ALMA MATER STUDIORUM – UNIVERSITÀ DI BOLOGNA
CAMPUS DI CESENA

DIPARTIMENTO DI INFORMATICA – SCIENZA E INGEGNERIA
Corso di Laurea in Ingegneria e Scienze Informatiche

INTEGRAZIONE DI DIGITAL TWINS E MIXED REALITY: UN CASO DI STUDIO

Elaborato in
SISTEMI EMBEDDED E INTERNET OF THINGS

Relatore
Prof. ALESSANDRO RICCI

Presentata da
ALBERTO DI GIROLAMO

Correlatore
Dott. SAMUELE BURATTINI

Anno Accademico 2021 – 2022

A me stesso.

Obsession is going to beat talent every time.

Indice

Introduzione

Questa tesi ha come proposito esplorare le architetture che integrano i Digital Twins in applicazioni di Mixed Reality. L'obiettivo cardine è quello di mostrare una possibile realizzazione di un sistema di questo tipo.

Per raggiungere l'obiettivo posto si è scelto di progettare un sistema che sia realizzabile come soluzione in un contesto reale. L'architettura presentata viene suddivisa in sottosistemi indipendenti in base alla tecnologia presa di riferimento.

Il progetto che sarà presentato successivamente non lo si di deve considerare come miglior linea guida per la realizzazione di questa particolare architettura, ma rappresenta una prova che la Realtà Mista e il concetto di Gemello Digitale possono coesistere in un sistema unico.

La prima parte di questo elaborato si occupa di spiegare i concetti e le tecnologie che si andranno poi a utilizzare.

Nel primo capitolo vengono introdotti i Digital Twins. Nelle prime sezioni viene spiegata la definizione concettuale di Gemello Digitale, seguita da una suddivisione delle diverse tipologie dei DT esistenti. Una breve presentazione del rapporto tra i Digital Twins e IoT precede la descrizione delle tecnologie e piattaforme utilizzate per la realizzazione del modulo Digital Twins che compone il progetto.

Vengono presentati alcuni servizi offerti dalla suite di Azure dando maggior importanza al servizio Digital Twins essendo l'organo cardine di questo modulo.

Nel secondo capitolo si presenta il concetto di Extended Reality riportando le tre macro sezioni che compongono questa tecnologia: Virtual Reality, Augmented Reality e Mixed Reality.

Come dispositivo utilizzato per interagire con la Mixed Reality si è scelto di utilizzare il visore Microsoft di seconda generazione chiamato HoloLens.

Nella sezione successiva viene presentata la piattaforma di sviluppo utilizzata per la realizzazione di questo modulo: Unity. A seguire vengono analizzate le tecnologie a supporto, utilizzate per migliorare la produttività nella realizzazione di applicazioni in Mixed Reality: MRTK e OpenXR.

Nella prima parte della tesi, la Mixed Reality e i Digital Twins sono stati trattati come argomenti distinti. Il terzo capitolo vuole essere un'anticipazione circa le possibili applicazioni dell'architettura presentata successivamente.

Viene spiegato cosa significa unire la Mixed Reality con i Digital Twins evidenziando le possibili migliorie che si possono ottenere applicando queste tecnologie in un qualunque ambiente di lavoro.

Successivamente vengono riportate le caratteristiche principali che un sistema di questo tipo deve avere. Viene data una definizione di interoperabilità e analizzato il significato. A seguire viene presentato il concetto di modularità.

Le sezione successiva si occupa di presentare al lettore le possibili interazioni che possono avvenire in un sistema che si basa sulla Mixed Reality.

A seguire si è scelto realizzare un breve excursus riguardo il concetto di ambiente condiviso in un sistema di Mixed Reality e Digital Twins presentando i principali problemi che si devono affrontare realizzando un architettura di questo tipo.

Nell'ultima sezione di questo capitolo viene presentato il concetto di Mirror World. Rappresenta una realtà utopistica caratterizzata da una radicata presenza della tecnologia dei Digital Twins e della Mixed Reality nella nostra vita.

Nel quarto capitolo vine presentata e realizzata una possibile architettura di un sistema che interagisce con i Digital Twins tramite Mixed Reality.

Per la realizzazione dell'architettura è stato preso in considerazione uno scenario reale. Si è scelto di realizzare un'applicazione a supporto dei medici che offrisse, durante un'operazione chirurgica, la possibilità di visualizzare il monitor dei parametri vitali tramite ologramma.

Il sistema presentato è stato suddiviso in tre macro sezioni: Physical Asset, Digital Twins e Hologram. Questi sottosistemi vengono analizzati uno per volta e per ogni sezione viene realizzata l'architettura inerente.

Come ultimo capitolo vengono riportate delle considerazioni finali circa il sistema realizzato mostrando delle possibili alternative architetturali rispetto alla soluzione presentata.

Capitolo 1

Introduzione ai Digital Twins

Questo capitolo vuole presentare al lettore un'introduzione generale riguardo il mondo dei Digital Twins, partendo da una definizione concettuale fino ad arrivare alle tecnologie utilizzate per implementare un sistema che si basa sui DT.

1.1 Digital Twins

L'idea di utilizzare i "Gemelli" risale al programma Apollo della NASA, dove sono stati costruiti due mezzi spaziali identici per consentire di rispecchiare le condizioni del veicolo durante la missione.

Con il termine Digital Twin [12] (DT, Gemello Digitale) facciamo riferimento alla copia virtuale di una qualsiasi entità fisica (Gemello Fisico). I due "Gemelli" sono interconnessi da uno scambio di dati in tempo reale. Un Gemello Digitale è un modello virtuale progettato per riflettere un oggetto fisico. Tutte le informazioni e i dati che vengono raccolti ed elaborati, vengono riportati alla copia digitale. La controparte virtualizzata può essere usata per eseguire simulazioni, effettuare esperimenti, oppure per replicare l'oggetto originale.

1.1.1 Tipologie di Digital Twins

Esistono diverse tipologie di Digital Twins a seconda del livello di "ingrandimento" del device.

All'interno di un sistema è possibile usufruire di più tipi di Gemelli che interagiscono fra di loro. Di seguito vengono riportate le tipologie di Digital Twins esistenti:

- *Component twins*: rappresenta il più piccolo componente digitalizzato, consiste nell'unità di base del Digital Twins.

- *Asset twins*: raggruppa più Component twins che lavorano assieme. Un *Asset twins* permette di analizzare l'interazione dei componenti che lo compongono. I dati ottenuti possono essere elaborati e trasformati in informazioni utili.

- *System twins*: permette di analizzare il comportamento che possiedono diverse risorse sotto un unico sistema funzionante. I *system twins* possono fornire visibilità circa l'interazione delle risorse e suggerire miglioramenti delle prestazioni.

- *Process twins*: rappresenta il livello macro d'ingrandimento. Permette di analizzare un intero impianto di produzione.

Sebbene i DT e le simulazioni rappresentino entrambi una replica di un oggetto fisico, possiedono una grande differenza. Una simulazione, rispetto a un DT, rispecchia un determinato comportamento dell'oggetto reale, mentre un Gemello Digitale essendo una replica completa, può eseguire una qualsiasi simulazione senza fare alcuna distinzione.

Un'altra diversità è data dalla tipologia di connessione. Un DT necessita di uno scambio bidirezionale in tempo reale di dati con il Gemello Fisico, mentre una simulazione non trae vantaggio utilizzando una connessione real time.

1.1.2 Il rapporto tra Digital Twin e IoT

Lo sviluppo della tecnologia IoT è uno dei fattori che hanno portato i Digital Twins a questo livello di notorietà.

I dati con cui operano i Digital Twin vengono ottenuti tramite sensori IoT posizionati su oggetti/sistemi fisici che, rilevando i dati, consentendo di costruire la versione digitale dell'ambiente.

A differenza dei dispositivi IoT, un Gemello Digitale oltre a poter rappresentare un qualsiasi oggetto, può replicare sia un sistema che un singolo processo.

In un contesto aziendale dove sono presenti numerosi dispositivi IoT il ruolo dei Digital Twins possiede un valore inestimabile. Più flussi informativi rilevati dai device IoT ci sono, maggiore è la complessità nel mantenerli con un certo ordine logico.

I Digital Twins fungono da repository per i dati rilevati dai dispositivi IoT contestualizzandoli. Le informazioni provenienti da un'unica controparte fisica vengono mappate nella stessa destinazione digitale. Mantenendo quindi

un certo ordine logico della struttura dei dati. Poiché le informazioni tra loro inerenti puntano allo stesso gemello è facile integrare un software per la gestione di questi ultimi.

Il software che si occupa della gestione dei Digital Twins deve semplificare l'accesso e l'integrazione dei sistemi con i gemelli digitali. Per quanto riguarda l'interfacciamento verso la raccolta delle informazioni, il SW[1] deve stabilire un protocollo di comunicazione per lo scambio dei dati con i relativi sensori, determinando delle regole e delle metodologie precise per permettere la lettura delle informazioni possedute.

In seguito verrà esposto il software utilizzato in questa tesi.

1.1.3 Il valore aggiunto dai Digital Twins alle aziende

In tutti i settori aziendali è possibile adottare i Digital Twins. Questa tecnologià la si può utilizzare per migliorare la progettazione di nuove linee di produzione. Prima di realizzare fisicamente un sistema è possibile stabilire a priori eventuali problemi o possibili ottimizzazioni da svolgere. Le tecniche di visualizzazione dei Digital Twins permettono di rendere i problemi più visibili.

I DT permettono di emulare l'ambiente fisico dando modo ai progettisti di aggiornare un sistema esistente o di realizzarne uno nuovo riducendo il rischio di costosi errori. Spesso lavorare su degli scenari simulati risulta più veloce e facile rispetto a lavorare su asset fisici.

1.2 Piattaforme per lo sviluppo dei Digital Twins

Esistono numerose soluzioni per realizzare Digital Twins. In questa sezione è stato deciso di presentare la piattaforma sviluppata da Microsoft che verrà successivamente impiegata nello sviluppo della parte di progetto.

1.2.1 Azure Digital Twins

Azure Digital Twins[8] (ADT) è un PaaS[2] IoT che permette di creare una rappresentazione digitale di oggetti e luoghi reali.

Implementare Azure Digital Twins comporta numerosi benefici tra cui permette di modellare qualsiasi ambiente in modo scalabile e interrogare il sistema per estrarre informazioni in tempo reale.

[1] Software

[2] Platform as a Service è una tipologia di architettura a servizi in cui l'hardware e la piattaforma software vengono firniti da terze parti. L'utente può sviluppare applicazioni senza dover creare o gestire l'infrastruttura.

In ADT vengono definite le entità digitali in tipologie di gemelli personalizzati chiamati *models*. Questo dà all'utente la possibilità di definire un vocabolario per creare il proprio grafo di Digital Twins.

Un modello è simile a una classe in un linguaggio *OOP*[3] e definisce un particolare oggetto e il suo comportamento. I modelli sono definiti in un linguaggio simile a JSON chiamato *Digital Twins Definition Language (DTDL)*[5]

Quando si vuole definire un nuovo modello, il primo elemento da implementare è l'interfaccia e può contenere i seguenti campi:

- *Proprietà* : rappresentano i campi che determinano le caratteristiche di un'entità, come `schema` che identifica il tipo di dato (double, dateTime ecc.) oppure `contents` che racchiude un insieme di proprietà per definire determinati contenuti dell'interfaccia.

- *Telemetria* : rappresentano misurazioni o eventi. Questi campi vengono usati per descrivere le letture dei sensori del dispositivo. A differenza delle proprietà, questi dati non vengono archiviati in un Digital Twin, ma sono più simili a un flusso di eventi. Si trattano di un set di messaggi di dati con durata breve. Se non si configura l'ascolto dell'evento quando si verifica, questo viene perso.

- *Relazione* : le relazioni rappresentano in modo in cui un Digital Twin può essere coinvolto con altri DT.

- *Componenti* : consentono di compilare l'interfaccia del modello come assembly di altre interfacce.

A seguire viene riportato un esempio[9] d'interfaccia di un dispositivo termostato. Presenta una telemetria che riporta la misurazione della temperatura e una proprietà di lettura/scrittura che controlla la temperatura desiderata.

[3]Paradigma di programmazione orientato ad oggetti

```
{
    "@id": "dtmi:com:example:Thermostat;1",
    "@type": "Interface",
    "displayName": "Thermostat",
    "contents": [
        {
            "@type": "Telemetry",
            "name": "temp",
            "schema": "double"
        },
        {
            "@type": "Property",
            "name": "setPointTemp",
            "writable": true,
            "schema": "double"
        }
    ],
    "@context": "dtmi:dtdl:context;2"
}
```

Listato 1.1: Esempio di codice DTDL di un dispositivo termostato

1.2.2 Azure Hub IoT

L'hub IoT di Azure[4] è un servizio gestito nel cloud di Microsoft, funge da hub per la comunicazione bidirezionale tra l'applicazione IoT e i relativi dispositivi collegati. Sono supportati diversi modelli di comunicazione, è possibile inviare i dati di telemetria[4] dai dispositivi al cloud, caricare file e tenere sotto controllo i dispositivi direttamente sulla piattaforma Azure.

Ogni soluzione IoT è diversa, l'hub di Azure può supportare carichi elevati di lavoro, pertanto Microsoft offre un servizio scalabile in base alle necessità del cliente.

Attraverso l'hub IoT di Azure è possibile eseguire comandi (es. reboot) richiamando metodi nei dispositivi. Questi metodi rappresentano *request-reply interaction* con un dispositivo utilizzando una chiamata HTTP che può avere sia esito negativo che positivo.

[4]Esempi di telemetria ricevuti da un dispositivo possono includere dati del sensore, ad esempio velocità o temperatura, un messaggio di errore oppure un messaggio informativo

1.2.3 Azure Functions

Azure Functions è un servizio offerto da Microsoft che consente d'implementare la logica del sistema in blocchi di codice immediatamente disponibile. Quesi blocchi prendono il nome di *funzioni* o *functions* e possono essere scritti in diversi linguaggi: *C, Java, JavaScript, PowerShell e Python.*

Il modello utilizzato segue un architettura *serverless*, consentendo di scrivere meno codice e di non gestire l'uso dei server sia da parte dell'utilizzatore che da parte dello sviluppatore.

Le functions di Azure sono scalabili, quando la richiesta di esecuzione aumenta, più risorse vengono allocate automaticamente al servizio. Viceversa, man mano che le richieste diminuiscono, tutte le risorse extra e le istanze dell'applicazione vengono eliminate.

Nell'immagine riportata successivamente è possibile visualizzare una tabella che mostra alcuni scenari di Azure Functions possibili.

Per...	**Quindi...**
Creare un'API Web	Implementare un endpoint per le applicazioni Web usando il trigger HTTP
Elaborare caricamenti di file	Eseguire codice quando un file viene caricato o cambiato in archiviazione BLOB
Creare un flusso di lavoro serverless	Concatenare una serie di funzioni tra loro tramite Durable Functions
Rispondere alle modifiche di database	Eseguire logica personalizzata quando un documento viene creato o aggiornato in Cosmos DB
Eseguire attività pianificate	Eseguire il codice in intervalli di tempo predefiniti
Creare sistemi affidabili per le code di messaggi	Elaborare le code di messaggi con Archiviazione code, bus di servizio o Hub eventi
Analizzare flussi di dati IoT	Raccogliere ed elaborare i dati dei dispositivi IoT
Elaborare dati in tempo reale	Usare Funzioni e SignalR per rispondere ai dati al momento

Figura 1.1: [3] Alcune degli scenari di Azure Functions possibili. Tabella originale disponibile nella documentazione Microsoft [3]

Capitolo 2

Extended Reality

Con il termine Extended Reality (XR, Realtà Estesa) ci riferiamo a tutte quelle tecnologie immersive che estendono la realtà unendo il mondo reale con un mondo virtuale.

Attualmente esistono tre diverse tipologie di Realtà Estesa, ciascuna definita da un preciso livello d'interazione tra il digitale e il reale.

2.1 Virtual Reality

La Virtual Reality o VR (Realtà Virtuale) rappresenta la tecnologia più conosciuta al momento. Attraverso l'uso di un visore (es. Oculus Rift riportato in figura 2.1) veniamo immersi completamente in un mondo digitale, dove è possibile interagire con gli elementi al suo interno.

L'idea che si trova alla base è quella di sostituire completamente il mondo reale con uno digitale.

L'hardware e il software generano immagini e video 3D trasmettendoli in un display. Lo schermo si trova all'interno del visore, il quale è fissato sulla testa dell'utente sopra gli occhi. Per interagire con mondo virtuale è possibile utilizzare controller manuali come guanti tattili oppure joystick. Spostando la posizione della testa viene replicato il movimento all'interno della scena digitale, il visore infatti è costituito da sensori che raccolgono informazioni circa gli stimoli del corpo.

I visori VR cercano di simulare la percezione uditiva dei suoni nel mondo reale utilizzando una tecnologia di audio spaziale, rendendo l'esperienza più immersiva.

Figura 2.1: Oculus Rift S

2.2 Augmented Reality

Con Augmented Reality AR indichiamo quella tecnologia che integra delle informazioni digitali con l'ambiente circostante dell'utente. A differenza della Virtual Reality che crea un ambiente totalmente artificiale, con l'AR si vuole aggiungere e sovrapporre al mondo reale delle informazioni percettive.

L' Augmented Reality fornisce elementi visivi corredati da suoni attraverso un dispositivo come uno smartphone o un visore particolare. Queste informazioni alterano la percezione del mondo reale.

Utilizzando specifiche applicazioni è possibile sperimentare la realtà aumentata direttamente sullo schermo del telefono, visualizzando il mondo reale con aggiunta d'immagini digitali.

Nel 2016 venne lanciato un gioco per smartphone che utilizzava questa tecnologia: Pokemon Go. Permetteva di "fondere" il digitale con il mondo fisico rendendo l'esperienza di gioco unica per ogni utente.

Un ulteriore utilizzo di questa tecnologia è quella di ottenere informazioni del soggetto puntato dalla camera dello smartphone e visualizzare tali informazioni direttamente sullo schermo (come rappresenta l'immagine 2.2). Utile in un contesto come un museo o per ottenere indicazione in una città.

Figura 2.2: Immagine dimostrativa del Museo Archeologico Virtuale di Napoli, MAV 5.0

2.3 Mixed Reality

L'ultima tecnologia è la Mixed Reality o MR. Porta ad un passo successivo la Augmented Reality, propone di ricoprire il mondo reale con oggetti digitali generati dal computer. HoloLens 2 rappresenta la seconda generazione dei visori di Microsoft per la Mixed Reality. A differenza della Realtà Virtuale dove indossando un visore cancelliamo completamente il mondo reale immergendoci in uno digitale, i visori per la Realtà Mista ci permettono di vedere il mondo reale avendo la possibilità di posizionare oggetti digitali sotto forma di ologrammi. Queste immagini olografiche si adattano all'ambiente reale circostante interagendo con esso.

Attraverso l'uso dei sensori che dispongono i visori, con la Mixed Reality abbiamo la possibilità di mappare la realtà che ci circonda e interagirci a un livello superiore del normale. Gli ologrammi che posizioniamo sono visti e gestiti come veri oggetti.

I sistemi MR sono progettati per dare ai loro utenti l'illusione che gli oggetti digitali si trovino nello stesso spazio di quelli fisici. Abbiamo la possibilità di "posizionarli" su di una parete oppure "appoggiarli" su di un tavolo. In base alle nostre preferenze possiamo stabilire se gli oggetti olografici devono rispondere alla fisica o meno.

L'allineamento in tempo reale degli oggetti virtuali con l'ambiente fisico costituisce una sfida complessa, ma rappresenta una caratteristica importante

per questi sistemi.

La realtà mista pone una serie di severi requisiti tecnologici per un'implementazione realistica. I visori devono avere un'alta risoluzione e un elevato contrasto per una corretta visualizzazione. Importante è anche il rilevamento preciso della posizione e dell'orientamento degli oggetti rispetto all'utente.

Possiamo dire quindi che la Mixed Reality è composta da diverse tecnologie presentate di seguito:

- *Detection environmental*: rappresenta la capacità di mappare uno spazio e sovrapporre informazioni digitali su quelle reali.
- *Human understanding*: comprensione umana, una tecnologia che attraverso dei sensori tiene traccia dei movimenti e degli input dell'utente.
- *Suono spaziale*: come nella Virtual Reality viene implementato un'audio spaziale, progettato per rendere l'esperienza più immersiva e realistica.
- *Position detection*: capacità di monitorare la posizione dell'utente e dell'ambiente circostante.
- *3D holograms*: implementare delle risorse tridimensionali olografiche. Rappresentano una digitalizzazione di un oggetto o ambiente reale.

La Mixed Reality è sempre più presente nelle nostre vite. Spesso ne facciamo uso e non ce ne rendiamo neanche conto. Un semplice esempio lo possiamo avere quando utilizziamo i social network: i cosiddetti filtri che utilizziamo per scattare foto o registrare video, non sono altro che delle maschere sviluppate in realità aumentata che l'utente può applicare sul prorpio volto.

Una delle prime case automobilistiche a utilizzare la realtà mista nella progettazione è stata Ford. Grazie a questa tecnologia, ha permesso agli ingengeri e ai progettisti di utilizzarla in combinazione con i prototipi di argilla per una progettazione più rapida riducendo i costi.

Anche nel marketing l'implementazione della Mixed Reality sta avendo un grosso successo. Attraverso delle anteprime virtuali è possibile far capire le potenzialità e il funzionamento della merce senza avere la necessità di avere fisicamente in mano il prodotto. Alcune piattaforme di shopping online hanno messo a disposizione delle applicazioni che permettono di creare un outfit indossandolo virtualmente.

L'utilizzo della Mixed Reality possiede un grosso potenziale anche in ambito medico. L'applicazione della MR in ambito ospedaliero può spaziare su più campi come nella riabilitazione motoria, nelle terapie di disturbi psichiatrici oppure per il supporto all'operatore sanitario durante interventi chirurgici o

visite mediche. Il modello di realtà aumentata è stato adottato anche nella elaborazione d'immagini TAC e di RMN, il medico è così in grado di sovrapporre radiologie sul corpo del paziente durante l'intervento, aumentando la sicurezza e la precisione.

2.4 Tecnologie a supporto di applicazioni di Mixed Reality

A seguire vengono presentate le tecnologie che possono essere utilizzare per realizzare un'applicazione di Mixed Reality. In questa sezione è stato deciso di introdurre HoloLens come dispositivo per interagire con un sistema MR e Unity con tecnologie annesse come motore grafico utilizzato per la realizzazione dell'applicazione. Esistono numerose soluzioni possibili per realizzare un'applicazione di Mixed Reality, sia in termini di visori che per quanto riguarda le piattaforme di sviluppo, ma in questa tesi non verranno presentate poiché è stato scelto di utilizzare quelle descritte per realizzare un'applicazione che verrà sviluppata nel capitolo 4.

2.4.1 HoloLens 2

HoloLens 2 (riportato nell'immagine 2.3)rappresentano la seconda generazione degli *smartglass*[1] per la Mixed Reality prodotti da Microsoft. Questo visore è pensato per un uso professionale industriale. Disponibile nel corso del 2019 è dotato di un processore Qualcomm Snapdragon 850. Possiede una connettività Wi-Fi 802.11 ac, Bluetooth 5.0 e USB C. Rispetto alla generazione precedente possiede un campo visivo raddoppiato con una densità olografica di 47 pixel.

Con HoloLens 2 è possibile interagire con gli ologrammi utilizzando direttamente le proprie mani, oppure attraverso Eye tracking.

Per sviluppare soluzioni per Hololens 2 è possibile utilizzare piattaforme di sviluppo AR e VR come Unity (presentato nella prossima sezione), Unreal e Vuforia. Per semplificare la progettazione è possibile usufruire delle API aperte come Khronos e OpenXR.

Per aumentare la produttività e facilitare lo sviluppo di applicazioni per la Mixed Reality utilizzando HoloLens, Microsoft ha messo a disposizione un set di componenti e funzionalità chiamato *MRTK*. Questa tecnologia verrà presentata nei moduli successivi.

[1] "Occhiali intelligenti" sono sei dispositivi indossabili che attraverso un display permettono di vedere informazioni aggiuntive a chi li indossa.

Figura 2.3: HoloLens 2, smartglass per la Mixed Reality sviluppati da Microsoft

2.4.2 Unity, un motore grafico

Unity è un motore grafico multipiattaforma sviluppato da Unity Technologies. Nato in origine per lo sviluppo di giochi ottimizzato per ambienti 3D, include successivamente funzionalità per lo sviluppo di applicazioni per la Extended Reality.

Unity offre la possibilità di creare sia giochi 2D che 3D. Mette a disposizione un'API di scripting in C#. Le potenzialità di questo motore grafico sono molteplici, in questo elaborato facciamo riferimento a quello che concerne la creazione di applicazioni per la Extender Reality, più precisamente riguardanti la Mixed Reality.

Si userà la piattaforma Unity per creare una scena 3D, importare asset/oggetti, connettere la scena Unity Azure Digital Twins e successivamente creare un'applicazione in HoloLens per interagire con il gemello digitale.

2.4.3 Microsoft Mixed Reality Toolkit

Microsoft Mixed Reality Toolkit (MRTK) [6] è un kit di sviluppo multipiattaforma open source progettato per la realizzazione di applicazioni per la Mixed Reality.

MRTK consente la prototipazione rapida tramite un simulatore che consente di visualizzare le modifiche. Supporta un'ampia gamma di piattaforme, tra cui Microsoft HoloLens, Oculus, OpenVR e Ultraleap per il tracciamento della mano.

Questo toolkit è modulare, permette quindi di importare i plug-in necessari senza il bisogno di implementare tutti i componenti nel progetto. In questo

modo si riesce a mantenere le dimensioni del progetto più contenute e risulta semplificata la gestione.

Alcune delle classi di MRTK più usate per la manipolazione degli ologrammi sono le seguenti:

- `ObjectManipulator`: consente di spostare, ridimensionare e ruotare un oggetto usando le mani.
- `BoundsControl`: consente di trasformare gli oggetti disegnando un cubo attorno all'oggetto per capire la possibilità di manipolazione. In HoloLens 2 viene aggiunto anche un feedback visivo per permettere di percepire la distanza delle mani dall'oggetto.
- `NearInteractionGrabbable`: è possibile aggiungere un componente NearInteractionGrabbable a qualsiasi *GameObject* per renderlo afferrabile in collisione.
- `RotationAxisConstraint`: permettere di limitare la rotazione del *GameObject* in determinati assi.

2.4.4 OpenXR

OpenXR [7] è uno standard API *open royalty*[2] sviluppato da Khronos, che si occupa di fornire ai motori un'accesso nativo alle piattaforme e ai dispositivi di Extended Reality (come HoloLens). OpenXR non è un motore di Mixed Reality, ma consente ai motori come Unity di scrivere codice portatile semplificando lo sviluppo dei software AR/VR, rendendo possibile quindi il riuso del codice su più piattaforme.

A seguire viene riportata un'immagine che mostra graficamente il problema della frammentazione delle tecnologie risolto da OpenXR.

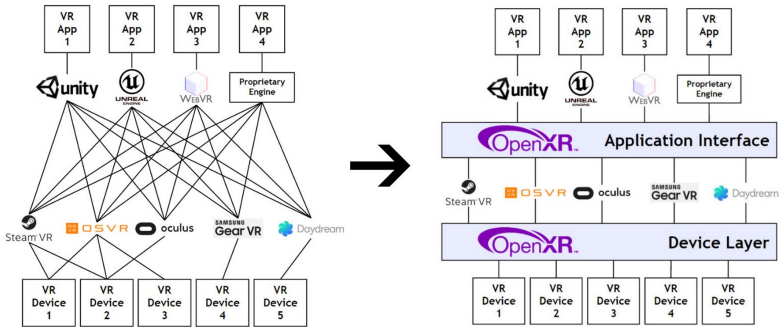


Figura 2.4: Problema della frammentazione risolto implementando OpenXR

[2]tipo di licenza che permette l'utilizzo di una risorsa senza limiti di tempo e spazio, dopo aver pagando una cifra iniziale

Capitolo 3

La Mixed Reality interseca i Digital Twins

Il concetto di Realità Aumentata esiste da prima che nascessero i Digital Twins. La Mixed Reality, come qualunque altra tecnologia di virtualizzazione della realtà, può essere applicata con diverse finalità, che spaziano dal gaming a scopi accademici (come simulazioni di volo) fino ad arrivare negli ambienti industriali.

Abbiamo detto che i Digital Twins sono una sorta di copia digitale di determinati oggetti fisici. Queste due parti sono connesse e comunicano tra loro attraverso uno scambio di dati real-time. Ciò significa che tutti i cambiamenti subiti dalla controparte fisica si riflettono nel gemello digitalizzato. Questo fenomeno avviene attraverso un concetto chiamato *twinning*, un processo durante il quale il gemello virtuale viene creato e connesso all'oggetto del mondo reale.I Digital Twins sono utilizzati maggiormente in ambito IoT per permettere la gestione dei dati raccolti dai dispositivi collegati in internet, sia in un contesto industriale che in uno casalingo.

Possiamo constatare quindi che i sistemi di Mixed Reality non nascono originariamente con l'intento di essere implementati con i Digital Twins.

3.1 Integrare i Digital Twins con la Mixed Reality

Fino ad ora i Digital Twins e la Mixed Reality sono stati affrontati come argomenti ben distinti, effettivamente entrambe le tecnologie esistono come sistemi *stand-alone*. Questo pur essendo vero, non nega il fatto che possa esistere una possibile interazione tra questi due mondi.

Se pensiamo quindi d'implementare in un ambiente simulato i Digital Twins, portiamo a un livello superiore il concetto d'interazione fra il mondo reale con quello virtuale. La simulazione si baserà quindi su una continua raccolta di dati e scambio di informazioni tra i due mondi.

Unendo queste due tecnologie, che in un primo momento possono sembrare incompatibili, otteniamo un'infinità di applicazioni possibili. Molti dei lavori che oggi vengono effettuati e che in un primo momento non consideriamo inerenti con queste tecnologie, potrebbero avere un miglioramento produttivo e qualitativo implementandole.

L'utilizzo della Mixed Reality permette di superare i limiti e vincoli fisici che si potrebbero incontrare durante lo svolgimento di un lavoro.

Consideriamo un' applicazione pratica in un possibile scenario reale. Immaginiamoci all'interno di un contesto lavorativo dove l'utente ha a che fare con numerosi elementi informativi di output come vari monitor, led ecc.. Tenere tutti i valori sotto controllo può risultare difficoltoso, sopratutto se i display da monitorare sono dislocati in posizioni differenti. In questo caso la Mixed Reality arriva in soccorso: in un'ottica futura, se tutti i dispositivi possedessero un rispettivo Digital Twin, l'operatore potrebbe comporsi a proprio bisogno l'area di lavoro, visualizzando i monitor che più gli interessano.

Anche la sperimentazione è una parte importante del processo di sviluppo di nuovi sistemi. Tuttavia, testare ed eseguire esperimenti su prototipi reali richiede un elevato costo sia finanziariamente e sia in termini di tempo. Con una replica digitale, tuttavia, è possibile testare le nuove soluzioni ed eseguire simulazioni con modifiche rapide. Il Digital Twin aiuta a collaudare le modifiche ai sistemi esistenti riducendo i tempi del fermo macchina. Successivamente all'esecuzione del test, se l'esperimento non ha prodotto i risultati desiderati, le modifiche non verrebbero eseguite a scapito delle risorse fisiche.

Ad esempio, in una linea di produzione, un collegamento inefficace può essere perfezionato e testato senza dover arrestare l'intero processo, non sostenendo così costi aggiuntivi.

Visualizzare i Digital Twins con sistemi di MR permette di monitorare un processo produttivo e migliorare la consapevolezza circa il suo funzionamento. Pensiamo di essere affianco a una macchina, il suo Digital Twin può essere sovrapposto a essa in modo da poter visualizzare il suo funzionamento interno e comprenderne i flussi di dati.

Il concetto chiave di questi sistemi è la **personalizzazione** e **modularizzazione** a proprio piacimento.

Per rendere ciò possibile, è necessario che esita un certo livello d'**interoperabilità** dei dispositivi e delle architetture. Permettendo così una completa integrazione dei diversi sistemi.

3.2 Interoperabilità tra sistemi eterogenei

In un sistema che integra tecnologie eterogenee, come i Digital Twins e la Mixed Reality, l'interoperabilità rappresenta una caratteristica fondamentale per stabilire la qualità del sistema.

L'interoperabilità può essere definita[1] quindi, come la capacità di due sistemi di comunicare e condividere servizi tra loro. I diversi elementi che compongono un'architettura di questo tipo dovrebbero cooperare e comunicare senza soluzione di continuità. Questo non accade per l' eterogeneità riguardante i dispositivi, le tipologie di comunicazione, i servizi offerti ecc..

3.2.1 Scomposizione del concetto di Interoperabilità

Per comprendere appieno il concetto di interoperabilità è possibile provare a classificarlo[10] come segue.

- Interoperabilità dei dispositivi: I Digital Twins possono rappresentare numerosi dispositivi/sistemi fisici differenti. É possibile suddividerli in base alla loro complessità di progettazione e funzionamento. Diverse caratteristiche e architetture assieme ad un'assenza di standard di comunicazione porta a un aumento d'incompatibilità.

- Interoperabilità della rete: diversamente dai computer desktop, i Digital Twins generalmente si basano su rete wireless a corto raggio. Solitamente queste tecnologie di comunicazione sono inaffidabili. Con interoperabilità a livello di rete si intende permettere lo scambio continuo di messaggi tra sistemi con reti diverse. Progettare una rete per DT porta a gestire diversi problemi come il routing, l'ottimizzazione delle risorse, la sicurezza, la QoS[1] e il supporto alla mobilità.

- Interoperabilità sintattica: si intende l'interoperabilità della struttura dei dati utilizzata per lo scambio d'informazioni tra sistemi di Digital Twins eterogenei. Il contenuto dei messaggi deve essere serializzato[2] con un determinato formato (es. XML o JSON). Il destinatario attraverso regole sintattiche decodifica il messaggio. Se la codifica del mittente è incompatibile con la decodifica del destinatario sorgono problemi di interoperabilità.

- Interoperabilità semantica: rappresenta la capacità di due o più sistemi di scambiarsi informazioni con un significato non ambiguo e preciso.

[1] Quality of Service

[2] La serializzazione rappresenta un processo per trasmettere su una rete l'intero stato di oggetto permettendo poi di ricrearlo attraverso la deserializzazione.

Molti DT gestiti da piattaforme diverse, seppur utilizzino un formato ben definito (come XML o JSON già citati precedentemente) non possiedono un modello in comune che descrive le varie procedure per interpretare correttamente il messaggio. Ad esempio, un DT realizzato con la piattaforma di Azure non è detto che sia compatibile con un servizio offerto da un'altra azienda.

- Interoperabilità della piattaforma: sorgono a causa della diversità dei sistemi operativi (OS), linguaggi di programmazione, strutture dati e architetture. Questa non uniformità impedisce agli sviluppatori di sviluppare applicazioni di Mixed Reality con Digital Twins multipiattaforma.

3.3 Human-Computer Interaction nei sistemi MR per interagire con i DT

Se volessimo rappresentare in uno schema l'interazione della Miexed Reality tra il mondo digitale con quello fisico la potremmo posizionare al centro (come rappresentato nella figura 3.1).

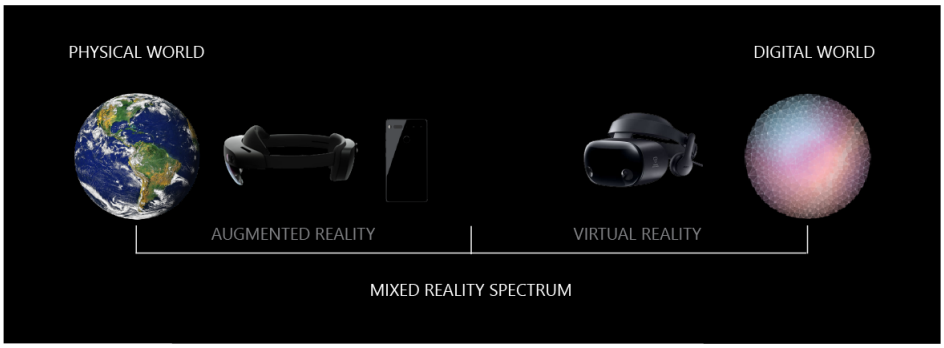


Figura 3.1: Rappresentazione della Mixed Reality rispetto al Mondo Fisico con quello Digitale

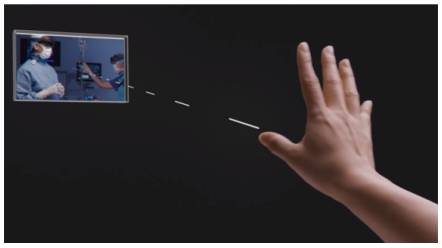
La differenza sostanziale tra un classico programma sviluppato per una piattaforma "standard" (PC, Smartphone, Server ecc.) con una sviluppata invece per i sistemi ER è il tipo di interazione necessario tra l'utente e il sistema. Per interagire con i Digital Twin olografici esistono diverse tipologie di input/output più o meno convenzionali, ma per permettere una completa interazione con un sistema di Realtà Mista è necessario ampliare questo spettro.

Per ottenere una corretta user experience è necessario mappare l'ambiente circostante per ricevere input da questo.

Non essendo seduti davanti a uno schermo "statico" è necessario implementare un sistema di output sonoro che non si basi solamente su due canali, ma è necessario implementare un audio 3D. Questo comporta ad un posizionamento virtuale di sorgenti sonore nello spazio circostante, con una determinata posizione e profondità. Tutte queste caratteristiche rappresentano l'interazione uomo-macchina. Con HoloLens 2 è possibile utilizzare numerosi gesti per interagire con gli ologrammi dei Digital Twins:

- *Tocco*: toccare (figura 3.2a) o afferrare direttamente gli ologrammi, viene visualizzato un puntatore vicino alla punta del dito quando la mano viene intercettata.
- *Raggio della mano*: tenendo la mano aperta con il palmo in direzione dell'oggetto da puntare viene visualizzato un puntatore (la figura 3.2b mostra un esempio esplicativo).
- *Sguardo*: attraverso l'eye tracking con HoloLens è possibile selezionare elementi direttamente con lo sguardo.

(a) Tocco dell'ologramma utilizzando l'indice



(b) Interazione attraverso il palmo della mano

Figura 3.2: Possibili gesti con HoloLens 2

3.4 Digital Twins in ambienti Mixed Reality condivisi

In questo capitolo è stato trattato l'argomento circa l'intersezione della Mixed Reality con i Digital Twins in un contesto *single user*. Per riassumere, è stato affermato che l'implementazione dei Gemelli Digitali in un ambiente di Realtà Mista porta numerosi vantaggi, sopratutto se si fa riferimento a un ambiente lavorativo.

L'utente che fa uso di queste tecnologie riesce a interagire con il modo fisico attraverso la sovrapposizione del digitale con il reale, riuscendo a superare i

limiti fisici. In un mondo dove tutto risulta digitalizzato è possibile modellare a proprio piacimento qualunqe cosa, adattandolo ai propri bisogni e necessità.

Fino a ora questa tecnologia interseca un mondo reale che per sua natura è condiviso da più persone, con una realtà virtuale limitata ad un solo utente. Questa peculiarità rende la tecnologia presentata molto limitata. Creando un sistema dove la Mixed Reality interseca i Digital Twins anche in ambienti condivisi, si porta l'utilità di questa tecnologia a un livello ancora superiore.

Si prenda come esempio un contesto lavorativo ospedaliero dove più medici sono occupati con un unico paziente. Senza la possibilità di condividere l'ambiente virtuale ognuno riuscirebbe a interagire con il "proprio" *digital world*. In questo caso si utilizzerebbe questa tecnologia con il solo scopo di creare il proprio desktop olografico.

Pensando invece di ampliare questa situazione con un ambiente condiviso, si potrebbe pensare a uno scenario dove i medici possono visualizzare ologrammi comuni a tutti, in modo da poter cooperare assieme.

La figura riportata successivamente mostra come dei medici possono cooperare assieme in un ambiente di Mixed Reality condiviso.

Figura 3.3: Rappresentazione di un contesto ospedaliero dove i medici utilizzano la Mixed Reality in una situazione di shared experience

In questa sezione non si vuole entrare nei tecnicismi di questo particolare ramo della tecnologia presentata, poiché questo non risulta il core centrale di questa tesi. Si vuole fornire al lettore una presentazione generale di come questa particolare funzione potrebbe portare numerosi benefici circa l'interazione tra Mixed Reality e Digital Twins. Di conseguenza vengono riportati alcuni spunti per eventuali approfondimenti e sviluppi futuri.

3.4.1 Peculiarità di un ambiente condiviso

Grazie al concetto di *Shared Experience* è possibile immaginare una situazione in cui più utenti sono riuniti per affrontare un unico problema. In un ambiente di Mixed Reality condiviso le risorse visualizzate, che in questo caso possono far riferimento a Digital Twins, sono comuni a tutti. Di conseguenza la realtà digitale che viene creata non appartiene a una singola persona, ma risulta condivisa con il gruppo. In una situazione di questo tipo è possibile parlare di *coworking*, dove l'ambiente di lavoro condiviso non è più fisico ma digitale.

Quando si parla di esperienza condivisa non si fa riferimento solo a una condivisione one-to-one, ma si vuole prendere in considerazione una situazione in cui più utenti sono interconnessi. In base al numero di persone presenti nel contesto condiviso, bisogna decidere come e cosa deve essere digitalizzato oltre all'ologramma del Digital Twins a cui si sta facendo riferimento.

Nel caso in cui gli utenti siano presenti fisicamente nello stesso spazio reale, come nell'esempio medico riportato precedentemente, sarà necessario condividere solamente l'ologramma del Digital Twins.

In uno scenario in cui invece gli utenti non sono fisicamente vicini, prendiamo come esempio sempre uno scenario sanitario dove alcuni medici vogliono seguire l'operazione da remoto, è necessario stabilire alcune regole per effettuare la condivisione.

3.5 Verso un Mirror World

I Digital Twins stanno proliferando in molti settori provocando un'irreversibile conseguenza: la creazione di un *Mirror World* o *Mondo Speculare*.

I Digital Twins sono stati inizialmente implementati per la loro capacità di monitorare e simulare il comportamento dei dispositivi. Con il passare del tempo e a seguito dell'aumento esponenziale dei DT sempre più interconnessi fra di loro, si stanno creando modelli digitali d'intere fabbriche o addirittura d'intere città.

In letteratura un *Mirror World* (riferimento all'articolo [11]) è una rappresentazione del mondo reale in forma digitale. Questo termine è differente rispetto al concetto di mondo virtuale, in quanto questo possiede poche connessioni con il mondo reale rappresentando così una simulazione. Il *Mondo Speculare* invece, possiede collegamenti diretti con il modello reale, proprio come i Digital Twins con la loro controparte fisica.

Assimilare il concetto di Mirror World è stato utile per sviluppare l'applicazione presentata nel capitolo successivo.

L'immagine riportata successivamente vuole mostrare una rappresentazione concettuale del Mirror World, sottolineando il fatto che ogni parte che compone il mondo reale possiede una controparte digitalizzata.

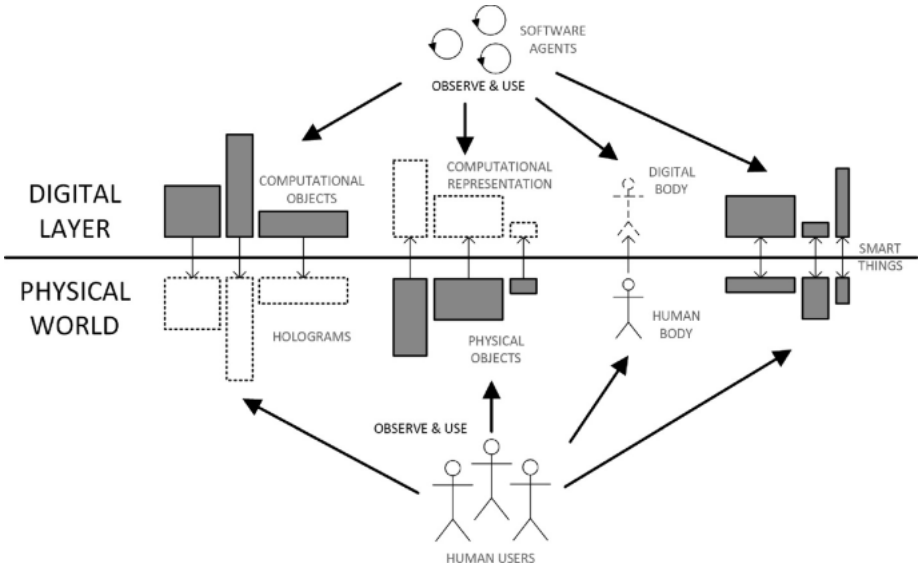


Figura 3.4: Rappresentazione concettuale del Mirror World basato su Software Agent e Digital Twins

Avere una città completamente digitalizzata sia in termini di architettura 3D e sia in termini di comportamenti che avvengono all'interno della città stessa, non sarebbe solo la rappresentazione digitale più completa, ma anche la prima riproduzione computerizzata del nostro mondo visibile all'intelligenza artificiale. Far capire a un software cosa rappresentano gli oggetti di uso comune in un'intera città porterebbe allo sviluppo di nuovi tipi di applicazioni e risulterebbe fondamentale per inaugurare l'industria 4.0 . Questa rivoluzione industriale riguarda quindi la continua interazione e connessione tra il mondo fisico con quello digitale.

Consideriamo per ipotesi che il mondo reale sia già visibile dall'intelligenza artificiale e che esistano dei Digital Twins riguardo tutto il mondo che ci circonda. Questa rete risultante di DT creerebbe enormi quantità di dati utilizzabili sia per sviluppare applicazioni e sia per far comprendere all'intelligenza artificiale come cooperare con il mondo reale.

Molte sarebbero le possibili applicazioni di questa tecnologia.

Un'autovettura autonoma è un veicolo che è in grado di rilevare e analizzare l'ambiente che la circonda per riuscire a raggiungere una destinazione automaticamente, senza l'aiuto di un operatore umano. Un software di simulazione per veicoli autonomi potrebbe sfruttare la tecnologia del *Mirror World*.

Capitolo 4

Un Caso di Studio

In questo capitolo si vuole realizzare un'applicazione a supporto dei medici. Lo scopo principale è quello di progettare una possibile architettura che integra i Digital Twins in un ambiente di Realità Mista con lo scopo di aiutare e supportare gli operatori sanitari con l'adozione di nuove tecnologie.

Con questo progetto non si vuole determinare la migliore architettura per sistemi di questo tipo, ma analizzare e determinare le possibili implementazioni e futuri sviluppi.

Alla base di questo progetto vi è l'esplorazione della tecnologia che unisce la Mixed Reality con oggetti digitalizzati tramite Digital Twins.

Lo studio che si vuole affrontare si incentra sull'idea d'implementare l'uso di dispositivi di Realità Mista in ambito ospedaliero, nello specifico in una sala operatoria.

Riuscire a implementare la Mixed Reality porterebbe numerosi vantaggi. Le applicazioni sarebbero innumerevoli e i benefici ottenuti altrettanto. Porterebbe a un'evoluzione della classica procedura ospedaliera permettendo di applicare nuove tecniche di lavoro e di migliorare quelle già presenti. Aumenterebbe l'efficienza e l'efficacia del lavoro dei medici, perché verrebbero affiancati e supportati da queste nuove tecnologie.

4.1 Descrizione dello scenario

Il caso reale che si vuole affrontare, è quello di digitalizzare e rendere disponibile tramite visore di Mixed Reality al medico chirurgo il monitor multiparametrico.

L'operatore sanitario deve indossare il visore HoloLens 2 e avviare l'applicazione precedentemente installata.

Si è scelto di non dover indicare la sala operatoria in cui ci si trova per rendere il sistema più modulabile e generico possibile.

Il medico può scegliere in qualunque momento di toccare il dispositivo che vuole digitalizzare, come ad esempio il monitor dei parametri vitali, per ottenere la sua controparte olografica. Successivamente è libero di gestire e muovere l'ologramma come meglio desidera per averlo sempre sotto controllo. Le informazioni più importanti del paziente sarebbero così sempre disponibili in qualsiasi momento e aggiornate in tempo reale.

Idealmente questo sistema è possibile applicarlo anche in uno scenario dove sono presenti più medici in una stessa sala operatoria, con ognuno un proprio visore. Ogni operatore sanitario potrebbe così scegliere di visualizzare le informazioni che più desidera posizionate fisicamente nello spazio a lui più comodo. Quest'architettura è stata presentata nella sezione 3.4.

Eventualmente si potrebbero collegare questi dispositivi in modo tale da far visualizzare ai medici il medesimo ologramma comune a tutti. Con questa metodologia però si andrebbe a perdere una caratteristica importante di questo sistema, cioè la completa personalizzazione, non avendo più la possibilità di gestire a proprio piacimento l'ambiente di lavoro. Nell'architettura presentata questa soluzione viene scartata.

4.2 Architettura generale del sistema

Per affrontare il progetto si è scelto di suddividerlo in tre macro sottosistemi distinti riportai in figura 4.1.

Figura 4.1: Suddivisione in tre sottosistemi

Come già esplicitato, questo sistema nasce attraverso l'interazione di diversi sottosistemi originariamente indipendenti. Di conseguenza l'architettura trattata non la si vuole presentare al lettore come la soluzione definitiva, ma si vuole mostrare che lo stato di avanzamento di queste tecnologie è già ad un livello tale da poter sviluppare le prime applicazioni di questo genere.

Il sistema presentato quindi, viene posto come punto di partenza per sviluppi futuri.

Si è scelto di suddividere l'architettura in tre macro sezioni: Physical Asset (PA), Digital Twins (DT) e Hologram (HOLO). Questa scelta è stata fatta per mantenere l'organizzazione logicamente divisa nelle sue parti e per ricordare che questi sottosistemi nascono come *stand alone*.

4.2.1 Physical Asset

Il Physical Asset rappresenta l'oggetto fisico del quale si dovrà creare l'ologramma.

In base all'architettura scelta si deve collegare l'oggetto in una rete interna alla sala operatoria oppure tramite rete Internet. Deve disporre di un codice a barre bidimensionale per essere individuato dal dispositivo per la realtà mista HoloLens 2.

Il PA deve mantenere aggiornato il Digital Twins collegato trasmettendo i dati rilevati.

4.2.2 Digital Twins

Il secondo livello della struttura è formato dal Digital Twin del Physical Asset. Rappresenta il Gemello Digitale dell'oggetto preso in questione. Corrisponde alla copia esatta del modello fisico originale.

Tra i due sottosistemi è necessario che sia presente uno scambio di dati real time per permettere al sistema di essere efficiente.

A seconda della scelta d'implementazione è possibile utilizzare una connessione bidirezionale, se anche dal dispositivo digitale è necessario inviare dati al physical asset, oppure monodirezionale, se la comunicazione deve avvenire solo in un senso. A seguire si effettueranno delle riflessioni in merito alla soluzione da implementare.

Per creare i Gemelli Digitali si è scelto di utilizzare la suite Azure di Microsoft. Nello specifico si andrà a utilizzare Azure Digital Twins per la creazione dei gemelli.

4.2.3 Hologram

L'ultima macro sezione è rappresentata dallo sviluppo dell'ologramma del Digital Twin. Per lo sviluppo si andrà a utilizzare il motore grafico di Unity appoggiandosi al kit di sviluppo multipiattaforma MRTK (trattato nel capitolo 2.4.3) con il supporto a OpenXR.

Per permettere di comunicare con il rispettivo Digital Twin è necessario stabilire una connessione tra i due sistemi. Anche in questo caso può essere bidirezionale o monodirezionale in base alle funzionalità che deve avere il sistema.

Per permettere lo scambio di dati tra queste due sezione è possibile farlo tramite numerose tipologie di connessione. Alcune possibilità possono essere quelle di creare un client-server tra il dispositivo HoloLens e una macchina server. Un'ulteriore possibilità è quella di utilizzare delle Function che mette a disposizione la piattaforma di Azure per accedere ai Digital Twin.

A seguito verranno studiate le possibili soluzioni e verrà implementata la migliore per questo tipo di progetto.

4.3 Progettazione del sottosistema: Physical Asset

In questo progetto si è scelto di prendere in considerazione come Physical Asset il monitor dei parametri vitali. La soluzione che si sta presentando non vuole essere limitata a solo questo tipo di PA. Per generalizzare il sistema e per facilitare la progettazione sperimentale, è stato scelto di emulare un qualunque Physical Asset attraverso l'uso di un Esp32.

Si è scelto di utilizzare Esp32 come emulatore di un Physical Asset per la sua semplicità di collegare sensori e attuatori per lo scambio di dati.

Per simulare il rilevamento dei segnali dei parametri vitali si utilizzeranno dei sensori collegati nei pin GPIO.

Come scenario di lavoro si è pensato di utilizzare un'architettura il più possibile *wireless oriented* con i pregi e le conseguenze che ne derivano. Questa scelta è stata presa considerando l'ambiente di una sala operatoria, dove eventuali cavi potrebbero essere d'intralcio durante un'operazione chirurgica.

Prendiamo in considerazione un monitor multiparametrico di nuova generazione con la possibilità di connetterlo in rete per il trasferimento dei dati. Esp32 con il suo modulo WiFi può simulare questo dispositivo. Sarà necessario predisporre la sala operatoria di una rete WiFi interna per collegare i dispositivi fisici a un server locale.

Effettuata la connessione dei device con il WiFi, questi dovranno essere a conoscenza anche dell' indirizzo Ip della macchina che fungerà da server e della porta che verrà utilizzata per effettuare lo scambio di dati.

Al termine dalla preconfigurazione sarà possibile avviare una connessione TCP per inviare i dati del paziente al server locale.

4.3.1 Esp32 come Physical Asset

Esp32 è un System on a Chip (SoC) a basso costo creato da Espressif Systems che possiede connettività Wi-Fi e Bluetooth. Ulteriori caratteristiche vengono riportate nella figura 4.2 riportata successivamente.

Technical specifications

Power supply voltage (USB)	5V
Input/output voltage	3.3V
Required operating current	Min. 500mA
Soc	ESP32-Wroom 32
Clock frequency range	80 MHz / 240MHz
R.A.M.	512kb
External flash memory	4MB
I/o pins	34
Interfaces	SPI, I2C, I2S, Can, Uart
Wi-fi protocols	802.11 b/g/n (802.11n up to 150 Mbps)
Wi-Fi frequency	2.4 GHz - 2.5 GHz
Bluetooth	V4.2 - BLE and Classic Bluetooth
Wireless antenna	PCB
Dimensions	56x28x13mm

Figura 4.2: Specifiche tecniche della scheda Esp32 NoneMCU

Progettato per dispositivi mobili, elettronica indossabile e applicazioni IoT, ESP32 raggiunge un consumo energetico estremamente basso grazie a funzioni di risparmio energetico.

4.4 Sviluppo del sottosistema: Physical Asset

Per programmare su Esp32 è stato scelto di utilizzare l'IDE Arduino.

Di seguito viene riportato un estratto di codice del Physical Device simulato dall'esp. Come prima cosa viene effettuata la connessione alla rete WiFi interna della sala operatoria. Successivamente viene stabilita la porta e l'indirizzo Ip del server con il quale verrà effettuata la connessione per trasmettere i dati rilevati dal paziente. Come esempio è stato deciso d'inviare il segnale della frequenza cardiaca, la saturazione di ossigeno, la frequenza respiratoria e la temperatura corporea. Questi valori verranno presentati successivamente.

```
#include <WiFi.h>
#include <WiFiMulti.h>

WiFiMulti WiFiMulti;
void setup()
{
    Serial.begin(115200);
    WiFiMulti.addAP(SSD, Password);

    Serial.println("WiFi connected");
    Serial.println("IP address: ");
    Serial.println(WiFi.localIP());

}
void loop()
{
    const uint16_t port = 8080;
    const char * host = "192.168.1.201"; // ip or dns

    Serial.print("Connecting to ");
    Serial.println(host);

    // Use WiFiClient class to create TCP connections
    WiFiClient client;

    if (!client.connect(host, port)) {
        Serial.println("Connection failed.");
        Serial.println("Waiting 5 seconds before retrying...");
        delay(5000);
        return;
    }
```

```
    client.print("PR: " + getPRValue());
    client.print("|TEMP: " + getTEMPValue());
    client.print("|SPO2: " + getSPO2Value());
    client.print("|RR: " + getRRValue());

}
```

Listato 4.1: Estratto del codice client. Viene effettuata la connessione con il WiFi e successivamente avviata una TCP connections al server

In una visione *cable oriented*, tralasciando le complicanze fisiche date dalla presenza di cavi nell'ambiente di lavoro, avremmo però dei benefici in termini di affidabilità e in termini di ritardo del segnale. Una soluzione di questo tipo la si può prendere in considerazione in un'ottica dove la presenza di numerosi cavi non risulta un intralcio per un operatore umano (pensiamo ad esempio in una filiera produttiva) oppure dove l'affidabilità del sistema risulta la caratteristica più importante (vedi ad esempio una situazione critica come potrebbe essere un robot chirurgico).

A seguire viene riportato l'output ottenuto dal Physical Asset. Mostra le informazioni circa l'avvenuta connessione al server e i dati trasmessi. L'aggiornamento avviene con un delay di 1 secondo.

Figura 4.3: Output ottenuto dal Physica Asset simulato da un Esp32. La schermata è stata ottenuta tramite la comunicazione seriale

4.4.1 Leggere i dati del Physical Asset

L'infrastruttura della sala operatoria, oltre ad essere composta da una rete WiFi per il collegamento via wireless dei device, è necessario che sia presente anche una macchina che fungerà da Server per ricevere i dati inviati dai Physical Asset. Il Server deve essere sempre pronto a ricevere nuovi dati dai device.

Il sistema per sua natura deve essere **responsiveness**[1], deve mantenere un comportamento affidabile per tutto il suo ciclo di vita.

Queste specifiche risultano necessarie poiché si avrà a che fare con possibili informazioni critiche.

Immaginiamoci per un momento in uno scenario reale. Ci troviamo all'interno di una sala operatoria durante un intervento chirurgico. Il medico deve tenere sempre sotto controllo i parametri vitali. Sia che queste informazioni siano visualizzate tramite un monitor classico oppure che siano visualizzate tramite un ologramma, è indispensabile che i dati in questione rispettino la realtà attuale. É necessario che le condizioni fisiche del paziente (quali: ECG, frequenza cardiaca, pressione arteriosa, ecc.) siano sempre chiare e disponibili in qualunque momento.

In un sistema come questo è necessario utilizzare un'infrastruttura che utilizzi un protocollo di comunicazione affidabile. In questo caso di studio è stato deciso di utilizzare il protocollo di trasmissione TCP.

Il codice del server TCP è stato scritto in C#. Stabilita la porta e l'indirizzo Ip che i client utilizzeranno per connettersi, verrà aperta la connessione TCP e si rimarrà in attesa.

A seguire viene riportato un estratto di codice della classe che si occupa della creazione del server.

```
 class  cServerClass
{
    public event MessageEventHandler Message;
    public delegate void MessageEventHandler(cServerClass sender,
        string Data);

    //Server Control
    public IPAddress ServerIP = IPAddress.Parse("192.168.1.201");
    public int ServerPort = 8080;
    public TcpListener myserver;

    public Thread Comthread;
    public bool IsLiserning = true;

    //Clients
    private TcpClient client;
    private StreamReader clientdata;

    public cServerClass()
```

[1]In italiano reattività, è un concetto che si riferisce alla specifica di un sistema di completare i compiti assegnati entro un dato tempoi.

```
    {
        myserver = new TcpListener(ServerIP, ServerPort);
        myserver.Start();

        Comthread = new Thread(new ThreadStart(Hearing));
        Comthread.Start();
    }

    private void Hearing()
    {
        while (!IsLiserning)
        {
            if (myserver.Pending() ==true)
            {
                client = myserver.AcceptTcpClient();
                clientdata = new StreamReader(client.GetStream());
            }
            Message?.Invoke(this, clientdata.ReadLine());
            Thread.Sleep(10);
        }
    }
}
```

Listato 4.2: Estratto del codice server scritto in C#. Viene impostato l'indirizzo Ip e la porta del server TCP e rimane in attesa di nuovi messaggi

L'immagine riportata successivamente rappresenta l'applicazione Server in esecuzione. Viene mostrato in una `TextBox` la cronologia dei dati ricevuti dal Physical Asset. Le misurazioni avvengono con una cadenza di un secondo e vengono inviate al server con un unica trasmissione. Attraverso un semplice algoritmo, il messaggio viene suddiviso in quattro parti che corrispondono alle seguenti informazioni:

- PR: la frequenza cardiaca indica il numero di battiti compiuti dal muscolo cardiaco in un minuto.
- TEMP: misurazione in gradi centigradi della temperatura corporea.
- SPO2: la saturazione di ossigeno è un indice ematico che permette di conoscere la percentuale di emoglobina satura di ossigeno, rispetto alla quantità di emoglobina presente nel sangue. La quantificazione di questo valore avviene con calcolo in %, con valori fisiologici compresi tra il 95 – 100%.

- RR: la frequenza respiratoria rappresenta gli atti respiratori compiuti in un minuto.

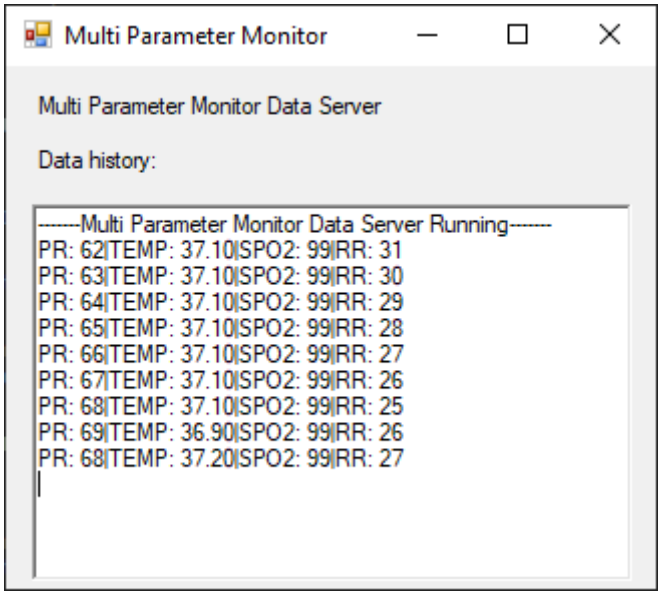


Figura 4.4: Schermata di output del server. Mostra la cronologia dei dati ricevuti.

4.5 Progettazione del sottosistema: Digital Twins

Fino a ora è stato affrontato il problema circa il rilevamento e la trasmissione dei dati di un Physical Asset. É stato detto che i sottosistemi riportati in questo elaborato nascono per essere indipendenti. L'infrastruttura creata fino ad ora potrebbe essere già utilizzata per inviare a terzi, tramite connessione HTTP, i dati rilevati.

L'obiettivo di questo studio non è solamente quello di visualizzare in output informazioni trasmesse via Interent, ma possiede un fine più complesso.

Idealmente sarebbe possibile visualizzare direttamente i dati di un Physical Asset senza integrare l'intera infrastruttura. Questo eliminerebbe però il modello logico ideale che si trova alla base di questo progetto: la modularizzazione e l'interoperabilità dell'intero sistema.

Per proseguire la progettazione è necessario creare il Digital Twin del Physical Asset.

Per il livello di supporto circa l'implementazione del servizio, si è scelto di utilizzare il prodotto offerto da Microsoft presentato nella sezione 1.2.1: Azure Digital Twins.

Come già affrontato in precedenza, idealmente i Digital Twins sono documenti JSON che memorizzano informazioni sullo stato del dispositivo a cui fanno riferimento. L'hub IoT di Azure permette di gestire un DT per ogni dispositivo che viene connesso all' hub.

Nell'architettura presentata, il servizio di Azure Hub IoT non verrà implementato. Come detto in precedenza la sua caratteristica principale è quella di rendere il sistema scalabile, di supportare carichi elevati di lavoro.

Come situazione reale del caso di studio si è scelto di utilizzare una sala operatoria e come esempio di Physical Asset è stato preso in considerazione il monitor dei parametri vitali. Il dominio applicativo di questi scenari non è elevato, di conseguenza l'implementazione di un IoT Hub professionale non porterebbe miglioramenti significativi al sistema. Un eventuale approfondimento di questa tecnologia porterebbe fuori strada lo scopo di questa presentazione. Si è scelto quindi di non implementare il servizio offerto da Azure ma di realizzarne uno adatto a questo scopo.

Ciò non toglie nulla al fatto che in futuro si possa implementare un servizio di Iot Hub professionale, progettando quindi il sistema con una visione più espansiva.

4.5.1 Come interagisce il Digital Twins con il sistema

L'applicazione che si sta realizzando la si può rappresentare come un sistema distribuito, dove un insieme di processi presenti in sottosistemi diversi sono interconnessi tra loro tramite uno scambio di messaggi. Il Digital Twins da realizzare è quel'elemento che nella rappresentazione riportata nell'immagine 4.1 si trova al centro del sistema. Funge da collegamento tra il dispositivo fisico che deve rilevare i dati in input e il sottosistema utilizzato come mezzo di interazione tra l'utente e l'applicazione.

Il DT deve permettere di visualizzare le informazioni e per fare ciò deve mantenere un collegamento con il Physical Asset.

Essendo un'applicazione distrubuita è necessario stabilire dei protocolli di comunicazione tra i vari sottosistemi. In precendenza per quanto riguarda la sezione del Physical Asset è stato scelto di collegare il device in rete per permettere di inviare i dati rilevati in un server interno tramite connessione TCP. Le informazioni ricevute devono mantenere aggiornate le proprietà del rispettivo Digital Twins, di conseguenza è necessario ora stabilire un collegamento tra il server TCP con un'instanza di Azure Digital Twins.

4.6 Sviluppo del sottosistema: Digital Twins

In questa sezione si vuole presentare l'effettiva architettura di questo sottositema. Come detto già in precendenza, l'applicazione che si è scelto di utilizzare per la gestione dell'istanza del Digital Twins appartiene a uno dei servizi offerti da Azure. La soluzione presentata necessita quindi di una connessione a Internet.

4.6.1 Preparare l'ambiente di Azure Digital Twins

Arrivati a questo punto è necessario digitalizzare il Physical Asset creando il suo Digital Twin.

Dopo aver effettuato l'accesso alla piattaforma di Azure ed essere entrati nella sezione Azure Digital Twins, la prima cosa che bisogna fare è creare una nuova istanza. Sul portale in questione aggiungeremo un nuovo Digital Twin impostando un gruppo di risprse, il nome dell'istanza da creare e selezioneremo la regione geografica opportuna. Successivamente bisogna occuparsi di configurare le autorizzazioni di accesso utente. Azure Digital Twins utilizza Azure Active Directory[2] per il controllo degli accessi in base al ruolo di appartenenza. Al termine della configurazione sarà possibile creare effettivamente il Digital Twins.

Microsoft mette a disposizione agli utenti un'applicazione per esplorare e gestire i gemelli digitali: *Azure Digital Twins Explorer*. Successivamente viene riportata in figura 4.5 la schermata principale della piattaforma.

[2]Azure Active Directory è un servizio di gestione delle identità e degli accessi basato sul cloud.[2]

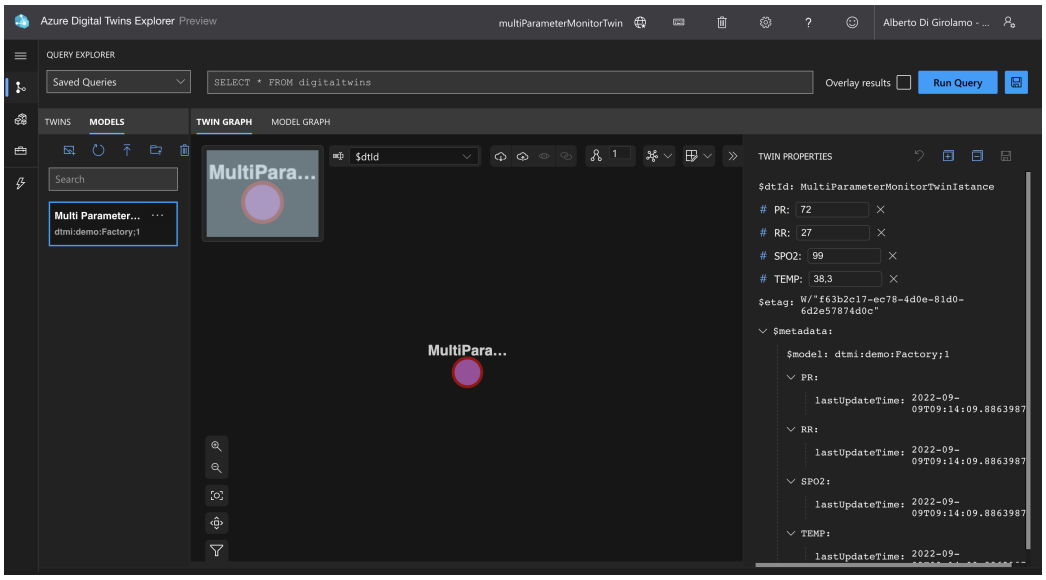


Figura 4.5: Schermata principale di Azure Digital Twins Explorer

Questo strumento consente di connettere le istanze di Digital Twin per visualizzare e modificare i dati raccolti. Per l'autenticazione iniziale è necessario inserire l'URL dell'istanza di Azure Digital Twins a cui si vuole accedere. Successivamente è necessario caricare i modelli. Ogni modello descrive un singolo tipo di entità in termini di proprietà, telemetria, relazioni e componenti. Come già citato in precedenza, il linguaggio di riferimento è chiamato Digital Twin Definition Language (DTDL). Attraverso l'esecuzione di una query riportata successivamente in un linguaggio simile a SQL, è possibile recuperare gli elementi del grafico e visualizzare i dati contenuti.

```
SELECT * FROM digitaltwins T WHERE T.Temperature > 73
```

Listato 4.3: Esempio di query in Azure Digital Twins Explorer per la visualizzazione delle informazioni dei Digital Twins

4.6.2 Aggiornamento dei dati del Digital Twin

Avviato un server per leggere le informazioni real-time dei Physical Asset e dopo aver creato il rispettivo Digital Twin nella piattaforma di Azure, è necessario collegare questi due sottosistemi.

Come soluzione implementata si è scelto di realizzare un'applicazione in C#, a seguire viene riportato un estratto di codice con la relativa spiegazione.

```
using  Azure;
using Azure.DigitalTwins.Core;
using Azure.Identity;

class DigitalTwinsMultiParameterMonitor
{
   // <Async_signature>
   static async Task Main(string[] args)
   {
       // </Async_signature>
       // <Authentication_code>
       string adtInstanceUrl = "https://multiParameterMonitorTwin.api
                                .wcus.digitaltwins.azure.net";

       var credential = new DefaultAzureCredential();
       var client = new DigitalTwinsClient(new Uri(adtInstanceUrl),
           credential);

       //Set property value
       var updateTwinData = new JsonPatchDocument();
       updateTwinData.AppendAdd("/PR", PRvalue);
       updateTwinData.AppendAdd("/RR", RRvalue);
       updateTwinData.AppendAdd("/SPO2", SPO2value);
       updateTwinData.AppendAdd("/TEMP", TEMPvalue);

       await client
                  .UpdateDigitalTwinAsync("multiParameterMonitorTwin",
                     updateTwinData)
                  .ConfigureAwait(false);

   }
}
```

Listato 4.4: Estratto di codice per aggiornare i dati di un Digital Twin.

Per realizzare la soluzione è stato necessario implementare le librerie che forniscono l'accesso al servizio Azure Digital Twins per la gestione dei gemelli, modelli e relazioni : `Azure.DigitalTwins.Core`. Inoltre è necessario implementare la libreria `Azure.Identity` che fornisce il supporto per l'autenticazione attraverso un token di *Azure Active Directory*. La classe che è stata scelta di utilizzare per effettuare l'accesso ai servizi di Azure è *DefaultAzureCredential*. Per autenticare la sessione vengono effettuati diversi tentativi riportati successivamente in figura 4.6.

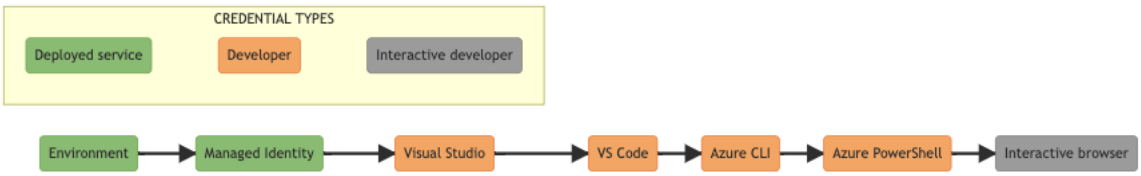


Figura 4.6: Elenco delle tipologie di autenticazione che la classe *DefaultAzureCredential* prova a effettuare.

Successivamente vengono presentati i tentativi di autenticazione che vengono fatti in ordine dalla classe *DefaultAzureCredential*:

1. Environment: tramite variabili d'ambiente.
2. Managed Identity: tramite account utilizzato per distribuire l'applicazione in un host di Azure.
3. Visual Studio: autenticazione tramite le credenziali utilizzate su Visual Studio.
4. Visual Studio Code: autenticazione tramite le credenziali utilizzate su Visual Studio Code.
5. Azure CLI: tramite credenziali utilizzate per effettuare l'accesso nell'interfaccia a riga di comando di Azure, utilizzando `az login`.
6. Azure PowerShell: tramite credenziali utilizzate per effettuare l'accesso nella PowerShell.
7. Interactive browser: tramite autenticazione interattiva nel browser di sistema.

Il campo `adtInstanceUrl` rappresenta la stringa di connessione dell'istanza Digital Twin. Questa è reperibile direttamente nel portale di Azure.

L'effettivo aggiornamento dei dati avviene con il comando `UpdateDigitalTwinAsync` eseguito sull'oggetto client, indicandogli come parametri il nome dell'istanza Digital Twin e il documento JSON.

4.7 Progettazione del sottosistema: Hologram

A questo punto della progettazione l'infrastruttura è composta da un Physical Asset digitalizzato a cui è collegato il rispettivo Digital Twin. Come possiamo mostrare al medico le informazioni ottenute dal Digital Twin del

monitor a parametri vitali? Qua entra in gioco la Mixed Reality. Utilizzando un visore Hololens 2 il medico può decidere dove visualizzare le informazioni attraverso la manipolazione di un ologramma.

Per proseguire con la creazione di questa architettura è necessario presentare delle tecnologie che verranno poi usate successivamente.

Per il progetto in questione è stato scelto di utilizzare Unity come ambiente di sviluppo per realizzare questo sottosistema.

Proseguendo con la progettazione dell'architettura è necessario ora incentrarsi su tutto quello che riguarda l'output del sistema, cioà l'interazione che avrà il medico con l'ologramma collegato al Digital Twin.

Per semplicità progettuale si andranno a visualizzare i dati del monitor a parametri vitali attraverso una casella di testo senza realizzare quindi graficamente un ologramma che rappresenti effettivamente l'oggetto in questione. Ovviamente per rendere il sistema più completo è possibile sviluppare una versione di monitor multiparametrico più esteticamente complesso e gradevole. Questo però non fa parte dell'obiettivo di questa tesi.

4.7.1 Definizione di REST API

Il gemello digitale del monitor a parametri vitali rispecchia completamente il dispositivo fisico, permette di esplorare i dati attuali attraverso chiamate HTTP POST.

RESTful API è un'interfaccia utilizzata da due sistemi informatici per scambiare dati conforme ai vincoli dello stile architetturale REST. Un' API (Application programming interface) definisce le regole da seguire per la comunicazione tra sistemi software. Con il termine REST (Representational State Transfer) indichiamo un'architettura che impone condizioni sull'utilizzo di un'API. Le API che seguono lo stile architetturale REST sono chiamate API REST.

Lo stile architetturale REST impone che il trasferimento delle informazioni avviene tramite un formato standard (JSON, HTML, XLT, Python, PHP o testo semplice). Affinché un'API sia considerata RESTful, deve rispettare i criteri indicati di seguito:

- Possedere un'architettura client-server, con richieste gestite tramite HTTP.
- Una comunicazione client-server stateless[3].

[3]Fa riferimento ad un metodo di comunicazione in cui il server completa ogni richiesta del client indipendentemente da tutte le richieste precedenti. I client possono richiedere risorse in qualsiasi ordine e ogni richiesta è senza stato o isolata da altre richieste. Questi vincoli di progettazione di REST API implicano che il server può ogni volta comprendere e soddisfare del tutto la richiesta.

- Memorizzazione dei dati nella cache per ottimizzare le interazioni.

Utilizzare le REST API rendono il sistema **scalabile** perché REST ottimizza le interazioni client-server, la comunicazione stateless rimuove il carico di lavoro dal server. Le varie componenti del server vengono gestite in modo indipendente tra di loro, permettendo un elevato livello di **flessibilità**, le eventuali modifiche a livello server non comportano cambiamenti al client. Un'altra caratteristica importante delle REST API è quella di essere indipendente circa la tecnologia utilizzata. Il client e il server possono essere progettati con diversi linguaggi di programmazione.

La comunicazione attraverso le REST API avviene seguendo i passaggi elencati:

- Client invia una richiesta formattata correttamente al server.
- Il server autentica il client.
- Il server, se l'autenticazione è avvenuta con successo, riceve la richiesta e la elabora.
- Il server risponde al client con le informazioni richiesto oppure con il relativo errore.

Nell'architettura presentata in questa tesi si utilizzeranno delle richieste HTTP POST, indicando la risorsa da richiamare utilizzando un URL.

La risposte possibili del server possono essere molteplici, di seguito vengono elencati alcuni codici con i relativi significati.

- 200: Risposta di successo generica.
- 201: Risposta del metodo POST avvenuta con successo.
- 400: Richiesta errata che il server non può elaborare.
- 404: Risorsa non trovata.

Il corpo del messaggio viene riportato in un formato ben definito. Di seguito viene riportato un esempio del messaggio ottenuto tramite richiesta all'istanza DT `multiParameterMonitorTwin` creata precedentemente. Il formato del messaggio ottenuto è in JSON.

```
{
  "result": [
    {
      "$dtId": "multiParameterMonitorTwin",
      "$etag": "W/\"de61b953-66b0-401d-ae2d-cfa2b21846c0\"",
      "$metadata": {
        "$lastUpdateTime": "2022-08-31T13:57:21.9165697Z",
        "$model":
            "dtmi:contosocom:DigitalTwins:multiParameterMonitorTwin;1",
        "PR": {
          "lastUpdateTime": "2022-08-31T13:57:21.9165697Z"
        },
        "TEMP": {
          "lastUpdateTime": "2022-08-31T13:57:21.9165697Z"
        },"SPO2": {
          "lastUpdateTime": "2022-08-31T13:57:21.9165697Z"
        },"RR": {
          "lastUpdateTime": "2022-08-31T13:57:21.9165697Z"
        }
      },
      "PR": 75,
      "TEMP": 36.4,
      "SPO2": 99,
      "RR": 30,
    }
  ]
}
```


I client che utilizzano le REST API devono autenticarsi al server per ottenere l'autorizzazione a effettuare le richieste. *Bearer Authentication* è una tecnica di autenticazione che si basa sull'uso di un token. Il token Bearer è una stringa crittografata di caratteri che il server genera in risposta alla richiesta di login. Il client invia il token nell'header della richiesta per accedere alle risorse.

Per richiedere il token è sufficiente effettuare il login nella Azure CLI ed eseguire il comando seguente:

```
az account get-access-token --resource <your-resource>
```

4.7.2 Lettura dei dati del Digital Twins tramite ologramma

Per visualizzare i dati posseduti dal Digital Twin è possibile farlo tramite numerosi metodi. Utilizzando la piattaforma *Azure Digital Twins Explorer* già presentata nel capitolo 4.6.1, è possibile visualizzare graficamente le relazioni che hanno tra di loro i Digital Twins e i rispettivi valori.

Dopo aver effettuato l'accesso con il proprio account Microsoft su Azure CLI, utilizzando il seguente comando:

```
az dt twin query --dt-name multiParameterMonitorTwin --query-command
    "SELECT * FROM DigitalTwins"
```

è possibile visualizzare direttamente in console le informazioni del DT indicato.

Un ulteriore metodo per leggere i dati di un Digital Twin è quello di creare una semplice applicazione in C# come quella presentata in precedenza nel listato 4.4. Dopo aver implementato le dipendenze alle librerie necessarie, aver indicato l'url del Digital Twin di riferimento e dopo aver effettuato l'accesso alla piattaforma è possibile eseguire delle query per leggere i dati del DT. Un esempio di codice da poter implementare al programma presentato precedentemente è riportato in seguito.

```
 string  query = "SELECT * FROM digitaltwins";
AsyncPageable<BasicDigitalTwin> queryResult =
    client.QueryAsync<BasicDigitalTwin>(query);

//Read value property
await foreach (BasicDigitalTwin twin in queryResult)
{
    Console.WriteLine(JsonSerializer.Serialize(twin));
}
```

Listato 4.5: Esempio di codice per leggere i dati aggiornati di un Digital Twin.

4.8 Sviluppo del sottosistema: Hologram

In questa sezione viene riportato lo sviluppo del sottosistema Hologram. L'obiettivo è mostrare tramite ologramma le informazioni ricevute dal Digital Twin collegato.

4.8.1 Sincronizzare i dati dell'ologramma con quelli del Digital Twin

Una soluzione possibile per aggiornare la grafica dell'ologramma e mantenerla sincronizzata con il Digital Twin è quella d'implementare il codice presentato in precedenza nel listato 4.5 in un script C# Unity di una TextMeshPro.

Ipoteticamente questa soluzione dovrebbe andare bene, ma a oggi, purtroppo Unity non offre supporto per leggere direttamente i dati di Azure Digital Twins utilizzando l'SDK. Di conseguenza per ora non è possibile implementare le direttive delle librerie che sono necessarie per collegarsi all'istanza DT.

Come si può quindi interfacciare la richiesta in modo diretto senza utilizzare dipendenze esterne?

Una possibile soluzione la si può trovare utilizzando le REST API di Azure Digital Twins.

4.8.2 Chiamare le REST API da Unity

Come si è potuto apprendere con l'ultima sezione, utilizzare le REST API per mantenere aggiornato l'ologramma risulta per questo progetto, la soluzione vincente.

Esistono ulteriori soluzioni possibili per ricevere aggiornamenti da un Digital Twin dinamicamente. Rimanendo nella suit di Azure ad esempio è possibile utilizzare delle Azure Functions per leggere i dati attraverso trigger HTTP, consentendo quindi di chiamare una funzione attraverso una richiesta Hypertext Transfer Protocol. Successivamente per visualizzare con un ologramma le informazioni real time sarebbe necessario scrivere uno script che attraverso il polling[4] controlli eventuali aggiornamenti di dati.

Si vuole ricordare che il *GameObject* utilizzato in questa architettura è formato da una *TextMeshPro*. Di conseguenza i dati visualizzati avranno una formattazione testuale. Non verranno quindi riportate informazioni grafiche come il segnale ECG formato da un'onda elettrocardiografica.

Aggiunto il *GameObject* per la realizzazione dell'ologramma del monitor multiparametrico nella scena, è possibile implementare uno script C# che ne descrive il suo funzionamento.

Sarà necessario passare nel *body* della richiesta HTTP il JSON contenente la query per l'interrogazione dell'istanza del Digital Twin. Successivamente

[4]Verifica ciclica di presenza di aggiornamenti da parte della CPU

bisogna aggiungere un *headers* con una chiave di tipo *Authorization* e aggiungerci il *Bearer token*. Fatto ciò è possibile eseguire la chiamata HTTP POST e aspettare la risposta. Se non si riceve nessun errore è possibile aggiornare graficamente il *GameObject*.

A seguire viene riportato un estratto di script C# di quanto descritto sopra.

```
async Task Update()
{
    string json = @"{'query':'SELECT * FROM DIGITALTWINS'}";

    var data = new StringContent(json, Encoding.UTF8,
        "application/json");
    var url =
        "https://multiParameterMonitorTwin.api.wcus.digitaltwins
                .azure.net/query?api-version=2020-10-31";
    using var client = new HttpClient();
    client.DefaultRequestHeaders.Authorization = new
        AuthenticationHeaderValue("Bearer", "<TOKEN>");

    var response = await client.PostAsync(url, data);
    var result = await response.Content.ReadAsStringAsync();

    TextMeshPro.SetText(dataValue[i]);
}
```

Listato 4.6: Estratto di codice per aggiornare collegare l'ologramma al suo Digital Twin.

4.8.3 Ottenere l'ologramma del Digital Twin

L'architettura presentata fido a ora fornisce lo scheletro principale per collegare un Physical Asset in rete, creare il suo Digital Twin e visualizzare le sue informazioni tramite ologramma con un visore per la Mixed Reality.

La soluzione ottenuta permette quindi di creare un'applicazione che, una volta avvitata su HoloLens, mostra fin da subito l'ologramma realizzato tramite Unity. Questo sistema potrebbe essere già una soluzione più che accettabile. Tuttavia, analizzando l'abito ospedaliero presentato in questa tesi, l'attuale soluzione non risulta efficiente.

Si consideri per un momento la situazione reale: il medico chirurgo una volta indossato il visore e avviata l'applicazione visualizzerebbe continuamente l'ologramma del monitor multiparametrico. Questa situazione potrebbe risultare scomoda poiché non sempre vi è la necessità di visualizzare i dati olo-

graficamente. É vero anche che il visore per la Mixed Reality HoloLens 2 da la possibilità di sollevare le lenti quando non si vuole adoperare questa tecnologia. Ciò non togle il fatto che questa soluzione risulta limitata e scomoda.

4.8.4 Una possibile soluzione

Una soluzione più adatta a questo problema è quella di abilitare la visualizzazione dell'ologramma desiderato tramite comando gestuale, vocale o eye tracking. Idealmente la soluzione più comoda sarebbe quella di toccare il Physical Asset, dando così la possibilità all'applicazione sul visore di recepire il dispositivo toccato per identificarlo. Successivamente una volta determinato il PA in questione, verrebbe creato l'ologramma permettendo al medico di muoverlo a suo piacimento.

Una architettura di questo tipo però risulterebbe complessa da realizzare. Si dovrebbe implementare un'intelligenza artificiale addestrata a individuare e riconoscere dispositivi medici per permettere di visualizzare l'ologramma selezionato dall'utente. Non sarebbe inapplicabile come sistema, nonostante ciò è stato scelto di non implementare una soluzione di questo tipo per rendere l'architettura del sistema più semplice, poiché questo argomento non fa parte dell'obiettivo di questa tesi.

4.8.5 Utilizzare un QR Code per identificare il Physical Asset

Si è scelto dunque, di realizzare un sistema che si basa sull'utilizzo di codici a barre bidimensionali. In questo caso si è deciso d'implementare un QR Code affianco al Physical Asset da digitalizzare. Così facendo il medico può voltare la testa verso il monitor multiparametrico con affianco in codice bidimensionale. L'applicazione su HoloLens scansiona continuamente l'ambiente circostante tramite le webcam integrate. Una volta identificato il QR Code e letta l'informazione contenuta, è possibile tramite Script C#, stabilire se il codice scansionato fa riferimento ad un Physica Asset. Nel caso in cui lo fosse, apparirebbe un ologrmma con un elemento di input come può essere un bottone. Se il medico decidesse di cliccarlo, tramite script C#, si ricaverebbe l'indirizzo che identifica l'istanza del Digital Twin tramite lettura del codice a barre. Successivamente, una volta realizzata la connessione con la piattaforma Azure e ricevuti i dati aggiornati, verrebbe visualizzato l'ologramma del monitor multiparametrico con i relativi dati.

L'immagine 4.7 riporta successivamente, è stata scattata tramite HoloLens. All'avvio dell'applicazione sviluppata, viene rilevato e scansionato il QRCode, successivamente viene generato un pulsante adiacente. Premendo il bottone

viene generato affianco a esso un ologramma che rappresenta il monitor multiparametrico. La rappresentazione digitale del monitor è già funzionante e rispecchia i dati reali. Se il pulsante viene premuto nuovamente l'ologramma scompare.

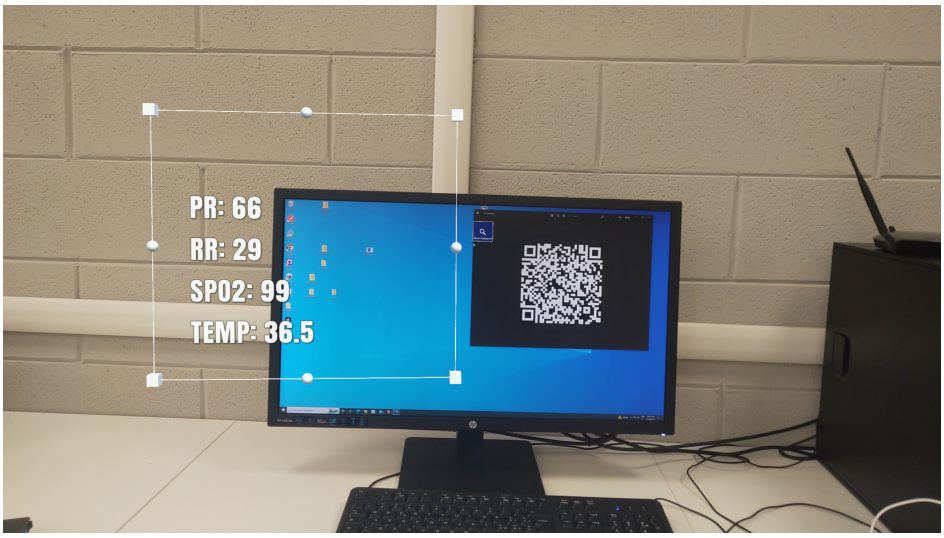


Figura 4.7: Immagine scattata dal dispositivo HoloLens

4.8.6 Ologramma modulabile in base al Digital Twins da visualizzare

Con l'architettura presentata, l'ologramma del monitor dei parametri vitali che il medico visualizza, viene proiettato tramite formattazione testuale. In questo particolare caso di studio, dove il dominio applicativo è basato solamente sulla visualizzazione del monitor multiparametrico, la soluzione presentata risulta la più pratica da implementare.

Se volessimo invece rendere l'applicazione più generica, bisogna fare in modo che in base al dispositivo da digitalizzare, si ottenga una sua versione olografica inerente all'oggetto da rappresentare. Per fare un esempio, se volessimo visualizzare olograficamente una RSM[5], una soluzione che si basa su una visualizzazione testuale non risulterebbe efficace.

Risulta quindi necessario poter visualizzare i Digital Twins con un formato di output più rappresentativo.

Come soluzione generale si potrebbe pensare d'implementare nel QRCode che identifica il Digital Twins, oltre al riferimento del DT stesso per effettuare il collegamento, anche un'informazione riguardante la tipologia di output da

[5]Risonanza magnetica.

rappresentare. In modo da ottenere per ogni coppia *dato-valore* un'etichetta aggiuntiva che indica il formato di output da utilizzare, come: DataText, Image2D, Image3D, Video, ecc.. In modo da ottenere più moduli grafici che si possono poi comporre digitalmente per creare l'ologramma.

L'unione dei diversi moduli ottenuti è poi possibile farlo o tramite script C#, realizzando dei Prefab appositi, oppure in real-time, quando si effettua la scansione del QRCode. Prendiamo come esempio un Digital Twins che fa riferimento a una RSM. La visualizzazione di una risonanza magnetica in ambito ospedaliero risulta una pratica comune, di conseguenza sarebbe utile possedere un ologramma con delle definizioni standard. In questo caso digitalizzando la RSM è possibile ottenere un modulo formato da una TextMeshPro per visualizzare i dati anagrafici del paziente con eventuali note, affiancato ad un'altro modulo dove verrà invece rappresentata in 3D o 2D la risonanza magnetica. In questo caso è possibile pensare di realizzare un Prefab.

Un'altra situazione la si può ottenere quando si scansiona il QRCode di un Digital Twin che non possiede un prefabbricato grafico. In questo caso è possibile pensare a una soluzione che si basa sulla visualizzazione dei diversi moduli informativi "distaccati" gli uni fra gli altri, lasciando al medico il compito di comporre l'ologramma come meglio desidera.

4.9 Soluzioni alternative

Durante la realizzazione del sistema sono state prese in considerazioni diverse possibili soluzioni. In base a determinate scelte, che sono state a loro tempo motivate, è stato deciso di realizzare il sistema che si basa sull'architettura presentata[6], sottolineando il fatto che questa soluzione risulta una ma non l'unica esistente.

4.9.1 Alternative architetturali per il sottosistema Digital Twins

Nel corso della progettazione è stato deciso di non implementare il servizio di Azure chiamato IoT Hub. Questa scelta è stata fatta non perché il suo utilizzo risultava non inerente e inadatto, ma piuttosto perché una sua implementazione non portava una significativa miglioria al sistema. In un'ottica più espansionistica invece, l'implementazione di questo servizio risulterebbe necessario, poiché porterebbe al sistema un elevato livello di scalabilità.

[6]I sottosistemi realizzzati sono disponibili nel repository GitHub https://github.com/AlbertoDiGirolamo/ProgettoTesi

Nell'architettura presentata è stato deciso di realizzare un server interno che fungesse da Hub per il collegamento dei Physical Asset. Successivamente il server si doveva occupare di aggiornare i dati del Digital Twins attraverso l'uso di librerie esterne fornite direttamente da Azure. Un'altro sviluppo possibile di questa parte si basa sull'uso delle REST API anche per aggiornare i dati dei DT, di conseguenza le API non verrebbero utilizzate solo leggere le informazioni tramite script C# da HoloLens.

Se avessimo voluto creare un'infrastruttura senza un dispositivo che fungesse da Hub sarebbe stato possibile. Eliminato questo dispositivo, bisogna fare in modo che sia possibile aggiornare i dati del Digital Twin collegato al Physical Asset direttamente dal codice del PA. Dopo aver effettuato la connessione alla rete WiFi interna della sala operatoria (questa ovviamente deve essere connessa a internet), è possibile utilizzare le REST API direttamente da qui.

Questa non risulta una buona soluzione, in quanto elimina una parte importante dell'architettura: una macchina che funge da "raccoglitore" di tutti i Physical Asset presenti in un singolo contesto, nel nostro caso una sala operatoria. Usufruire di un Hub permette di mantenere organizzati i singoli Digital Twins. In un'ottica futura dove sono presenti numerosi DT, un dispositivo Hub risulta necessario per la gestione complessiva del sistema.

4.9.2 Alternative architetturali per il sottosistema Hologram

Un'altra scelta che è stata fatta durante la progettazione del sistema riguarda la scelta architetturale che collega la parte olografica di HoloLens con il Digital Twin. É stato scelto di non implementare una struttura client-server in loco che collegasse il visore di Mixed Reality a un server interno alla struttura per effettuare il trasferimento dei dati.

Per rendere il sistema meno complesso infatti è stato deciso di utilizzare le REST API offerte da Azure, permettendo quindi di collegare direttamente l'ologramma con il Digital Twin senza utilizzare servizi esterni.

Se avessimo voluto invece realizzare un sistema completamente interno all'ospedale, sarebbe stato necessario realizzare un server interno che gestisse tutti i client presenti, per interfacciarli con i rispettivi Digital Twin. Di conseguenza anche il servizio di Azure non sarebbe stato implementato, ma sarebbe stato necessario realizzare un sistema ad hoc. Ad esempio questa scelta potrebbe essere presa in considerazione se si avesse necessità di evitare problemi di collegamento con la rete Internet. Un possibile scenario sempre in ambito medico chirurgico, potrebbe essere quello di possedere un Digital Twin anche di oggetti utilizzati in situazioni critiche, come bisturi robotizzati, laser chirur-

gici ecc., di conseguenza eventuali problemi di rete potrebbero provocare gravi conseguenze.

Conclusioni

La tesi appena presentata ha permesso di esplorare architetture che integrano tecnologie originariamente indipendenti. É stato realizzato il prototipo di un sistema che integra tecnologie di Realtà Mista incorporando i Digital Twins.

Come progetto è stato scelto di realizzare un'applicazione sanitaria che permettesse al medico chirurgo di visualizzare il monitor multiparametrico collegato al paziente sotto forma di ologramma. Questa soluzione ha come fine la realizzazione di un'applicazione a supporto del medico, permettendo quindi di migliorare la sua efficienza lavorativa.

L'architettura realizzata è stata quindi suddivisa in tre sottosistemi indipendenti: Physical Asset, Digital Twins e Hologram. Queste tre macro sezioni sono state sviluppate indipendentemente e solo successivamente collegate tra di loro per permettere lo scambio di messaggi.

La realizzazione del sistema è stata possibile grazie all'uso di Azure, che ha messo a disposizione un servizio PaaS. Questo ha permesso di alleggerire notevolmente il carico di lavoro. Pur avendo offerto una solida base di partenza per la progettazione dell'architettura, l'implementazione del servizio di Azure Digital Twins non è stato fin da subito chiara. Infatti pur possedendo una discreta documentazione ufficiale, sono stati necessari numerosi tentativi prima di riuscire a realizzare un' architettura con queste caratteristiche.

Sono stati affrontati gli aspetti caratterizzanti di questa architettura come la modularità, l'estendibilità, la composisionalità e l'interoperatibilità. I sottosistemi trattai, nascono per essere indipendenti. Con questa tesi si ha voluto dimostrare che un sistema di questo tipo è realizzabile.

Il servizio offerto da Azure ha reso l'architettura estensibile. Grazie alla potenza computazionale che offre Microsoft, è possibile usufruire della piattaforma con diversi livelli di piani di abbonamento in base alle esigenze del progetto da realizzare. Tramite questi servizi è possibile aggiungere ulteriori funzionalità alla piattaforma senza compromettere il sistema corrente.

Un'altro aspetto, che durante l'analisi iniziale è stato richiesto, era quello di realizzare un'architettura modulabile. Il sistema è stato fin da subito suddiviso in tre macro sottosistemi riguardanti le tre tecnologie principali uti-

lizzate: trasmissione dei dati di un Physical Asset, gestione di un Digital Twin e visualizzazione dei dati ottenuti tramite ologramma.

Con questa tesi si è potuto dimostrare che l'utilizzo della Mixed Reality e dei Digital Twins in un ambiente lavorativo porta numerosi benefici, sia in termini di aumento della qualità del lavoro, e sia in termini di risparmio dei costi, di conseguenza l'uso di questa tecnologia è destinato ad aumentare esponenzialmente.

Per rendere solida questa tecnologia è necessario però definire un'architettura standard a cui fare riferimento. Essendo una tecnologia polifunzionale è importante mantenere un elevato livello di modulabilità dei vari componenti.

Come è stato presentato in questa tesi, ogni modulo dell'architettura deve essere indipendente dal resto del sistema. Ogni macro sezione deve poter esistere anche al di fuori dell'architettura.

Un sistema con queste caratteristiche sarebbe una solida base per numerosi sviluppi futuri.

Ringraziamenti

Vorrei ringraziare il Professor Alessandro Ricci e il Dott. Samuele Burattini nonchè rispettivamente il relatore e il correlatore di questa tesi per avermi seguito durante la fase della stesura.

Ringrazio i miei genitori che mi hanno sempre sostenuto e appoggiato in ogni mia scelta.

Infine un ringraziamento speciale va a tutti i miei amici di una vita e ai compagni di università che mi hanno accompagnato durante tutto questo percorso.

Bibliografia

[1] Jussi Kiljander, Alfredo D'elia, Francesco Morandi, Pasi Hyttinen, Janne Takalo-Mattila, Arto Ylisaukko-Oja, Juha-Pekka Soininen, and Tullio Salmon Cinotti. Semantic interoperability architecture for pervasive computing and internet of things. *IEEE access*, 2:856–873, 2014.

[2] Microsoft. *Che cos'è Azure Active Directory?* `https://docs.microsoft.com/it-it/azure/active-directory/fundamentals/active-directory-whatis`.

[3] Microsoft. *Introduzione alle funzioni di Azure.* `https://docs.microsoft.com/it-it/azure/azure-functions/functions-overview`.

[4] Microsoft. *IoT concepts and Azure IoT Hub.* `https://docs.microsoft.com/en-us/azure/iot-hub/iot-concepts-and-iot-hub`.

[5] Microsoft. *Digital Twin Definition Language (DTDL) per i modelli.* `https://docs.microsoft.com/it-it/azure/digital-twins/concepts-models`, 2022.

[6] Microsoft. *Mixed Reality Toolkit (MRTK).* `https://docs.microsoft.com/it-it/windows/mixed-reality/develop/unity/new-openxr-project-with-mrtk`, 2022.

[7] Microsoft. *OpenXR.* `https://docs.microsoft.com/it-it/windows/mixed-reality/develop/native/openxr`, 2022.

[8] Microsoft. *What is Azure Digital Twins?* `https://docs.microsoft.com/en-us/azure/digital-twins/overview`, 2022.

[9] Azure Microsoft. *Interface examples.* `https://github.com/Azure/opendigitaltwins-dtdl/blob/master/DTDL/v2/dtdlv2.md#property`, 2022.

[10] Mahda Noura, Mohammed Atiquzzaman, and Martin Gaedke. Interoperability in internet of things: Taxonomies and open challenges. *Mobile networks and applications*, 24(3):796–809, 2019.

[11] Alessandro Ricci, Michele Piunti, Luca Tummolini, and Cristiano Castelfranchi. The mirror world: Preparing for mixed-reality living. *Pervasive Computing, IEEE*, 14:60–63, 06 2015.

[12] Concetta Semeraro, Mario Lezoche, Hervé Panetto, and Michele Dassisti. *Digital twin paradigm: A systematic literature review*, volume 130. `https://www.sciencedirect.com/science/article/pii/S0166361521000762`, 2021.