NetBreak

Progetto API Market

Studio di Fattibilità

Informazioni sul documento

Nome del file	StudioDiFattibilita 0_2_1.pdf
Data di creazione	02 Dicembre 2016
Ultima modifica e versione	08 Dicembre 2016 Versione 0.2.1
Stato	Non approvato
Redatto da	Nicolò Scapin Davide Scarparo Alberto Nicolè Andrea Scalabrin Marco Casagrande Dan Serbanoiu
Verificato da	Davide Scarparo Alberto Nicolè
Approvato da	Andrea Scalabrin
Uso	Interno
Distribuzione	NetBreak
Destinato a	Prof. Tullio Vardanega, Prof. Riccardo Cardin, NetBreak

Abstract

Questo documento raccoglie lo studio di fattibilità dei capitolati proposti per l'A.A. 2016/2017. Lo studio consiste in un'analisi generale per ogni capitolato, al fine di individuare tecnologie e punti critici.

Changelog

Descrizione	Autore e Ruolo	Data e versione
Modifiche al capitolato C3	Nicolò Scapin Analista	2016-12-08 0.2.1
Verifica del documento	Alberto Nicolè Verificatore	2016-12-07 0.2.0
Modifiche dei paragrafi sulla base della verifica	Davide Scarparo Analista	2016-12-06 0.1.1
Verifica del documento	Davide Scarparo Verificatore	2016-12-06 0.1.0
Accorpati i documenti e modifiche minori	Andrea Scalabrin Analista	2016-12-05 0.0.9
Stesura del capitolato C2	Marco Casagrande Analista	2016-12-03 0.0.8
Stesura del capitolato C5	Alberto Nicolè Analista	2016-12-03 0.0.7
Stesura del capitolato C4	Davide Scarparo Analista	2016-12-03 0.0.6
Stesura del capitolato C3	Nicolò Scapin Analista	2016-12-03 0.0.5
Stesura del capitolato C6	Dan Serbanoiu Analista	2016-12-03 0.0.4
Stesura del capitolato C1	Andrea Scalabrin Analista	2016-12-02 0.0.3
Stesura dell'introduzione	Andrea Scalabrin Analista	2016-12-02 0.0.2
Creato template documento	Andrea Scalabrin Analista	2016-12-02 0.0.1

Indice

1 Introduzione

1.1 Scopo del documento

Lo scopo del documento è quello di presentare le motivazioni e la breve analisi che ha indirizzato il gruppo verso la scelta del capitolato C1. Tutti i capitolati vengono analizzati con la medesima metodologia, evidenziandone in particolare le tecnologie necessarie, gli aspetti cruciali e una fase conclusiva contenente il giudizio e le opinioni del gruppo.

1.2 Scopo del prodotto

Lo scopo del prodotto è la realizzazione di un API Market per l'acquisto e la vendita di microservizi. Il sistema offrirà la possibilità di registrare nuove API per la vendita, permetterà la consultazione e ricerca ai potenziali acquirenti, gestendo i permessi tramite creazione e controllo di relative API key. Il sistema, oltre alla webapp stessa, sarà corredato di un API Gateway per la gestione delle richieste e il controllo delle chiavi, e fornirà funzionalità avanzate di statistiche per il gestore della piattaforma nonchè per gli utilizzatori.

1.3 Riferimenti normativi

- NormeDiProgetto 1_0_0.pdf

1.4 Riferimenti informativi

- **Capitolato C1:** APIM: An API Market Platform
  `http://www.math.unipd.it/~tullio/IS-1/2016/Progetto/C1.pdf`
- **Capitolato C2:** AtAVi: Accoglienza tramite Assistente Virtuale
  `http://www.math.unipd.it/~tullio/IS-1/2016/Progetto/C2.pdf`
- **Capitolato C3:** DeGeOP: A Designer and Geo-localizer Web App for Organizational Plants
  `http://www.math.unipd.it/~tullio/IS-1/2016/Progetto/C3.pdf`
- **Capitolato C4:** eBread: applicazione di lettura per dislessici
  `http://www.math.unipd.it/~tullio/IS-1/2016/Progetto/C4.pdf`
- **Capitolato C5:** Monolith: an interactive bubble provider
  `http://www.math.unipd.it/~tullio/IS-1/2016/Progetto/C5.pdf`
- **Capitolato C6:** SWEDesigner: editor di diagrammi UML con generazione di codice
  `http://www.math.unipd.it/~tullio/IS-1/2016/Progetto/C6.pdf`

1.5 Glossario

Per semplificare la consultazione e disambiguare alcune terminologie tecniche, alcune voci indicate con la lettera G a pedice sono descritte approfonditamente nel documento apposito Glossario 1_0_0.pdf

2 Capitolato C1 (Scelto)

2.1 Descrizione

Il capitolato proposto da ItalianaSoftware riguarda la creazione di una web app per la gestione di microservizi. Le funzionalità che dovranno essere fornite saranno la vendita e l'acquisto di microservizi, tramite un apposito marketplace, e per ogni microservizio, sarà fornita dal fornitore l'interfaccia pubblica delle proprie API. Il progetto non richiede soltanto di gestire la compravendita dei microservizi attraverso il marketplace, ma anche di integrare delle funzioni di controllo tramite un API Gateway. Quest'ultimo dovrà essere in grado di effettuare un'analisi statistica dei dati di utilizzo più rilevanti e regolare l'accesso alle API registrate tramite opportune API key, limitando l'utilizzo alle API a coloro che non sono in possesso di una chiave valida. La creazione della web app sarà affrontata tramite l'utilizzo delle consuete tecnologie per lo sviluppo web lato front-end e back-end. La peculiarità del progetto è la realizzazione dell'API Gateway in linguaggio Jolie, come da richiesta del committente. Jolie, infatti, rappresenta un nuovo emegente linguaggio di programmazione open-source orientato ai microservizi.

2.2 Dominio applicativo

Lo scopo di questo prodotto è fornire la possibilità a sviluppatori e utilizzatori di avere a propria disposizione un valido strumento per l'acquisto e la vendita regolamentata di microservizi. Questo garantisce degli standard qualitativi al cliente finale, che può valutare e scegliere ciò che più rispetta le proprie esigenze. Il bacino di utenza riguarda, dunque, tutte le aziende e gli sviluppatori che si affacciano al mondo dei microservizi, oltre a coloro che li metteranno a disposizione.

2.3 Tecnologie

Per la realizzazione di questo capitolato sono necessarie conoscenze di base per lo sviluppo di applicativi web. Nel nostro particolare caso la scelta può ricadere su:

- **HTML5 e CSS3** per la struttura e l'aspetto grafico.
- **Bootstrap 3** come framework CSS.
- **Javascript e jQuery** per la parte comportamentale front-end.
- **PHP 7** per le funzionalità back-end.
- **Oracle MySQL** come database SQL.
- **Jolie** per la realizzazione dell'API Gateway

2.4 Aspetti critici

L'aspetto cruciale nella realizzazione del progetto riguarda, secondo la nostra analisi, la corretta implementazione dell'API Gateway, che permetta le funzioni richieste dal committente. La volontà di introdurre un Service Level Agreement (SLA) per i servizi, e la conseguente necessità di gestione avanzata delle statistiche tramite API Gateway, evidenziano come questa parte del progetto sia in realtà il punto su cui è necessario prestare la massima attenzione.

2.5 Considerazioni conclusive

Il capitolato è stato designato come il più allettante da parte del nostro gruppo per numerosi aspetti. Si manifesta, come prima cosa, l'interesse condiviso da parte di tutti i componenti per le tecnologie inerenti alla parte web. Allo stesso tempo, acquisire un nuovo linguaggio di programmazione per questa sfida risulta un'esperienza formativa e interessante: di fatto, oggigiorno, queste nuove tecnologie vengono sempre più utilizzate, anche da grosse aziende (vedi Amazon, Netflix, Google, Facebook, etc.). Infine, il gruppo ha manifestato un forte interesse nell'intraprendere un'attività lavorativa con un'azienda emergente, nata proprio da un progetto universitario.

3 Capitolato C2

3.1 Descrizione

Il capitolato proposto da zero12 riguarda la creazione di un'applicazione web che permetta agli ospiti di un'azienda di interrogare un assistente vocale dedicato all'accoglienza. Il software sfrutterà il sistema di comunicazione già esistente, basato su Slack, per notificare l'arrivo dei visitatore. Tra le funzionalità obbligatorie da fornire, vi è la richiesta dei dati personali dell'ospite e di sue eventuali necessità (moduli, informazioni, caffè...), che verranno inoltrate all'interessato tramite Slack. Il software dovrà utilizzare obbligatoriamente la lingua inglese per adattarsi agli assistenti virtuali già disponibili. L'applicativo sarà suddiviso in tre parti: l'interfaccia web per interagire con l'utente, i servizi AWS Lambda per interagire con l'API dell'assistente virtuale e l'interfacciamento. Sarà importante non solo la realizzazione del software, ma anche uno studio del mercato attuale e l'analisi del comportamento degli utenti, così da migliorarne la prestazione e la soddisfazione nell'utilizzo.

3.2 Dominio applicativo

Il progetto risulta interessante per qualunque organizzazione che voglia gestire in modo automatico i visitatori presso la sua sede. In particolare, è destinato a tutte le aziende che utilizzano Slack: un punto di forza vista la sua ampia diffusione. Inoltre, il recente interesse nel campo della sintesi vocale potrebbe fornire un trampolino di lancio al progetto. Se il software saprà garantire la soddisfazione dei visitatori e miglioramenti nell'organizzazione dell'azienda utilizzatrice, potrà espandersi anche in altri settori.

3.3 Tecnologie

Le tecnologie consigliate per il raggiungimento degli scopi del progetto proposto sono:

- **HTML5, CSS3 e Javascript** per lo sviluppo dell'interfaccia Web
- **Bootstrap 3** come framework CSS
- **NodeJS o Swift** come linguaggio di programmazione per lo sviluppo dell'applicazione
- **Express** come eventuale framework NodeJS
- **Amazon Web Services** come infrastruttura di cloud computing
- **MongoDB o DynamoDB** come database NoSQL
- **SiriSDK o AlexaSDK** come assistenti virtuali

3.4 Aspetti critici

Un punto cruciale del progetto riguarda la capacità di progettare correttamente le interfacce rivolte ai servizi AWS Lambda. Questi gestiranno lo scambio di dati tra gli utenti e gli altri elementi interni quali Slack, database ed assistente virtuale. L'apprendimento e l'integrazione di un numero relativamente alto di nuove tecnologie potrebbe, inizialmente, disorientare e far perdere la visione d'insieme del progetto. Di natura totalmente diversa è l'analisi del comportamento degli utenti; un attività che richiede flessibilità nel rivalutare il proprio punto di vista, mediando tra le necessità di sviluppatori ed utenti, e la disponibilità di un numero ragionevole di volontari, il quanto più differenti tra loro.

3.5 Considerazioni conclusive

Il capitolato richiede la conoscenza e lo studio di un buon numero di tecnologie attuali e diffuse, permettendo nel contempo di realizzare un'applicazione concreta ed utile. Nonostante ciò, alcuni punti che l'hanno resa meno appetibile ai nostri occhi sono:

- Sfruttare in modo superficiale le tante tecnologie richieste in fase di sviluppo;
- La maggior parte delle tecnologie da utilizzare richiede molto tempo per la comprensione e l'apprendimento.

conta nel progetto è la comunicazione tra esse. Vista l'eterogeneità del gruppo, si è deciso di puntare ad un capitolato che impiegasse un minor numero di differenti tecnologie, ma che allo stesso tempo permettesse un utilizzo approfondito e dedicato. Inoltre, l'analisi di mercato e lo studio del comportamento degli utenti appaiono come compiti non banali per chi non sia accuratamente informato riguardo al settore della sintesi vocale. Per i motivi sopracitati, dunque, questo capitolato è passato in secondo piano rispetto agli altri proposti.

4 Capitolato C3

4.1 Descrizione

Il capitolato proposto da RiskApp riguarda la simulazione di eventuali danni economici dovuti a un evento esterno che può interrompere il business di una azienda. L'applicazione web deve poter offrire all'utente la possibilità di inserire tramite appositi form un piano dettagliato dei processi produttivi. Una volta inviati i dati al server di back-end e l'algoritmo RiskApp simulerà come un certo tipo di danno può impattare nel bilancio aziendale in termici di fatturato e quota di mercato. L'algoritmo fornirà un grafo con la struttura del processo produttivo e i possibili scenari di danno che possono colpire l'azienda e il relativo danno economico. Si richiede che la Web app restituisca il grafo sia integrato in una mappa geografica e sia fruibile tramite dispositivo mobile e in particolare su tablet, con supporto alle gestures tipiche e l'inserimento con riconoscimento vocale.

4.2 Dominio applicativo

Lo scopo del progetto è fornire una Web app per l'inserimento dei dati e la successiva elaborazione per quantificare il danno economico causato da una interruzione del processo produttivo di una azienda. La Web App sarà quindi utile anche in ambito assicurativo, in quanto permettere di quantificare correttamente l'importo assicurativo di una polizza.

4.3 Tecnologie

- **Bootstrap, Javascript con framework React, hammer.js e Yeoman** per quanto riguarda la part front-end
- **Django** per quanto riguarda la parte back-end
- **Python3** per conoscere il prodotto attuale di RiskApp
- **PostgreSQL e noSQL** come database
- **RStudio** con Shiny per quanto riguarda il processamento dati

4.4 Aspetti critici

La nostra analisi ha portato a individuare diversi punti critici. Il primo riguarda trasportare il processo produttivo dell'azienda, restituito come grafo dall'algoritmo, su una mappa geografica, con evidenziato il percorso della merce e i luoghi di lavorazione e deposito della merce. Crediamo che questo processo sia complesso e soggetto a possibili ed eventuali errori sulla creazione della mappa geografica. Un secondo aspetto critico può essere individuato riguardo la fruizione della Web app anche offline, quindi l'incapsulamento dei dati e il successivo invio una volta collegati alla rete. L'ultimo riguarda la restituzione dei dati nello stesso formato di input: i dati inviati al server dell'elaborazione devono essere convertiti e la lettura nel formato di input necessita un'ulteriore decodifica. Questo aspetto potrebbe essere risolto in parte con lo studio dell'algoritmo di RiskApp.

4.5 Considerazioni conclusive

Il capitolato genera interesse e curiosità poiché approfondisce un argomento di cui, purtroppo, ultimamente sentiamo spesso al telegiornale a causa dei recenti terremoti nel centro Italia. In questi luoghi La maggior parte delle piccole e medie aziende ha difficoltà nello stipulare una polizza assicurativa perché, come accennato dal committente, è complicato stipulare una polizza vista l'imprevedibilità di fenomeni naturali, quali terremoti e della loro intensità. La Web app

viene in aiuto sia a queste aziende che alle compagnie di assicurazione, ma di fatto non siamo a conoscenza dell'algoritmo e solo la comprensione dello stesso e del suo codice, oltre ad uno studio di Python 3, potrebbe chiarirci le idee. Pertanto, al gruppo sono rimasti molti dubbi sulla possibile e reale implementazione del prodotto richiesto e del suo possibile utilizzo offline, tali da indurci a scartare la scelta di questo progetto.

5 Capitolato C4

5.1 Descrizione

Il capitolato proposto da Mivoq consiste nella realizzazione di un'applicazione in ambiente Android per dispositivi mobili, come smartphone e tablet, che sia in grado di agevolare la lettura alle persone affette da dislessia, grazie all'aiuto della sintesi vocale. Per rendere il prodotto finale quanto più riusabile, dovranno essere progettate due componenti ben distinte: una libreria per accedere alle funzionalità di sintesi vocale e alle informazioni per la sincronizzazione a partire dal testo, e un'applicazione, che potrà consistere in un lettore di e-book o un client di messaggistica. Dovranno essere implementate obbligatoriamente le seguenti funzionalità: lettura di almeno una sorgente di testo; con conseguente riproduzione dell'audio sintetizzato, ed evidenziazione del testo in modo sincronizzato rispetto all'audio (in stile karaoke). Altre funzionalità che possono essere prese in considerazione per una eventuale implementazione sono: possibilità di cambiare la voce e la lingua, possibilità di modicare la velocità di riproduzione dell'audio e possibilità di modicare la visualizzazione del testo da leggere (ad esempio, colori dello sfondo e del font, spaziatura fra i caratteri, dimensione e tipo dei caratteri, layout di visualizzazione).

5.2 Dominio applicativo

Il prodotto finale è destinato a tutte le persone affette da dislessia, ovvero con evidenti problemi nella lettura di semplice testo, sia esso in qualsiasi formato, in modo da dimostrare l'efficacia della sintesi vocale. Infatti, questa tecnologia, ad oggi, non è sfruttata al massimo delle sue capacità, ma è conosciuta solo per il suo utilizzo in ambiti come le voci guida dei navigatori satellitari, gli annunci dei mezzi di trasporto pubblico, i centralini telefonici, i comandi vocali sui dispositivi mobili, etc...

5.3 Tecnologie

Le tecnologie consigliate per il raggiungimento degli scopi del progetto proposto sono:

- **FA-TTS** (**F**lexible and **A**daptive **T**ext-**T**o-**S**peech), come motore di sintesi vocale open-source;
- **Android**, come piattaforma ed ambiente di sviluppo;
- **Java**, come linguaggio di programmazione per lo sviluppo dell'applicazione (vincolato dalla scelta di Android come piattaforma);
- **ePub**, come libreria open-source per lo sviluppo di un e-book reader;
- **Telegram**, come client di messaggistica open-source.

5.4 Aspetti critici

Uno degli aspetti critici è che, generalmente, i servizi di sintesi vocale su dispositivi mobili non forniscono le informazioni necessarie alla sincronizzazione di audio e testo: in questo specifico caso, l'azienda committente suggerisce l'utilizzo dell'engine FA-TTS, premettendo che, nonostante anche questo motore di sintesi non fornisca informazioni per la sincronizzazione, sia semplice da modificare in modo tale che le fornisca. La modifica si presume venga fatta da Mivoq, tuttavia si richiede uno studio sulla configurazione e sull'utilizzo delle funzionalità del motore di sintesi, tecnologia del tutto sconosicuta al gruppo.

5.5 Considerazioni conclusive

Questo capitolato non è stato scelto perchè, seppure la sintesi vocale sia una tecnologia in forte sviluppo ed utilizzo, la realizzazione di un'applicazione in ambiente Android destinata a dispositivi mobili, non è per niente banale. Android è una piattaforma molto vasta, ed essendo per il gruppo il primo approccio a tale ambiente, è richiesta una buona e solida formazione autodidatta, che in termini di tempo potrebbe risultare insufficiente al fine di sviluppare l'applicazione richiesta. Inoltre, la dislessia è un forte punto su cui bisogna ragionare, affinchè si possa produrre qualcosa che effettivamente venga utilizzato nell'immediato con buoni risultati. Quindi, dato che l'applicazione richiede un alto livello di flessibilità ed usabilità rispetto al target di utenti, a nostro avviso occorre una buona esperienza in merito e una formazione minima su Android e il suo interfacciamento con il motore di sintesi vocale, il quale è il core del progetto.

6 Capitolato C5

6.1 Descrizione

Il capitolato proposto da Red Babel tratta la creazione di un framework Monolith, implementato come package di Rocket.chat. Lo scopo di questo framework è fornire agli sviluppatori uno strumento per l'implementazione di bolle interattive all'interno del sistema di messaggistica Rocket.chat. Queste bolle interattive nascono dalla neccessità sempre più frequente di trasmettere informazioni in modo sempre più evoluto e frequente. Sono stati individuati 3 tipi di bolle interattive, la cui implementazione è richiesta obbligatoriamente::

- Rich media bubble: rappresentano quei contenuti come link a video o a pagine web, dei quali è utile mostrare direttamennte il contenuto all'interno di una bolla nel sistema di messaggistica;
- Self-updating bubble: rappresentano quei contenuti dei quali è interessante consultare lo stato anche a distanza di tempo dalla condivisone, come può essere la variazione di prezzo di un articolo o le previsioni del tempo;
- Editing bubble: il contenuto più comune, appartenetente a questo tipo di bolle, è il sondaggio. In questo caso un utente può creare un sondaggio composto da un quesito e da alcune riposte, mentre gli altri utenti hanno la possibilità di scegliere una o più delle risposte.

6.2 Dominio

Questo framweork si offre a tutti gli sviluppatori che necessitano di introdurre un sistema più avanzato per lo scambio di dati, come può essere un'azienda che vuole offrire una customer communication dashboard ai suoi clienti come strumento di assistenza o un bot che risponda in modo automatico alle domande più frequenti.

6.3 Tecnologie

Per la realizzazione di questo capitolato è necessario l'utilizzo delle seguenti tecnologie:

- **Javascript 6th edition (ES6)** per lo sviluppo della parte backend del framework;
- **Angular 2** per la parte frontend del framework;
- **HTML5, CSS3 e SASS** (CSS prepocessor) per la struttura e l'aspetto grafico;

6.4 Aspetti critici

Questo package deve essere facilmente installabile in un server dove risiede un'istanza di Rocket.chat, ad esempio attraverso un package manager. Le API che il framework deve fornire dovranno dare la possibilità allo sviluppatore di implementare una grande varietà di bolle interattive, che potranno essere poi inserite in un contesto più complesso come appunto una customer communication. Questo potrà generare delle difficoltà nel riconoscimento del tipo di contenuto da visualizzare e nel modo in cui dovrà essere visualizzato.

6.5 Considerazioni conclusive

Il capitolato tratta un argomento sicuramente interessante ed attuale, vista l'enorme mole di dati che ogni giorno viene scambiata tra utenti di tutto il mondo. In questo caso specifico non riteniamo cosi interessante l'implementazione di questo framework poichè grandi colossi mondiali, Telegram in primis, ma anche Whatsapp ed Apple stanno implmentando queste funzioni con risultati notevoli e in continuo miglioramento.

7 Capitolato C6

7.1 Descrizione

Il capitolato proposto da Zucchetti consiste nella creazione di un editor UML, che a partire dal diagramma delle classi, diagramma delle attività e altre possibili tipologie di diagrammi, generi il codice corrispondente all'applicazione descritta. Il linguaggio di implementazione di un progetto generato automaticamente potrà essere Java o Javascript. Le caratteristiche peculiari del progetto riguardano la possibilità non solo di generare codice, ma possibilmente anche operare delle decisioni che sotto forma di input al generatore di codice, dia risultati diversi nel codice prodotto, a parità di linguaggio per diagrammi di uno stesso progetto. E' necessario integrare anche un sistema di machine learning che, tenendo conto di quali sono le modifiche usualmente effettuate post generazione codice, renda sempre piu' definitivo il codice prodotto, diminuendo le modifiche necessarie da parte dell'utilizzatore dell'editor.

7.2 Dominio applicativo

Lo scopo di questo prodotto e' fornire agli sviluppatori software uno strumento per l'implementazione automatica totale o parziale del codice di un progetto, a partire dai diagrammi UML dello stesso. Il codice generato utilizzando pattern di progetto risultera' standardizzato, fornendo una base comune piu' facilmente estendibile o modificabile dagli implementatori. Gli utilizzatori stimati sono l'amministratore, i progettisti, i programmatori e i verificatori di un progetto software.

7.3 Tecnologie

Sono necessarie conoscenze estese per lo sviluppo di applicativi web:

- **HTML/CSS/JS** per il lato Client (obbligatorio).
- **Framework** e **Librerie** a scelta open source.
- **Java** o **Javascript** per il lato Server (obbligatorio).
- **OrientDB** o altro database NoSQL a grafo.

7.4 Aspetti critici

Gli aspetti cruciali nello sviluppare un generatore automatico di software consistono nel trovare un'associazione automatica accettabile tra lo specifico problema descritto nei diagrammi e il codice generato. Si deve evitare un incremento di complessità ed è necessario mantenere la consistenza dei diagrammi al modificarsi del codice. Infatti, una successiva estensione degli stessi non deve alterare riscritture o modifiche importanti da parte degli implementatori del codice generato.

7.5 Considerazioni conclusive

Il capitolato e' stato scartato dal nostro gruppo per un serie di motivi. E' richiesta anzitutto una conoscenza approfondita del linguaggio UML nonchè di una forte esperienza di utilizzo, che nessuno di noi ancora possiede. Inoltre, è necessaria la conoscenza di un gran numero di design pattern e della loro corretta applicazione a seconda dei differenti contesti di utilizzo. É anche necessario integrare un servizio di machine learning, che a partire dal codice, renda compilazioni successive dei diagrammi di uno stesso progetto o progetti diversi sempre piu' efficienti e precise e nessuno del nostro gruppo ha esperienza in questo. Infine, occorre cimentarsi nell'apprendimento dei database non relazionali (NoSQL).