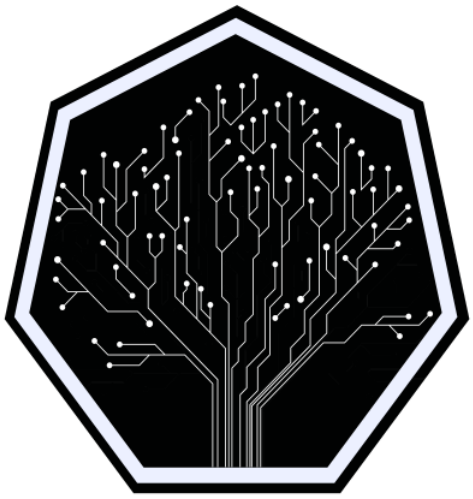

VRAM SOFTWARE

Progetto "Predire in Grafana"

Studio di Fattibilità

07 dicembre 2019

Versione	1.0.0
Approvazione	Vittorio Corrizzato
Redazione	Rebecca Schiavon
	Marco Rampazzo
	Massimo Toffoletto
	Marco Dalla Libera
Verifica	Alessandro Spreafico
	Vittorio Santagiuliana
Stato	Approvato
Uso	Interno
Destinato a	Zucchetti
	Prof. Tullio Vardanega
	Prof. Riccardo Cardin
Email di riferimento	vram.software@gmail.com

Descrizione

Studio di Fattibilità dei capitolati proposti e scelta del capitolato.

Registro delle modifiche

Versione	Data	Nominativo	Ruolo	Descrizione
1.0.0	10-12-2019	Vittorio Corrizzato	Responsabile di progetto	Approvazione finale del documento
0.6.0	09-12-2019	Alessandro Spreafico	Verificatore	Verifica dello studio di fattibilità del capitolato 6
0.5.0	08-12-2019	Vittorio Santagiuliana	Verificatore	Verifica dello studio di fattibilità del capitolato 4
0.4.0	08-12-2019	Vittorio Santagiuliana	Verificatore	Verifica dello studio di fattibilità del capitolato 2
0.3.0	07-12-2019	Alessandro Spreafico	Verificatore	Verifica dello studio di fatibilità del capitolato 3
0.2.0	07-12-2019	Alessandro Spreafico	Verificatore	Verifica dello studio di fattibilità del capitolato 5
0.1.0	07-12-2019	Vittorio Santagiuliana	Verificatore	Verifica dello studio di fattibilità del capitolato 1
0.0.9	09-12-2019	Marco Dalla Libera e Alessandro Spreafico	Analista e Verificatore	Correzione e modifica struttura documento
0.0.8	06-12-2019	Marco Dalla Libera	Analista	Stesura dello studio di fattibilità del capitolato 6
0.0.7	05-12-2019	Rebecca SchiavonS	Analista	Stesura dello studio di fattibilità del capitolato 5
0.0.6	05-12-2019	Marco Rampazzo	Analista	Stesura dello studio di fattibilità del capitolato 3
0.0.5	05-12-2019	Rebecca Schiavon	Analista	Stesura dello studio di fattibilità del capitolato 1

0.0.4	04-12-2019	Massimo Toffoletto	Analista	Stesura dello studio di fattibilità del capitolato 4
0.0.3	04-12-2019	Massimo Toffoletto	Analista	Stesura dello studio di fattibilità del capitolato 2
0.0.2	03-12-2019	Marco Rampazzo	Analista	Definita struttura interna del documento
0.0.1	03-12-2019	Marco Dalla Libera	Analista	Creato documento LaTeX con il titolo

Indice

1 Introduzione

1.1 Scopo del documento

Nel seguente documento verranno analizzati i capitolati$_G$ presentati nell'ambito del progetto di Ingegneria del Software. Saranno esposte le ragioni che hanno portato il gruppo alla scelta del capitolato 4 (*Predire in Grafana*) e alla conseguente esclusione delle altre proposte.

1.2 Glossario

Questo documento verrà corredato da un *Glossario v.1.0.0* dove saranno illustrati i termini tecnici o altamente specifici per evitare ambiguità in essi. Le voci interessate saranno identificate da una 'G' a pedice.

1.3 Riferimenti

1.3.1 Normativi

- **Norme di Progetto**: *Norme di Progetto v1.0.0.*

1.3.2 Informativi

- **Capitolato$_G$ d'appalto C1 - Autonomous Highlights Platform**: https://www.math.unipd.it/~tullio/IS-1/2019/Progetto/C1.pdf
- **Capitolato$_G$ d'appalto C2 - Etherless**: https://www.math.unipd.it/~tullio/IS-1/2019/Progetto/C2.pdf
- **Capitolato$_G$ d'appalto C3 - NaturalAPI**: https://www.math.unipd.it/~tullio/IS-1/2019/Progetto/C3.pdf
- **Capitolato$_G$ d'appalto C4 - Predire in Grafana**: https://www.math.unipd.it/~tullio/IS-1/2019/Progetto/C4.pdf
- **Capitolato$_G$ d'appalto C5 - Stalker**: https://www.math.unipd.it/~tullio/IS-1/2019/Progetto/C5.pdf
- **Capitolato$_G$ d'appalto C6 - ThiReMa - Things Relationship Management**: https://www.math.unipd.it/~tullio/IS-1/2019/Progetto/C6.pdf

2 Capitolato 1 - Autonomous Highlights Platform

2.1 Informazioni generali

- **Nome**: Autonomous Highlights Platform;
- **Proponente**: Zero12;
- **Committente**: Prof. Tullio Vardanega e Prof. Riccardo Cardin;

2.2 Descrizione

Autonomous Highligths Platform prevede lo sviluppo di una piattaforma web capace di ricevere in input dei video di eventi sportivi e fornire in output un video contenente i momenti salienti.

2.3 Obiettivi del progetto

Il progetto prevede la creazione di un software che riceve in input dei video di eventi sportivi come una partita di calcio, Formula1, Motogp o altri sport e riesca a creare in autonomia un video di massimo 5 minuti con i soli suoi momenti salienti (highlights) da fornire in output. A questo fine la piattaforma dovrà essere dotata di un modello di machine learning$_G$ in grado di identificare ogni momento importante dell'evento. In particolare il flusso di generazione del video sarà così organizzato:

- Caricamento del video dell'evento desiderato;
- Individuazione degli highlights;
- Estrazione delle parti individuate dal video originario;
- Generazione del video dei momenti salienti.

2.4 Requisiti di progetto

Al fine del corretto svolgimento del progetto sarà necessario rispettare i seguenti vincoli forniti dall'azienda proponente:

- Utilizzo di Sage Maker$_G$ per la costruzione del modello di intelligenza artificiale$_G$;
- Strutturazione di un'architettura a multiservizi$_G$;

- Permettere il caricamento dei video tramite riga di comando;
- Creazione di un'interfaccia web per l'analisi e il controllo dello stato di elaborazione del video.

2.5 Tecnologie interessate

Le tecnologie previste per la realizzazione del capitolato sono:

- **Amazon web services**$_G$, in particolare:
  - **AWS Elastic Container Service**$_G$ come servizio per la gestione dei contenitori ad alte prestazioni;
  - **AWS Dynamo DB**$_G$, database non relazionale in NoSQL$_G$ per il supporto e la gestione dei tag;
  - **AWS Elastic Transcoder**$_G$ per la conversione e la rielaborazione dei video;
  - **AWS Sage Maker**$_G$ per la creazione di un modello per il riconoscimento degli eventi salienti se si sceglie di implementare un addestramento supervisionato dell'intelligenza artificiale;
  - **AWS Rekognition**$_G$ da usare nel caso si scelga, invece, l'apprendimento non supervisionato e quindi l'utilizzo di modelli già esistenti.
- **NodeJS**$_G$ per lo sviluppo di API Restful JSON$_G$ al fine di garantire una scalabilità ottimale;
- **Python**$_G$ per lo sviluppo delle componenti necessarie di machine learning;
- **HTML5**$_G$, **CSS3**$_G$ e **Javascript**$_G$ per lo sviluppo dell'interfaccia web.

2.6 Aspetti positivi

- Le tecnologie proposte risultano innovative e utili per la loro larga diffusione nel mondo del lavoro;
- La documentazione delle tecnologie reperibile è ben approfondita;
- Le specifiche del capitolato sono state fornite in modo chiaro e preciso.

2.7 Criticità e fattori di rischio

- La maggior parte delle tecnologie proposte, seppure molto interessanti, non è prevista dal nostro corso di laurea. Dunque lo svolgimento di questo capitolato richiederebbe quindi un numero di ore di apprendimento difficilmente quantificabile.
- Si è ritenuto non semplice e dispendioso in termini di tempo il reperimento e l'analisi dei video per l'apprendimento della macchina

2.8 Conclusioni

Lo scopo del capitolato è risultato accattivante, tuttavia la quantità di nuove tecnologie da apprendere ha demotivato il gruppo a scegliere questo progetto.

3 Capitolato 2 - *Etherless*

3.1 Informazioni generali

- **Nome**: Etherless;
- **Proponente**: Red Babel;
- **Committente**: Prof. Tullio Vardanega e Prof. Riccardo Cardin.

3.2 Descrizione

Etherless è una *Cloud Application Platform* che permette agli sviluppatori che la utilizzano di caricare nel cloud delle funzioni *JavaScript*. Tali procedure possono poi essere acquistate da terzi tramite l'impiego della criptovaluta *Ethereum* implementata grazie alla tecnologia della *blockchain*.

3.3 Obiettivi di progetto

Il progetto si propone di aiutare gli sviluppatori fornendo:

- un framework *Serverless* che gestisca il costo computazionale della funzione;
- un servizio di *Smart Contracts* che sovrintenda il processo di pagamento.

Quest'ultima tecnologia sarà di supporto anche per l'acquirente in quanto assicurerà il completamento della transazione solo a lavoro verificato e completato.

3.4 Requisiti di progetto

È richiesto di dividere lo sviluppo in tre fasi:

- **Local**: utilizzo dell'applicativo in ambiente locale, in cui può essere utilizzato *Ethereum testrpc* di *Truffle* per l'emulazione della blockchain;
- **Test**: utilizzo dell'applicativo in ambiente di testing, in cui può essere usata la soluzione proposta al punto precedente;
- **Staging**: utilizzo dell'applicativo in un ambiente pubblico, in tal caso si potrà usufruire di *Ropsten Ethereum* come rete di testing.

Inoltre è obbligatorio separare il software in tre parti:

- **etherless-cli**: modulo attraverso il quale lo sviluppatore interagisce con Etherless;
- **etherless-smart**: modulo per l'interazione tra *etherless-cli* e la parte server;
- **etherless-server**: modulo che ascolta gli eventi emessi da *etherless-smart* per attivare le rispettive funzioni lambda.

Ognuna di queste operazioni dovrà poi essere caricata e versionata tramite *GitHub* o *GitLab*.

3.5 Tecnologie interessate

Per l'implementazione delle varie funzionalità vengono date dall'azienda delle linee guida sulle tecnologie da utilizzare:

- **Typescript 3.6**: linguaggio di programmazione da impiegare, tramite l'approccio *Promise* o *async-await*, nello sviluppo della piattaforma *Etherless*;
- **Solidity**: linguaggio per la creazione e la gestione degli *Smart Contracts*;
- **AWS Lambda**: piattaforma computazione serverless fornita da *Amazon* per la coordinazione degli eventi;

- **Serverless Framework**: framework Web per la creazione di applicazioni su *AWS Lambda*;
- **typescript-eslint**: strumento di analisi statica del codice *Typescript*.

AWS Lambda può essere corredata con altri componenti quali: *AWS API Gateway* (per eventi HTTP), *AWS DynamoDB* (database non relazionale) o *AWS S3* (servizio di memorizzazione). Tutto questo poi può essere gestito tramite *AWS CloudFormation* (piattaforma per l'organizzazione delle risorse AWS).

3.6 Aspetti positivi

- Ethereum e la tecnologia della blockchain più in generale sono tematiche molto attuali e innovative che suscitano interesse tra i componenti del gruppo anche per le possibili applicazioni future;
- L'azienda ha avvallato delle richieste chiare e precise, aspetto valutato positivamente dal gruppo.

3.7 Criticità e fattori di rischio

- L'azienda ha sede in Olanda, il rapporto con i proponenti potrebbe essere quindi meno efficace;
- Seppur siano proposte tematiche interessanti il gruppo ha presentato dei dubbi riguardanti la difficoltà e le tempistiche per l'approfondimento delle tecnologie presentate.

3.8 Conclusioni

Il capitolato è apparso stimolante per quanto riguarda le tecnologie utilizzate e convincente nella sua esposizione. Tuttavia la distanza geografica e la mole di lavoro prevista sono risultate un ostacolo nell'effettiva realizzazione del software. Per questo motivi il gruppo ha deciso di vertere la sua scelta su un altro progetto.

4 Capitolato 3 - NaturalAPI

4.1 Informazioni generali

- Nome: NaturalAPI;

- Proponente: Teal Blue;
- Committente: Prof. Tullio Vardanega e Prof. Riccardo Cardin;

4.2 Descrizione

L'obbiettivo di questo capitolato è quello di far parlare a tutti gli stakeholders$_G$ un linguaggio comune, in modo da velocizzare ed evitare confusione durante la progettazione. Teal Blue vuole utilizzare le specifiche e i requisiti di un software, scritti in linguaggio naturale (Inglese, Italiano, ecc.), per generare API$_G$.

4.3 Finalità del progetto

Il prodotto da sviluppare è **NaturalAPI** un toolkit$_G$ che dovrà generare API$_G$ complete e test automatici a partire da un linguaggio naturale. Per la realizzazione di questo prodotto dovranno essere sviluppati tre PoC$_G$ o feature$_G$.

1. **NaturalAPI Discover:** *estrattore di BDL$_G$.*
   L'estrattore *Discover* ha lo scopo di estrarre entità, processi e combinazioni tra essi da un documento testuale di business.
2. **NaturalAPI Design:** *parser$_G$ di scenari e caratteristiche.*
   Questa PoC$_G$ dovrà creare una BAL$_G$ API$_G$ in tempo reale a partire dai documenti di Gherkin$_G$ e da un BDL$_G$.
3. **NaturalAPI Develop:** *esportatore di linguaggio.*
   Questa feature$_G$ dovrà convertire un BAL$_G$ in casi di test e API$_G$ nel linguaggio di programmazione scelto, supportando la creazione e l'aggiornamento di nuove repository$_G$.

Ogni feature$_G$ sopraindicata dovrà essere accessibile attraverso almeno due dei seguenti modi: interfaccia da linea di comando, GUI$_G$ minimale o un'interfaccia web REST$_G$. La parte logica del prodotto finale dovrà essere esportata in una delle seguenti modalità: come una libreria, come parte di un eseguibile o come un processo indipendente locale o remoto.

4.4 Tecnologie interessate

- NLP$_G$ o Natural Language Processing cioè un trattamento informatico del linguaggio naturale, che si occupa della realizzazione di sistemi in grado di comprendere il linguaggio naturale

- Dependency Parser$_G$ cioè un parser che si occupa di analizzare la struttura grammaticale di una frase in linguaggio naturale per identificare le relazioni tra parole chiavi e parole che le modificano
- BDD$_G$ Behaviour Driven Development che è un processo di sviluppo software agile che ha, alla sua base, una continua comunicazione tra tutti gli stakeholders$_G$ di un progetto informatico
- Hiptest$_G$ e Cucumber$_G$ che sono degli strumenti software che supportano il processo BDD$_G$; il primo serve per eseguire test automatici, mentre il secondo legge le specificazioni software in linguaggio naturale, scritte con alcune regole di sintassi (Gherkin$_G$), e controlla che il software rispetti i requisiti
- Generazione API$_G$ e DLS$_G$ utilizzando:
  - OpenAPI$_G$: uno standard per descrivere API$_G$
  - Swagger$_G$: un framework$_G$ che aiuta a sviluppare servizi web REST$_G$
  - OWL$_G$: un linguaggio web semantico
- Un qualsiasi framework$_G$ a scelta come Qt$_G$, React$_G$, ecc.

4.5 Aspetti positivi

- Il proponente non impone vincoli sui linguaggi di programmazione da utilizzare, questo permette al gruppo di sviluppare un prodotto con un linguaggio o un framework$_G$ considerato più interessante.
- I requisiti, le tecnologie e il modo di esportare il prodotto finale sono spiegati in modo preciso e esauriente.
- L'azienda Teal Blue si dimostra disponibile a incontri ed a una comunicazione aperta con il fornitore.

4.6 Criticità e fattori di rischio

- Questo capitolato non ha suscitato molto interesse nel gruppo a causa dell'eccessiva astrattezza del capitolato e per il fatto che i concetti su cui prepararsi non erano, per il gruppo VRAM Software, stimolanti.
- Le tecnologie da imparare sono molte ed è arduo quantificare il tempo necessario per raggiungere una preparazione sufficiente per gestire in modo produttivo le tecnologie elencate.

4.7 Conclusione

Il gruppo ha trovato interessante l'idea di Teal Blue di rendere la comunicazione tra tutti stakeholders$_G$ chiara e veloce, tuttavia abbiamo deciso di orientarci verso un progetto meno astratto.

5 Capitolato 4 - Predire in Grafana

5.1 Informazioni generali

- Nome: Predire in Grafana$_G$: Monitoraggio predittivo per DevOps$_G$;
- Proponente: Zucchetti;
- Committente: Prof. Tullio Vardanega e Prof. Riccardo Cardin;

5.2 Descrizione

Con questo capitolato, Zucchetti mira a realizzre due plug-in$_G$ per lo strumento di monitoraggio Grafana$_G$che applichi gli algoritmi di Support Vector Machine(SVM)$_G$ e Regressione Lineare(RL)$_G$ al flusso di dati che sono stati ricevuti per allarmi o segnalazione tra la linea di produzione del software e le segnalazioni tra gli operatori del servizio Cloud$_G$.

5.3 Requisiti e finalità di progetto

Il sistema richiede due plug-in$_G$ di Grafana$_G$ scritti in Javascript$_G$ che eseguiranno i calcoli letti da un file json$_G$ e produrranno valori tali da essere aggiunti al flusso di monitoraggio. In particolare viene richiesto di:

- Produrre un file json$_G$ dai dati di addestramento con i parametri per le previsioni attraverso SVM$_G$ per le classificazioni o RL$_G$
- Leggere la definizione del predittore$_G$ dal file in formato json$_G$
- Associare i predittori$_G$ letti dal file json$_G$ al flusso di dati presente in Grafana$_G$
- Applicare la previsione e fornire i nuovi dati ottenuti dalla previsione al sistema di Grafana$_G$
- Rendere disponibili i dati al sistema di creazione di grafici e dashboard$_G$ per la loro visualizzazione

Opzionalemente è richiesto di implementare le seguenti funzionalità:

- Offrire la possibilità di definire degli "alert$_G$" in base a livelli di soglia raggiunti dai nodi collegati alle previsioni
- Fornire i dati che definiscono bontà dei modelli di previsione utilizzati
- Offrire la possibilità di applicare delle trasformazioni alle misure lette dal campo per ottenere delle regressioni$_G$ esponenziali o logaritmiche
- Offrire la possiblità di addestrare SVM$_G$ o RL$_G$ direttamente in Grafana$_G$
- Implementare dei meccanismi di apprendimento di flusso, per poter disporre di sistemi di previsione che si adattano costantemente ai dati rilevati sul campo
- Utilizzare altri metodi di previsione che possano dare benefici alla previsione dei dati, tra cui la versione delle SVM$_G$ adattate alla regressione$_G$ o piccole reti neurali$_G$ per la classificazione

5.4 Tecnologie interessate

- **Javascript**: linguaggio di programmazione richiesto per costruire i plug-in$_G$ di Grafana$_G$ ed eseguire i calcoli tramite SVM$_G$ e RL$_G$ per fare le previsioni
- **Grafana**: sistema di monitoraggio open-source$_G$ estendibile con plug-in$_G$ javascript$_G$
- **Libreria Javascript per reti neurali**: librerie che permettono di implementare reti neurali$_G$ per migliorare le previsioni, utili a soddisfare uno dei requisiti opzionali

5.5 Aspetti positivi

- Svolgere questo capitolato permette di acquisire competenze in ambito di analisi e previsione dei dati, molto richieste in ambito lavorativo
- La presentazione del problema è stata molto chiara e l'azienda è apparsa molto disponibile.
- Gli algoritmi di SVM$_G$ e RL$_G$ verranno forniti dall'azienda

5.6 Criticità e fattori di rischio

- Gli algoritmi da utilizzare sono basati su concetti matematici e ciò ha demotivato il gruppo per lo svolgimento
- Servono delle basi di machine learning$_G$ non fornite nel nostro corso di laurea

5.7 Conclusione

Il gruppo ha trovato molto chiaro e definito il problema, ma i concetti matematici richiesti per affrontarlo hanno demotivato l'interesse del gruppo verso questa scelta.

6 Capitolato 5 - Stalker

6.1 Informazioni

- **Nome**: Stalker
- **Proponente**: Imola Informatica
- **Committente**: Prof. Tullio Vardanega

6.2 Descrizione

La soluzione software proposta da Imola Informatica prevede lo sviluppo di un'applicazione mobile e un'infrastruttura server di supporto, con lo scopo di monitorare le presenze in forma anonima o autenticata in uno o più luoghi circoscritti. Il progetto nasce dalla necessità di tracciare il numero di persone presenti all'interno dei locali, richiesta dalle normative, perciò gli ambiti di utilizzo potranno variare, dal controllare l'affluenza in luoghi di interesse al verificare le presenze del personale nel luogo di lavoro.

6.3 Requisiti

Un'applicazione mobile con i seguenti requisiti:

- Recupero lista organizzazioni
- Possibilità di effettuare login tramite LDAP$_G$per organizzazioni che lo richiedono

- Storico accessi
- Visualizzazione in tempo reale della propria presenza all'interno di un luogo monitorato e cronometro del tempo trascorso al suo interno

Un'interfaccia web per l'amministrazione con le seguenti funzionalità:

- Funzionalità di login
- Creazione/modifica/eliminazione di organizzazioni
- Aggiunta/modifica/rimozione di luoghi, definiti da coordinate geografiche
- Configurazione collegamento server LDAP$_G$
- Invio di notifiche push$_G$ alle applicazioni per segnalare l'aggiornamento delle liste di organizzazioni e luoghi
- Monitoraggio del numero di utenti presenti nei luoghi dell'organizzazione in tempo reale
- Ricerca sugli accessi dei dipendenti e creazione di report sulla frequentazione dei luoghi
- Gestione delle autorizzazioni per gli utenti dell'interfaccia web

L'infrastruttura su cui si baserà l'applicazione dovrà sottostare ai seguenti vincoli:

- Comunicazioni solo all'entrata od uscita dall'area interessata
- Cifratura di tutte le comunicazioni tra app e server
- Architettura scalabile$_G$ (verticalmente e orizzontalmente) e tollerante a picchi di traffico per il lato server
- Test di carico che dimostrino il funzionamento in varie situazioni

6.4 Tecnologie interessate

- Java$_G$o Swift$_G$(applicazione mobile)
- NodeJS$_G$o Python$_G$(back-end$_G$)
- Utilizzo di protocolli asincroni$_G$ per le comunicazioni tra app e server

- HTML5$_G$, CSS3$_G$e Javascript$_G$(interfaccia web lato server)
- Utilizzo del pattern di Publisher/Subscriber$_G$, ovvero mittenti e destinatari dimessaggi dialogano attraverso un tramite(dispatcher)
- Utilizzo dell'IAAS Kubernetes o di un PAAS, Openshift o Rancher, per il rilascio delle componenti server
- API REST$_G$ esposte dal server, o gRPC$_G$in alternativa
- GPS$_G$/sistemi ibridi di geolocalizzazione
- LDAP$_G$(Lightweighgt Directory Access Protocol)
- Test unitari$_G$ e d'integrazione per tutte le componenti applicative

6.5 Aspetti positivi

- Lo sviluppo di applicazioni mobili è una conoscenza richiesta
- Argomento interessante, anche grazie alla potenziale utilità nell'ambito della sicurezza

6.6 Criticità e fattori di rischio

- I requisiti sono numerosi e richiedono conoscenze non banali
- Lo sviluppo lato server potrebbe portare complicazioni, viste le numerose tecnologie coinvolte e i requisiti in scalabilità
- La precisione ottenibile con le tecnologie attuali non è sufficiente per rendere l'applicazione usabile nella realtà
- Nella presentazione è stato discusso il funzionamento del sistema GPS ma completamente tralasciata la funzionalità dei backend di geolocalizzazione presenti nei moderni sistemi mobile (ad esempio i Google Mobile Services per Android)

6.7 Conclusioni

Nonostante il nobile obiettivo, la difficile fattibilità di un prodotto concreto e la possibilità di complicazioni scoraggia il gruppo nella decisione d'intraprendere questo capitolato.

7 Capitolato 6 - ThiReMa

7.1 Informazioni generali

- Nome: ThiReMa;
- Proponente: Sanmarco Informatica;
- Committente: Prof. Tullio Vardanega e Prof. Riccardo Cardin;

7.2 Descrizione

Lo scopo di questo progetto è di creare un'applicazione web che permetta di valutare la correlazione fra dati operativi$_G$ e fattori influenzanti$_G$in ambito IoT$_G$. I dati saranno ricavati da dispositivi IoT$_G$ eterogenei tramite piattaforma Apache Kafka$_G$, quindi il software si occuperà di raccolta, storicizzazione, monitoraggio, analisi e presentazione dei dati. L'obiettivo finale del progetto è di rendere i processi aziendali intelligenti trasformando i dati in informazioni, così da permettere ad esempio la pianificazione di manutenzioni predittive calcolate su informazioni reali, in automatico.

7.3 Obiettivi di progetto

Realizzare il prodotto **ThiReMa**, una web application che raccoglierà dati da dispositivi IoT$_G$ eterogenei tramite piattaforma Apache Kafka, li monitorerà e storicizzerà in un time series database$_G$ e li presenterà elaborati all'utente tramite interfaccia web. Inoltre dovrà inviare notifiche tempestive agli operatori tramite un servizio basato su Telegram$_G$e permettere, sempre tramite tale servizio, di comandare i dispositivi IoT$_G$ stessi.

- **Piattaforma Kafka**: Questa piattaforma di stream-processing$_G$ distribuita si occuperà della raccolta dei dati dai vari sensori e del loro indirizzamento al database. Sarà inoltre usata per leggere, elaborare e presentare gli storici dei dati stessi.
- **Time series database**: Questi database sono molto efficienti per archiviare dati IoT$_G$ in quanto usano il timestamp$_G$ della lettura come chiave, a cui poi basterà associare il valore letto. Oltre ai dati delle misurazioni dovranno essere storicizzati i loro metadati$_G$ ed i dati degli utenti del sistema. Alcuni esempi consigliati sono: PostgreSQL$_G$, TimescaleDB$_G$, ClickHouse$_G$.

- **Interfaccia web**: Questa interfaccia dovrà permettere la visualizzazione dei dati raccolti, la loro correlazione e la loro gestione. Dovrà permettere di gestire i sensori, gli impianti e gli utenti del sistema.

Ogni feature$_G$ dovrà essere istanziata tramite uso della tecnologia di contanerizzazione$_G$ Docker$_G$, così da rendere il più possibile le componenti del sistema manutenibili singolarmente.

7.4 Tecnologie interessate

- Kafka$_G$: piattaforma di stream-processing$_G$ distribuito, permette di raccogliere e monitorare flussi di record di dati come se fossero una coda di messaggi. Permette anche la storicizzazione e l'elaborazione di questi stream di dati.

- Time series database$_G$: sono database molto efficienti nella storicizzazione di dati IoT$_G$, in quanto occupano poca memoria pur mantenendo le informazioni basilari necessarie. A questi database è possibile affiancare altri database per contenere i metadati dei dispositivi IoT$_G$ ed i dati degli utenti. Alcuni possibili database sono PostgreSQL$_G$, TimescaleDB$_G$, ClickHouse$_G$.

- JAVA$_G$: popolare linguaggio di programmazione interpretato ed orientato agli oggetti. È consigliato il suo uso per realizzare la business logic$_G$ del programma tramite piattaforma Kafka$_G$.

- Bootstrap$_G$: popolare framework$_G$ CSS utilizzato per realizzare l'interfaccia grafica di siti ed applicazioni web. È consigliato il suo uso per la realizzazione dell'interfaccia web dell'applicazione.

- Docker$_G$: è una tecnologia di contaneirizzazione$_G$ che permette di eseguire in modo efficiente più applicativi software in ambienti dedicati ed isolati risiedenti in un'unica macchina fisica. Tali ambienti sono detti container e permettono di eseguire più servizi in un'unica macchina fisica rendendoli indipendenti gli uni dagli altri, è quindi possibile fermare un container lasciando gli altri regolarmente attivi.

7.5 Aspetti positivi

- Sviluppo di competenze nell'ambito IoT$_G$, molto richieste dal mondo del lavoro ed interessanti per i membri del gruppo.

- Sviluppo di conoscenze in ambito Big Data$_G$ ed analisi dei dati, molto richieste attualmente dal mondo del lavoro e probabilmente anche in futuro.
- Utilizzo di Java, un linguaggio di programmazione molto popolare e previsto da un insegnamento del nostro corso di studi.
- Architettura del sistema ben definita e componenti ben chiare.
- L'ambito del capitolato è interessante per tutti i membri del gruppo.

7.6 Criticità e fattori di rischio

- Numero elevato di componenti da realizzare ed integrare che comportano di conseguenza molte tecnologie da apprendere.
- Probabile necessità di effettuare test in sede aziendale per avere accesso ai dispositivi fisici.

7.7 Conclusioni

Il gruppo ha trovato l'ambito di applicazione del problema interessante ed ha apprezzato la chiarezza con cui sono definite le varie parti del sistema. Proprio l'elevato numero di componenti da realizzare ha però fatto calare la volontà dei membri del gruppo a svolgere questo progetto.