Pilz GmbH & Co.KGè una società che opera nel campo della sicurezza tecnica di automazione, con soluzioni complete per la sicurezza e tecnologia di controllo. Oltre al suo quartier generale a Ostfildern, Germania, Pilz è rappresentata da filiali e succursali a un totale di 26 paesi del mondo e impiega attualmente 1.400 dipendenti.

Gli *strumenti per l'analisi della rete* sono stati utilizzati per osservare i messaggi del protocollo SafetyNET p fra i nodi della *rete di test*. Grazie a questi strumenti è stato possibile analizzare in dettaglio ogni singola componente dei messaggi, fattore fondamentale per la caratterizzazione sperimentale del protocollo, oggetto di questa tesi. Inoltre, è stato possibile generare e iniettare traffico anomalo sulla *rete di test* per simulare errori, con lo scopo di visualizzare il comportamento dei singoli nodi in seguito al loro verificarsi.

Gli *strumenti per l'analisi della rete* utilizzati in questa tesi, differenziati in software e hardware, sono stati:

- un *Personal Computer* (*PC*) con due schede di rete, *eth0* ed *eth1*, su cui sono stati installati gli strumenti software necessari allo svolgimento delle analisi. In particolare:
  - la piattaforma software *PAS4000*, prodotta da Pilz GmbH & Co.KG, che permette la programmazione dei dispositivi del sistema PSS4000;
  - l'*analizzatore di rete Wireshark*, che permette di visualizzare il flusso di traffico tra i dispositivi della rete;
  - il generatore di pacchetti *packETH*, che permette di iniettare pacchetti nella rete, al fine di studiare gli effetti prodotti dalla presenza di traffico anomalo durante una comunicazione basata su SafetyNET p.
- un *managed switch*, con funzione di *port mirroring* fra due delle sue porte, per collegare i dispositivi e permettere l'analisi del traffico.

Per capire l'interazione che intercorre tra le due componenti, *rete di test* e *strumenti per l'analisi della rete*, si considera ora la struttura di connessione riportata in Figura 5.1, che rappresenta una delle basilari configurazioni utilizzate durante le analisi.

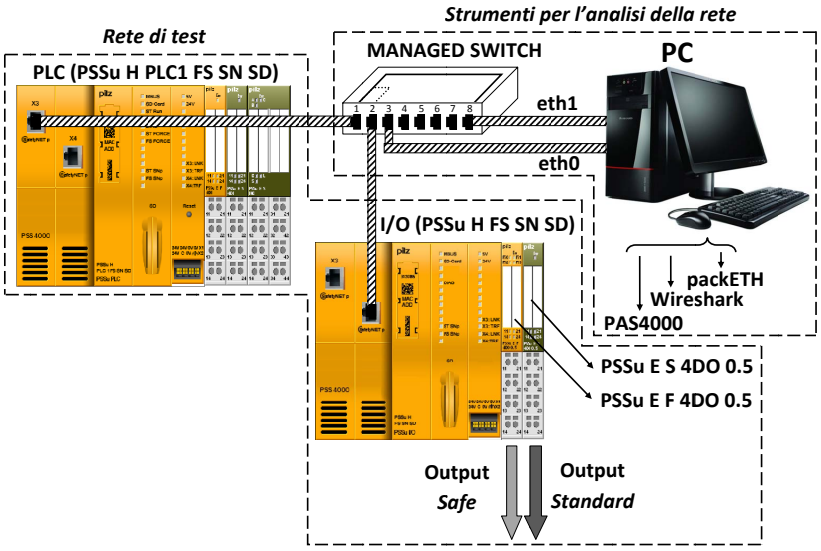


Figura 5.1: esempio di struttura del *set up* per le analisi

In questa specifica struttura il PLC, il modulo I/O e le due schede di rete *eth0* ed *eth1* del PC sono collegati attraverso il *managed switch*. Nello switch è stata configurata la funzione di *port mirroring* tra la porta 1, a cui è connesso il PLC, e la 8, a cui è connessa la scheda di rete *eth 1*. Il modulo I/O, invece, è stato connesso ad una qualsiasi porta libera, senza alcuna configurazione speciale.

Durante gli esperimenti sono stati sviluppati ed eseguiti sul PLC dei semplici programmi, che generano dei segnali di input e output e li inviano, tramite la rete Ethernet SafetyNET p, ad un nodo remoto per la loro attuazione. Questi programmi sono stati realizzati ed installati sulla *rete di test* tramite la suite software PAS4000. Nello specifico, la semplicità dei programmi ha permesso di distinguere le fondamentali differenze intercorrenti fra la comunicazione di dati *standard* e quella di dati *safe* e come in una comunicazione mista possano coesistere.

L'analisi dei messaggi, trasferiti durante l'esecuizione dei programmi di test, è resa possibile grazie alla funzione di *port mirroring*, disponibile sullo switch, che permette di replicare il traffico fra il PLC e il modulo I/O alla scheda di rete *eth1* del PC, su cui è in "ascolto" l'*analizzatore di traffico Wireshark*. In tale programma è risultato utile il *dissezionatore* del protocollo SafetyNET p sviluppato da Pilz GmbH & Co.KG, che è in grado di riconoscere il formato SafetyNET p dei messaggi e isolare i vari campi in una forma facilmente leggibile. Per quanto riguarda l'iniezione dei pacchetti nella *rete di test* è stata utilizzata una seconda scheda di rete *eth1* del PC, connessa ad una generica porta dello switch.

5.1 Sistema fisico: *PSS4000*

PSS4000 di Pilz GmbH & Co.KG rappresenta un sistema di automazione innovativo e mirato per la gestione di funzioni di comando, sia *standard* che *safe*, movimentazione e visualizzazione. L'utilizzo di un solo dispositivo per il controllo classico e per la safety permette un risparmio di risorse hardware e software; quando, invece, le funzioni sono separate tale risparmio è impedito dall'inevitabile ridondanza di apparecchiature e cablaggi.

Il sistema di comunicazione SafetyNET p rappresenta l'elemento di connessione tra i diversi dispositivi che compongono il sistema PSS4000. SafetyNET p, e quindi Ethernet real-time, è la rete di trasmissione industriale per lo scambio dati di automazione *standard* e *safe*.

Figura 5.2: dispositivo appartenente al sistema PSS4000

Ciascun dispositivo, detto *PSSuniversal*, appartenete a PSS4000 attua l'approccio di sicurezza *failsafe*. Il sistema, in caso di rilevamento di errori sul canale dedicato alla safety, si porta in uno *stato di sicurezza* in cui vengono disabilitate tutte le uscita associate a variabili *safe*, in modo da non procurare danni ad altri dispositivi o situazioni di pericolo per il personale e l'ambiente.

La piattaforma di controllo PSS4000 permette di creare sistemi di controllo decentralizzati connettendo tra di loro due o più nodi. Ogni nodo comunica con gli altri per mezzo del protocollo SafetyNET p ed è composto da due elementi hardware fondamentali:

- Un **modulo principale (o testata)** che contiene la logica per la trasmissione dei dati sulla rete e può avere un'unità di elaborazione integrata per l'esecuzione del programma di controllo.

- Un certo numero di **moduli di I/O** per l'interfacciamento con i sensori e gli attuatori dell'impianto, che si distinguono in moduli per applicazioni *standard* e moduli per applicazioni *safe*.

Il primo dispositivo di ogni nodo è sempre una *testata* e alla sua destra possono essere collegati per mezzo di un bus proprietario (MBUS) dei *moduli I/O*, in ordine arbitrario, per applicazioni *standard* (*ST*) e *safe* (*FS-failsafe*).

Come riportato in Tabella 5.1, ogni nodo può essere costituito da un massimo di 64 moduli I/O, ad esclusione della *testata*. Nel conteggio sono inclusi i *moduli di alimentazione*, i *moduli ST* e i *moduli FS*.

Tipi di modulo I/O	**Max numero per nodo**
Modulo ST	64 moduli
Modulo FS	64 moduli
Bit del modulo ST	240 bit di input 240 bit di output
Bit del modulo ST	256 bit di input 256 bit di output

Tabella 5.1: limitazioni sul numero di *moduli I/O* per nodo

Per la distinzione dei moduli a disposizione del sistema PSS4000 vengono utilizzate delle denominazioni che forniscono informazioni relative alle loro funzioni. Tali denominazioni iniziano con **PSSu** e sono seguite da combinazioni, a più livelli, di lettere e cifre numeriche. Le nomenclature più utilizzate sono: **H** per i moduli principali (*head module*) ed **E** per i moduli I/O elettronici, che a loro volta vengono distinti in **S**, moduli *standard*, e **F**, moduli *safe* (*failsafe*). Sono, inoltre, disponibili moduli impiegati per applicazioni in condizioni ambientali gravose relativamente alla temperatura e all'umidità. Questi moduli vengono indicati come moduli PSSu *Coated Version* e non si differenziano dalle altre versioni dei moduli per quanto riguarda le funzionalità. Per poter essere distinti, sono contrassegnati dalla lettera **-T** alla fine del nome del prodotto.

5.1.1 Moduli principali

Il sistema PSS4000 mette a disposizione dell'utilizzatore due tipologie di testata:

- **PSSuniversal PLC (PSSu PLC1 H FS SN SD)**

  Il dispositivo *PSSu H PLC1 FS SN SD*, è un PLC che possiede una risorsa ST e una FS in cui possono essere inseriti, tramite il programma PAS4000, i vari programmi da eseguire, distinguendo fra applicazioni *standard* e *safe*. Per eseguire le applicazioni *safe* il modulo dispone di uno specifico processore con una struttura pluricanale diversificata, al fine di garantire la ridondanza necessaria per soddisfare i requisiti di sicurezza a cui afferisce.

- **PSSuniversal I/O (PSSu H FS SN SD)**

  Il dispositivo PSSu H FS SN SD viene impiegato come testata per la realizzazione di nodi di I/O decentralizzati senza capacità di elaborazione.

Visto che ogni sistema di automazione deve svolgere determinate funzioni, che vengono scritte all'interno di un programma, è necessario che, all'interno della rete, vi siano dei dispositivi in grado di elaborato e svolgere funzioni di controllo. Per questo, in una rete SafetyNET p realizzata con la piattaforma di controllo PSS4000, deve sempre essere presente almeno un nodo, che abbia come *modulo principale* un PSSuniversal PLC.

Grazie alla combinazione di sistemi decentralizzati PSSuniversal I/O con i sistemi PSSuniversal PLC è possibile ottenere un'efficace decentralizzazione di tutti i segnali periferici in grandi impianti e linee di produzione.

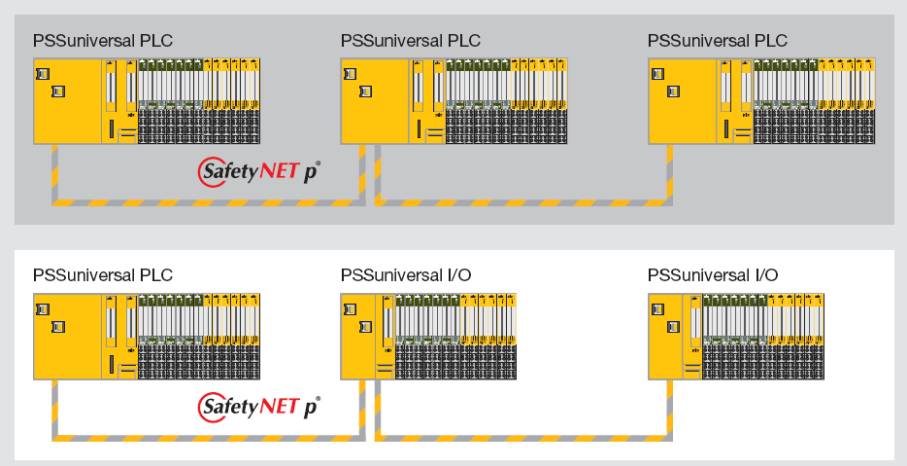
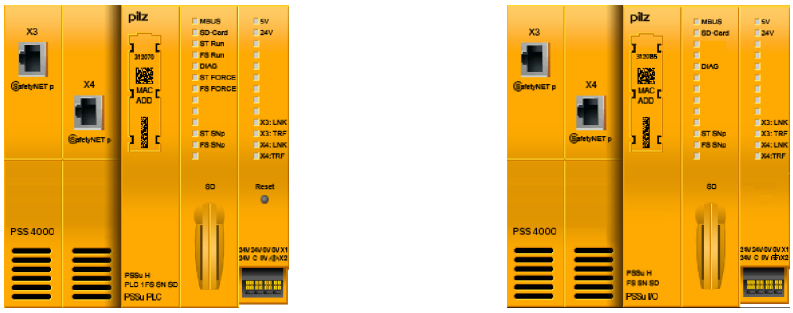


Figura 5.3: esempi di collegamento dei sistemi PSSu

Come si può vedere in Figura 5.4(a) e 5.4(b), le testate possiedono 2 porte Ethernet per il collegamento ad una rete SafetyNET p, appositi LED per l'indicazione dello stato del dispositivo e di eventuali guasti o errori, ed una scheda SD[2] per la memorizzazione del sistema operativo, dell'indirizzamento dei dati e del programma da eseguire. Le porte di rete sono collegate internamente ad uno switch per consentire la connessione in cascata di più testate.

[2] *Secure Digital* (*SD*) è il più diffuso formato di schede di memoria utilizzato per la memorizzare in formato digitale grandi quantità di informazioni all'interno di memorie flash.



(a) PSSuniversal PLC (b) PSSuniversal I/O

Figura 5.4: *moduli principali* dei sistemi PSSu

Un bus interno, detto *MBUS* (*Module BUS*), fornisce l'alimentazione ai moduli del sistema PSSu e trasferisce i dati dal *modulo principale* ai *moduli I/O*. Per il trasferimento delle informazioni sono presenti due diversi bus a seconda del tipo di informazione, *standard* o *safe*, che deve essere trasmessa.

Per quanto riguarda l' alimentazione del sistema PSSu, sono presenti due distinte linee a 24V, denominate *Module Supply* e *Periphery Supply*. Il *Module Supply* fornisce alimentazione alla logica interna del *modulo principale* e dei *moduli I/O*. Il *Periphery Supply*, invece, alimenta gli ingressi e le uscite sui morsetti dei *moduli I/O*. Va notato che il potenziale di massa di ciascun sistema di alimentazione ha collegamenti separati. In questa maniera, utilizzando due alimentatori separati per i due circuiti, il *Module Supply* e il *Periphery Supply* sono separati galvanicamente. Ciò aumenta l'*avaiability* del sistema perchè, in caso di cortocircuito del *Periphery Supply* (e quindi sul campo), viene mantenuto il *Module Supply* e, dunque, il *modulo principale* può continuare a scambiare i dati tramite la rete SafetyNET p.

Tutte le funzioni di trasmissione e di alimentazione dell'MBUS sono gestite e realizzate, logicamente e fisicamente, dai *moduli principali* dei sistemi PSSu. Infatti, sulla loro scocca, sono presenti sia due alimentazioni contrassegnate con X1 e X2, rispettivamente il *Module Supply* e il *Periphery Supply*, che le luci relative allo stato di funzionamento dell'MBUS.

L'interfaccia di rete permette il trasferimento sia di dati standard che di dati relativi alla safety tra i nodi. Durante la comunicazione, che utilizza il protocollo SafetyNET p, il modulo principale riceve segnali da altri nodi, li elabora secondo il programma utente e li inoltra ai *moduli I/O*, collegati tramite l'MBUS. In caso di guasto del sistema o errore su un dato safe, il *modulo principale* commuta le uscite *safe* in uno stato sicuro.

L'indirizzo MAC, necessario per la comunicazione tramite SafetyNET p, di ciascun dispositivo è preconfigurato ed è indicato sulla parte frontale del *modulo principale*.

☼	LED on
◐	LED lampeggiante
●	LED off

Figura 5.5: legenda dello stato dei LED

Vengono di seguito riportate le tabelle che descrivono la relazione fra l'indicazione fornita dai LED dei *moduli principali* e lo stato di funzionamento delle loro diverse parti. In queste tabelle, per indicare il comportamento dei LED, si farà riferimento alla legenda rappresentata in Figura 5.5.

MBUS

Il LED *MBUS* indica lo stato dei bus per i moduli I/O FS e ST ed è presente in entrambi i *moduli principali*.

Colore	Stato	Significato
- - -		nessun modulo disponibile
verde		I bus del modulo FS ed ST funzionano correttamente
rosso		non è stato possibile raggiungere almeno un modulo (ad es. durante il funzionamento è stato rimosso un modulo, la configurazione impostata e la configurazione prevista non corrispondono)
		Bus del modulo FS con errore in stato di STOP: grave errore FS

Figura 5.6: significato dello stato del LED *MBUS*

ST SNp

Il LED *ST SNp* indica lo stato del collegamento della parte *standard* del dispositivo a SafetyNET p ed è presente in entrambi i *moduli principali*.

Colore	Stato	Significato
- - -		Il collegamento ST a SafetyNET p non è attivo
verde		Collegamento ST a SafetyNET p avviato correttamente e in stato RUN
		Collegamento ST a SafetyNET p con errore non grave e in stato RUN:
rosso		Collegamento ST a SafetyNET p con errore e in stato STOP: errore ST non grave
		Collegamento ST a SafetyNET p con errore e in stato STOP: grave errore FS ed ST

Figura 5.7: significato dello stato del LED *ST SNp*

FS SNp

Il LED *FS SNp* indica lo stato del collegamento della parte *safe* del dispositivo a SafetyNET p ed è presente in entrambi i *moduli principali*.

Colore	Stato	Significato
- - -		Il collegamento FS a SafetyNET p non è attivo
verde		Collegamento FS a SafetyNET p avviato correttamente e in stato RUN
		Collegamento FS a SafetyNET p con errore non grave e in stato RUN:
rosso		Collegamento FS a SafetyNET p con errore e in stato STOP: grave errore FS
		Collegamento FS a SafetyNET p con errore e in stato STOP: grave errore FS ed ST

Figura 5.8: significato dello stato del LED *FS SNp*

5V e 24V

Il LED *5V* indica lo stato dell'alimentazione *Module Supply*.

Colore	Stato	Significato
- - -		Nessuna alimentazione o guasto all'alimentazione di "Module Supply"
verde		"Module Supply" è disponibile sul bus del modulo

Figura 5.9: significato dello stato del LED *5V*

Il LED *24V* indica lo stato dell'alimentazione *Periphery Supply*.

Colore	Stato	Significato
- - -		Nessuna alimentazione o guasto all'alimentazione di "Periphery Supply"
verde		"Periphery Supply" è disponibile sul bus del modulo

Figura 5.10: significato dello stato del LED *24V*

Questi LED sono presenti in entrambi i *moduli principali PSSu PLC1 H FS SN SD* e *PSSu H FS SN SD*.

DIAG

Il LED *DIAG* indica lo stato dell'intero dispositivo, comunicando l'eventuale presenza di un errore in una parte del sistema PSSu, ed è presente in entrambi i *moduli principali*.

Colore	Stato	Significato
- - -		Nessuna parte del sistema avviata, assenza di "Module Supply".
verde		Tutte le parti del sistema sono in stato RUN.
		La procedura di boot è attiva
rosso		Almeno una parte del sistema è interessata da un messaggio di errore grave (v. elenco di diagnostica).
		Almeno una parte del sistema FS è interessata da un messaggio di errore FS grave (v. elenco di diagnostica).
arancione		Almeno una parte del sistema PSSu è interessata dal messaggio „Attenzione“ (v. elenco di diagnostica).

Figura 5.11: significato dello stato del LED *DIAG*

ST RUN

Il LED *ST RUN* indica lo stato della risorsa ST, è presente solo nel *modulo principale PSSu PLC1 H FS SN SD*.

Colore	Stato	Significato
- - -		Risorsa ST non avviata
verde		Stato operativo „Risorsa ST avviata correttamente e in stato RUN“: Le funzioni della risorsa ST vengono eseguite correttamente.
		Stato operativo „Risorsa ST con errore funzionale in stato RUN“: Almeno una funzione della risorsa ST non viene eseguita.

Figura 5.12: significato dello stato del LED *ST RUN*

FS RUN

Il LED *FS RUN* indica lo stato della risorsa FS, è presente solo nel *modulo principale PSSu PLC1 H FS SN SD*.

Colore	Stato	Significato
- - -		Risorsa FS non avviata
verde		Stato operativo „Risorsa FS avviata correttamente e in stato RUN“: Le funzioni della risorsa FS vengono eseguite correttamente.
		Stato operativo „Risorsa FS con errore funzionale in stato RUN“: Almeno una funzione della risorsa FS non viene eseguita.

Figura 5.13: significato dello stato del LED *FS RUN*

SD CARD

Il LED *SD CARD* indica lo stato del supporto di memoria esterno ed è presente in entrambi i *moduli principali*.

Colore	Stato	Significato
- - -		Assenza tensione di alimentazione per "Module Supply"
rosso		Nessuna scheda SD oppure Scheda SD non riconosciuta oppure Scheda SD guasta
verde		I dati di indirizzamento e il progetto del dispositivo sul sistema PSSu e quelli sulla scheda SD coincidono
		Il tipo di dispositivo sulla scheda SD non coincide con il modulo principale oppure Nessun progetto del dispositivo sulla scheda SD
verde-rosso		I dati di indirizzamento e il progetto del dispositivo sul sistema PSSu e quelli sulla scheda SD non coincidono
arancione		Identificazione del dispositivo da parte dell'utente attiva

Figura 5.14: significato dello stato del LED *SD CARD*

ST FORCE

Il LED *ST FORCE* indica lo stato di forzatura delle variabili ST, presente solo nel *modulo principale PSSu PLC1 H FS SN SD*.

Colore	Stato	Significato
- - -		La forzatura della risorsa ST non è attiva
giallo		La forzatura della risorsa ST è attiva

Figura 5.15: significato dello stato del LED *ST FORCE*

FS FORCE

Il LED *FS FORCE* indica lo stato di forzatura delle variabili FS, presente solo nel *modulo principale PSSu PLC1 H FS SN SD*.

Colore	Stato	Significato
- - -		La forzatura della risorsa FS non è attiva
giallo		La forzatura della risorsa FS è attiva

Figura 5.16: significato dello stato del LED *FS FORCE*

X3:LNK, X3:TRF, X4:LNK e X4:TFR

Questi LED indicano lo stato per entrambe le interfacce SafetyNET p,*X3* e *X4*. Per ciascuna sono correlati due LED, ossia *LNK* che indica la presenza del collegamento e *TRF* per lo scambio dati in corso. Come i precedenti, sono presenti in entrambi i *moduli principali*.

Colore	Stato	Significato
---		Collegamento di rete assente
verde		Collegamento di rete corretto

Figura 5.17: significato dello stato del LED *X3:LNK* e *X4:LNK*

Colore	Stato	Significato
---		Nessuno scambio dati in corso
giallo		Scambio di dati corretto

Figura 5.18: significato dello stato del LED *X3:TRF* e *X4:TRF*

5.1.2 Moduli I/O

Ad ognuno dei dispositivi *PSSuniversal PLC* e *PSSuniversal I/O* possono essere collegati, a destra, *moduli I/O* per specifiche applicazioni ST e FS. Questi permettono un'ampia scalabilità, essendo affiancabili in sequenza casuale, siano essi *standard* o *safe*, senza restrizioni. Per uno schema di collegamento generale risulta, però, più sensato ordinare i *moduli I/O* in gruppi dello stesso tipo. Il numero massimo di moduli utilizzabili viene determinato dai limiti del sistema, riportati in Tabella 5.1.

Ogni *modulo I/O*, come si può vedere in Figura 5.19, è composto da morsettiere finalizzate a creare collegamenti verso il campo, utilizzando conduttori non schermati. Queste morsettiere sono disposte secondo una griglia costituita da *colonne di collegamento* e *linee di collegamento*. I collegamenti sono contrassegnati da numeri a due cifre; la prima cifra indica la *colonna di collegamento* mentre la seconda indica il *livello di collegamento*. Ad esempio, il collegamento 23 si trova nella seconda colonna, al terzo livello.

La funzione dei collegamenti dipende dal tipo di *modulo I/O*. I livelli di collegamento sono tipicamente posti nel seguente ordine: i livelli di collegamento 1 e 4 sono quelli relativi agli ingressi o alle uscite, a seconda del tipo di modulo, mentre i livelli di collegamento 2 e 3 sono utilizzati per funzioni di *Periphery Supply*, quando supportate dal modulo.

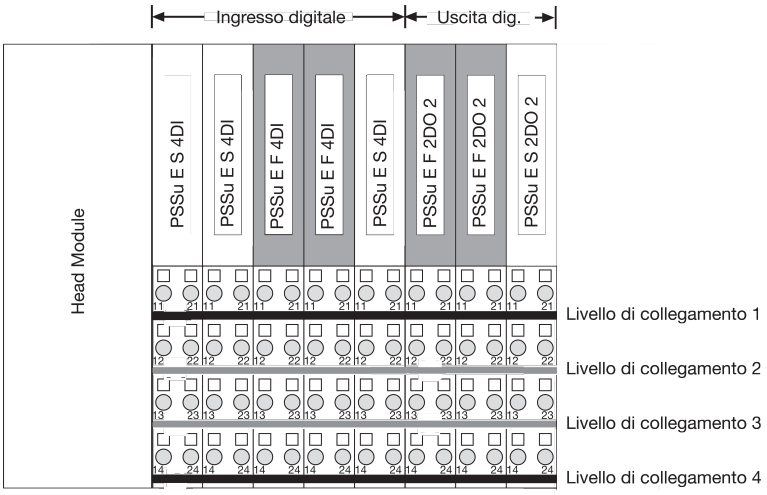
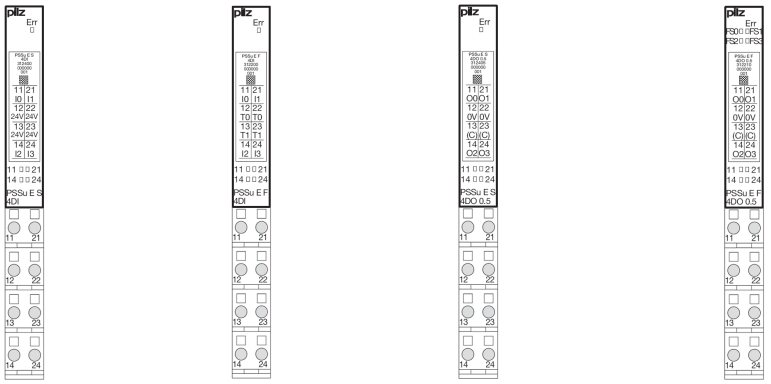


Figura 5.19: morsettiere dei *moduli I/O*

Tra la vasta gamma di *moduli I/O* disponibili per la gestione di segnali di controllo, nell'ambito del lavoro di analisi svolto in questa tesi, sono stati utilizzati unicamente moduli relativi al trattamento di ingressi ed uscite digitali nelle due versioni, *standard* e *safe*, che nello specifico sono: **PSSu E S 4DI**, **PSSu E F 4DI**, **PSSu E S 4DO 0.5** e **PSSu E F 4DO 0.5**.



(a) PSSu E S 4DI (b) PSSu E F 4DI (c) PSSu E S 4DO 0.5 (d) PSSu E F 4DO 0.5

Figura 5.20: *moduli I/O* usati in questa tesi

I moduli **PSSu E S 4DI** e **PSSu E F 4DI**, riportati in Figura 5.20(a) e 5.20(b), sono utilizzati per la gestione di ingressi digitali (*Digital Input - DI*), rispettivamente *standard* (S) e *failsafe* (F). Ciascun modulo presenta 5 LED, di cui 4 sono riferiti agli ingressi e servono per indicare lo stato di commutazione di ogni ingresso e 1, contrassegnato come *Err*, serve per indicare l'eventuale guasto del modulo.

Lo stato degli ingressi viene comunicato al *modulo principale* tramite l'MBUS. Il riconoscimento degli ingressi viene effettuato per tutti quei segnali che persistono per un tempo superiore alla somma del tempo di elaborazione massimo del modulo e del tempo di ciclo dell'MBUS. Vengono, invece, ignorati se persistono per un tempo inferiore al minimo tempo di elaborazione del modulo. Questo perchè, come si vedrà nella Sezione 5.2, i PLC della piattaforma PSS4000 durante il funzionamento ciclico operano su immagini degli ingressi e delle uscite. Ad ogni ciclo di processo viene creata inizialmente l'immagine degli ingressi da cui, durante l'esecuzione del programma, vengono letti i valori di ingresso, mentre i valori di uscita elaborati vengono scritti nell'immagine delle uscite. I valori di uscita vengono poi aggiornati, alla fine di ogni ciclo, in base ai dati contenuti nella relativa immagine.

Per quanto riguarda il modulo **PSSu E F 4DI**, ai *livelli di collegamento* 2 e 3 sono presenti 4 uscite denominate *trigger di test*. Le uscite *trigger di test* forniscono una tensione continua a 24V a cui è sovrapposta un'onda quadra che può essere di tipo T0 o T1. Nel modulo le uscite del *livello di collegamento* 2 sono relative al *trigger di test* T0 mentre quelle del 3 sono relative al *trigger di test* T1.

Ciascun ingresso può essere riferito, tramite il programma software PAS4000, a tensioni costanti oppure ai due tipi di *trigger di test* T0 o T1. Questa funzionalità è necessaria per l'utilizzo di dispositivi di sicurezza classici, come pulsanti di emergenza (*emergency stop*) e barriere fotoelettriche, in cui bisogna poterne

discriminare lo stato di funzionamento. Normalmente questi dispositivi operano secondo il principio della logica negata, per cui al valore logico basso è associato l'azionamento di sicurezza, e vengono connessi tra un'uscita *trigger di test* ed un ingresso del medesimo *modulo I/O*. Nel caso in cui l'ingresso digitale associato al dispositivo di sicurezza venga cortocircuitato a 24 volt, il modulo rileva la mancanza del trigger di test e può segnalare lo stato di malfunzionamento del dispositivo di sicurezza. L'informazione sullo stato del dispositivo di sicurezza viene, generalmente, attribuita ad una variabile detta *valid bit* che riporta lo stato di validità dei dati ricevuti da tale dispositivo. Nel caso in cui questo bit segnali un malfunzionamento, dato che non è possibile discriminare l'abilitazione del dispositivo dal malfunzionamento, il sistema viene portato nello stato *safe* per garantire l'incolumità del personale, dei macchinari e dell'ambiente.

Nel modulo **PSSu E S 4DI**, invece, ai *livelli di collegamento* 2 e 3 sono associate 4 uscite con tensione di alimentazione a 24V, fornita dal *Periphery Supply*, senza la sovrapposizione con alcun tipo di segnale modulante.

I moduli **PSSu E S 4DO 0.5** e **PSSu E F 4DO 0.5**, riportati in Figura 5.20(c) e 5.20(d), sono utilizzati per la gestione di uscite digitali (*Digital Output - DO*) *standard* (S) e *failsafe* (F), con corrente di carico per uscita di 0.5A.

Il modulo **PSSu E S 4DO 0.5** presenta 5 LED, di cui 4 sono riferiti alle uscite e servono per indicare lo stato di commutazione di ogni uscita e 1, contrassegnato come *Err*, serve per indicare l'eventuale guasto del modulo. Il modulo **PSSu E F 4DO 0.5**, invece, presenta un totale di 9 LED, in cui i 4 aggiuntivi, denominati *FS0*, *FS1*, *FS2*, *FS3*, indicano l'abilitazione dello stato *failsafe* di ciascuna uscita. In più, rispetto al modulo **PSSu E S 4DO 0.5**, a ciascun uscita del modulo **PSSu E F 4DO 0.5** è associata una *funzione di test*. Tramite tale funzione le uscite attive vengono verificate mediante regolari test di disattivazione, mentre quelle disattive vengono attivate mediante regolari test di attivazione. Questa *funzione di test* non modifica lo stato del carico (cioè non può essere attivato o disattivato) e può essere disattivata per tutte le uscite per cui l'impianto reagisce in modo sensibile all'impulso di test.

Il valore delle uscite viene comunicato dal *modulo principale* al modulo di uscita tramite l'MBUS. Per entrambi i moduli di uscita, ai *livelli di collegamento* 2 e 3 sono presenti 4 uscite con tensione di alimentazione a 0V che, come per le uscite del modulo ai livelli 1 e 4, vengono forniti dal *Periphery Supply*.

5.2 Piattaforma software: *PAS4000*

Per realizzare l'automazione di un impianto con uno o più sistemi di controllo PSS4000, è necessario creare un progetto tramite la suite software PAS4000.

Figura 5.21: logo della suite software PAS4000

Un progetto è costituito dal *programma utente* e dalla *configurazione hardware*. La *configurazione hardware* specifica quali dispositivi sono utilizzati nel progetto e le loro impostazioni. Il *programma utente* è composto da *Program Organisation Units* (*POU*), disponibili in tre differenti tipologie:

1. *Programmi* (*PRG*)

   Un progetto può contenere uno o più programmi. Un programma normalmente realizza il controllo di una parte dell'impianto. I singoli compiti del sistema di controllo possono essere effettuati da blocchi funzionali (FB) o da funzioni (FUN), richiamati all'interno del programma.

2. *Blocchi funzionali* (*FB*)

   I blocchi funzionali assumono specifici compiti individuali all'interno del programma utente. Un blocco funzionale non può essere eseguito in modo indipendente ma deve essere richiamato direttamente da un programma o da un altro blocco funzionale; in esso, però, all'inizio della gerarchia di chiamata deve essere presente un programma.
   I blocchi funzionali, in genere, hanno parametri di input e parametri di output; è peraltro possibile che un blocco funzionale abbia solo delle variabili locali. Tramite i parametri di input, vengono trasmessi i valori che influenzano l'elaborazione logica, mentre il blocco funzionale, dopo lo svolgimento della sua attività, scrive i valori elaborati nei parametri di output.
   I blocchi funzionali hanno a disposizione una porzione di memoria in cui i parametri di input, di output e le variabili locali vengono salvate nel momento della chiamata del blocco funzionale.
   E' possibile scrivere blocchi funzionali personalizzati e anche utilizzarne di standard, in dotazione con PAS4000. Pilz GmbH & Co.KG, infatti, offre blocchi funzionali, soggetti a una licenza, già funzionanti per una vasta gamma di applicazioni.
   Un blocco funzionale è disponibile per qualsiasi POU di un progetto e può essere richiamato un numero illimitato di volte.

3. *Funzioni* (*FUN*)

   Come i blocchi funzionali, le funzioni possono svolgere compiti individuali all'interno di un programma.
   Una funzione non può essere eseguita in modo indipendente, ma deve essere richiamata all'interno di un programma o di un blocco funzionale. Quando si richiama una funzione, possono essere trasmessi dei parametri di input che influenzano l'attività di elaborazione. Il risultato di tale elaborazione è definito *function value* è può essere un valore individuale, un array o una struttura e deve essere specificato in fase di creazione della funzione. Oltre al *function value*, una funzione può avere diversi parametri di output che possono essere ulteriormente elaborati dalla POU chiamante. Tutte le variabili di una funzione sono riportate al loro valore iniziale quando viene conclusa

la chiamata.
A differenza dei blocchi funzionali, le funzioni non richiedono un'area separata di memoria su una risorsa e, quindi, non sono in grado di memorizzare tutte le informazioni. L'ambito di una funzione è globale, cioè è disponibile per tutte le POU di un progetto e può essere richiamata un numero di volte illimitato.
Anche per questa tipologia di POU, come per i blocchi funzionali, è possibile scrivere delle funzioni personalizzate oppure utilizzare quelle standard fornite dal PAS4000.

Per prima cosa in un progetto vengono create le varie POU, che possono essere riutilizzate in qualsiasi progetto futuro. Successivamente queste POU vengono assegnate alle risorse dei dispositivi disponibili e, infine, si provvede all'assegnazione delle variabili simboliche agli ingressi e alle uscite del sistema fisico.

Ogni POU è costituita da una *parte dichiarativa* e da una *parte di istruzioni*. Le dichiarazioni delle variabili vengono eseguite nella *parte dichiarativa*. Nella *parte di istruzioni* viene visualizzato il testo della programmazione effettuata, indipendentemente dal linguaggio di programmazione utilizzato.

I *PSSuniversal PLC* sono compatibili con tutti i linguaggi di programmazione aderenti allo standard IEC 61131-3, sia per gli aspetti standard che di sicurezza. La gestione comune della programmazione delle funzioni *standard* e *safe* assicura il massimo livello di flessibilità.

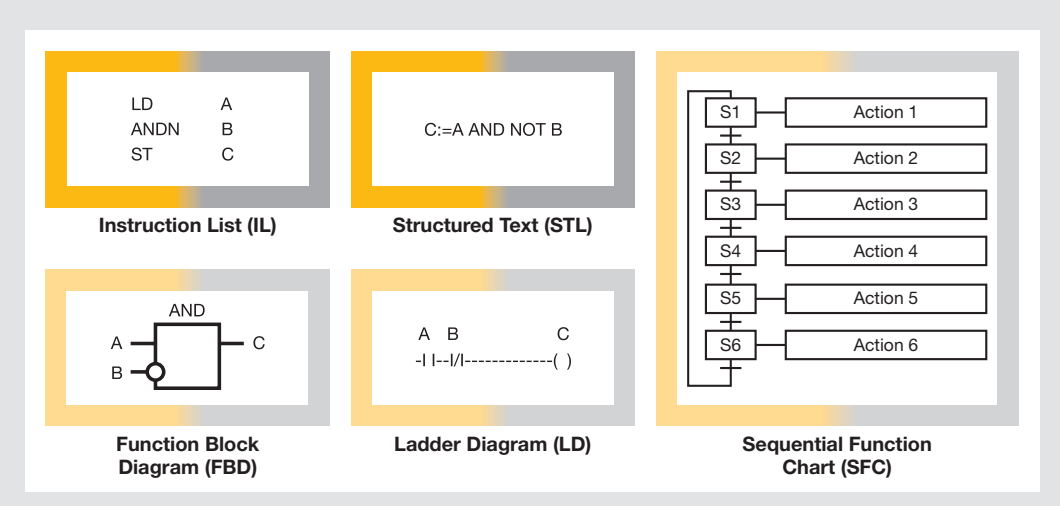


Figura 5.22: linguaggi di programmazione aderenti allo standard IEC 61131-3

Tra i vari linguaggi disponibili, per la programmazione dei dispositivi è stato utilizzato l'*Instruction List* (*IL*), un linguaggio di basso livello molto simile all'*assembly*. Un programma IL si presenta come una lista di istruzioni scritte in sequenza. Nella versione definita dallo standard IEC 61131, l'esecuzione del programma è lineare. Ogni istruzione è racchiusa all'interno di una riga e la prima istruzione ad essere eseguita è la prima che compare, seguendo l'ordine dall'alto verso il basso.

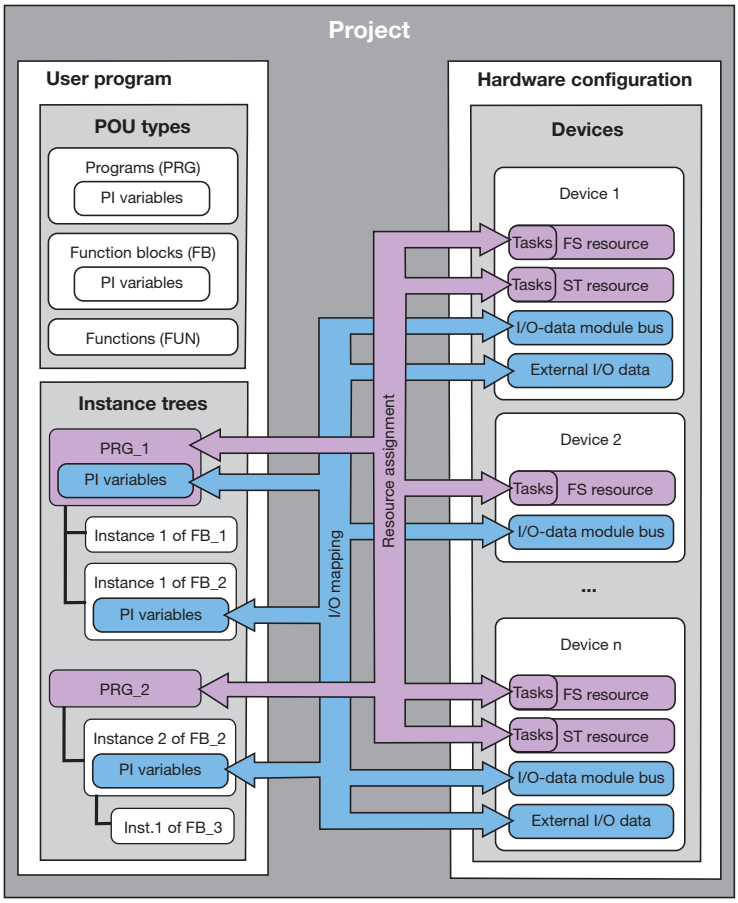


Figura 5.23: esempio di assegnazione delle POU ai *task*

Una o più POU, della tipologia programma, possono essere inserite in un *task*. L'ordine in cui vengono eseguite queste POU può essere influenzato dall'utente tramite la configurazione dei *task*; questo, però, non è necessario per la maggior parte dei progetti perchè la configurazione predefinita è sufficiente per la gestione dei *task*. Durante l'operazione di assegnazione questi *task* sono contenuti in una zona di memoria dei dispositivi utilizzati, detta *risorsa*. Ogni dispositivo appartenente al sistema PSS4000 può avere al massimo una risorsa *FailSafe* (*FS*) e una risorsa *STandard* (*ST*).
Nell'ambito della suite software PAS4000, vengono distinti 3 tipi di *task*:

1. *Task periodico*

   il task viene eseguito una volta sola in un tempo prestabilito, definito *tempo di ciclo del task*. Gli intervalli tra le esecuzioni possono variare ma se non vengono completate entro il tempo di ciclo del *task*, lo stato del dispositivo viene commutato nella condizione di funzionamento "*Resource with task fault in RUN*". I tempi ciclo dei *task*, sia per quelli FS che ST, sono compresi nel range $2ms \div 71min$;

2. *Event-driven task*

   il task è eseguito al verificarsi di un evento definito a priori;

3. *Freewheeling task*

   il task è eseguito nuovamente, appena la sua precedente esecuzione è completata.

Per ogni *task* possono essere definite delle priorità all'interno della risorsa di un dispositivo, che ne determinano l'importanza dell'esecuzione rispetto agli altri. Tali priorità determinano l'ordine in cui i *task* di una risorsa vengono eseguiti e possono essere: alta, media e bassa. Ciascuna risorsa può contenere un massimo di 9 *task*, 3 per ogni livello di priorità. Se un numero elevato di compiti sono programmati con priorità alta, questo può rallentare l'esecuzione di alcune funzioni *on-line* di PAS4000, come la visualizzazione dinamica del programma e dello stato delle variabili.
La configurazione predefinita di esecuzione dei *task* è progettata in modo tale che:

- i compiti sono eseguiti in base alla loro priorità. I compiti con la massima priorità sono eseguiti per primi, di seguito vengono considerati quelli con la priorità media e, infine, i compiti a bassa priorità. L'esecuzione dei *task* è interrotta se sono in attesa di esecuzione *task* di priorità superiore;

- per i *task periodici* il tempo di ciclo è usato come priorità secondaria: nei casi in cui i *task periodici* abbiano tutti la stessa priorità, viene eseguito per primo il *task* con minor tempo ciclo. Se ci sono i *task* di uguale priorità e tempo ciclo, l'ordine di esecuzione è casuale;

- se un dispositivo ha una risorsa FS e ST, i *task* in esse contenuti sono eseguiti dalla stessa CPU. I *task* FS con una determinata priorità vengono sempre eseguiti prima dei *task* ST con la stessa priorità;

- se più programmi POU vengono assegnati ad uno stesso *task*, i programmi vengono eseguiti, in ordine alfabetico, in base al nome.

All'inizio dell'esecuzione di un *task* viene creata l'immagine degli ingressi e delle uscite del processo, *PII* (*Process Image Input*) e *PIO* (*Process Image Output*). Successivamente, il programma viene eseguito e i dati vengono rispettivamente importati e scritti su tali immagini. Nel caso in cui vi siano più programmi nello stesso *task*, questo processo viene ripetuto per ciascuno di essi. Una volta che anche l'ultimo programma del *task* è terminato, vengono aggiornati i valori delle variabili conformemente con quelli presenti nell'immagine delle uscite di processo (PIO). La creazione di queste immagini garantisce che i dati non possono essere modificati durante l'intera esecuzione del *task*.

Normalmente tutte le variabili di I/O di un sistema di automazione hanno una diretta correlazione con l'hardware della logica programmabile. Con PAS4000 è possibile scrivere e richiamare funzioni di controllo, sia standard che di sicurezza, usando variabili simboliche. Il tipo di dato deve essere definito per ogni variabile e ne determina, ad esempio, il campo dei valori ammissibili, l'area di applicazione (FS e/o ST) e la quantità di spazio di memoria necessario per salvare la variabile.

La differenza principale fra le tipologie di dati che viene effettuata nella suite software PAS4000 è la seguente:

- *Elementary data type*

  queste tipologie di dati sono utilizzate per la dichiarazione di variabili. Per ogni variabile, di qualsiasi tipo di dato, è necessario un valore iniziale; se non viene assegnato manualmente, in fase dichiarativa, assumerà un valore predefinito di default. Tutti i tipi di dati elementari possono essere utilizzati sia su risorse FS che ST;

- *Safe data type*

  questi tipi di dati sono stati definiti dalla Pilz GmbH & Co.KG per le applicazioni legate alla *safety*. Le loro proprietà corrispondono a quelle dei rispettivi *Elementary data type*. La differenza è che i dati dichiarati di tipo *safe* in una POU, possono essere utilizzati soltanto da risorse FS.

L'uso abituale di queste due tipologie di dati, all'interno delle POU, permette di mantenere una netta separazione tra risorse FS e ST: i compiti non attinenti alla sicurezza possono essere programmati in *POU ST*, che possono essere eseguite sia da risorse ST che FS, mentre quelli connessi alla sicurezza dovono essere programmati in *POU FS*, che possono essere eseguite da risorse FS. Una POU viene definita *POU FS* qualora una sua variabile sia dichiarata come *Safe data type*. Se vi è la possibilità di accedere a variabili *Elementary data type*, allora viene definita *POU mista* o *FS+ST POU*.

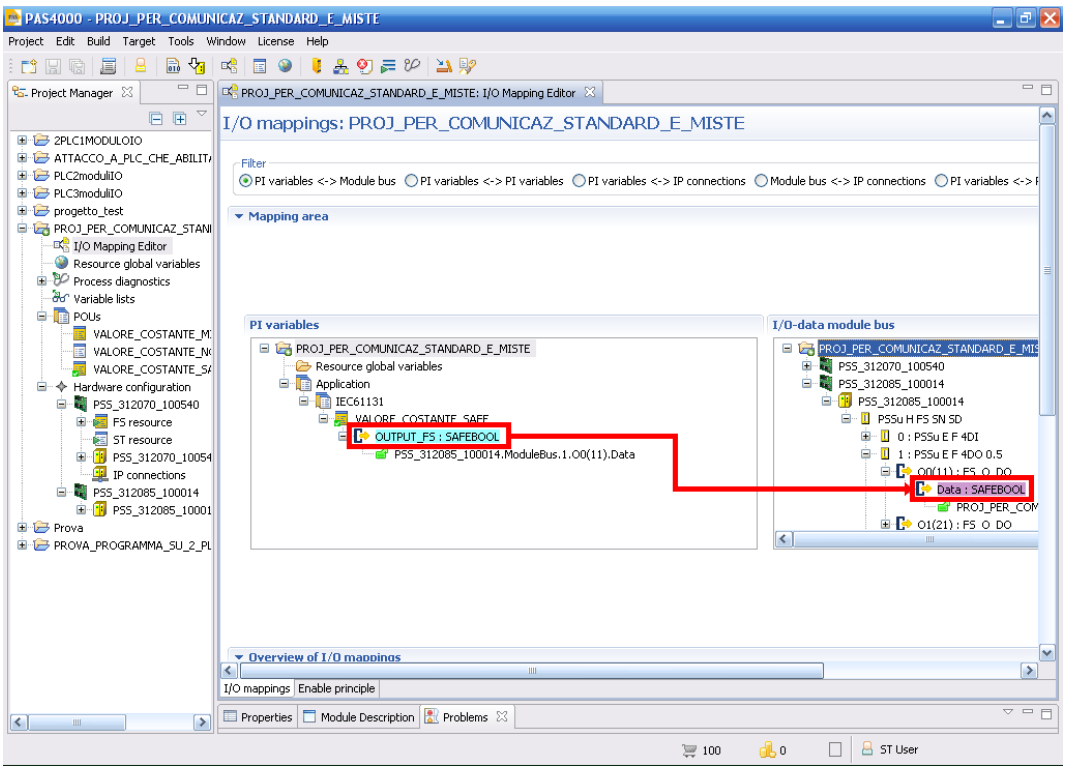


Figura 5.24: *screenshot* del *mapping* delle variabili in PAS4000

Una volta terminato il progetto software è necessario associare le variabili di I/O con l'hardware del sistema di controllo, scelto a posteriori. Questa procedura è detta *mapping* delle variabili o più semplicemente *I/O mapping*.

In Figura 5.24 è riportato uno *screenshot* del programma PAS4000 che evidenzia la finestra relativa all'associazione delle variabili simboliche con le uscite e gli ingressi dei dispositivi hardware utilizzati per l'esecuzione del programma di automazione realizzato. In questo caso specifico è stata generata solo una variabile simbolica *safe*, che si può vedere dentro al rettangolo rosso sul lato sinistro della schermata. Effettuare il *mapping* della variabile è semplice: si seleziona la variabile simbolica e la si trascina sull'uscita o sull'ingresso fisico, sulla parte destra della schermata, a cui si vuole associare. Normalmente ogni variabile viene mappata su una risorsa hardware della stessa tipologia, anche se è possibile associare una variabile ST ad un I/O FS. In questo caso, però, essa non sarà trattata e non verrà utilizzata per svolgere compiti FS.

Una volta effettuato il *mapping* di tutte le variabili I/O, il programma necessita di un'operazione di *build* che, oltre a creare il codice del programma eseguibile dai dispositivi, verifica la presenza di eventuali errori delle impostazioni.

Infine, rimane da trasferire il programma ai dispositivi e ciò si effettua tramite l'operazione detta di *download*. Ogni informazione relativa alle variabili di I/O è inclusa in una *tabella di mappatura* centrale, che è comune e utilizzabile sia per i processi *standard* sia per quelli *safe*. La Figura 5.25 illustra il principio di funzionamento della tabella di mappatura centrale in relazione a un'applicazione con tre sistemi.

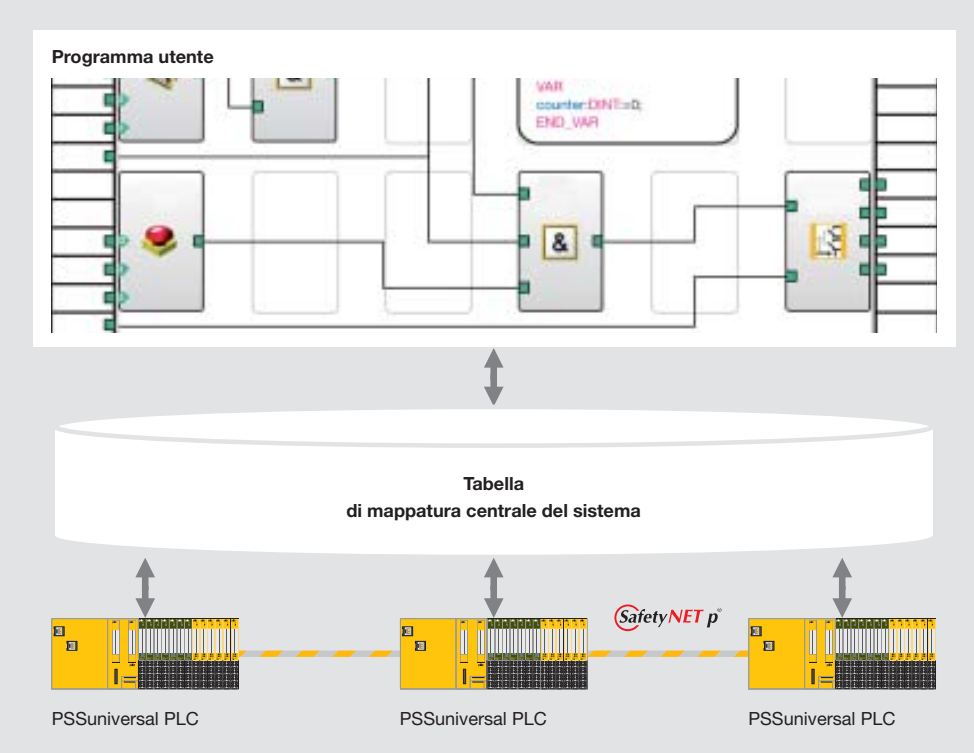


Figura 5.25: tabella di mappatura centrale del sistema.

5.3 *Managed Switch* con *Port mirroring*

Gli *switch* sono la versione più evoluta dei classici *hub*, che permettono a più dispositivi di essere interconnessi per formare una rete complessa.

Nella tecnologia delle reti informatiche, un *hub* rappresenta un dispositivo di rete che funge da nodo di smistamento di una rete di comunicazione dati organizzata con una topologia a stella. Nel caso delle reti Ethernet, un *hub* è un dispositivo che inoltra i dati in arrivo da una qualsiasi delle sue porte su tutte le altre. Per questo può essere definito anche come "ripetitore multiporta". La conseguenza di questo è che la banda totale disponibile in uscita dall'*hub* viene frazionata e ripartita tra tutti i dispositivi, a causa del moltiplicarsi dei dati da inviare a tutte le sue porte.

Differentemente, uno *switch* inoltra i frame in arrivo da una qualsiasi delle sue porte soltanto a quella cui è collegato il nodo destinatario del messaggio. Uno *switch* possiede, dunque, l'intelligenza necessaria a riconoscere i messaggi provenienti da una sua porta, immagazzinarli, decidere su quale porta inoltrarli e, infine, trasmetterli. Uno *switch* opera a livello *datalink*. Per decidere su quale porta inoltrare un frame ricevuto, uno *switch* deve possedere una funzione di instradamento che permette di associare le porte con gli indirizzi dei dispositivi all'interno di una tabella, detta *forwarding database*. Questa tabella viene aggiornata con cadenza periodica per motivi di scalabilità e flessibilità, dato il possibile verificarsi di variazione del numero o della posizione dei dispositivi connessi.

Quando un nodo A cerca di comunicare con un nodo B, lo *switch* ignora il frame, altrimenti se B è collegato ad una porta diversa, lo *switch* inoltra il frame sulla porta a cui è collegato B. Se, però, lo *switch* non conosce ancora a quale porta è collegato B, inoltra il frame su tutte le porte. Il nodo destinatario riceverà il pacchetto e risponderà al mittente, permettendo allo *switch* di scoprire a quale porta esso è collegato e di aggiornare la tabella di instradamento.

Alcuni messaggi possono avere un indirizzo di destinazione particolare, denominato *broadcast*. Esso indica che tutti i dispositivi della rete sono i destinatari del messaggio e quindi lo *switch* inoltra questi frame su tutte le porte. Analogamente accade per i messaggi con destinazione *multicast*, che sono indirizzati ad un gruppo di destinatari. Il protocollo *IGMP* viene usato da molti dispositivi per richiedere l'iscrizione ad un gruppo *multicast* e lo *switch* inoltra questi messaggi a tutte le porte a cui sono connessi i dispositivi appartenenti a tale gruppo.

La complessità del comportamento di uno *switch* è compensata dalle migliori prestazioni ottenibili. Il fatto che i frame vengano ritrasmessi selettivamente ha anche delle implicazioni di sicurezza informatica in quanto evita che un dispositivo possa facilmente intercettare, in gergo *sniffare*, il traffico instradato da e per un altro terminale. In realtà, ciò è vero solo in prima approssimazione, essendo state sviluppate raffinate tecniche di attacco come lo *Switch Flooding*, il *Port stealing* e l'*ARP Poisoning*, che permettono lo *sniffing* anche in presenza di *switch*.

Dato che gli attuali *switch*, a differenza dei classici *hub*, svolgono un'attività di indirizzamento corretto dei pacchetti e non inviano a tutte le porte i pacchetti, diventa difficile tramite programmi di analisi di rete monitorare il traffico. Per questo, negli *switch* più moderni ed evoluti, detti *managed switch*, è stata inserita la funzione *port mirroring* che consente di configurare lo *switch* per reindirizzare il traffico che attraversa alcune porte, verso una designata porta di controllo. Con

questa funzione lo *switch* invia una copia di tutti i pacchetti di rete ricevuti e inviati su una porta ad un altra prestabilita, permettendo ad un computer di vedere e analizzare, con appositi strumenti, tale traffico di rete, non altrimenti accessibile.

I *managed switch* hanno un'interfaccia di configurazione *web-based* dalla quale è possibile specificare le porte di origine e di destinazione del *mirroring*: tutto il traffico presente sulla porta di origine sarà replicato su quella di destinazione.

In Figura 5.26 è rappresentato lo schema di funzionamento logico del *managed switch* con *port mirroring* che è stato utilizzato per tutte le analisi.

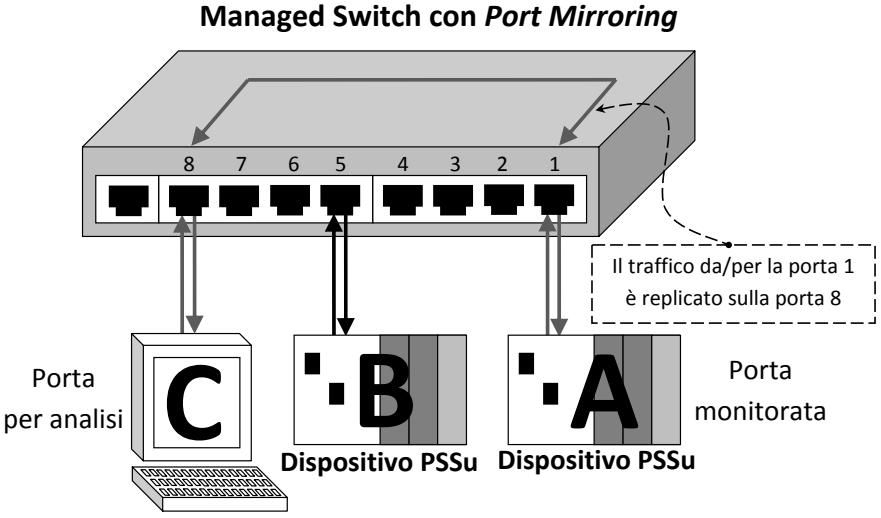


Figura 5.26: principio di funzionamento del *port mirroring*

Nel contesto di questa tesi si vuole analizzare il traffico tra i due dispositivi PSSuniversal A e B, tramite l'ausilio di un *analizzatore di traffico* installato sul PC C. Nello *switch* è stato impostato il *mirroring* tra le porte 1 e 8. In questo modo il traffico dati fra A e B viene indirizzato correttamente, con l'aggiunta che tutto il traffico in uscita e in ingresso alla porta a cui è connesso A viene replicato sulla porta 8, a cui è connesso il PC. Sul PC viene solamente utilizzato lo strumento software che permette di tracciare i messaggi, detto *analizzatore di traffico*, senza alterare in alcun modo la comunicazione tra A e B.

5.4 Analizzatore di traffico: *Wireshark*

Wireshark è un software per l'analisi di protocolli, detto *packet sniffer*, libero e open-source. Originariamente chiamato *Ethereal*, nel maggio del 2006 il progetto fu rinominato Wireshark. Nel 2008, dopo dieci anni di sviluppo, Wireshark è finalmente arrivato alla versione 1.0, la prima ritenuta completa, con le caratteristiche minime di attuazione. Attualmente Wireshark, nelle due versioni che sono state sviluppate parallelamente, è arrivato alla *Stable Release 1.4.6* e alla *Old Stable Release 1.2.16*.

Wireshark, è un software open-source rilasciato sotto la licenza GNU GPL (General Public License). Inoltre, tutto il codice sorgente è liberamente disponibile

sotto licenza GPL. Per questo è molto facile per gli utenti aggiungere nuovi *plug-in*, che permettono di decodificare agevolmente i protocolli di comunicazione.

Figura 5.27: logo dell'analizzatore di traffico *Wireshark*

Un analizzatore di traffico di rete cattura i pacchetti e li visualizza, ordinati temporalmente, dettagliando il più possibile i dati in essi contenuti. Si può pensare ad un analizzatore di pacchetti di rete come strumento di misura utilizzato per esaminare che cosa sta succedendo dentro un cavo di rete, in analogia ma a livello più alto, con un multimetro digitale che permette di avere informazioni delle grandezze in gioco all'interno di un cavo elettrico.

Wireshark può essere utilizzato per risolvere i problemi di rete, per esaminare i problemi di sicurezza e per eseguire il *debug* dell'implementazione dei protocolli di comunicazione. Oltre a questi esempi, Wireshark può essere utile in molte altre situazioni.

Alcune caratteristiche di cui dispone Wireshark sono la compatibilità con UNIX e Windows, l'acquisizione dati dei pacchetti in tempo reale da un'interfaccia di rete, la visualizzazione di informazioni di protocollo molto dettagliate per ogni pacchetto, l'apertura e il salvataggio dei pacchetti *sniffati* nel formato *.pcap*, l'importazione e l'esportazione dei pacchetti da e per programmi di acquisizione, il filtraggio di pacchetti in base a diversi criteri, la ricerca di pacchetti, la colorazione differenziata dei pacchetti di protocolli diversi e la creazione di statistiche varie. Wireshark può catturare in *real time* il traffico proveniente da diversi tipi di rete, comprese le LAN (Local Area Network) wireless, anche se tipicamente è utilizzato per reti Ethernet. Le tipologie di rete supportate per lo *sniffing* dipendono fortemente dal sistema operativo in cui è installato il programma. Una panoramica dei tipi di media supportati sono disponibili all'indirizzo: `http://wiki.wireshark.org/CaptureSetup/NetworkMedia`.

Questo analizzatore di traffico riesce a "comprendere" la struttura di diversi protocolli di rete; infatti, è in grado di individuare eventuali incapsulamenti, riconoscere i singoli campi e interpretarne il significato. Questa capacità è fornita da *dissezionatori* (in inglese *dissectors*), detti anche *plug-in*. Wireshark possiede *dissezionatori* per un gran numero di protocolli ritenuti "standard" ma ulteriori *plug-in* possono essere creati per nuovi protocolli. Ogni protocollo ha un proprio *dissezionatore*, in modo da sezionare completamente e correttamente i messaggi di tale tipologia.

Wireshark può essere utilizzato per analizzare il traffico del protocollo di rete SafetyNET p. In questa tesi, per agevolare l'analisi e la comprensione dei vari messaggi, si è utilizzato il *plug-in* relativo a SafetyNET p, gentilmente fornito da Pilz GmbH & Co.KG. Per installare e per poter utilizzare questo *dissezionatore* è stato necessario installare la versione *Old Stable Release 1.2.13* di Wireshark poichè è l'unica che lo supporta.

In Figura 5.28 è riportato uno *screenshot* di una schermata Wireshark in cui si comprende l'utilità del *plug-in* per SafetyNET p. Nella sezione in basso si vedono

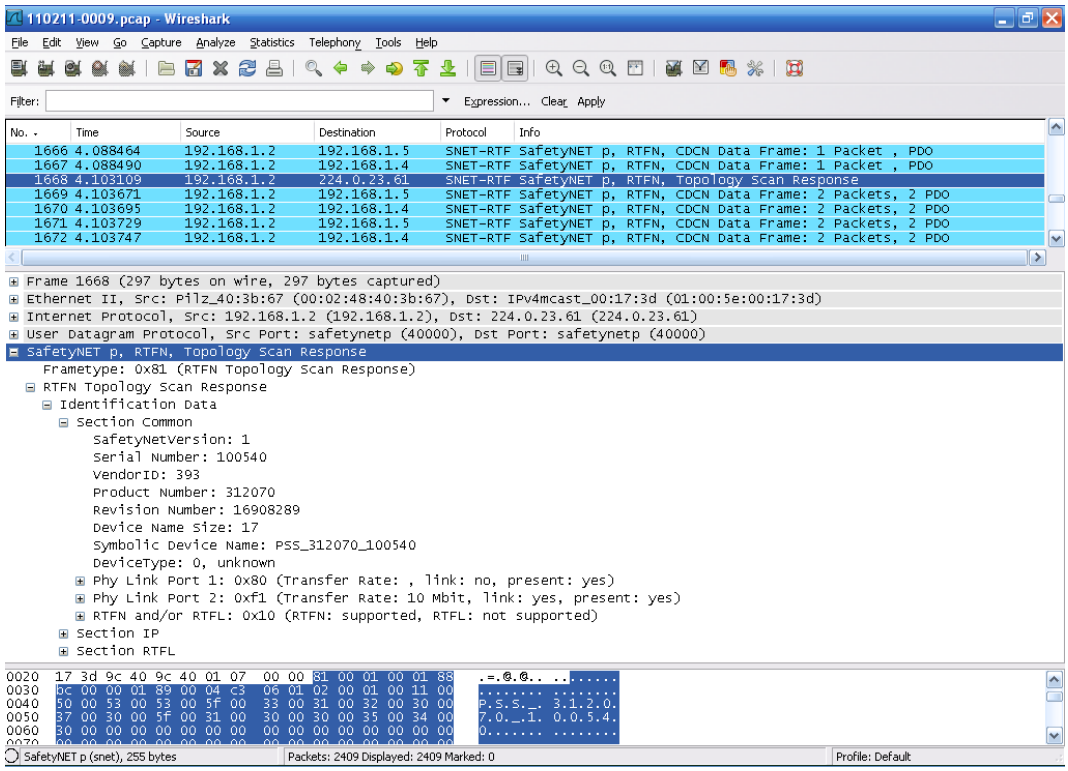


Figura 5.28: *screenshot* di una schermata di Wireshark

gli effettivi byte che compongono il messaggio *RTFNSNR response* (*Topology Scan Response*) e, in quella sopra, è riportata la sua decodifica. Si vede, quindi, che l'analisi dei vari messaggi è molto semplificata dall'ausilio del *dissezionatore*.

Va però segnalato che la versione utilizzata del *plug-in* relativo a SafetyNET p, *SNET Wireshark_ 03_ 12_ 2010.dll*, non era completa e, per questo, non ha agevolato l'analisi del protocollo in quanto non è in grado di riconosce la struttura ridondante dei messaggi *safe*, che interpreta come *standard*. Per tutte le altre tipologie di messaggi, come *MSC PDU*, *RTFNSNR request/response*, *CDCN subscribe request/acknowledge* e *CDCN connection still alive*, la decodifica viene eseguita in maniera corretta, seguendo alla lettera le specifiche del protocollo SafetyNET p.

Vanno, infine, sottolineati alcuni concetti fondamentali sul funzionamento di Wireshark. Wireshark non è un sistema di rilevamento delle intrusioni, infatti non avverte quando avvengono accessi sulla rete non autorizzati. Tuttavia, se accadono strani comportamenti sulla rete, questo programma può aiutare a capire cosa sta realmente accadendo. Inoltre, Wireshark non permette di manipolare i messaggi della rete; infatti non invia i pacchetti sulla rete ma si limita a catturarli.

5.5 Generatore di traffico Ethernet: *packETH*

Il software *packETH* è uno strumento per generare traffico Ethernet. Esso consente di creare e inviare un qualsiasi pacchetto, o sequenza di pacchetti, tramite la scheda di rete Ethernet del PC. Nato in ambiente Linux, oggi con la versione 1.6 è presente anche per Windows.

Con packETH è possibile creare e inviare pacchetti Ethernet con diversi protocolli: Ethernet II, Ethernet 802.3, 802.1Q, ARP, IPv4, IPv6, UDP, TCP, ICMP, IGMP. Qualsiasi altro protocollo deve essere scritto dall'utente come sequenza di byte all'interno del *payload* del frame Ethernet inviato nella rete. In questa maniera i dispositivi che riceveranno tale frame, scompatteranno i vari livelli di incapsulamento e, infine, otterranno i dati inseriti nel *payload*, che sono in grado di decodificare e interpretare.

E', inoltre, possibile impostare la sequenza di invio dei pacchetti, il ritardo tra i pacchetti, il numero di pacchetti da inviare, la velocità di trasmissione e cambiare i parametri per l'invio, come indirizzo IP e MAC. Infatti, il generatore di frame Ethernet permette di inviare in rete esattamente i pacchetti desiderati, specificandone gli indirizzi IP e MAC del destinatario e del mittente. In questo modo possono essere simulati, tramite PC, i messaggi scambiati fra due dispositivi, senza inserire nel frame Ethernet clonato gli indirizzi MAC e IP della scheda di rete da cui è stato generato.

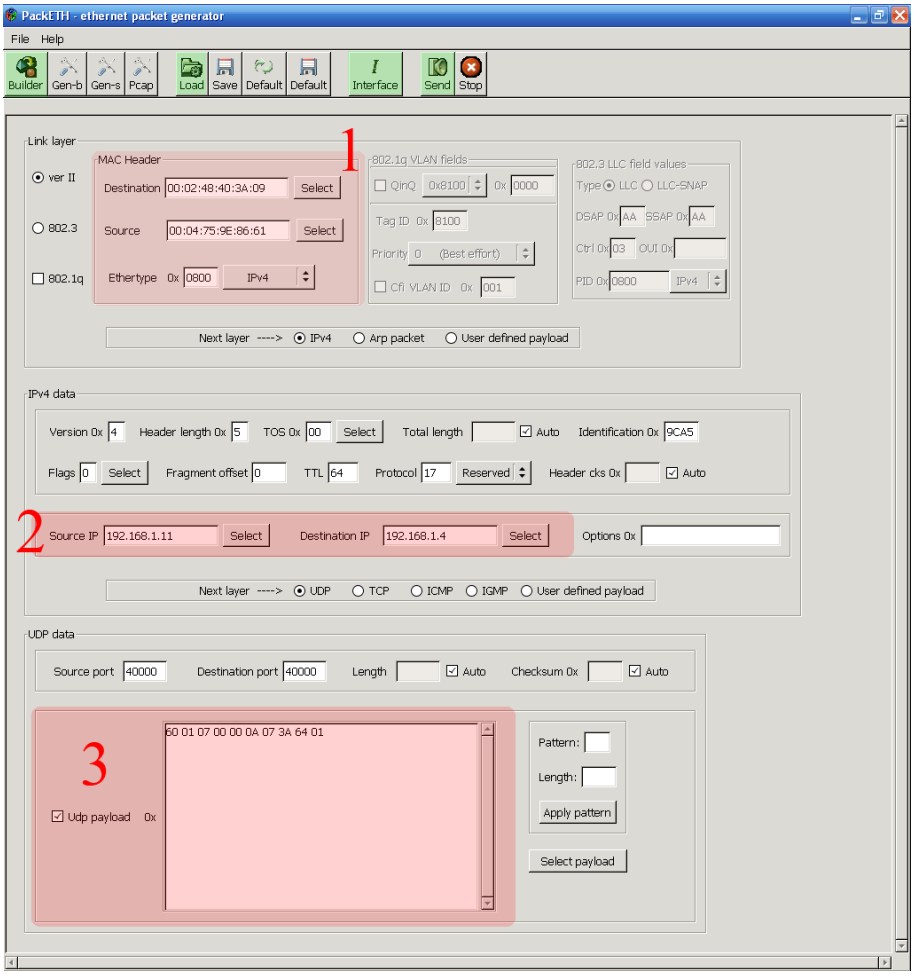


Figura 5.29: *screenshot* di una schermata di packETH

In packETH sono supportati i file *.pcap* ed è, quindi, possibile caricare frame da specifici file di salvataggio Wireshark, modificarne il contenuto e inviarli nella rete. Questo è l'uso che è stato fatto di packETH all'interno di questa tesi. Facen-

do riferimento alla Figura 5.29 vediamo come è stata operata tale procedura con packETH.

Per creare il frame da inviare è necessario premere il pulsante *builder* nella barra degli strumenti, che permette di comporre in tutte le sue parti il messaggio da trasmettere. I campi fondamentali, da settare ogni volta in base alle specifiche necessità, sono contrassegnati con dei rettangoli rossi in Figura 5.29 . Si modificano solo questi campi perchè per creare un frame da iniettare nella rete, si è sempre partiti da un frame *sniffato* nella stessa e riproposto con le opportune modifiche in uscita.

Il riquadro rosso contrassegnato con il numero 1 è la parte relativa al *MAC header* ed è la sezione che permette di impostare gli indirizzi MAC rispettivamente del mittente e del destinatario del frame. Il riquadro 2 nella sezione *IPv4 data*, invece, serve per impostare gli indirizzi IP del destinatario e del sorgente. Infine nel riquadro 3 della sezione *UDP data*, dove sono impostate le porte di destinazione e sorgente come 40000[3], è possibile specificare il *payload* che si vuole trasmettere dentro al frame UDP. Dato che packETH non è preconfigurato con apposite interfacce grafiche per trasmettere dati SafetyNET p, si è dovuto, ogni volta che si voleva mandare un messaggio SafetyNET p, inserire direttamente i byte "grezzi" all'interno del *payload UDP*.

Una volta configurato, in tutte le sue parti, il frame che si desidera inviare è necessario, tramite il pulsante *interface* della barra degli strumenti, selezionare la scheda di rete Ethernet del PC da cui si vuole inviare il messaggio. Poi, per inviare una sola copia del messaggio è sufficiente cliccare sul pulsante *send*. Se si vuole trasmettere più di una copia del messaggio settato tramite la schermata *builder* si utilizza il pulsante *Gen-b*, che apre una finestra in cui è possibile leggere la struttura in byte del messaggio e impostare il numero di frame da inviare, oltre che il ritardo in μs fra un invio e l'altro. Da alcuni test svolti utilizzando il PC a disposizione, si è verificato che il minimo intervallo di tempo raggiungibile tra l'invio di due frame successivi è pari a 1ms.

[3] 40000 non è altro che la traduzione in decimale del *Ethertype* esadecimale di SafetyNET p, 0x9C40

CAPITOLO

6

Analisi del protocollo

In questo capitolo sono presentate le differenti tipologie di analisi sperimentali effettuate sul protocollo di comunicazione SafetyNET p, implementato dai dispositivi PSSuniversal appartenenti al sistema PSS4000, con gli strumenti a disposizione presentati nel precedente capitolo. Queste analisi servono a costituire una conoscenza dettagliata del protocollo utile a fornire informazioni sulle strategie da adottare per simulare degli errori di comunicazione, che saranno oggetto del prossimo capitolo.

Le analisi del protocollo svolte in laboratorio possono essere distinte in tre grandi categorie: *comunicazioni semplici*, che permettono di effettuare un primo approccio con il protocollo, capendo come viene creata e gestita la comunicazione da parte dei dispositivi, *comunicazioni complesse*, che permettono di far luce su questioni specifiche come l'assegnazione univoca o meno dei PID, e *applicazioni reali*, in cui si realizza l'utilizzo di un dispositivo di sicurezza classica (nello specifico un *pulsante di emergenza*) mediante i *PSSuniversal* (*PSSu*) e la rete SafetyNET p.

Tutte le eventuali differenze temporali riportate nei vari schemi riassuntivi della comunicazione analizzata verranno, per semplicità e immediatezza nella comprensione, approssimate al valore intero più vicino. Ad esempio, nel caso di quantità temporali dell'ordine dei *ms* verranno tralasciate le cifre decimali, troncando o arrotondando al valore intero più vicino. Questa scelta è stata effettuata per due motivi: l'analizzatore di rete Wireshark non è uno strumento preciso nel rilevare le tempistiche, dato che nella misura influisce anche il tempo di processo dei dati acquisiti mediante la scheda di rete; inoltre, nell'ambito di questa tesi, è fondamentale la successione in cui sono trasmessi i diversi frame e la periodicità dal punto di vista qualitativo, piuttosto che quantitativo, di inoltro delle varie tipologie di messaggi.

Va, infine, segnalato che durante le sniffate delle comunicazioni di ciascuna categoria è stata riscontrata la trasmissione di molti messaggi aciclici sul *MeSsage Channel* (*MSC*), oltre ai dati di processo sul *Cylic Data Channel* (*CDC*), tra i dispositivi e il PC. Grazie al dissezionatore è stato possibile dare una forma concreta alla struttura degli *MSC service data* implementati da Pilz come *Pilz service message*, rappresentata in Figura 4.7 di pag.42. Come già accennato in quel contesto, l'MSC non influenza i meccanismi che si vogliono analizzare in questa tesi, tutti implementati a livello CDC, e per questo viene tralasciata un'analisi dettagliata di questa tipologia di frame. Di questi messaggi, per l'obiettivo perseguito, è sufficiente sapere che vengono trasmessi tra il PC e i dispositivi in fase di *download* del programma per impostare i nodi della rete e, durante la comunicazione, per lo scambio di informazioni di stato o di diagnostica. Esse avvengono indipendentemente dal fatto che il PC sia connesso alla rete; anche se scollegato, i dispositivi continuano a mandare i messaggi MSC per sottoporre le informazioni che li riguardano.

6.1 Comunicazioni semplici

L'obiettivo che si vuole raggiungere con l'analisi di queste comunicazioni è ottenere delle informazioni dettagliate sull'implementazione del protocollo SafetyNET p realizzata da Pilz. In particolare, si vuole capire la composizione della trasmissione, intesa come creazione e svolgimento della comunicazione, fra i dispositivi durante l'esecuzione di semplici funzionalità. L'analisi verrà effettuata per le due parti in cui si suddivide la comunicazione: *creazione della comunicazione*, in cui i dispositivi si inviano i messaggi preliminari che servono per abilitare la comunicazione, e lo *svolgimento della comunicazione*, in cui avviene la trasmissione dei dati di processo.

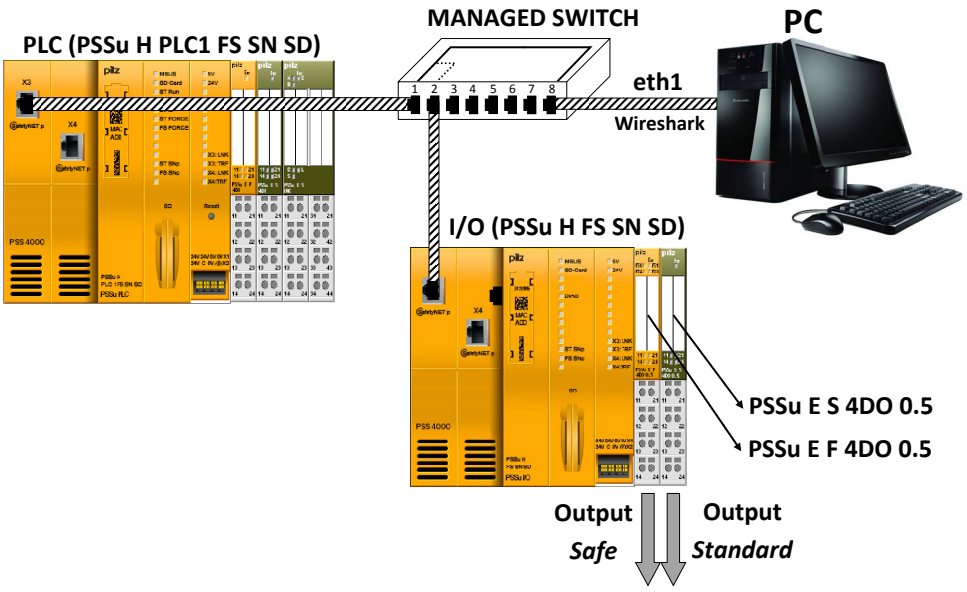


Figura 6.1: struttura del *set up* per le analisi delle *comunicazioni semplici*

La struttura della rete che si andrà ad analizzare è riportata in Figura 6.1 e sarà quella di riferimento durante l'esecuzione delle analisi di tutti i casi in esame in questa categoria. Nello specifico, sul *PSSu H PLC1 FS SN SD* (*PSSu PLC*) è installato un semplice programma che genera un'onda quadra. In base alla tipologia della comunicazione che si va ad analizzare, tale segnale è associato ad una variabile *standard* o *safe*. Nella tipologia di comunicazione mista, per permettere una più facile interpretazione dei frame sniffati, dato che il dissezionatore del protocollo SafetyNET p non distingue i messaggi *safe* e li indica come *standard*, sono state generate un'onda quadra *safe* e un segnale costante *standard* imposto a valore logico alto.

Per comunicazione mista si intende trattare dentro alla stessa POU sia variabili *elementary data type* che *safe data type* e non realizzare due distinte POU, per ogni tipologia di dato, che vanno inserite ognuna nella rispettiva risorsa ST o FS del dispositivo PSSu PLC. In questa maniera, essendo all'interno della stessa POU, entrambe le variabili saranno trattate con la stessa priorità, differentemente dall'altra realizzazione in cui le POU FS vengono trattate con priorità superiore rispetto alle ST.

In ciascuna tipologia di comunicazione queste variabili sono poi mappate su un'uscita del *moduli I/O* della rispettiva tipologia, *PSSu E S 4DO 0.5* o *PSSu E F 4DO 0.5*, connesse al dispositivo *PSSu H FS SN SD* (*PSSu I/O*).

Tramite Wireshark è possibile analizzare la trasmissione di queste variabili dal *PSSu PLC* al *PSSu I/O*, tramite la rete Ethernet SafetyNET p. Alla scheda Ethernet *eth1* del PC, su cui è in ascolto l'analizzatore di rete, vengono indirizzati tutti i frame scambiati fra i due dispositivi mediante la funzione di *port mirroring* dello switch. In quest'ultimo componente di rete è stato configurato il *mirroring* dei dati provenienti e diretti alla porta 1, a cui è connesso il PLC, sulla 8, a cui è connessa la scheda di rete *eth1* del PC.

In questa *configurazione semplice* di comunicazione, utilizzando la suite PAS4000, sono stati impostati manualmente gli indirizzi IP ai due dispositivi utilizzati, scegliendoli nella stessa sottorete. Per la scheda di rete *eth1* è stato impostato un indirizzo IP appartenete alla medesima sottorete utilizzata per i dispositivi PSSu per poter eseguire il caricamento del programma utente. In Tabella 7.5 sono riportate le attribuzioni MAC-IP per ciascun dispositivo.

Dispositivi	**MAC**	**IP**
PSSu PLC	00:02:48:40:3B:67	192.168.1.2
PSSu I/O	00:02:48:40:3A:09	192.168.1.4
eth1	00:04:75:7F:2C:7A	192.168.1.11

Tabella 6.1: associazione MAC/IP dei dispositivi nella *configurazione semplice*

6.1.1 Comunicazione puramente *standard*

Con questa analisi si vuole verificare l'implementazione delle specifiche del protocollo, definite nello standard SafetyNET p, e capire la strutturazione, anche temporale, della comunicazione *standard* tra due dispositivi.

Per questa analisi, facendo riferimento alla configurazione dell'apparato sperimentale di Figura 6.1, è stato scritto mediante la suite software PAS4000 un programma, chiamato *OndaQuadra_STANDARD*, che genera al suo interno un'onda quadra *standard* con periodo pari a $10T_c$, cioè 10 volte il tempo di ciclo della trasmissione e *duty cycle* del 50%. Il tempo ciclo T_c è stato impostato a $14ms$. Il programma utente viene caricato sul PSSu PLC e la variabile riferita a tale segnale viene associata all'uscita del modulo *PSSu E S 4DO 0.5* del dispositivo PSSu I/O. In questa maniera il valore del segnale viene trasferito dal nodo mittente, tramite la rete Ethernet SafetyNET p, al destinatario che abilita l'uscita prestabilita sulla scheda conformemente ai dati ricevuti.

```
PROGRAM OndaQuadra_STANDARD
------------------------------
VAR
MODULO:DINT:=4;
OUTPUT AT%Q*:BOOL;
(*è il segnale onda quadra*)
INDICE:DINT:=4;
END_VAR
------------------------------

LD      INDICE
GE      MODULO
JMPC    AZZERA
LD      INDICE
ADD     1
ST      INDICE
JMP     FINE
AZZERA:
LD      0
ST      INDICE
LDN     OUTPUT
ST      OUTPUT
FINE:

END_PROGRAM
```

Parte DICHIARATIVA delle variabili

Figura 6.2: codice del programma utente *OndaQuadra_STANDARD*

Dalle sniffate effettuate mediante l'analizzatore di rete Wireshark, si nota che in fase di *creazione della comunicazione*, rappresentata in Figura 6.3, i due dispositivi coinvolti si scambiano dei messaggi *CDCN subscribe* per la sottoscrizione dei dati di processo pubblicati.

Nello specifico, essendo il PSSu PLC che genera ed invia i dati *standard*, solamente il PSSu I/O provvede alla sottoscrizione degli unici dati presenti nella rete e,dunque, lo scambio di messaggi confermati *CDCN subscribe* avviene solo in un verso di comunicazione. Per fare ciò, tale nodo invia all'altro dei messaggi *CDCN subscribe request* in cui, come si può vedere in Figura 6.4, è presente il PID associato all'unica tipologia di informazione trasmessa fra i due.

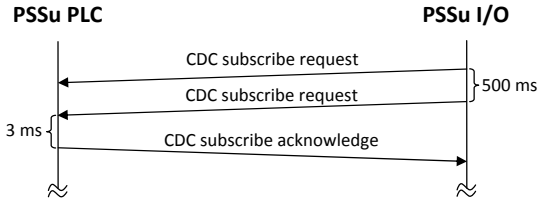


Figura 6.3: fase di *creazione della comunicazione standard*

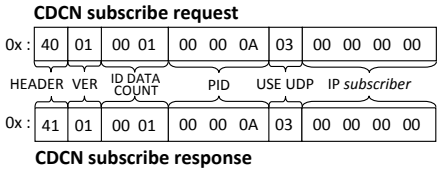


Figura 6.4: codifica dei messaggi *CDCN subscribe*

Questi messaggi, però, possono essere inviati da ciascun dispositivo solamente dopo che esso è transitato dallo stato di *start-up* allo stato di *inizializzazione*. Quindi, dall'accensione dei dispositivi alla visualizzazione dei messaggi *CDCN subscribe*, mediante l'analizzatore di rete, intercorrere del tempo che dipende dalla velocità del dispositivo di portarsi nello stato di inizializzazione.

In entrambi i messaggi *CDCN subscribe* il PID assume, nei 3 byte, il valore esadecimale 0x00 00 0A mentre il campo *IP address*, che riporta l'indirizzo IP del dispositivo *subscriber*, viene lasciato completamente vuoto 0x00 00 00 00. Tramite il valore 0x03 del campo *use udp* viene comunicato l'utilizzo del protocollo UDP; in particolare analizzando con il dissezionatore i singoli bit che lo compongono si nota che i 2 bit meno significativi (*Less Significative Bit* - LSB) sono associati ciascuno rispettivamente all'utilizzo del protocollo UDP e all'assegnazione di una VLAN, mentre gli altri sono riservati. Il campo *IP address* è lasciato vuoto in quanto, dato che è utilizzato il protocollo UDP di trasmissione che si appoggia già sull'IP, si fa riferimento direttamente all'indirizzo del mittente riportato nell'*UDP header*. In questo contesto, la presenza di tale campo potrebbe sembrare uno spreco di byte all'interno dei messaggi scambiati ma in un'ottica più generale non lo è: infatti, il formato di comunicazione RTFN permette la trasmissione di dati anche direttamente a livello MAC e, perciò, può essere necessario al *publisher* conoscere l'indirizzo IP del *subscriber*.

Il PSSu I/O continua ciclicamente a mandare messaggi di richiesta di sottoscrizione all'altro dispositivo fino a quando non risponde con un *CDCN subscribe acknowledge*, che ne permette la condivisione dei dati. I dispositivi PSSu PLC, essendo dal punto di vista computazionale più complessi dei PSSu I/O, impiegano molto più tempo ad entrare nello stato operativo perchè devono anche trasferire il programma utente, dalla memoria di massa alla memoria volatile RAM. Per questo motivo la richiesta di sottoscrizione avviene in maniera ciclica, con un periodo molto superiore rispetto al tempo ciclo del programma utente. Quando la risposta positiva alla richiesta di sottoscrizione avviene entro un determinato *tempo di validità di risposta* l'inoltro ciclico di *CDCN subscribe request* viene interrotto e comincia la trasmissione dei dati ciclici di processo, altrimenti non viene considerata e l'invio periodico continua. Nel caso analizzato, come si può vedere in Figura 6.3, il PSSu PLC risponde alla seconda richiesta di sottoscrizione dei dati da parte del PSSu I/O.

Dopo lo scambio consecutivo dei messaggi *CDCN subscribe request* e *CDCN subscribe acknowledge*, inizia la trasmissione dei dati tra il *publisher*, PSSu PLC, e il *subscriber*, PSSu I/O, che viene definita *svolgimento della comunicazione*. Analizzando questa comunicazione, rappresentata schematicamente in Figura 6.5, si vede che, ad ogni ciclo del programma, il PSSu PLC invia al PSSu I/O il valore del segnale onda quadra.

In Figura 6.5 è riportata una sezione dell'acquisizione che rappresenta la comunicazione dei valori assunti durante un intero periodo del segnale onda quadra. Infatti, tramite il PAS4000 è stato imposto che la variabile *standard* restasse per 5 cicli consecutivi allo stesso livello logico e poi si invertisse. Si può quindi generalizzare, senza perdita di specificità, la struttura della comunicazione dicendo che essa è costituita da una ripetizione continua dei blocchi rappresentati figura.

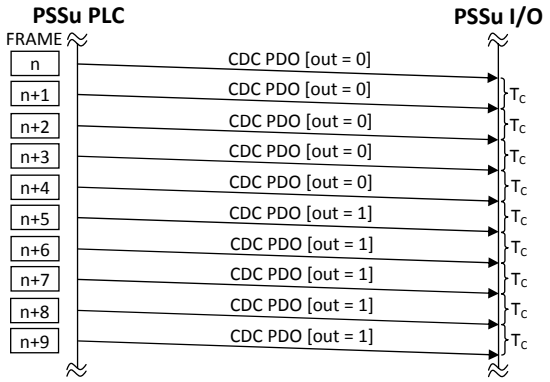


Figura 6.5: estratto dallo *svolgimento della comunicazione standard*

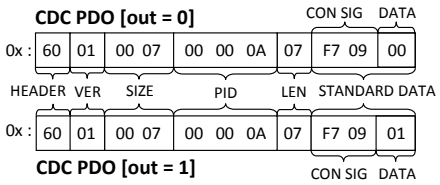


Figura 6.6: codifica dei messaggi *CDCN packet*

I messaggi CDC che trasportano i dati *standard* possiedono tutti la stessa codifica, rappresentata in Figura 6.6. L'*header* è pari a 0x60, essendo messaggi ciclici per il trasporto di dati di processo, il *CDCN protocol version* a 0x01, che indica la struttura di codifica del messaggio descritta nella prima versione dello standard e il *size* a 0x00 07, che riporta il numero di byte contenuto nel *CDC payload*. Il dispositivo comprende che, all'interno di questo campo, è trasmesso un unico *CDC packet* perchè il campo *len*, che ne indica la lunghezza in byte, assume lo stesso valore 0x07 del *size*. Va notato che i due campi possiedono due dimensioni in byte differenti, in quanto il numero di ottetti trasmessi in un *CDC packet* è limitato a 255 byte mentre quelli che possono essere trasmessi nel *payload* di un *SNp TYPE frame*, che comprende i campi *CDCN protocol version*, *size* e *CDC payload*, sono 1499. Per cui, sottraendo a 1499 i 3 byte dell'intestazione e convertendo in esadecimale tale risultato si ottiene una lunghezza massima di 0x05 D8 byte per il campo *CDC payload*, che per essere rappresentata necessita di 2 ottetti.

All'interno dell'unico pacchetto ciclico *CDC packet* sono presenti il PID, che corrisponde a quello scambiato tra i dispositivi in fase di sottoscrizione dei dati e il campo *standard data*, costituito da 3 byte totali, che viene ulteriormente distinto dal disezionatore SafetyNET p di Pilz, in *configuration signature* (*con sig*) di 2 byte e in *data* di 1 byte, evidenziati in Figura 6.7 in rosso.

Il campo *data* riporta il valore della variabile che, in questo caso, può essere solo 0x00 o 0x01 poichè è il valore booleano che serve ad azionare direttamente l'uscita di un modulo, mentre il *con sig* indica la struttura di configurazione. Dalle analisi effettuate si è riusciti a stabilire che i valori di questo campo rappresentano un codice, assegnato dal PAS4000 durante il *download* del programma, che associa il valore della variabile in *data* all'uscita del modulo I/O prestabilita. Ad esempio, quando devono essere trasmessi i valori di due variabili *standard* a due moduli I/O di uno stesso dispositivo PSSu, il mittente invece di inviare periodicamente due differenti *CDC packet*, in cui sono riportati un codice univoco identificativo della variabile e il valore assunto, ne trasmette un solo pacchetto, in cui allo stesso *con sig* sono accodati nel campo *data* i valori delle due uscite 0x0101. Il contenuto del campo *con sig* costituisce, quindi, un codice che rappresenta le uscite dei moduli I/O a cui devono essere associati i valori riportati in *data* e ne permette il corretto

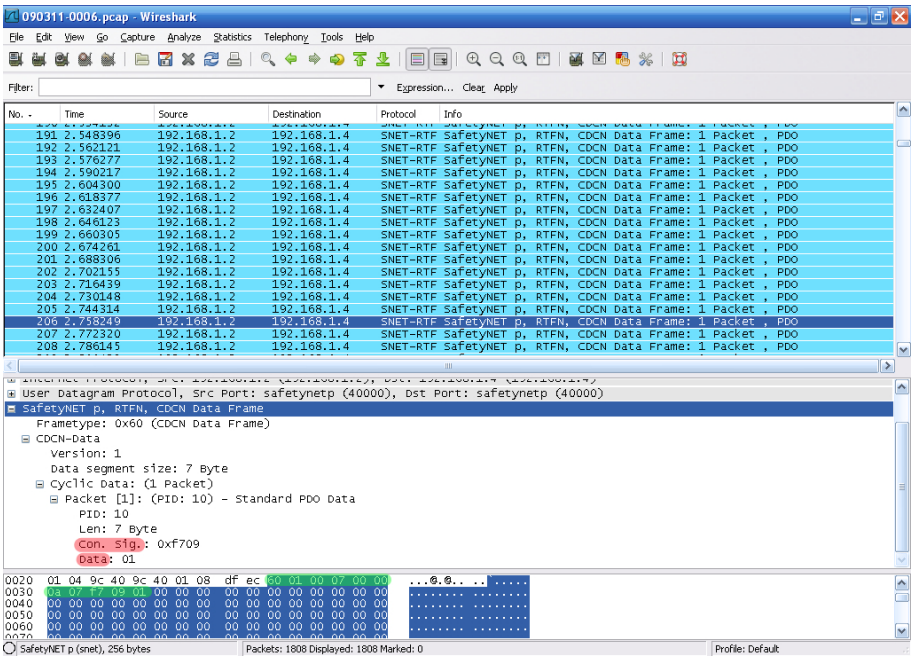


Figura 6.7: *screenshot* della schermata Wireshark della comunicazione *standard*

azionamento. Nella sezione successiva si vedrà che la distinzione di tali campi è effettuata anche per il *safe data*. In particolare, per la trasmissione dei valori di due variabili *safe*, nello stesso verso tra due dispositivi, viene utilizzato lo stesso valore di *con sig*, accodando poi nel campo *data* il valore effettivo di entrambe. Nel caso, però, di comunicazione mista nella stessa direzione fra i dispositivi, sono utilizzati due differenti valori di *con sig* per ciascuna tipologia di dato, *standard* o *safe*.

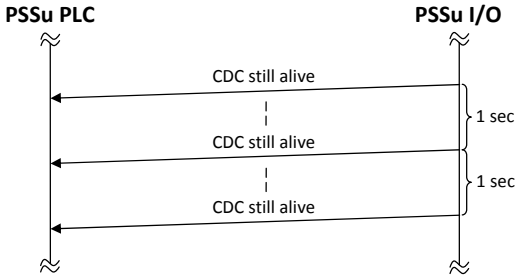


Figura 6.8: tempistiche dei messaggi *CDCN still alive*

Va considerato che quanto affermato nei capitoli precedenti riguardo alla creazione delle immagini di ingresso e di uscita da parte dei dispositivi PSSu non è in contrasto con la trasmissione dei dati di processo *standard* ad ogni periodo (anche se la variabile assume lo stesso valore per più periodi) perchè esse non riguardano l'attuazione delle uscite su un dispositivo remoto, tramite la rete Ethernet, ma sui moduli I/O, tramite l'MBUS, connessi al nodo stesso. Infatti, la trasmissione del valore del segnale onda quadra viene fornito ciclicamente dal PSSu PLC al PSSu I/O, a cui spetterà poi l'attuazione sulla scheda di uscita digitale *standard* a lui collegata mediante l'aggiornamento effettivo della variabile solamente nell'istante

in cui commuta di stato. Durante lo *svolgimento della comunicazione* questi dati ciclici vengo trasmessi tra i dispositivi solamente in un unico verso, dal PSSu PLC al PSSu I/O, in modalità non confermata. Quindi, per informare il *publisher* del suo stato operativo, il *subscriber* invia i messaggi *CDCN still alive* circa ogni secondo, come si può notare in Figura 6.8. Questa periodicità si è verificato essere indipendente dal tempo di ciclo del programma utente; ogni volta resta pari a $1sec$ nonostante siano state effettuate più prove impostando diversi valori di T_c.

6.1.2 Comunicazione puramente *safe*

Con questa analisi si vogliono verificare le differenze che intercorrono tra una comunicazione *standard* e una *safe*, aspettandosi già preliminarmente che la complessità sarà nettamente superiore. Non solo per la struttura ridondante dei dati di processo ciclici ma anche per la presenza dei messaggi confermati *Safe HeartBeat* (*SHB*).

La configurazione di rete utilizzata per questa analisi è riportata in Figura 6.1, in cui però sul PSSu PLC viene caricato il programma *OndaQuadra_SAFE*. In Figura 6.9 è riportato il codice del programma utente, realizzato tramite la suite software PAS4000. Si nota immediatamente che equivale al programma *OndaQuadra_STANDARD* (Figura 6.7 pag. 81) utilizzato per l'analisi precedente ma la definizione della variabile di uscita è definita come *safebool*, e non solo *bool*, perchè la si vuole utilizzare per la trasmissione di dati di processo ciclici *safe*. Viene così generato un segnale onda quadra *safe* con periodo pari a $10T_C$ e *duty cycle* del 50%, ossia ogni 5 tempi ciclo cambia il valore dello stato.

```
PROGRAM OndaQuadra_SAFE
-----------------------------
VAR
MODULO:DINT:=4;
OUTPUT AT%Q*:SAFEBOOL;
(*è il segnale onda quadra*)
INDICE:DINT:=4;
END_VAR
-----------------------------

LD      INDICE
GE      MODULO
JMPC    AZZERA
LD      INDICE
ADD     1
ST      INDICE
JMP     FINE
AZZERA:
LD      0
ST      INDICE
LDN     OUTPUT
ST      OUTPUT
FINE:

END_PROGRAM
```

Parte DICHIARATIVA delle variabili

Figura 6.9: codice del programma utente *OndaQuadra_SAFE*

La variabile viene mappata su una uscita del modulo *PPSu E F 4DO 0.5* del dispositivo PSSu I/O. Durante la comunicazione il valore del segnale viene trasferito, mediante la rete Ethernet SafetyNET p , dal PSSu PLC al destinatario, che ne provvede all'attuazione sulla scheda di uscita mediante il proprio MBUS.

Dalle sniffate effettuate con l'analizzatore di rete Wireshark, come già anticipato nella Sezione 5.4 di pag. 70, si è constatato che il dissezionatore SafetyNET p di Pilz non riesce ad identificare i messaggi *safe*, che invece interpreta come *standard*. Questo non è stato un problema, in quanto la struttura ridondante di tali messaggi ne ha permesso una semplice decodifica, andando a leggere direttamente i byte situati nel *SNp TYPE frame* contenuto all'interno del frame Ethernet sniffato mediante Wireshark. Va, quindi, segnalato che negli eventuali *screenshot* riguardanti una trasmissione *safe* o mista, riportati in questi capitoli, saranno presenti delle incongruenze nella decodifica che sono da imputare solamente al non ottimale funzionamento della versione del dissezionatore a disposizione.

Cominciando ad analizzare i risultati sperimentali ottenuti, si vede che la fase di *creazione della comunicazione*, rappresentata in Figura 6.10, è decisamente molto più complessa dell'equivalente *standard*. I dispositivi si scambiano i messaggi *CDCN subscribe* ma differiscono dal caso precedente per la presenza della richiesta di sottoscrizione effettuata da entrambi i dispositivi, per il numero di ritrasmissioni in caso di mancata risposta e per il numero maggiore di PID utilizzati per identificare l'informazione che si può sottoscrivere.

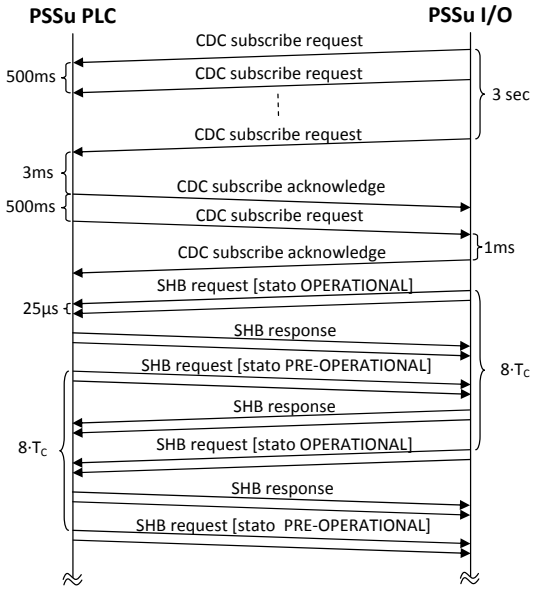


Figura 6.10: fase di *creazione della comunicazione safe*

La comunicazione dei messaggi *CDCN subscribe* è effettuata da entrambi i dispositivi perchè deve essere anche richiesta la sottoscrizione ai messaggi SHB da parte del nodo PSSu PLC al PSSu I/O. Le tempistiche di invio dei messaggi *CDCN subscribe* sono le stesse del caso precedente, anche se in questo caso il dispositivo PSSu PLC aspetta molto più tempo ad inviare la richiesta di sottoscrizione dei dati da parte del nodo PSSu I/O, perchè oltre alla fase di inizializzazione in cui effettua il caricamento del programma deve anche predisporre tutte le funzionalità e i meccanismi relativi alla comunicazione *safe*. Per tanto, le tempistiche per entrare nello stato di inizializzazione si dilatano.

La struttura di questi messaggi, riportata in Figura 6.11, è più complicata rispetto alla precedente perchè in ciascun messaggio di *request* e *acknowledge* sono

CDCN subscribe request MITTENTE: PSSu I/O DESTINATARIO: PSSu PLC

H	VER	ID DATA COUNT		PID		USE UDP	IP *subscriber*				PID		USE UDP	IP *subscriber*				PID		USE UDP	IP *subscriber*			
40	01	00	03	00	00	0A	03	00	00	00	00	00	00	14	03	00	00	00	00	00	00	32	03	00	00	00	00

CDCN subscribe acknowledge MITTENTE: PSSu PLC DESTINATARIO: PSSu I/O

H	VER	ID DATA COUNT		PID		USE UDP	IP *subscriber*				PID		USE UDP	IP *subscriber*				PID		USE UDP	IP *subscriber*			
41	01	00	03	00	00	0A	03	00	00	00	00	00	00	14	03	00	00	00	00	00	00	32	03	00	00	00	00

CDCN subscribe request MITTENTE: PSSu PLC DESTINATARIO: PSSu I/O

H	VER	ID DATA COUNT		PID		USE UDP	IP *subscriber*				PID		USE UDP	IP *subscriber*				
40	01	00	02	00	00	1E	03	00	00	00	00	00	00	28	03	00	00	00	00

CDCN subscribe acknowledge MITTENTE: PSSu I/O DESTINATARIO: PSSu PLC

H	VER	ID DATA COUNT		PID		USE UDP	IP *subscriber*				PID		USE UDP	IP *subscriber*				
41	01	00	02	00	00	1E	03	00	00	00	00	00	00	28	03	00	00	00	00

Figura 6.11: struttura dei vari messaggi *CDCN subscribe* sniffati

contenuti i PID relativi all'informazione disponibile per la sottoscrizione. Questa informazione rappresenta l'invio dei dati ciclici *safe*, in un verso solo, e dei messaggi SHB, in entrambi i versi di comunicazione. Nello specifico, vengono trasmesse 3 tipologie di messaggi da parte del dispositivo PSSu PLC e 2 da parte del PSSu I/O. In particolare, come si vedrà nella fase di *svolgimento della comunicazione*, i PID 0x00 00 0A, 0x00 00 14 e 0x00 00 32 sono rispettivamente associati ai messaggi contenenti il dato *safe*, l'*SHB request* e l'*SHB response*, inviati dal PSSu PLC, mentre i PID 0x00 00 28, 0x00 00 1E sono rispettivamente associati ai messaggi *SHB request* e l'*SHB response* inviati dal PSSu I/O.

Il campo *ID data count* è utilizzato all'interno dei messaggi *CDCN subscribe* per indicare il numero di *ID data* contenuti in esso; in questo caso assume il valore 0x03 per quelli inviati dal PSSu I/O, dato che sono appunto 3 le tipologie di PID a cui vuole sottoscriversi, e 0x02 per quelli dal PSSu PLC. Anche in questo caso, il campo *IP address*, che riporta l'indirizzo IP del dispositivo *subscriber*, viene lasciato completamente vuoto 0x00 00 00 00.

Una volta confermata la sottoscrizione dei dati *CDCN request*, in entrambi i sensi, non inizia subito la comunicazione dei dati ciclici ma comincia la trasmissione dei messaggi confermati SHB perchè, come si può notare dai messaggi *SHB request* inviati al PSSu I/O di Figura 6.10, il dispositivo PSSu PLC non è ancora entrato nello *stato operativo*. In questo stato l'unica tipologia di messaggi abilitati sono gli SHB. Dato che i messaggi confermati SHB sono utilizzati in una comunicazione *safe* per avere un riscontro più affidabile rispetto ai *CDCN still alive*, l'invio dello stato del dispositivo e l'abilitazione preliminare della comunicazione permettono al dispositivo più veloce di diventare operativo e di attendere l'operatività dell'altro nodo senza segnalare un errore di comunicazione, altrimenti generato se non fosse stato concepito alcun tipo di comunicazione SHB durante la fase preoperativa.

La trasmissione dei messaggi SHB avviene in doppiette, ossia ogni dispositivo invia due copie identiche dello stesso messaggio in successione ad una distanza temporale molto bassa, di circa $25\mu s$. Inoltre, l'invio dei messaggi *SHB request*, effettuato da ciascun dispositivo, avviene con periodicità pari a $8T_C$ e, quindi, dipendentemente dal tempo ciclo del programma utente.

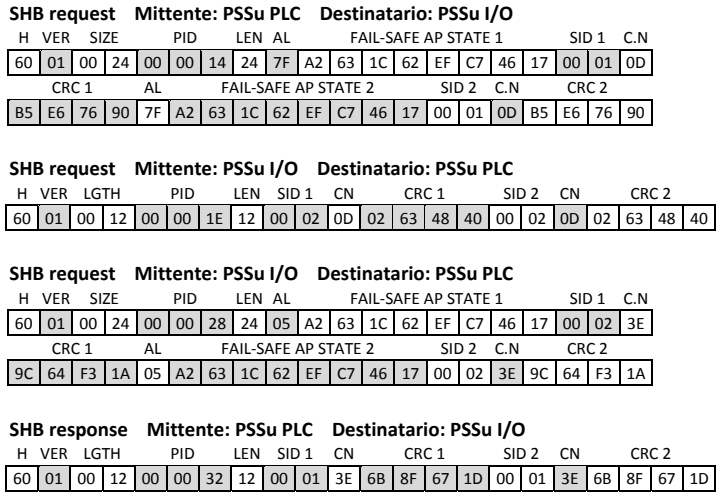


Figura 6.12: struttura di alcuni messaggi *SHB request* e *SHB response* sniffati

I messaggi SHB sono trasmessi nel canale CDC tramite il frame *CDCN PDU*, dentro al campo *data* di un*CDC packet*. Come nota in Figura 6.12, in accordo a quanto descritto nello standard del protocollo SafetyNET p, la struttura dei messaggi SHB è ridondata; il campo *PID* e *len* sono unici mentre vengono ripetuti il *fail-safe AL state*, *fail-safe AP state*, il *SID*, il *consecutive number* e ricalcolato il *CRC*.

Nello specifico, al dispositivo PSSu PLC è associato il valore del *SID* 0x00 01 mentre al PSSu I/O 0x00 02. Nell'implementazione del protocollo, effettuata da Pilz, sembra superfluo il ruolo del campo SID in quanto, all'interno di una comunicazione SHB fra due dispositivi, sono utilizzati PID differenti per ciascun messaggio e, dunque, esso identifica già univocamente i frame trasmessi senza che il SID porti ulteriore informazione per la differenziazione. Esso sarebbe, invece, determinante se venissero risparmiati dei PID, assegnandone uno per ciascun scambio di *request* e *response*. In questo caso, il PID sarebbe identico e per differenziare le richieste e le risposte verrebbe utilizzato il SID che specifica il dispositivo mittente. La strategia di implementazione attuata, però, è condivisibile in quanto permette una maggiore efficacia nel rilevamento degli eventuali errori di comunicazione (si veda la Figura 3.5 di pag. 31), come l'inserimento di un falso messaggio nella rete (*insertion*), la ricezione di un messaggio destinato ad un altro nodo della rete (*addressing*), gli errori di memoria negli switch (*Revolving memory failures within switches*) e il mascheramento di un messaggio *safe* da parte di uno *standard* che possiede casualmente la stessa struttura di byte ridondata nel proprio campo *standard data* (*masquerade*). Il campo *consecutive number*, che contiene un numero identificativo crescente del pacchetto trasmesso, è differente tra le varie comunicazioni *confermate* mentre è lo stesso per i messaggi *SHB requeste response* appartenenti ad ognuna

di esse. Il campo *fail-safe AP state* contiene la codifica dell'informazione sullo stato dell'applicazione di processo *safe* che dipende dalle scelte progettuali di Pilz. Nel caso in analisi esso assume il valore esadecimale 0xA2 63 1C 62 EF C7 46 17 in entrambi i versi delle comunicazioni *confermate* SHB, perchè i messaggi SHB sono riferiti alla trasmissione di dati *safe* tra gli stessi dispositivi. Il campo *fail-safe AL state*, invece, riporta la codifica dell'informazione sullo stato dell'*application layer safe* dei dispositivi: il valore 0x05 rappresenta lo stato *operational* e 0x7F quello *pre-operational*.

Questo susseguirsi di soli messaggi *SHB request* e *SHB response*, rappresentato in Figura 6.10, continua fino a che il dispositivo, ancora nello stato preoperativo, non entra nello stato operativo, in cui può cominciare a trasmettere i dati di processo. Per iniziare l'invio dei dati di processo non è necessario che il dispositivo, una volta operativo, informi l'altro nodo sul proprio stato perchè può immediatamente iniziare la comunicazione ciclica.

Analizzando ora la fase di *svolgimento della comunicazione*, in Figura 6.13, si vede che anche per la trasmissione dei dati ciclici *safe*, mediante *CDCN PDU*, è utilizzata la ripetizione dei frame a una distanza molto ravvicinata, come avviene nel caso dei messaggi SHB. Questa ulteriore ridondanza introdotta da Pilz rappresenta, oltre alle specifiche imposte dallo standard del protocollo SafetyNET p, un meccanismo che implementa le contromisure *data integrity assurance* e *Differential data integrity assurance* per rilevare gli errori di comunicazione *corruption* e *masquerade*. Con la ridondanza dell'intero frame, infatti, se il dispositivo riceve un messaggio *standard* che maschera esattamente la struttura e il valore dei campi di uno *safe*, o un messaggio corrotto inserito nella rete coerentemente con la comunicazione in atto, non trovandone una duplice copia a distanza ravvicinata, riesce a riscontrare l'errore.

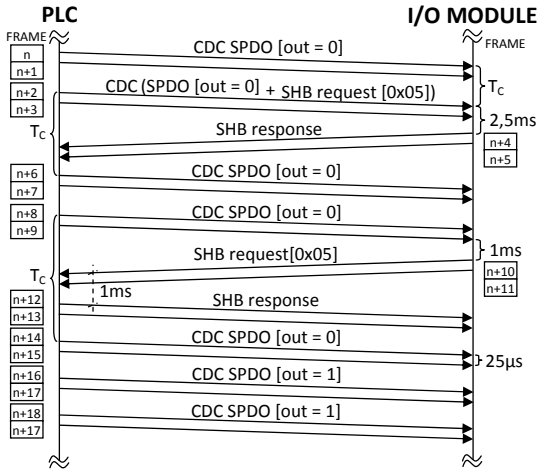


Figura 6.13: estratto dallo *svolgimento della comunicazione safe*

La trasmissione dei dati di processo *safe* avviene ogni tempo ciclo T_C e poichè è stato riscontrato sperimentalmente che, per diversi valori di T_C, anche la comunicazione dei messaggi *SHB request* avviene con una propria periodicità di $8T_C$, è possibile che il dispositivo PSSu PLC debba inviare, a brevissima distanza, entrambi i messaggi. Dal momento che il funzionamento real-time della rete è prio-

ritario, deve essere garantito il determinismo nell'invio dei messaggi. Per questo, il dispositivo, in tali casi, accoda sia i messaggi *safe* che gli *SHB request* in due *CDCN packet* all'interno dello stesso *CDCN PDU*, come rappresentato in Figura 6.14; negli altri casi, li invia in frame distinti.

Figura 6.14: struttura di alcuni frame *CDCN PDU* della comunicazione *safe*

La struttura del messaggio per la trasmissione dei dati di processo *safe* è, esattamente come per gli SHB, ridondata. In questo caso, però, non vengono distinti i due campi *fail-safe AL state* e *fail-safe AP state* ma ne è presente uno unico chiamato *safe data*, che contiene la codifica degli SPDO. Nel caso in esame, per azionare correttamente l'uscita digitale, esso assume il valore esadecimale 0x 3A E6 00 o 0x 3A E6 01; inoltre, può essere applicata la stessa distinzione dei campo *con sig* e *data* effettuata in precedenza nei messaggi dei dati cicli della comunicazione puramente *standard*. Va notato che il valore del *con sig*, pari a 0x 3A E6, è differente da quello del caso *standard* perchè esso dipende sia dalla configurazione dei dispositivi, che in questo caso è identica, sia dal programma eseguito e dalle uscite azionate sui moduli I/O, che invece sono variate. Il campo *data*, invece, pari a 0x 00 o 0x 01 indica il valore della variabile associata al segnale onda quadra trasmessa dal PSSu PLC al PSSu I/O, mediante la rete Ethernet SafetyNET p. Infine, il campo *consecutive number*, che contiene un numero identificativo crescente del pacchetto di dati *safe* trasmesso, assume valori differenti rispetto a quelli utilizzati da ciascuna comunicazione *confermata*. Si può, quindi, affermare che ciascuna tipologia di dato *safe* possiede un contatore per calcolare il valore del *consecutive number*.

Il flusso di messaggi riportato in Figura 6.10 è solo una porzione della comunicazione *safe*, che è composta dalla ripetizione consecutiva di questa sequenza. Per la trattazione è stato ritenuto più significativo estrapolare una parte di comunicazione che facesse risaltare la periodicità dei messaggi SHB piuttosto che riportare

tutti i messaggi relativi al valore dell'uscita per un intero periodo dell'onda quadra. Si è comunque scelta una parte della comunicazione che contenesse anche il cambio di stato dell'uscita trasferita. Va anche notato che la struttura del flusso dei messaggi rappresentato è solo una delle possibilità perchè dipende strettamente dal preciso istante in cui il dispositivo PSSu PLC ha cominciato ad inviare i dati di processo. Dato che non necessariamente l'inizio della trasmissione dei dati avviene sempre nello stesso momento, la struttura del flusso può subire modifiche nel susseguirsi dei frame nel rispetto, però, delle periodicità dei messaggi *CDCN PDU* e *SHB request* riscontrate. Ad esempio, tra la richiesta SHB e la risposta potrebbe interporsi una doppietta di frame contenenti i dati di processo *standard* oppure fra le *SHB request* effettuate nei due versi può essere presente un numero maggiore o minore di doppie CDCN PDU. Questo perchè è garantita la periodicità solo per le *SHB request* inviate dallo stesso dispositivo mentre da parte del destinatario l'invio delle *SHB response*, in caso occorra nello stesso istante in cui sia in attesa di un messaggio contenete i dati di processo *safe*, è eseguito subordinatamente alla ricezione dal CDCN PDU.

Per concludere, all'interno di questa comunicazione, come nel caso *standard* precedente, il dispositivo PSSu I/O trasmette periodicamente, circa ogni 1*sec*, i messaggi *CDCN still alive* per comunicare al PSSu PLC che funziona correttamente e che i dati possono essere inviati.

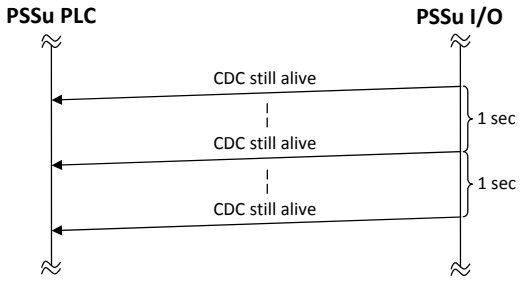


Figura 6.15: tempistiche dei messaggi *CDCN still alive*

Questa funzionalità, anche se superflua rispetto ai messaggi SHB, è mantenuta nel caso di una comunicazione *safe* perchè permette di discriminare eventuali situazioni di errori che possono intercorrere nella rete. Se la comunicazione *safe* viene interrotta può essere discriminato il caso in cui, ad esempio, l'interruzione è dovuta ad un errore di trasmissione che ha portato nello stato sicuro il nodo destinatario dei dati di azionamento dal caso in cui è stato rimosso completamente il collegamento Ethernet fra i due dispositivi; nel primo caso il mittente riceverebbe solamente i messaggi *CDCN still alive* da parte del destinatario e non gli SHB, mentre nel secondo non riceverebbe niente.

6.1.3 Comunicazione mista

Con questa analisi si vuole verificare la struttura di una comunicazione mista, in cui cioè coesistono la trasmissione di dati si *standard* che di dati *safe*. Tale comunicazione mista è stata realizzata mediante l'esecuzione di un unico programma

POU, scritto tramite la suite software PAS4000, in cui sono generate sia variabili *elementary data types* che *safe data type*.

In particolare, per questa analisi è stata utilizzata la configurazione dell'apparato sperimentale di Figura 6.1 ed è stato scritto il programma chiamato *OndaQuadra_ MISTA*, riportato in Figura 6.18.

Per semplificare la distinzione delle due tipologie di dati trattati, vengono generate all'interno dello stesso programma un'onda quadra *standard* con periodo pari a $10T_c$ e *duty cycle* del 50% e una variabile costante imposta a valore logico alto.

```
PROGRAM OndaQuadra_MISTA
-----------------------------
VAR
MODULO:DINT:=4;
OUTPUT_SAFE AT%Q*:SAFEBOOL;
(*è il segnale onda quadra*)
OUTPUT_STANDARD AT%Q*:BOOL;
(*è il segnale onda quadra*)
INDICE:DINT:=4;
END_VAR
-----------------------------

LD        TRUE
ST        OUTPUT_STANDARD
LD        INDICE
GE        MODULO
JMPC      AZZERA
LD        INDICE
ADD       1
ST        INDICE
JMP       FINE
AZZERA:
LD        0
ST        INDICE
LDN       OUTPUT_SAFE
ST        OUTPUT_SAFE
FINE:

END_PROGRAM
```

Parte DICHIARATIVA delle variabili

Figura 6.16: codice del programma utente *OndaQuadra_ MISTA*

Il programma utente viene caricato sul PSSu PLC e le variabili *standard* e *safe* vengono rispettivamente associate ad un'uscita dei moduli *PSSu E S 4DO 0.5* e *PSSu F 4DO 0.5* del dispositivo PSSu I/O. In questa maniera, il valore del segnale viene trasferito dal nodo mittente, tramite la rete Ethernet SafetyNET p, al destinatario che abilita l'uscita prestabilita sulla scheda conformemente ai dati ricevuti.

Dalle sniffate effettuate mediante l'analizzatore di rete Wireshark, si nota che il flusso dei frame durante la fase di *creazione della comunicazione* è analogo a quella della comunicazione puramente *safe* (Figura 6.11 di pag. 84), con l'unica differenza che, all'interno dei messaggi *CDCN subscribe*, vengono trasmessi anche i PID relativi ai dati di processo *standard*. In totale, quindi, il dispositivo PSSu I/O richiede al PSSu PLC la sottoscrizione di 4 tipologie di messaggi che trasportano informazione, mediante la trasmissione di un PID per i dati di processo *standard*, uno per quelli *safe* e due rispettivamente per i messaggi *SHB request* e *SHB resonse*. Nell'altro verso, invece, vengono sottoscritti solo i PID relativi ai messaggi SHB.

Durante la fase di *svolgimento della comunicazione* vengono ripresentate le caratteristiche del flusso analizzato nella comunicazione puramente *safe* in termini di successione della tipologia di messaggi e rispetto delle tempistiche e periodicità.

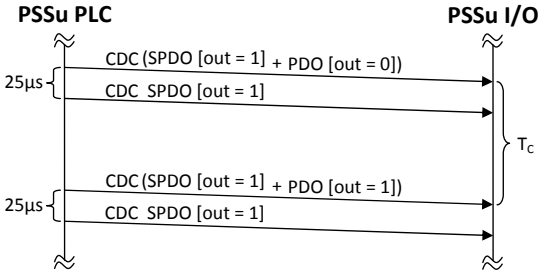


Figura 6.17: differente invio dei dati di processo in una comunicazione *mista*

La trasmissione dei dati *standard* viene effettuata accodando in un unico *CDCN PDU* il primo messaggio della doppietta *safe* con il messaggio ad essi relativo. Come si può notare in Figura 6.17 e in Figura 6.18, il primo frame della doppietta di dati è costituito da un frame *CDCN PDU*, che contiene un *CDC packet* per ogni tipo di dati, *standard* e *safe*, mentre il secondo contiene solo quello relativo ai dati di processo *safe*.

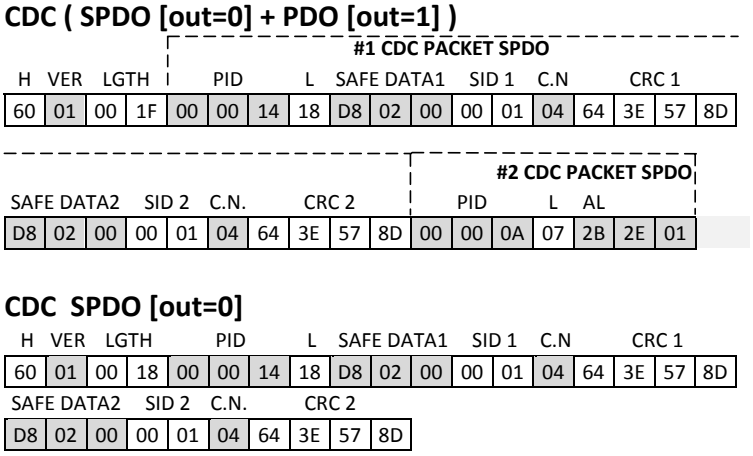


Figura 6.18: struttura di una doppietta della comunicazione mista sniffata

Questo inglobamento in un unico frame delle due tipologie di dati di processo permette di rispettare le tempistiche, scandite dal tempo ciclo, di invio dei messaggi fra i due dispositivi. A tal fine, anche nel frame che nel caso precedente conteneva accodati il messaggio con i dati di processo *safe* e *SHB request*, viene accodato anche il pacchetto relativo ai dati *standard* per un totale di 3 *CDCpacket* all'interno di un'unica *CDCN PDU*.

Infine, anche per questa tipologia di comunicazione vengono trasmessi messaggi *CDCN still alive* fra il dispositivo PSSu I/O e il PSSu PLC con le stesse tempistiche riscontrate nei precedenti casi di comunicazione puramente *standard* e *safe*.

6.1.4 Conclusioni

Da queste analisi è stato possibile comprendere come sono costituite le due tipologie di comunicazioni *standard* e *safe*, quali sono gli elementi che le differenziano e come

possono coesistere in una comunicazione mista.

Le parti comuni a tutte le tipologie di comunicazione sono la parte di scambio dei messaggi *CDC subscribe*, per sottoscrivere i dati necessari da parte di un dispositivo, e dei *CDCN still alive*. Sono state analizzate le tipologie di messaggi che possono essere trasmessi in ciascun stato di funzionamento dei dispositivi, verificando la conformità della struttura dei messaggi con quanto descritto nello standard e caratterizzando la periodicità con cui vengono scambiati i messaggi contenenti di dati di processo, i *CDCN subscribe*, gli *SHB request* e gli *still alive*.

Per la comunicazione *safe* si è visto come, rispetto a quella *standard*, vengono utilizzati dei messaggi confermati *SHB request* e *SHB response* per monitorare periodicamente lo stato dei dispositivi durante la trasmissione che prevedono la generazione ed assegnazione di molti PID in più.

Uno dei più importanti riscontri consiste nel fatto che, nell'implementazione del protocollo SafetyNET p effettuato da Pilz, i messaggi contenenti i dati di processo ciclici *safe* vengono mandati in doppiette che contengono lo stesso identico frame inviato però due volte ad una brevissima distanza temporale. Questo meccanismo di trasmissione è certamente stato concepito per agevolare ulteriormente la rilevazione di possibili errori di comunicazione.

In caso in cui più messaggi richiedano un invio simultaneo, per garantire il determinismo si è visto come all'interno di un *CDCN PDU* possano essere inseriti più pacchetti contenenti informazione. Nella doppietta di dati *safe* della comunicazione mista viene modificato il primo dei due frame, a cui viene accodato anche il messaggio contente i dati *standard*, per permettere un preciso invio periodico dei dati di processo.

Dalle analisi effettuate su queste comunicazioni *semplici* è stato anche possibile avere un primo approccio con l'utilizzo dei PID per quanto riguarda l'assegnazione, la numerosità e i valori utilizzati. Per capire meglio come vengono assegnati alle varie tipologie di informazione contenuta nei messaggi, è necessario effettuare ulteriori analisi di comunicazioni *complesse*. Con il termine complesso si fa riferimento a comunicazioni che coinvolgono più di due dispositivi, che implica un'analisi più impegnativa del traffico generato sulla rete.

6.2 Comunicazioni complesse

Con l'analisi di queste comunicazioni si vogliono ottenere ulteriori informazioni sulla gestione dei PID da parte dei dispositivi PSSu. In particolare, si vuole capire come vengono gestiti questi codici identificativi dei messaggi, inviando, tramite la rete Ethernet SafetyNET p, due variabili *safe* di uscita dal *PSSu PLC*, ciascuna ad un distinto modulo *PSSu I/O*.

Per vedere se possano essere riutilizzati gli stessi PID per la trasmissione dei valori della stessa variabile, sono stati analizzati due differenti casi: il primo in cui a ciascun dispositivi *PSSu I/O* viene mappata una differente variabile e il secondo in cui, invece, viene mappata la stessa variabile.

In entrambe le comunicazioni analizzate, per non appesantire troppo la trattazione, l'attenzione sarà focalizzata limitatamente alla generazione e all'assegnazione dei PID per ciascuna comunicazione. Le fasi di *creazione* e *svolgimento* della comu-

nicazione vengono tralasciate perchè, anche se più complesse come successione di occorrenza dei frame, non portano informazioni ulteriori; i concetti alla base sono gli stessi di quelli introdotti per le comunicazioni *semplici*. Verrà, però, riportata la struttura dei messaggi relativi ai dati di processo *safe* perchè è rilevante il legame che intercorre fra l'assegnazione dei PID e il *safe data* contenuto.

Con la prima comunicazione si vuole vedere quanti PID vengono assegnati ai messaggi contenenti informazione e come sono trasmessi i dati di processo *safe*, per poter poi fare un confronto con la seconda comunicazione in cui la stessa variabile *safe* viene mappata a delle uscite appartenenti ai moduli I/O di due diversi dispositivi PSSu I/O.

La struttura sperimentale della rete utilizzata per realizzare queste analisi è riportata in Figura 6.19, in cui rispetto a quella per le analisi delle comunicazioni *semplici* è stato introdotto un ulteriore nodo PSSu I/O. Nello specifico, sul PSSu PLC è installato un programma che genera due distinte variabili *safe*. In base alla tipologia della comunicazione che si va ad analizzare, vengono mappate entrambe le variabili, o solo una di esse, su un'uscita del *moduli I/O PSSu E F 4DO 0.5* di due differenti dispositivi *PSSu I/O*.

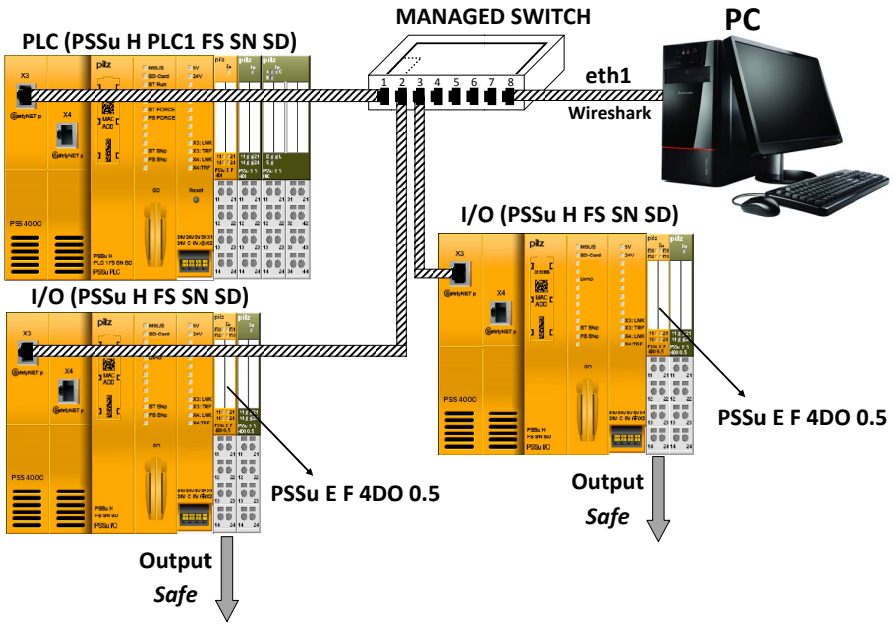


Figura 6.19: struttura del *set up* per le analisi delle *comunicazioni complesse*

Tramite Wireshark è possibile analizzare la comunicazione tra il dispositivo *PSSu PLC* e ciascuno dei due *PSSu I/O*, in cui avviene la trasmissione delle variabili *safe* tramite la rete Ethernet SafetyNET p. In questa configurazione, il *PSSu PLC* resta connesso alla porta 1 dello switch, la scheda di rete *eth1* del PC alla 8 e i due dispositivi, indistintamente, alla 2 e alla 3. Il traffico di rete coinvolge sempre, come mittente o destinatario, il PLC perchè non esistono comunicazioni dirette tra i *PSSu I/O*. Per questo, mediante la funzione di *port mirroring* dello switch, tutti questi messaggi vengono replicati anche alla scheda Ethernet *eth1* del PC, su cui è in ascolto l'analizzatore di rete Wireshark. In questo modo, è possibile ricostruire agevolmente la struttura della comunicazione di tutta la rete.

Per semplificare la nomenclature dei due dispositivi PSSu I/O e per distinguerli si utilizzerà una cifra numerica che fa riferimento all'ultimo campo del loro indirizzo IP; il PSSu I/O 4 sarà il dispositivo con IP paria a 192.168.1.4 mentre il PSSu I/O 5 quello con IP 192.168.1.5.

In Tabella 6.2 sono riportate le attribuzioni MAC-IP per ciascun dispositivo.

Dispositivi	MAC	IP
PSSu PLC	00:02:48:40:3B:67	192.168.1.2
PSSu I/O 4	00:02:48:40:3A:09	192.168.1.4
PSSu I/O 5	00:02:48:40:4F:9F	192.168.1.5
eth1	00:04:75:7F:2C:7A	192.168.1.11

Tabella 6.2: associazione MAC/IP dei dispositivi nella *configurazione complesse*

6.2.1 Comunicazione di variabili *safe* distinte

In questa analisi è stato utilizzata la configurazione dell'apparato sperimentale di Figura 6.19, in cui sul dispositivo PSSu PLC è stato caricato un programma che genera due variabili *safe* distinte, chiamate *out1* e *out2*, che vengono mappate rispettivamente su un'uscita del modulo I/O dei due differenti nodi PSSu I/O. Così, il valore delle variabili viene trasmesso, mediante la rete Ethernet SafetyNET p, dal mittente ai destinatari che provvederanno ad azionare le uscite relative.

Dalle sniffate, effettuate mediante l'analizzatore di rete Wireshark, sono stati rilevati i PID mediante l'analisi dei messaggi *CDCN subscribe* che vengono scambiati fra i dispositivi durante la fase di *creazione della comunicazione*, per la sottoscrizione delle informazioni necessarie. Grazie all'analisi dei singoli frame scambiati, durante lo *svolgimento della comunicazione*, è stato possibile mettere in relazione ciascun PID con l'informazione a cui è associato, come riportato in Tabella 6.3.

PID	Informazione	Mittente	Destinatario
0x 00 00 0A	dato di processo *safe* (valore out1)	PSSu PLC	PSSu I/O 4
0x 00 00 14	dato di processo *safe* (valore out2)	PSSu PLC	PSSu I/O 5
0x 00 00 1E	*SHB request*	PSSu PLC	PSSu I/O 4
0x 00 00 28	*SHB response*	PSSu I/O 4	PSSu PLC
0x 00 00 1E	*SHB request*	PSSu PLC	PSSu I/O 5
0x 00 00 46	*SHB response*	PSSu I/O 5	PSSu PLC
0x 00 00 32	*SHB request*	PSSu I/O 4	PSSu PLC
0x 00 00 3C	*SHB response*	PSSu PLC	PSSu I/O 4
0x 00 00 50	*SHB request*	PSSu I/O 5	PSSu PLC
0x 00 00 5A	*SHB response*	PSSu PLC	PSSu I/O 5

Tabella 6.3: associazione PID e informazione

Come si può notare, ai messaggi *SHB request* inviati dal dispositivo PSSu PLC rispettivamente al PSSu I/O 4 e PSSu I/O 5 è associato un PID con lo stesso valore. Ciò non risulta sorprendente in quanto non è necessario discriminare tra le richieste SHB effettuate a ciascuno dei mittenti ma le risposte da essi ritornate; infatti, entrambi i dispositivi rispondono al PSSu PLC con una *SHB response* a cui sono attribuiti due PID differenti.

CDC SPDO [out1=1] MITTENTE: PSSu PLC DESTINATARIO: PSSu I/O 4

H	VER	LGTH		PID		L	SAFE DATA1		SID 1	C.N		CRC 1			
60	01	00	18	00	00	0A	18	67	8F	01	00	01	4E	B6	9A	A9	40

SAFE DATA2		SID 2		C.N.		CRC 2			
67	8F	01	00	01	4E	B6	9A	A9	40

CDC SPDO [out2=1] MITTENTE: PSSu PLC DESTINATARIO: PSSu I/O 5

H	VER	LGTH		PID		L	SAFE DATA1		SID 1	C.N		CRC 1			
60	01	00	18	00	00	14	18	15	C7	01	00	01	4E	EC	72	00	9E

SAFE DATA2		SID 2		C.N.		CRC 2			
15	C7	01	00	01	4E	EC	72	00	9E

Figura 6.20: struttura dei messaggi relativi alla trasferimento di ciascun dato *safe*

Per la trasmissione delle due differenti variabili *safe*, invece, vengono utilizzati due messaggi distinti, ognuno con il proprio PID. In particolare, la struttura di uno dei due frame delle doppietta di messaggi inviati in successione dal PSSu PLC a ciascun nodo PSSu I/O è riportata in Figura 6.20

All'interno del campo *safe data* sono utilizzati per entrambe le variabili due differenti valori di *con sig*, 0x 67 8F e 0x 15 C7 , che comunicano al dispositivo che riceve il messaggio su quale uscita deve essere azionato il valore 0x 01 contenuto nel successivo campo *data*.

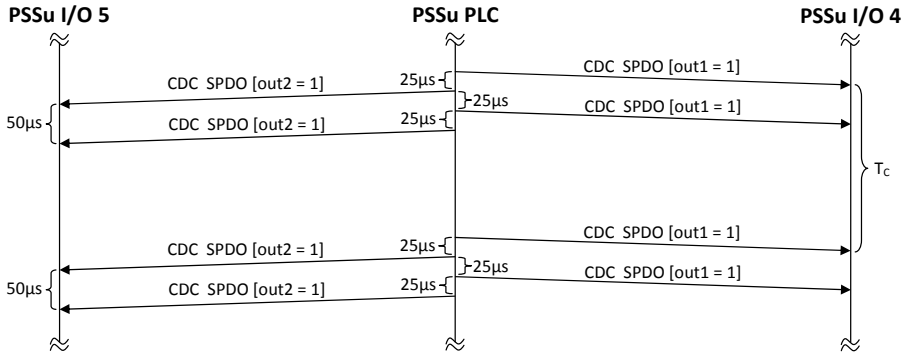


Figura 6.21: differente invio dei dati di processo *safe*

Per ciascun messaggio viene utilizzato lo stesso campo *consecutive number* perchè il PSSu PLC deve inviare i dati di processo agli altri due dispositivi con la stessa periodicità e quindi, una volta cominciata la comunicazione esso trasmetterà lo stesso numero di messaggi ad entrambi. Per garantire l'esatto determinismo nell'invio dei messaggi vengono, come nei casi precedentemente analizzati, accodati nello stesso frame sia i messaggi SHB che quelli relativi ai dati di processo e, in questo specifico caso, il PSSu PLC trasmette le doppiette intervallando ciascun frame per la trasmissione di una variabile *safe* con quello relativo all'altra, come si può notare in Figura 6.21. Il tempo che intercorre tra i frame appartenenti alla stessa doppietta è aumentato in quanto, per limiti hardware, il dispositivo PSSu PLC non riesce ad inviare i messaggi relativi alle due diverse variabili in un tempo inferiore a $25\mu s$.

Grazie a questa analisi è stato possibile capire come avviene la trasmissione di due variabili differenti a due distinti dispositivi ma soprattutto avere ulteriori

informazioni sull'associazione dei PID e su come più dispositivi possano sottoscriverne lo stesso, come è accaduto per i dispositivi nei confronti del messaggio *SHB request* inviato dal PSSu PLC.

6.2.2 Comunicazione di una variabile *safe*

Questa analisi ricalca il caso precedente in cui, però, alle uscite del modulo I/O di ciascun dispositivo PSSu PLC viene mappata la stessa variabile *safe*. Lo scopo è vedere se possono essere utilizzati gli stessi PID per trasmettere il medesimo dato *safe* a due differenti dispositivi oppure se ne devono essere necessariamente utilizzati di diversi.

In questa analisi è stato utilizzata la configurazione dell'apparato sperimentale di Figura 6.19, in cui sul dispositivo PSSu PLC è stato caricato un programma che genera una variabile *safe*, mappata su un'uscita del modulo I/O dei due differenti nodi PSSu I/O. In questa maniera, il valore della variabile viene trasmesso, mediante la rete Ethernet SafetyNET p, dal mittente ad entrambi i destinatari, che provvederanno ad azionare le uscite relative.

Dalle sniffate realizzate mediante l'analizzatore di rete, è stato possibile estrarre dai messaggi *CDCN subscribe*, che vengono scambiati durante la *creazione della comunicazione*, i PID utilizzati dai dispositivi per inviare le differenti tipologie messaggi *CDCN PDU* e metterli in relazione con l'informazione trasmessa in ciascun messaggio grazie all'analisi dello *svolgimento della comunicazione*. L'associazione di ciascun PID con l'informazione contenuta nel messaggio a cui è associato è riportato in Tabella 6.4.

PID	Informazione	Mittente	Destinatario
0x 00 00 0A	dato di processo *safe*	PSSu PLC	PSSu I/O 4
0x 00 00 0A	dato di processo *safe*	PSSu PLC	PSSu I/O 5
0x 00 00 14	*SHB request*	PSSu PLC	PSSu I/O 4
0x 00 00 1E	*SHB response*	PSSu I/O 4	PSSu PLC
0x 00 00 14	*SHB request*	PSSu PLC	PSSu I/O 5
0x 00 00 3C	*SHB response*	PSSu I/O 5	PSSu PLC
0x 00 00 28	*SHB request*	PSSu I/O 4	PSSu PLC
0x 00 00 32	*SHB response*	PSSu PLC	PSSu I/O 4
0x 00 00 46	*SHB request*	PSSu I/O 5	PSSu PLC
0x 00 00 50	*SHB response*	PSSu PLC	PSSu I/O 5

Tabella 6.4: associazione PID e informazione

Si nota subito che, a differenza del caso precedente, viene utilizzato il medesimo PID per i due dispositivi nell'identificazione dei messaggi contenti i dati di processo *safe*, visto che sono riferiti alla stessa variabile.

Per ciascun messaggio che trasmette i dati processo ai due dispositivi viene utilizzato lo stesso valore del campo *consecutive number*, come nell'analisi precedente, e inoltre lo stesso PID e valore contenuto nel campo *safe data*; sono, quindi, identici byte a byte. La codifica dei frame costituenti le doppiette dei messaggi inviate, in questo caso, dal PSSu PLC a ciascuno dei dispositivi PSSu I/O è rappresentato in Figura 6.22. Per quanto riguarda la sincronizzazione della trasmissione dei messaggi è rispettata la stessa modalità riscontrata nel caso di analisi precedenti.

CDC SPDO [out=1] MITTENTE: PSSu PLC DESTINATARIO: PSSu I/O 4

H	VER	LGTH		PID		L	SAFE DATA 1				SID 1	C.N		CRC 1		
60	01	00	1A	00	00	0A	1A	67	8F	01	01	00	01	07	FB	36	9C	79

SAFE DATA 2				SID 2		C.N		CRC 2		
67	8F	01	01	00	01	07	FB	36	9C	79

CDC SPDO [out=1] MITTENTE: PSSu PLC DESTINATARIO: PSSu I/O 5

H	VER	LGTH		PID		L	SAFE DATA 1				SID 1	C.N		CRC 1		
60	01	00	1A	00	00	0A	1A	67	8F	01	01	00	01	07	FB	36	9C	79

SAFE DATA 2				SID 2		C.N		CRC 2		
67	8F	01	01	00	01	07	FB	36	9C	79

Figura 6.22: struttura dei messaggi relativi alla trasferimento di ciascun dato *safe*

La struttura dei messaggi inviati a ciascun dispositivo risulta la stessa anche se viene effettuata una distinzione all'interno del campo *safe data*. Il valore contenuto in *data*, il campo successivo al *cons sig* 0x 67 8F, non assume un singolo valore ma bensì due distinti 0x 01 01, uno per ogni mappatura della variabile. Questa tipologia di codifica, che attribuisce allo stesso *cons sig* due valori distinti, si è riscontrato essere utilizzata anche quando due variabili, entrambe *safe* o *standard*, vengono mappate su uscite differenti dello stesso o di più moduli I/O relativi ad un unico dispositivo. In quel caso, il campo *con sig* è utilizzato come identificativo dell'azionamento di due uscite prestabilite in fase di *download* del programma, e i valori contenuti in sequenza nel *data* verranno riferiti rispettivamente a tali uscite.

6.2.3 Conclusioni

Da queste analisi è stato possibile verificare come l'utilizzo degli stessi PID possa essere effettuato da più dispositivi. Si è riscontrata la possibilità da parte di più dispositivi di sottoscrivere, oltre ai messaggi *SHB request* provenienti dal PSSu PLC, anche i messaggi per la trasmissione dei dati ciclici di processo *safe*. Si può affermare, per tanto, che la sottoscrizione di un qualsiasi PID non è esclusivamente permessa ad un prestabilito dispositivo ma, dato che è consentita ad almeno due, può essere effettuata da qualsiasi altro nodo della rete. Tutto ciò fornisce una conferma parziale della possibilità di simulare un nodo aggiuntivo della rete per sottoscrivere i dati di processo per farli inviare anche a tale dispositivo virtuale. Questo argomento viene trattato più nel dettaglio nel capitolo successivo, in cui verrà anche spiegato il contesto in cui viene utilizzata la caratteristica riscontrata con questa analisi.

6.3 Applicazioni reali

Tramite la suite software PAS4000 Pilz mette a disposizione dei blocchi funzionali preprogrammati per l'azionamento di molteplici dispositivi di sicurezza classica (pulsanti di emergenza, barriere fotoelettriche) che produce. Essi sono stati realizzati per fornire dei programmi certificati secondo i canoni dell'implementazione della *safety*. Grazie ad essi è, infatti, possibile utilizzare dei meccanismi aggiuntivi che permettono di rilevare lo stato dei dispositivi classici, mappato su una variabile chiamata *valid bit*. Sulla base del valore di tali variabili è possibile, mediante l'utilizzo di altre variabili fornite dai blocchi funzionali, mettere in sicurezza parte dell'impianto.

Sul sito di Pilz sono presenti diversi esempi sulla programmazione di questi blocchi funzionali, necessari per permetterne l'inserimento in un progetto. In questa sezione si andrà ad analizzare un blocco funzionale specifico, chiamato *emergency stop*, che permette la corretta gestione di un pulsante di sicurezza inserito in un impianto.

I pulsanti di arresto di emergenza sono progettati per un utilizzo industriale pratico e gestiscono la sicurezza di uomini e macchine in situazioni che richiedono un arresto di emergenza. Macchine e impianti, in conformità alla Direttiva Macchine, devono essere dotati di un tale pulsante di arresto per poter evitare un pericolo o ridurne l'entità. I dispositivi di emergenza, in caso di pericolo, vengono azionati manualmente ed inviano un segnale che porta all'interruzione del movimento pericoloso. Il comando di arresto blocca il dispositivo di emergenza fino a che questo non viene sbloccato manualmente. Nel blocco funzionale, infatti, sono presenti diverse variabili che permettono il monitoraggio dell'azionamento del pulsante, collegato ad un ingresso *safe* e dello stato di validità del segnale ad esso relativo.

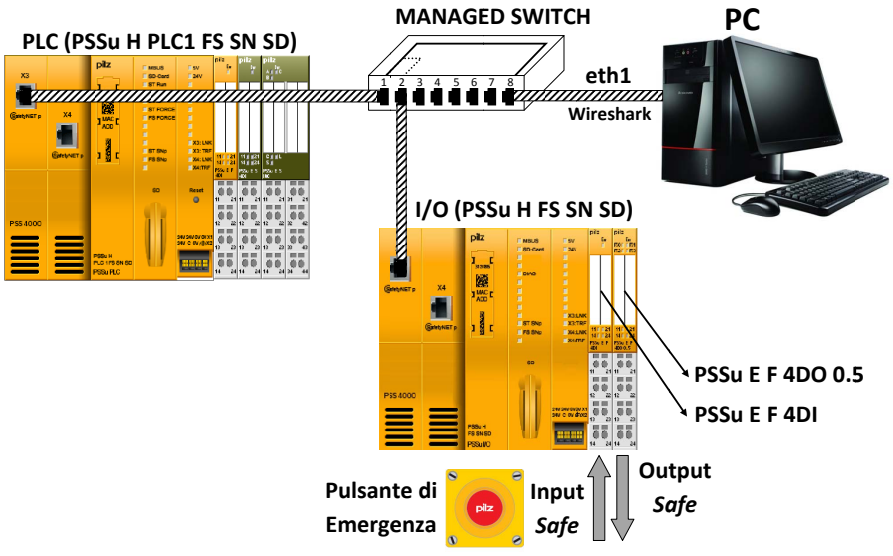


Figura 6.23: struttura del *set up* per le analisi delle *applicazioni reali*

L'obiettivo che si vuole raggiungere con l'analisi di questa comunicazione, che rappresenta un esempio reale molto utilizzato nelle realizzazioni industriali, è ot-

tenere delle informazioni ulteriori che permettano di capire come viene gestita la trasmissione tra i dispositivi, mediante la rete Ethernet SafetyNET p, dei valori degli ingressi e come può essere forzato lo stato *failsafe* per tutte le variabili, sia *standard* che *safe*, che devono essere gestite in seguito all'azionamento, o al verificarsi di un guasto, del pulsante di sicurezza (chiamato anche *fungo*, in tedesco *Pilz*).

La struttura sperimentale della rete utilizzata per realizzare queste analisi è riportata in Figura 6.23 in cui, rispetto a quella utilizzata nelle comunicazioni *semplici*, è stato sostituito nel dispositivo PSSu I/O il modulo I/O di ingresso *PSSu E S 4DI*, che prima non veniva utilizzato, con *PSSu E F 4DI*. Sul PSSu PLC è installato un programma che gestisce, in base al valore assunto da una variabile *safe* di ingresso riferita al pulsante di emergenza, l'attuazione di un'uscita *safe*. Le variabili vengono mappate rispettivamente su un ingresso del modulo I/O *PSSu E F 4DI* e su un'uscita di *PSSu E F 4DO 0.5* appartenenti allo stesso dispositivo *PSSu I/O*.

Tramite Wireshark è possibile analizzare la trasmissione di queste variabili dal *PSSu PLC* al *PSSu I/O* tramite la rete Ethernet SafetyNET p. Alla scheda Ethernet *eth1* del PC, su cui è in ascolto l'analizzatore di rete, vengono indirizzati tutti i frame scambiati fra i due dispositivi mediante la funzione di *port mirroring* dello switch. In quest'ultimo componente di rete è stato configurato il *mirroring* dei dati provenienti e diretti alla porta 1, a cui è connesso il PLC, sulla 8, a cui è connessa la scheda di rete *eth1* del PC.

In Tabella 6.5 sono riportate le attribuzioni MAC-IP per ciascun dispositivo utilizzate in questa configurazione di comunicazione.

Dispositivi	MAC	IP
PSSu PLC	00:02:48:40:3B:67	192.168.1.2
PSSu I/O	00:02:48:40:3A:09	192.168.1.4
eth1	00:04:75:7F:2C:7A	192.168.1.11

Tabella 6.5: associazione MAC/IP dei dispositivi nelle *applicazioni reali*

6.3.1 Emergency Stop

Con questa analisi si vuole dimostrare come i dispositivi PPSu possono essere utilizzati in *applicazioni reali*, e non solo nel semplice trasferimento di onde quadre o costanti utilizzati per lo studio del protocollo; si vuole, inoltre, capire come viene generata e gestita la trasmissione di eventuali segnali di ingresso.

Per questa analisi, facendo riferimento alla configurazione dell'apparato sperimentale di Figura 6.5, è stato scritto, mediante la suite software PAS4000 , un programma chiamato *Ondaquadra_SAFE_EmergencyStop*, che utilizza il blocco funzionale *FS_EmergencyStop*, il quale permette di monitorare il pulsante d'arresto di emergenza e che genera al suo interno un'onda quadra *safe* con periodo pari a $10T_c$ e *duty cycle* del 50%. Il blocco è in grado di rilevare se il pulsante di arresto di emergenza è stato azionato, così come la rilevazione errata del segnale di ingresso ad essa associata.

Il programma utente viene caricato sul PSSu PLC e le variabili di ingresso e di uscita vengono associate rispettivamente ad un ingresso e a un'uscita del modulo *PSSu E F 4DI* e *PSSu E F 4DO 0.5* del dispositivo PSSu I/O. Così, sia il valore dell'ingresso che quello dell'uscita vengono trasferiti, tramite la rete Ethernet SafetyNET p, fra i dispositivi in verso opposto; il PSSu I/O trasmette l'input acquisito al PSSu PLC che, dopo averlo elaborato, gli restituisce il valore dell'output, secondo cui verrà conformemente attivata l'uscita del suo modulo I/O.

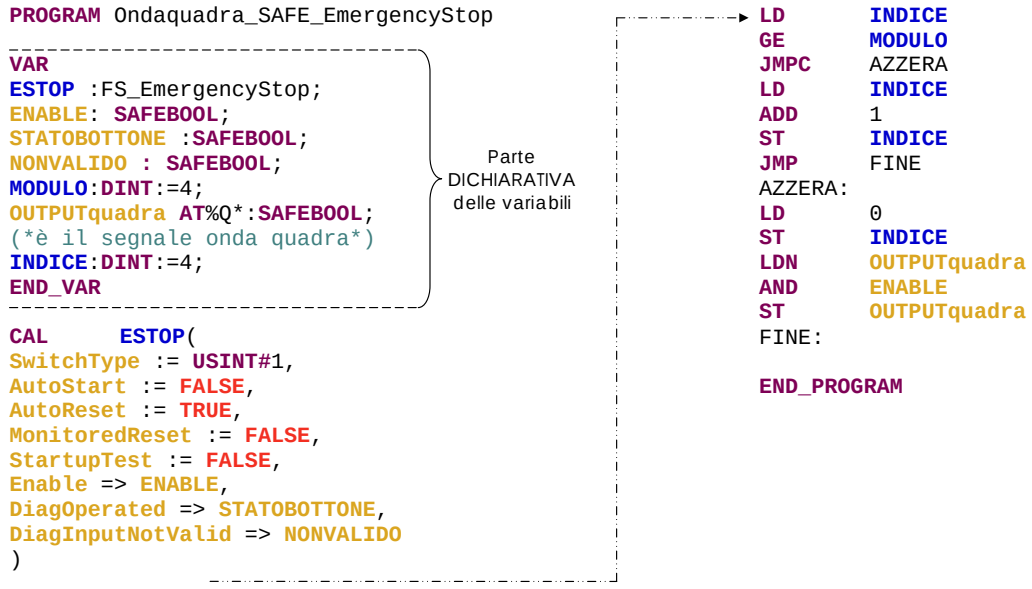


Figura 6.24: codice del programma utente *Ondaquadra_SAFE_EmergencyStop*

Come si può vedere in Figura 6.24, nel blocco funzionale *FS_EmergencyStop* sono presenti diversi parametri che permettono di configurare, in base alle proprie esigenze, l'attuazione del pulsante di emergenza. I possibili parametri di configurazione e i rispettivi significati sono rappresentati in Tabella 6.6. In essa sono, inoltre, descritte le variabili non utilizzate nel caso in esame e i valori di default per ciascuna variabile, che vengono assunti se non espressamente dichiarati.

Parametro	Data Type	Valore	Significato
SwitchType	SAFEUSINT	USINT#1	interruttore con 1 contatti NC
		USINT#3	interruttore con 2 contatti NC
AutoStart	SAFEBOOL	TRUE	reset automatico in fase di start-up
		FALSE	reset dipende da input *Reset*
AutoReset	SAFEBOOL	TRUE	reset automatico in fase di restart
		FALSE	reset dipende da input *Reset*
MonitoredReset	SAFEBOOL	FALSE	fronte di salita di *Reset*
		TRUE	fronte di discesa di *Reset*
StartupTest	SAFEBOOL	FALSE	no funzione test in start-up
		TRUE	richiesta funzione test in start-up
SimultaneityTime	TIME	T#0...3000ms	tempo commutazione dei 2 contatti
		T#100ms	di default

Tabella 6.6: significato dei parametri del blocco funzionale *FS_EmergencyStop*

Oltre a questi parametri, nel blocco funzionale *FS_ EmergencyStop*, è necessario provvedere all'assegnazione delle variabili diagnostiche e di *Enable* alle variabili che vengono effettivamente utilizzate all'interno del programma. In particolare, in questo caso sono utilizzati solamente le variabili *Enable*, che rappresentano il valore di uscita abilitato dal pulsante di emergenza, contemplando anche la presenza di eventuali errori sul dispositivo stesso, *DiagOperated*, che riporta informazioni sullo stato del pulsante, e *DiagInputNotValid* che comunica la validità totale del dispositivo di emergenza sulla base di quella di ciascun suo ingresso. In Tabella 6.7 sono riportate tutte le variabili di diagnostica disponibili nel blocco funzione.

Variabili diagnostiche	**Data Type**	**Valore**	**Significato**
DiagOperated	SAFEBOOL	TRUE	E-STOP premuto
		FALSE	E-STOP premuto
DiagReadyForReset	SAFEBOOL	TRUE	E-STOP non pronto per reset
		FALSE	E-STOP non pronto per reset
DiagReadyForTest	SAFEBOOL	TRUE	richiesta funzione test
		FALSE	non richiesta funzione test
DiagSwitchError	SAFEBOOL	TRUE	1 input commutato oltre SimultaneityTime
		FALSE	2 output commutati oltre SimultaneityTime
InputNC1_Valid	SAFEBOOL	TRUE	segnale valido
		FALSE	segnale invalido
InputNC2_Valid	SAFEBOOL	TRUE	segnale valido
		FALSE	segnale invalido
DiagInputNotValid	SAFEBOOL	TRUE	entrambi ingressi validi
		FALSE	uno degli ingressi invalido

Tabella 6.7: significato delle *variabili diagnostiche* del FB *FS_ EmergencyStop*

L'uscita effettiva del pulsante di emergenza è rappresentata dalla variabile *Enable*, che ne riporta lo stato controllandone in contemporanea sia la pressione che la validità del segnale in ingresso. Essa deve poi essere utilizzata all'interno del programma per scatenare la reazione appropriata che si necessita verificare in caso di pressione o di errore interno sul fungo, in quanto l'evento verificatosi sul dispositivo di emergenza non effettua in automatico nessuna azione sul nodo a cui è connesso. Il dispositivo PSSu non viene portato nello stato *failsafe* perchè potendo comandare più macchinari contemporaneamente, che possono prevedere l'utilizzo di fungo manuale di emergenza ciascuno, con la loro pressione di uno dei pulsanti si desidera bloccare solo la macchina ad esso relativa e non tutte.

Nel programma utilizzato, a *DiagOperated* e *DiagInputNotValid* sono associate rispettivamente le variabili *STATOBOTTONE* e *NONVALIDO* perchè, nella fase preliminare di realizzazione dell'applicazione reale, è stato necessario poter disporre anche di informazioni diagnostiche sullo stato di azionamento del pulsante di emergenza e sulla sua validità. L'*Enable*, invece, è stato utilizzato all'interno del programma per scatenare la reazione appropriata, ossia provvedendo a resettare il valore della variabile *OUTPUTquadra* che ne disabilita la relativa uscita. Nel valore delle variabili diagnostiche può essere determinato il motivo per cui il segnale *Enable*, il cui valore è normalmente a livello logico alto, viene resettato.

Queste non sono le uniche variabili necessarie per l'implementazione del funzionamento di un pulsante di emergenza in un sistema; come si può vedere in Figura

6.25, durante la fase di *mapping* delle variabili del programma utente vengono automaticamente generati gli ingressi *InputNC1* e *InputNC2*, relativi ai segnali di ingresso al fungo, e *Reset*, utilizzato opzionalmente per sincronizzarne il reset e il restart. In questo ambito, il *Reset* non è stato contemplato e, avendo posto il parametro *SwitchType* al valore USINT#1 per indicare che si sta considerando un pulsante ad un solo ingresso, viene utilizzata solo la variabile *InputNC1* per inserire il dispositivo di emergenza.

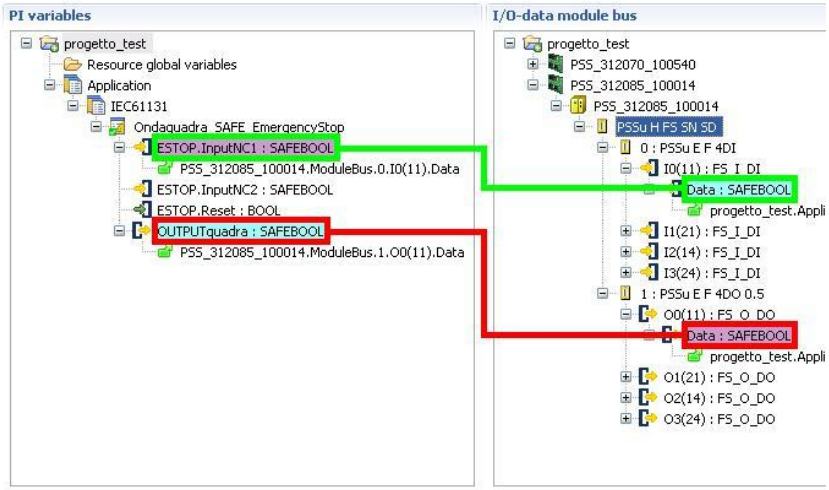


Figura 6.25: *mapping I/O* nel programma *Ondaquadra_SAFE_EmergencyStop*

L'ingresso del modulo I/O *PSSu E F 4DI*, in cui viene mappata la variabile *InputNC1*, è stato impostato, durante la fase di definizione dell'hardware mediante la suite software PAS4000, in modo da ritenere validi solo i segnali a 24V, a cui è sovrapposta un'onda quadra di tipo *trigger di test 0* (*T0*) (si veda la Sezione 5.1.2 a pag. 60). Questo permette di discriminare lo stato di funzionamento del pulsante di sicurezza anche in caso di guasto in quanto, se all'ingesso dell'interruttore avviene un cortocircuito con i 24V dell'alimentazione, il modulo rivela la mancanza del trigger di test e modifica il *valid bit* (*vb*), che riporta la validità dei dati ricevuti.

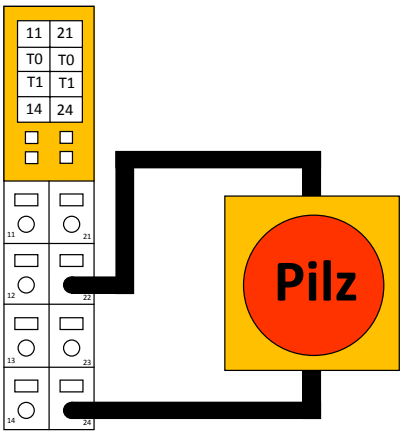


Figura 6.26: schema di collegamento di un pulsante di emergenza

Per poter garantire questa discriminazione il pulsante di emergenza viene normalmente connesso, come rappresentato in Figure 6.26, tra l'ingresso utilizzato dal programma utente e una uscita del *livello di collegamento*, relativo al trigger di test T0, presenti sullo stesso modulo I/O. Il segnale in uscita dal fungo, che è quello in ingresso al modulo I/O, sarà T0 in caso di non abilitazione del dispositivo di emergenza e 24V in caso di cortocircuito.

Dato che, in questa analisi, era interessante poter simulare le varie tipologie di errore possibili sul segnale di ingresso relativo al dispositivo di emergenza, non è stato utilizzato un pulsante ma è stato simulato da una connessione cablata, connessa direttamente al valore di tensione desiderata. E' stato così possibile simulare quattro diverse situazioni, rappresentate in Figura 6.27, tra cui:

A) *pulsante di emergenza non premuto*:

   collegando l'ingresso all'uscita del *livello di collegamento* T0 perchè *InputNC1*, che funziona in logica negata, è normalmente chiuso (NC) e quindi attivo a livello logico basso;

B) *pulsante di emergenza premuto*:

   collegando l'ingresso alla riferimento di massa dell'alimentazione dei dispositivi PSSu per simulare la pressione del fungo;

C) *errata portante T1*:

   collegando l'ingresso all'uscita del *livello di collegamento* T1 si simula un errore di validità perchè la portante T1 ha una fase diversa rispetto a T0;

D) *perdita portante 24V*:

   collegando l'ingresso ai 24V dell'alimentazione dei dispositivi PSSu si simula un errore di validità perchè manca la portante.

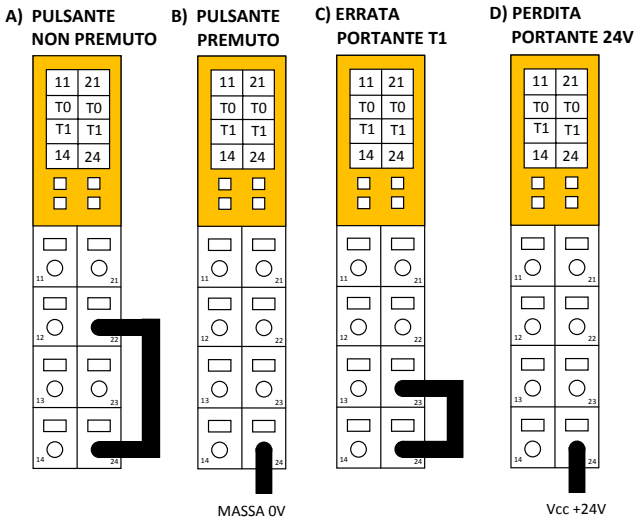


Figura 6.27: simulazioni effettuate per il pulsante di sicurezza

Dalle sniffate è stato possibile comprendere come viene inviato il valore della variabile di ingresso del pulsante di emergenza e il *valid bit* (*vb*) per comunicare la presenza della portante corretta o meno, dal dispositivo PSSu I/O al PSSu PLC.

La parte di *creazione della comunicazione* è analoga a quella puramente *safe* mentre lo *svolgimento della comunicazione* è più complicato, a causa della presenza della trasmissione del valore del segnale di ingresso da parte del PSSu I/O. E' stato possibile estrarre dai messaggi *CDCN subscribe*, che vengono scambiati durante la *creazione della comunicazione*, i PID utilizzati dai dispositivi per inviare le differenti tipologie di messaggi *CDCN PDU* e metterli in relazione con l'informazione trasmessa in ciascuno di essi, grazie all'analisi dello *svolgimento della comunicazione*. L'associazione di ciascun PID con l'informazione contenuta nel messaggio a cui è associato è riportato in Tabella 6.9.

PID	Informazione	Mittente	Destinatario
0x 00 00 0A	dato di processo *safe* (output)	PSSu PLC	PSSu I/O
0x 00 00 14	dato di processo *safe* (input+valid bit)	PSSu PLC	PSSu I/O
0x 00 00 32	*SHB request*	PSSu PLC	PSSu I/O
0x 00 00 3C	*SHB response*	PSSu I/O	PSSu PLC
0x 00 00 1E	*SHB request*	PSSu I/O	PSSu PLC
0x 00 00 28	*SHB response*	PSSu PLC	PSSu I/O

Tabella 6.8: associazione PID e informazione

Dalla Figura 6.28, che riporta il flusso dei messaggi scambiati fra i dispositivi in fase di *svolgimento della comunicazione*, si vede che i frame riguardanti il trasferimento dei dati di ingresso, nel verso da PSSu I/O a PSSu PLC, e di quelli di uscita, in verso opposto, avvengono in doppiette, dato che sono entrambi *safe*, e periodicamente, con periodo pari al tempo ciclo del programma utente, con un breve sfasamento tra i due sensi di comunicazione. Come per tutte le altre comunicazioni di questo capitolo, nel caso in cui ciascun dispositivo debba inviare sia messaggi contenenti i dati di processo che messaggi SHB con la stessa scadenza temporale, li accoda in un unico frame *CDCN PDU*, permettendo così, per entrambi, la consegna entro i termini prestabiliti.

La porzione di flusso dei messaggi riportata in Figura 6.28 viene replicata ciclicamente durante l'intera comunicazione e, per tale motivo, sono stati volutamente omessi i valori delle variabili trasmesse *InputNC1* (*in*), con il *valid bit* (*vb*), e *OUTPUTquadra* (*out*), in quanto i valori del segnale di ingresso possono variare influenzando il valore dell'uscita.

Tra tutti i messaggi trasmessi si procede ora ad analizzare la struttura di quelli contenenti i dati di processo, riferiti all'ingresso fornito dal pulsante di emergenza, scambiati fra il dispositivo PSSu I/O e il PSSu PLC, perchè sono gli unici che, fino ad ora, non erano stati riscontrati nei casi sperimentali. Facendo riferimento alla Figura 6.29, si vede che i messaggi che trasmettono i dati di ingresso *safe* possiedono la stessa struttura di quelli che inviano quelli relativi alle uscite. Il campo *consecutive number* è indipendente dai valori assunti da quello dei messaggi che trasmettono la variabile di uscita *safe* e il *SID* assume il valore 0x 00 02 perchè, in questo senso della comunicazione, il mittente (sender) è il dispositivo PSSu I/O.

Il campo *safe data* può essere suddiviso a sua volta negli ormai noti *con sig* e *data*, con il campo *data* costituito da 2 byte, nonostante trasmetta il valore di

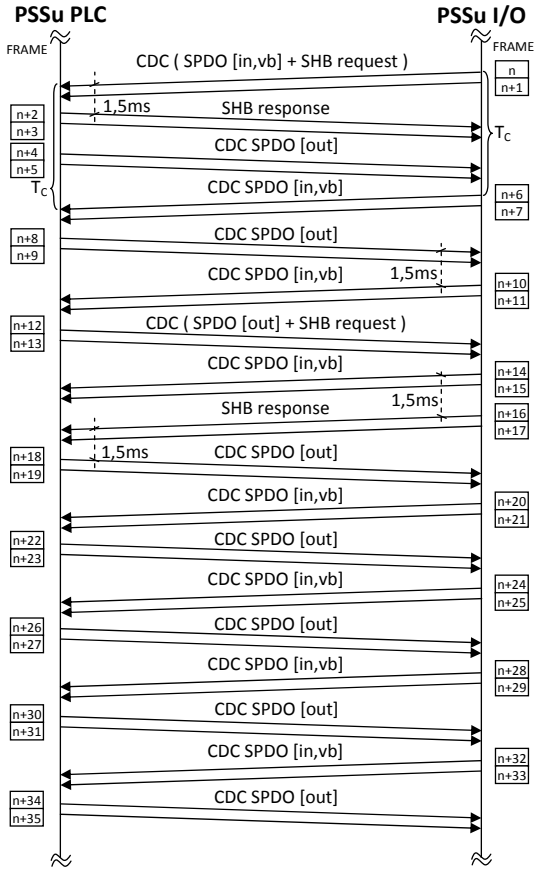


Figura 6.28: struttura della comunicazione nell'applicazione dell'*emergency stop*

un solo ingresso. Il secondo byte, dalle prove sperimentali, si è riscontrato essere l'informazione relativa al valore del *valid bit*, che deve essere trasmessa assieme al valore dell'ingresso per comunicare la validità del dato ricevuto. Questa sofisticazione è stata introdotta in fase di programmazione, configurando l'ingresso del modulo I/O del PSSu I/O relativo al *trigger di test* T0. La scheda deve fornire delle informazioni sulla validità dell'ingresso ricevuto, che vengono così desunte dalla presenza o meno del corretto segnale portante.

Vediamo ora i valori assunti dal campo *data*, dal momento che il *con sig* non varia, nei 4 casi sperimentati di azionamento del pulsante di emergenza effettuati.

Nel caso di *pulsante di emergenza non premuto*, che viene simulato mediante il collegamento diretto tra l'uscita T0 e l'ingresso del modulo I/O, il valore del campo data è `0x 01 01`. Il primo byte indica il valore dell'ingresso, che è a livello logico alto perchè il dispositivo lavora in logica negata e non è, in questo caso, attivato, e il secondo indica il valore del *valid bit*, che è stato settato in quanto il modulo I/O riconosce la portante per cui è stato impostato in ingresso e, quindi, conferma la validità del valore acquisito. Questo valore di ingresso, con la relativa informazione di validità, viene trasmesso al PSSu PLC che provvede all'abilitazione dell'uscita in maniera appropriata. Queste informazioni trasmesse vengono utilizzate, nel programma, per stabilire il valore della variabile *Enable* che è utilizzata per realizzare l'opportuna reazione all'evento verificatosi sul pulsante di emergenza. Qui, non

CDC SPDO [in=1,vb=1] MITT: PSSu I/O DEST: PSSu PLC

H	VER	LGTH		PID		L	SAFE DATA 1				SID 1		C.N		CRC 1		
60	01	00	1A	00	00	14	1A	03	C7	01	01	00	02	15	D8	0C	F6	1A

SAFE DATA 2				SID 2		C.N		CRC 2		
03	C7	01	01	00	02	15	D8	0C	F6	1A

CDC SPDO [in=0,vb=1] MITT: PSSu I/O DEST: PSSu PLC

H	VER	LGTH		PID		L	SAFE DATA 1				SID 1		C.N		CRC 1		
60	01	00	1A	00	00	14	1A	03	C7	00	01	00	02	B7	FD	21	A8	A4

SAFE DATA 2				SID 2		C.N		CRC 2		
03	C7	00	01	00	02	B7	FD	21	A8	A4

CDC SPDO [in=0,vb=0] MITT: PSSu I/O DEST: PSSu PLC

H	VER	LGTH		PID		L	SAFE DATA 1				SID 1		C.N		CRC 1		
60	01	00	1A	00	00	14	1A	03	C7	00	00	00	02	BA	08	F4	8E	10

SAFE DATA 2				SID 2		C.N		CRC 2		
03	C7	00	00	00	02	BA	08	F4	8E	10

Figura 6.29: struttura dei messaggi per il trasferimento del segnale di ingresso

essendo premuto il fungo, non si vuole la messa in sicurezza della variabile di uscita associata al segnale onda quadra. Grazie all'istruzione di AND logico fra la variabile *Enable*, che è verificato essere a livello logico alto, e *OUTPUTquadra*, il valore di quest'ultima viene mantenuto inalterato e azionato sulla rispettiva uscita del modulo I/O del PSSu I/O.

Nel caso di *pulsante di emergenza premuto*, che viene simulato mediante il collegamento diretto tra l'ingresso e il riferimento di massa dell'alimentazione del sistema, il valore del campo data è 0x 00 01. Il primo byte indica il valore dell'ingresso, che è a livello logico basso perchè il dispositivo di emergenza è stato attivato mentre il secondo indica il valore del *valid bit*, che ne conferma la validità in quanto, sia in caso di pressione del pulsante che di guasto, il segnale acquisito sarà a livello logico basso e il comportamento da scatenare in questo caso dovrà essere lo stesso, indipendentemente dal fattore che ne ha causato il cambio di stato. Questi valori, ricevuti dal PSSu PLC, resettano la variabile *Enable* che nel programma provvede a disabilitare l'uscita *OUTPUTquadra* imponendo la variabile a cui è associata a livello logico basso. In questa maniera, in seguito alla pressione del pulsante di emergenza l'uscita viene disabilitata senza però segnalare alcun errore, in quanto il fungo non provvede a portare il dispositivo a cui è collegato nello stato di *failsafe* ma deve essere programmata la reazione appropriata che si desidera ottenere.

Per i casi *errata portante T1* e *perdita portante 24V*, invece, si ha la perdita della portante T0 perchè l'ingresso viene connesso rispettivamente all'uscita T1 del modulo I/O e alla tensione di alimentazione a 24V dei dispositivi PSSu. La codifica è la stessa in entrambi i casi perchè, dal punto di vista logico, hanno lo stesso comportamento: entrambe compromettono la validità del dato ricevuto. La perdita della portante viene tradotta nel valore 0x 00 00 assunto dal campo *data*, in cui sia il valore dell'ingresso che il *valid bit* vengono imposti a valore logico basso per rafforzare la non validità del segnale ricevuto, anche se, seguendo la logica di attribuzione dei segnali, ci si aspettava un valore di 0x 01 00. La determinazione del valore di *Enable* è ottenuta sulla base del *valid bit*, per cui è improbabile che gli stati trasmessi nel campo *data* siano rispettivamente lo stato della variabile *Enable*

e *valid bit*.

In questi casi, a differenza dei due precedenti, la presenza di un errore sul pulsante di emergenza non solo è comunicato attraverso lo stato della variabile diagnostica *DiagInputNotValid* ma viene anche segnalato all'esterno dall'accensione di una luce rossa sul led *Err* del modulo I/O *PSSu E F 4DI*. Inoltre, la presenza di un errore sul modulo I/O viene propagato fino al LED *DIAG*, che diventa rosso, indicando che lo stato del dispositivo non è cambiato e che almeno una parte del sistema è interessata da un errore grave, in quanto rappresenta un errore che deve essere gestito dall'utente. Infatti, l'azionamento di altre uscita sul modulo viene effettuato normalmente e ricollegando il pulsante in posizione corretta, di non abilitazione o errore, la luce del LED diventa istantaneamente verde. L'uscita *OUTPUTquadra*, infine, viene disabilitata dal programma in relazione al valore logico basso assunto dalla variabile *Enable*.

Infine, va notato che durante lo *svolgimento della comunicazione* vengono trasmessi i messaggi *CDC still alive* in entrambi i versi, ciascun dispositivi invia all'altro dei dati di processo (Figura 6.30). Per valutare lo stato della comunicazione e per comunicare la propria operatività ciascun dispositivo manda queste informazioni con una periodicità prefissata di circa 1*sec*.

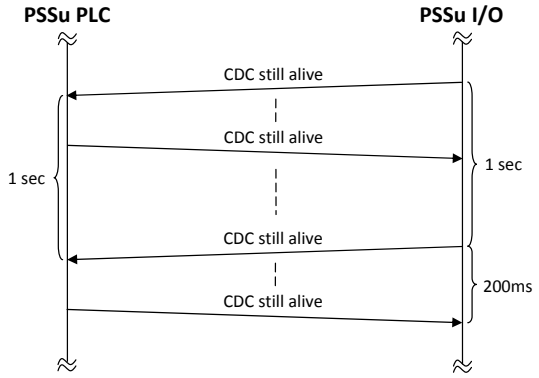


Figura 6.30: tempistiche dei messaggi *CDCN still alive*

6.4 Conclusione

Dalle analisi sperimentali si è verificata l'implementazione della struttura e la tipologia dei messaggi definiti nello standard del protocollo di comunicazione SafetyNET p. Sono state, infatti, esaminate le strutture e la periodicità dei messaggi più importanti all'interno di una comunicazione come i *CDCN subscribe request* e *acknowledge*, gli *SHB request* e *response*, i *CDCN still alive* e i *CDCN PDU* contenenti i dati di processo *standard* o *safe*.

Il risultato di maggior rilevanza è stato comprendere come i PID, elementi discriminanti per identificare l'informazione contenuta nei messaggi ciclici del canale CDC, venissero generati e gestiti all'interno di una generica comunicazione. Si è riscontrato che i PID, nell'implementazione del protocollo effettuata da Pilz, non vengono assegnati sempre alla stessa tipologia di informazione contenuta nei messaggi ma la loro associazione varia di volta in volta. Ad esempio, il PID 0x 00 00 14

è stato utilizzato in casi sperimentali differenti per trasmettere valori di ingressoo di uscita.

Inoltre, per messaggi adibiti a specifici compiti che possono riguardare più destinatari, non sono utilizzati differenti PID ma uno unico per entrambi. Ad esempio, il dispositivo PSSu PLC manda a due nodi PSSu I/O la *SHB request* utilizzando il medesimo PID, così come per trasmettere i valori della stessa variabile ai due destinatari. E', quindi, possibile permette la sottoscrizione dello stesso PID a diversi dispositivi, dimostrando che esso è strettamente legato all'informazione trasmessa nel messaggio e non al suo destinatario. Nonostante ciò, è possibile determinare il numero massimo di PID che possono essere utilizzati in una comunicazione composta da n dispositivi.

Infine, si nota come la generazione dei PID parta sempre dal valore 0x 00 00 0A, incrementandolo di 10, in decimale, per la determinazione dei successivi. In Tabella 6.9 vengono riportati i PID riscontrati nelle varie analisi, effettuate in questa parte sperimentale, e il relativo valore decimale, che permette di comprendere meglio il meccanismo applicato per la loro determinazione.

PID esadecimale	**PID decimale**
0x 00 00 0A	10
0x 00 00 14	20
0x 00 00 28	30
0x 00 00 32	40
0x 00 00 3C	50
0x 00 00 46	60
0x 00 00 50	70
0x 00 00 5A	80

Tabella 6.9: tabella PID riscontrati durante le analisi

Queste considerazioni riscontrate dalle analisi non verranno utilizzate nei successivi capitoli per conseguire lo scopo di questa tesi perchè è una condizione strettamente dipendente dall'implementazione effettuata da Pilz, che potrebbe non essere più verificata nelle successive versioni dei firmware dei dispositivi o nelle implementazioni del protocollo effettuate da altre ditte produttrici di sistemi di comunicazione industriali.

Ad ogni modo, i PID sono un elemento determinante in quanto solo grazie alla loro conoscenza è possibile sottoscrivere, mediante i *CDCN subscribe*, i messaggi generati da un dispositivo, la cui accettazione permette l'effettivo invio dei dati di interesse. Come si vedrà nel capitolo successivo, ciascun dispositivo della rete può confermare la sottoscrizione solo dei PID relativi ai dati che effettivamente produce; richieste di sottoscrizione di PID non afferenti al *publisher* non vengono da lui autorizzate.

CAPITOLO

7

Analisi sperimentale dei meccanismi di protezione del *black channel*

In questo capitolo sono riportate le prove sperimentali con cui sono stati simulati errori di comunicazione sul protocollo di comunicazione SafetyNET p mediante l'iniezione di frame anomali, modificati di volta in volta in base alle diverse esigenze. Allo scopo di generalizzare il processo di induzione degli errori in una sequenza di operazioni da eseguire, sono state analizzate le modalità di reperimento delle informazioni minime necessarie per bloccare la comunicazione dall'esterno.

Nel capitolo precedente è stata analizzata sperimentalmente la struttura dei messaggi, costituita a livello del *black channel*, per la trasmissione dei dati di processo *safe* e la loro doppia ripetizione. In questo capitolo, si vuole verificare se i meccanismi di *detection* presenti nella struttura dei messaggi *safe*, come la ridondanza interna ed esterna, la presenza di un *consecutive number*, di un *SID* e di un *CRC*, siano sufficienti per proteggere la comunicazione dall'iniezione di frame anomali o corrotti all'interno della rete. Dal momento che, in caso di riscontro di errori, il sistema viene portato in uno stato *failsafe*, in cui non è più possibile realizzare il processo produttivo previsto, è necessario che questo avvenga solamente per i casi in cui si realizzi un vero e proprio fallimento. Si vuole verificare, quindi, se sia possibile simulare degli errori sulla rete in cui è in atto una comunicazione *safe*, dapprima mediante la modalità "*man in the middle*" e poi dall'esterno alla rete.

L'utilizzo dell'approccio "*man in the middle*" prevede la collocazione fisica di chi effettua l'inserimento dei messaggi, detto *attaccante*, tra i dispositivi della rete. Si è così in grado di sniffare la trasmissione in atto, che permette di conoscere

esattamente le tipologie di dati trasmessi e, soprattutto, l'assegnazione dei PID all'informazione trasmessa in ciascun messaggio.

La possibilità di bloccare la comunicazione anche dall'esterno della rete è un fattore di primaria importanza per quanto riguarda la *security* del protocollo SafetyNET p, essendo possibile interfacciare la rete di processo con quella aziendale connessa ad Internet. Infatti, idealmente, sarebbe possibile accedere mediante Internet alla rete di comunicazione industriale per compromettere l'*avaiability* del sistema produttivo.

L'obiettivo non è quello di mandare dei messaggi *safe* validi, sia in termini di struttura che di tempistiche, per modificare il comportamento dell'applicazione svolta dal sistema ma si vuole inviarne di errati, modificando i campi che caratterizzano la trasmissione dei dati *safe*, per stimolare i meccanismi di rilevamento (*detection*) degli errori. Nello specifico, vengono inseriti dei messaggi *safe* corrotti in cui verranno alterati il *consecuitive number*, in relazione a quello utilizzato durante la comunicazione nell'esatto istante di inoltro del frame anomalo, o il *CRC*, per una o entrambe le ripetizioni interne del messaggio. Il campo *SID* all'interno dei messaggi anomali, invece, non è stato alterato in quanto la modifica richiedeva il ricalcolo del CRC per garantire l'integrità del dato inviato, che non si è stati in grado di ricalcolare nonostante fosse stato reperito dallo standard del protocollo SafetyNET p il polinomio generatore. Inoltre, per l'invio di questi messaggi corrotti non è stata effettuata la ripetizione del messaggio (due copie identiche inviate con intertempo di $25\mu s$), poichè da prove sperimentali si è verificato che lo strumento software packETH, con il PC a disposizione, non permette di inviare doppiette di frame a distanze inferiori ad $1ms$. Inviando un solo frame della doppietta viene simulato sempre l'errore di ritardo inaccettabile, *unacceptable delay*, ma non viene constatato dal dispositivo in quanto, nell'unico messaggio inviato, sono presenti altri errori che inducono già delle appropriate segnalazioni e procedure di messa in sicurezza di parte del sistema, senza che il dispositivo possa arrivare a verificare la mancanza di messaggi ridondati.

In base al tipo di errore rilevato dai meccanismi di *detection*, implementati per i messaggi *safe* dal *black channel*, il comportamento messo in atto può essere diverso. Il dispositivo può essere portato nello stato di *failsafe*, in cui vengono azzerate solo le uscite associate a variabili *safe*, oppure può essere generato un errore interno, che disabilita anche tutte le uscite *standard*. Poichè mediante lo stato dei LED delle *testate* dei nodi non è possibile discriminare in maniera specifica i differenti effetti degli errori, nelle prove sperimentali si farà riferimento ad una comunicazione mista, in cui è trasmessa fra i dispositivi un'onda quadra *standard* e una costante *safe*, per poterli discriminare agevolmente.

Una volta comprovata la possibilità di indurre errori di comunicazione nella rete, mediante la configurazione "*man in the middle*", si è cercato di stabilire quali tra le informazioni disponibili fossero necessarie per realizzare l'interruzione della trasmissione e le modalità in cui è possibile reperirle senza conoscere il numero e la configurazione di connessione dei dispositivi, il programma utente in esecuzione e la tipologia, il numero e la periodicità dei messaggi trasmessi. Questo permette di svincolarsi dall'approccio "*man in the middle*" e indurre gli errori da un dispositivo esterno alla rete, che non ha nessuna informazione a disposizione sulla rete e sulla comunicazione in atto.

Verrà, infine, inserito un *modello di attacco* in cui è descritta la successione delle operazioni che deve effettuare un *attaccante*, esterno alla rete, per mandare in stato *failsafe* i dispositivi e compromettere così l'*avaiability* del processo produttivo realizzato dal sistema di controllo.

E' doveroso precisare che l'obiettivo della tesi è quello di testare l'efficacia dei meccanismi di *detection* implementati nel protocollo SafetyNET p, dal *black channel*, e come un'eventuale debolezza possa, ad esempio, essere utilizzata durante un attacco esterno per compromettere l'*avaiability* del sistema. Il *modello di attacco*, in quest'ottica, è stato inserito per fornire una controprova al fatto che l'implementazione della *safety* sia robusta a tutte le tipologie di errori verificabili in una rete di comunicazione industriale e per sottolineare come, è possibile compromettere l'*avaiability* di un intero impianto produttivo tramite semplici messaggi con struttura alterata.

7.1 Iniezione di messaggi in una rete SafetyNET p

In questa sezione vengono riportate le prove sperimentali effettuate per simulare degli errori di comunicazione tramite l'inserimento di frame corrotti nella rete. Lo scopo è di verificare la possibilità di indurre degli errori e di capire in che modo si comportano i dispositivi.

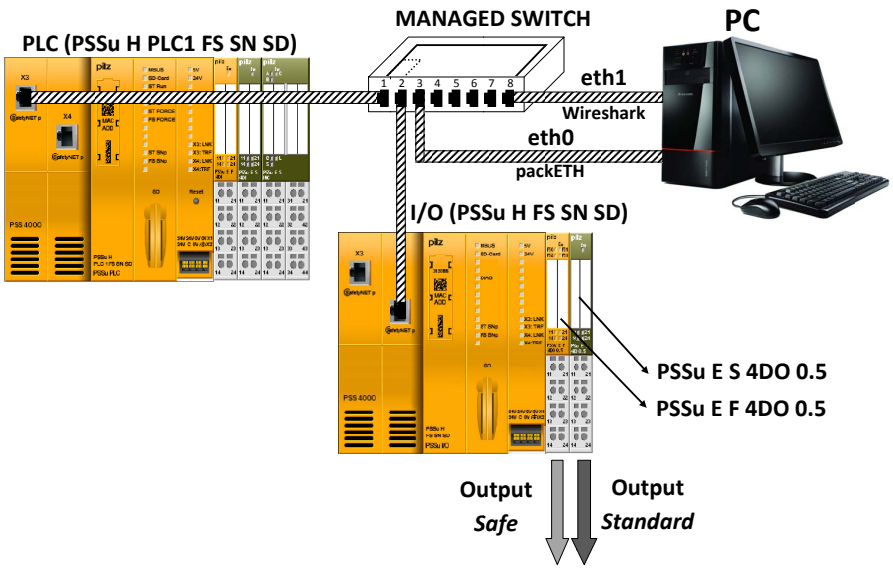


Figura 7.1: struttura del *set up* per la simulazione di errori di comunicazione

In particolare, viene utilizzata la configurazione di rete di Figura 7.1, in cui sono presenti i dispositivi PSSu PLC e PSS I/O, lo switch e il PC con le schede di rete *eth1* e *eth0* rispettivamente utilizzate per monitorare il traffico di rete, tramite l'analizzatore di rete Wireshark, e per iniettare i frame corrotti, mediante il generatore di pacchetti Ethernet packETH. Sul PSSu PLC è installato un programma utente che genera una costante *safe*, *out1*, e un'onda quadra *standard*, *out2*, e trasmette i loro valori, mediante la rete Ethernet SafetyNET p, al nodo PSSu I/O che

provvede all'abilitazione delle uscite dei moduli I/O *PSSu E S 4DO 0.5* e *PSSu E F 4DO 0.5*, a cui sono associate.

Alla scheda Ethernet *eth1* del PC, su cui è in ascolto l'analizzatore di rete, vengono indirizzati tutti i frame scambiati fra i due dispositivi mediante la funzione di *port mirroring* dello switch. In quest'ultimo componente di rete è stato configurato il *mirroring* dei dati provenienti e diretti alla porta 1, a cui è connesso il PLC, sulla 8, a cui è connessa la scheda di rete *eth1* del PC. La scheda di rete *eth0*, invece, è stata connessa con il dispositivo PSSu I/O ad una qualsiasi porta libera, senza alcuna configurazione speciale.

Nello specifico, le prove effettuate prevedono la simulazione di un errore di comunicazione iniettando nella rete un messaggio contenente i dati *safe* sniffati precedentemente tramite l'analizzatore di rete.

Vengono distinti due casi: il caso in cui il frame viene inoltrato senza alterarne il contenuto, per simulare un messaggio con il *consecuitve number* fuori sequenza rispetto a quella di trasmissione, e quello in cui viene modificato indirettamente o direttamente, mediante la manipolazione dei campi *SID* e *consecutive number*, il CRC, per simulare la perdita di integrità del dato trasmesso. Il messaggio, appartenente ad una doppietta per la comunicazione di un dato *safe*, viene iniettato singolarmente perchè non è possibile inviare frame a distanze inferiori ad $1ms$ mediante il generatore di pacchetti Ethernet packETH e perchè, come si dimostrerà durante questa sezione, l'alterazione dei campi *consecutive number* e del *CRC* permette di simulare un errore nella comunicazione che porta i dispositivi nello stato *failsafe* con un unico frame corrotto.

○	LED acceso
◐	LED lampeggiante
●	LED spento

Tabella 7.1: legenda dello stato dei LED

Per verificare lo stato in cui si portano i dispositivi, verranno considerati gli stati dei LED dei due dispositivi e non il flusso dei messaggi dopo l'invio del frame corrotto, in quanto di più immediata comprensione e di maggior significato. Mediante altre prove sperimentali, infatti, si è constatato che dalla analisi della comunicazione risulta più difficile discriminare lo stato di funzionamento dei dispositivi; se uno entra in *failsafe* l'altro continuerà comunque a inviare i messaggi ciclici e, solo dopo la mancata ricezione di un numero prestabilito di *CDCN still alive*, provvederà a rinviare delle *CDCN subscibe request*, che non otterranno risposta.

In Figura 7.2 sono riportati gli stati dei LED dei due dispositivi, secondo la legenda di Figura 7.1, relativi al normale funzionamento della comunicazione in cui nella rete non è stato iniettato alcun frame corrotto.

Tra i LED del dispositivo PSSu PLC, viene riportato solo l'*FS RUN* e non l'*ST RUN*, che riguarda lo stato della risorsa ST, in quanto il programma utilizzato nelle analisi sperimentali gestisce sia variabili *standard* che *safe* e, per questo, viene inserito nella sua risorsa FS, come tutte le POU ST+FS o miste. La scelta di utilizzare un programma misto è stata effettuata per determinare il comportamento manifestato da un dispositivo sulle due tipologie di variabili, appartenenti alla stessa risorsa.

PSSu I/O

LED	Colore		
	-	verde	rosso
MBUS		○	
SDCARD		○	
DIAG		○	
ST SNp		○	
FS SNp		○	
out1 FS		○	
out2 ST		○	

PSSu PLC

LED	Colore		
	-	verde	rosso
MBUS		○	
SDCARD		○	
DIAG		○	
ST SNp		○	
FS SNp		○	
FS RUN		○	

Figura 7.2: stato dei LED dei dispositivi prima dell'iniezione frame corrotto

In questa configurazione di rete, utilizzando la suite PAS4000, sono stati impostati manualmente gli indirizzi IP ai due dispositivi utilizzati, scegliendoli nella stessa sottoclasse. Per le schede di rete *eth1* e *eth0* è stato impostato un indirizzo IP appartenete alla medesima sottorete di quelli utilizzati per i dispositivi PSSu per permettere, sia il caricamento del programma utente che la simulazione di un frame corrotto mandato da un dispositivo dello stesso sistema. In Tabella 7.5 sono riportate le attribuzioni MAC-IP per ciascun dispositivo.

Dispositivi	MAC	IP
PSSu PLC	00:02:48:40:3B:67	192.168.1.2
PSSu I/O	00:02:48:40:3A:09	192.168.1.4
eth0	00:04:75:9E:86:61	192.168.1.13
eth1	00:04:75:7F:2C:7A	192.168.1.11

Tabella 7.2: associazione MAC/IP dei dispositivi della rete di comunicazione

7.1.1 Falsificazione del *consecutive number*

In questa analisi sperimentale si vogliono simulare, mediante la falsificazione del *consecutive number*, gli errori di comunicazione *Unitended repetition* e *incorrect sequence* (si veda la Figura 3.5 di pag. 31), relativi alla ripetizione di un messaggio valido e alla ricezione di messaggi in sequenza errata.

La falsificazione del *consecutive number* può essere simulata iniettando un frame sniffato durante la comunicazione fra i dispositivi. Questo campo è costituito da 1 byte perciò, dato che i valori assunti possono essere 256 replicati ciclicamente durante tutta la comunicazione, non è possibile sapere se il relativo valore contenuto nel frame corrotto sia maggiore o minore rispetto a quello del messaggio che doveva essere consegnato al dispositivo destinatario. In questo modo, se il *consecutive number* è minore al valore assunto nell'ultima doppietta di dati *safe* viene simulata la ripetizione ulteriore di un messaggio valido, se è maggiore, invece, viene simulata la ricezione di messaggi in sequenza sbagliata. In entrambi i casi, comunque, il comportamento messo in atto dal dispositivo è lo stesso perchè le due tipologie di errori fanno riferimento solamente ad una alterazione del campo *consecutive number*, come si può vedere in Figura 3.5 di pag. 31.

Per questa analisi, facendo riferimento alla configurazione dell'apparato sperimentale di Figura 7.1, è stata sniffata la comunicazione del sistema da cui è stato

estratto un unico frame, utilizzato dal PSSu PLC per comunicare al nodo PSSu I/O i dati di processo *safe*. Successivamente, tale frame, la cui struttura è riportata in Figura 7.3, è stato inoltrato nella rete senza che vi siano apportate modifiche al contenuto. L'unica modifica apportata, mediante packETH, è stata l'aggiornamento degli indirizzi IP e MAC del mittente del frame, inserendo quelli relativi alla scheda di rete *eth0* nell'*IP Header*. Questo è stato necessario per evitare che lo switch inoltrasse al PC tutto il traffico destinato al PSSu PLC dopo il cambiamento dell'assegnazione porta-IP nel *forwarding database*. Lo switch, infatti, constata l'arrivo, da due porte, di differenti frame provenienti dallo stesso IP mittente e, quindi, aggiorna il database di assegnazione porta-dispositivo, inserendo come PSSu PLC la scheda di rete del PC, fino ad un successivo aggiornamento della tabella di attribuzione. In questa maniera quello che verrebbe simulato non sarebbe l'inserimento di un frame corrotto nella normale comunicazione ma una deviazione del flusso di messaggi che causerebbe una perdita di pacchetti. In seguito al corretto ripristino dell'instradamento dei frame tra i due dispositivi, la comunicazione non avverrebbe perchè interrotta dalla mancata consegna di un certo numero di messaggi. Queste tecniche si basano su dei principi, noti in letteratura, diversi da quelli che si vogliono simulare in questa tesi e, per questo, non verranno utilizzati.

Il cambiamento del mittente del frame, effettuato nell'*IP Header*, non viene recepito dal dispositivo destinatario, in quanto non esiste nessuna procedura attuata a livelli inferiori all'*application layer*, del modello ISO/OSI, per segnalare l'unico mittente designato all'invio dei messaggi. Infatti, nel dispositivo, ogni livello spacchetta i dati da inoltrare al successivo fino all'*application layer*, a cui arrivano solamente i byte dei messaggi SafetyNET p, in cui non vengono specificati gli indirizzi IP e MAC del mittente.

Frame CDC SPDO corrotto

MITTENTE : scheda eth0 PC DESTINATARIO : PSSu I/O

H	VER	LGTH		PID		L	SAFE DATA1		SID 1		C.N		CRC 1			
60	01	00	18	00	00	14	18	D8	02	01	00	01	24	91	41	70	51

SAFE DATA2		SID 2		C.N.		CRC 2			
D8	02	01	00	01	24	91	41	70	51

Figura 7.3: frame utilizzato per la falsificazione del *consecutive number*

In seguito all'iniezione del singolo frame corrotto, lo stato dei dispositivi è rappresentato mediante i loro LED. Come si può notare in Figura 7.4, sul dispositivo PSSu I/O, che è il destinatario del frame corrotto, viene disabilitata l'uscita *safe* e viene spento il LED *FS SNp*, che segnala la non attività del collegamento FS a SafetyNET p.

In entrambi i dispositivi non viene generato un errore interno ma il PSSu I/O viene portato nello stato *failsafe*, in cui è chiusa la comunicazione FS con la conseguente disabilitazione di tutte le uscite *safe* azionate. L'errore non viene segnalato dal LED *DIAG*, relativo allo stato dell'interno dispositivo, perchè non è considerato grave visto che il funzionamento della parte *standard* del nodo avviene correttamente. Questo può essere un problema, in quanto solo mediante delle procedure di diagnostica remota è possibile accorgersene; mentre dalla constatazione dei so-

PSSu I/O

LED	Colore		
	-	verde	rosso
MBUS		○	
SDCARD		○	
DIAG		○	
ST SNp		○	
FS SNp	●		
out1 FS	●		
out2 ST		○	

PSSu PLC

LED	Colore		
	-	verde	rosso
MBUS		○	
SDCARD		○	
DIAG		○	
ST SNp		○	
FS SNp		○	
FS RUN		○	

Figura 7.4: stato dei LED dei dispositivi dopo iniezione frame corrotto

li LED, non essendo segnalato niente con luci rosse o lampeggianti, può essere frainteso lo spegnimento delle uscite *safe* e di *FS SNp* con il voluto utilizzo del dispositivo solamente per il trattamento di dati *standard*.

Il nodo PSSu PLC continua ad inviare entrambe le tipologie di dati; infatti, da un'analisi del flusso dei messaggi scambiati fra i dispositivi, *FS SNp*, *ST SNp*, e *FS RUN* e dall'osservazione dei LED, si vede che la risorsa FS, in cui è inserita la POU con il programma utilizzato, risulta correttamente avviata e che la trasmissione di entrambi i dati di processo, *standard* e *safe*, avviene indipendentemente dallo stato del dispositivo destinatario. Compete ad esso provvedere all'abilitazione delle tipologie dei dati in base allo stato in cui si trova. In questa simulazione, infatti solo le variabili *standard* sono mantenute attive perchè lo stato di *failsafe* impone solo la chiusura delle uscite *safe*.

Il comportamento del PSSu PLC di inviare in ogni caso tutte le proprie variabili, garantisce che eventuali altri moduli connessi alla rete possano ricevere entrambe le tipologie dei dati di processo sottoscritti. Questo comportamento è stato verificato in una configurazione di rete, in cui dal PSSu PLC erano trasmesse due coppie differenti di uscite *standard* e *safe* a due dispositivi PSSu I/O. Iniettando il frame che provvede alla falsificazione del *consecutive number*, sniffato dalla comunicazione, si è verificato che veniva compromesso solamente il comportamento del dispositivo a cui era destinato, portandolo nello stato *failsafe*, mentre gli altri continuavano a scambiarsi i dati e ad abilitare correttamente entrambe le uscite.

7.1.2 Falsificazione del *CRC*

In questa analisi sperimentale si vuole simulare, mediante la falsificazione del *CRC*, l'errore di comunicazione *Corruption* (si veda la Figura 3.5 di pag. 31), relativo alla corruzione del contenuto di un messaggio.

La falsificazione del *CRC* può essere simulata modificando arbitrariamente uno o più dei campi *SID*, *consecutive number* e lo stesso *CRC*, di un frame sniffato durante la comunicazione fra i dispositivi e iniettandolo nella rete.

Dal momento che la struttura interna del messaggio *safe* è ridondata, verranno distinti due casi: nel primo, chiamato *parziale falsificazione*, verrà modificato, indirettamente o direttamente, soltanto il *CRC* di una delle due ripetizioni mentre nel secondo, chiamato *totale falsificazione* quelli di entrambe.

Frame CDC SPDO corrotto

MITTENTE : scheda eth0 PC DESTINATARIO : PSSu I/O

H	VER	LGTH		PID		L	SAFE DATA1		SID 1	C.N		CRC 1			
60	01	00	18	00	00	14	18	D8	02	01	XX	XX	XX	XX	XX	XX	XX

SAFE DATA2			SID 2	C.N.		CRC 2			
D8	02	01	YY	YY	YY	YY	YY	YY	YY

Figura 7.5: frame utilizzato per la falsificazione del *CRC*

Per questa analisi, facendo riferimento alla configurazione dell'apparato sperimentale di Figura 7.1, è stata sniffata la comunicazione del sistema da cui è stato estratto un unico frame utilizzato dal PSSu PLC per comunicare al nodo PSSu I/O i dati di processo *safe*. Successivamente, tale frame, la cui struttura è riportata in Figura 7.5, è stato modificato in base al caso da simulare e inoltrato nella rete. Mediante packETH, sono stati, inoltre, aggiornati gli indirizzi IP e MAC del mittente del frame, inserendo quelli relativi alla scheda di rete *eth0* nell'*Ethernet Header*.

Totale falsificazione

Il frame sniffato durante lo svolgimento della comunicazione è stato corrotto mediante la modifica del *CRC* relativo a entrambe le repliche, indicate in Figura 7.5 con XX e YY, che sono presenti affiancate all'interno del messaggio *safe*. La modifica di anche uno solo dei campi di queste ripetizioni altera il *CRC* in quanto non è più coerente con i dati su cui è calcolato. Da ulteriori analisi si è provato che il risultato ottenuto modificando singolarmente i campi *SID*, *consecutive number*, *CRC* o combinazioni di essi, è lo stesso in qualsiasi caso.

In seguito all'iniezione del frame corrotto, con i campi modificati in modo che il *CRC* delle due repliche sia errato, lo stato dei dispositivi è rappresentato in Figura 7.6.

PSSu I/O

LED	**Colore**		
	-	verde	rosso
MBUS		○	
SDCARD		○	
DIAG		○	
ST SNp		○	
FS SNp		○	
out1 FS		○	
out2 ST		○	

PSSu PLC

LED	**Colore**		
	-	verde	rosso
MBUS		○	
SDCARD		○	
DIAG		○	
ST SNp		○	
FS SNp		○	
FS RUN		○	

Figura 7.6: stato dei LED dei dispositivi dopo iniezione frame corrotto

Come si può notare, lo stato dei LED e dei dispositivi è restato inalterato rispetto alla normale condizione di comunicazione; le due tipologie di uscite restano attive e i LED di stato non lampeggiano ne vengono spenti, ad indicare che non è stato rilevato alcun errore sulla rete.

Da ulteriori prove sperimentali, qui brevemente descritte, si è constatato come il valore del PID di un messaggio sia un fattore discriminante la validità o meno del frame ricevuto. Ad esempio, se al dispositivo PSSu I/O, che utilizza il PID 0x00 00 14 per ricevere il dato *safe*, viene inviato un frame corrotto per falsificazione del *consecutive number* utilizzando un PID diverso, esso non verrà considerato perchè associato ad un PID che non riguarda i dati da lui sottoscritti. Nel caso in analisi, la presenza di entrambi i *CRC* sbagliati non permette di discriminare in quale dei campi, su cui essi sono calcolati (*PID*, *len*, *safe data*, *SID* e *consecutive number*), si sia verificato l'errore. Potendo essere sbagliato anche il PID, il frame corrotto non viene considerato. Dall'analisi del valore del *consecutive number* dei frame ricevuti immediatamente dopo il dispositivo sarà eventualmente portato nello stato *failsafe* se si riscontrerà di aver tralasciato dei dati destinati a tale nodo. In questo modo, la segnalazione di un errore di *corruption* è tutelata dal *consecutive number*, che provvede a verificare la correttezza della decisione effettuata.

Parziale falsificazione

A differenza del caso precedente, il frame sniffato durante lo svolgimento della comunicazione è stato corrotto mediante la modifica del *CRC* relativo ad una soltanto delle repliche di contromisure che sono affiancate all'interno del messaggio *safe*, indicate in Figura 7.5 con XX e YY. La modifica di anche uno solo dei campi di questa ripetizione altera il *CRC* in quanto non è più coerente con i dati su cui è calcolato. Da ulteriori analisi si è provato che il risultato ottenuto modificando singolarmente i campi *SID*, *consecutive number*, *CRC* o combinazioni di essi, è lo stesso in qualsiasi caso. Questo perchè non è possibile capire quale sia quello alterato ma solo determinare la presenza di un errore sull'integrità dei dati.

Lo stato dei dispositivi in seguito all'iniezione del frame corrotto, con i campi modificati in modo che il *CRC* di una sola delle due repliche sia errato, è rappresentato in Figura 7.7.

PSSu I/O

LED	**Colore**		
	-	verde	rosso
MBUS			◐
SDCARD		○	
DIAG			◐
ST SNp		○	
FS SNp			◐
out1 FS	●		
out2 ST	●		

PSSu PLC

LED	**Colore**		
	-	verde	rosso
MBUS		○	
SDCARD		○	
DIAG			◐
ST SNp		○	
FS SNp		○	
FS RUN		○	

Figura 7.7: stato dei LED dei dispositivi dopo iniezione frame corrotto

A differenza del caso di falsificazione del *consecutive number*, in entrambi i dispositivi viene generato un errore interno che disabilita non solo le uscite *safe* ma anche quelle *standard*. Questo errore interno è espresso dallo stato rosso lampeggiante dei LED *MBUS*, *DIAG* e *FS SNp*. Essi indicano rispettivamente la presenza di un grave errore in stato di stop nel bus del modulo FS, che almeno una parte

del sistema FS è interessata da un messaggio di errore grave e che il collegamento FS a SafetyNET p ha un grave errore FS e ST ed è in stato di stop.

Il dispositivo PSSu I/O, che presenta tutti questi LED rossi lampeggianti, è stato portato nello stato di stop mentre il PSSu PLC è mantenuto nello stato operativo perchè i LED *FS RUN*, *ST RUN* e *MBUS* sono verdi ed è alterato solo il *DIAG* per comunicare che l'invio dei dati non può essere correttamente effettuato, dato che uno dei dispositivi della rete è in stop.

In base alla replica in cui viene falsificato il *CRC*, i LED dei dispositivi impiegano un tempo differente per cambiare lo stato del *DIAG* e *FS SNp* mentre nel nodo PSSu I/O, appena inviato il frame, l'*MBUS* diventa subito rosso lampeggiante ed entrambe le uscite vengono disabilitate. In particolare, nel caso in cui vengano modificati i valori della prima doppietta, contrassegnata con XX, si è verificato che i LED impiegano circa 2 minuti per cambiare di stato mentre modificando quelli contrassegnati con YY impiegano circa 20 secondi. Il risultato fondamentale è che, in entrambi i casi, vengano spente tutte le tipologie di uscite mediante la disabilitazione dell'MBUS del dispositivo PSSu I/O.

7.1.3 Invio messaggio *standard* con PID relativo ad un *safe*

In questo ambito è stata eseguita anche un'analisi sperimentale volta ad indurre un errore nella comunicazione mediante l'inoltro nella rete di un frame corrotto con la struttura più semplice, che potesse ottenere gli stessi effetti ottenuti con il caso della falsificazione del *CRC*. In questo modo, la creazione del messaggio anomalo verrebbe semplificata e la quantità di byte necessari da inviare sulla rete sarebbe maggiormente contenuta.

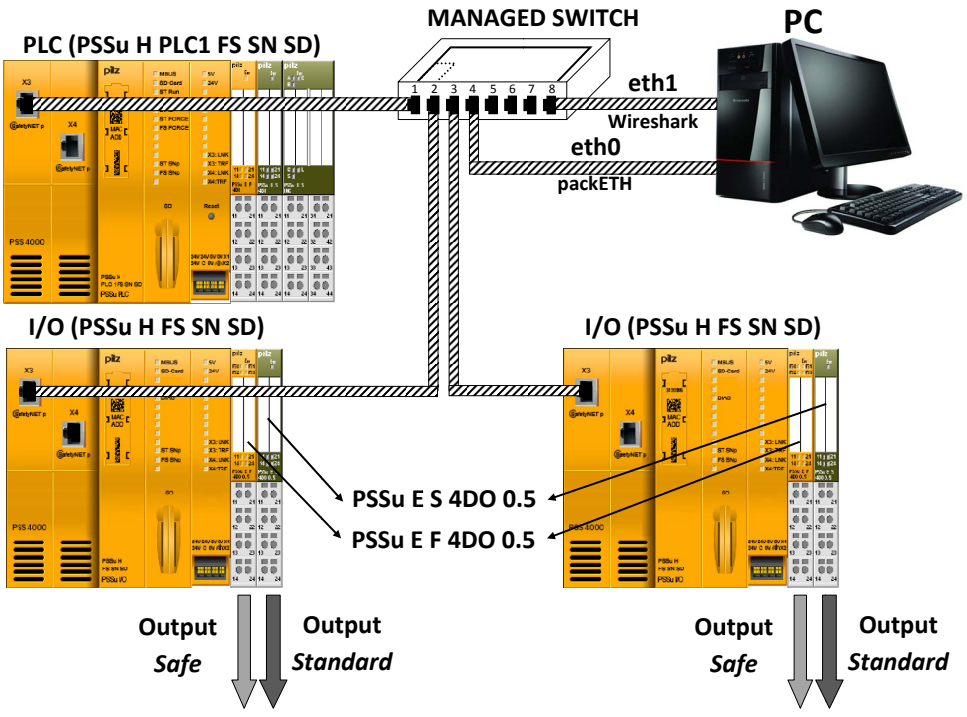


Figura 7.8: struttura del *set up* per la simulazione di errori di comunicazione

La struttura del frame corrotto ricalca la struttura di un messaggio per la trasmissione di dati *standard* solo con il valore del PID relativo alle informazioni *safe* e il campo *standard data* riempito con valori casuali, come riportato in Figura 7.9, il cui numero totale di byte sia congruente a quello contenuto nel campo *len*.

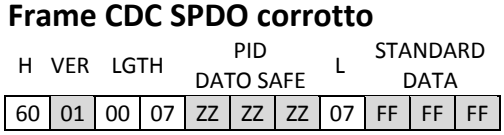


Figura 7.9: frame utilizzato per indurre errori di comunicazione

In questa analisi è utilizzata la configurazione di rete di Figura 7.8 in cui, rispetto alla precedente, è stato inserito un ulteriore dispositivo PSSu I/O che riceve dal PSSu PLC, mediante la rete Ethernet SafetyNET p, il valore di altre variabili *standard*, *out3*, e *safe*, *out4*, e che provvede ad attuare le uscite dei moduli I/O *PSSu E S 4DO 0.5* e *PSSu E F 4DO 0.5*, a cui sono associate.

In Tabella 7.5 sono riportate le attribuzioni MAC-IP per ciascun dispositivo.

Dispositivi	MAC	IP
PSSu PLC	00:02:48:40:3B:67	192.168.1.2
PSSu I/O 4	00:02:48:40:3A:09	192.168.1.4
PSSu I/O 5	00:02:48:40:4F:9F	192.168.1.5
eth0	00:04:75:9E:86:61	192.168.1.13
eth1	00:04:75:7F:2C:7A	192.168.1.11

Tabella 7.3: associazione MAC/IP dei dispositivi della rete di comunicazione

Si proverà ad inviare il messaggio anomalo, in due differenti casi: ad uno dei dispositivi PSSu I/O e al PSSu PLC, per verificare l'entità delle conseguenze al verificarsi di un errore di comunicazione e come queste si possano ripercuotere sulla rete. La presenza del secondo nodo PSSu I/O permette di discriminare come il blocco di un dispositivo possa influire sulla comunicazione che intercorre tra gli altri. Per questo, nel messaggio verranno inseriti i PID validi per i destinatari della trasmissione di informazioni *safe*, siano essi messaggi ciclici *CDCN PDU* oppure messaggi *SHB*. In qualsiasi caso, anche se non utilizzati per la trasmissione di dati di processo, i messaggi *SHB* possiedono la stessa struttura ridondata classica della tipologia *safe* e, quindi, possono essere indotti gli stessi errori di comunicazione. Nello specifico, i valori dei PID 0x ZZ ZZ ZZ utilizzati nei frame corrotti, inviati dalla scheda di rete *eth0* del PC, saranno 0x 00 00 1E e 0x 00 00 28 per quelli destinati rispettivamente al PSSu I/O 4 e al PSSu I/O 5, che sono relativi alla ricezione dei dati di processo *safe*, e 0x 00 00 46 per quello destinato al PSSu PLC, che sono utilizzati per ricevere gli *SHB request* spediti da uno degli altri nodi. In quest'ultimo caso, è indifferente la conoscenza del dispositivo che invia il messaggio SHB associato a tale PID, in quanto l'errore indotto sulla comunicazione e, quindi, il comportamento della rete sarà il medesimo in ciascun caso.

I risultati ottenuti mediante queste analisi sperimentali vengono presentati uno dopo l'altro per poter svolgere agevolmente dei paragoni fra i comportamenti ottenuti.

Gli stati dei LED dei dispositivi della rete in seguito all'invio del frame anomalo dalla scheda *eth0* del PC al nodo PSSu I/O 4, sono riportati in Figura 7.10 mentre quelli causati dalla trasmissione del messaggio corrotto al PSSu PLC sono riportati in Figura 7.11. Entrambe le simulazioni sono state effettuate ogni volta iniettando un unico frame corrotto nella rete e provvedendo, fra ogni prova, a resettare l'alimentazione dei dispositivi per eliminare l'errore di comunicazione e riportarli nello stato operativo

PSSu I/O 4 **PSSu I/O 5**

LED	**Colore**			**Colore**		
	-	verde	rosso	-	verde	rosso
MBUS			○		○	
SDCARD		○			○	
DIAG			◐		○	
ST SNp		○			○	
FS SNp			◐		○	
output FS	●				○	
output ST	●				○	

PSSu PLC

LED	**Colore**		
	-	verde	rosso
MBUS		○	
SDCARD		○	
DIAG			○
ST SNp		○	
FS SNp		○	
FS RUN		○	

Figura 7.10: stato dei LED dei dispositivi dopo iniezione frame corrotto

PSSu I/O 4 **PSSu I/O 5**

LED	**Colore**			**Colore**		
	-	verde	rosso	-	verde	rosso
MBUS		○			○	
SDCARD		○			○	
DIAG			○			○
ST SNp	●			●		
FS SNp	●			●		
output FS	●			●		
output ST	●			●		

PSSu PLC

LED	**Colore**		
	-	verde	rosso
MBUS	●		
SDCARD		○	
DIAG			◐
ST SNp		○	
FS SNp			◐
FS RUN	●		

Figura 7.11: stato dei LED dei dispositivi dopo iniezione frame corrotto

Dal confronto tra gli stati dei LED si può vedere come il comportamento nei due casi sia differente: nel primo viene generato un errore interno solamente al dispositivo PSSu I/O 4 che ne disabilita entrambe le uscite, e il suo verificarsi viene riportato al PLC, mentre nel secondo viene generato un errore sul PSSu PLC, che non potendo più comunicare i dati di processo agli altri dispositivi, ne propaga l'errore.

Nel caso di invio del frame corrotto al dispositivo PSSu I/O, lo stato dei LED è il medesimo del caso di falsificazione parziale del *CRC* tranne per la luce rossa relativa all'*MBUS*, che in questo caso non lampeggia e indica l'impossibilità, da parte del nodo, di raggiungere almeno un modulo I/O. Questo comportamento in apparenza differente è, invece, da considerarsi identico al caso precedente in quanto la disabilitazione che si effettua su entrambe le tipologie di uscite, avviene mediante l'invio di un unico frame.

Nel caso di invio del frame corrotto al dispositivo PSSu PLC, a differenza del PSSu I/O del caso di falsificazione parziale del *CRC*, i LED relativi all'*MBUS* e

all'*FS RUN*, che non è presente nella testata *PSSu H FS SN SD*, vengono spenti, per indicare rispettivamente che nessun modulo è disponibile e che la risorsa FS non è attiva. Il caso diventa più complesso del precedente perchè, dato che la risorsa FS non è attiva, il programma utente contenutovi non può essere eseguito; ciò comporta la generazione di un errore interno ai dispositivi PSSu I/O dovuti dalla mancata consegna dei dati di processo. Questi rappresentano l'errore di comunicazione *unacceptable delay* in quanto i dispositivi aspettano dei dati che non arriveranno mai. In questi nodi il LED *MBUS* viene acceso di rosso, per indicare che non è stato possibile raggiungere almeno un modulo I/O, entrambe le tipologie di uscite vengono spente e anche i LED di stato per *ST SNp* e l'*FS SNp*, indicano che entrambi i collegamenti ST e FS a SafetyNET p non sono più attivi.

Mediante l'analisi sperimentale qui effettuata si verifica che il dispositivo PSSu I/O riceve il frame corrotto con il PID relativo ad un messaggio con struttura *safe* e, non riuscendo a rilevare la ridondanza della struttura, va in errore nella stessa modalità della falsificazione del *CRC*, ossia disabilitando entrambe le tipologie di uscite da lui azionate. Se poi il messaggio corrotto viene mandato al PSSu PLC, cioè in generale al dispositivo che contiene il programma utente da eseguire, l'impatto che si può ottenere sulla rete sarà decisamente più esteso in quanto bloccando la comunicazione effettuata da tale nodo, vengono disabilitate entrambe le tipologie di uscite di tutti i dispositivi che ricevono tali valori.

7.2 Problemi di *avaiability*

Mediante le analisi sperimentali della sezione precedente si è verificato il corretto comportamento dei dispositivi di una rete basata sul protocollo SafetyNET p, in seguito alla simulazione di errori di comunicazione effettuati alterando l'infrastruttura aggiuntiva, implementata dal *black channel*, dei messaggi per la trasmissione dei dati *safe*.

I test effettuati sulla comunicazione, mediante i metodo di analisi "*man in the middle*", rappresentano anche una serie di operazioni che potrebbe essere sfruttata per realizzare un attacco alla rete, dato che l'*avaiability* del sistema viene irreparabilmente compromessa mediante l'invio di un singolo frame corrotto.

Dal momento che una delle caratteristiche principali del protocollo di comunicazione SafetyNET p è la compatibilità con lo *standard Ethernet*, è garantito un semplice interfacciamento tra la rete di processo e quella aziendale connessa ad Internet, che espone a rischio di attacco anche la linea produttiva.

Un eventuale attaccante, con scopi dannosi nel confronto della azienda, potrebbe provocare un blocco della produzione in seguito all'induzione di errori della comunicazione mediante la stimolazione dei meccanismi di *detection* effettuata con l'iniezione nella rete di processo di un singolo frame corrotto. Uno o più dispositivi della rete verrebbero bloccati completamente, necessitando di un reset dell'alimentazione per eliminare gli errori interni; ciò causerebbe una considerevole compromissione dell'*avaiability* del sistema e una grave perdita economica per la mancata produzione.

Essendo un aspetto rilevante, anche se subordinato alla garanzia della *safety* nell'impianto, viene ora simulata la possibilità di indurre gli errori riportati nel

caso precedente, spostando l'analisi da un approccio "*man in the middle*" in uno di attacco dall'esterno della rete. Per questo motivo, verranno verificate le possibilità di inviare i frame corrotti anche da un dispositivo esterno alla rete e di raccogliere i dati essenziali per poter indurre un errore sulla comunicazione, quali i PID e gli indirizzi MAC e IP dei nodi della rete.

7.2.1 Informazioni minime

La configurazione di rete utilizzata per queste analisi sperimentali, rappresentata in Figura 7.12, coinvolge un PSSu PLC in cui è presente un programma che genera un'onda quadra *safe* e la trasmette, tramite la rete Ethernet SafetyNET p, ad un dispositivo PSSu I/O, che provvede all'azionamento di un'uscita del suo modulo I/O *PSSu E F 4DO 0.5*.

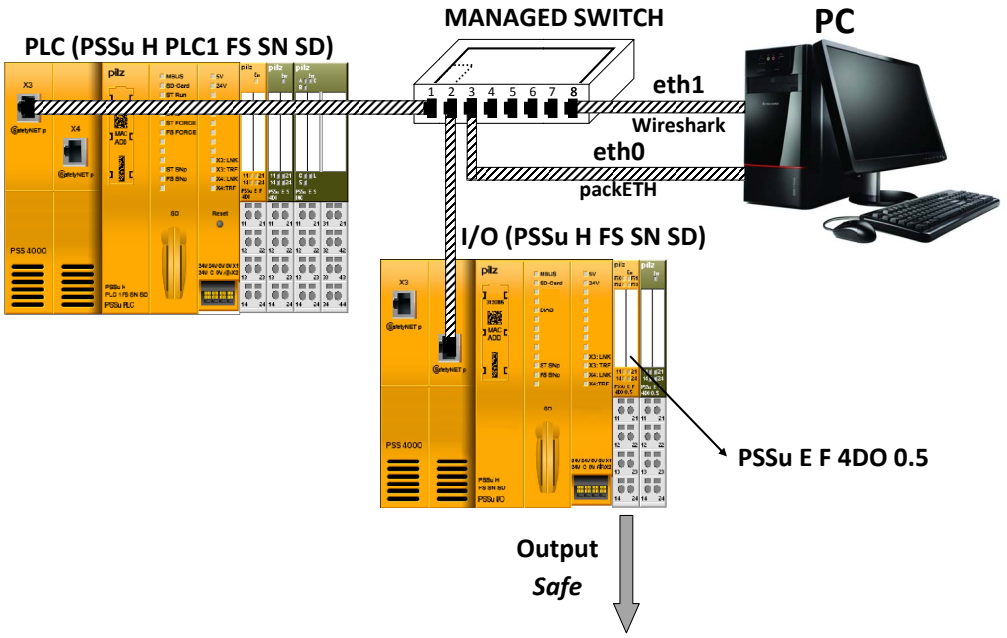


Figura 7.12: struttura del *set up* per il reperimento delle informazioni necessarie per il blocco della comunicazione

Questa struttura di rete è stata utilizzata per semplificare la trattazione e il reperimento delle informazioni, in quanto il numero di PID utilizzati sono in un numero contenuto. Mediante l'analizzatore di rete Wireshark è stato possibile reperire, durante la fase di *creazione della comunicazione*, i PID utilizzati da entrambi i dispositivi per inviare i messaggi ciclici contenenti i dati di processo e i messaggi SHB, che vengono riportati in Tabella 7.4.

Dispositivi	**PID**		
	Dati *safe*	**SHB request**	**SHB responce**
PSSu PLC	0x 00 00 0A	0x 00 00 14	0x 00 00 32
PSSu I/O	- - -	0x 00 00 28	0x 00 00 1E

Tabella 7.4: stato dei LED dei dispositivi dopo iniezione frame corrotto

In Tabella 7.5 sono riportate le attribuzioni MAC-IP per ciascun dispositivo.

Dispositivi	MAC	IP
PSSu PLC	00:02:48:40:3B:67	192.168.1.2
PSSu I/O 4	00:02:48:40:3A:09	192.168.1.4
PSSu I/O 5	00:02:48:40:4F:9F	192.168.1.5
eth0	00:04:75:9E:86:61	173.22.1.5
eth1	00:04:75:7F:2C:7A	192.168.1.11

Tabella 7.5: associazione MAC/IP dei dispositivi nella configurazione di analisi

In particolare, alla scheda di rete *eth0*, utilizzata per iniettare i messaggi anomali, è stato cambiato indirizzo IP per dimostrare la possibilità di iniettare dei frame corrotti tramite un dispositivo al di fuori della rete di processo. Infatti, è stato verificato come sia possibile interagire con i dispositivi della rete. Possono, quindi, essere generalizzati i discorsi fatti in precedenza sull'induzione di errori di comunicazione nella rete, mediante l'invio di frame corrotti, anche da un dispositivo esterno

Una volta stabilita la possibilità di iniettare traffico anomalo malevolo dall'esterno è necessario capire come possono essere reperite le informazioni necessarie per costituire i messaggi corrotti, come indirizzi MAC e IP del destinatario e PID corretti, in quanto non è possibile connettersi fisicamente alla rete per sniffare il traffico scambiato fra i dispositivi.

In una situazione reale non è possibile avere a disposizione il flusso dei dati trasmessi, così diventa necessario provvedere a stimolare delle procedure che permettano di ricevere dei messaggi contenenti informazioni direttamente al dispositivo utilizzato per effettuare l'attacco, che in questo caso è rappresentato dalla scheda di rete *eth0*.

Reperimento indirizzi MAC e IP

Per il reperimento delle informazioni riguardanti gli indirizzi MAC e IP dei dispositivi, si è pensato di inviare dalla scheda di rete *eth0* un frame che simula un messaggio *RTFN scan network read request* (*RTFNSNR request*), costituito dalla presenza del solo campo *header* di un byte imposto pari a 0x80, da un dispositivo esterno ad uno qualsiasi di quelli della rete. Come si può vedere dalla struttura del frame di risposta *RTFNSNR response*, rappresentata in Figura 4.3 di pag. 38, il dispositivo comunica diverse informazioni, tra cui sono presenti, nel campo *IPv4 address* e *Device MAC*, rispettivamente il suo l'indirizzo IP e MAC.

Dal punto di vista logico, questa modalità di analisi può sembrare un controsenso in quanto, per sapere gli indirizzi MAC e IP di un dispositivo, gli si indirizza un frame, che però suppone la conoscenza di quello che sarebbe l'incognita. Così non è, perchè si è visto come il dispositivo ricevente la *RTFNSNR request* risponda con una *RTFNSNR response* non all'indirizzo IP del mittente ma in *multicast*, il cui IP e MAC sono rispettivamente 224.0.23.61 e 01:00:5E:00:17:3D. Il traffico *multicast* viene gestito attraverso il protocollo *IGMPv1* (*Internet Group MAnagment Protocol*) che istituisce un meccanismo di comunicazione *publisher-subscriber*, in cui tutti i dispositivi che hanno sottoscritto l'iscrizione al gruppo ricevono i pacchetti. Tutti

dispositivi PSSu, appartenenti al sistema PSS4000, sottoscrivono in ogni comunicazione l'IP 224.0.23.61, che è stato registrato il 13 marzo 2009 nel *IPv4 multicast address space registry* dell'organizzazione internazionale IANA come indirizzo IP *multicast* relativo al protocollo SafetyNET p di Pilz.

Si è provato, quindi, a trasmettere il messaggio *RTFNSNR request* in *multicast* per simulare una situazione reale, in cui non è possibile sapere già in anticipo gli indirizzi MAC e IP dei dispositivi della rete. Si è così verificato sperimentalmente che, come si può vedere dallo *screenshot* della schermata di Wiresharkriportato in Figura 7.13, mediante l'invio di un unica *RTFNSNR request* in *multicast*, ciascun dispositivo invia sempre in *multicast* la propria *RTFNSNR response*, evidenziata in rosso, in cui sono contenuti i relativi indirizzi MAC e IP, contrassegnati in verde.

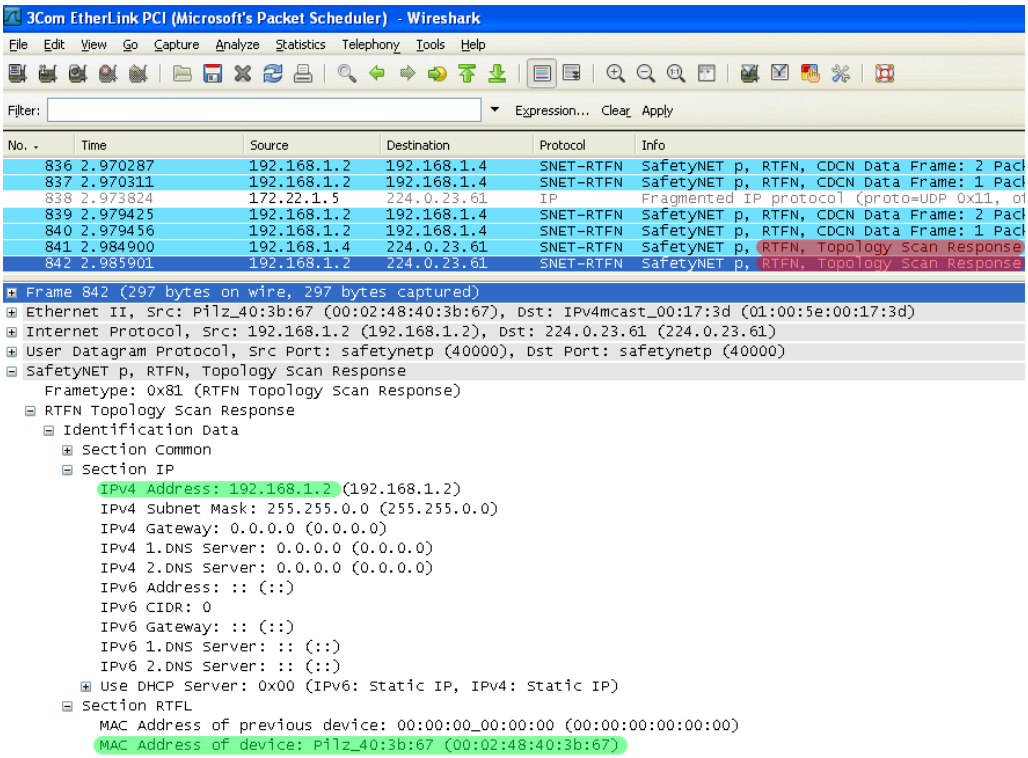


Figura 7.13: *screeshot* di una schermata di Wireshark

Per riceve i dati inviati in *multicast* dai dispositivi, anche quello simulato per l'invio della richiesta di scansione deve essere iscritto allo stesso gruppo IGMP. A tal fine, è sufficiente inviare preventivamente un messaggio *IGMPv1 membership report*, la cui struttura è rappresentata in Figura 7.14, con il quale il dispositivo richiede la sottoscrizione al gruppo associato all'indirizzo *multicast* 224.0.23.61.

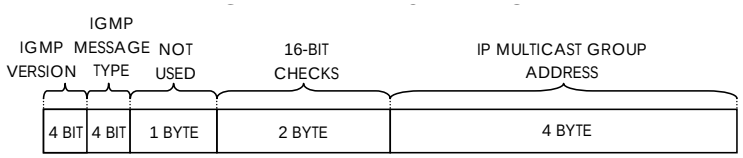
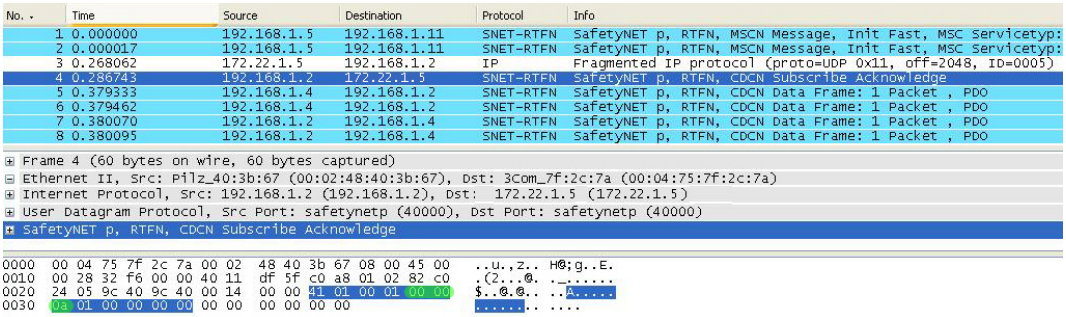


Figura 7.14: struttura di un messaggio *IGMPv1 membership report*

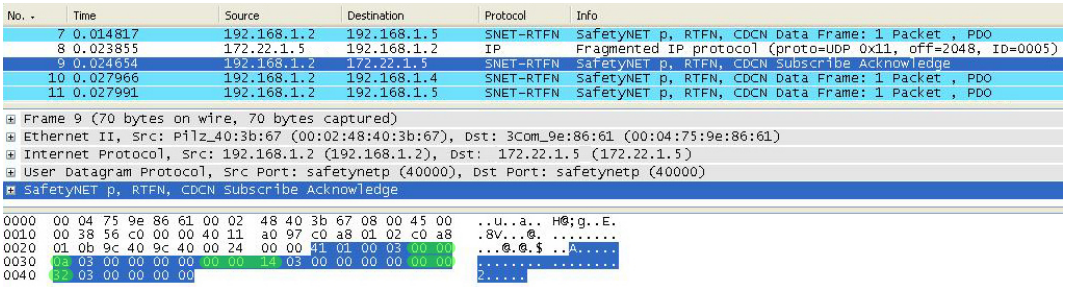
Nel caso specifico, per l'iscrizione al gruppo *multicast* viene spedito un messaggio *IGMPv1 membership report* con indirizzo IP e MAC del destinatario rispettivamente pari a 224.0.23.61 e 01:00:5E:00:17:3D, in cui il valore dei byte è 0x12 00 F6 C1 E0 00 17 3D. In questo modo, i messaggi *multicast* verranno inoltrati, da tutti i dispositivi della rete, anche al dispositivo attaccante dopo l'invio della *RTFNSNR request*.

Reperimento PID

Gli unici messaggi che, essendo confermati, permettono di reperire il valore dei PID utilizzati in ciascun dispositivo sono i *CDCN subscribe*. In Figura 7.15(a) e Figura 7.15(b) sono rappresentati gli *screenshot* di Wireshark di due differenti analisi, in cui sono mandati dalla scheda *eth0* al PSSu PLC, durante la comunicazione, rispettivamente un messaggio *CDCN subscribe request* contente tutti gli effettivi PID utilizzati dal destinatario, 0x 00 00 0A, 0x 00 00 14 e 0x 00 00 32, e uno che ne contiene solo uno di corretto, 0x 00 00 0A.



(a) *CDCN subscribe acknowledge* all'unico PID corretto



(b) *CDCN subscribe acknowledge* ai 3 PID corretti

Figura 7.15: *screeshot* di schermate di Wireshark

Da queste simulazioni si vede come ciascun dispositivo sia in grado di ricevere i messaggi *CDCN subscribe request* durante la comunicazione, anche da parte di un nodo non appartenente alla rete. La risposta *CDCN subscribe acknowledge*, inviata al mittente della richiesta, fornisce conferma di sottoscrizione solamente dei PID validi che il dispositivo utilizza per trasmettere i dati ciclici. Nelle figure tali PID sono contrassegnati in verde.

Mediante altre simulazioni è stato verificato che la lunghezza del messaggio *CDCN subscribe request*, in cui vengono inseriti i PID di cui si vuole richiedere la sottoscrizione, non è un fattore determinante per ricevere risposta; può essere

spedito un numero di PID anche nettamente superiore a quelli realmente utilizzati e ciò consente, comunque, di avere conferma di tutti quelli corretti all'interno del messaggio.

Sulla base di queste analisi e di quelle effettuate nel punto precedente, si è pensato di creare un messaggio *CDCN subscribe request* che contenesse la richiesta di sottoscrizione a tutti i 5 PID, utilizzati per lo scambio dei dati ciclici tra i nodi, e inviarlo nella rete, simulando la trasmissione da parte di un dispositivo esterno, in *multicast* con l'indirizzo IP 224.0.23.61 e MAC 01:00:5E:00:17:3D.

In Figura 7.16 è mostrato quello che accade in seguito alla richiesta di sottoscrizione. Nello specifico, al frame 276 viene spedita in *multicast* la *CDCN subscribe request*, nei frame 278 e 279 ciascun dispositivo risponde al mittente con una *CDCN subscribe acknowledge* sottoscrivendo i PID ad esso relativi e, dal frame 282, si vede che, avendo sottoscritto il PID dei dati *safe*, vengono inviati dal PSSu PLC all'attaccante esterno i messaggi richiesti.

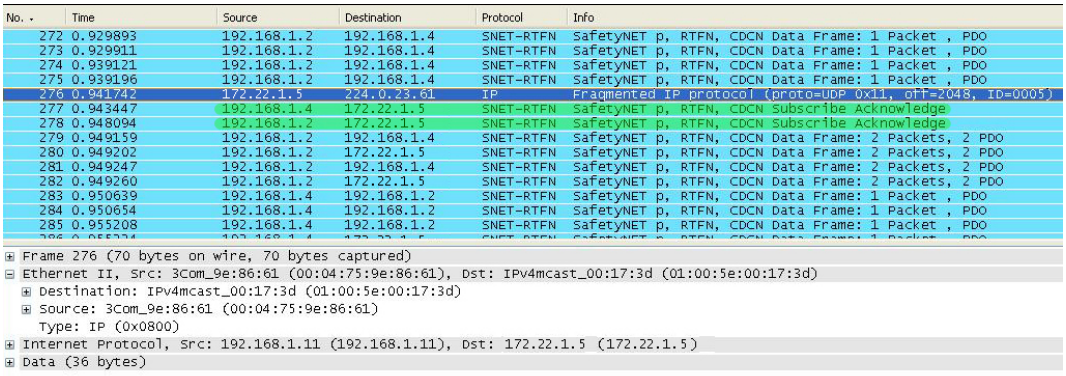


Figura 7.16: *screeshot* di una schermata di Wireshark

In seguito alla sottoscrizione dei PID corretti i dispositivi aprono la comunicazione ed inviano copia dei dati anche al nodo che li ha richiesti perchè non c'è nessuna modalità per discriminare fra quelli autorizzati e quelli estranei.

Va precisato nuovamente che l'instradamento dei dati al nodo esterno, effettuato dai dispositivi in seguito alla richiesta di sottoscrizione, viene perpetrato soltanto per un periodo limitato, in quanto la mancata ricezione di un numero prestabilito di messaggi *CDCN still alive* provoca l'interruzione del traffico dei messaggi.

7.3 Modello di attacco

In seguito alle analisi sperimentali effettuate, e grazie alla conoscenza più dettagliata dei processi di generazione e utilizzo dei PID, vengono ora proposte una serie di operazioni da eseguire per reperire le informazioni necessarie a creare il frame corrotto e indurre, così, errori di comunicazione. Questa sequenza di operazioni viene definita *modello di attacco* in quanto può essere utilizzato per attaccare la rete anche dall'esterno.

Dal momento che è possibile inviare un frame *CDCN subscribe request* in *multicast*, ciascun dispositivo provvederà a dare conferma direttamente all'attaccante della sottoscrizione dei PID, rendendo così superfluo il reperimento degli indirizzi

IP e MAC, mediante sottoscrizione al gruppo *multicast* e l'invio di un messaggio *RTFNSNR request*.

I frame *CDCN subscribe request* trasmessi vengono costituiti da un solo PID per renderne più difficile il rintracciamento da parte dei sistemi di intrusione. Quindi, dato che il campo *PID* di un frame *CDCN PDU* è costituito da 3 byte, è necessario inviare $2^{24} = 16777216$ messaggi *CDCN subscribe request* per effettuare la verifica di tutti i possibili PID. Per ogni risposta ritornata all'attaccante dovranno, poi, essere salvati in una tabella gli indirizzi IP e MAC e i PID relativi ad ogni dispositivo rilevato. Una volta conclusa l'operazione di reperimento indirizzi e PID, la tabella completa avrà n righe, una per ogni dispositivo, e $m + 2$ colonne, che equivale al massimo numero di PID $m_n + 2$ appartenenti a ciascuna riga.

La fase di inoltro dei frame corrotti consiste nell'utilizzare una qualsiasi struttura di dato *safe*, in cui è effettuata la falsificazione del *CRC*, da inviare ad ogni dispositivo rilevato inserendo ogni volta uno dei PID rilevati, tranne quelli da ciascuno effettivamente pubblicati, che non verrebbero considerati in quanto non sottoscritti. In questo modo, i frame corrotti che riportano il medesimo PID verranno inviati a tutti i dispositivi della rete, meno che a quello che li ha pubblicati, dato che non è possibile sapere a quali di essi vengono mandati i messaggi con il valore di tale PID. Come verificato anche dall'ultimo caso di induzione degli errori tramite la struttura di messaggio *standard*, il dispositivo destinatario del frame corrotto non va a verificare la corretta lunghezza in byte del campo *safe data* perchè genera subito l'errore una volta riscontrata l'incongruenza su *CRC*; per questo, si può utilizzare la stessa dimensione del messaggio anomalo utilizzato nei casi precedenti.

In seguito allo svolgimento di tutte le operazioni descritte, in tutti i dispositivi *safe* appartenenti alla rete viene generato un errore interno che ne abilita la transizione nello stato di STOP, in cui tutte le uscite di processo *standard* e *safe* vengono disabilitate, causando una conseguente interruzione di una parte o della totalità del processo produttivo, compromettendone così l'*avaiability*.

La procedura di attacco, che in questa tesi è stata realizzata dall'operatore, può essere svolta da un programma installato sul dispositivo attaccante, secondo l'algoritmo di Figura 7.17.

Supponendo di inviare i frame con un unico PID per volta, ogni $100ms$, il tempo stimato per reperire le informazioni, relative all'associazione dispositivi-PID, risulta $T_{max} \approx 19giorni\ 10h$.

Dato che è possibile inviare un frame *CDCN subscribe request* di lunghezza arbitraria, il numero massimo di *ID data* contenuti in esso, calcolato sui 1499 byte che possono essere trasmessi in un unico *SNp TYPE frame*, risulta pari a 186. Quindi, per effettuare la verifica di tutti i 2^{24} possibili PID è necessario inviare 90201 messaggi *CDCN subscribe request* in *multicast*. Supponendo di inviare in un unico frame il numero massimo di PID ogni $100ms$, il tempo stimato per reperire le informazioni si riduce a $T_{ott1} = (T_{max}/186) \approx 2h\ 30min$. La quantità dei byte del frame è molto superiore rispetto a quelli che possono transitare normalmente in una comunicazione e per questo risulta più facile il rintracciamento da parte dei sistemi di rilevamento di intrusioni.

Infine, se si considera il fatto che nei dispositivi PSSu, appartenenti al sistema PSS4000 realizzato da Pilz, i PID vengono generati incrementando sempre di

```
ALGORITMO Ricezione PID e indirizzi IP e MAC dei dispositivi della rete
1:  Inizializzazione PID=0x000000
2:  Inizializzazione Tabella
3:  Inizializzazione righeTabella=0
4:  for  PID=0 to 2^24-1
5:    Invio frame CDCN subscribe request (PID)
6:    if  (Ricezione CDCN subscribe acknowledge (MAC, IP, PID))
7:          if  (MAC e IP del dispositivo sono noti)
8:                in Tabella aggiungi alla riga corrispondente il PID
9:          else
10:               crea una nuova riga nella Tabella
11:               Tabella [p,1]=MAC dispositivo mittente
12:               Tabella [p,2]=IP dispositivo mittente
13:               Tabella [p,3]=PID
14:         end if
15:   righeTabella ++
16:   end if
17: end for
18: for  i=0 to righe Tabella   \\ viene fissato il destinatario
19:  for  j=0 to righeTabella    \\ viene fissata la riga da cui estrarre i PID
20:         if  (i≠j)  \\ non vengono inviati i PID al dispositivo che li pubblica
21:            for  ogni PID della riga j
22:               Invio frame CDC SPDO corrotto (Tabella[i,1], Tabella[i,2], PID)
23:            end for
24:         end if
25:   end for
26: end for
```

Figura 7.17: pseudocodice dell'algoritmo del *modello di attacco*

un valore costante pari a 10 in decimale (tralasciando che vengono generati partendo dal valore 0x00 00 0A), la stima temporale, effettuando un invio di frame ogni $100ms$, si riduce ulteriormente a $T_{ott2} = (T_{max}/10)/186 \approx 15min\ 2sec$, in quanto i possibili valori dei PID diventano 9021. Questa possibilità è strettamente dipendente dall'implementazione effettuata da Pilz, che potrebbe non essere più verificata nelle successive versioni dei firmware dei dispositivi o nelle implementazioni del protocollo effettuate da altre ditte produttrici di sistemi di comunicazione industriali.

Ad ogni cambiamento del programma, e rispettivo caricamento sui dispositivi, l'associazione dei PID può cambiare e, quindi, l'operazione per il reperimento dei PID necessari per bloccare la comunicazione fra i nodi deve essere effettuata nuovamente. In una situazione reale, però, il programma di gestione dell'impianto

non viene aggiornato frequentemente e il ciclo di vita è molto lungo, per cui anche il tempo impiegato nel caso peggiore T_{max} per il reperimento delle informazioni necessarie, in cui ai dispositivi sono inviati frame *CDCN subscribe request* per la sottoscrizione di un singolo PID per volta, risulta un più che ottimo compromesso.

Il tempo per effettuare l'invio dei frame corrotti, necessario per generare gli errori di comunicazione, non è stato stimato in quanto strettamente dipendente dal numero di dispositivi della rete e del numero totale di PID utilizzati. Inoltre, esso risulta trascurabile rispetto al tempo fisso per la generazione della tabella di associazioni; infatti, difficilmente sarà necessario inviare una quantità di frame corrotti rilevante rispetto ai 2^{24} *CDCN subscribe request* utilizzati per il reperimento delle informazioni.

CAPITOLO

8

Conclusioni e sviluppi futuri

Dalle analisi effettuate è stato verificato come il comportamento del protocollo di comunicazione industriale SafetyNET p, in un caso reale implementato da Pilz sui sistemi PSS4000, rispettasse lo standard riportato dalle norme IEC 61784.

Successivamente è stato studiato il comportamento di tale sistemi, connessi in varie configurazioni, alla presenza di messaggi corrotti nella rete di comunicazione. In particolare, è stato verificato che i dispositivi si comportano in maniera corretta, portando il sistema in uno stato di sicurezza *failsafe*, in seguito al rilevamento di errori di comunicazione. Purtroppo, con il generatore di pacchetti packETH a disposizione è stato possibile simulare solo una certa classe di errori. Mediante l'invio di singoli messaggi e non di una sequenza che permettesse di simulare in maniera completa il comportamento di un dispositivo, non è stato ad esempio possibile indurre degli errori dovuti a tempistiche errate nell'esecuzione del protocollo. Si è, però, notato che dal punto di vista dell'attaccante è possibile, mediante la generazione di un singolo messaggio anomalo, portare i dispositivi in stato *failsafe*, compromettendo così l'*avaiability* del sistema in maniera semplice e subdola. Infatti, un singolo messaggio è difficilmente rintracciabile da sistemi di rilevamento di intrusioni ed inoltre questo messaggio presenta la struttura di quelli di protocollo, che ne autorizza l'attraversamento dell'intera rete di comunicazione.

Infine, l'algoritmo del *modello di attacco*, utilizzato per generare degli errori nella rete di comunicazione basata sul protocollo SafetyNET p, non deve essere visto in un'ottica negativa ma in un'accezione positiva, in quanto ha permesso di mettere in luce diverse problematiche che possono verificarsi a livello dei meccanismi di *detection*, implementati dal *black channel* sulla struttura dei messaggi *safe*.

Tra gli sviluppi futuri sarà necessario completare l'analisi, andando a stimolare tutti i possibili errori di comunicazione, anche quelli legati alle tempistiche, mediante lo sviluppo di un generatore "intelligente" di pacchetti SafetyNET p, in grado di

sostituirsi completamente ad uno degli apparati Pilz, provvedendo autonomamente a decodificare e inoltrare i messaggi sulla rete.

Una volta completata l'analisi dettagliata si cercherà di proporre delle contromisure per rendere il protocollo immune da questi attacchi, che seppur non in grado di compromettere la *safety*, sono in grado di produrre un ingente danno economico a causa di una riduzione dell'*avaiability* dell'impianto produttivo.

Siccome risulta impossibile cambiare la definizione dello standard del protocollo SafetyNET p, contenuto nelle norme IEC 61784, è necessario sviluppare delle procedure aggiuntive che garantiscano l'integrità dei dati rispetto ad un attacco esterno, da inserire nelle versioni successive di tale standard. Infatti, analizzando la struttura dei messaggi *CDCN request* e *CDCN PDU*, riportate nel Capitolo 4, è riportato il campo di un byte *CDC protocol version* che è utilizzato per specificare la versione del protocollo da utilizzare per la decodifica del messaggio. In base al valore in esso contenuto, il dispositivo che riceve il dato è in grado di decodificare i byte in maniera differente. In questa maniera possono essere implementati dei meccanismi di protezione che sfruttano il valore del *CDC protocol version* per poter aggiungere interpretare qualche campo aggiuntivo, necessario per trasportare le informazioni aggiuntive per tutelare la comunicazione dalla generazione di eventuali errori dall'esterno.

Bibliografia

[1] Florian, L. *Progetto e sviluppo di un sistema di diagnostica predittivo per deviatoi ferroviari.* Tesi di laurea Magistrale in Ingegneria Elettronica, Dipartimento di Ingegneria dell'Informazione (DEI), Università degli studi di Padova, a.a 2009/2010.

[2] Smith, D.J., Simpson, K.G.L. *Functional Safety. A Straightforward Guide to applying IEC 61508 and Related Standards.* Elsevier Butterworth-Heinemann, Seconda Edizione 2004 (Prima Edizione 2001).

[3] Cox, S., Tait, R. *Safety, Reliability and Risk Management: an integrated approach* Butterworth-Heinemann, Seconda Edizione 1998 (Prima Edizione 1991).

[4] Lodari, C. *Ethernet industriali a confronto.* Rivista Fieldbus & Networks, Edizioni Fiera Milano, Novembre 2010.

[5] Zhang, S. *Evaluating Industrial Ethernet. What is Standard?* Rockwell Automation Inc., 2007.
`http://literature.rockwellautomation.com`

[6] *Functional safety and IEC 61508: A basic guide.* International Electrotechnical Commission (IEC),2004.

[7] *What Makes Ethernet Industrial?* Neteon Technologies, 2004
`http://www.neteon.net`

[8] Alessandroni, V. *L'evoluzione delle reti di comunicazione industriale.* Rivista Automazione Oggi, Edizioni Fiera Milano, Aprile 2005.

[9] Paveri, C. *Reti safety per l'industria.* Rivista Fieldbus & Networks, Edizioni Fiera Milano, Maggio 2009.

[10] *Industrial Ethernet: A Control Engineer's Guide.* Cisco Systems, 2010

[11] *Why Industrial Ethernet?* Safety Network International e.V., 2009

[12] Franzoso, M. *SafetyBus p va su SafetyNet.* Rivista Fieldbus & Networks, Edizioni Fiera Milano, Novembre 2006.

[13] *SafetyNET p. White Paper. Version 1.0*Safety Network International e.V., 2006.

[14] *CONNECTED – The magazine for safety and automation.* Safety Network International e.V., Marzo 2008.

[15] *The open forum for safety and automation.* Safety Network International e.V., 2009.

[16] *SafetyNET p Real-time Ethernet. For complete automation* Safety Network International e.V., 2011.

[17] *SafetyNET p System Description.* Safety Network International e.V., Version 1.0

[18] IEC. *IEC 61784-3-3. Industrial communication networks - Profiles - Part 3-3: Functional safety fieldbuses - Additional specifications for CPF 3 - PROFIsafe (FSCP 3/1)*, 2007.

[19] IEC. *IEC 61784-3-2. Industrial communication networks - Profiles - Part 3-2: Functional safety fieldbuses - Additional specifications for CPF 2 - CIP Safety (FSCP 2/1)*, 2008.

[20] IEC. *IEC 61158-4. Industrial Communication Networks - Fieldbus specifications - Part 4-18: Data-link layer protocol specification - Type 18 elements*, 2010.

[21] IEC. *IEC 61508-1. Functional safety of electrical/electronic/programmable electronic safety-related systems - Part 1 - General requirements*, 2010.

[22] IEC. *IEC 61784-3-18. Industrial Communication Networks - Profiles - Part 3 - 18: Functional safety fieldbuses - Additional specifications for CPF 18 - SafetyNET p (FSCP 18/1)*, 2011.