ALMA MATER STUDIORUM – UNIVERSITÀ DI BOLOGNA

FACOLTÀ DI SCIENZE MATEMATICHE FISICHE E NATURALI
CORSO DI LAUREA MAGISTRALE IN INFORMATICA

TESI DI LAUREA
in
INTELLIGENZA ARTIFICIALE

UNA REALIZZAZIONE DISTRIBUITA DI TEST NEUROPSICOLOGICI

CANDIDATO
Federico Tarantino

RELATORE
Chiar.mo Prof. Maurizio Gabbrielli

ANNO ACCADEMICO 2010/2011

SESSIONE PRIMA

Alle persone per me più importanti . . .

Introduzione

Uno dei grandi vantaggi dell'informatica è la sua diffusione capillare all'interno di qualunque area di studio o ambiente di lavoro. Questo aspetto, per essere fronteggiato, porta alla formazione di specialisti che devono fare della duttilità una delle loro maggiori capacità.
Tra le aree che maggiormente hanno usufruito dell'esponenziale crescita del mondo informatizzato c'è sicuramente la medicina, che ha l'onere di dover essere sempre al passo con i tempi, in modo da garantire sempre il meglio per il genere umano.
Questa tesi, pur nel suo piccolo, vuole seguire questa linea di continuo scambio tra informatica e medicina, sfruttando tecnologie all'avanguardia per migliorare la fruizione di servizi diretti al cittadino.
Precisamente, il campo medico di pertinenza è la neuropsicologia, la nueroscienza che si caratterizza per il suo obiettivo di studiare i processi cognitivi e comportamentali correlandoli con i meccanismi anatomo funzionali che ne sottendono il funzionamento (cfr.[1]).
In presenza di disturbi delle funzioni cognitive, tra i molteplici esami sottoposti al paziente, sono utilizzati alcuni test il cui scopo è mettere in luce le deficienze possedute dal soggetto preso in esame. Ovviamente, esistendo molti tipi di mancanze, ogni test è stato creato con l'obiettivo di focalizzarsi su un preciso disturbo. Sarà comunque sempre l'esaminatore che avrà il compito di decidere quali test somministrare, sulla base delle caratteristiche del paziente.
L'esecuzione di questi test è effettuata tramite un colloquio con il dottore, e per alcune prove, è previsto in aggiunta l'uso di foglio e penna. In quest'ultimo caso il paziente dovrà scrivere, disegnare o crociare.

Questa tesi fornisce una realizzazione di alcuni test, scelti in collaborazione di un neuropsicologo, che cercano di sostituire degnamente foglio e penna con touch screen e dita.
L'obiettivo è quello di ricreare il test senza aggiungere agenti esterni derivanti dalla nuova interfaccia, per far sì che la fruizione del test sia buona almeno quanto quella del corrispettivo non digitale.
Per raggiungere un risultato del genere deve essere studiato ogni minimo particolare, per far sì che il paziente non incorra in alcuna difficoltà durante lo svolgimento del test.
Per cominciare, la scelta delle tecnologie a supporto è stata adeguata. Per quanto riguarda il dispositivo touch screen, è fuori discussione che al momento uno dei migliori prodotti è l'Apple iPad 2, in grado di fornire uno schermo da 10 pollici e una sensibilità al tocco fuori dal comune. Ottimo anche il supporto alla programmazione per questa tipologia di dispositivi.
Siccome i test sono stati pensati sin dall'inizio come software distribuito, per gestire il centro nevralgico del sistema è stato scelto Jolie, un paradigma di programmazione orientato ai servizi (SOC).
La tesi è articolata in diverse sezioni: all'inizio saranno esaminati i particolari dei test neuropsicologici scelti da realizzare, mentre in un secondo momento saranno mostrati i dettagli di progettazione e succesivamente di implementazione.

Indice

Capitolo 1

I test neuropsicologici

I disturbi delle funzioni cognitive richiedono un lungo ed attento esame, che comprende sia una valutazione di base relativamente comune a tutti i pazienti, sia modalità diverse da caso a caso. Infatti, l'esame deve essere via via più misurato secondo il profilo che emerge nel corso della valutazione. Per fare ciò è di fondamentale importanza l'esperienza dell'esaminatore, che, una volta ottenuto un profilo generale, sceglierà i test indicati a focalizzare i deficit del paziente.
La valutazione neuropsicologica comprende una serie di passaggi, che partono dall'incontro con il paziente fino ad arrivare ad identificare gli aspetti cognitivi da indagare in modo più approfondito.

1.1 Finalità dei test

L'esame neuropsicologico può avere molteplici scopi: in primo luogo esso mira a fornire un quadro completo di un paziente, dando informazioni sulle sue abilità cognitive, e in alcuni casi può addirittura essere uno strumento diagnostico indispensabile. Una seconda finalità, collegata alla precedente, è di tipo prognostico, cioè fornisce indicazioni sull'esito di alcune patologie. Altre finalità, importanti come le precedenti, sono la pianificazione dell'assistenza e degli interventi e l'indirizzamento verso un progetto riabilitativo per il paziente, mirato a ripristinare le funzioni riconosciute deficitarie.

1.2 Vantaggi nell'implementazione dei test

Come si può facilmente immaginare, l'interazione tra esaminatore e paziente è continua durante la somministrazione di un test. Perciò la presenza del dottore è indispensabile. Se poi si tiene conto della lunga durata di un colloquio causata dall'esecuzione di più test, giungiamo facilmente alla conclusione che una delle criticità di queste prove è proprio il tempo.
Oltre alla durata intrinseca del test, si deve aggiungere un tempo complessivo di accodamento dei pazienti che aspettano, che porta a liste di attesa spesso piene anche per mesi.
Pensando che questo tipo di disturbi possono essere aggravati dalla mancata cura in tempi brevi, è subito evidente che qualunque soluzione che mantenga la qualità dell'operato ma che faccia risparmiare tempo è ben voluta.
Infatti il vantaggio più grande derivante dall'implementazione di test neuropsicologici è proprio questo: si immagini un ambiente in cui più persone contemporaneamente, ognuno con il proprio dispositivo tra le mani, svolga un determinato test, con un unico supervisore a controllo di tutti i pazienti. Già con questo semplice compromesso, il risparmio sarebbe elevatissimo, specie se si considera il fatto che ogni paziente non deve per forza svolgere lo stesso test.
Di situazioni che porterebbero miglioramenti se ce ne sono comunque tante altre, come ad esempio la possibilità di svolgere test direttamente da casa con la supervisione di un familiare (ed invio dei risultati ad un server centrale), oppure far svolgere dei test premilinari mentre il paziente è in sala d'attesa ancora prima di averci un colloquio.
Nonostante già questo basti ad accertare l'utilità di una versione digitale dei test, di vantaggi ce ne sono sicuramente degli altri, come una gestione centralizzata dei risultati.
Si immagini un server unico al quale tutti i dispositivi inviano i propri risultati. Creando un'interfaccia ad-hoc, destinata al neuropsicologo, rendiamo facile ed immediata la fruizione dei risultati dei test. Seguire l'evoluzione dello stato di un paziente o estrarre risultati di vecchi test diventano operazioni istantanee, non sottovalutando il minore impatto ambientale dovuto

al risparmio di centinaia di fogli di carta che dovrebbero essere conservati per ogni paziente.
Un'ultima ma altrettanto importante valutazione deriva dall'uso delle dita al posto della penna, che potrebbe rendere ancora più naturale l'esecuzione dei test. Questo fattore diventa rilevante in quanto minimizza gli agenti esterni che potrebbero influenzare l'esito dei test.
Al contrario, un aspetto negativo della fruizione dei test tramite touch screen potrebbe essere la scetticità dei pazienti nei confronti di un dispositivo che non conoscono e la conseguente difficoltà ad adeguarsi a tali metodi, specie se consideriamo che il target di pazienti con disturbi cognitivi si attesta su una fascia di età medio-alta.

1.3 I test scelti per l'implementazione

Dopo aver giustificato la reale utilità della realizzazione dei test in versione digitale, è il momento di vedere i test nel dettaglio.
Di queste prove ne esistono tantissime varianti, ognuna focalizzata su qualche aspetto cognitivo. In più spesso per ogni test esistono anche diverse versioni, a seconda dell'ideatore che lo ha proposto.
Sotto l'ausilio di un neuropsicologo, sono stati scelti 4 test da realizzare, precisamente:

- Clock Test
- Stime Cognitive Test
- Token Test
- Trail Making Test

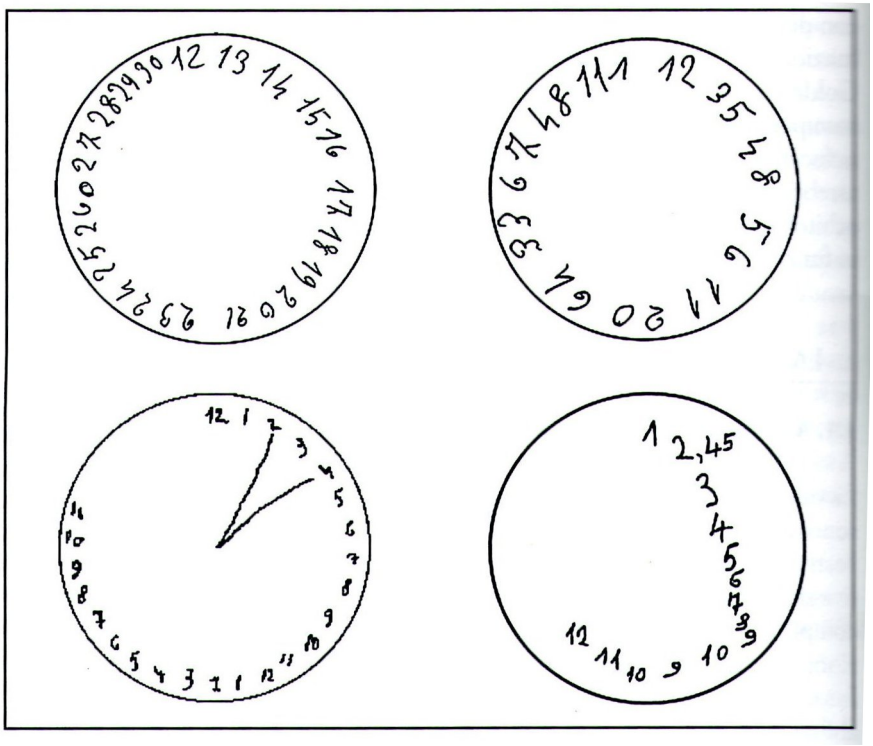
I criteri della scelta sono stati soprattutto la diffusione di questi test, molto usati in ambito neuropsicologico, e l'elevato adattamento ad un dispositivo touch screen, in quanto essi prevedono l'utilizzo di pulsanti o il disegno di linee, operazioni naturali per questa tipologia di dispositivi.
Prima di passare alla fase di progettazione e realizzazione dei test, è opportuno studiare i dettagli di ognuno di essi.

1.3.1 Clock Test

Il Clock Test (o test dell'orologio) è uno strumento di rapida somministrazione e di largo impiego clinico; ne sono state elaborate svariate versioni. In una versione tarata in Italia, si chiede al paziente di scrivere i numeri di un orologio all'interno di un cerchio; al termine, si chiede di disegnare le lancette che segnano le due e quarantacinque. I pazienti frontali possono commettere vari tipi di errori, soprattutto determinati da un decifit nel controllo attenzionale o da una difficoltà nel pianificare il disegno, anche quando in altri test che misurano un disturbo simile ottengono un punteggio normale.
Il punteggio assegnato sarà così calcolato:

- 0, 1, 2 o 4 punti per la corretta scrittura dei numeri
- 0, 2 o 3 punti per la corretta disposizione dei numeri
- 0, 1 o 2.5 punti per la corretta scrittura delle lancette
- +0,5 punti se le lancette sono corrette e di lunghezza diversa

per un massimo di 10 punti.
L'esaminatore potrebbe essere interessato anche ad altri aspetti che non interessano il punteggio, come l'ordine in cui vengono scritti i numeri, la grandezza dei numeri o eventuali episodi di disgrafia grave.



1.3.2 Stime Cognitive Test

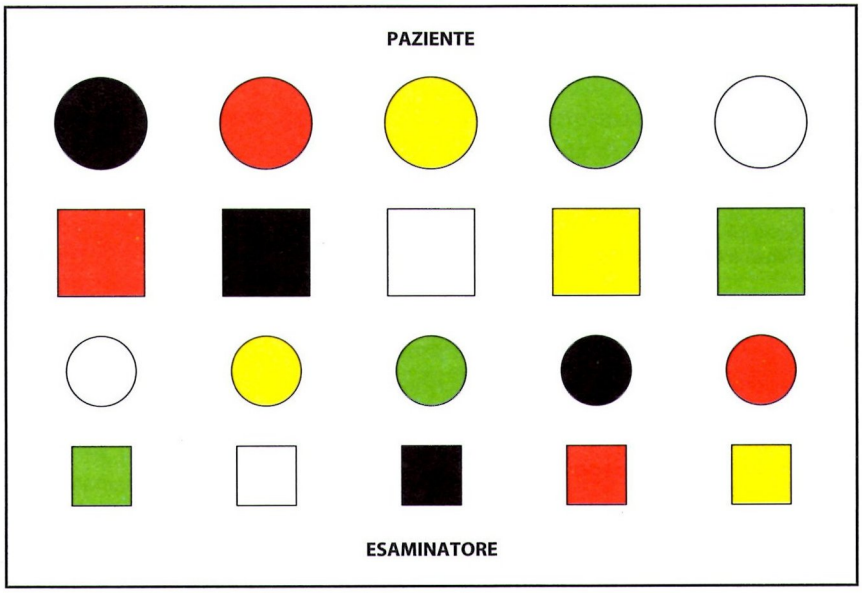
Stime Cognitive è un test che valuta il ragionamento astratto e le capacità di elaborare risposte plausibili. Le domande dei test richiedono un'elaborazione delle conoscenze generali comuni (ad esempio, ″quanto è alto un semaforo?″), per cui un deficit di memoria semantica può determinare una prestazione patologica. Un test analogo comprende quesiti su tempi e pesi.
Nella versione implementata, le domande sono 5:

- Quanto costa un litro di latte fresco?
- Quanto dista Milano da Roma?
- Quanto è lunga una chitarra?
- Quanto dura una messa?
- In Olanda, quanti canguri ci sono?

Per ogni domanda non viene volontariamente specificata l'unità di misura della risposta, in quanto potrebbe influenzare la risposta stessa. Ovviamente i valori accettati sono all'interno di alcuni range prefissati. Unica eccezione per l'ultima domanda, in cui la risposta è 0.
Il punteggio è, per ogni domanda, 1 in caso di risposta esatta, 0 altrimenti, per un totale massimo di 5.

1.3.3 Token Test

Il Token Test è un test usato per valutare la comprensione. La prova consiste di una serie di ordini di complessità crescente che richiedono operazioni da compiere su gettoni di varia forma (quadrati e cerchi, piccoli e grandi) e colore (neri, bianchi, gialli, rossi e verdi). Il compito misura un insieme di abilità connesse con la comprensione lessicale e sintattica, tra cui la memoria a breve termine fonologica. Il token test permette di discriminare tra pazienti afasici e non afasici ed è relativamente sensibile per la diagnosi di deficit di linguaggio di grado lieve e in soggetti con bassa scolarità. La prestazione al test non è influenzata da eventuali deficit intellettivi associati.



Nella versione implementata, sono previsti 36 ordini, divisi in 6 parti diverse. Per tutti gli ordini, tranne per quelli della sesta parte, è previsto che se il paziente non risponde entro 5 secondi oppure risponde in maniera er-

rata/incompleta, si rimettono in ordine i gettoni e viene data una seconda possibilità. Per gli ordini della sesta parte non si ripete mai la domanda.

- Parte I: con tutti i gettoni
  1. Tocchi un cerchio
  2. Tocchi un quadrato
  3. Tocchi un gettone giallo
  4. Ne tocchi uno rosso
  5. Ne tocchi uno nero
  6. Ne tocchi uno verde
  7. Ne tocchi uno bianco
- Parte II: con i soli gettoni grandi
  8. Tocchi il quadrato giallo
  9. Tocchi il cerchio nero
  10. Tocchi il cerchio verde
  11. Tocchi il quadrato bianco
- Parte III: con tutti i gettoni
  12. Tocchi il cerchio bianco piccolo
  13. Tocchi il quadrato giallo grande
  14. Tocchi il quadrato verde grande
  15. Tocchi il cerchio nero piccolo
- Parte IV: con i soli gettoni grandi
  16. Tocchi il cerchio rosso ed il quadrato verde
  17. Tocchi il quadrato giallo ed il quadrato nero
  18. Tocchi il quadrato bianco ed il cerchio verde

19. Tocchi il cerchio bianco ed il cerchio rosso

- Parte V: con tutti i gettoni

20. Tocchi il cerchio bianco grande ed il quadrato verde piccolo
21. Tocchi il cerchio nero piccolo ed il quadrato giallo grande
22. Tocchi il quadrato verde grande ed il quadrato rosso grande
23. Tocchi il quadrato bianco grande ed il cerchio verde piccolo

- Parte VI: con i soli gettoni grandi

24. Metta il cerchio rosso sopra il quadrato verde
25. Tocchi il cerchio nero con il quadrato rosso
26. Tocchi il cerchio nero ed il quadrato rosso
27. Tocchi il cerchio nero oppure il quadrato rosso
28. Metta il quadrato verde lontano dal quadrato giallo
29. Se c' è un cerchio blu, tocchi il quadrato rosso
30. Metta il quadrato verde vicino al cerchio rosso
31. Tocchi lentamente i quadrati e rapidamente i cerchi
32. Metta il cerchio rosso fra il quadrato giallo e quello verde
33. Tocchi tutti i cerchi, eccetto quello verde
34. Tocchi il cerchio rosso, anzi il quadrato bianco
35. Invece del quadrato bianco, tocchi il cerchio giallo
36. Insieme al cerchio giallo, tocchi il cerchio nero

Il punteggio è così assegnato: 1 punto per risposta corretta, 0.5 punti per risposta corretta dopo la ripetizione (non esiste per la parte VI), 0 punti per risposta scorretta. Per un totale massimo di 36 punti.

1.3.4 Trail Making Test

I pazienti con trauma cranico hanno spesso decifit di allerta e di attenzione sostenuta. A volte presentano deficit di attenzione selettiva, ma soprattutto sono compromessi nei compiti che esaminano l'attenzione divisa.
Questa componente dell'attenzione è difficile da valutare, poichè presuppone l'integrità delle altre componenti, in particolare dell'attenzione selettiva: se il paziente ha difficoltà a focalizzare l'attenzione su un compito, sarà maggiore la difficoltà a svolgerne due o più contemporaneamente.
Il Trail Making Test valuta proprio questo aspetto.
Il test è costituito da due parti (A e B). Nella parte A, il soggetto deve unire con una linea in ordine crescente 25 numeri cerchiati e stampati su un foglio (1,2,3,ecc.). Nella parte B il soggetto deve collegare alternativamente 13 numeri e 12 lettere in ordine progressivo, ugualmente cerchiati e stampati su un secondo foglio (1,A,2,B,ecc.). Entrambe le parti devono essere eseguite il più velocemente possibile. In questo caso il doppio compito è dovuto al fatto che il soggetto deve tenere a mente le sequenze di numeri e lettere e, nello stesso tempo, unire un bersaglio con il successivo.
Nella versione implementata, entrambi le parti A e B sono preceduti da una versione delle prove a 8 cerchi (che non sarà presa in considerazione per il punteggio finale), valida per istruire il soggetto. Infatti non è possibile nè alzare la penna dal foglio, nè toccare numeri/lettere diversi dal successivo.
Se il paziente non riesci ad eseguire la versione preventiva, il test non sarà nemmeno somministrato.
Per ogni parte del test sarà assegnato un punteggio separatamente, tramite una coppia di valori che rappresentano il tempo in secondi che il paziente ha impiegato per terminare la prova e il numero di errori commessi.

Capitolo 2

Architettura dell'applicazione

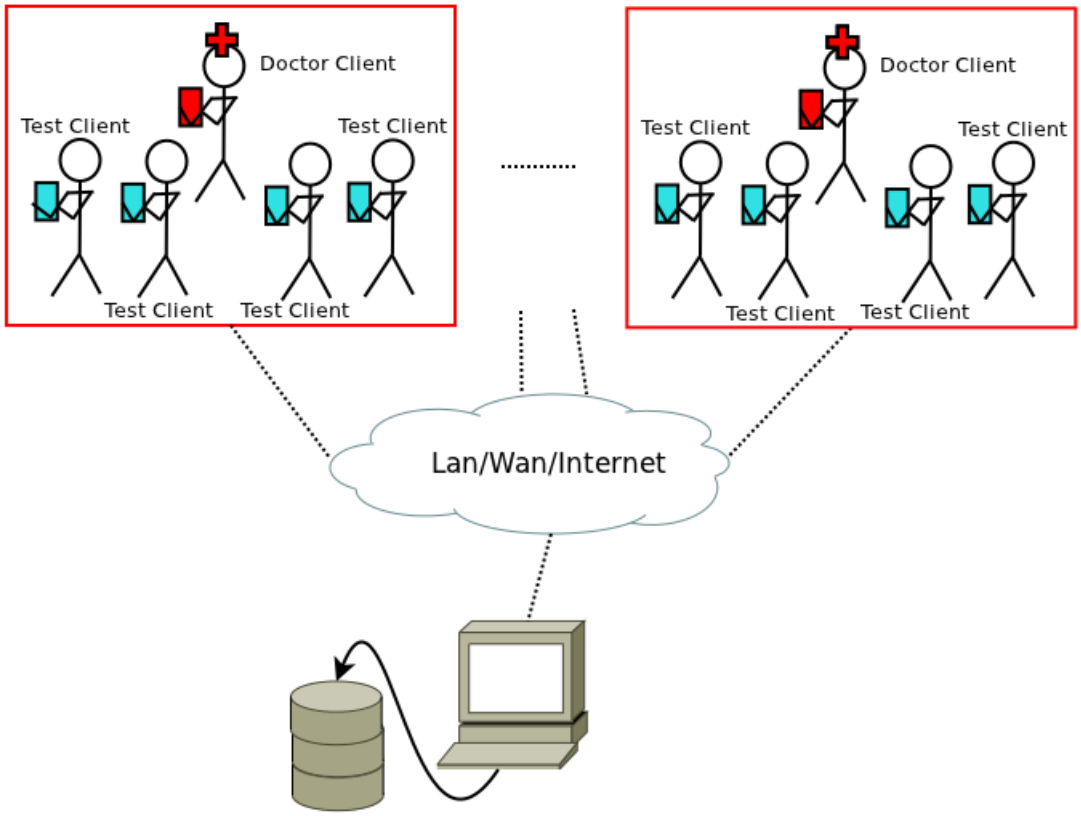
La realizzazione dei test neuropsicologici presentati nel capitolo precedente non può concretizzarsi nella mera trasposizione di ognuno di essi senza una logica che unisca il tutto.
Il compito di un informatico è quella di creare un sistema completo e complesso che metta in condizione gli utenti che usufruiranno del software (sia medici che pazienti) di estraniarsi completamente da come tutto il sistema funzioni.
Per raggiungere uno scopo del genere, la progettazione è di fondamentale importanza, perché oltre a fornire alla fine un prodotto valido, deve porre le basi per gli sviluppi futuri e per qualsiasi estensione del software.
Sin dall'inizio l'applicazione è stata pensata come un software distribuito, in cui alla fine di ogni test i risultati vengono inviati ad un nodo centrale. Le motivazioni sono tante: si è voluto innanzitutto evitare di sovraccaricare di importanza il dispositivo touch screen su cui gireranno i test. Infatti i dati, se salvati solo in locale, potrebbero andare persi o comunque dovrebbero essere continuamente sicronizzati nel caso lo stesso paziente svolga in tempi diversi delle prove su dispositivi diversi.
Avere invece un server centrale dove sono salvati i dati di tutti i test, porta degli enormi vantaggi:

- Maggior sicurezza dei dati: è molto più facile rendere sicuro un unico server (accesso riservato, backup) che tutti i dispositivi, specie se pen-

siamo che all'aumentare dei test implementati, aumenteranno anche il numero di dispositivi utilizzati dai medici

- Aumento delle funzioni aggiuntive: avendo i dati agglomerati in un unico posto, possiamo facilmente tenere traccia dell'evoluzione dello stato di un paziente o svolgere delle indagini statistiche sui dati
- Possibilità di implementare i test su più piattaforme: non essendo vincolati dalla tecnologia che si sceglie per eseguire i test (in quanto l'importanza del test sono i risultati che sono sul server), si possono utilizzare piattaforme diverse per test diversi, cercando di trovare ogni volta la soluzione migliore
- Nessun vincolo sul luogo dei test: essendo i dispositivi collegati in rete con il server centrale, si aggiunge la possibilità di poter eseguire i test da casa (sotto la supervisione di un familiare) o da ospedali diversi

L'architettura dell'applicazione sarà questa:



In questa visione ogni singolo dispositivo touch screen dedicato al paziente

contiene al suo interno una copia dei test e i meccanismi di invio dei risultati ad un server centrale.
Inoltre è previsto un applicativo su una qualsiasi piattaforma dedicato alle funzioni del medico (fruizione dei risultati, controllo dello stato del paziente, ecc.).

Capitolo 3

Tecnologie e strumenti scelti

Nel capitolo precedente sono state inquadrate bene le parti del sistema. Il passo successivo, che verrà visto in questo capitolo, si occuperà di associare ad ognuna di esse una tecnologia, scelta con criterio.

3.1 iPad con iOS (+ Xcode)

Il dispositivo touch screen scelto per ospitare i test neuropsicologici è l'Apple iPad 2, con sistema operativo iOS. Questo dispositivo è tra i più avanzati e completi del mercato, e non ha certo bisogno di troppe presentazioni.
Uno dei suoi punti di forza è Xcode, il potente SDK a supporto degli sviluppatori che aiuta a prendere dimestichezza con l'ambiente di sviluppo, molto chiuso e rigido.
Questo strumento di sviluppo è obbligato per programmare su iPad, e contiene un insieme di tool di sviluppo molto potenti, di cui fanno parte:

- un IDE grafico, con molti tool di supporto che facilitano la programmazione, tra cui il cosiddetto ″Interface Builder″, che permette di creare GUI per le applicazione senza l'uso di codice.

- l'Apple LLVM Compiler, un compilatore integrato che accetta codice scritto in C, Objective-C e C++. Questo strumento contiene anche altre funzioni di correzione automatica.

- strumenti per testare le performarce e il comportamento delle applicazione.

- un simulatore del sistema operativo iOS che permette di provare direttamente sul computer le applicazioni senza trasferirle sul dispositivo fisico.

Tramite Xcode è possibile usare facilmente Cocoa, il framework a disposizione degli sviluppatori contenente centinaia di API a supporto della programmazione.
La scelta è caduta su iPad per cercare di limitare al massimo le differenze che ci sono tra l'uso di penna e foglio rispetto all'uso delle dita su uno schermo. Infatti quest'ultimo rileva click e altre gesture con estrema precisione, aspetto determinante per il corretto esito dei test.
Inoltre lo schermo è molto grande (10 pollici), e permette così allo sviluppatore di ricreare abbastanza fedelmente la controparte cartacea dei test.
L'unico svantaggio derivante dalla scelta di questo dispositivo è il dover seguire delle norme molto rigide imposte da Apple per quanto riguarda il rilascio delle applicazione o anche il semplice test sui dispositivi fisici.

3.2 Objective-C

Un piccolo cenno al linguaggio di programmazione utilizzato per lo sviluppo su iPad è dovuto.
Objective-C è un linguaggio di programmazione orientato agli oggetti, sviluppato da Brad Cox durante gli anni ottanta. Come suggerisce il nome, è un'estensione del linguaggio C, con cui mantiene la totale retrocompatibilità. Questo permette blocchi di codice C immersi in un programma scritto in Objective-C. La sua grande diffusione è derivata dall'integrazione di questo linguaggio all'interno del sistema operativo Mac OS X di Apple.
La sintassi deriva da C, mentre per la parte di programmazione ad oggetti, l'ispirazione è stata il linguaggio Smalltalk.

3.3 Jolie

Jolie (Java Orchestration Language and Interpreter Engine) è un linguaggio di programmazione open-source orientato ai servizi. Jolie è un progetto open-source, sviluppato all'Università di Bologna. Con questo linguaggio è possibile scrivere dei servizi da zero, oppure utilizzare (anche integrandoli in nuovi servizi) quelli già esistenti.
La sintassi è molto pulita ed è ispirata al linguaggio C, e permette tramite i relativi costrutti, di implementare due tipologie di servizi:

- One-Way: il programma rimane in attesa di un messaggio ed eventualmente esegue del codice.
- Request-Response: rimane anch'esso in attensa di un messaggio, ma quando ne riceve uno esegue il codice associato e invia una risposta al mittente.

Il linguaggio, oltre al classico statement che permette l'esecuzione di una sequenza di operazioni (denotato da ;), possiede anche uno statement per il parallelismo (denotato da |) e un altro per permette l'esecuzione di una scelta non deterministica (denotato da ++). Questo linguaggio è stato utilizzato nello sviluppo del sistema per la gestione della parte server, implementando dei servizi appositi che i test neuropsicologici invocheranno per comunicare i loro risultati.
Per la comunicazione Jolie supporta diversi protocolli, tra cui il protocollo http e il protocollo sodep, esclusivamente implementato in questo linguaggio.
La scelta di questo linguaggio è dovuta alla sua facilità d'uso, che nasconde una complessa struttura, e dal poter scrivere software sotto forma di servizi web.

3.4 iOS2Jolie

Dai paragrafi precedenti si evince che il collegamento tra i test scritti su iOS in Objective-C e i servizi scritti in Jolie non è così scontato da eseguire.
Infatti, le due piattaforme, non usando lo stesso linguaggio, non hanno delle

funzione embedded comuni ad entrambi.
Diventa quindi d'obbligo l'aggiunta di una libreria di funzioni che permetta lo scambio di informazioni. Questa libreria, sviluppata da chi scrive e chiamata ″iOS2Jolie″, è in Objective-C ed affianca l'implementazione dei test. Essa contiene varie funzioni che permettono lo scambio di informazioni tra le due piattaforme usando il protocollo http, supportato pienamente da Jolie.
Infatti l'obiettivo di questa libreria di funzioni è adeguarsi ad uno dei vari formati proposti da Jolie. É stato scelto il protocollo http poiché è supportato da entrambi i lati ed ampiamente diffuso, mentre per il formato del corpo del messaggio è stato scelto xml perchè comodo ed adeguato alla tipologia di informazioni che verranno trasmesse. Tuttavia xml è stato usato solo per quei test i cui risultati sono dei semplici numeri o delle piccole stringe. Per alcuni test in cui si prevede l'invio di quantità di dati maggiori, come immagini, è stato adottato il formato multipart, che prevede di mandare più pacchetti http, anche in formati diversi tra loro (specificati nell'header di ogni pacchetto)

3.5 MySQL

Ultima ma non per questo meno importante, viene presentata in questo paragrafo la tecnologia usata per il salvataggio permanente dei dati.
É stato scelto il DBMS ″MySQL″, soprattutto perché è open-source. In questo modo viene evitato il vincolo con altri database proprietari ma allo stesso tempo si ha un sistema altamente sviluppato, essendo il DBMS maggiormente supportato dalla comunità open-source.
Per la gestione del database è stato utilizzato XAMPP, un pacchetto di servizi open-source che comprende, oltre al database mysql con il suo pannello di controllo in PHP (″phpMyAdmin″), server Apache, PHP e Perl.

Capitolo 4

Realizzazione dell'applicazione

Questo capitolo mostrerà la realizzazione concreta di tutte le parti del sistema. All'inizio la spiegazione riguarderà lo sviluppo dei test su iPad, dopodichè si passerà allo sviluppo della libreria di funzione iOS2Jolie fino a giungere all'implementazione del server Jolie.
Ogni singola parte si occuperà di mostrare ogni aspetto rilevante, sia riguardo il comportamento del software, sia sugli aspetti di sviluppo più importanti.

4.1 Realizzazione dei test neuropsicologici su iPad

Dopo un'attenta analisi dell'uso dei test e degli eventuali sviluppi futuri la conclusione è stata che è preferibile sviluppare un'applicazione separata per ogni test. In questo modo il dottore deve semplicemente avviare dall'iPad la prova che ha deciso di somministrare al paziente, senza altre procedure di routine. L'unica indispensabile richiesta è l'inserimento di alcuni dati per l'autenticazione del cliente.
Un altro vantaggio derivante dalla divisone delle applicazioni è l'isolamento di ognuna di esse. Eventuali problemi che potrebbero incorrere al software o al dispositivo, limiterebbero il danno solo al test in esecuzione in quel momento, non compromettendo i dati delle prove eseguite in precedenza o da eseguire successivamente.

Inoltre, ipotizzando una situazione in cui il paziente esegue i test direttamente dalla propria abitazione, una facilitazione giungerebbe proprio dal caricare sull'iPad in possesso dell'esaminando solo le applicazioni necessarie.
Di seguito verranno viste le parti comuni ad ogni applicazione, che si limitano all'interfaccia iniziale, e successivamente ognuna singolarmente.

4.1.1 Interfaccia iniziale

L'interfaccia pensata per queste applicazioni è volutamente minimale.
Essa presente solo due text field in cui vanno inseriti username e password forniti dal medico, e un pulsante che avvia il test.
Tuttavia, le funzionalità non visibili all'utente sono molte. Innanzitutto, essendo questa interfaccia la prima vista caricata dall'applicazione, si occupa di inizializzare tutte le strutture dati che rappresentano il test vero e proprio.
Dopo che il paziente inserisce i propri dati tramite una tastiera virtuale a comparsa e clicca sul pulsante ″inizia test″, viene effettuato un check della connessione tra iPad e server. Se il check ha successo, la prova parte senza che l'utente si accorga di niente, con abilitati i meccanismi di invio dei risultati al server, altrimenti viene mostrata una finestra di dialogo in cui si chiede se si vuole continuare il test in modalità di prova.
In caso di risposta positiva, l'applicazione è eseguita normalmente, con l'unica differenza derivante dalla totale inibizione dei meccanismi di comunicazione con il server.
In un primo momento si è pensato di implementare la registrazione dei pazienti direttamente nelle applicazioni, ma si è giunti alla conclusione che questa decisione non fosse corretta in quanto l'utente avrebbe usufruito di funzionalità non dedicategli.
Sarà l'applicazione destinata al dottore che conterrà al suo interno tutta la gestione dei pazienti, a partire dalla registrazione sino all'andamento nei test ed alla possibile cancellazione.
L'interfaccia sull'iPad alla fine risulterà così:

1) schermata di scelta del test

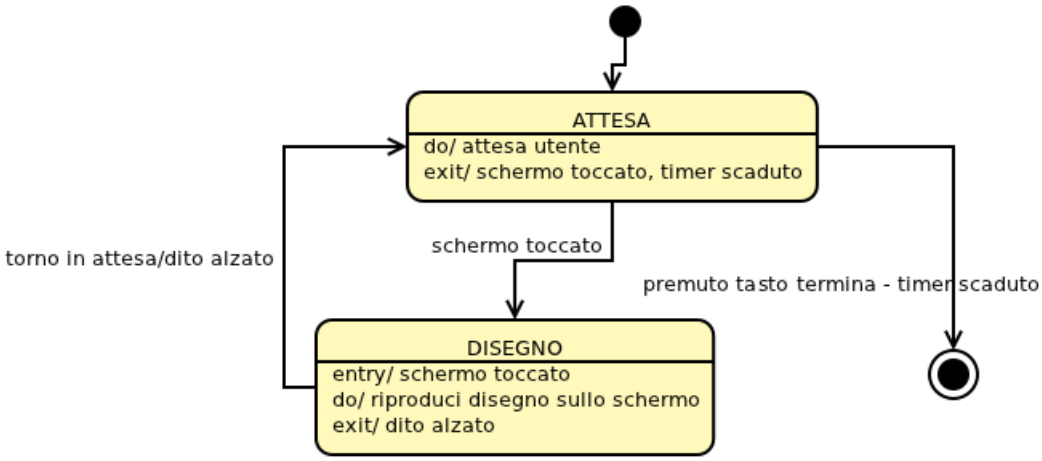
2) schermata iniziale

3) inserimento dei dati tramite tastiera virtuale

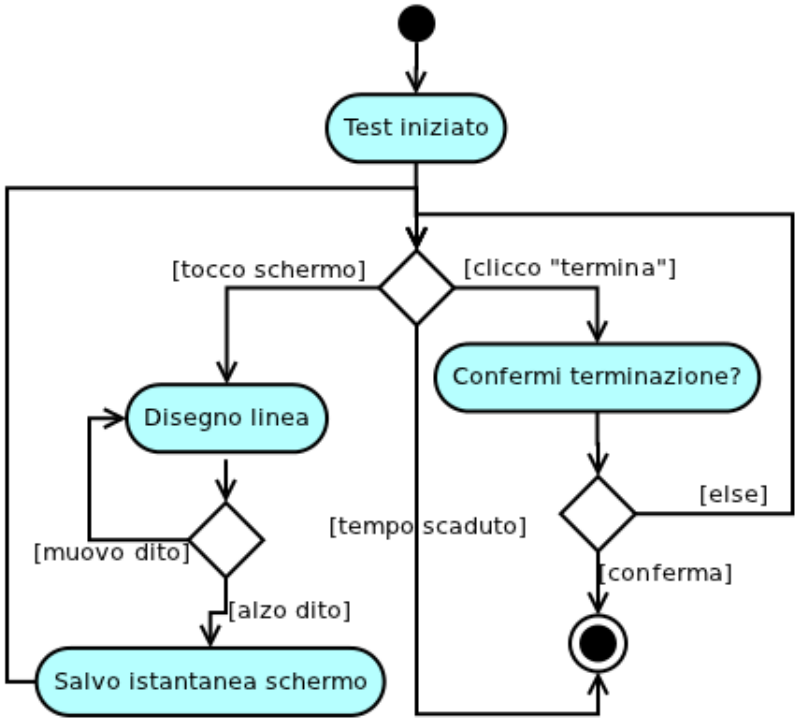
4) alert per l'attivazione della modalità di prova

4.1.2 Clock Test

Il Clock Test è il più lineare dei 4 a livello di sviluppo, e la conferma arriva direttamente dai diagrammi di sviluppo.

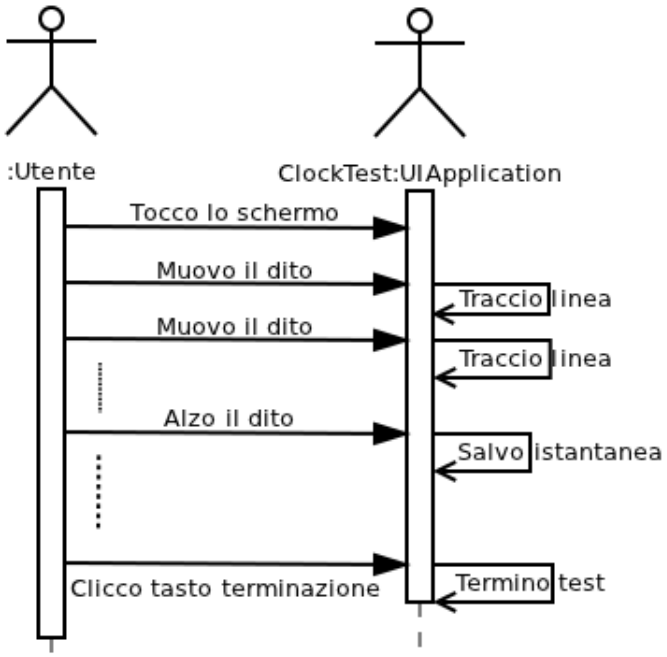


Come si evince dal diagramma degli stati, l'applicazione sostanzialmente o si trova in attesa di un input dell'utente (disegno di una linea o click sul pulsante di terminazione del test) oppure disegna la linea rimarcando il movimento del dito sullo schermo.
Il passaggio da uno stato all'altro è sancito dal tocco del dito sul touch screen.

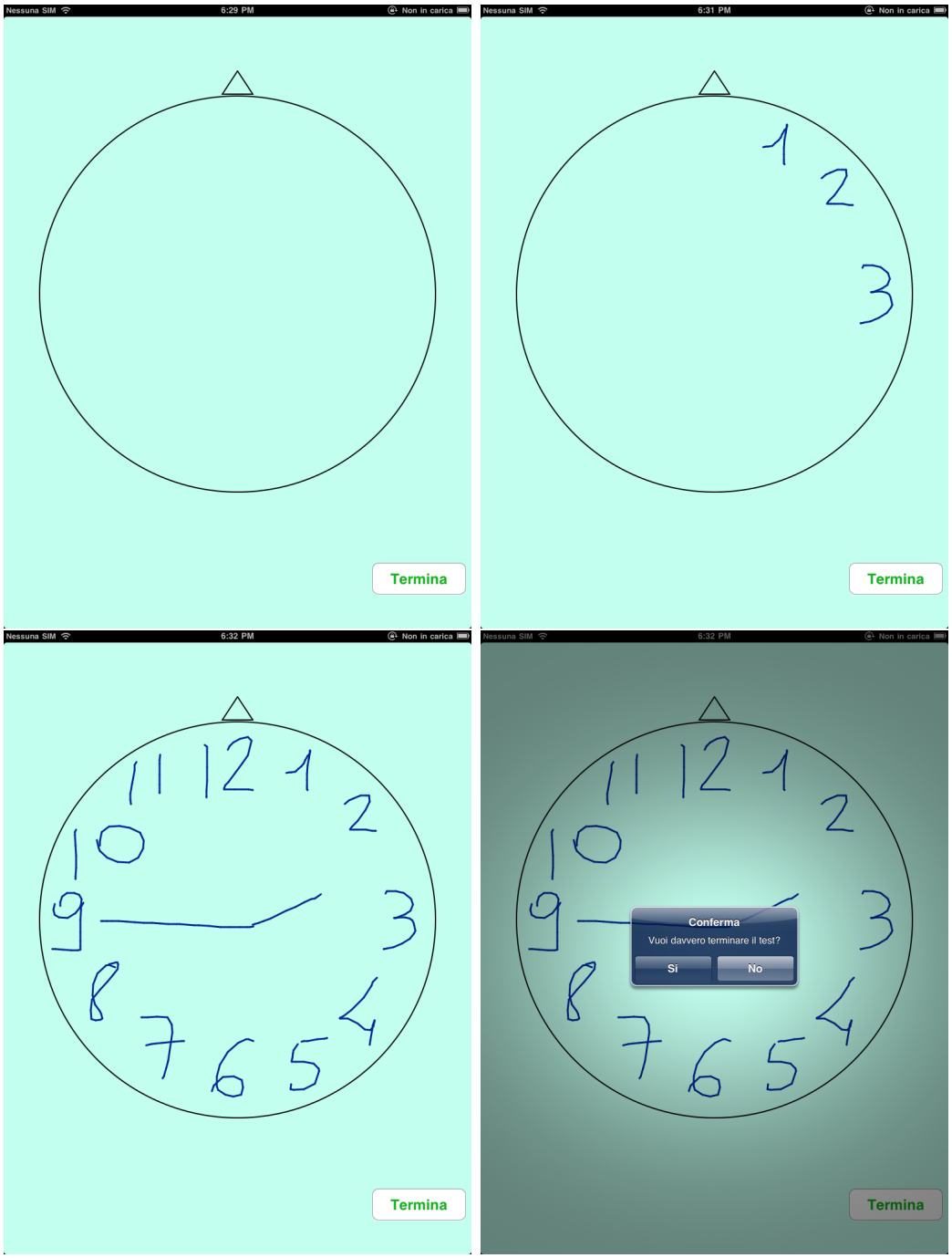


Invece dal diagramma delle attività si capiscono molto meglio le reali attività svolte dall'applicazione.
Dopo che il dito tocca lo schermo, ogni volta che esso viene mosso l'applicazione disegna una linea per il tratto seguito dal dito, e in più, quando l'utente alza il dito, viene salvata ogni volta un'istantanea dello schermo. Questa feature è stata introdotta per permette al neuropsicologo di avere traccia di tutto il test e non solo del risultato finale. Infatti il medico è interessato anche ad alcuni eventi temporali, come ad esempio l'ordine con cui vengono scritti i numeri nell'orologio, o se il paziente scrive prima i numeri 3,6,9,12 nel quadrante per darsi una misura visiva, o altri ancora.
Inoltre il diagramma delle attività denota meglio l'esistenza di un timer globale, che rappresenta il tempo a disposizione del paziente. La terminazione dell'applicazione può quindi avvenire o se il timer globale scade (per il Clock Test equivale a 300 secondi) oppure se l'utente clicca sul pulsante per terminare lui stesso l'esecuzione. Verrà anche chiesta una conferma di terminazione per maggiore sicurezza.

A seguire, un esempio di esecuzione mostrato tramite un diagramma di sequenza:



L'applicazione sull'iPad alla fine risulterà così:



1) schermata iniziale

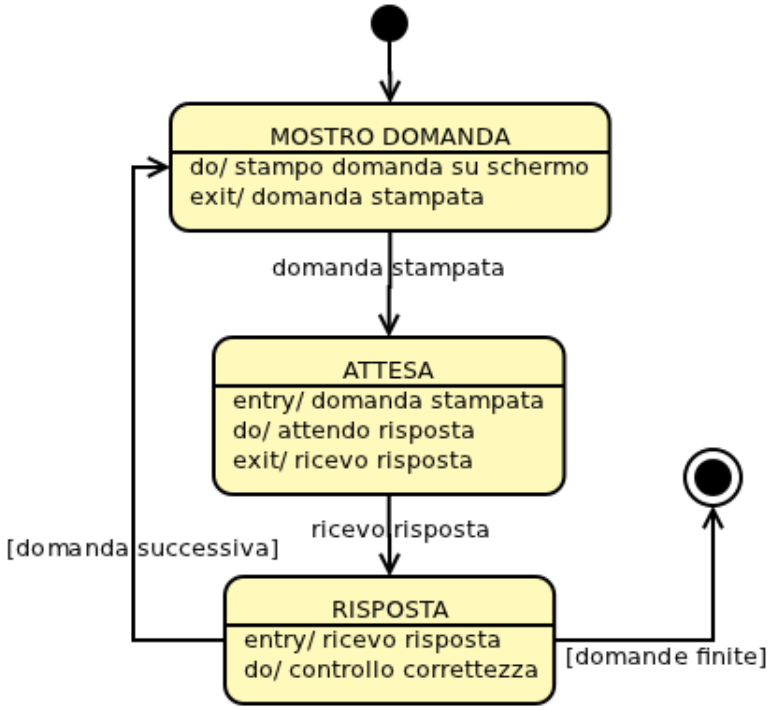
2) dopo l'inserimento di alcuni numeri

3) test completato correttamente

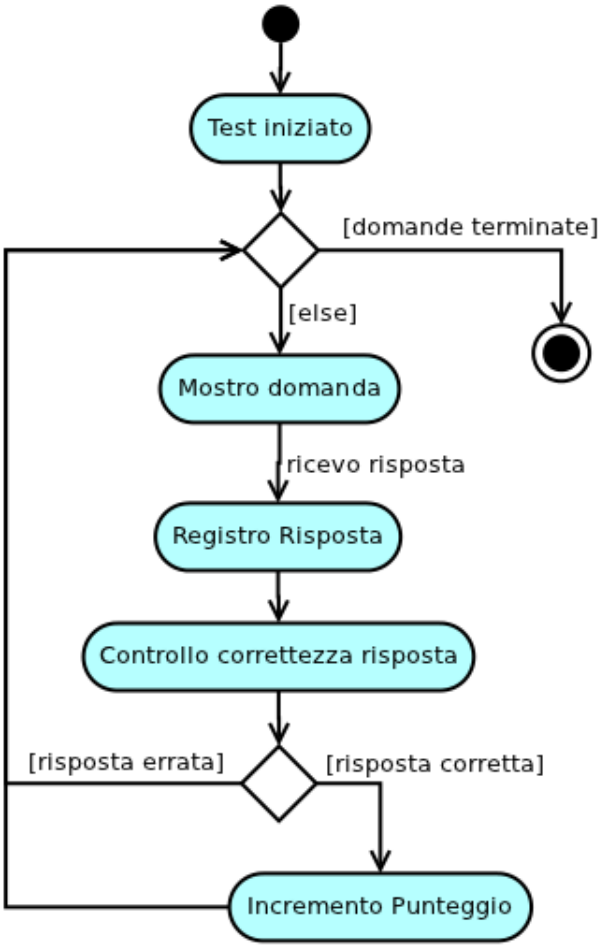
4) schermata di conferma terminazione test

4.1.3 Stime Cognitive Test

Stime Cognitive Test ha un'esecuzione lineare. Infatti le iterazioni con l'utente sono minime, in quanto si riducono ad alcune risposte numeriche.

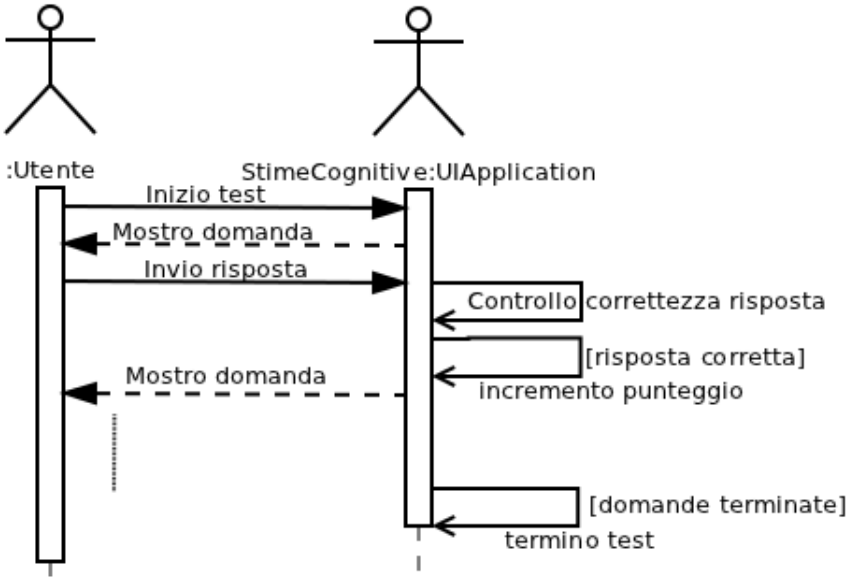


Come si vede dal diagramma degli stati l'applicazione attende la risposta del paziente e, quando la riceve, la registra e propone la domanda successiva, sino all'esaurimento dei quesiti.

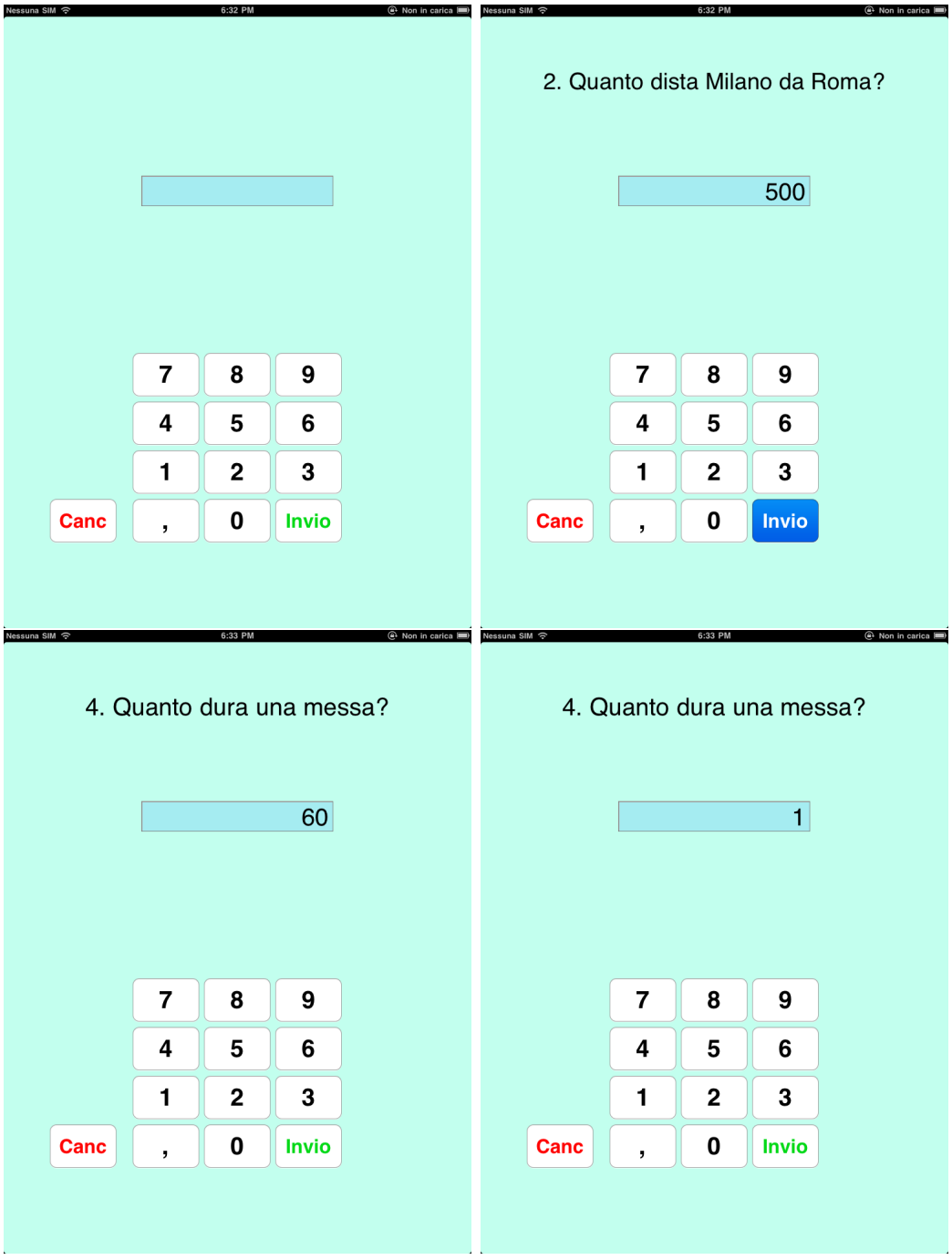


Il diagramma delle attività conferma la linearità del test, che per ogni domanda, dopo averla mostrata all'utente, attende una risposta, ne controlla la correttezza e incrementa il punteggio se necessario.
Bisogna sottolineare due aspetti molto importanti riguardanti l'implementazione di Stime Cognitive. Innanzitutto è voluta l'assenza dell'indicazione delle unità di misura per le risposte in attesa. Ovviamente la correttezza della risposta è giustificata sulle misure più probabili in base alla domanda. Ad esempio, alla domanda ″Quanto dura una messa?″, le risposte plausibili sono accettate sia in ore che in minuti. In più, siccome tutte le risposte sono di tipo numerico, è stato creato un semplice tastierino numerico, che facilita l'inserimento delle risposte. Senza questo accorgimento, sarebbe stata usata la tastiera virtuale di iOS, che avrebbe sicuramente creato confusione, visto

il grande numero di pulsanti a disposizione, in questo caso molto più del necessario.
A seguire, un esempio di esecuzione mostrato tramite un diagramma di sequenza:



L'applicazione sull'iPad alla fine risulterà così:



1) schermata iniziale

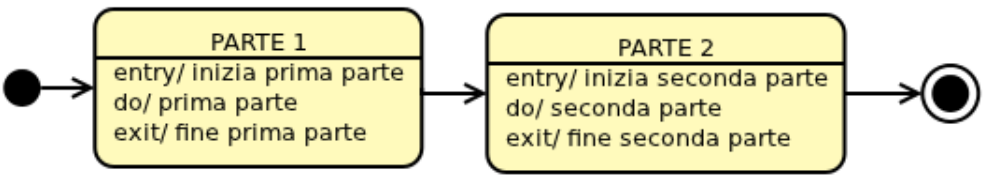
2) premuto invio per dare una risposta

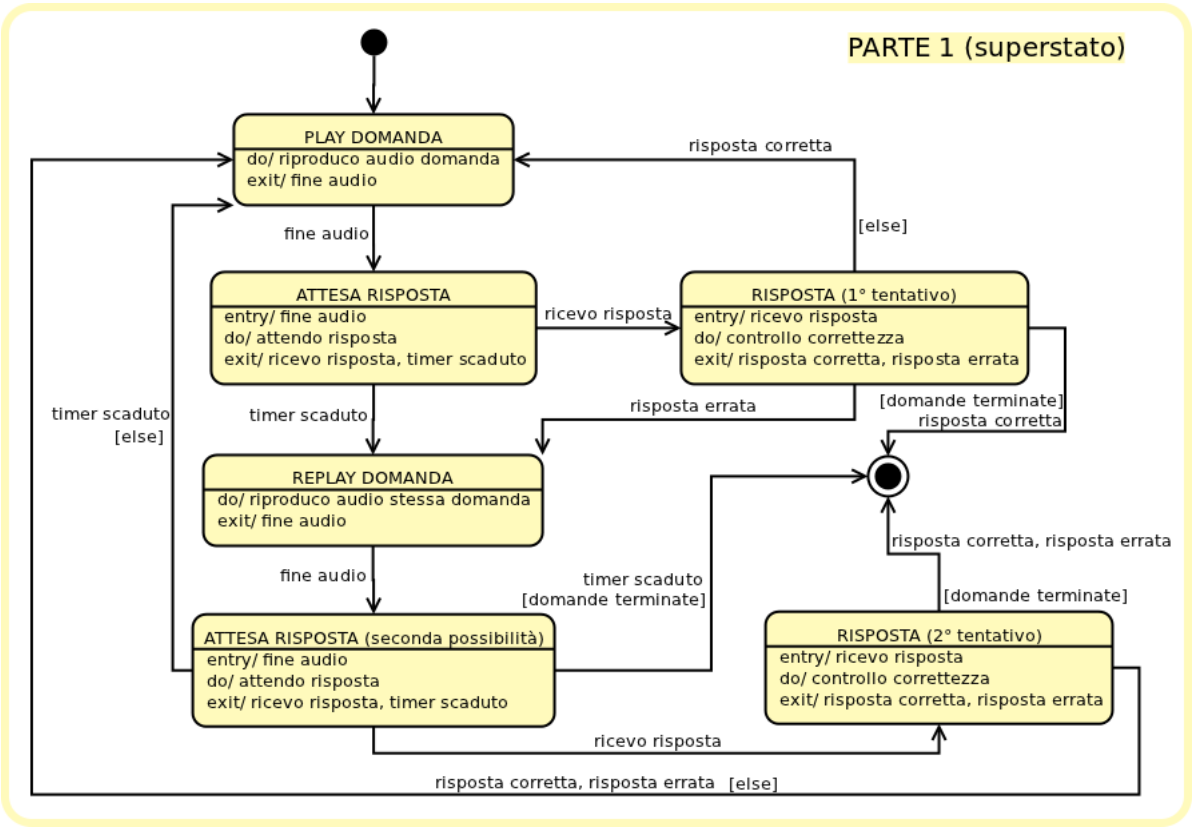
3) risposta alla domanda 4 con unità di misura in minuti

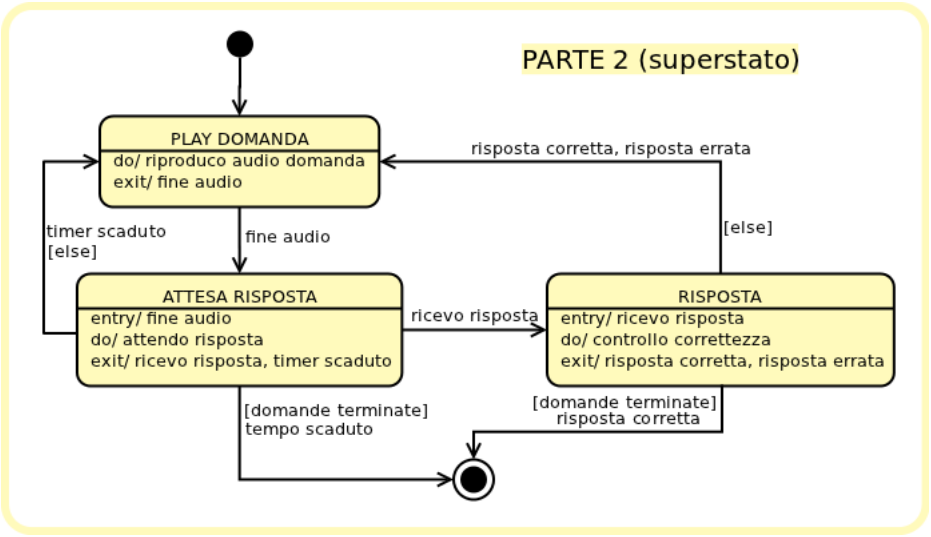
4) risposta alla domanda 4 con unità di misura in ore

4.1.4 Token Test

Il Token Test è sicuramente l'applicazione più complessa, in quanto contiene molte funzionalità che dovevano essere implementate per riprodurre fedelmente la natura del test.
Come è stato mostrato nella spiegazione di questa prova, c'è una netta separazione che divide il test in due parti. Infatti, se fino alla domanda 24 in caso di risposta scorretta/incompleta veniva data una seconda possibilità al paziente, dalla domanda 25 fino alla fine del test, questa feature non deve esistere.
Questa eventuale ripetizione ha influito su tutta l'implementazione del test, perciò la decisione finale è stata separare nettamente le due parti, come se fossero progettati separatamente e poi successivamente accostate. A spingere ulteriormente la scelta in questa direzione si è aggiunta la diversa tipologia di domande che le due parti dell'applicazione contengono. Mentre nella prima parte le risposte consistono esclusivamente nella scelta di uno o più token, nella seconda parte ci sono alcune domande che richiedono lo spostamento di alcuni token e il calcolo di relative distanze. Questa differenza costringe lo sviluppatore ad adottare due metologie di programmazione diversa, portando quindi alla scelta di cui sopra.
Un'altra differenza sostanziale rispetto allo Stime Cognitive Test, è la differente somministrazione delle varie domande. Nel test precedente le domande venivano stampate su schermo, mentre in questo test, essendo di comprensione verbale, le domande sono in formato audio.
Detto ciò, il diagramma degli stati sarà composto da due superstati: uno per ogni parte del test.

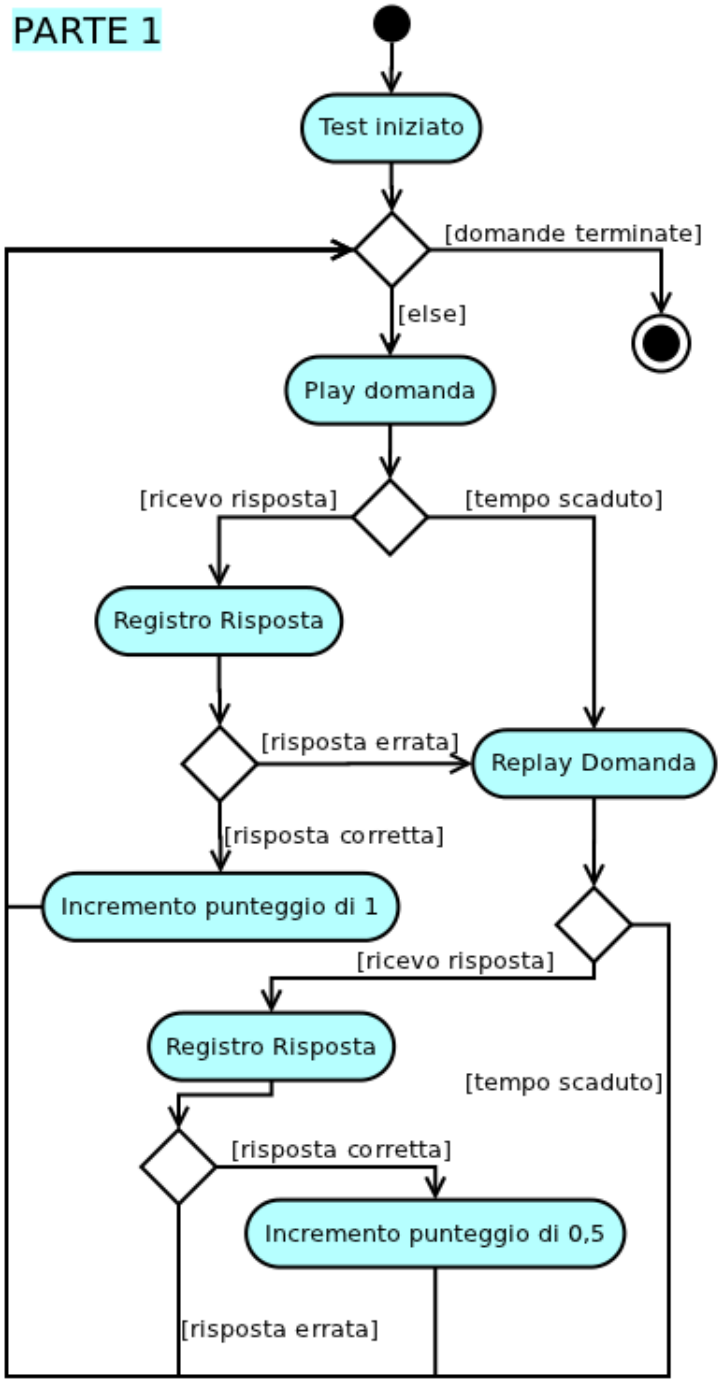


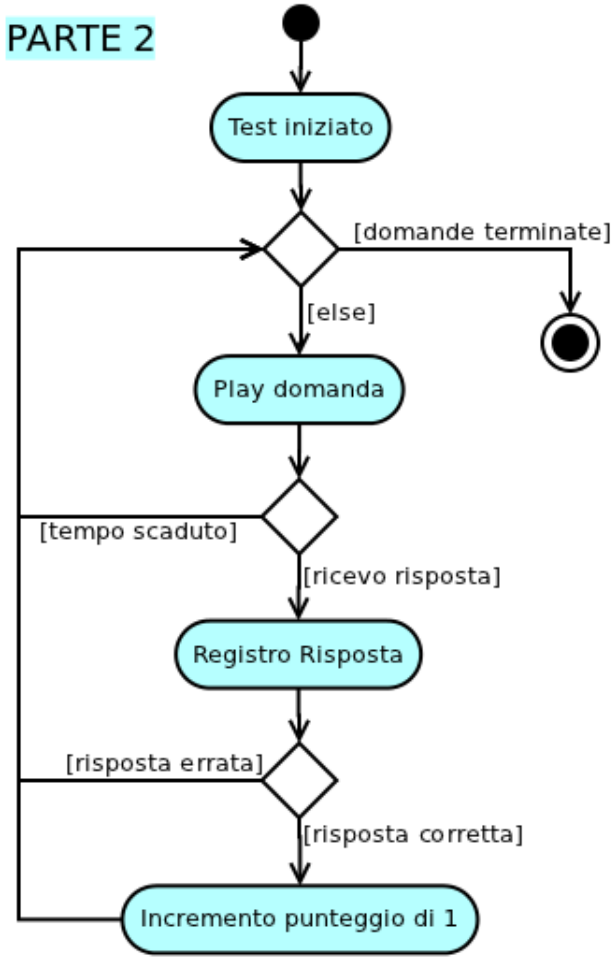




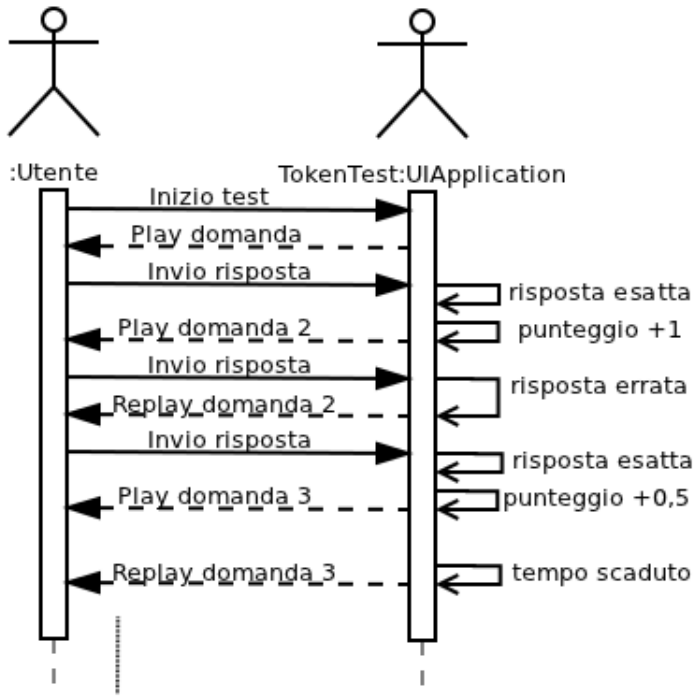
I due superstati mostrano nel modo più chiaro possibile le caratteristiche previste dall'applicazione. Una piccola precisazione sulla domanda numero 29 (che fa parte della seconda parte): qualunque risposta data è errata, la risposta giusta è possibile soltanto facendo scadere il tempo. Anche se nel

diagramma degli stati non è presente, ovviamente questa eccezione è correttamente gestita dall'applicazione.
Anche per il diagramma delle attività vale lo stesso discorso fatto per il diagramma degli stati: le due parti saranno mostrate separatamente.

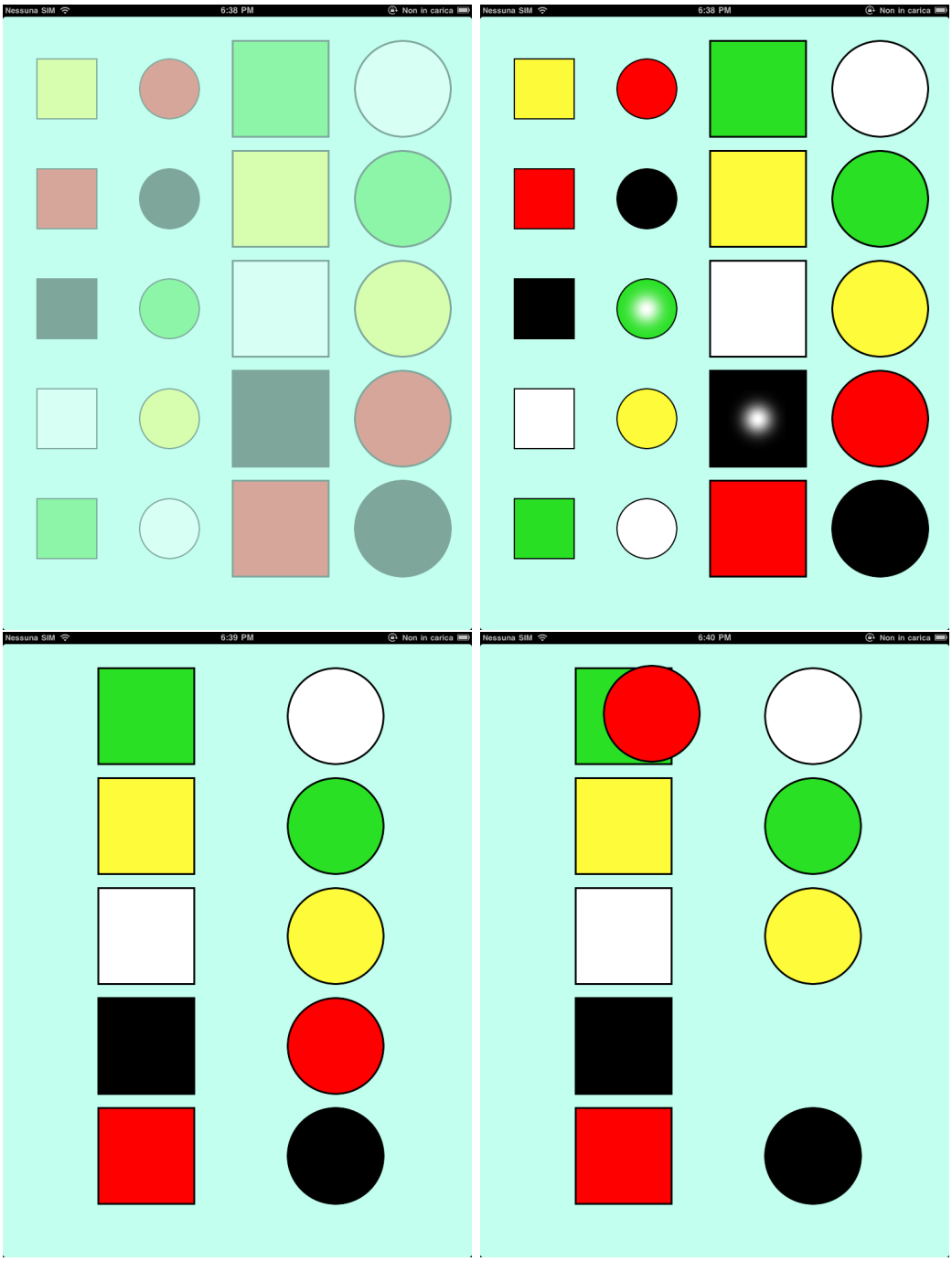

PARTE 1
Test iniziato
[domande terminate]
[else]
Play domanda
[ricevo risposta]
[tempo scaduto]
Registro Risposta
[risposta errata]
Replay Domanda
[risposta corretta]
Incremento punteggio di 1
[ricevo risposta]
Registro Risposta
[tempo scaduto]
[risposta corretta]
Incremento punteggio di 0,5
[risposta errata]



I diagrammi delle attività in questo test ricalcano molti bene i diagrammi degli stati. Mostrano inoltre la gestione del punteggio: viene aggiunto 1 punto se la risposta corretta è data dopo la prima presentazione della domanda, 0.5 punti se dopo la seconda presentazione, 0 altrimenti. Anche qui la particolarità della domanda 29 non è visibile (andrebbe aggiunto un punto quando scade il timer).
In questo test il diagramma di sequenza è molto interessante:



Rispetto ai due test precedenti, la linearità della sequenza di azioni tra utente ed applicazione è molto bassa.
Token Test risulta essere una prova molto lunga e complessa a livello di sviluppo, specie per le varietà di domande ma soprattutto per la varietà di risposte che devono essere gestite. Il risultato però è molto appagante, e ritrae alla perfezione la controparte non digitale.
L'applicazione sull'iPad alla fine risulterà così:



1) schermata opaca usata mentre è in play una domanda

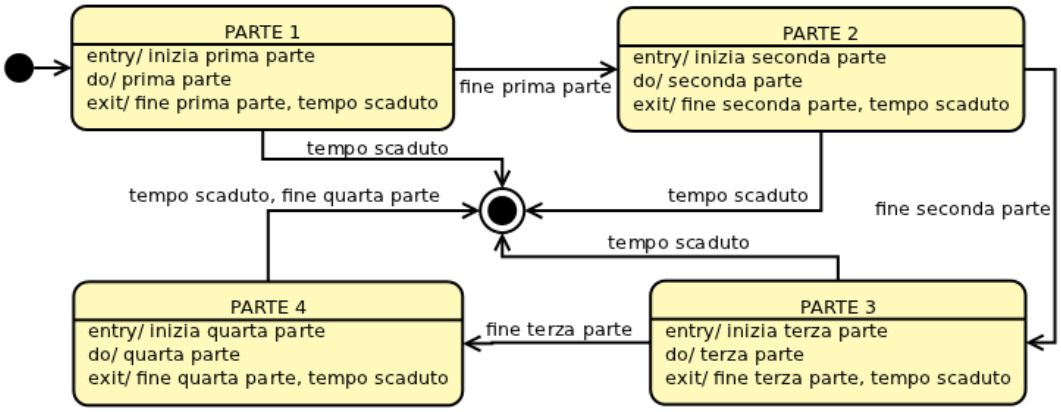
2) momento di una risposta multitouch

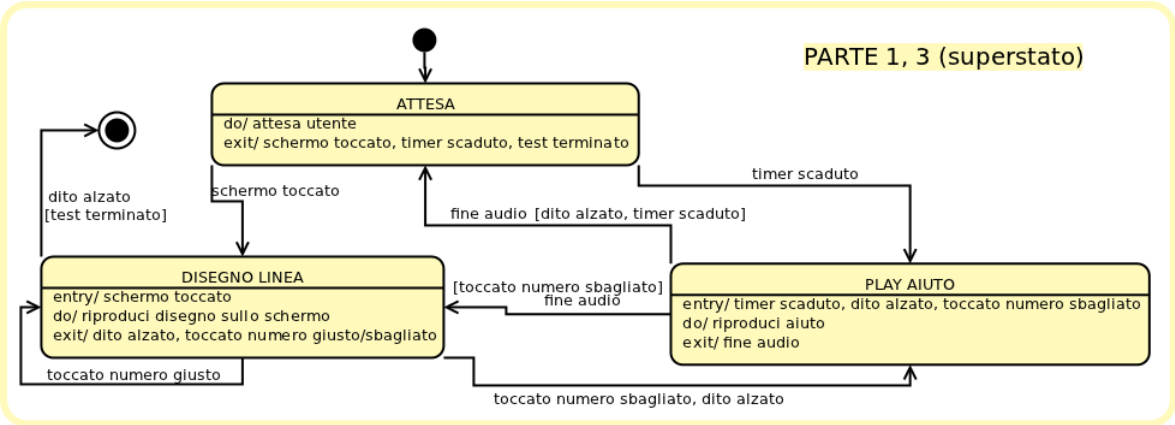
3) schermata usata per le domande con soli gettoni grandi

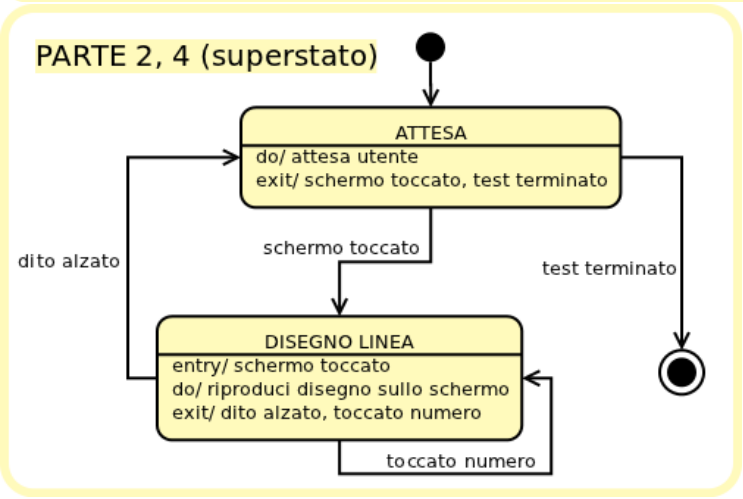
4) esempio di domanda che include movimento di gettoni

4.1.5 Trail Making Test

Il Trail Making Test è un'applicazione divisa in 4 sottotest. Nel primo si devono collegare con una linea i numeri da 1 a 8, nel secondo i numeri da 1 a 25, nel terzo 8 numeri e lettere alternati, nel quarto 25 numeri e lettere alternati. Le prove a 8 elementi sono delle prove preventive somministrate per verificare la possibilità che il paziente riesca ad eseguire il test a 25 elementi, e non contribuiscono al punteggio finale. Infatti, se non dovesse portare a termine queste prove preventive, il resto del test non è somministrato.
Per gestire questa eventualità, è dato un tempo massimo di 5 minuti per svolgere i pre-test a 8 elementi, mentre per i test completi sono dati rispettivamente 240 e 420 secondi.
Essendo i test di crescente difficoltà, anche per la prova a 25 elementi composti da soli numeri, se il paziente non terminasse in tempo la prova sarebbe data per fallita, senza passare alla parte terza.
Dal punto di vista dello sviluppo, le parti sono identiche a coppie (prima e terza, seconda e quarta). L'unica differenza è data dalle label dell'interfaccia grafica, che presentano lettere oltre che numeri nella terza e quarta prova. Perciò i diagrammi di sviluppo saranno riferiti alle coppie di sottotest.

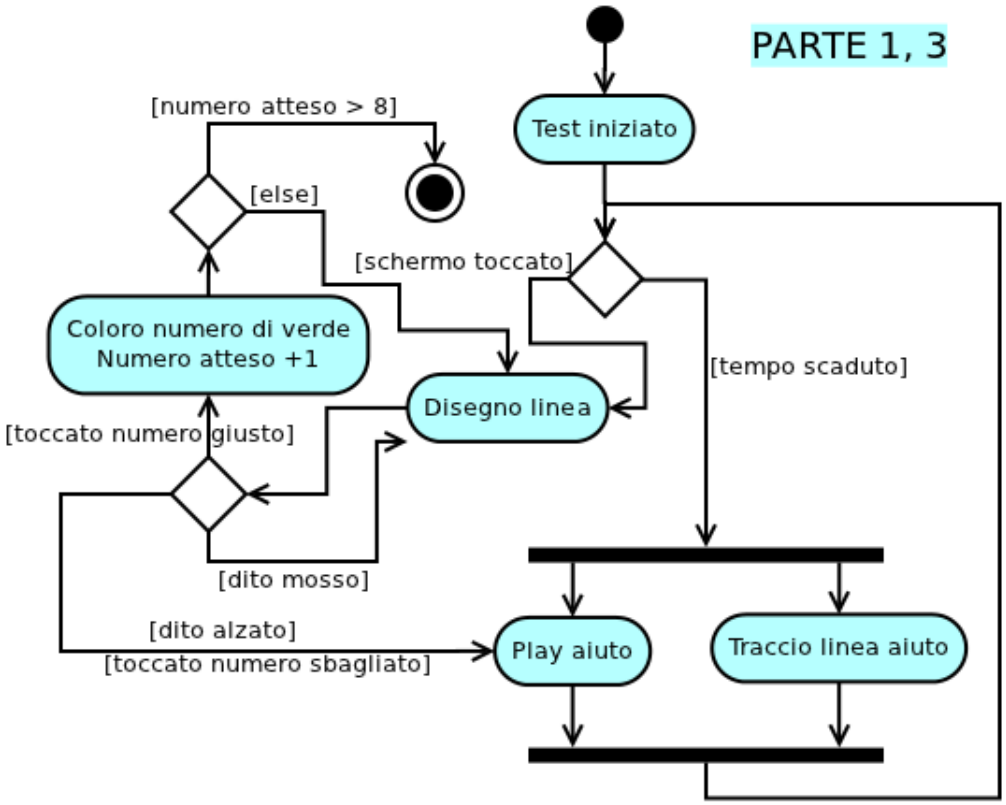


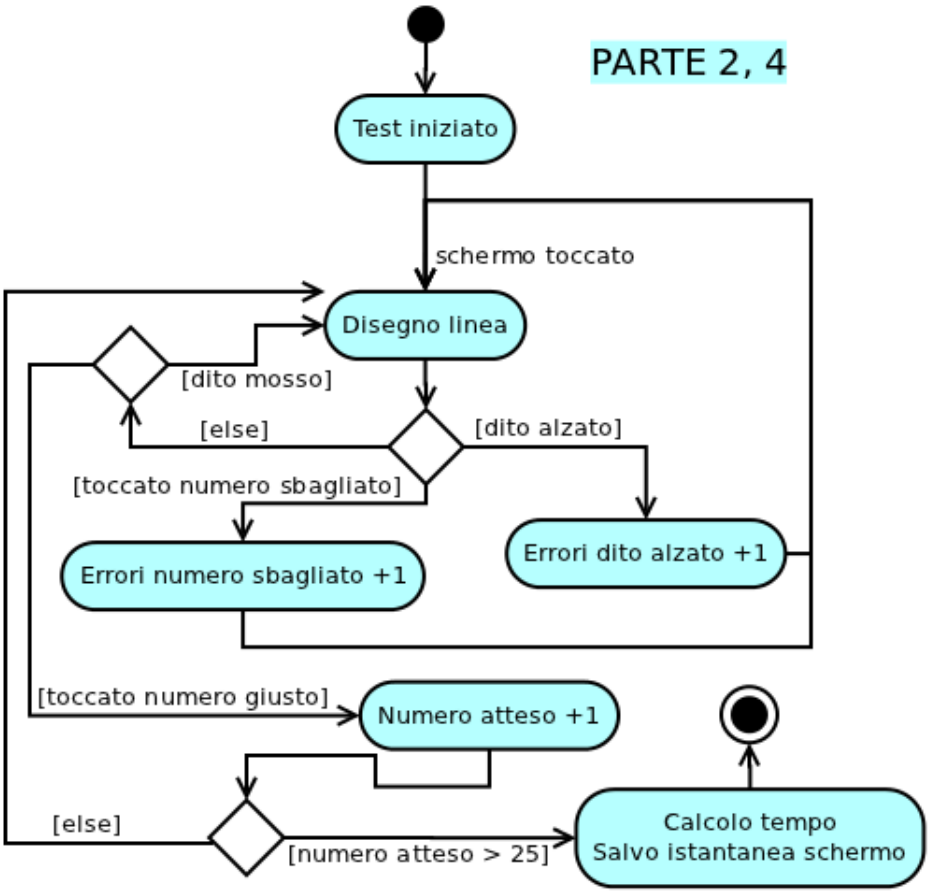




Dai diagrammi degli stati si nota la principale differenza tra le due coppie di test. Nella coppia a 8 elementi è stato implementato un sistema di aiuti che ha il compito di guidare il paziente verso la totale comprensione del test, che verrà somministrato pienamente solo dopo questa prova introduttiva.
Dal diagramma degli stati riferito alle parti 1 e 3 si capiscono anche le tipologie di errori in cui l'utente può incappare: alzare il dito, toccare un numero sbagliato o aspettare del tempo senza svolgere azioni. Mentre nei primi due casi viene solo redarguito con una frase che evidenzia l'errore appena commesso, nel terzo caso, oltre alla frase, è disegnata automaticamente la prossima linea che dovrebbe essere stata tracciata dal paziente, esortandolo a riprodurla.
Un ulteriore aiuto viene dalla colorazione verde dello sfondo di un numero selezionato correttamente.

I diagrammi delle attività mostrano meglio questi aspetti.





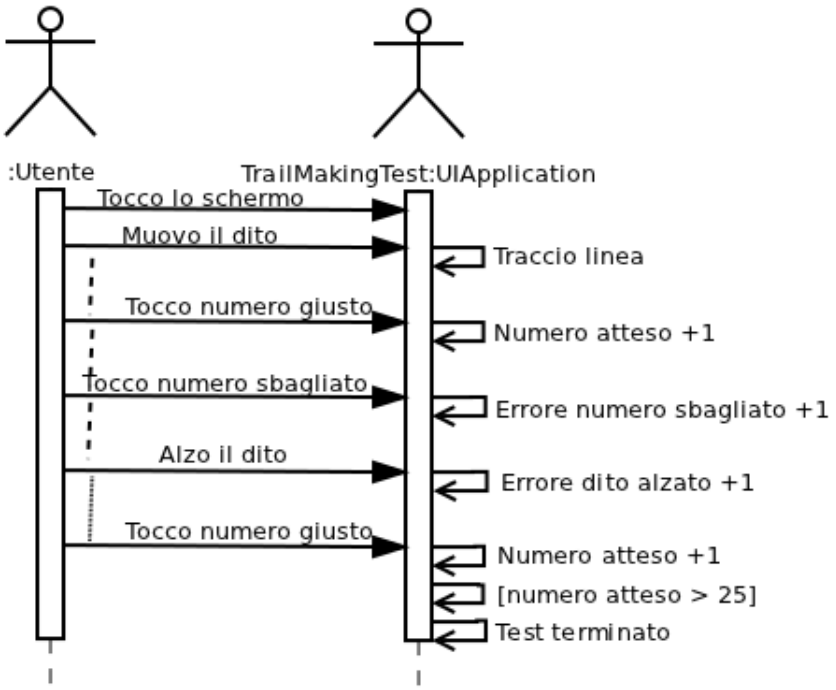
Da questi diagrammi si nota molto meglio la differente gestione degli errori a seconda della parte del test in cui ci troviamo. Dal secondo notiamo come ad ogni errore ci sia solo un incremento, ma l'utente non viene avvisato in alcun modo dell'accaduto.

Inoltre, come nel Clock Test, viene salvata un'istantanea dello schermo, ma questa volta solo alla fine (e solo nelle parti 2 e 4). Infatti al medico, visto che come risultato ottiene solo il numero degli errori e il tempo impiegato, risulta molto utile quest'immagine, in quanto può studiare meglio eventuali errori commessi.

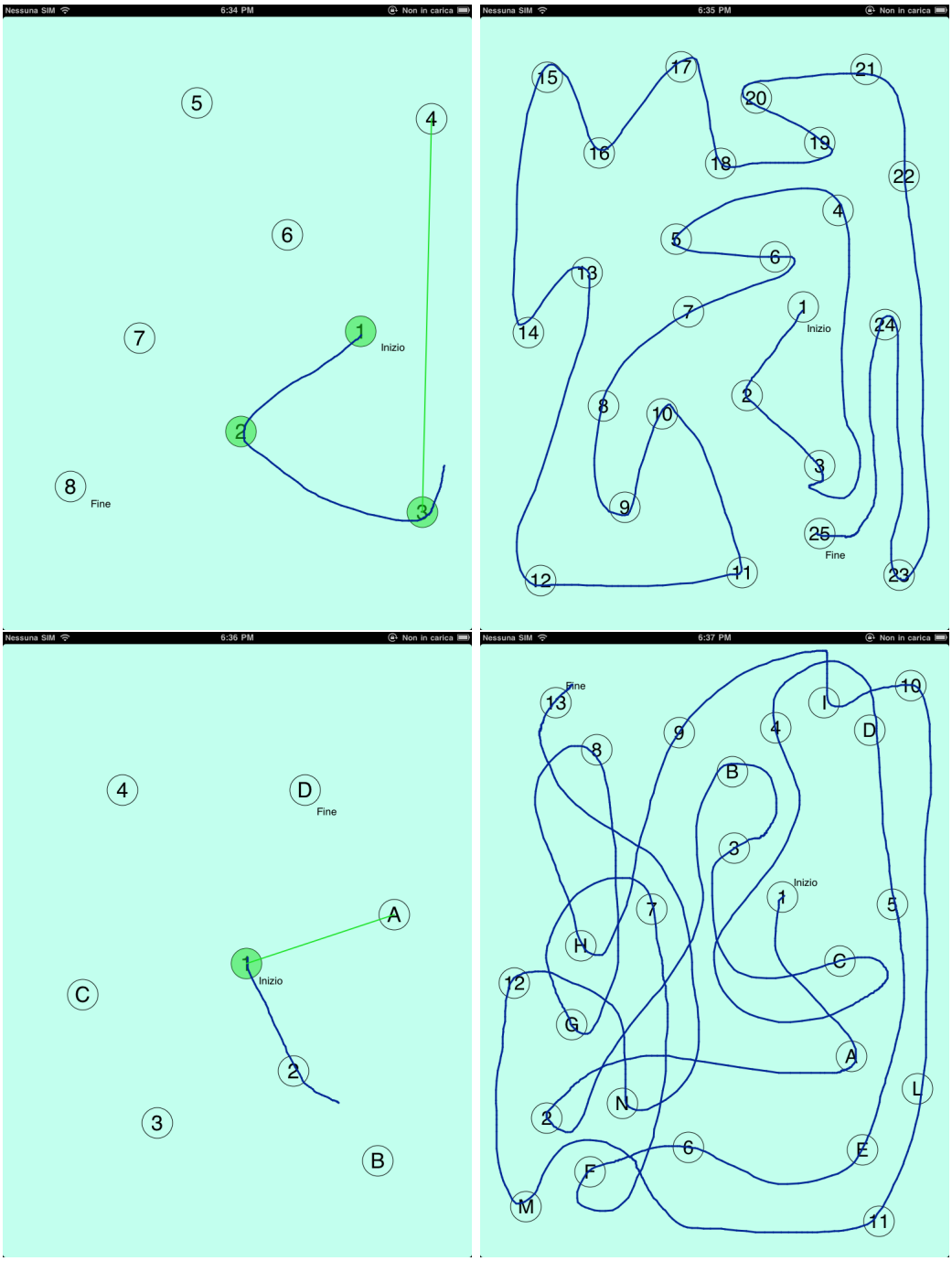
I diagrammi non mostrano l'esistenza di un timer globale la cui scadenza, a prescindere dal punto di esecuzione in cui ci troviamo, innesca la fine forzata del test.

Come al solito, ecco una possibile esecuzione mostrata da un diagramma di

sequenza:



Da sottolineare che il test, per essere terminato correttamente prima che il timer globale scada, costringe il paziente a toccare la sequenza corretta di numeri, dove ovviamente tra un numero e l'altro sono possibili gli errori. Se ad esempio dopo aver toccato il numero 1 si passa al numero 3, per continuare correttamente la prova è obbligatorio passare sul numero 2 e poi riprendere la sequenza corretta (cioè bisogna toccare nuovamente il numero 3).
L'applicazione sull'iPad alla fine risulterà così:



1) test introduttivo A con aiuto per continuare

2) test con soli numeri completo

3) test introduttivo B con aiuto dopo errore

4) test con numeri e lettere completo

4.2 Realizzazione del server

Il server è il centro nevralgico del sistema.
Le sue funzioni girano intorno ad un solo obiettivo: la gestione dei risultati.
In un primo momento una soluzione plausibile era che ogni dottore avesse il suo iPad, su cui i suoi pazienti svolgono i test, e su cui erano presenti i meccanismi per studiare i risultati salvati sul dispositivo.
Nonostante questa eventualità fosse molto più semplice da realizzare, essa limitava le funzionalità del progetto. Innanzitutto i vantaggi temporali erano molto minori, in quanto tutti i pazienti avrebbero dovuto usare lo stesso dispositivo, annullando qualsiasi tipo di parallelismo. In più si sarebbe creata una dipendenza dalla piattaforma usata, limitando gli sviluppi futuri, sia per quanto riguarda l'uso di dispositivi non Apple, sia per quanto riguarda ulteriori funzionalità derivanti dalla centralizzazione dei risultati.
É stato usato Jolie per la sua nuova concezione di SOA (Service-Oriented Architecture). In questo linguaggio qualsiasi funzione è un servizio. Questa scelta porta grandi vantaggi alla creazione di un server.
I vantaggi della SOA sono lampanti:

- Migliore organizzazione delle funzionalità
- Riutilizzo del codice
- Maggiore integrazione tra i vari servizi

Passiamo all'analisi dei servizi sviluppati per questo progetto.
Per comunicare con l'esterno Jolie usa le ″**inputPort**″ e le ″**outputPort**″. Le prime aprono un collegamento in input, dichiarando, tramite un'interfaccia, i nomi dei servizi a disposizione dell'ambiente esterno. Invece le seconde indicano i servizi a disposizione del server che è possibile richiamare dall'esterno.
In questo sistema è prevista un'unica inputPort, in ascolto sulla porta 2001, con 5 servizi a disposizione: uno per ogni test più un servizio per effettuare un check della connessione al server. I servizi relativi ai test effettuano poi un'operazione di tipo **insert** sul database mysql connesso a jolie.

É prevista la creazione di un'altra inputPort per gestire invece i servizi relativi al prelievo dei dati dal database, usati nell'applicativo destinato al dottore.

4.2.1 Database

Il database ha una struttura piuttosto semplice.
Questo il codice SQL usato per la creazione del database e delle tabelle, successivamente analizzato:

```
1  #creazione database
2  CREATE DATABASE `jolie_server` DEFAULT CHARACTER SET utf8
       COLLATE utf8_general_ci;
3
4  #tabella clock test
5  CREATE TABLE `jolie_server`.`clock_test` (
6  `cf` VARCHAR( 16 ) CHARACTER SET utf8 COLLATE utf8_general_ci
       NOT NULL ,
7  `data` TIMESTAMP NOT NULL ,
8  `count` SMALLINT NOT NULL ,
9  PRIMARY KEY ( `cf` , `data` )
10 ) ENGINE = MYISAM CHARACTER SET utf8 COLLATE utf8_general_ci;
11
12 #tabella stime cognitive
13 CREATE TABLE `jolie_server`.`stime_cognitive_test` (
14 `cf` VARCHAR( 16 ) CHARACTER SET utf8 COLLATE utf8_general_ci
       NOT NULL ,
15 `data` TIMESTAMP NOT NULL ,
16 `punteggio` INT( 1 ) NOT NULL ,
17 `risp1` FLOAT NOT NULL ,
18 `risp2` FLOAT NOT NULL ,
19 `risp3` FLOAT NOT NULL ,
20 `risp4` FLOAT NOT NULL ,
21 `risp5` FLOAT NOT NULL ,
22 PRIMARY KEY ( `cf` , `date` )
23 ) ENGINE = MYISAM CHARACTER SET utf8 COLLATE utf8_general_ci;
24
25 #tabella trail making test
26 CREATE TABLE `jolie_server`.`trail_making_test` (
27 `cf` VARCHAR( 16 ) CHARACTER SET utf8 COLLATE utf8_general_ci
       NOT NULL ,
28 `data` TIMESTAMP NOT NULL ,
29 `type` VARCHAR( 1 ) CHARACTER SET utf8 COLLATE
       utf8_general_ci NOT NULL ,
30 `secondi` SMALLINT NOT NULL ,
```

```
31 `err_numeri` SMALLINT NOT NULL ,
32 `err_dito` SMALLINT NOT NULL ,
33 PRIMARY KEY ( `cf` , `date` , `type` )
34 ) ENGINE = MYISAM CHARACTER SET utf8 COLLATE utf8_general_ci;
35
36 #tabella token test
37 CREATE TABLE `jolie_server`.`token_test` (
38 `cf` VARCHAR( 16 ) CHARACTER SET utf8 COLLATE utf8_general_ci
       NOT NULL ,
39 `data` TIMESTAMP NOT NULL ,
40 `punteggio` FLOAT NOT NULL ,
41 PRIMARY KEY ( `cf` , `data` )
42 ) ENGINE = MYISAM CHARACTER SET utf8 COLLATE utf8_general_ci;
```

Tutti i campi *text* sono codificati con utf-8, standard per la codifica di caratteri Unicode. Come si vede dal codice SQL le tabelle previste sino ad adesso sono 4, una per ogni test.

Per tutti i test il dato che ci permette di creare un collegamento con il paziente è il suo codice fiscale, nella maggior parte dei casi univoco. Sarà comunque effettuato un controllo in fase di registrazione del paziente per evitare codici fiscali identici. Inoltre un dato importantissimo presente in tutti i test è la data, nel formato standard di SQL (TIMESTAMP) a 19 caratteri ″**yyyy-mm-dd hh:mm:ss**″.

La tabella del clock test prevede in più solo il numero di immagini salvate per una determinata esecuzione della prova (di tipo *SMALLINT*). Infatti questo test è quello che al momento ha più bisogno del medico per dare un primo risultato.

La tabella dello stime cognitive test invece contiene un campo *punteggio* che può andare da 0 a 5, e rappresenta il risultato del test. Per un maggiore dettaglio, sono salvate anche le 5 singole risposte. Mentre il punteggio è un intero di tipo *SMALLINT*, le singole risposte sono rappresentate dal tipo *FLOAT*, potendo avere una parte decimale.

La tabella del token test semplicemente contiene il risultato del test (di tipo *SMALLINT*), un punteggio che può andare da 0 a 36.

Per finire, la tabella del trail making test contiene il tipo di prova a cui si riferisce quella entry (A o B), i secondi impiegati e gli errori commessi, divisi tra quelli in cui si è toccato il numero sbagliato e quelli in cui si è alzato il

dito dallo schermo. Tolto il tipo della prova, che è una stringa di un solo carattere (*VARCHAR(1)*), gli altri campi sono numerici (*SMALLINT*).
Come si vede dal database, per adesso non sono previsti campi che contengono l'immagine per i test che ne producono (clock e trail making), in quanto la codifica usata da iOS non è riconosciuta dal tipo BLOB di MySQL, il tipo designato per il salvataggio di dati binari. Pur essendo la scelta criticabile, si sottolinea che non è voluta, ma che è stato impossibile, per adesso, convertire i dati in un formato riconosciuto dal database. Dall'altra parte, salvando le immagini direttamente sul file system del server, diventa più facile recuperarle da iPad tramite un invio diretto dei dati così come erano stati ricevuti, saltando una sicura riconversione di questi ultimi.

4.2.2 Servizi Jolie

I servizi Jolie rispecchiano la natura pulita di questo linguaggio. Anche l'uso del database o di file esterni è di estrema immediatezza, grazie alle librerie di servizi già fornite con il linguaggio.
Questo il codice scritto per la parte server del sistema, seguita come al solito da un'analisi della stessa:

```
1  // Server services
2
3  include "database.iol"
4  include "console.iol"
5  include "file.iol"
6
7  execution { concurrent }
8
9  interface ServerInterface{
10   RequestResponse:
11     checkConnection,
12     clockTest,
13     stimeCognitiveTest,
14     tokenTest,
15     trailMakingTest
16 }
17
18 inputPort Server {
19   Location: "socket://localhost:2001"
```

```
20    Protocol: http
21    Interfaces: ServerInterface
22  }
23
24  init {
25    with (connectionInfo){
26      .host = "localhost";
27      .driver = "mysql";
28      .database = "jolie_server";
29      .username = "root";
30      .password = "*****"
31    };
32    connect@Database(connectionInfo)()
33  }
34
35  main {
36    [
37    checkConnection(message)(retval){
38      println@Console("--- servizio checkConnection ---")();
39      println@Console("Richiesta connessione al server...")();
40      if (message == "check"){
41        println@Console("Richiesta concessa...\n")();
42        retval = "ok"
43      } else {
44        retval = "error";
45        println@Console("Richiesta NON concessa...\n")()
46      }
47    }
48    ] {nullProcess}
49
50    [
51    clockTest(message)(retval){
52      println@Console("--- servizio clockTest ---")();
53      println@Console("Salvataggio risultati nel database...")
          ();
54      if (string(message.count) == "1"){
55        update@Database("insert into clock_test values ('"+
            message.cf+"', '"+message.data+"', "+message.count+"
            )")(queryret)
56      } else {
57        update@Database("update clock_test set count="+message.
            count+"where cf='"+message.cf+"' and data='"message.
            data+"'")(queryret)
58      };
59      if (queryret){
60        retval = "ok";
61        println@Console("Dati salvati correttamente sul
            database...")()
62      } else {
```

```
63        retval = "error";
64        println@Console("Errore durante il salvataggio dei dati
              nel database...")()
65      };
66      directory = "clocktestimg/";
67      image.filename = directory+message.cf+"_"+message.data+"_
            "+message.count+".jpg";
68      image.content = message.content;
69      println@Console("Salvataggio dell'immagine sul server..."
            )();
70      writeFile@File(image)();
71      println@Console("Immagine salvata correttamente sul
            server...\n")()
72    }
73    ] {nullProcess}
74
75    [
76    stimeCognitiveTest(message)(retval){
77      println@Console("--- servizio stimeCognitiveTest ---")();
78      println@Console("Salvataggio risultati nel database...")
            ();
79      update@Database("insert into stime_cognitive_test values
            ('"+message.cf+"', '"+message.data+"', "+message.
            punteggio+", "+message.risp1+", "+message.risp2+", "+
            message.risp3+", "+message.risp4+", "+message.risp5+")
            ")(queryret);
80      if (queryret){
81        retval = "ok";
82        println@Console("Dati salvati correttamente sul
              database...\n")()
83      } else {
84        retval = "error";
85        println@Console("Errore durante il salvataggio dei dati
              sul database...\n")()
86      }
87    }
88    ] {nullProcess}
89
90    [
91    tokenTest(message)(retval){
92      println@Console("--- servizio tokenTest ---")();
93      println@Console("Salvataggio risultati nel database...")
            ();
94      update@Database("insert into token_test values ('"+
            message.cf+"', '"+message.data+"', "+message.punteggio
            +")")(queryret);
95      if (queryret){
96        retval = "ok";
97        println@Console("Dati salvati correttamente sul
```

```
            database...\n")()
98      } else {
99        retval = "error";
100       println@Console("Errore durante il salvataggio dei dati
              sul database...\n")()
101     }
102   }
103   ] {nullProcess}
104
105   [
106   trailMakingTest(message)(retval){
107     println@Console("--- servizio trailMakingTest ---")();
108     println@Console("Salvataggio risultati nel database...")
          ();
109     update@Database("insert into trail_making_test values ('"
          +message.cf+"', '"+message.data+"', '"+message.type+"
          ', "+message.secondi+", "+message.err_numeri+", "+
          message.err_dito+")")(queryret);
110     if (queryret){
111       retval = "ok";
112       println@Console("Dati salvati correttamente sul
            database...")()
113     } else {
114       retval = "error";
115       println@Console("Errore durante il salvataggio dei dati
              nel database...")()
116     };
117     directory = "trailmakingtestimg/";
118     image.filename = directory+message.cf+"_"+message.data+"_
          "+message.type+".jpg";
119     image.content = message.content;
120     println@Console("Salvataggio dell'immagine sul server..."
          )();
121     writeFile@File(image)();
122     println@Console("Immagine salvata correttamente sul
          server...\n")()
123   }
124   ] {nullProcess}
125
126 }
```

Analizzando il codice, salta subito all'occhio l'influenza del linguaggio C sulla sintassi. La keyword *include* permette di linkare dei file in cui sono presenti dei servizi di libreria, in questo caso per il database, l'output su console e la lettura e scrittura di file.

Subito dopo troviamo una riga fondamentale per il funzionamento del server,

che indica la modalità di esecuzione del codice, in questo caso concorrente. Ciò significa che tutti i servizi scritti successivamente a questa riga (i blocchi tra parentesi quadre) sono tutti in attesa in modalità concorrente.
Subito dopo vengono specificati la lista dei servizi tramite l'interfaccia *Server-Interface*, usata nell'inputPort che apre alla ricezione dall'esterno.
Dopodichè il blocco *init* specifica il codice eseguito all'avvio del server. In questo caso è stato usato per la connessione al database. Subito dopo il blocco main, il cui nome è già esaustivo, contiene l'implementazione dei vari servizi, molto simili fra loro. Unici servizi di libreria degni di nota, *update@Database* che permette di effettuare un inserimento nel database (per le query il servizio da usare è *query*) e il comando writeFile@File che salva le immagini in cartelle specifiche del server per i test che ne hanno bisogno.
Da notare come il codice mostri al meglio il concetto esteso di servizio, espanso per qualsiasi funzione. Anche un semplice output su console è un servizio offerto dall'interfaccia *Console* tramite il nome *println*.
Si noti anche come la variabile locale *message* usata come contenitore dei dati in arrivo, per ogni servizio abbia una struttura interna sempre diversa: questo è dovuto alla struttura del pacchetto HTTP ricevuto. La variabile si ″adatta″ alle informazione che riceve. Un altro grande vantaggio di Jolie è la totale assenza di differenze nella scrittura del servizio dovute ai diversi formati dei pacchetti ricevuti. Infatti mentre per i risultati di alcuni test si usa come formato del body l'xml, per altri si ricorre a multipart. Nel codice non vi è traccia di tale differenza.
Comoda anche la notazione usata affinchè i servizi siano perennemente in ascolto. La sintassi

```
1     ...
2     [
3         //codice del servizio
4     ] { nullProcess }
5     ...
```

equivale praticamente a

```
1     ...
2     while(true){
3         //codice del servizio
4     }
5     ...
```

con l'aggiunta dell'esecuzione concorrente tra i vari blocchi.

4.3 Realizzazione della libreria iOS2Jolie

Questa libreria di funzioni è stata sviluppata per la comunicazione tra i test realizzati su iPad e il server Jolie. Essendo stata realizzata su iPad, è scritta in Objective-C, con il supporto del framework Cocoa Touch fornito da Apple. L'interfaccia della libreria contiene una funzione per ogni servizio Jolie mostrato nel paragrafo precedente. Non si è usata un unica funzione perchè le varie prove mandano dati diversi tra loro e quindi una scelta di questo tipo non avrebbe portato nessun vantaggio.
La libreria contiene ovviamente altre funzioni ausiliarie non presenti nell'interfaccia, destinate alla creazione e all'invio di pacchetti HTTP. In questo caso è usata sempre la stessa funzione per tutti i test che inviano solo delle stringhe e che quindi comunicano tramite xml, mentre per i due test che prevedono invio di immagini è stato adottato il formato multipart, ed è stata disegnata una funzione ad-hoc. Questa differenza si riscontra sostanzialmente nel corpo del pacchetto HTTP.
Questa la struttura di un header HTTP dei primi:

```
1 POST /index.html HTTP/1.1
2 Host: www.serveraddress.com
3 Content-type: text/xml
4
5 <nome del servizio>
6         <dato1>qualcosa</dato1>
7         <dato2>qualcos'altro</dato2>
8         ...
9 </nome del servizio>
```

Mentre per i secondi la struttura è leggermente più complicata:

```
1  POST /index.html HTTP/1.1
2  Host: www.serveraddress.com
3  Content-type: multipart/mixed; boundary=0xKhTmLbOuNdArY
4
5  --0xKhTmLbOuNdArY
6  name="dato1"
7  Content-type: text/plain
8
9  qualcosa
10
11 --0xKhTmLbOuNdArY
12 name="dato2"
13 Content-type: text/xml
14
15 <qualcosaltro>dato2</qualcosaltro>
16
17 --0xKhTmLbOuNdArY
18 name="image"
19 Content-type: image/jpeg
20
21 //
22 // image's byte arrays
23 //
24
25 --0xKhTmLbOuNdArY--
```

Il formato multipart permette l'invio di dati non omogenei in pacchetti separati (definiti da un boundary). É in questo modo che dati come delle immagini sono solitamente inviate.

Riguardo al metodo http usato per l'invio delle informazioni, è stato usato sempre il metodo POST.

Capitolo 5

Sviluppi futuri e conclusioni

É arrivato il momento di tirare le somme sul lavoro svolto.
Il progetto è stato molto interessante e stimolante, specie perchè i frutti prodotti sono degli strumenti che potrebbero realmente essere utilizzati e dare un contributo ad attività mirate a migliorare la vita dell'uomo.
Ovviamente anche dal punto di vista informatico è stato molto interessante, sia per quanto riguardava la sfida implementativa mirata a creare una controparte digitale più fedele possibile a quella cartacea dei test, sia per le tecnologie scelte, innovative ma così diverse tra loro.
A seguire un'analisi di come questo progetto potrebbe evolvere e una valutazione del prodotto finale.

5.1 Sviluppi futuri

Di sviluppi possibili ce ne sono davvero tanti, e i lettori più attenti ne avranno già scorti alcuni accennati durante la lettura.
Innanzitutto la prima obbligata evoluzione sarà creare un'applicazione per iPad destinata al dottore, per la fruizione dei risultati, ancora non pronta, ma in fase di progettazione. In realtà si potrebbe pensare a degli strumenti multipiattaforma, visto che sarebbe molto comodo poter usufruire dei risultati anche da una semplice interfaccia web.
Altre espansioni potrebbero essere:

- Implementazioni di altri test neuropsicologici, in modo da aumentare sempre di più l'inserimento di queste tecnologie nell'ambiente
- Implementazione di un sistema di avvisi, destinato al paziente, che sia da guida per i prossimi incontri con il medico e per i prossimi test da effettuare
- Miglioramento della sicurezza nella comunicazione tra iPad e il server, tuttora effettuato tramite http. Una possibile soluzione sarebbe adottare il procollo https
- Miglioramento della gestione delle immagini lato server, attualmente non immagazzinate nel database, a causa dei problemi già spiegati nel paragrafo apposito
- Implementazione dei test su piattaforme diverse. In questo momento è in fase di espansione il mercato dei tablet, anche con sistemi operativi diversi da iOS, tra cui spicca Android. Si potrebbe pensare anche ad un implementazione di questi test come applicazioni web, in modo da poterne poi usufruire su qualsiasi SO. Inoltre in questo modo si eliminerebbe anche la divisione delle applicazioni tuttora presente, in quanto il tutto sarebbe gestito da un server web
- Creazione di ″ambienti privati″, per permettere ad ogni dottore di seguire tutti e soli i propri pazienti, privatamente rispetto agli altri colleghi
- Miglioramento generale dei test già implementati, ad esempio con aggiunte di nuove feature su richiesta del medico

e sicuramente tante altre.

5.2 Conclusioni

In conclusione, l'intero progetto è stato sopra le aspettative. Il feedback di chi probabilmente userà queste applicazioni è stato positivo sin dalle prime

versioni, e la reale utilità è saltata subito agli occhi.
Gli obiettivi prefissati sono stati ampiamente raggiunti, e si è quindi soddisfatti del prodotto finale, che oltre ad essere già perfettamente usabile, ha prodotto un ulteriore stimolo verso le possibili espansioni elencate nel paragrafo precedente.

Appendice

In questa sezione è mostrato l'intero codice della libreria iOS2Jolie e delle applicazioni per iPad. Si è scelto di spostare qui queste informazioni per questione di leggibilità.
La parte B del Trail Making Test è identica, a livello di codice, alla parte A (cambia solo l'interfaccia grafica). É stata perciò omessa.
Inoltre i file qui mostrati sono solo quelli inerenti ai test veri e propri. Non saranno mostrati file utili solo al funzionamento del software.

Sorgente di iOS2Jolie

```
1  //
2  //  iOS2Jolie.h
3  //
4
5  @interface iOS2Jolie : NSObject {
6    //variabili per controlli
7    int clockSended; //conto le volte che mando i risultati del clock test
8    BOOL tmtAsended; //specifico se ho mandato la parte A del tmt
9      NSString *serverUrl; //l'url del server
10 }
11
12 -(BOOL)checkConnection:(BOOL)withAlert;
13 -(void)clockTestServer:(NSString *)cf data:(NSString *)data number:(int)
       number imageData:(NSData *)image;
14 -(void)stimeCognitiveTestServer:(NSString *)cf data:(NSString *)data
       punteggio:(int)punteggio risposte:(float[5])risposte;
15 -(void)tokenTestServer:(NSString *)cf data:(NSString *)data punteggio:(float
       )punteggio;
16 -(void)trailMakingTestServer:(NSString *)cf data:(NSString *)data type:(
       NSString *)type seconds:(int)seconds numberError:(int)numberError
       fingerError:(int)fingerError imageData:(NSData *)image;
17
18 @property int clockSended;
19 @property BOOL tmtAsended;
```

```
20 @property (nonatomic, retain) NSString *serverUrl;
21
22 @end
```

```
1  //
2  //  iOS2Jolie.m
3  //
4
5  #import "iOS2Jolie.h"
6
7  @implementation iOS2Jolie
8
9  @synthesize clockSended;
10 @synthesize tmtAsended;
11 @synthesize serverUrl;
12
13 //funzione di inizializzazione
14 -(id)init{
15     self = [super init];
16     if (self) {
17         /* class-specific initialization goes here */
18         clockSended = 0;
19         tmtAsended = FALSE;
20         serverUrl = [[NSString alloc] initWithString:@"localhost"];
21     }
22     return self;
23 }
24
25 //funzione usata per la connessione al server jolie dei servizi con solo xml
26 -(NSString *)connectToService:(NSString *)service withBody:(NSMutableData *)
       body{
27   //set connessione
28   NSString *urlString = [NSString stringWithFormat:@"http://%@:2001/%@",
         serverUrl,service];
29   NSMutableURLRequest *request = [[[NSMutableURLRequest alloc] init]
         autorelease];
30   [request setURL:[NSURL URLWithString:urlString]];
31   [request setHTTPMethod:@"POST"];
32
33   //set headers
34   [request addValue:@"text/xml" forHTTPHeaderField: @"Content-Type"];
35
36   //create the body
37   NSMutableData *postBody = [NSMutableData data];
38   [postBody appendData:body];
39   [request setHTTPBody:postBody];
40
41   //get and return response
42   NSHTTPURLResponse* urlResponse = nil;
43   NSError *error = [[NSError alloc] init];
44   NSData *responseData = [NSURLConnection sendSynchronousRequest:request
         returningResponse:&urlResponse error:&error];
45   NSString *result = [[NSString alloc] initWithData:responseData encoding:
         NSASCIIStringEncoding];
46   if ([urlResponse statusCode] >= 200 && [urlResponse statusCode] < 300) {
47     return result;
48   } else {
49     NSString *statusCode = [[NSString alloc] initWithFormat:@"%d",[
           urlResponse statusCode]];
50     return statusCode;
```

```
51    }
52  }
53
54  //funzione usata per la connessione al server jolie con multipart per il
        servizio clockTest
55  -(void)clockTestServerConnect:(NSString *)cf data:(NSString *)data number:(
        int)number imageData:(NSData *)image{
56    //set connessione
57      NSString *urlString = [NSString stringWithFormat:@"http://%@:2001/
            clockTest",serverUrl];
58    NSMutableURLRequest *request = [[[NSMutableURLRequest alloc] init]
          autorelease];
59    [request setURL:[NSURL URLWithString:urlString]];
60    [request setHTTPMethod:@"POST"];
61    NSString *boundary = @"0xKhTmLbOuNdArY";
62
63    //set header
64    NSString *contentType = [NSString stringWithFormat:@"multipart/form-data;
          boundary=%@", boundary, nil];
65    [request setValue:contentType forHTTPHeaderField:@"Content-Type"];
66
67    //set body
68    NSMutableData *postBody =[NSMutableData data];
69    [postBody appendData:[[NSString stringWithFormat:@"--%@\r\n", boundary]
          dataUsingEncoding:NSASCIIStringEncoding allowLossyConversion:NO]];
70    [postBody appendData:[[NSString stringWithFormat:@"name=\"cf\"\r\n"]
          dataUsingEncoding:NSASCIIStringEncoding]];
71    [postBody appendData:[[NSString stringWithFormat:@"Content-Type: text/
          plain\r\n\r\n"] dataUsingEncoding:NSASCIIStringEncoding]];
72    [postBody appendData:[[NSString stringWithFormat:@"%@\r\n",cf]
          dataUsingEncoding:NSASCIIStringEncoding]];
73    [postBody appendData:[[NSString stringWithFormat:@"--%@\r\n", boundary]
          dataUsingEncoding:NSASCIIStringEncoding allowLossyConversion:NO]];
74    [postBody appendData:[[NSString stringWithFormat:@"name=\"data\"\r\n"]
          dataUsingEncoding:NSASCIIStringEncoding]];
75    [postBody appendData:[[NSString stringWithFormat:@"Content-Type: text/
          plain\r\n\r\n"] dataUsingEncoding:NSASCIIStringEncoding]];
76    [postBody appendData:[[NSString stringWithFormat:@"%@\r\n",data]
          dataUsingEncoding:NSASCIIStringEncoding]];
77    [postBody appendData:[[NSString stringWithFormat:@"--%@\r\n", boundary]
          dataUsingEncoding:NSASCIIStringEncoding allowLossyConversion:NO]];
78    [postBody appendData:[[NSString stringWithFormat:@"name=\"count\"\r\n"]
          dataUsingEncoding:NSASCIIStringEncoding]];
79    [postBody appendData:[[NSString stringWithFormat:@"Content-Type: text/
          plain\r\n\r\n"] dataUsingEncoding:NSASCIIStringEncoding]];
80    [postBody appendData:[[NSString stringWithFormat:@"%d\r\n",number]
          dataUsingEncoding:NSASCIIStringEncoding]];
81    [postBody appendData:[[NSString stringWithFormat:@"--%@\r\n", boundary]
          dataUsingEncoding:NSASCIIStringEncoding allowLossyConversion:NO]];
82    [postBody appendData:[[NSString stringWithFormat:@"Content-Disposition:
          form-data; name=\"content\"\r\n"] dataUsingEncoding:
          NSASCIIStringEncoding]];
83    [postBody appendData:[[NSString stringWithFormat:@"Content-Type: image/
          jpeg\r\n"] dataUsingEncoding:NSASCIIStringEncoding]];
84    [postBody appendData:[@"Content-Transfer-Encoding: binary\r\n\r\n"
          dataUsingEncoding:NSASCIIStringEncoding]];
85    [postBody appendData:image];
86    [postBody appendData:[[NSString stringWithFormat:@"\r\n--%@--\r\n",
          boundary] dataUsingEncoding:NSASCIIStringEncoding allowLossyConversion
          :NO]];
87    [request setHTTPBody:postBody];
88
```

```
89    [NSURLConnection connectionWithRequest:request delegate:self];
90    //non gestisco la risposta in maniera sincrona, causa tempo.
91    //controllo invece la connessione al server e la risposta asincrona
92  }
93
94  //funzione usata per la connessione al server jolie con multipart per il
        servizio trailMakingTest
95  - (NSString *)trailMakingTestServerConnect:(NSString *)cf data:(NSString *)
      data type:(NSString *)type seconds:(int)seconds numberError:(int)
      numberError fingerError:(int)fingerError imageData:(NSData *)image{
96    //set connessione
97      NSString *urlString = [NSString stringWithFormat:@"http://%@:2001/
            trailMakingTest",serverUrl];
98    NSMutableURLRequest *request = [[[NSMutableURLRequest alloc] init]
          autorelease];
99    [request setURL:[NSURL URLWithString:urlString]];
100   [request setHTTPMethod:@"POST"];
101   NSString *boundary = @"0xKhTmLbOuNdArY";
102
103   //set header
104   NSString *contentType = [NSString stringWithFormat:@"multipart/form-data;
          boundary=%@", boundary, nil];
105   [request setValue:contentType forHTTPHeaderField:@"Content-Type"];
106
107   //set body
108   NSMutableData *postBody =[NSMutableData data];
109   //cf
110   [postBody appendData:[[NSString stringWithFormat:@"--%@\r\n", boundary]
          dataUsingEncoding:NSASCIIStringEncoding allowLossyConversion:NO]];
111   [postBody appendData:[[NSString stringWithFormat:@"name=\"cf\"\r\n"]
          dataUsingEncoding:NSASCIIStringEncoding]];
112   [postBody appendData:[[NSString stringWithFormat:@"Content-Type: text/
          plain\r\n\r\n"] dataUsingEncoding:NSASCIIStringEncoding]];
113   [postBody appendData:[[NSString stringWithFormat:@"%@\r\n",cf]
          dataUsingEncoding:NSASCIIStringEncoding]];
114   //data
115   [postBody appendData:[[NSString stringWithFormat:@"--%@\r\n", boundary]
          dataUsingEncoding:NSASCIIStringEncoding allowLossyConversion:NO]];
116   [postBody appendData:[[NSString stringWithFormat:@"name=\"data\"\r\n"]
          dataUsingEncoding:NSASCIIStringEncoding]];
117   [postBody appendData:[[NSString stringWithFormat:@"Content-Type: text/
          plain\r\n\r\n"] dataUsingEncoding:NSASCIIStringEncoding]];
118   [postBody appendData:[[NSString stringWithFormat:@"%@\r\n",data]
          dataUsingEncoding:NSASCIIStringEncoding]];
119   //tipo del test
120   [postBody appendData:[[NSString stringWithFormat:@"--%@\r\n", boundary]
          dataUsingEncoding:NSASCIIStringEncoding allowLossyConversion:NO]];
121   [postBody appendData:[[NSString stringWithFormat:@"name=\"type\"\r\n"]
          dataUsingEncoding:NSASCIIStringEncoding]];
122   [postBody appendData:[[NSString stringWithFormat:@"Content-Type: text/
          plain\r\n\r\n"] dataUsingEncoding:NSASCIIStringEncoding]];
123   [postBody appendData:[[NSString stringWithFormat:@"%@\r\n",type]
          dataUsingEncoding:NSASCIIStringEncoding]];
124   //secondi impiegati
125   [postBody appendData:[[NSString stringWithFormat:@"--%@\r\n", boundary]
          dataUsingEncoding:NSASCIIStringEncoding allowLossyConversion:NO]];
126   [postBody appendData:[[NSString stringWithFormat:@"name=\"secondi\"\r\n"]
          dataUsingEncoding:NSASCIIStringEncoding]];
127   [postBody appendData:[[NSString stringWithFormat:@"Content-Type: text/
          plain\r\n\r\n"] dataUsingEncoding:NSASCIIStringEncoding]];
128   [postBody appendData:[[NSString stringWithFormat:@"%d\r\n",seconds]
          dataUsingEncoding:NSASCIIStringEncoding]];
```

```
129    //numero di errori numero sbagliato
130    [postBody appendData:[[NSString stringWithFormat:@"--%@\r\n", boundary]
           dataUsingEncoding:NSASCIIStringEncoding allowLossyConversion:NO]];
131    [postBody appendData:[[NSString stringWithFormat:@"name=\"err_numeri\"\r\n
           "] dataUsingEncoding:NSASCIIStringEncoding]];
132    [postBody appendData:[[NSString stringWithFormat:@"Content-Type: text/
           plain\r\n\r\n"] dataUsingEncoding:NSASCIIStringEncoding]];
133    [postBody appendData:[[NSString stringWithFormat:@"%d\r\n",numberError]
           dataUsingEncoding:NSASCIIStringEncoding]];
134    //numero di errori dito alzato
135    [postBody appendData:[[NSString stringWithFormat:@"--%@\r\n", boundary]
           dataUsingEncoding:NSASCIIStringEncoding allowLossyConversion:NO]];
136    [postBody appendData:[[NSString stringWithFormat:@"name=\"err_dito\"\r\n"]
            dataUsingEncoding:NSASCIIStringEncoding]];
137    [postBody appendData:[[NSString stringWithFormat:@"Content-Type: text/
           plain\r\n\r\n"] dataUsingEncoding:NSASCIIStringEncoding]];
138    [postBody appendData:[[NSString stringWithFormat:@"%d\r\n",fingerError]
           dataUsingEncoding:NSASCIIStringEncoding]];
139    //immagine
140    [postBody appendData:[[NSString stringWithFormat:@"--%@\r\n", boundary]
           dataUsingEncoding:NSASCIIStringEncoding allowLossyConversion:NO]];
141    [postBody appendData:[[NSString stringWithFormat:@"Content-Disposition:
           form-data; name=\"content\"\r\n"] dataUsingEncoding:
           NSASCIIStringEncoding]];
142    [postBody appendData:[[NSString stringWithFormat:@"Content-Type: image/
           jpeg\r\n"] dataUsingEncoding:NSASCIIStringEncoding]];
143    [postBody appendData:[@"Content-Transfer-Encoding: binary\r\n\r\n"
           dataUsingEncoding:NSASCIIStringEncoding]];
144    [postBody appendData:image];
145    [postBody appendData:[[NSString stringWithFormat:@"\r\n--%@--\r\n",
           boundary] dataUsingEncoding:NSASCIIStringEncoding allowLossyConversion
           :NO]];
146    [request setHTTPBody:postBody];
147
148    //get and return response
149    NSHTTPURLResponse* urlResponse = nil;
150    NSError *error = [[NSError alloc] init];
151    NSData *responseData = [NSURLConnection sendSynchronousRequest:request
           returningResponse:&urlResponse error:&error];
152    NSString *result = [[NSString alloc] initWithData:responseData encoding:
           NSASCIIStringEncoding];
153    if ([urlResponse statusCode] >= 200 && [urlResponse statusCode] < 300) {
154      return result;
155    } else {
156      NSString *statusCode = [[NSString alloc] initWithFormat:@"%d",[
             urlResponse statusCode]];
157      return statusCode;
158    }
159  }
160
161  //testa la connessione al server
162  - (BOOL)checkConnection:(BOOL)withAlert{
163    NSMutableData *body = [NSMutableData data];
164    [body appendData:[[NSString stringWithString:@"<checkConnection>check</
           checkConnection>"] dataUsingEncoding:NSASCIIStringEncoding]];
165    NSString *result = [self connectToService:@"checkConnection" withBody:body
           ];
166    //creo l'alert
167    UIAlertView *alert = [[UIAlertView alloc] init];
168    [alert setTitle:@"Connessione"];
169    [alert setDelegate:self];
170    [alert addButtonWithTitle:@"Ok"];
```

```
171     if ([result rangeOfString:@"<checkConnectionResponse>ok</
            checkConnectionResponse>"].location != NSNotFound) {
172       if (withAlert) {
173         [alert setMessage:@"Connessione al server presente!"];
174         [alert show];
175       }
176       [alert release];
177       return TRUE;
178     } else {
179       if (withAlert) {
180         [alert setMessage:@"Server non raggiungibile. Controlla la tua
                connessione!"];
181         [alert show];
182       }
183       [alert release];
184       return FALSE;
185     }
186   }
187
188   //funzione per l'invio dei dati del clock test
189   - (void)clockTestServer:(NSString *)cf data:(NSString *)data number:(int)
        number imageData:(NSData *)image{
190     BOOL check = [self checkConnection:FALSE];
191     //se esiste la connessione al server procedo con l'invio dei dati
192     if (check) {
193       clockSended++;
194       //connessione
195       [self clockTestServerConnect:cf data:data number:number imageData:image
            ];
196     }
197   }
198
199   //funzione per l'invio dei dati del stime cognitive test
200   - (void)stimeCognitiveTestServer:(NSString *)cf data:(NSString *)data
        punteggio:(int)punteggio risposte:(float[5])risposte{
201     UIAlertView *alert = [[UIAlertView alloc] init];
202     [alert setDelegate:self];
203     [alert addButtonWithTitle:@"Ok"];
204     [alert setTitle:@"Stime Cognitive Test"];
205     BOOL check = [self checkConnection:FALSE];
206     //se esiste la connessione al server procedo con l'invio dei dati
207     if (check) {
208       //creo il body
209       NSMutableData *body = [NSMutableData data];
210       [body appendData:[[NSString stringWithFormat:@"<stimeCognitiveTest>"]
              dataUsingEncoding:NSASCIIStringEncoding]];
211       [body appendData:[[NSString stringWithFormat:@"<cf>%@</cf>",cf]
              dataUsingEncoding:NSASCIIStringEncoding]];
212       [body appendData:[[NSString stringWithFormat:@"<data>%@</data>",data]
              dataUsingEncoding:NSASCIIStringEncoding]];
213       [body appendData:[[NSString stringWithFormat:@"<punteggio>%d</punteggio>
              ",punteggio] dataUsingEncoding:NSASCIIStringEncoding]];
214       [body appendData:[[NSString stringWithFormat:@"<risp1>%f</risp1>",
              risposte[0]] dataUsingEncoding:NSASCIIStringEncoding]];
215       [body appendData:[[NSString stringWithFormat:@"<risp2>%f</risp2>",
              risposte[1]] dataUsingEncoding:NSASCIIStringEncoding]];
216       [body appendData:[[NSString stringWithFormat:@"<risp3>%f</risp3>",
              risposte[2]] dataUsingEncoding:NSASCIIStringEncoding]];
217       [body appendData:[[NSString stringWithFormat:@"<risp4>%f</risp4>",
              risposte[3]] dataUsingEncoding:NSASCIIStringEncoding]];
218       [body appendData:[[NSString stringWithFormat:@"<risp5>%f</risp5>",
              risposte[4]] dataUsingEncoding:NSASCIIStringEncoding]];
```

```
219     [body appendData:[[NSString stringWithFormat:@"</stimeCognitiveTest>"]
            dataUsingEncoding:NSASCIIStringEncoding]];
220     //connessione
221     NSString *result = [self connectToService:@"stimeCognitiveTest" withBody
            :body];
222     if ([result rangeOfString:@"<stimeCognitiveTestResponse>ok</
            stimeCognitiveTestResponse>"].location != NSNotFound) {
223       [alert setMessage:@"Dati salvati sul server!"];
224     } else {
225       [alert setMessage:@"Errore durante il salvataggio dei dati!"];
226     }
227   } else {
228     [alert setMessage:@"Server non raggiungibile. Controlla la tua
            connessione!"];
229   }
230   [alert show];
231     [alert release];
232 }
233
234 //funzione per l'invio dei dati del token test
235 - (void)tokenTestServer:(NSString *)cf data:(NSString *)data punteggio:(
      float)punteggio{
236   UIAlertView *alert = [[UIAlertView alloc] init];
237   [alert setDelegate:self];
238   [alert addButtonWithTitle:@"Ok"];
239   [alert setTitle:@"Token Test"];
240   BOOL check = [self checkConnection:FALSE];
241   //se esiste la connessione al server procedo con l'invio dei dati
242   if (check) {
243     //creo il body
244     NSMutableData *body = [NSMutableData data];
245     [body appendData:[[NSString stringWithFormat:@"<tokenTest>"]
            dataUsingEncoding:NSASCIIStringEncoding]];
246     [body appendData:[[NSString stringWithFormat:@"<cf>%@</cf>",cf]
            dataUsingEncoding:NSASCIIStringEncoding]];
247     [body appendData:[[NSString stringWithFormat:@"<data>%@</data>",data]
            dataUsingEncoding:NSASCIIStringEncoding]];
248     [body appendData:[[NSString stringWithFormat:@"<punteggio>%f</punteggio>
            ",punteggio] dataUsingEncoding:NSASCIIStringEncoding]];
249     [body appendData:[[NSString stringWithFormat:@"</tokenTest>"]
            dataUsingEncoding:NSASCIIStringEncoding]];
250     //connessione
251     NSString *result = [self connectToService:@"tokenTest" withBody:body];
252     if ([result rangeOfString:@"<tokenTestResponse>ok</tokenTestResponse>"].
            location != NSNotFound) {
253       [alert setMessage:@"Dati salvati sul server!"];
254     } else {
255       [alert setMessage:@"Errore durante il salvataggio dei dati!"];
256     }
257   } else {
258     [alert setMessage:@"Server non raggiungibile. Controlla la tua
            connessione!"];
259   }
260   [alert show];
261     [alert release];
262 }
263
264 //funzione per l'invio dei dati del trail making test
265 - (void)trailMakingTestServer:(NSString *)cf data:(NSString *)data type:(
      NSString *)type seconds:(int)seconds numberError:(int)numberError
      fingerError:(int)fingerError imageData:(NSData *)image{
266     //se siamo al test A non mostro alert
```

```
267     BOOL showAlert;
268     if ([type isEqualToString:@"A"]) {
269         showAlert = FALSE;
270     } else {
271         showAlert = TRUE;
272     }
273   UIAlertView *alert = [[UIAlertView alloc] init];
274   [alert setDelegate:self];
275   [alert addButtonWithTitle:@"Ok"];
276   [alert setTitle:@"Trail Making Test"];
277   BOOL check = [self checkConnection:FALSE];
278   //se esiste la connessione al server procedo con l'invio dei dati
279   if (check) {
280     //connessione
281     NSString *result = [self trailMakingTestServerConnect:cf data:data type:
          type seconds:seconds numberError:numberError fingerError:fingerError
           imageData:image];
282         if ([type isEqualToString:@"A"] && [result rangeOfString:@"<
              trailMakingTestResponse>ok</trailMakingTestResponse>"].location
              != NSNotFound) {
283             tmtAsended = TRUE;
284         }
285         if ([type isEqualToString:@"B"]) {
286             if ([result rangeOfString:@"<trailMakingTestResponse>ok</
                  trailMakingTestResponse>"].location != NSNotFound) {
287                 [alert setMessage:@"Dati salvati sul server!"];
288             } else {
289                 [alert setMessage:@"Errore durante il salvataggio dei dati!"
                        ];
290             }
291         }
292   } else {
293     [alert setMessage:@"Server non raggiungibile. Controlla la tua
          connessione!"];
294   }
295     if (showAlert) {
296         [alert show];
297     }
298     [alert release];
299 }
300
301 @end
```

Sorgente del Clock Test

```
1  //
2  //  Clock_TestViewController.h
3  //
4
5  #import <UIKit/UIKit.h>
6  #import "iOS2Jolie.h"
7
8  @class ClockTest;
9
10 @interface Clock_TestViewController : UIViewController {
11   ClockTest *clockTest;
```

```
12    UITextField *username;
13    UITextField *password;
14      UIAlertView *alert;
15    iOS2Jolie *ios2jolie;
16      BOOL isExample;
17      NSString *todayDate;
18      NSString *serverUrl;
19  }
20
21  @property (nonatomic, retain) IBOutlet ClockTest *clockTest;
22  @property (nonatomic, retain) IBOutlet UITextField *username;
23  @property (nonatomic, retain) IBOutlet UITextField *password;
24  @property BOOL isExample;
25  @property (nonatomic, retain) NSString *todayDate;
26  @property (nonatomic, retain) NSString *serverUrl;
27
28  - (IBAction) launchHomeTest:(id)sender;
29
30  @end
```

```
1   //
2   //  Clock_TestViewController.m
3   //
4
5   #import "Clock_TestViewController.h"
6   #import "ClockTest.h"
7   #import <QuartzCore/QuartzCore.h>
8
9   @implementation Clock_TestViewController
10
11  @synthesize clockTest;
12  @synthesize username;
13  @synthesize password;
14  @synthesize isExample;
15  @synthesize todayDate;
16  @synthesize serverUrl;
17
18  // Implement viewDidLoad to do additional setup after loading the view,
        typically from a nib.
19  - (void)viewDidLoad {
20      //SFONDO GRADIENTE
21      CGColorSpaceRef rgb = CGColorSpaceCreateDeviceRGB();
22      CGFloat comps1[] = {0.0, 0.0, 0.0, 1.0};
23      CGFloat comps2[] = {0.0, 0.0, 0.18, 1.0};
24      CGColorRef color1 = CGColorCreate(rgb, comps1);
25      CGColorRef color2 = CGColorCreate(rgb, comps2);
26      CGColorSpaceRelease(rgb);
27      CAGradientLayer *gradient = [CAGradientLayer layer];
28      gradient.frame = self.view.bounds;
29      gradient.colors = [NSArray arrayWithObjects:(id)color1,(id)color2,nil];
30      [self.view.layer insertSublayer:gradient atIndex:0];
31      [super viewDidLoad];
32      //setto l'alert in caso di connessione assente
33      alert = [[UIAlertView alloc] init];
34      [alert setTitle:@"Clock Test"];
35      [alert setDelegate:self];
36      [alert setMessage:@"Server non raggiungibile. Vuoi continuare il test
            senza l'invio dei risultati?"];
37      [alert addButtonWithTitle:@"Ok"];
38      [alert addButtonWithTitle:@"No"];
```

```
39      //alloco la libreria ios2jolie e setto la data
40      ios2jolie = [[iOS2Jolie alloc] init];
41      NSDateFormatter *today = [[NSDateFormatter alloc] init];
42      [today setDateFormat:@"yyyy-MM-dd HH:mm:ss"];
43      todayDate = [[NSString alloc] initWithString:[today stringFromDate:[
            NSDate date]]];
44      serverUrl = @"localhost";
45  }
46
47  //funzione che porta al menu dei test dopo aver inserito i dati utente
48  - (IBAction) launchHomeTest:(id)sender{
49    if (username.text.length > 0 && password.text.length > 0) {
50      [username resignFirstResponder];
51      [password resignFirstResponder];
52          ////ESCAPE PER IL SETTING SEGRETO DELL'URL DEL SERVER
53          if ([username.text isEqualToString:@"serverurl"]) {
54              ios2jolie.serverUrl = password.text;
55              serverUrl = password.text;
56              return;
57          }
58          ////FINE ESCAPE
59          if ([ios2jolie checkConnection:FALSE]) {
60              //TODO: check nel database dell'utente
61              [ios2jolie release];
62              isExample = FALSE;
63              clockTest = [[ClockTest alloc] initWithNibName:@"ClockTest"
                    bundle:nil];
64              clockTest.parent = self;
65              [self.view setUserInteractionEnabled:FALSE];
66              [self.view addSubview:clockTest.view];
67          } else {
68              [alert show];
69          }
70    }
71  }
72
73  //chiamata al click sull'alert
74  - (void)alertView:(UIAlertView *)alertView clickedButtonAtIndex:(NSInteger)
        buttonIndex{
75    if (buttonIndex == 0){
76          [ios2jolie release];
77          isExample = TRUE;
78      clockTest = [[ClockTest alloc] initWithNibName:@"ClockTest" bundle:nil];
79          clockTest.parent = self;
80          [self.view setUserInteractionEnabled:FALSE];
81          [self.view addSubview:clockTest.view];
82    }
83  }
84
85  // Override to allow orientations other than the default portrait
        orientation.
86  - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)
        interfaceOrientation {
87      return NO;
88  }
89
90  - (void)dealloc {
91    [super dealloc];
92  }
93
94  @end
```

```
1  //
2  //  ClockTest.h
3  //
4
5  #import <UIKit/UIKit.h>
6  #import "Clock_TestViewController.h"
7  #import <AVFoundation/AVFoundation.h>
8  #import "iOS2Jolie.h"
9
10 @interface ClockTest : UIViewController <AVAudioPlayerDelegate>{
11   UIImageView *drawImageClock;
12   UIView *fine;
13   Clock_TestViewController *parent;
14   AVAudioPlayer *audioPlayer;
15   NSURL *url;
16   NSError *error;
17   iOS2Jolie *ios2jolie;
18 }
19
20 @property (nonatomic, retain) UIImageView *drawImageClock;
21 @property (nonatomic, retain) IBOutlet UIView *fine;
22 @property (nonatomic, retain) Clock_TestViewController *parent;
23 @property (nonatomic, retain) AVAudioPlayer *audioPlayer;
24 @property (nonatomic, retain) NSURL *url;
25 @property (nonatomic, retain) NSError *error;
26
27 - (IBAction)richiestaTerminazione:(id)sender;
28
29 @end
```

```
1  //
2  //  ClockTest.m
3  //
4
5  #import "ClockTest.h"
6  #import <QuartzCore/QuartzCore.h>
7  #include <stdlib.h>
8
9  @implementation ClockTest
10
11 @synthesize drawImageClock;
12 @synthesize fine;
13 @synthesize parent;
14 @synthesize audioPlayer,url,error;
15
16 CGPoint lastPoint;
17 NSTimer *clockTimer;
18 UIAlertView *alert;
19 int seconds = 0;
20 int imageNumber = 0;
21 BOOL terminato = FALSE;
22
23 // Implement viewDidLoad to do additional setup after loading the view,
       typically from a nib.
24 - (void)viewDidLoad {
25   [super viewDidLoad];
26   //creo la view in cui disegno
27   drawImageClock = [[UIImageView alloc] initWithFrame:CGRectMake(0, 0, 768,
         1024)];
28   //funzioni di disegno
```

```
29    //definisce il context in cui disegnare
30    UIGraphicsBeginImageContext(drawImageClock.frame.size);
31    //delimita l'area in cui disegnare
32    [drawImageClock.image drawInRect:CGRectMake(0, 0, drawImageClock.frame.
          size.width, drawImageClock.frame.size.height)];
33    CGContextRef context = UIGraphicsGetCurrentContext();
34    CGContextSetLineWidth(context, 2.0);
35    CGContextSetLineCap(context, kCGLineCapRound);
36    //Crea un cerchio (il quadrante dell'orologio)
37    CGContextAddEllipseInRect(context, CGRectMake(59, 130, 650, 650));
38    CGContextDrawPath(context, kCGPathStroke);
39    //Crea l'appendino dell'orologio
40    CGContextBeginPath(context);
41    CGContextMoveToPoint(context, 384, 89);
42    CGContextAddLineToPoint(context, 359, 127);
43    CGContextAddLineToPoint(context, 409, 127);
44    CGContextAddLineToPoint(context, 384, 89);
45    CGContextStrokePath(context);
46    //prende la view prima di disegnarci sopra
47    drawImageClock.image = UIGraphicsGetImageFromCurrentImageContext();
48    //disegna le nuove linee
49    UIGraphicsEndImageContext();
50      //alloco i servizi per l'invio dei dati
51      ios2jolie = [[iOS2Jolie alloc] init];
52      ios2jolie.serverUrl = parent.serverUrl;
53    //creo l'alert di conferma in caso di pressione del bottone
54    alert = [[UIAlertView alloc] init];
55    [alert setTitle:@"Conferma"];
56    [alert setMessage:@"Vuoi davvero terminare il test?"];
57    [alert setDelegate:self];
58    [alert addButtonWithTitle:@"Si"];
59    [alert addButtonWithTitle:@"No"];
60    //mostra la vista in cui disegno
61    [self.view setUserInteractionEnabled:FALSE];
62    [self.view addSubview:drawImageClock];
63    //allocazione url e play dell'intro
64    url = [NSURL fileURLWithPath:[NSString stringWithFormat:@"%@/intro.mp3",[[
          NSBundle mainBundle] resourcePath]]];
65    audioPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:url error:&
          error];
66    audioPlayer.numberOfLoops = 0;
67    audioPlayer.delegate = self;
68    NSLog(@"Play intro");
69    [audioPlayer play];
70  }
71
72  //richiamata alla fine dell'audio di introduzione
73  -(void)audioPlayerDidFinishPlaying:(AVAudioPlayer *)player successfully:(
        BOOL)flag{
74    clockTimer = [NSTimer scheduledTimerWithTimeInterval:300.0 target:self
          selector:@selector(testTerminato) userInfo:NULL repeats:NO];
75    [self.view setUserInteractionEnabled:TRUE];
76    [parent.view setUserInteractionEnabled:TRUE];
77  }
78
79  //salva la vista dell'orologio in un'immagine
80  - (void)saveScreenShot{
81    //salvo nelle immagini
82    UIGraphicsBeginImageContext(drawImageClock.frame.size);
83    [drawImageClock.layer renderInContext:UIGraphicsGetCurrentContext()];
84    UIImage *screenShot = UIGraphicsGetImageFromCurrentImageContext();
85    UIGraphicsEndImageContext();
```

```
86    //aggiungiamo la label sulla image
87    CGColorSpaceRef colorSpace = CGColorSpaceCreateDeviceRGB();
88      CGContextRef context = CGBitmapContextCreate(NULL, screenShot.size.width
          , screenShot.size.height, 8, 4 * screenShot.size.width, colorSpace,
          kCGImageAlphaPremultipliedFirst);
89      CGContextDrawImage(context, CGRectMake(0, 0, screenShot.size.width,
          screenShot.size.height), screenShot.CGImage);
90      //CGContextSetRGBFillColor(context, 0.0, 0.0, 1.0, 1);
91      CGContextSelectFont(context, "Times New Roman", 30, kCGEncodingMacRoman)
          ;
92      CGContextSetTextDrawingMode(context, kCGTextFill);
93      CGContextSetRGBFillColor(context, 0, 0, 0, 1);
94    imageNumber++;
95    char numImage[40];
96    char nomecognome[50];
97    NSDateFormatter *today = [[NSDateFormatter alloc] init];
98    [today setDateFormat:@"dd/MM/yyyy"];
99    sprintf(nomecognome, "%s" ,[parent.username.text UTF8String]);
100   sprintf(numImage,"data: %s - num: %d",[[today stringFromDate:[NSDate date
        ]] UTF8String],imageNumber);
101   CGContextShowTextAtPoint(context, 20, 50, nomecognome, strlen(nomecognome)
        );
102   CGContextShowTextAtPoint(context, 20, 20, numImage, strlen(numImage));
103     CGImageRef imageMasked = CGBitmapContextCreateImage(context);
104     CGContextRelease(context);
105     CGColorSpaceRelease(colorSpace);
106   //salviamo nell'album o sul server
107     if (parent.isExample) {
108         UIImageWriteToSavedPhotosAlbum([UIImage imageWithCGImage:imageMasked
              ], self, nil, nil);
109     } else {
110         [ios2jolie clockTestServer:parent.username.text data:parent.
              todayDate number:imageNumber imageData:UIImageJPEGRepresentation
              ([UIImage imageWithCGImage:imageMasked], 0.85)];
111     }
112 }
113
114 //funzione chiamata per finire il test
115 - (void)testTerminato{
116   //salvo l'immagine con l'ultima vista
117   [self saveScreenShot];
118   //cambio vista
119   self.view = self.fine;
120   [parent.view setUserInteractionEnabled:FALSE];
121   [self.view setUserInteractionEnabled:FALSE];
122   terminato = TRUE;
123     if (!parent.isExample) {
124         UIAlertView *endAlert = [[UIAlertView alloc] init];
125         [endAlert setTitle:@"Clock Test"];
126         [endAlert setDelegate:nil];
127         [endAlert addButtonWithTitle:@"Fine"];
128         //controllo che i dati siano stati inviati correttamente
129         if (imageNumber == [ios2jolie clockSended]) {
130             [endAlert setMessage:@"Dati inviati correttamente!"];
131             [endAlert show];
132         } else {
133             [endAlert setMessage:@"Alcuni dati potrebbero non essere stati
                  inviati"];
134             [endAlert show];
135         }
136     }
137 }
```

```
138
139 //funzione chiamata al termine del test
140 - (IBAction)richiestaTerminazione:(id)sender{
141   //invalido il clock se attivo
142   if ([clockTimer isValid]) {
143     //creo due clock per determinare i secondi rimanenti del timer
144     NSDate *clockDate = [clockTimer fireDate];
145     //invalido il timer
146     [clockTimer invalidate];
147     NSDate *currentDate = [NSDate date];
148     NSCalendar *calendar = [[[NSCalendar alloc] initWithCalendarIdentifier:
          NSGregorianCalendar] autorelease];
149     NSDateComponents *components = [calendar components:NSSecondCalendarUnit
           fromDate:currentDate toDate:clockDate options:0];
150     seconds = components.second;
151     //mostro l'alert
152     [alert show];
153   }
154 }
155
156 //chiamata al click sull'alert
157 - (void)alertView:(UIAlertView *)alertView clickedButtonAtIndex:(NSInteger)
      buttonIndex{
158   if (buttonIndex == 0){
159     [self testTerminato];
160   } else if (buttonIndex == 1) {
161     clockTimer = [NSTimer scheduledTimerWithTimeInterval:seconds target:self
           selector:@selector(testTerminato) userInfo:NULL repeats:NO];
162   }
163 }
164
165 - (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event {
166
167   UITouch *touch = [touches anyObject];
168   lastPoint = [touch locationInView:drawImageClock];
169 }
170
171 - (void)touchesMoved:(NSSet *)touches withEvent:(UIEvent *)event {
172
173   UITouch *touch = [touches anyObject];
174   CGPoint currentPoint = [touch locationInView:drawImageClock];
175
176   //disegno del tratto
177   UIGraphicsBeginImageContext(drawImageClock.frame.size);
178   [drawImageClock.image drawInRect:CGRectMake(0, 0, drawImageClock.frame.
        size.width, drawImageClock.frame.size.height)];
179   CGContextRef context = UIGraphicsGetCurrentContext();
180   CGContextSetLineCap(context, kCGLineCapRound);
181   CGContextSetLineWidth(context, 3.0);
182   CGContextSetRGBStrokeColor(context, 0.0, 0.15, 0.58, 1.0);
183   CGContextBeginPath(context);
184   CGContextMoveToPoint(context, lastPoint.x, lastPoint.y);
185   CGContextAddLineToPoint(context, currentPoint.x, currentPoint.y);
186   CGContextStrokePath(context);
187   drawImageClock.image = UIGraphicsGetImageFromCurrentImageContext();
188   UIGraphicsEndImageContext();
189
190   lastPoint = currentPoint;
191 }
192
193 - (void)touchesEnded:(NSSet *)touches withEvent:(UIEvent *)event {
194   //disegno del tratto
```

```
195   UIGraphicsBeginImageContext(drawImageClock.frame.size);
196   [drawImageClock.image drawInRect:CGRectMake(0, 0, drawImageClock.frame.
          size.width, drawImageClock.frame.size.height)];
197   CGContextRef context = UIGraphicsGetCurrentContext();
198   CGContextSetLineCap(context, kCGLineCapRound);
199   CGContextSetLineWidth(context, 3.0);
200   CGContextSetRGBStrokeColor(context, 0.0, 0.15, 0.58, 1.0);
201   CGContextMoveToPoint(context, lastPoint.x, lastPoint.y);
202   CGContextAddLineToPoint(context, lastPoint.x, lastPoint.y);
203   CGContextStrokePath(context);
204   CGContextFlush(context);
205   drawImageClock.image = UIGraphicsGetImageFromCurrentImageContext();
206   UIGraphicsEndImageContext();
207   //salvo un'immagine con le modifiche nella vista
208   if (!terminato) {
209     [self saveScreenShot];
210   }
211 }
212
213 - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)
      interfaceOrientation {
214     // Overriden to allow any orientation.
215     return NO;
216 }
217
218 - (void)dealloc {
219   [alert release];
220   [drawImageClock release];
221   [fine release];
222   [parent release];
223   [audioPlayer release];
224   [url release];
225   [error release];
226   [ios2jolie release];
227   [super dealloc];
228 }
229
230 @end
```

Sorgente dello Stime Cognitive Test

```
1  //
2  //  Stime_CognitiveViewController.h
3  //
4
5  #import <UIKit/UIKit.h>
6  #import "iOS2Jolie.h"
7
8  @class StimeCognitiveTest;
9
10 @interface Stime_CognitiveViewController : UIViewController {
11   StimeCognitiveTest *stimeCognitive;
12   UITextField *username;
13   UITextField *password;
14     UIAlertView *alert;
15   iOS2Jolie *ios2jolie;
```

```
16      BOOL isExample;
17      NSString *todayDate;
18      NSString *serverUrl;
19  }
20
21  @property (nonatomic, retain) IBOutlet StimeCognitiveTest *stimeCognitive;
22  @property (nonatomic, retain) IBOutlet UITextField *username;
23  @property (nonatomic, retain) IBOutlet UITextField *password;
24  @property BOOL isExample;
25  @property (nonatomic, retain) NSString *todayDate;
26  @property (nonatomic, retain) NSString *serverUrl;
27
28  - (IBAction) launchHomeTest:(id)sender;
29
30  @end
```

```
1   //
2   //  Stime_CognitiveViewController.m
3   //
4
5   #import "Stime_CognitiveViewController.h"
6   #import "StimeCognitiveTest.h"
7   #import <QuartzCore/QuartzCore.h>
8
9   @implementation Stime_CognitiveViewController
10
11  @synthesize stimeCognitive;
12  @synthesize username;
13  @synthesize password;
14  @synthesize isExample;
15  @synthesize todayDate;
16  @synthesize serverUrl;
17
18  // Implement viewDidLoad to do additional setup after loading the view,
        typically from a nib.
19  - (void)viewDidLoad {
20      //SFONDO GRADIENTE
21      CGColorSpaceRef rgb = CGColorSpaceCreateDeviceRGB();
22      CGFloat comps1[] = {0.0, 0.0, 0.0, 1.0};
23      CGFloat comps2[] = {0.0, 0.0, 0.18, 1.0};
24      CGColorRef color1 = CGColorCreate(rgb, comps1);
25      CGColorRef color2 = CGColorCreate(rgb, comps2);
26      CGColorSpaceRelease(rgb);
27      CAGradientLayer *gradient = [CAGradientLayer layer];
28      gradient.frame = self.view.bounds;
29      gradient.colors = [NSArray arrayWithObjects:(id)color1,(id)color2,nil];
30      [self.view.layer insertSublayer:gradient atIndex:0];
31      [super viewDidLoad];
32      //setto l'alert in caso di connessione assente
33      alert = [[UIAlertView alloc] init];
34      [alert setTitle:@"Stime Cognitive Test"];
35      [alert setDelegate:self];
36      [alert setMessage:@"Server non raggiungibile. Vuoi continuare il test
            senza l'invio dei risultati?"];
37      [alert addButtonWithTitle:@"Ok"];
38      [alert addButtonWithTitle:@"No"];
39      //alloco la libreria ios2jolie e setto la data
40      ios2jolie = [[iOS2Jolie alloc] init];
41      NSDateFormatter *today = [[NSDateFormatter alloc] init];
42      [today setDateFormat:@"yyyy-MM-dd HH:mm:ss"];
```

```
43      todayDate = [[NSString alloc] initWithString:[today stringFromDate:[
            NSDate date]]];
44      serverUrl = @"localhost";
45  }
46
47  //funzione che porta al menu dei test dopo aver inserito i dati utente
48  - (IBAction) launchHomeTest:(id)sender{
49      if (username.text.length > 0 && password.text.length > 0) {
50      [username resignFirstResponder];
51      [password resignFirstResponder];
52          ////ESCAPE PER IL SETTING SEGRETO DELL'URL DEL SERVER
53          if ([username.text isEqualToString:@"serverurl"]) {
54              ios2jolie.serverUrl = password.text;
55              serverUrl = password.text;
56              return;
57          }
58          ////FINE ESCAPE
59          if ([ios2jolie checkConnection:FALSE]) {
60              //TODO: check nel database dell'utente
61              [ios2jolie release];
62              isExample = FALSE;
63              stimeCognitive = [[StimeCognitiveTest alloc] initWithNibName:@"
                    StimeCognitiveTest" bundle:nil];
64              stimeCognitive.parent = self;
65              [self.view setUserInteractionEnabled:FALSE];
66              [self.view addSubview:stimeCognitive.view];
67          } else {
68              [alert show];
69          }
70    }
71  }
72
73  //chiamata al click sull'alert
74  - (void)alertView:(UIAlertView *)alertView clickedButtonAtIndex:(NSInteger)
        buttonIndex{
75    if (buttonIndex == 0){
76          [ios2jolie release];
77          isExample = TRUE;
78      stimeCognitive = [[StimeCognitiveTest alloc] initWithNibName:@"
            StimeCognitiveTest" bundle:nil];
79          stimeCognitive.parent = self;
80          [self.view setUserInteractionEnabled:FALSE];
81          [self.view addSubview:stimeCognitive.view];
82    }
83  }
84
85  // Override to allow orientations other than the default portrait
        orientation.
86  - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)
        interfaceOrientation {
87      return NO;
88  }
89
90  - (void)dealloc {
91    [super dealloc];
92  }
93
94  @end
```

```
1  //
```

```
2  //  StimeCognitiveTest.h
3  //
4
5  #import <UIKit/UIKit.h>
6  #import <AVFoundation/AVFoundation.h>
7  #import "Stime_CognitiveViewController.h"
8  #import "iOS2Jolie.h"
9
10 @interface StimeCognitiveTest : UIViewController <AVAudioPlayerDelegate>{
11   Stime_CognitiveViewController *parent;
12   AVAudioPlayer *audioPlayer;
13   UILabel *stimeLabel;
14   UITextField *stimeText;
15   UIView *fine;
16     iOS2Jolie *ios2jolie;
17 }
18
19 @property (nonatomic, retain) Stime_CognitiveViewController *parent;
20 @property (nonatomic, retain) IBOutlet UILabel *stimeLabel;
21 @property (nonatomic, retain) IBOutlet UITextField *stimeText;
22 @property (nonatomic, retain) IBOutlet UIView *fine;
23 @property (nonatomic, retain) AVAudioPlayer *audioPlayer;
24
25 - (IBAction) tastierino:(UIButton *)sender;
26
27 @end
```

```
1  //
2  //  StimeCognitiveTest.m
3  //
4
5  #import "StimeCognitiveTest.h"
6
7  @implementation StimeCognitiveTest
8
9  @synthesize stimeLabel;
10 @synthesize stimeText;
11 @synthesize fine;
12 @synthesize parent;
13 @synthesize audioPlayer;
14
15 BOOL virgola = FALSE; //bool che indica se cliccata la virgola
16 int indexDomanda = 0; //domanda corrente
17 NSArray *domande;  //array con le domande da presentare
18 float risposteSaved[] = {-1,-1,-1,-1,-1};
19 int risposte = 0;  //numero di risposte esatte
20 NSURL *url;
21 NSError *error;
22 NSTimer *globalTimer;
23
24 // Implement viewDidLoad to do additional setup after loading the view,
       typically from a nib.
25 - (void)viewDidLoad {
26   //creo array domande
27   domande = [[NSArray alloc] initWithObjects:
28       @"1. Quanto costa un litro di latte fresco?",
29       @"2. Quanto dista Milano da Roma?",
30       @"3. Quanto e' lunga una chitarra?",
31       @"4. Quanto dura una messa?",
32       @"5. In Olanda, quanti canguri ci sono?",
```

```
33        nil];
34      [super viewDidLoad];
35    //disabilito il tocco
36    [self.view setUserInteractionEnabled:FALSE];
37      //alloco i servizi per l'invio dei dati
38      ios2jolie = [[iOS2Jolie alloc] init];
39      ios2jolie.serverUrl = parent.serverUrl;
40    //play della spiegazione introduttiva
41    url = [NSURL fileURLWithPath:[NSString stringWithFormat:@"%@/intro.mp3",
          [[NSBundle mainBundle] resourcePath]]];
42    audioPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:url error:&
          error];
43    audioPlayer.numberOfLoops=0;
44    audioPlayer.delegate=self;
45    NSLog(@"Play intro");
46    [audioPlayer play];
47  }
48
49  //funzione chiamata alla fine di ogni audio (per noi l'intro)
50  -(void)audioPlayerDidFinishPlaying: (AVAudioPlayer *)player successfully: (
        BOOL)flag{
51    //setto la label della domanda
52    stimeLabel.text=[domande objectAtIndex:indexDomanda];
53    //creo timer globale
54    globalTimer = [NSTimer scheduledTimerWithTimeInterval:300.0 target:self
          selector:@selector(globalEnd) userInfo:NULL repeats:NO];
55    //riabilito il tocco
56    [self.view setUserInteractionEnabled:TRUE];
57    [parent.view setUserInteractionEnabled:TRUE];
58  }
59
60  //funzione chiamata al termine del timer globale
61  - (void) globalEnd{
62    //non ha finito il test nei tempi oppure test finito
63    self.view = self.fine;
64    [parent.view setUserInteractionEnabled:FALSE];
65    [self.view setUserInteractionEnabled:FALSE];
66      if (!parent.isExample) {
67          [ios2jolie stimeCognitiveTestServer:parent.username.text data:parent
                .todayDate punteggio:risposte risposte:risposteSaved];
68      }
69  }
70
71  - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)
        interfaceOrientation {
72      // Overriden to allow any orientation.
73      return NO;
74  }
75
76  - (void)dealloc {
77    [fine release];
78    [stimeText release];
79    [stimeLabel release];
80    [parent release];
81    [audioPlayer release];
82    [url release];
83    [error release];
84    [super dealloc];
85  }
86
87  //controlla se la risposta dell'utente e' valida
88  void esaminaRisposte (float valore){
```

```
89    //salviamo in un array il valore della risposta per usi futuri
90    risposteSaved[indexDomanda] = valore;
91    //controllo se la risposta e' giusta
92    switch (indexDomanda) {
93      case 0:
94        if ((valore >= 0.50 && valore <= 2.0) || (valore >= 50.0 && valore <=
              99.0)) {
95          risposte++;
96        }
97        break;
98      case 1:
99        if (valore >= 400.0 && valore <= 800.0) {
100         risposte++;
101       }
102       break;
103     case 2:
104       if ((valore >= 60.0 && valore <= 150.0) || (valore >= 0.60 && valore
              <= 1.50)) {
105         risposte++;
106       }
107       break;
108     case 3:
109       if ((valore >= 30.0 && valore <= 90.0) || (valore >= 0.30 && valore <=
               0.59) || (valore >= 1.0 && valore <= 1.30)) {
110         risposte++;
111       }
112       break;
113     case 4:
114       if (valore == 0.0) {
115         risposte++;
116       }
117       break;
118     default:
119       break;
120   }
121   NSLog(@"risposte esatte: %d",risposte);
122 }
123
124 - (IBAction) tastierino:(UIButton *)sender{
125   if (sender.tag==-1 && virgola==FALSE) { //cliccata la virgola ma non
          ancora presente nel display
126     if (stimeText.text.length==0) { //display vuoto
127       stimeText.text = [stimeText.text stringByAppendingString:@"0,"];
128     } else {  //display non vuoto
129       stimeText.text = [stimeText.text stringByAppendingString:@","];
130     }
131     virgola = TRUE;
132   } else if (sender.tag==-2) {  //cliccato il tasto invio
133     //controllo correttezza risposta
134     esaminaRisposte([stimeText.text stringByReplacingOccurrencesOfString:@",
          " withString:@"."].floatValue);
135     if (indexDomanda < 4) { //carico la domanda successiva
136       stimeLabel.text = [domande objectAtIndex:++indexDomanda];
137       stimeText.text = @"";
138       virgola = FALSE;
139     } else {
140       //test finito carico la vista conclusiva
141       [self globalEnd];
142     }
143   } else if (sender.tag > 0) {  //cliccato un numero maggiore di zero
144     if (stimeText.text!=@"0") {
```

```
145       stimeText.text = [stimeText.text stringByAppendingString:[NSString
              stringWithFormat:@"%d",sender.tag]];
146     }
147   } else if (sender.tag == 0) { //cliccato lo zero
148     if (stimeText.text.length==0) {
149       stimeText.text = [stimeText.text stringByAppendingString:@"0"];
150     } else if (stimeText.text!=@"0") {
151       stimeText.text = [stimeText.text stringByAppendingString:@"0"];
152     }
153   } else if (sender.tag == -3) {  //cliccato il tasto canc
154     stimeText.text = @"";
155     virgola = FALSE;
156   }
157 }
158
159 @end
```

Sorgente del Token Test

```
1  //
2  //  Token_TestViewController.h
3  //
4
5  #import <UIKit/UIKit.h>
6  #import "iOS2Jolie.h"
7
8  @class TokenTest;
9  @class TokenTestIV;
10
11 @interface Token_TestViewController : UIViewController {
12     TokenTestIV *tokenTestIV;
13   TokenTest *tokenTest;
14     float punteggioTotale;
15     UITextField *username;
16   UITextField *password;
17     UIAlertView *alert;
18   iOS2Jolie *ios2jolie;
19     BOOL isExample;
20     NSString *todayDate;
21     NSString *serverUrl;
22 }
23
24 @property (nonatomic, retain) IBOutlet TokenTest *tokenTest;
25 @property (nonatomic, retain) IBOutlet TokenTestIV *tokenTestIV;
26 @property (nonatomic, retain) IBOutlet UITextField *username;
27 @property (nonatomic, retain) IBOutlet UITextField *password;
28 @property BOOL isExample;
29 @property (nonatomic, retain) NSString *todayDate;
30 @property (nonatomic, retain) NSString *serverUrl;
31
32 - (IBAction) launchHomeTest:(id)sender;
33 - (void) partIV;
34 - (void) setPunteggio:(float)addValue;
35 - (float) getPunteggio;
36
37 @end
```

```
1  //
2  //  Token_TestViewController.m
3  //
4
5  #import "Token_TestViewController.h"
6  #import "TokenTest.h"
7  #import "TokenTestIV.h"
8  #import <QuartzCore/QuartzCore.h>
9
10 @implementation Token_TestViewController
11
12 @synthesize tokenTest;
13 @synthesize tokenTestIV;
14 @synthesize username;
15 @synthesize password;
16 @synthesize isExample;
17 @synthesize todayDate;
18 @synthesize serverUrl;
19
20 // Implement viewDidLoad to do additional setup after loading the view,
       typically from a nib.
21 - (void)viewDidLoad {
22     punteggioTotale = 0;
23     //SFONDO GRADIENTE
24     CGColorSpaceRef rgb = CGColorSpaceCreateDeviceRGB();
25     CGFloat comps1[] = {0.0, 0.0, 0.0, 1.0};
26     CGFloat comps2[] = {0.0, 0.0, 0.18, 1.0};
27     CGColorRef color1 = CGColorCreate(rgb, comps1);
28     CGColorRef color2 = CGColorCreate(rgb, comps2);
29     CGColorSpaceRelease(rgb);
30     CAGradientLayer *gradient = [CAGradientLayer layer];
31     gradient.frame = self.view.bounds;
32     gradient.colors = [NSArray arrayWithObjects:(id)color1,(id)color2,nil];
33     [self.view.layer insertSublayer:gradient atIndex:0];
34     [super viewDidLoad];
35     //setto l'alert in caso di connessione assente
36     alert = [[UIAlertView alloc] init];
37     [alert setTitle:@"Token Test"];
38     [alert setDelegate:self];
39     [alert setMessage:@"Server non raggiungibile. Vuoi continuare il test
           senza l'invio dei risultati?"];
40     [alert addButtonWithTitle:@"Ok"];
41     [alert addButtonWithTitle:@"No"];
42     //alloco la libreria ios2jolie e setto la data
43     ios2jolie = [[iOS2Jolie alloc] init];
44     NSDateFormatter *today = [[NSDateFormatter alloc] init];
45     [today setDateFormat:@"yyyy-MM-dd HH:mm:ss"];
46     todayDate = [[NSString alloc] initWithString:[today stringFromDate:[
           NSDate date]]];
47     serverUrl = @"localhost";
48 }
49
50 //funzione che porta al menu dei test dopo aver inserito i dati utente
51 - (IBAction) launchHomeTest:(id)sender{
52     if (username.text.length > 0 && password.text.length > 0) {
53     [username resignFirstResponder];
54     [password resignFirstResponder];
55         ////ESCAPE PER IL SETTING SEGRETO DELL'URL DEL SERVER
56         if ([username.text isEqualToString:@"serverurl"]) {
57             ios2jolie.serverUrl = password.text;
58             serverUrl = password.text;
```

```
59              return;
60          }
61          ////FINE ESCAPE
62          if ([ios2jolie checkConnection:FALSE]) {
63              //TODO: check nel database dell'utente
64              [ios2jolie release];
65              isExample = FALSE;
66              tokenTest = [[TokenTest alloc] initWithNibName:@"TokenTest"
                    bundle:nil];
67              tokenTest.parent = self;
68              [self.view addSubview:tokenTest.view];
69          } else {
70              [alert show];
71          }
72    }
73  }
74
75  //chiamata al click sull'alert
76  - (void)alertView:(UIAlertView *)alertView clickedButtonAtIndex:(NSInteger)
        buttonIndex{
77    if (buttonIndex == 0){
78          [ios2jolie release];
79          isExample = TRUE;
80      tokenTest = [[TokenTest alloc] initWithNibName:@"TokenTest" bundle:nil];
81          tokenTest.parent = self;
82          [self.view addSubview:tokenTest.view];
83    }
84  }
85
86  //inizio della parte VI
87  //easter egg: che numero abbiamo usato al posto del VI? (senza motivo!)
88  -(void) partIV{
89      tokenTestIV = [[TokenTestIV alloc] initWithNibName:@"TokenTestIV" bundle
            :nil];
90      tokenTestIV.parent = self;
91      [self.view addSubview:tokenTestIV.view];
92  }
93
94  //incrementa il punteggio dalle due parti del test
95  -(void) setPunteggio:(float)addValue{
96      punteggioTotale+=addValue;
97      NSLog(@"punteggio attuale: %f",punteggioTotale);
98  }
99
100 -(float)getPunteggio{
101     return punteggioTotale;
102 }
103
104 // Override to allow orientations other than the default portrait
        orientation.
105 - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)
        interfaceOrientation {
106     return NO;
107 }
108
109 - (void)dealloc {
110     [super dealloc];
111 }
112
113 @end
```

```
1  //
2  //  TokenTest.h
3  //
4
5  #import <UIKit/UIKit.h>
6  #import <AVFoundation/AVFoundation.h>
7  #import "Token_TestViewController.h"
8
9  @interface TokenTest : UIViewController <AVAudioPlayerDelegate>{
10   NSTimer *timerRepeat;
11   AVAudioPlayer *audioPlayer;
12   UIButton *button11;
13   UIButton *button12;
14   UIButton *button13;
15   UIButton *button14;
16   UIButton *button15;
17   UIButton *button21;
18   UIButton *button22;
19   UIButton *button23;
20   UIButton *button24;
21   UIButton *button25;
22   UIButton *button31;
23   UIButton *button32;
24   UIButton *button33;
25   UIButton *button34;
26   UIButton *button35;
27   UIButton *button41;
28   UIButton *button42;
29   UIButton *button43;
30   UIButton *button44;
31   UIButton *button45;
32   Token_TestViewController *parent;
33     float punteggio; //punteggio finale del test
34     int domanda;  //indice della domanda corrente
35     NSMutableArray *risposte; // array contenente le risposte corrette
36     NSArray *smallButton; //array dei bottoni piccoli
37     NSArray *bigButton;  //array dei bottoni grandi
38     NSArray *domandeBigOnly;  //domande in cui devono essere visibili solo i
           bottoni grandi
39     BOOL sbagliatoSi;  //booleano che indica se il paziente ha dato almeno
           una risposta sbagliata ad una domanda
40     BOOL domandeBigOnlyBool;  //booleano che indica se siamo attualmente ad
           una domanda in cui devono essere visibili solo i bottoni grandi
41     BOOL giaRipetuto;  //booleano che indica se abbiamo gia' ripetuto la
           domanda
42     BOOL haIniziatoARispondere;  //booleano per domande multirisposta che
           indica se il paziente ha gia' risposto parzialmente
43     BOOL commentoAttivo; //boolenao che indica se dobbiamo riprodurre un
           commento prima della domanda
44     NSError *error;
45     NSURL *urlRepeat; //recupera il file audio del commento
46     NSURL *url;  //recupera il file audio da riprodurre
47 }
48
49 @property (nonatomic, retain) NSTimer *timerRepeat;
50 @property (nonatomic, retain) AVAudioPlayer *audioPlayer;
51 @property (nonatomic, retain) IBOutlet UIButton *button11;
52 @property (nonatomic, retain) IBOutlet UIButton *button12;
53 @property (nonatomic, retain) IBOutlet UIButton *button13;
54 @property (nonatomic, retain) IBOutlet UIButton *button14;
55 @property (nonatomic, retain) IBOutlet UIButton *button15;
```

```
56 @property (nonatomic, retain) IBOutlet UIButton *button21;
57 @property (nonatomic, retain) IBOutlet UIButton *button22;
58 @property (nonatomic, retain) IBOutlet UIButton *button23;
59 @property (nonatomic, retain) IBOutlet UIButton *button24;
60 @property (nonatomic, retain) IBOutlet UIButton *button25;
61 @property (nonatomic, retain) IBOutlet UIButton *button31;
62 @property (nonatomic, retain) IBOutlet UIButton *button32;
63 @property (nonatomic, retain) IBOutlet UIButton *button33;
64 @property (nonatomic, retain) IBOutlet UIButton *button34;
65 @property (nonatomic, retain) IBOutlet UIButton *button35;
66 @property (nonatomic, retain) IBOutlet UIButton *button41;
67 @property (nonatomic, retain) IBOutlet UIButton *button42;
68 @property (nonatomic, retain) IBOutlet UIButton *button43;
69 @property (nonatomic, retain) IBOutlet UIButton *button44;
70 @property (nonatomic, retain) IBOutlet UIButton *button45;
71 @property (nonatomic, retain) Token_TestViewController *parent;
72 @property (nonatomic, retain) NSMutableArray *risposte;
73 @property (nonatomic, retain) NSArray *smallButton;
74 @property (nonatomic, retain) NSArray *bigButton;
75 @property (nonatomic, retain) NSArray *domandeBigOnly;
76 @property (nonatomic, retain) NSError *error;
77 @property (nonatomic, retain) NSURL *urlRepeat;
78 @property (nonatomic, retain) NSURL *url;
79
80 -(IBAction)receiveAnswer:(UIButton *)sender;
81
82 @end
```

```
1  //
2  //  TokenTest.m
3  //
4
5  #import "TokenTest.h"
6
7  @implementation TokenTest
8
9  @synthesize audioPlayer;
10 @synthesize timerRepeat;
11 @synthesize button11;
12 @synthesize button12;
13 @synthesize button13;
14 @synthesize button14;
15 @synthesize button15;
16 @synthesize button21;
17 @synthesize button22;
18 @synthesize button23;
19 @synthesize button24;
20 @synthesize button25;
21 @synthesize button31;
22 @synthesize button32;
23 @synthesize button33;
24 @synthesize button34;
25 @synthesize button35;
26 @synthesize button41;
27 @synthesize button42;
28 @synthesize button43;
29 @synthesize button44;
30 @synthesize button45;
31 @synthesize parent;
32 @synthesize risposte;
```

```
33 @synthesize smallButton;
34 @synthesize bigButton;
35 @synthesize domandeBigOnly;
36 @synthesize error;
37 @synthesize urlRepeat;
38 @synthesize url;
39
40 const int MAXDOMANDE = 23; //numero di domande complessivo del test
41 int numRisposte[] = {1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,2,2,2,2,2,2,2,2}; //
       numero risposte attese
42 int numRisposteBackup[] = {1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,2,2,2,2,2,2,2,2,2};
       //backup numRisposte
43
44 //nasconde i bottoni piccoli e sposta i grandi
45 -(void)hiddenSmallButton{
46   for (UIButton *b in smallButton){
47     b.hidden=TRUE;
48   }
49   for (UIButton *b in bigButton){
50     CGRect tempFrame = b.frame;
51     if (b.tag>20) {
52       tempFrame.origin.x-=220;
53       [b setFrame:tempFrame];
54     } else {
55       tempFrame.origin.x-=110;
56       [b setFrame:tempFrame];
57     }
58
59   }
60 }
61
62 //mostra i bottoni piccoli e sposta i grandi
63 -(void)showSmallButton{
64   for (UIButton *b in smallButton){
65     b.hidden=FALSE;
66   }
67   for (UIButton *b in bigButton){
68     CGRect tempFrame = b.frame;
69     if (b.tag>20) {
70       tempFrame.origin.x+=220;
71       [b setFrame:tempFrame];
72     } else {
73       tempFrame.origin.x+=110;
74       [b setFrame:tempFrame];
75     }
76
77   }
78 }
79
80 //disabilito i bottoni durante il play
81 -(void)enableDisableAllButton:(BOOL)mode howAlpha:(float)alpha{
82   for (UIButton *b in smallButton){
83     b.enabled=mode;
84     b.alpha=alpha;
85   }
86   for (UIButton *b in bigButton){
87     b.enabled=mode;
88     b.alpha=alpha;
89   }
90 }
91
```

```
92  // Implement viewDidLoad to do additional setup after loading the view,
        typically from a nib.
93  - (void)viewDidLoad {
94      //inizializzo le variabili globali
95      punteggio = 0;
96      domanda = 0;
97      sbagliatoSi = FALSE;
98      domandeBigOnlyBool = FALSE;
99      giaRipetuto = FALSE;
100     haIniziatoARispondere = FALSE;
101     commentoAttivo = TRUE;
102     [super viewDidLoad];
103   //creo due array di bottoni (piccoli e grandi) per la disabilitazione
104   smallButton = [[NSArray alloc] initWithObjects:button31,button32,button33,
          button34,button35,button41,button42,button43,button44,button45,nil];
105   bigButton = [[NSArray alloc] initWithObjects:button11,button12,button13,
          button14,button15,button21,button22,button23,button24,button25,nil];
106   //disabilito i bottoni
107   [self enableDisableAllButton:FALSE howAlpha:0.7];
108   //creo un array di domande durante le quali devo disabilitare i bottoni
          piccoli
109   domandeBigOnly = [[NSArray alloc] initWithObjects:[NSNumber numberWithInt
          :7],[NSNumber numberWithInt:8],[NSNumber numberWithInt:9],[NSNumber
          numberWithInt:10],[NSNumber numberWithInt:15],[NSNumber numberWithInt
          :16],[NSNumber numberWithInt:17],[NSNumber numberWithInt:18],nil];
110   //inizializzo le risposte
111   //risposte parte 1
112   NSMutableArray *risposta1 = [[NSMutableArray alloc] initWithObjects:[
          NSNumber numberWithInt:11],[NSNumber numberWithInt:12],[NSNumber
          numberWithInt:13],[NSNumber numberWithInt:14],[NSNumber numberWithInt
          :15],[NSNumber numberWithInt:31],[NSNumber numberWithInt:32],[NSNumber
           numberWithInt:33],[NSNumber numberWithInt:34],[NSNumber numberWithInt
          :35],nil];
113   NSMutableArray *risposta2 = [[NSMutableArray alloc] initWithObjects:[
          NSNumber numberWithInt:21],[NSNumber numberWithInt:22],[NSNumber
          numberWithInt:23],[NSNumber numberWithInt:24],[NSNumber numberWithInt
          :25],[NSNumber numberWithInt:41],[NSNumber numberWithInt:42],[NSNumber
           numberWithInt:43],[NSNumber numberWithInt:44],[NSNumber numberWithInt
          :45],nil];
114   NSMutableArray *risposta3 = [[NSMutableArray alloc] initWithObjects:[
          NSNumber numberWithInt:13],[NSNumber numberWithInt:22],[NSNumber
          numberWithInt:34],[NSNumber numberWithInt:41],nil];
115   NSMutableArray *risposta4 = [[NSMutableArray alloc] initWithObjects:[
          NSNumber numberWithInt:14],[NSNumber numberWithInt:25],[NSNumber
          numberWithInt:31],[NSNumber numberWithInt:42],nil];
116   NSMutableArray *risposta5 = [[NSMutableArray alloc] initWithObjects:[
          NSNumber numberWithInt:15],[NSNumber numberWithInt:24],[NSNumber
          numberWithInt:32],[NSNumber numberWithInt:43],nil];
117   NSMutableArray *risposta6 = [[NSMutableArray alloc] initWithObjects:[
          NSNumber numberWithInt:12],[NSNumber numberWithInt:21],[NSNumber
          numberWithInt:33],[NSNumber numberWithInt:45],nil];
118   NSMutableArray *risposta7 = [[NSMutableArray alloc] initWithObjects:[
          NSNumber numberWithInt:11],[NSNumber numberWithInt:23],[NSNumber
          numberWithInt:35],[NSNumber numberWithInt:44],nil];
119   //risposte parte 2
120   NSMutableArray *risposta8 = [[NSMutableArray alloc] initWithObjects:[
          NSNumber numberWithInt:22],nil];
121   NSMutableArray *risposta9 = [[NSMutableArray alloc] initWithObjects:[
          NSNumber numberWithInt:15],nil];
122   NSMutableArray *risposta10 = [[NSMutableArray alloc] initWithObjects:[
          NSNumber numberWithInt:12],nil];
```

```
123     NSMutableArray *risposta11 = [[NSMutableArray alloc] initWithObjects:[
            NSNumber numberWithInt:23],nil];
124     //risposte parte 3
125     NSMutableArray *risposta12 = [[NSMutableArray alloc] initWithObjects:[
            NSNumber numberWithInt:35],nil];
126     NSMutableArray *risposta13 = [[NSMutableArray alloc] initWithObjects:[
            NSNumber numberWithInt:22],nil];
127     NSMutableArray *risposta14 = [[NSMutableArray alloc] initWithObjects:[
            NSNumber numberWithInt:21],nil];
128     NSMutableArray *risposta15 = [[NSMutableArray alloc] initWithObjects:[
            NSNumber numberWithInt:32],nil];
129     //risposte parte 4
130     NSMutableArray *risposta16 = [[NSMutableArray alloc] initWithObjects:[
            NSNumber numberWithInt:14],[NSNumber numberWithInt:21],nil];
131     NSMutableArray *risposta17 = [[NSMutableArray alloc] initWithObjects:[
            NSNumber numberWithInt:22],[NSNumber numberWithInt:24],nil];
132     NSMutableArray *risposta18 = [[NSMutableArray alloc] initWithObjects:[
            NSNumber numberWithInt:23],[NSNumber numberWithInt:12],nil];
133     NSMutableArray *risposta19 = [[NSMutableArray alloc] initWithObjects:[
            NSNumber numberWithInt:11],[NSNumber numberWithInt:14],nil];
134     //risposte parte 5
135     NSMutableArray *risposta20 = [[NSMutableArray alloc] initWithObjects:[
            NSNumber numberWithInt:11],[NSNumber numberWithInt:45],nil];
136     NSMutableArray *risposta21 = [[NSMutableArray alloc] initWithObjects:[
            NSNumber numberWithInt:32],[NSNumber numberWithInt:22],nil];
137     NSMutableArray *risposta22 = [[NSMutableArray alloc] initWithObjects:[
            NSNumber numberWithInt:21],[NSNumber numberWithInt:25],nil];
138     NSMutableArray *risposta23 = [[NSMutableArray alloc] initWithObjects:[
            NSNumber numberWithInt:23],[NSNumber numberWithInt:33],nil];
139     //risposte globali
140     risposte=[[NSMutableArray alloc] initWithObjects:risposta1,risposta2,
            risposta3,risposta4,risposta5,risposta6,risposta7,risposta8,risposta9,
            risposta10,risposta11,risposta12,risposta13,risposta14,risposta15,
            risposta16,risposta17,risposta18,risposta19,risposta20,risposta21,
            risposta22,risposta23,nil];
141     //creazione url del repeat
142     urlRepeat = [[NSURL alloc] initFileURLWithPath:[NSString stringWithFormat:
            @"%@/repeat.mp3", [[NSBundle mainBundle] resourcePath]]];
143     //play della spiegazione introduttiva
144     url = [NSURL fileURLWithPath:[NSString stringWithFormat:@"%@/intro.mp3",
            [[NSBundle mainBundle] resourcePath]]];
145     audioPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:url error:&
            error];
146     audioPlayer.numberOfLoops=0;
147     audioPlayer.delegate=self;
148     NSLog(@"Play intro");
149     [audioPlayer play];
150   }
151
152   //passa alla domanda successiva
153   -(void)nextAsk{
154     if (domanda==MAXDOMANDE-1) {  //domande terminate
155       NSLog(@"Ho finito, punteggio prima parte:%f",punteggio);
156       //termino il test
157           [parent setPunteggio:punteggio];
158           //passo alla sesta parte del test
159       [parent partIV];
160     } else {
161       //disabilito bottoni
162       [self enableDisableAllButton:FALSE howAlpha:0.7];
163       domanda++;
164       sbagliatoSi = FALSE;
```

```
165     haIniziatoARispondere = FALSE;
166     giaRipetuto=FALSE;
167     //nascondo i piccoli se necessario
168     if (!domandeBigOnlyBool && [domandeBigOnly containsObject:[NSNumber
            numberWithInt:domanda]]) {
169       [self hiddenSmallButton];
170       domandeBigOnlyBool=TRUE;
171     } else if (domandeBigOnlyBool && ![domandeBigOnly containsObject:[
            NSNumber numberWithInt:domanda]]) {
172       [self showSmallButton];
173       domandeBigOnlyBool=FALSE;
174     }
175     //play prossima domanda
176     url = [[NSURL alloc] initFileURLWithPath:[NSString stringWithFormat:@"%@
            /%@.mp3", [[NSBundle mainBundle] resourcePath],[NSString
            stringWithFormat:@"%d",domanda+1]]];
177     [audioPlayer release];
178     audioPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:url error:&
            error];
179     audioPlayer.numberOfLoops=0;
180     audioPlayer.delegate=self;
181     NSLog(@"Domanda numero %d",domanda+1);
182     [audioPlayer play];
183   }
184 }
185
186 //dopo la prima presentazione dice "proviamo di nuovo" se il paziente ha
        sbagliato o non ha risposto
187 -(void)repeatAsk{
188   //disabilito bottoni
189   [self enableDisableAllButton:FALSE howAlpha:0.7];
190   //risetto numRisposte per azzerare le risposte precedenti
191   numRisposte[domanda]=numRisposteBackup[domanda];
192     //risetto le risposta della domanda corrente che avevo negato
193     NSMutableArray *domandaCorrente = [risposte objectAtIndex:domanda];
194     for (int i=0;i<numRisposte[domanda];i++) {
195         NSNumber *temp = [domandaCorrente objectAtIndex:i];
196         if ([temp intValue]<0) {
197             NSNumber *reset = [NSNumber numberWithInt:-[temp intValue]];
198             [domandaCorrente replaceObjectAtIndex:i withObject:reset];
199         }
200     }
201   //ripeto la domanda attuale
202   sbagliatoSi=FALSE;
203   giaRipetuto = TRUE;
204   commentoAttivo=TRUE;
205   NSLog(@"Ripeto domanda numero %d",domanda+1);
206   //dico "proviamo di nuovo" oltre a ripetere la domanda
207   [audioPlayer release];
208   audioPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:urlRepeat error
          :&error];
209   audioPlayer.numberOfLoops=0;
210   audioPlayer.delegate=self;
211   [audioPlayer play];
212 }
213
214 //funzione chiamata ad ogni tocco su un pulsante
215 -(IBAction)receiveAnswer:(UIButton *)sender{
216   if (![audioPlayer isPlaying]) {
217         if ((giaRipetuto==FALSE && (haIniziatoARispondere==FALSE ||
              numRisposte[domanda]>1)) || (giaRipetuto && numRisposte[domanda
              ]>1)) {
```

```
218             //se sono alla prima presentazione della domanda e la risposta
                    del paziente e' la prima
219             //oppure
220             //ho gia' ripetuto e attendo altre risposte
221             [timerRepeat invalidate];
222             if (numRisposte[domanda]>1) {
223                 //concedo altri 5 secondo se la domanda e' multirisposta e
                        non ho finito di rispondere
224                 timerRepeat = [NSTimer scheduledTimerWithTimeInterval:5.0
                        target:self selector:@selector(replayAsk:) userInfo:[
                        NSString stringWithFormat:@"%d",-domanda] repeats:NO];
225             }
226         } else {
227             //invalido il timer perche' siamo all'ultima risposta di una
                    domanda, prima della ripetizione
228             [timerRepeat invalidate];
229         }
230         haIniziatoARispondere = TRUE;
231         numRisposte[domanda]--;
232         if (![[risposte objectAtIndex:domanda] containsObject:[NSNumber
                numberWithInt:sender.tag]]) {
233             //ho commesso un errore
234             sbagliatoSi=TRUE;
235         } else {
236             //nego la risposta per evitare ripetizioni nelle multirisposte
237             NSMutableArray *domandaCorrente = [risposte objectAtIndex:
                    domanda];
238             [domandaCorrente replaceObjectAtIndex:[domandaCorrente
                    indexOfObject:[NSNumber numberWithInt:sender.tag]]
                    withObject:[NSNumber numberWithInt:-sender.tag]];
239             //gestione del punteggio: 1 punto giusto, 0.5 punto se giusto
                    dopo la ripetizione
240             if (numRisposte[domanda]==0 && !sbagliatoSi) {
241                 if (giaRipetuto) {
242                     punteggio+=0.5;
243                 } else {
244                     punteggio+=1;
245                 }
246                 NSLog(@"punteggio:%f",punteggio);
247             }
248         }
249         if (numRisposte[domanda]==0) {
250             NSLog(@"Ho dato tutte le risposte");
251             if (sbagliatoSi && !giaRipetuto) { //ha risposto male alla prima
                     presentazione, ripeto la domanda e attendo nuovamente tutte
                     le risposte
252                 //ripeto la domanda attuale
253                 haIniziatoARispondere = FALSE;
254                 [self repeatAsk];
255             } else if (giaRipetuto || !sbagliatoSi) {  //ha risposto bene,
                    vado alla prossima domanda
256                 [self nextAsk];
257             }
258         }
259     }
260 }
261
262 //ripete la domanda
263 -(void)replayAsk:(NSTimer *)timer{
264   int param = ((NSString*)[timer userInfo]).intValue;
265   NSLog(@"param: %d",param);
```

```
266   if (domanda==param || domanda == -param) {  //ripeto la domanda se e'
          scaduto il tempo oppure se e' scaduto il tempo tra una risposta e l'
          altra
267     if (!giaRipetuto) {
268       [self repeatAsk];
269     } else if (!haIniziatoARispondere || domanda == -param){
270       //non ha dato nessuna risposta o risposta incompleta dopo la
              ripetizione, vado alla domanda successiva
271       NSLog(@"Non risponde nemmeno dopo aver ripetuto la domanda %d",domanda
              );
272       [self nextAsk];
273     }
274   }
275 }
276
277 -(void)audioPlayerDidFinishPlaying: (AVAudioPlayer *)player successfully: (
      BOOL)flag{
278   if (commentoAttivo) {
279     commentoAttivo=FALSE;
280     //play della prossima domanda
281     [audioPlayer release];
282     if (domanda==0) {  //solo per la prima domanda
283       url = [NSURL fileURLWithPath:[NSString stringWithFormat:@"%@/%@.mp3",
              [[NSBundle mainBundle] resourcePath],[NSString stringWithFormat:@"
              %d",domanda+1]]];
284     }
285     audioPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:url error:&
          error];
286     audioPlayer.numberOfLoops=0;
287     audioPlayer.delegate=self;
288     NSLog(@"Domanda numero %d",domanda+1);
289     [audioPlayer play];
290   } else {
291     //riabilito i bottoni,eventualmente ripetizione
292     [self enableDisableAllButton:TRUE howAlpha:1.0];
293     timerRepeat = [NSTimer scheduledTimerWithTimeInterval:5.0 target:self
          selector:@selector(replayAsk:) userInfo:[NSString stringWithFormat:@
          "%d",domanda] repeats:NO];
294   }
295 }
296
297 - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)
      interfaceOrientation {
298     // Overriden to allow any orientation.
299     return NO;
300 }
301
302 - (void)dealloc {
303   [timerRepeat release];
304   [audioPlayer release];
305   [button11 release];
306   [button12 release];
307   [button13 release];
308   [button14 release];
309   [button15 release];
310   [button21 release];
311   [button22 release];
312   [button23 release];
313   [button24 release];
314   [button25 release];
315   [button31 release];
316   [button32 release];
```

```
317     [button33 release];
318     [button34 release];
319     [button35 release];
320     [button41 release];
321     [button42 release];
322     [button43 release];
323     [button44 release];
324     [button45 release];
325     [parent release];
326     [risposte release];
327     [smallButton release];
328     [bigButton release];
329     [domandeBigOnly release];
330     [error release];
331     [urlRepeat release];
332     [url release];
333     [super dealloc];
334 }
335 
336 @end
```

```
1  //
2  //  TokenTestIV.h
3  //
4
5  #import <UIKit/UIKit.h>
6  #import <AVFoundation/AVFoundation.h>
7  #import "Token_TestViewController.h"
8  #import "iOS2Jolie.h"
9
10 @interface TokenTestIV : UIViewController <AVAudioPlayerDelegate>{
11   UIView *fine;
12   AVAudioPlayer *audioPlayer;
13   UIButton *button11;
14   UIButton *button12;
15   UIButton *button13;
16   UIButton *button14;
17   UIButton *button15;
18   UIButton *button21;
19   UIButton *button22;
20   UIButton *button23;
21   UIButton *button24;
22   UIButton *button25;
23     Token_TestViewController *parent;
24     float punteggio; //punteggio finale del test
25     int domanda;  //indice della domanda corrente
26     NSMutableArray *risposte; // array contenente le risposte corrette
27     NSMutableArray *squareTag; //array che contiene i tag dei soli quadrati
28     NSMutableArray *circleTag; //array che contiene i tag dei soli cerchi
29     NSArray *bigButton;  //array dei bottoni grandi
30     NSTimer *timer; //timer che ci indica se e' scaduto il tempo
31     NSTimer *question31Timer;
32     NSDate *currentDate;
33     BOOL sbagliatoSi;  //booleano che indica se il paziente ha dato almeno
           una risposta sbagliata ad una domanda
34     BOOL firstQuestion; //boolenao che indica se dobbiamo riprodurre un
           commento prima della domanda
35     BOOL isPlaying; //booleano che indica se i token sono disattivati
36     NSError *error;
37     NSURL *url;  //recupera il file audio da riprodurre
```

```
38      CGPoint currentPoint;
39      CGPoint lastPoint;
40      CGPoint initialCoord[10]; //array delle coordinate originali dei token
41      float clickTime31[9];
42      iOS2Jolie *ios2jolie;
43  }
44
45  @property (nonatomic, retain) IBOutlet UIView *fine;
46  @property (nonatomic, retain) AVAudioPlayer *audioPlayer;
47  @property (nonatomic, retain) IBOutlet UIButton *button11;
48  @property (nonatomic, retain) IBOutlet UIButton *button12;
49  @property (nonatomic, retain) IBOutlet UIButton *button13;
50  @property (nonatomic, retain) IBOutlet UIButton *button14;
51  @property (nonatomic, retain) IBOutlet UIButton *button15;
52  @property (nonatomic, retain) IBOutlet UIButton *button21;
53  @property (nonatomic, retain) IBOutlet UIButton *button22;
54  @property (nonatomic, retain) IBOutlet UIButton *button23;
55  @property (nonatomic, retain) IBOutlet UIButton *button24;
56  @property (nonatomic, retain) IBOutlet UIButton *button25;
57  @property (nonatomic, retain) Token_TestViewController *parent;
58  @property (nonatomic, retain) NSMutableArray *risposte;
59  @property (nonatomic, retain) NSMutableArray *squareTag;
60  @property (nonatomic, retain) NSMutableArray *circleTag;
61  @property (nonatomic, retain) NSArray *bigButton;
62  @property (nonatomic, retain) NSError *error;
63  @property (nonatomic, retain) NSURL *url;
64  @property (nonatomic, retain) NSTimer *timer;
65  @property (nonatomic, retain) NSTimer *question31Timer;
66  @property (nonatomic, retain) NSDate *currentDate;
67
68  -(IBAction)receiveAnswer:(UIButton *)sender;
69
70  @end
```

```
1   //
2   //  TokenTestIV.m
3   //
4
5   #import "TokenTestIV.h"
6
7   @implementation TokenTestIV
8
9   @synthesize fine;
10  @synthesize audioPlayer;
11  @synthesize button11;
12  @synthesize button12;
13  @synthesize button13;
14  @synthesize button14;
15  @synthesize button15;
16  @synthesize button21;
17  @synthesize button22;
18  @synthesize button23;
19  @synthesize button24;
20  @synthesize button25;
21  @synthesize parent;
22  @synthesize risposte;
23  @synthesize squareTag;
24  @synthesize circleTag;
25  @synthesize bigButton;
26  @synthesize error;
```

```
27 @synthesize url;
28 @synthesize timer;
29 @synthesize question31Timer;
30 @synthesize currentDate;
31
32 const int MAXDOMANDEIV = 13; //numero di domande complessivo del test
33 int numRisposteIV[]     = {0,0,2,1,0,0,0,10,0,4,1,1,2}; //numero risposte
       attese
34 bool risposteOneTouch[] = {0,0,1,1,0,1,0, 1,0,1,1,1,1}; //array che indica
       se siamo in una domanda la cui risposta e' data da un classico tocco
35 UIButton *currentToken = NULL;
36
37 //disabilito i bottoni durante il play
38 -(void)enableDisableAllButton:(BOOL)mode howAlpha:(float)alpha{
39   for (UIButton *b in bigButton){
40         b.alpha=alpha;
41     b.enabled=mode;
42   }
43 }
44
45 //disabilito l'interazione durante le domande con movimento
46 -(void)enableDisableAllButtonNoAlpha:(BOOL)mode{
47   for (UIButton *b in bigButton){
48         b.userInteractionEnabled = mode;
49   }
50 }
51
52 //salva in currentToken il tag del token selezionato
53 -(UIButton*)inWhichToken{
54     ///// CONTROLLARE IN CASO DI SOVRAPPOSIZIONE QUALE PRENDE
55     for (UIButton *b in bigButton){
56         if (b.tag < 16) {
57             //controlliamo se siamo in un cerchio
58             if (sqrt(pow(b.center.x-currentPoint.x, 2)+pow(b.center.y-
                   currentPoint.y, 2))<=80){
59                 NSLog(@"Selezionato token con tag: %d",b.tag);
60                 lastPoint = currentPoint;
61                 return b;
62             }
63         } else {
64             //controlliamo se siamo in un quadrato
65             if (b.center.x-80 <= currentPoint.x && currentPoint.x <= b.
                   center.x+80 &&
66                 b.center.y-80 <= currentPoint.y && currentPoint.y <= b.
                       center.y+80){
67                 NSLog(@"Selezionato token con tag: %d",b.tag);
68                 lastPoint = currentPoint;
69                 return b;
70             }
71         }
72   }
73     return NULL;
74 }
75
76 //resetta la posizione dei token
77 -(void)resetTokenCoord{
78     //resetto le posizioni dei token
79     int count = 0;
80     for (UIButton *b in bigButton){
81         b.center = initialCoord[count];
82         count++;
83   }
```

```
84  }
85
86  //verifica se un punto e' interno ad una determianta area
87  -(BOOL)checkArea: (int)px py: (int)py leftx: (int)leftx rightx: (int)rightx
        upy: (int)upy downy: (int)downy {
88      if (leftx<px && px<rightx && upy<py && py<downy) {
89          return TRUE;
90      } else return FALSE;
91  }
92
93  //valutiamo le risposte nelle domande con movimento dei token
94  -(void)evaluateCoord{
95      switch (domanda+24) {
96          case 24:{
97                  //cerchio rosso su quadrato verde
98                  UIButton *greenSquare = [bigButton objectAtIndex:1];
99                  UIButton *redCircle = [bigButton objectAtIndex:0];
100                 if (currentToken == redCircle && sqrt(pow(redCircle.center.x
                        -greenSquare.center.x, 2)+pow(redCircle.center.y-
                        greenSquare.center.y, 2))<80) {
101                     punteggio++;
102                     NSLog(@"sovrapposti correttamente!");
103                     NSLog(@"punteggio: %f",punteggio);
104                 }
105             }
106             break;
107         case 25:{
108                 //tocco cerchio nero con quadrato rosso
109                 UIButton *redSquare = [bigButton objectAtIndex:5];
110                 UIButton *blackCircle = [bigButton objectAtIndex:8];
111                 if (currentToken == redSquare){
112                     //distanza dei centri < ~160
113                     if (sqrt(pow(blackCircle.center.x-redSquare.center.x, 2)
                            +pow(blackCircle.center.y-redSquare.center.y, 2))
                            <160) {
114                         punteggio++;
115                         NSLog(@"Si toccano, distanza < 160");
116                         NSLog(@"punteggio: %f",punteggio);
117                     } else {
118                         //uno spigolo del quadrato interno al cerchio
119                         CGPoint leftup    = CGPointMake(redSquare.center.x
                                -80, redSquare.center.y-80);
120                         CGPoint leftdown  = CGPointMake(redSquare.center.x
                                -80, redSquare.center.y+80);
121                         CGPoint rightup   = CGPointMake(redSquare.center.x
                                +80, redSquare.center.y-80);
122                         CGPoint rightdown = CGPointMake(redSquare.center.x
                                +80, redSquare.center.y+80);
123                         if (sqrt(pow(blackCircle.center.x-leftup.x, 2)+pow(
                                blackCircle.center.y-leftup.y, 2))<80 || sqrt(
                                pow(blackCircle.center.x-leftdown.x, 2)+pow(
                                blackCircle.center.y-leftdown.y, 2))<80 || sqrt(
                                pow(blackCircle.center.x-rightup.x, 2)+pow(
                                blackCircle.center.y-rightup.y, 2))<80 || sqrt(
                                pow(blackCircle.center.x-rightdown.x, 2)+pow(
                                blackCircle.center.y-rightdown.y, 2))<80){
124                             punteggio++;
125                             NSLog(@"Si toccano, spigolo quadrato interno al
                                    cerchio");
126                             NSLog(@"punteggio: %f",punteggio);
127                         } else {
```

```
128                         //uno "spigolo" del cerchio all'interno del
                                quadrato
129                         CGPoint up    = CGPointMake(blackCircle.center.x
                                , blackCircle.center.y-80);
130                         CGPoint down  = CGPointMake(blackCircle.center.x
                                , blackCircle.center.y+80);
131                         CGPoint left  = CGPointMake(blackCircle.center.x
                                -80, blackCircle.center.y);
132                         CGPoint right = CGPointMake(blackCircle.center.x
                                +80, blackCircle.center.y);
133                         if ([self checkArea:up.x py:up.y leftx:leftup.x
                                rightx:rightup.x upy:leftup.y downy:leftdown
                                .y] || [self checkArea:down.x py:down.y
                                leftx:leftup.x rightx:rightup.x upy:leftup.y
                                 downy:leftdown.y] || [self checkArea:left.x
                                 py:left.y leftx:leftup.x rightx:rightup.x
                                upy:leftup.y downy:leftdown.y] || [self
                                checkArea:right.x py:right.y leftx:leftup.x
                                rightx:rightup.x upy:leftup.y downy:leftdown
                                .y]) {
134                             punteggio++;
135                             NSLog(@"Si toccano, \"spigolo\" cerchio
                                    interno al quadrato");
136                             NSLog(@"punteggio: %f",punteggio);
137                         }
138                     }
139                 }
140             }
141         }
142         break;
143     case 28:{
144             //quadrato verde lontano da quadrato rosso
145             UIButton *greenSquare = [bigButton objectAtIndex:1];
146             UIButton *yellowSquare = [bigButton objectAtIndex:2];
147             if (currentToken == greenSquare && sqrt(pow(yellowSquare.
                    center.x-greenSquare.center.x, 2)+pow(yellowSquare.
                    center.y-greenSquare.center.y, 2))>180) {
148                 punteggio++;
149                 NSLog(@"allontanato correttamente!");
150                 NSLog(@"punteggio: %f",punteggio);
151             }
152         }
153         break;
154     case 30:{
155             //quadrato verde vicino a cerchio rosso
156             UIButton *greenSquare = [bigButton objectAtIndex:1];
157             UIButton *redCircle = [bigButton objectAtIndex:0];
158             if (currentToken == greenSquare && sqrt(pow(redCircle.center
                    .x-greenSquare.center.x, 2)+pow(redCircle.center.y-
                    greenSquare.center.y, 2))<600) {
159                 punteggio++;
160                 NSLog(@"avvicinato correttamente!");
161                 NSLog(@"punteggio: %f",punteggio);
162             }
163         }
164         break;
165     case 32:{
166             //cerchio rosso tra quadrato giallo e quadrato verde
167             UIButton *greenSquare = [bigButton objectAtIndex:1];
168             UIButton *yellowSquare = [bigButton objectAtIndex:2];
169             UIButton *redCircle = [bigButton objectAtIndex:0];
170             int minx,maxx,miny,maxy;
```

```
171                 minx = MIN(greenSquare.center.x-80, yellowSquare.center.x
                        -80);
172                 maxx = MAX(greenSquare.center.x+80, yellowSquare.center.x
                        +80);
173                 miny = MIN(greenSquare.center.y, yellowSquare.center.y);
174                 maxy = MAX(greenSquare.center.y, yellowSquare.center.y);
175                 if (currentToken == redCircle && [self checkArea:redCircle.
                        center.x py:redCircle.center.y leftx:minx rightx:maxx
                        upy:miny downy:maxy]) {
176                     punteggio++;
177                     NSLog(@"in mezzo correttamente!");
178                     NSLog(@"punteggio: %f",punteggio);
179                 }
180             }
181             break;
182         default:
183             break;
184     }
185 }
186
187 //funzione per la valutazione della domanda 31 (tocchi lentamente i quadrati
        e velocemente i cerchi)
188 -(void)question31evaluate:(int)currentTag{
189     //se non ha gia' sbagliato in precedenza entro nella funzione
190     if (!sbagliatoSi) {
191         //tengo traccia dei tempi tra i vari click
192         if (numRisposteIV[domanda]==9) {
193             //siamo al primo click
194             question31Timer = [NSTimer scheduledTimerWithTimeInterval:60.0
                        target:self selector:@selector(nextAsk) userInfo:nil repeats
                        :NO];
195         } else {
196             //siamo ai click successivi
197             currentDate = [NSDate date];
198             NSCalendar *calendar = [[[NSCalendar alloc]
                        initWithCalendarIdentifier:NSGregorianCalendar] autorelease
                        ];
199             NSDateComponents *components = [calendar components:
                        NSSecondCalendarUnit fromDate:currentDate toDate:[
                        question31Timer fireDate] options:0];
200             clickTime31[9-numRisposteIV[domanda]-1] = 60.0 - components.
                        second;
201             NSLog(@"secondi: %f",60.0 - components.second);
202         }
203         if (numRisposteIV[domanda]>=5) {
204             //aspetto il click su un quadrato
205             if ([squareTag containsObject:[NSNumber numberWithInt:currentTag
                        ]]) {
206                 [squareTag removeObjectAtIndex:[squareTag indexOfObject:[
                        NSNumber numberWithInt:currentTag]]];
207                 NSLog(@"lunghezza array sq:%d",[squareTag count]);
208             } else {
209                 //quadrato gia' cliccato, oppure cliccato un cerchio
210                 sbagliatoSi = TRUE;
211             }
212         } else {
213             //aspetto il click su un cerchio
214             if ([circleTag containsObject:[NSNumber numberWithInt:currentTag
                        ]]) {
215                 [circleTag removeObjectAtIndex:[circleTag indexOfObject:[
                        NSNumber numberWithInt:currentTag]]];
216                 NSLog(@"lunghezza array ci:%d",[circleTag count]);
```

```
217             } else {
218                 //cerchio gia' cliccato, oppure click su un quadrato
219                 sbagliatoSi = TRUE;
220             }
221         }
222         if (numRisposteIV[domanda]==0) {
223             //controllo i tempi
224             for (int i=8; i>0; i--) {
225                 clickTime31[i]-=clickTime31[i-1];
226             }
227             float meanSquare = (clickTime31[0]+clickTime31[1]+clickTime31
                    [2]+clickTime31[3])/4;
228             float meanCircle = (clickTime31[5]+clickTime31[6]+clickTime31
                    [7]+clickTime31[8])/4;
229             NSLog(@"mediaSq: %f, mediaCi: %f",meanSquare,meanCircle);
230             //percentuale di incremento del 50%
231             if (meanCircle*1.5 > meanSquare) {
232                 sbagliatoSi = TRUE;
233             }
234         }
235     }
236     if (numRisposteIV[domanda]==0) {
237         [question31Timer invalidate];
238     }
239 }
240
241 // Implement viewDidLoad to do additional setup after loading the view,
        typically from a nib.
242 - (void)viewDidLoad {
243     //inizializzo le variabili globali
244     punteggio = 0;
245     domanda = 0;
246     sbagliatoSi = FALSE;
247     firstQuestion = TRUE;
248     isPlaying = TRUE;
249     //alloco i servizi per l'invio dei dati
250     ios2jolie = [[iOS2Jolie alloc] init];
251     ios2jolie.serverUrl = parent.serverUrl;
252     [super viewDidLoad];
253   //creo array di bottoni grandi per la disabilitazione
254   bigButton = [[NSArray alloc] initWithObjects:button14,button21,button22,
          button23,button24,button25,button12,button13,button15,button11,nil];
255     //inizializzo gli array con i tag
256     circleTag = [[NSMutableArray alloc] initWithObjects:[NSNumber
            numberWithInt:11],[NSNumber numberWithInt:12],[NSNumber
            numberWithInt:13],[NSNumber numberWithInt:14],[NSNumber
            numberWithInt:15],nil];
257     squareTag = [[NSMutableArray alloc] initWithObjects:[NSNumber
            numberWithInt:21],[NSNumber numberWithInt:22],[NSNumber
            numberWithInt:23],[NSNumber numberWithInt:24],[NSNumber
            numberWithInt:25],nil];
258     int count = 0;
259     for (UIButton *b in bigButton){
260         initialCoord[count] = b.center;
261         count++;
262   }
263   //disabilito i bottoni
264   [self enableDisableAllButton:FALSE howAlpha:0.7];
265   //inizializzo le risposte
266   //risposte solo parte 6
267     NSMutableArray *risposta24 = [[NSMutableArray alloc] init];
268     NSMutableArray *risposta25 = [[NSMutableArray alloc] init];
```

```
269     NSMutableArray *risposta26 = [[NSMutableArray alloc] initWithObjects:[
            NSNumber numberWithInt:15],[NSNumber numberWithInt:25],nil];
270   NSMutableArray *risposta27 = [[NSMutableArray alloc] initWithObjects:[
          NSNumber numberWithInt:15],[NSNumber numberWithInt:25],nil];
271     NSMutableArray *risposta28 = [[NSMutableArray alloc] init];
272     NSMutableArray *risposta29 = [[NSMutableArray alloc] initWithObjects:[
            NSNumber numberWithInt:-1],nil];
273     NSMutableArray *risposta30 = [[NSMutableArray alloc] init];
274     NSMutableArray *risposta31 = [[NSMutableArray alloc] initWithObjects:[
            NSNumber numberWithInt:11],[NSNumber numberWithInt:12],[NSNumber
            numberWithInt:13],[NSNumber numberWithInt:14],[NSNumber
            numberWithInt:15],[NSNumber numberWithInt:21],[NSNumber
            numberWithInt:22],[NSNumber numberWithInt:23],[NSNumber
            numberWithInt:24],[NSNumber numberWithInt:25],nil];
275     NSMutableArray *risposta32 = [[NSMutableArray alloc] init];
276   NSMutableArray *risposta33 = [[NSMutableArray alloc] initWithObjects:[
          NSNumber numberWithInt:11],[NSNumber numberWithInt:13],[NSNumber
          numberWithInt:14],[NSNumber numberWithInt:15],nil];
277   NSMutableArray *risposta34 = [[NSMutableArray alloc] initWithObjects:[
          NSNumber numberWithInt:23],nil];
278   NSMutableArray *risposta35 = [[NSMutableArray alloc] initWithObjects:[
          NSNumber numberWithInt:13],nil];
279   NSMutableArray *risposta36 = [[NSMutableArray alloc] initWithObjects:[
          NSNumber numberWithInt:13],[NSNumber numberWithInt:15],nil];
280   //risposte globali
281   risposte=[[NSMutableArray alloc] initWithObjects:risposta24,risposta25,
          risposta26,risposta27,risposta28,risposta29,risposta30,risposta31,
          risposta32,risposta33,risposta34,risposta35,risposta36,nil];
282   //play della spiegazione introduttiva
283   url = [NSURL fileURLWithPath:[NSString stringWithFormat:@"%@/intro2.mp3",
          [[NSBundle mainBundle] resourcePath]]];
284   audioPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:url error:&
          error];
285   audioPlayer.numberOfLoops=0;
286   audioPlayer.delegate=self;
287   NSLog(@"Play intro");
288   [audioPlayer play];
289 }
290
291 //passa alla domanda successiva
292 -(void)nextAsk{
293     isPlaying = TRUE;
294     if (!risposteOneTouch[domanda]) {
295         //valutiamo le risposte
296         [self evaluateCoord];
297         //resetto le coordinate
298         [self resetTokenCoord];
299     }
300     currentToken = NULL;
301     //domande terminate
302   if (domanda==MAXDOMANDEIV-1) {
303     NSLog(@"Ho finito, punteggio parte due:%f",punteggio);
304         [parent setPunteggio:punteggio];
305     //termino il test
306     self.view = self.fine;
307         [self.view setUserInteractionEnabled:FALSE];
308         //mando dati al server
309         if (!parent.isExample) {
310             [ios2jolie tokenTestServer:parent.username.text data:parent.
                    todayDate punteggio:[parent getPunteggio]];
311         }
312   } else {
```

```
313      //disabilito bottoni
314          [self enableDisableAllButtonNoAlpha:TRUE];
315      [self enableDisableAllButton:FALSE howAlpha:0.7];
316          //caso particolare domanda 29, non deve toccare niente
317          if (domanda+24 == 29) {
318              NSLog(@"Aggiungo un +1 per la domanda 29");
319              punteggio++;
320          }
321      domanda++;
322      sbagliatoSi = FALSE;
323      //play prossima domanda
324      url = [[NSURL alloc] initFileURLWithPath:[NSString stringWithFormat:@"%@
             /%@.mp3", [[NSBundle mainBundle] resourcePath],[NSString
             stringWithFormat:@"%d",domanda+24]]];
325      [audioPlayer release];
326      audioPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:url error:&
             error];
327      audioPlayer.numberOfLoops=0;
328      audioPlayer.delegate=self;
329      NSLog(@"Domanda numero %d",domanda+24);
330      [audioPlayer play];
331    }
332  }
333
334  //funzione chiamata ad ogni tocco su un pulsante
335  -(IBAction)receiveAnswer:(UIButton *)sender{
336    if (![audioPlayer isPlaying]) {
337          if (numRisposteIV[domanda]>1) {
338              //se la risposta del paziente e' la prima in una domanda
                     multirisposta concedo altri secondi
339              [timer invalidate];
340              timer = [NSTimer scheduledTimerWithTimeInterval:5.0 target:self
                     selector:@selector(nextAsk) userInfo:nil repeats:NO];
341          } else if (numRisposteIV[domanda]==1){
342              [timer invalidate];
343          }
344          numRisposteIV[domanda]--;
345          //siamo alla domanda 29, in cui il paziente deve stare fermo
346          if (domanda+24 == 29) {
347              [timer invalidate];
348              NSLog(@"Ha toccato un pulsante nonostante siamo alla domanda 29,
                      -1!");
349              punteggio--;
350              [self nextAsk];
351          } else {
352              //siamo in una domanda la cui risposta e' data da classici click
353              if (risposteOneTouch[domanda]){
354                  if (![[risposte objectAtIndex:domanda] containsObject:[
                         NSNumber numberWithInt:sender.tag]]) {
355                      //ho commesso un errore
356                      sbagliatoSi=TRUE;
357                  } else {
358                      if (domanda+24 == 31) {
359                          [self question31evaluate:sender.tag];
360                      }
361                      //gestione del punteggio: 1 punto giusto
362                      if (numRisposteIV[domanda]==0 && !sbagliatoSi) {
363                          punteggio+=1;
364                          NSLog(@"punteggio:%f",punteggio);
365                      }
366                      //rimozione risposta esatta dall'array (tranne che per
                             la 31 e per la 29)
```

```
367                     if (domanda+24 != 31) {
368                         NSMutableArray *domandaCorrente = [risposte
                                objectAtIndex:domanda];
369                         [domandaCorrente removeObjectAtIndex:[
                                domandaCorrente indexOfObject:[NSNumber
                                numberWithInt:sender.tag]]];
370                     }
371                 }
372                 if (numRisposteIV[domanda]==0) {
373                     [self nextAsk];
374                 }
375             }
376         }
377     }
378 }
379
380 // richiamata alla fine di un audio
381 -(void)audioPlayerDidFinishPlaying: (AVAudioPlayer *)player successfully: (
        BOOL)flag{
382   if (firstQuestion) {
383         //siamo alla prima domanda
384     firstQuestion=FALSE;
385     //play della prossima domanda
386     [audioPlayer release];
387         url = [NSURL fileURLWithPath:[NSString stringWithFormat:@"%@/%@.mp3"
                , [[NSBundle mainBundle] resourcePath],[NSString
                stringWithFormat:@"%d",domanda+24]]];
388     audioPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:url error:&
            error];
389     audioPlayer.numberOfLoops=0;
390     audioPlayer.delegate=self;
391     NSLog(@"Domanda numero %d",domanda+24);
392     [audioPlayer play];
393   } else {
394         isPlaying = FALSE;
395         if (risposteOneTouch[domanda]) {
396             //domanda da rispondere con click
397             [self enableDisableAllButton:TRUE howAlpha:1.0];
398         } else {
399             //domanda in cui si usa il movimento delle dita
400             [self enableDisableAllButton:TRUE howAlpha:1.0];
401             [self enableDisableAllButtonNoAlpha:FALSE];
402         }
403     timer = [NSTimer scheduledTimerWithTimeInterval:5.0 target:self selector
            :@selector(nextAsk) userInfo:nil repeats:NO];
404   }
405 }
406
407 -(void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event{
408     if (!isPlaying) {
409         UITouch *touch = [touches anyObject];
410         currentPoint = [touch locationInView:self.view];
411         currentToken = [self inWhichToken];
412     }
413 }
414
415 -(void)touchesMoved:(NSSet *)touches withEvent:(UIEvent *)event{
416     if (currentToken!=NULL && !isPlaying) {
417         UITouch *touch = [touches anyObject];
418         currentPoint = [touch locationInView:self.view];
419         CGPoint tempPoint;
420         tempPoint.x = currentToken.center.x + (currentPoint.x-lastPoint.x);
```

```
421         tempPoint.y = currentToken.center.y + (currentPoint.y-lastPoint.y);
422         currentToken.center = tempPoint;
423         lastPoint = currentPoint;
424     }
425 
426 }
427 
428 -(void)touchesEnded:(NSSet *)touches withEvent:(UIEvent *)event{
429 
430 }
431 
432 - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)
      interfaceOrientation {
433     // Overriden to allow any orientation.
434     return NO;
435 }
436 
437 - (void)dealloc {
438   [fine release];
439   [timer release];
440   [audioPlayer release];
441   [button11 release];
442   [button12 release];
443   [button13 release];
444   [button14 release];
445   [button15 release];
446   [button21 release];
447   [button22 release];
448   [button23 release];
449   [button24 release];
450   [button25 release];
451   [parent release];
452   [risposte release]; // array contenente le risposte corrette
453   [squareTag release]; //array che contiene i tag dei soli quadrati
454   [circleTag release]; //array che contiene i tag dei soli cerchi
455   [bigButton release];  //array dei bottoni grandi
456   [question31Timer release];
457   [currentDate release];
458   [error release];
459   [url release];
460   [currentToken release];
461   [super dealloc];
462 }
463 
464 @end
```

Sorgente del Trail Making Test

```
1  //
2  //  Trail_Making_Test_AViewController.h
3  //
4 
5  #import <UIKit/UIKit.h>
6  #import "iOS2Jolie.h"
7 
8  @class TrailMakingTestA;
```

```
9  @class TrailMakingTestA25;
10 @class TrailMakingTestB;
11 @class TrailMakingTestB25;
12
13 @interface Trail_Making_Test_AViewController : UIViewController {
14   TrailMakingTestA25 *trailMakingTestA25;
15   TrailMakingTestB25 *trailMakingTestB25;
16   TrailMakingTestA *trailMakingTestA;
17   TrailMakingTestB *trailMakingTestB;
18   UIView *fine;
19     UITextField *username;
20   UITextField *password;
21     UIAlertView *alert;
22   iOS2Jolie *ios2jolie;
23     BOOL isExample;
24     NSString *todayDate;
25     NSString *serverUrl;
26 }
27
28 @property (nonatomic, retain) IBOutlet TrailMakingTestA *trailMakingTestA;
29 @property (nonatomic, retain) IBOutlet TrailMakingTestB *trailMakingTestB;
30 @property (nonatomic, retain) IBOutlet TrailMakingTestA25 *
       trailMakingTestA25;
31 @property (nonatomic, retain) IBOutlet TrailMakingTestB25 *
       trailMakingTestB25;
32 @property (nonatomic, retain) IBOutlet UIView *fine;
33 @property (nonatomic, retain) IBOutlet UITextField *username;
34 @property (nonatomic, retain) IBOutlet UITextField *password;
35 @property BOOL isExample;
36 @property (nonatomic, retain) NSString *todayDate;
37 @property (nonatomic, retain) NSString *serverUrl;
38
39 - (IBAction) launchHomeTest:(id)sender;
40 - (void) nextTest;
41 - (void) endTest;
42
43 @end
```

```
1  //
2  //  Trail_Making_Test_AViewController.m
3  //
4
5  #import "Trail_Making_Test_AViewController.h"
6  #import "TrailMakingTestA.h"
7  #import "TrailMakingTestA25.h"
8  #import "TrailMakingTestB.h"
9  #import "TrailMakingTestB25.h"
10 #import <QuartzCore/QuartzCore.h>
11
12 @implementation Trail_Making_Test_AViewController
13
14 @synthesize trailMakingTestA,trailMakingTestA25,trailMakingTestB,
       trailMakingTestB25;
15 @synthesize fine;
16 @synthesize username;
17 @synthesize password;
18 @synthesize isExample;
19 @synthesize todayDate;
20 @synthesize serverUrl;
21
```

```
22 int cascata = 0;
23
24 // Implement viewDidLoad to do additional setup after loading the view,
       typically from a nib.
25 - (void)viewDidLoad {
26     //SFONDO GRADIENTE
27     CGColorSpaceRef rgb = CGColorSpaceCreateDeviceRGB();
28     CGFloat comps1[] = {0.0, 0.0, 0.0, 1.0};
29     CGFloat comps2[] = {0.0, 0.0, 0.18, 1.0};
30     CGColorRef color1 = CGColorCreate(rgb, comps1);
31     CGColorRef color2 = CGColorCreate(rgb, comps2);
32     CGColorSpaceRelease(rgb);
33     CAGradientLayer *gradient = [CAGradientLayer layer];
34     gradient.frame = self.view.bounds;
35     gradient.colors = [NSArray arrayWithObjects:(id)color1,(id)color2,nil];
36     [self.view.layer insertSublayer:gradient atIndex:0];
37     [super viewDidLoad];
38     //setto l'alert in caso di connessione assente
39     alert = [[UIAlertView alloc] init];
40     [alert setTitle:@"Trail Making Test"];
41     [alert setDelegate:self];
42     [alert setMessage:@"Server non raggiungibile. Vuoi continuare il test
           senza l'invio dei risultati?"];
43     [alert addButtonWithTitle:@"Ok"];
44     [alert addButtonWithTitle:@"No"];
45     //alloco la libreria ios2jolie e setto la data
46     ios2jolie = [[iOS2Jolie alloc] init];
47     NSDateFormatter *today = [[NSDateFormatter alloc] init];
48     [today setDateFormat:@"yyyy-MM-dd HH:mm:ss"];
49     todayDate = [[NSString alloc] initWithString:[today stringFromDate:[
           NSDate date]]];
50     serverUrl = @"localhost";
51 }
52
53 //funzione che porta al menu dei test dopo aver inserito i dati utente
54 - (IBAction) launchHomeTest:(id)sender{
55     if (username.text.length > 0 && password.text.length > 0) {
56     [username resignFirstResponder];
57     [password resignFirstResponder];
58         ////ESCAPE PER IL SETTING SEGRETO DELL'URL DEL SERVER
59         if ([username.text isEqualToString:@"serverurl"]) {
60             ios2jolie.serverUrl = password.text;
61             serverUrl = password.text;
62             return;
63         }
64         ////FINE ESCAPE
65         if ([ios2jolie checkConnection:FALSE]) {
66             //TODO: check nel database dell'utente
67             [ios2jolie release];
68             isExample = FALSE;
69             trailMakingTestA = [[TrailMakingTestA alloc] initWithNibName:@"
                   TrailMakingTestA" bundle:nil];
70             trailMakingTestA.parent = self;
71             [self.view setUserInteractionEnabled:FALSE];
72             [self.view addSubview:trailMakingTestA.view];
73         } else {
74             [alert show];
75         }
76   }
77 }
78
79 //chiamata al click sull'alert
```

```
80  - (void)alertView:(UIAlertView *)alertView clickedButtonAtIndex:(NSInteger)
        buttonIndex{
81    if (buttonIndex == 0){
82          [ios2jolie release];
83          isExample = TRUE;
84      trailMakingTestA = [[TrailMakingTestA alloc] initWithNibName:@"
            TrailMakingTestA" bundle:nil];
85          trailMakingTestA.parent = self;
86          [self.view setUserInteractionEnabled:FALSE];
87          [self.view addSubview:trailMakingTestA.view];
88    }
89  }
90
91  //richiamata quando viene rimossa la subview
92  - (void)nextTest{
93    switch (cascata) {
94      case 0:
95        trailMakingTestA25 = [[TrailMakingTestA25 alloc] initWithNibName:@"
              TrailMakingTestA25" bundle:nil];
96        trailMakingTestA25.parent = self;
97        //[trailMakingTestA dealloc];
98        [self.view setUserInteractionEnabled:FALSE];
99        [self.view addSubview:trailMakingTestA25.view];
100       cascata++;
101       break;
102     case 1:
103       trailMakingTestB = [[TrailMakingTestB alloc] initWithNibName:@"
              TrailMakingTestB" bundle:nil];
104       trailMakingTestB.parent = self;
105       //[trailMakingTestA25 dealloc];
106       [self.view setUserInteractionEnabled:FALSE];
107       [self.view addSubview:trailMakingTestB.view];
108       cascata++;
109       break;
110     case 2:
111       trailMakingTestB25 = [[TrailMakingTestB25 alloc] initWithNibName:@"
              TrailMakingTestB25" bundle:nil];
112       trailMakingTestB25.parent = self;
113       //[trailMakingTestB dealloc];
114       [self.view setUserInteractionEnabled:FALSE];
115       [self.view addSubview:trailMakingTestB25.view];
116       cascata++;
117       break;
118     case 3:
119       [self endTest];
120       break;
121
122     default:
123       break;
124   }
125
126
127 }
128
129 //richiamata se l'utente non riesce a completare un test (e quindi non deve
        svolgerne altri)
130 - (void) endTest{
131   self.view = self.fine;
132   [self.view setUserInteractionEnabled:FALSE];
133 }
134
```

```
135 // Override to allow orientations other than the default portrait
        orientation.
136 - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)
        interfaceOrientation {
137     return NO;
138 }
139
140 - (void)dealloc {
141   [trailMakingTestA25 release];
142   [trailMakingTestB25 release];
143   [trailMakingTestA release];
144   [trailMakingTestB release];
145   [username release];
146   [password release];
147   [fine release];
148   [super dealloc];
149 }
150
151 @end
```

```
1  //
2  //  TrailMakingTestA.h
3  //
4
5  #import <UIKit/UIKit.h>
6  #import <AVFoundation/AVFoundation.h>
7  #import "Trail_Making_Test_AViewController.h"
8
9  @interface TrailMakingTestA : UIViewController <AVAudioPlayerDelegate>{
10   Trail_Making_Test_AViewController *parent;
11   CGPoint lastPoint;
12   UIImageView *drawImage;
13   UILabel *label1;
14   UILabel *label2;
15   UILabel *label3;
16   UILabel *label4;
17   UILabel *label5;
18   UILabel *label6;
19   UILabel *label7;
20   UILabel *label8;
21   NSTimer *globalTimer;
22   NSTimer *localTimer;
23   AVAudioPlayer *audioPlayer;
24   int expectedValue; //il prossimo valore da toccare
25   int liftFinger;  //il numero di volte che il dito e' stato alzato
26   int missNumber;  //il numero di errori (numero sbagliato)
27   BOOL expectedZero; //indica che abbiamo toccato un numero e dobbiamo
         uscire dal suo cerchio
28   BOOL localTimerBool;
29   NSURL *urlIntro;
30   NSURL *urlStart;
31   NSURL *urlFinger;
32   NSURL *urlNumber;
33   NSURL *urlHelper;
34   NSError *error;
35 }
36
37 @property (nonatomic, retain) Trail_Making_Test_AViewController *parent;
38 @property (nonatomic, retain) AVAudioPlayer *audioPlayer;
39 @property (nonatomic, retain) IBOutlet UILabel *label1;
```

```
40 @property (nonatomic, retain) IBOutlet UILabel *label2;
41 @property (nonatomic, retain) IBOutlet UILabel *label3;
42 @property (nonatomic, retain) IBOutlet UILabel *label4;
43 @property (nonatomic, retain) IBOutlet UILabel *label5;
44 @property (nonatomic, retain) IBOutlet UILabel *label6;
45 @property (nonatomic, retain) IBOutlet UILabel *label7;
46 @property (nonatomic, retain) IBOutlet UILabel *label8;
47 @property (nonatomic, retain) UIImageView *drawImage;
48 @property (nonatomic, retain) NSTimer *globalTimer;
49 @property (nonatomic, retain) NSTimer *localTimer;
50 @property int expectedValue;
51 @property int liftFinger;
52 @property int missNumber;
53 @property BOOL expectedZero;
54 @property BOOL localTimerBool;
55 @property (nonatomic, retain) NSURL *urlIntro;
56 @property (nonatomic, retain) NSURL *urlStart;
57 @property (nonatomic, retain) NSURL *urlFinger;
58 @property (nonatomic, retain) NSURL *urlNumber;
59 @property (nonatomic, retain) NSURL *urlHelper;
60 @property (nonatomic, retain) NSError *error;
61
62 @end
```

```
1  //
2  //  TrailMakingTestA.m
3  //
4
5  #import "TrailMakingTestA.h"
6  #import <QuartzCore/QuartzCore.h>
7
8  @implementation TrailMakingTestA
9
10 @synthesize audioPlayer;
11 @synthesize label1;
12 @synthesize label2;
13 @synthesize label3;
14 @synthesize label4;
15 @synthesize label5;
16 @synthesize label6;
17 @synthesize label7;
18 @synthesize label8;
19 @synthesize drawImage;
20 @synthesize globalTimer;
21 @synthesize localTimer;
22 @synthesize parent;
23 @synthesize expectedValue;
24 @synthesize liftFinger;
25 @synthesize missNumber;
26 @synthesize expectedZero;
27 @synthesize localTimerBool;
28 @synthesize urlIntro;
29 @synthesize urlStart;
30 @synthesize urlFinger;
31 @synthesize urlNumber;
32 @synthesize urlHelper;
33 @synthesize error;
34
35 float xCenterA[] = {586,390,687,702,318,466,224,111};  //centri dei cerchi
36 float yCenterA[] = {515,679,811,167,141,357,526,769};
```

```
37
38 // Implement viewDidLoad to do additional setup after loading the view,
       typically from a nib.
39 - (void)viewDidLoad {
40   //setto le variabili globali
41   expectedValue = 1; //il prossimo valore da toccare
42   liftFinger = 0;  //il numero di volte che il dito e' stato alzato
43   missNumber = 0;  //il numero di errori (numero sbagliato)
44   expectedZero = FALSE; //indica che abbiamo toccato un numero e dobbiamo
         uscire dal suo cerchio
45   localTimerBool = FALSE;
46   [super viewDidLoad];
47   //creo la view in cui disegno
48   drawImage = [[UIImageView alloc] initWithFrame:CGRectMake(0, 0, 768, 1024)
         ];
49   //funzioni di disegno
50   //definisce il context in cui disegnare
51   UIGraphicsBeginImageContext(drawImage.frame.size);
52   //delimita l'area in cui disegnare
53   [drawImage.image drawInRect:CGRectMake(0, 0, drawImage.frame.size.width,
         drawImage.frame.size.height)];
54   CGContextRef context = UIGraphicsGetCurrentContext();
55   CGContextSetLineWidth(context, 1.0);
56   //Crea i cerchi
57   CGContextAddEllipseInRect(context, CGRectMake(561, 490, 50, 50));
58   CGContextAddEllipseInRect(context, CGRectMake(365, 654, 50, 50));
59   CGContextAddEllipseInRect(context, CGRectMake(662, 786, 50, 50));
60   CGContextAddEllipseInRect(context, CGRectMake(677, 142, 50, 50));
61   CGContextAddEllipseInRect(context, CGRectMake(293, 116, 50, 50));
62   CGContextAddEllipseInRect(context, CGRectMake(441, 332, 50, 50));
63   CGContextAddEllipseInRect(context, CGRectMake(199, 501, 50, 50));
64   CGContextAddEllipseInRect(context, CGRectMake( 86, 744, 50, 50));
65   CGContextDrawPath(context, kCGPathStroke);
66   //prende la view prima di disegnarci sopra
67   drawImage.image = UIGraphicsGetImageFromCurrentImageContext();
68   //disegna le nuove linee
69   UIGraphicsEndImageContext();
70   //mostra la vista in cui disegno
71   [self.view setUserInteractionEnabled:FALSE];
72   [self.view addSubview:drawImage];
73   //allocazione di tutti gli URL
74   urlIntro = [NSURL fileURLWithPath:[NSString stringWithFormat:@"%@/
         tmta_prova.mp3", [[NSBundle mainBundle] resourcePath]]];
75   urlStart = [[NSURL alloc] initFileURLWithPath:[NSString stringWithFormat:@
         "%@/start.mp3", [[NSBundle mainBundle] resourcePath]]];
76   urlFinger = [[NSURL alloc] initFileURLWithPath:[NSString stringWithFormat:
         @"%@/finger.mp3", [[NSBundle mainBundle] resourcePath]]];
77   urlNumber = [[NSURL alloc] initFileURLWithPath:[NSString stringWithFormat:
         @"%@/number.mp3", [[NSBundle mainBundle] resourcePath]]];
78   urlHelper = [[NSURL alloc] initFileURLWithPath:[NSString stringWithFormat:
         @"%@/helper.mp3", [[NSBundle mainBundle] resourcePath]]];
79   //play della spiegazione introduttiva
80   audioPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:urlIntro error
         :&error];
81   audioPlayer.numberOfLoops=0;
82   audioPlayer.delegate=self;
83   NSLog(@"Play intro");
84   [audioPlayer play];
85 }
86
87 //funzione chiamata al termine del timer globale
88 - (void) globalEnd{
```

```
89    //non ha finito il test nei tempi, test finito
90    [self.view setUserInteractionEnabled:FALSE];
91    [parent endTest];
92  }
93
94  //funzione per la riproduzione dei vari audio
95  - (void) playSound:(int)numUrl{
96    BOOL isPlaying = audioPlayer.playing;
97    NSLog(@"isPlaying: %d",isPlaying);
98    if (!isPlaying) {
99      switch (numUrl) {
100       case 1:
101         audioPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:urlStart
                error:&error];
102         break;
103       case 2:
104         audioPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:urlFinger
                error:&error];
105         break;
106       case 3:
107         audioPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:urlNumber
                error:&error];
108         break;
109       case 4:
110         audioPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:urlHelper
                error:&error];
111         break;
112       default:
113         break;
114     }
115     audioPlayer.numberOfLoops=0;
116     [audioPlayer play];
117   }
118 }
119
120 //funzione di aiuto
121 - (void) helper{
122   localTimerBool = FALSE;
123   [self playSound:4];
124   //disegno del tratto
125   UIGraphicsBeginImageContext(drawImage.frame.size);
126   [drawImage.image drawInRect:CGRectMake(0, 0, drawImage.frame.size.width,
        drawImage.frame.size.height)];
127   CGContextRef context = UIGraphicsGetCurrentContext();
128   CGContextSetLineCap(context, kCGLineCapRound);
129   CGContextSetLineWidth(context, 2.0);
130   CGContextSetRGBStrokeColor(context, 0.11, 0.89, 0.15, 1.0);
131   int n = 1;
132   if (expectedValue>1) {
133     n=2;
134   }
135   CGContextMoveToPoint(context, xCenterA[expectedValue-n], yCenterA[
        expectedValue-n]);
136   CGContextAddLineToPoint(context, xCenterA[expectedValue-n+1], yCenterA[
        expectedValue-n+1]);
137   CGContextStrokePath(context);
138   CGContextFlush(context);
139   drawImage.image = UIGraphicsGetImageFromCurrentImageContext();
140   UIGraphicsEndImageContext();
141 }
142
143 //funzione chiamata al termine di ogni audio (per noi l'intro)
```

```
144 -(void)audioPlayerDidFinishPlaying: (AVAudioPlayer *)player successfully: (
        BOOL)flag{
145   globalTimer = [NSTimer scheduledTimerWithTimeInterval:300.0 target:self
          selector:@selector(globalEnd) userInfo:NULL repeats:NO];
146   localTimer = [NSTimer scheduledTimerWithTimeInterval:10.0 target:self
          selector:@selector(helper) userInfo:NULL repeats:NO];
147   localTimerBool = TRUE;
148   [self.view setUserInteractionEnabled:TRUE];
149   [parent.view setUserInteractionEnabled:TRUE];
150 }
151
152 //controlla se la coordinata attuale e' in qualche cerchio
153 - (int) isInCircle:(float)x y:(float)y{
154   for (int i=0; i<8; i++) {
155     if (sqrt(pow(x-xCenterA[i], 2)+pow(y-yCenterA[i], 2))<=25){
156       return i+1;
157     }
158   }
159   return 0;
160 }
161
162 //riempie il cerchio con un colore
163 - (void) fillEllipse{
164   //disegno del cerchio pieno
165   UIGraphicsBeginImageContext(drawImage.frame.size);
166   [drawImage.image drawInRect:CGRectMake(0, 0, drawImage.frame.size.width,
          drawImage.frame.size.height)];
167   CGContextRef context = UIGraphicsGetCurrentContext();
168   CGContextSetLineWidth(context, 1.0);
169   CGContextSetRGBFillColor(context, 0.11, 0.89, 0.15, 0.5);
170   CGContextFillEllipseInRect(context, CGRectMake(xCenterA[expectedValue
          -1]-25, yCenterA[expectedValue-1]-25, 50, 50));
171   drawImage.image = UIGraphicsGetImageFromCurrentImageContext();
172   UIGraphicsEndImageContext();
173 }
174
175 - (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event {
176
177   UITouch *touch = [touches anyObject];
178   lastPoint = [touch locationInView:drawImage];
179
180   int whereIs = [self isInCircle:lastPoint.x y:lastPoint.y];
181   //if per l'inizio del tratto, else per i casi in cui ricomincia dopo aver
          alzato il dito
182   if (expectedValue==1){ //aspetto il primo tocco
183     if (whereIs==1) {  //inizio bene
184       NSLog(@"Ho iniziato");
185       //disegno del cerchio pieno
186       [self fillEllipse];
187       expectedValue++;
188       expectedZero=TRUE;
189       if (localTimerBool) {
190         localTimerBool = FALSE;
191         [localTimer invalidate];
192       }
193       localTimer = [NSTimer scheduledTimerWithTimeInterval:10.0 target:self
              selector:@selector(helper) userInfo:NULL repeats:NO];
194       localTimerBool = TRUE;
195     } else {          //inizio fuori dall'uno
196       NSLog(@"Ho iniziato sbagliato");
197       missNumber++;
198       [self playSound:1];
```

```
199       }
200     } else if (expectedValue>1) {
201       //controlliamo se ricomincia bene dopo aver alzato il dito
202       if (whereIs!=0 && expectedValue!=whereIs && whereIs!=expectedValue-1) {
203         //ho ricominciato sbagliato: da un numero che non e' ne' il precedente
                ne' quello atteso
204         NSLog(@"Ho ricominciato sbagliato");
205         missNumber++;
206         [self playSound:3];
207       } else if (expectedValue==whereIs) {
208         //ho ricominciato giusto: dal numero successivo a quello gia'
              attraversato
209         NSLog(@"Ho ricominciato giusto dal successivo");
210         //disegno del cerchio pieno
211         [self fillEllipse];
212         expectedValue++;
213         expectedZero=TRUE;
214         if (localTimerBool) {
215           localTimerBool = FALSE;
216           [localTimer invalidate];
217         }
218         localTimer = [NSTimer scheduledTimerWithTimeInterval:10.0 target:self
              selector:@selector(helper) userInfo:NULL repeats:NO];
219         localTimerBool = TRUE;
220       } else if (whereIs==expectedValue-1) {
221         //ho ricominciato giusto: dall'ultimo numero attravesato
222         expectedZero=TRUE;
223         NSLog(@"Ho ricominciato giusto dal precedente");
224         if (localTimerBool) {
225           localTimerBool = FALSE;
226           [localTimer invalidate];
227         }
228         localTimer = [NSTimer scheduledTimerWithTimeInterval:10.0 target:self
              selector:@selector(helper) userInfo:NULL repeats:NO];
229         localTimerBool = TRUE;
230       }
231     }
232   }
233
234   - (void)touchesMoved:(NSSet *)touches withEvent:(UIEvent *)event {
235
236     UITouch *touch = [touches anyObject];
237     CGPoint currentPoint = [touch locationInView:drawImage];
238
239     int whereIs = [self isInCircle:currentPoint.x y:currentPoint.y];
240     //controllo dove sono durante il tratto
241     if (expectedZero) { //se sono ancora dentro al cerchio di un numero
242       if (whereIs==0) {
243         NSLog(@"Sono uscito fuori dal numero");
244         expectedZero=FALSE;
245       }
246     } else {
247       if (whereIs==expectedValue) {  //se sono entrato nel cerchio giusto
248         NSLog(@"Mi sono mosso nel numero giusto");
249         //disegno del cerchio pieno
250         [self fillEllipse];
251         expectedValue++;
252         expectedZero=TRUE;
253         if (localTimerBool) {
254           localTimerBool = FALSE;
255           [localTimer invalidate];
256         }
```

```
257         localTimer = [NSTimer scheduledTimerWithTimeInterval:10.0 target:self
                selector:@selector(helper) userInfo:NULL repeats:NO];
258         localTimerBool = TRUE;
259       } else if (whereIs!=0) {   //se sono entrato nel cerchio sbagliato
260         NSLog(@"Mi sono mosso nel numero sbagliato");
261         missNumber++;
262         expectedZero=TRUE;
263         [self playSound:3];
264       }
265     }
266
267
268
269     //disegno del tratto
270     UIGraphicsBeginImageContext(drawImage.frame.size);
271     [drawImage.image drawInRect:CGRectMake(0, 0, drawImage.frame.size.width,
            drawImage.frame.size.height)];
272     CGContextRef context = UIGraphicsGetCurrentContext();
273     CGContextSetLineCap(context, kCGLineCapRound);
274     CGContextSetLineWidth(context, 3.0);
275     CGContextSetRGBStrokeColor(context, 0.0, 0.15, 0.58, 1.0);
276     CGContextBeginPath(context);
277     CGContextMoveToPoint(context, lastPoint.x, lastPoint.y);
278     CGContextAddLineToPoint(context, currentPoint.x, currentPoint.y);
279     CGContextStrokePath(context);
280     drawImage.image = UIGraphicsGetImageFromCurrentImageContext();
281     UIGraphicsEndImageContext();
282
283     lastPoint = currentPoint;
284   }
285
286   - (void)touchesEnded:(NSSet *)touches withEvent:(UIEvent *)event {
287     expectedZero=FALSE;
288     if (expectedValue==9){  //test finito
289       NSLog(@"Ho finito, errori:%d, ho alzato il dito %d volte",missNumber,
              liftFinger);
290       if (localTimerBool) {
291         [localTimer invalidate];
292       }
293       [globalTimer invalidate];
294       [self.view setUserInteractionEnabled:FALSE];
295       [parent nextTest];
296     } else {
297       //incremento errore dito alzato
298       NSLog(@"Ho alzato il dito prima di finire");
299       liftFinger++;
300       [self playSound:2];
301     }
302
303     //disegno del tratto
304     UIGraphicsBeginImageContext(drawImage.frame.size);
305     [drawImage.image drawInRect:CGRectMake(0, 0, drawImage.frame.size.width,
            drawImage.frame.size.height)];
306     CGContextRef context = UIGraphicsGetCurrentContext();
307     CGContextSetLineCap(context, kCGLineCapRound);
308     CGContextSetLineWidth(context, 3.0);
309     CGContextSetRGBStrokeColor(context, 0.0, 0.15, 0.58, 1.0);
310     CGContextMoveToPoint(context, lastPoint.x, lastPoint.y);
311     CGContextAddLineToPoint(context, lastPoint.x, lastPoint.y);
312     CGContextStrokePath(context);
313     CGContextFlush(context);
314     drawImage.image = UIGraphicsGetImageFromCurrentImageContext();
```

```
315   UIGraphicsEndImageContext();
316 }
317
318 - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)
        interfaceOrientation {
319     // Overriden to allow any orientation.
320     return NO;
321 }
322
323 - (void)dealloc {
324   [parent release];
325   [drawImage release];
326   [label1 release];
327   [label2 release];
328   [label3 release];
329   [label4 release];
330   [label5 release];
331   [label6 release];
332   [label7 release];
333   [label8 release];
334   [audioPlayer release];
335   [urlIntro release];
336   [urlStart release];
337   [urlFinger release];
338   [urlNumber release];
339   [urlHelper release];
340   [error release];
341   [super dealloc];
342 }
343
344 @end
```

```
1  //
2  //  TrailMakingTest25A.h
3  //
4
5  #import <UIKit/UIKit.h>
6  #import <AVFoundation/AVFoundation.h>
7  #import "Trail_Making_Test_AViewController.h"
8  #import "iOS2Jolie.h"
9
10 @interface TrailMakingTestA25 : UIViewController <AVAudioPlayerDelegate>{
11   Trail_Making_Test_AViewController *parent;
12   CGPoint lastPoint;
13   UIImageView *drawImage25;
14   NSTimer *timerStop25;
15   int expectedValue25; //il prossimo valore da toccare
16   int liftFinger25;  //il numero di volte che il dito e' stato alzato
17   int missNumber25;  //il numero di errori (numero sbagliato)
18   int remainingTime;
19   BOOL expectedZero25; //indica che abbiamo toccato un numero e dobbiamo
         uscire dal suo cerchio
20   UIImage *screenShotTop;
21   NSURL *url;
22   NSError *error;
23   AVAudioPlayer *audioPlayer25;
24     iOS2Jolie *ios2jolie;
25 }
26
27 @property (nonatomic, retain) Trail_Making_Test_AViewController *parent;
```

```
28 @property (nonatomic, retain) UIImageView *drawImage25;
29 @property (nonatomic, retain) NSTimer *timerStop25;
30 @property int expectedValue25;
31 @property int liftFinger25;
32 @property int missNumber25;
33 @property int remainingTime;
34 @property BOOL expectedZero25;
35 @property (nonatomic, retain) UIImage *screenShotTop;
36 @property (nonatomic, retain) NSURL *url;
37 @property (nonatomic, retain) NSError *error;
38 @property (nonatomic, retain) AVAudioPlayer *audioPlayer25;
39
40 @end
```

```
1  //
2  //  TrailMakingTestA25.m
3  //
4
5  #import "TrailMakingTestA25.h"
6  #import <QuartzCore/QuartzCore.h>
7
8  @implementation TrailMakingTestA25
9
10 @synthesize drawImage25;
11 @synthesize timerStop25;
12 @synthesize parent;
13 @synthesize expectedValue25;
14 @synthesize liftFinger25;
15 @synthesize missNumber25;
16 @synthesize remainingTime;
17 @synthesize expectedZero25;
18 @synthesize screenShotTop;
19 @synthesize url;
20 @synthesize error;
21 @synthesize audioPlayer25;
22
23 //centri dei cerchi
24 float xCenterA25[] = {529,437,556,586,321,483,341,202,237,298,429, 99,175,
       79,110,195,329,394,556,452,632,694,686,663,556};
25 float yCenterA25[] =
       {475,620,735,317,364,393,483,637,803,650,910,923,419,517, 99,223,
       82,240,206,133, 86,261,914,504,846};
26
27 // Implement viewDidLoad to do additional setup after loading the view,
       typically from a nib.
28 - (void)viewDidLoad {
29   //setto le variabili globali
30   expectedValue25 = 1; //il prossimo valore da toccare
31   liftFinger25 = 0;  //il numero di volte che il dito e' stato alzato
32   missNumber25 = 0;  //il numero di errori (numero sbagliato)
33   remainingTime = 0;
34   expectedZero25 = FALSE; //indica che abbiamo toccato un numero e dobbiamo
         uscire dal suo cerchio
35     //alloco i servizi per l'invio dei dati
36     ios2jolie = [[iOS2Jolie alloc] init];
37     ios2jolie.serverUrl = parent.serverUrl;
38     [super viewDidLoad];
39   //salvo l'immagine della vista iniziale (con sfondo bianco)
40   self.view.backgroundColor = [UIColor whiteColor];
41   UIGraphicsBeginImageContext(self.view.frame.size);
```

```
42    [self.view.layer renderInContext:UIGraphicsGetCurrentContext()];
43    screenShotTop = [[UIImage alloc] initWithData:[NSData dataWithData:
          UIImagePNGRepresentation(UIGraphicsGetImageFromCurrentImageContext())
          ]];
44    UIGraphicsEndImageContext();
45    self.view.backgroundColor = [UIColor colorWithRed:0.76 green:1.0 blue:0.93
           alpha:1.0];
46    //creo la view in cui disegno
47    drawImage25 = [[UIImageView alloc] initWithFrame:CGRectMake(0, 0, 768,
          1024)];
48    //funzioni di disegno
49    //definisce il context in cui disegnare
50    UIGraphicsBeginImageContext(drawImage25.frame.size);
51    //delimita l'area in cui disegnare
52    [drawImage25.image drawInRect:CGRectMake(0, 0, drawImage25.frame.size.
          width, drawImage25.frame.size.height)];
53    CGContextRef context = UIGraphicsGetCurrentContext();
54    CGContextSetLineWidth(context, 1.0);
55    //Crea i cerchi
56    CGContextAddEllipseInRect(context, CGRectMake(504, 450, 50, 50));
57    CGContextAddEllipseInRect(context, CGRectMake(412, 595, 50, 50));
58    CGContextAddEllipseInRect(context, CGRectMake(531, 710, 50, 50));
59    CGContextAddEllipseInRect(context, CGRectMake(561, 292, 50, 50));
60    CGContextAddEllipseInRect(context, CGRectMake(296, 339, 50, 50));
61    CGContextAddEllipseInRect(context, CGRectMake(458, 368, 50, 50));
62    CGContextAddEllipseInRect(context, CGRectMake(316, 458, 50, 50));
63    CGContextAddEllipseInRect(context, CGRectMake(177, 612, 50, 50));
64    CGContextAddEllipseInRect(context, CGRectMake(212, 778, 50, 50));
65    CGContextAddEllipseInRect(context, CGRectMake(273, 625, 50, 50));
66    CGContextAddEllipseInRect(context, CGRectMake(404, 885, 50, 50));
67    CGContextAddEllipseInRect(context, CGRectMake( 74, 898, 50, 50));
68    CGContextAddEllipseInRect(context, CGRectMake(150, 394, 50, 50));
69    CGContextAddEllipseInRect(context, CGRectMake( 54, 492, 50, 50));
70    CGContextAddEllipseInRect(context, CGRectMake( 85,  74, 50, 50));
71    CGContextAddEllipseInRect(context, CGRectMake(170, 198, 50, 50));
72    CGContextAddEllipseInRect(context, CGRectMake(304,  57, 50, 50));
73    CGContextAddEllipseInRect(context, CGRectMake(369, 215, 50, 50));
74    CGContextAddEllipseInRect(context, CGRectMake(531, 181, 50, 50));
75    CGContextAddEllipseInRect(context, CGRectMake(427, 108, 50, 50));
76    CGContextAddEllipseInRect(context, CGRectMake(607,  61, 50, 50));
77    CGContextAddEllipseInRect(context, CGRectMake(669, 236, 50, 50));
78    CGContextAddEllipseInRect(context, CGRectMake(661, 889, 50, 50));
79    CGContextAddEllipseInRect(context, CGRectMake(638, 479, 50, 50));
80    CGContextAddEllipseInRect(context, CGRectMake(531, 821, 50, 50));
81    CGContextDrawPath(context, kCGPathStroke);
82    //prende la view prima di disegnarci sopra
83    drawImage25.image = UIGraphicsGetImageFromCurrentImageContext();
84    //disegna le nuove linee
85    UIGraphicsEndImageContext();
86    //mostra la vista in cui disegno
87    [self.view addSubview:drawImage25];
88    //disattivo il touch
89    [self.view setUserInteractionEnabled:FALSE];
90    //play della spiegazione introduttiva
91    url = [NSURL fileURLWithPath:[NSString stringWithFormat:@"%@/tmta.mp3", [[
          NSBundle mainBundle] resourcePath]]];
92    audioPlayer25 = [[AVAudioPlayer alloc] initWithContentsOfURL:url error:&
          error];
93    audioPlayer25.numberOfLoops=0;
94    audioPlayer25.delegate=self;
95    NSLog(@"Play intro");
96    [audioPlayer25 play];
```

```
97  }
98
99  //chiamata alla fine del test (anche nel caso di tempo terminato)
100 - (void)finishTest{
101   NSLog(@"Ho finito, errori:%d, ho alzato il dito %d volte",missNumber25,
          liftFinger25);
102   //salvo nelle immagini
103   UIGraphicsBeginImageContext(drawImage25.frame.size);
104   [drawImage25.layer renderInContext:UIGraphicsGetCurrentContext()];
105   UIImage *screenShot = UIGraphicsGetImageFromCurrentImageContext();
106   //aggiungo il layer sottostante
107   [screenShotTop drawAtPoint:CGPointMake(0, 0)];
108   [screenShot drawAtPoint:CGPointMake(0, 0)];
109   [drawImage25.layer renderInContext:UIGraphicsGetCurrentContext()];
110   UIImage *screenShotOut = UIGraphicsGetImageFromCurrentImageContext();
111   UIGraphicsEndImageContext();
112   //aggiungiamo la label sulla image
113   CGColorSpaceRef colorSpace = CGColorSpaceCreateDeviceRGB();
114     CGContextRef context = CGBitmapContextCreate(NULL, screenShotOut.size.
          width, screenShotOut.size.height, 8, 4 * screenShotOut.size.width,
          colorSpace, kCGImageAlphaPremultipliedFirst);
115     CGContextDrawImage(context, CGRectMake(0, 0, screenShotOut.size.width,
          screenShotOut.size.height), screenShotOut.CGImage);
116     CGContextSelectFont(context, "Times New Roman", 30, kCGEncodingMacRoman)
          ;
117     CGContextSetTextDrawingMode(context, kCGTextFill);
118     CGContextSetRGBFillColor(context, 0, 0, 0, 1);
119   char infoImage[40];
120   char nomecognome[50];
121   NSDateFormatter *today = [[NSDateFormatter alloc] init];
122   [today setDateFormat:@"dd/MM/yyyy"];
123   sprintf(nomecognome, "%s" ,[parent.username.text UTF8String]);
124   sprintf(infoImage,"data: %s - tempo: %d sec",[[today stringFromDate:[
          NSDate date]] UTF8String],remainingTime);
125   CGContextShowTextAtPoint(context, 20, 50, nomecognome, strlen(nomecognome)
          );
126   CGContextShowTextAtPoint(context, 20, 20, infoImage, strlen(infoImage));
127     CGImageRef imageMasked = CGBitmapContextCreateImage(context);
128     CGContextRelease(context);
129     CGColorSpaceRelease(colorSpace);
130   //salviamo nell'album o sul server
131     if (parent.isExample) {
132         UIImageWriteToSavedPhotosAlbum([UIImage imageWithCGImage:imageMasked
              ], self, nil, nil);
133     } else {
134         [ios2jolie trailMakingTestServer:parent.username.text data:parent.
              todayDate type:@"A" seconds:remainingTime numberError:
              missNumber25 fingerError:liftFinger25 imageData:
              UIImageJPEGRepresentation([UIImage imageWithCGImage:imageMasked
              ], 0.85)];
135     }
136   //carico il prossimo test o termino
137   [self.view setUserInteractionEnabled:FALSE];
138   if (remainingTime == 0) {
139     [parent endTest];
140   } else {
141     [parent nextTest];
142   }
143 }
144
145 //funzione chiamata alla fine di ogni audio (per noi l'intro)
```

```
146 -(void)audioPlayerDidFinishPlaying: (AVAudioPlayer *)player successfully: (
        BOOL)flag{
147   //aggiungo un timer per il limite di tempo
148   timerStop25 = [NSTimer scheduledTimerWithTimeInterval:240.0 target:self
          selector:@selector(finishTest) userInfo:NULL repeats:NO];
149   [self.view setUserInteractionEnabled:TRUE];
150   [parent.view setUserInteractionEnabled:TRUE];
151 }
152
153 //controlla se la coordinata attuale e' in qualche cerchio
154 - (int) isInCircle25:(float)x y:(float)y{
155   for (int i=0; i<25; i++) {
156     if (sqrt(pow(x-xCenterA25[i], 2)+pow(y-yCenterA25[i], 2))<=25){
157       return i+1;
158     }
159   }
160   return 0;
161 }
162
163 - (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event {
164
165   UITouch *touch = [touches anyObject];
166   lastPoint = [touch locationInView:drawImage25];
167
168   int whereIs = [self isInCircle25:lastPoint.x y:lastPoint.y];
169   //if per l'inizio del tratto, else per i casi in cui ricomincia dopo aver
          alzato il dito
170   if (expectedValue25==1){ //aspetto il primo tocco
171     if (whereIs==1) {  //inizio bene
172       NSLog(@"Ho iniziato");
173       expectedValue25++;
174       expectedZero25=TRUE;
175     } else {          //inizio fuori dall'uno
176       NSLog(@"Ho iniziato sbagliato");
177       missNumber25++;
178     }
179   } else if (expectedValue25>1) {
180     //controlliamo se ricomincia bene dopo aver alzato il dito
181     if (whereIs!=0 && expectedValue25!=whereIs && whereIs!=expectedValue25
          -1) {
182       //ho ricominciato sbagliato: da un numero che non e' ne' il precedente
            ne' quello atteso
183       NSLog(@"Ho ricominciato sbagliato");
184       missNumber25++;
185     } else if (expectedValue25==whereIs) {
186       //ho ricominciato giusto: dal numero successivo a quello gia'
            attraversato
187       NSLog(@"Ho ricominciato giusto dal successivo");
188       expectedValue25++;
189       expectedZero25=TRUE;
190     } else if (whereIs==expectedValue25-1) {
191       //ho ricominciato giusto: dall'ultimo numero attravesato
192       expectedZero25=TRUE;
193       NSLog(@"Ho ricominciato giusto dal precedente");
194     }
195   }
196 }
197
198 - (void)touchesMoved:(NSSet *)touches withEvent:(UIEvent *)event {
199
200   UITouch *touch = [touches anyObject];
201   CGPoint currentPoint = [touch locationInView:drawImage25];
```

```
202
203     int whereIs = [self isInCircle25:currentPoint.x y:currentPoint.y];
204     //controllo dove sono durante il tratto
205     if (expectedZero25) { //se sono ancora dentro al cerchio di un numero
206       if (whereIs==0) {
207         NSLog(@"Sono uscito fuori dal numero");
208         expectedZero25=FALSE;
209       }
210     } else {
211       if (whereIs==expectedValue25) {  //se sono entrato nel cerchio giusto
212         NSLog(@"Mi sono mosso nel numero giusto");
213         expectedValue25++;
214         expectedZero25=TRUE;
215       } else if (whereIs!=0) {   //se sono entrato nel cerchio sbagliato
216         NSLog(@"Mi sono mosso nel numero sbagliato");
217         missNumber25++;
218         expectedZero25=TRUE;
219       }
220     }
221
222     //disegno del tratto
223     UIGraphicsBeginImageContext(drawImage25.frame.size);
224     [drawImage25.image drawInRect:CGRectMake(0, 0, drawImage25.frame.size.
            width, drawImage25.frame.size.height)];
225     CGContextRef context = UIGraphicsGetCurrentContext();
226     CGContextSetLineCap(context, kCGLineCapRound);
227     CGContextSetLineWidth(context, 3.0);
228     CGContextSetRGBStrokeColor(context, 0.0, 0.15, 0.58, 1.0);
229     CGContextBeginPath(context);
230     CGContextMoveToPoint(context, lastPoint.x, lastPoint.y);
231     CGContextAddLineToPoint(context, currentPoint.x, currentPoint.y);
232     CGContextStrokePath(context);
233     drawImage25.image = UIGraphicsGetImageFromCurrentImageContext();
234     UIGraphicsEndImageContext();
235
236     lastPoint = currentPoint;
237   }
238
239   - (void)touchesEnded:(NSSet *)touches withEvent:(UIEvent *)event {
240     expectedZero25=FALSE;
241     if (expectedValue25==26){  //test finito
242       //calcolo il tempo impiegato per effettuare il test
243       NSDate *clockDate = [timerStop25 fireDate];
244       //invalido il timer
245       [timerStop25 invalidate];
246       NSDate *currentDate = [NSDate date];
247       NSCalendar *calendar = [[[NSCalendar alloc] initWithCalendarIdentifier:
              NSGregorianCalendar] autorelease];
248       NSDateComponents *components = [calendar components:NSSecondCalendarUnit
               fromDate:currentDate toDate:clockDate options:0];
249       remainingTime = 240 - components.second;
250       [self finishTest];
251     } else {
252       //incremento errore dito alzato
253       NSLog(@"Ho alzato il dito prima di finire");
254       liftFinger25++;
255     }
256
257     //disegno del tratto
258     UIGraphicsBeginImageContext(drawImage25.frame.size);
259     [drawImage25.image drawInRect:CGRectMake(0, 0, drawImage25.frame.size.
            width, drawImage25.frame.size.height)];
```

```
260     CGContextRef context = UIGraphicsGetCurrentContext();
261     CGContextSetLineCap(context, kCGLineCapRound);
262     CGContextSetLineWidth(context, 3.0);
263     CGContextSetRGBStrokeColor(context, 0.0, 0.15, 0.58, 1.0);
264     CGContextMoveToPoint(context, lastPoint.x, lastPoint.y);
265     CGContextAddLineToPoint(context, lastPoint.x, lastPoint.y);
266     CGContextStrokePath(context);
267     CGContextFlush(context);
268     drawImage25.image = UIGraphicsGetImageFromCurrentImageContext();
269     UIGraphicsEndImageContext();
270 }
271
272 - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)
        interfaceOrientation {
273     // Overriden to allow any orientation.
274     return NO;
275 }
276
277 - (void)dealloc {
278     [parent release];
279     [drawImage25 release];
280     [timerStop25 release];
281     [screenShotTop release];
282     [url release];
283     [error release];
284     [audioPlayer25 release];
285     [super dealloc];
286 }
287
288 @end
```

Bibliografia

[1] Umiltà C., (1999), *Manuale di neuroscienze*, Bologna, il Mulino

[2] Vallar G., Papagno C., (2007), *Manuale di neuropsicologia*, Bologna, il Mulino

[3] Montesi F., (2010) *JOLIE: a Service-oriented Programming Language*, Bologna

[4] JOLIE: Java Orchestration Language Interpreter Engine, http://www.jolie-lang.org/

[5] JOLIE, Language Tutorials, http://www.jolie-lang.org/language_tutorials.php

[6] Apple iPad 2, http://www.apple.com/ipad/

[7] Apple iOS, http://developer.apple.com/technologies/ios/

[8] Apple Developer, http://developer.apple.com/

[9] Apple Xcode, http://developer.apple.com/xcode/index.php

[10] Apple Cocoa Touch, http://developer.apple.com/technologies/ios/cocoa-touch.html

[11] Cox B., Objective-C Language, http://virtualschool.edu/objectivec/

[12] Objective-C documentation, http://developer.apple.com/library/mac/ /documentation/Cocoa/Conceptual/ObjectiveC/ObjC.pdf

[13] Oracle MySQL, http://www.mysql.com

[14] Apache XAMPP, http://www.apachefriends.org/it/xampp.html

[15] HTTP, http://www.w3.org/Protocols/

[16] XML, http://www.w3.org/XML/