AtAVi

Definizione di Prodotto v1.0.0

Sommario
Questo documento specifica e descrive strumenti, regole e convenzioni utilizzate dal gruppo Co.Code nel corso della realizzazione del *progetto*$_{\text{g}}$ AtAVi.

Versione	1.0.0
Data di redazione	2017-02-02
Redazione	
Verifica	
Approvazione	
Uso	Esterno
Distribuzione	prof. Tullio Vardanega prof. Riccardo Cardin

Diario delle modifiche

Versione	Riepilogo	Autore	Ruolo	Data

Indice

Elenco delle figure

Introduzione

Scopo del documento

Lo scopo di questo documento consiste nella definizione in dettaglio della struttura e funzionamento delle componenti del progetto AtAVi. Questo documento sarà usato come guida dai *Programmatori*del gruppo.

Scopo del prodotto

Si vuole creare un'applicazione web che permetta ad un ospite, in visita all'ufficio di Zero12, di interrogare un assistente virtuale per annunciare la propria presenza, avvisare l'interessato del suo arrivo sul sistema di comunicazione aziendale (*Slack*$_g$) e nel frattempo essere intrattenuto con varie attività.

Glossario

Allo scopo di evitare ogni ambiguità nel linguaggio e rendere più semplice e chiara la comprensione dei documenti, viene allegato il "*Glossario v1.0.0*". Le parole in esso contenute sono scritte in corsivo e marcate con una 'g' a pedice (p.es. *Parola*$_g$).

Riferimenti

Riferimenti Normativi

- "*Norme di Progetto v2.0.0*";
- "*Analisi dei Requisiti v2.0.0*";

Riferimenti Informativi

- Design patterns:
  - strutturali: http://www.math.unipd.it/ tullio/IS-1/2016/Dispense/E04.pdf;
  - creazionali: http://www.math.unipd.it/ tullio/IS-1/2016/Dispense/E05.pdf;
  - comportamentali: http://www.math.unipd.it/ tullio/IS-1/2016/Dispense/E06.pdf;
  - architetturali: http://www.math.unipd.it/ tullio/IS-1/2016/Dispense/E08.pdf.
- Slide dell'insegnamento - Diagrammi delle classi: http://www.math.unipd.it/ tullio/IS-1/2016/Dispense/E02a.pd
- Slide dell'insegnamento - Diagrammi dei packages: http://www.math.unipd.it/ tullio/IS-1/2016/Dispense/E02b.pdf;
- Slide dell'insegnamento - Diagrammi di sequenza: http://www.math.unipd.it/ tullio/IS-1/2016/Dispense/E03a.pdf;

Standard di progetto

Architettura dell'applicazione

Diagrammi riassuntivi dei package

Specifica dei componenti

Diagrammi di sequenza

Tracciamento

Design Patterns

Architetturali

Architettura a microservizi

- **Scopo:**l'architettura a microservizi è un approccio allo sviluppo di una singola applicazione come insieme di piccoli servizi, ciascuno dei quali viene eseguito da un proprio processo e comunica con un meccanismo snello, spesso una HTTP API;
- **Vantaggi:**
  - ogni microservizio è relativamente piccolo, quindi più semplice da implementare e da capire per gli sviluppatori;
  - ogni microservizio è indipendente dagli altri; è quindi possibile distribuire nuove versioni più frequentemente e isolare i possibili errori.
- **Svantaggi:**
  - l'architettura risulta maggiormente complessa perchè risulta essere un sistema distribuito;
  - la gestione di più microservizi potrebbe risultare in un carico di lavoro maggiore rispetto ad una sua versione monolitica.
- **Utilizzo:**

Architettura event-driven

- **Scopo:**anche se non è un vero e proprio pattern, l'architettura event-driven è un particolare tipo di architettura asicrona per sistemi distribuiti basata sugli eventi.
- **Vantaggi:**
  - per definizione, questo tipo di architettura è particolarmente adatto ad ambienti di tipo asicrono basati sugli eventi, come l'interazione con gli utenti.
- **Svantaggi:**
  - i sistemi che utilizzano tale architettura sono spesso distribuiti: ciò comporta un maggiore livello di complessità.
- **Utilizzo:**

Client-side discovery

- **Scopo:**all'interno di un'architettura a microservizi, questi ultimi si trovano spesso in posizioni non fissate in quanto decise dinamicamente. Un metodo per la loro localizzazione consiste nel pattern Client-side discovery, che consiste nella richiesta della posizione di uno specifico microservizio da parte del client ad un registro, che conosce le posizioni di tutte le istanze dei servizi.
- **Vantaggi:**
  - permette di allocare dinamicamente diverse istanze di diversi servizi.
- **Svantaggi:**
  - crea dipendenze tra il registro e il client.
- **Utilizzo:**

Data Access Object

- **Scopo:**
- **Vantaggi:**
  - –
- **Svantaggi:**
  - – .
- **Utilizzo:**

Dependency Injection?

- **Scopo:**
- **Vantaggi:**
  - –
- **Svantaggi:**
  - – .
- **Utilizzo:**

Strutturali

Facade

- **Scopo:**
- **Vantaggi:**
  - –
- **Svantaggi:**
  - – .
- **Utilizzo:**

Adapter

- **Scopo:**
- **Vantaggi:**
  - –
- **Svantaggi:**
  - – .
- **Utilizzo:**

Creazionali

Singleton

- **Scopo:**
- **Vantaggi:**
  - –
- **Svantaggi:**
  - .
- **Utilizzo:**

Comportamentali

Observer

- **Scopo:**
- **Vantaggi:**
  - –
- **Svantaggi:**
  - .
- **Utilizzo:**