UNIVERSITÀ DEGLI STUDI DI PARMA

Dipartimento di Matematica e Informatica

Corso di Laurea in Informatica

Tesi di Laurea

Applicazioni iOS: analisi dei linguaggi di programmazione utilizzati e sviluppi futuri della piattaforma

Candidato:

Federico Bertoli

Relatore:

Prof. Roberto Alfieri

Anno Accademico 2015/2016

Indice

Introduzione **4**

1 Objective-C **5**
1.1 Cenni storici 5
1.2 Caratteristiche del linguaggio 7
1.2.1 Sintassi 7
1.2.2 Gestione della memoria 7
1.2.3 Compilatore 7
1.2.4 Utilizzo in iOS e frameworks Cocoa 7

2 Swift **8**
2.1 Cenni storici 8
2.2 Caratteristiche 8
2.2.1 Sintassi 8
2.2.2 Gestione della memoria 8
2.2.3 Compilatore 8
2.2.4 Utilizzo in iOS e frameworks Cocoa 8

3 Confronto tra i linguaggi (trovare un titolo adeguato) **9**
3.1 Punti di forza e debolezze dell'Objective-C 9
3.2 Punti di forza e debolezze di Swift 9
3.3 Confronto delle performance e gestione della memoria 9

4 Il futuro della piattaforma iOS **10**
4.1 Quale futuro per i due linguaggi? 10

Conclusioni **11**

Ringraziamenti **12**

Riferimenti bibliografici **13**

INDICE 2

Appendice **14**

TODO

Introduzione

Lo sviluppo di applicazioni in ambito mobile ha subito una crescita esponenziale da quando, nel 2008, Apple ha permesso di accedere agli strumenti di sviluppo interni (nello specifico il software Xcode e i relativi SDK) e ha inserito nella versione 2.0 del software iOS l'applicazione App Store, permettendo di fatto a chiunque di rendere la propria idea una realtà, supportata da un'infrastruttura di servizi ben collaudata, concentrando i propri sforzi solamente sullo sviluppo del software.

Nello stesso anno anche Google, tramite l'Android Market ha permesso la pubblicazione di applicazioni sul proprio store, creando così una "corsa all'oro" che solamente ora, 8 anni dopo, sta subendo un sensibile calo dopo anni di crescita significativa e costante.

Precedentemente a queste due piattaforme lo sviluppo in ambito mobile era frenato da fattori importanti quali la mancanza di software di sviluppo stabili e continuamente aggiornati, la mancanza di sistemi operativi sufficientemente avanzati e la scarsa potenza dei dispositivi dell'epoca.

Ciò che ha portato a questa tesi è stato un interesse molto forte verso questo mondo con nemmeno una decade sulle spalle (se si parla di smartphone) e ancora pieno di sviluppi, la maggior parte di questi solo annunciati e ancora in piena fase di progettazione (si pensi per esempio alla realtà aumentata, alle features appena annunciate sui nuovi dispositivi come le doppie fotocamere, i microfoni HAAC o i sensori per il touchscreen in 3 dimensioni).

In questa tesi abbiamo focalizzato l'attenzione sull'ecosistema creato da Apple per i propri dispositivi iOS e in particolare sui linguaggi utilizzati per lo sviluppo, Objective-C e Swift.

Capitolo 1

Objective-C

1.1 Cenni storici

Nei primi anni ottanta, la pratica comune dell'ingegneria del software era basata sulla programmazione strutturata. Questa modalità era stata sviluppata per poter suddividere programmi di grandi dimensioni in parti più piccole, principalmente per facilitare il lavoro di sviluppo e di manutenzione del software. Ciononostante, col crescere della dimensione dei problemi da risolvere, la programmazione strutturata divenne sempre meno utile, dato che conduceva alla stesura di un numero sempre maggiore di procedure, ad uno spaghetti code e ad uno scarso riuso del codice sorgente.

Venne ipotizzato poi che la programmazione orientata agli oggetti potesse essere una potenziale soluzione al problema. In effetti Smalltalk aveva già affrontato molte di queste questioni ingegneristiche, pur con lo svantaggio di necessitare di una macchina virtuale che interpretava un oggetto in memoria chiamato immagine contenente tutti gli strumenti necessari. L'immagine Smalltalk era molto grossa, usava tendenzialmente un'enorme quantità di memoria per l'epoca e girava molto lentamente anche per la mancanza di un supporto specifico dell'hardware alle macchine virtuali.

Objective C fu creato principalmente da Brad Cox e Tom Love all'inizio degli anni ottanta alla Stepstone. Entrambi erano stati introdotti a Smalltalk durante la loro permanenza al Programming Technology Center della ITT Corporation nel 1981. Cox aveva iniziato ad interessarsi ai problemi legati alla riusabilità del software e si accorse che un linguaggio simile a Smalltalk sarebbe stato estremamente valido per costruire potenti ambiente di sviluppo per i progettisti di ITT.

Cox iniziò così a modificare il compilatore C per aggiungere alcune delle caratteristiche di Smalltalk. Ottenne così ben presto una implementazione funzionante di una estensione ad oggetti del linguaggio C che chiamò OOPC (Object-Oriented Programming in C). Nel frattempo Love fu assunto da Schlumberger Research nel 1982 ed ebbe l'opportunità di acquisire la prima copia commerciale di Smalltalk-80 che influenzò in seguito lo sviluppo della loro creatura.

Per dimostrare che il linguaggio costituiva un reale progresso, Cox mostrò che per realizzare componenti software intercambiabili erano necessari pochi adattamenti pratici agli strumenti già esistenti. Nello specifico, era necessario supportare gli oggetti in modo flessibile con un insieme di librerie software che fossero usabili e consentissero al codice sorgente (e ad ogni risorsa necessaria al codice) di essere raccolto in un solo formato multipiattaforma.
Cox e Love formarono infine una nuova impresa, la Productivity Products International (PPI), per commercializzare il loro prodotto che accoppiava un compilatore Objective C con una potente classe di librerie.

Nel 1986 Cox pubblicò la sua descrizione dell'Objective C nella sua forma originale nel libro Object-Oriented Programming, An Evolutionary Approach. Sebbene fosse attento a puntualizzare che la questione della riusabilità del software non poteva essere esaurita dal linguaggio di programmazione, l'Objective C si trovò spesso ad essere confrontato, caratteristica per caratteristica, con gli altri linguaggi.

Nel 1988, NeXT, la compagnia fondata da Steve Jobs dopo Apple, ottenne la licenza dell'Objective C da Stepstone (allora proprietaria del marchio) e realizzò il proprio compilatore Objective C e le librerie sulle quali basò l'interfaccia utente di NeXTSTEP. Sebbene le workstation NeXTSTEP non riuscissero ad avere un forte impatto sul mercato, i loro strumenti vennero ampiamente apprezzati dall'industria del settore. Ciò portò NeXT ad abbandonare la produzione di hardware ed a focalizzarsi sugli strumenti software, vendendo NeXTSTEP (e OpenStep) come piattaforma per la programmazione.

In seguito il progetto GNU iniziò a lavorare sul clone libero che chiamò GNUstep, basato sullo standard OpenStep. Dennis Glatting scrisse il primo run-time gnu-objc nel 1992 e Richard Stallman lo seguì subito dopo con un secondo. Il run-time GNU Objective C, che è usato dal 1993, è stato sviluppato da Kresten Krab Thorup mentre era studente universitario in Danimarca.

1.2 Caratteristiche del linguaggio

1.2.1 Sintassi

TODO

1.2.2 Gestione della memoria

TODO

1.2.3 Compilatore

TODO

1.2.4 Utilizzo in iOS e frameworks Cocoa

TODO

Capitolo 2

Swift

2.1 Cenni storici

TODO

2.2 Caratteristiche

2.2.1 Sintassi

TODO

2.2.2 Gestione della memoria

TODO

2.2.3 Compilatore

TODO

2.2.4 Utilizzo in iOS e frameworks Cocoa

TODO

Capitolo 3

Confronto tra i linguaggi (trovare un titolo adeguato)

3.1 Punti di forza e debolezze dell'Objective-C

TODO

3.2 Punti di forza e debolezze di Swift

TODO

3.3 Confronto delle performance e gestione della memoria

TODO

Capitolo 4

Il futuro della piattaforma iOS

4.1 Quale futuro per i due linguaggi?

TODO

Conclusioni

TODO

Ringraziamenti

TODO

Bibliografia

[1] Objective-C
Publications, Books, Articles, Interviews, etc.
`http://virtualschool.edu/objectivec/`

Appendice

Il codice del progetto è disponibile su GitHub all'indirizzo
`https://github.com/FrancescoSalvatore/thesis`