Microgestionali in Python

di

Antonio Valente

Cos'è un Microgestionale

Cos'è un Microgestionale

- Applicazione a carattere gestionale

Cos'è un Microgestionale

- Applicazione a carattere gestionale
- Deve risolvere **un problema** nel modo più semplice possibile, e integrandosi nel metodo di lavoro in uso nell'azienda

Cos'è un Microgestionale

- Applicazione a carattere gestionale
- Deve risolvere **un problema** nel modo più semplice possibile, e integrandosi nel metodo di lavoro in uso nell'azienda
- Economico, di semplice utilizzo e rapido apprendimento

Cos'è un Microgestionale

- Applicazione a carattere gestionale
- Deve risolvere **un problema** nel modo più semplice possibile, e integrandosi nel metodo di lavoro in uso nell'azienda
- Economico, di semplice utilizzo e rapido apprendimento
- Sviluppato in tempi brevi, spesso in modo incrementale secondo metodi di sviluppo agile

Cos'è un Microgestionale

- Applicazione a carattere gestionale
- Deve risolvere **un problema** nel modo più semplice possibile, e integrandosi nel metodo di lavoro in uso nell'azienda
- Economico, di semplice utilizzo e rapido apprendimento
- Sviluppato in tempi brevi, spesso in modo incrementale secondo metodi di sviluppo agile
- Ben supportato, e aperto a modifiche ed evoluzioni

Cos'è un Microgestionale

- Applicazione a carattere gestionale
- Deve risolvere **un problema** nel modo più semplice possibile, e integrandosi nel metodo di lavoro in uso nell'azienda
- Economico, di semplice utilizzo e rapido apprendimento
- Sviluppato in tempi brevi, spesso in modo incrementale secondo metodi di sviluppo agile
- Ben supportato, e aperto a modifiche ed evoluzioni
- Aperto a integrazioni con altri software

A chi è utile:
Le Microimprese

- Meno di dieci dipendenti
- Fatturato annuo inferiore a 2 milioni di euro
- Spesso a gestione familiare
- Risorse limitate, sia in termini economici che in termini di personale
- Danno impiego ad oltre il 50% dei lavoratori italiani
- Predominio netto in Italia

Perché i Microgestionali

Necessità di informatizzare o automatizzare un processo aziendale (produttivo, organizzativo, ecc).
Alternative:

Perché i Microgestionali

Necessità di informatizzare o automatizzare un processo aziendale (produttivo, organizzativo, ecc).
Alternative:

- Acquisire un nuovo software gestionale che gestisca anche l'aspetto desiderato

Perché i Microgestionali

Necessità di informatizzare o automatizzare un processo aziendale (produttivo, organizzativo, ecc).
Alternative:

- Acquisire un nuovo software gestionale che gestisca anche l'aspetto desiderato

- Cercare una funzione simile nel gestionale attuale ed adattarvisi

Perché i Microgestionali

Necessità di informatizzare o automatizzare un processo aziendale (produttivo, organizzativo, ecc).
Alternative:

- Acquisire un nuovo software gestionale che gestisca anche l'aspetto desiderato

- Cercare una funzione simile nel gestionale attuale ed adattarvisi

- Richiedere agli sviluppatori del gestionale attuale l'implementazione della nuova funzionalità

Perché i Microgestionali

Necessità di informatizzare o automatizzare un processo aziendale (produttivo, organizzativo, ecc).
Alternative:

- Acquisire un nuovo software gestionale che gestisca anche l'aspetto desiderato

- Cercare una funzione simile nel gestionale attuale ed adattarvisi

- Richiedere agli sviluppatori del gestionale attuale l'implementazione della nuova funzionalità

- Implementare un software microgestionale ed eventualmente integrarlo nell'infrastruttura attuale

Strumenti utilizzati

- MS Access
- MS Visual Basic
- MS Visual FoxPro o altri strumenti derivati da dBase
- Borland Delphi
- Altri strumenti vari
- Python

Perchè Python?

Perchè Python?

- Gratuito, disponibile su tantissime piattaforme

Perchè Python?

- Gratuito, disponibile su tantissime piattaforme
- Semplice, potente, leggibile e facile da mantenere

Perchè Python?

- Gratuito, disponibile su tantissime piattaforme
- Semplice, potente, leggibile e facile da mantenere
- Massimizza la produttività dello sviluppatore

Perchè Python?

- Gratuito, disponibile su tantissime piattaforme
- Semplice, potente, leggibile e facile da mantenere
- Massimizza la produttività dello sviluppatore
- Dispone di librerie e framework per praticamente qualunque esigenza, molto spesso open source

Perchè Python?

- Gratuito, disponibile su tantissime piattaforme
- Semplice, potente, leggibile e facile da mantenere
- Massimizza la produttività dello sviluppatore
- Dispone di librerie e framework per praticamente qualunque esigenza, molto spesso open source
- Permette una prototipizzazione rapida, favorendo lo sviluppo passo per passo guidato dal committente

Perchè Python?

- Gratuito, disponibile su tantissime piattaforme
- Semplice, potente, leggibile e facile da mantenere
- Massimizza la produttività dello sviluppatore
- Dispone di librerie e framework per praticamente qualunque esigenza, molto spesso open source
- Permette una prototipizzazione rapida, favorendo lo sviluppo passo per passo guidato dal committente
- Semplifica l'integrazione con altri software e ambienti

Scegliere l'architettura

Scegliere l'architettura

- *La scelta dell'architettura di un software va valutata caso per caso*

Scegliere l'architettura

- *La scelta dell'architettura di un software va valutata caso per caso*
- Thin client: più adatta in ambienti di rete eterogenei e con molti utenti; l'interfaccia utente risulta limitata e poco flessibile, l'amministrazione più complessa

Scegliere l'architettura

- *La scelta dell'architettura di un software va valutata caso per caso*
- Thin client: più adatta in ambienti di rete eterogenei e con molti utenti; l'interfaccia utente risulta limitata e poco flessibile, l'amministrazione più complessa
- Fat client: adatta ai microgestionali, se ci si può legare a un solo DBMS

Scegliere l'architettura

- *La scelta dell'architettura di un software va valutata caso per caso*
- Thin client: più adatta in ambienti di rete eterogenei e con molti utenti; l'interfaccia utente risulta limitata e poco flessibile, l'amministrazione più complessa
- Fat client: adatta ai microgestionali, se ci si può legare a un solo DBMS
- Full client: adatta per semplificare al massimo il deploy e l'amministrazione del software. Pochissimo scalabile, scelta pessima per ambienti multiutente

Gli strumenti: Categorie

Gli strumenti: Categorie

- Strumenti di sviluppo integrati

Gli strumenti: Categorie

- Strumenti di sviluppo integrati
- Frameworks e librerie indipendenti

Gli strumenti integrati: Dabo

- www.dabodev.com
- Guida l'utente tramite GUI in tutte le fasi di sviluppo
- Permette il disegno del database, degli oggetti applicativi, dell'interfaccia utente
- Permette la definizione in modo visuale di reports

Gli strumenti integrati: GNU Enterprise

- www.gnuenterprise.org
- Parte del progetto GNU
- Metaprogetto composto da vari sottoprogetti con scopi e caratteristiche vagamente simili a Dabo
- I vari componenti possono essere utilizzati indipendentemente
- Ancora molto acerbo

Gli strumenti integrati: vantaggi e svantaggi

- Strumenti complessi, con tempi di apprendimento mediamente lunghi
- Una volta imparati, semplificano di molto lo sviluppo
- Scarsa flessibilità per casi d'uso diversi da quelli previsti dagli sviluppatori
- Possibilità di integrazione con altri software limitata

Gli strumenti indipendenti: Twisted

- www.twistedmatrix.com
- Fornisce strumenti per creare client e server per svariati protocolli
- Evita i problemi della programmazione multithreaded tramite il suo approccio asincrono
- Indispensabile per creare proxy e interfacce di rete verso altri software

Gli strumenti indipendenti: SQLAlchemy

- www.sqlalchemy.org
- Libreria per l'astrazione dal database che contiene anche un modulo ORM
- Permette di accedere a svariati DBMS tramite oggetti python astraendone le differenze
- Tramite il suo ORM permette di realizzare in modo relativamente semplice la persistenza di oggetti arbitrari su tabelle di database

Gli strumenti indipendenti: wxPython

- www.wxpython.org
- Toolkit grafico per la creazione di GUI multipiattaforma con look&feel nativo
- Equivalente agli altri due famosi toolkit: PyQT e PyGTK, tranne che in casi specifici
- Grosso bacino di utenza
- Documentazione scadente

Gli strumenti indipendenti: Reportlab

- www.reportlab.org

- Libreria per creare files PDF dall'aspetto professionale
- Permette di creare in modo relativamente semplice files PDF contenenti testo, grafici, immagini, ecc.
- Utilizzabile per creare reports nelle proprie applicazioni

Best practices

- ***Analizzare approfonditamente le esigenze del committente***
- Non fidarsi delle specifiche dichiarate, andare sempre a fondo e proporre le proprie idee
- Stabilire il tipo di architettura più adatto allo specifico progetto
- Predisporre suite di test estensive, il refactoring è essenziale

Best practices: Il Database

- Scegliere con attenzione il DBMS da utilizzare, e tentare di non legarsi troppo a uno specifico strumento
- Progettare attentamente il database. Strutturarlo in modo da rendere semplici future espansioni

Best practices: Gli oggetti business

- Isolare lo strato di accesso ai dati dallo strato di business, ad esempio tramite il pattern Data Mapper

- Non legare strettamente la definizione del database a quella della gerarchia di classi. Le due devono poter variare indipendentemente se nasce questa esigenza

- Progettare il mapping tra classi e database, implementandolo tramite strumenti preesistenti o propri. Tentare di ridurre le ripetizioni, senza violare l'indipendenza

Best practices: L'interfaccia utente

- Mai inserire codice applicativo nell'interfaccia utente
- Tenere la UI semplice e lineare. Evitare schermate piene di elementi o con troppo testo. Raggruppare gli elementi per pertinenza logica
- Rendere la UI coerente. Fare in modo che il look&feel sia uniforme, e che le risposte dell'interfaccia agli eventi siano quelle che l'utente si attende.
- Guidare l'utente tramite finestre di scelta rapida o altri meccanismi. Non affidarsi alla memoria dell'utente. Fornire valori di default sensati se possibile

Conclusioni

- Utilizzare python e gli strumenti che esso rende possibili per ridurre le parti noiose dello sviluppo di un microgestionale
- Cercare e testare gli strumenti più adatti al proprio progetto e ai propri gusti
- Python consente di creare soluzioni avanzate in tempi ridotti.

Grazie dell'attenzione