Alma Mater Studiorum - Università degli Studi di Bologna
Campus di Cesena

SCUOLA DI SCIENZE

Corso di Laurea in Ingegneria e Scienze Informatiche

WEBAPP PER L'APPRENDIMENTO DELLA MATEMATICA ATTRAVERSO CRUCIVERBA NUMERICO

Tesi di Laurea in:
Sistemi Multimediali

Presentata da:
Eugenio Severi

Relatore:
Chiar.ma Prof.ssa Paola Salomoni

Sessione II
Anno Accademico 2014/2015

Indice

INDICE ..... **I**

INTRODUZIONE **1**

1 IL GAMING NELLA DIDATTICA **5**

1.1 LA GAMIFICATION 5
1.1.1 Cenni storici *6*
1.1.2 Meccaniche della Gamification *7*
1.1.3 Applicazioni della Gamification *8*
1.2 MONETIZZAZIONE 11
1.3 SERIOUS GAMES 12
1.3.1 Cenni storici *14*
1.3.2 Pedagogia dei serious games *15*
1.3.3 Applicazioni dei serious games *15*
1.4 SMARTPHONE, TABLET E SOCIAL NETWORK 16
1.4.1 L'espansione del pubblico *18*
1.5 PRIVACY E SICUREZZA 19

2 PROGETTO: SPECIFICHE, ARCHITETTURA E TECNOLOGIE COINVOLTE 23

2.1 INTRODUZIONE 23
2.2 MARKET SHARE DEI SISTEMI OPERATIVI E VERSIONI SUPPORTATE 24
2.3 ANALISI DELLE TECNOLOGIE DI SVILUPPO NATIVE 25
2.4 WEBAPP 27
2.5 APPLICAZIONI IBRIDE 30
2.5.1 Apache Cordova *31*
2.6 SCELTA PER IL PROGETTO 33
2.7 SPECIFICHE COMUNI A TUTTI I GIOCHI 34
2.8 ARCHITETTURA DI RETE DELL'APPLICAZIONE 39

3 IMPLEMENTAZIONE 41

3.1 PROGETTO IMPLEMENTATO 41
3.2 ALGORITMO DI GENERAZIONE CRUCIVERBA 42

3.2.1 Ottimizzazioni ed efficienza 43
3.2.2 Test dell'algoritmo 46
3.2.2.1 Test numero 1 46
3.2.2.2 Test numero 2 48
3.2.2.3 Test numero 3 51
3.3 GESTIONE DELLE DEFINIZIONI 52
3.3.1 Efficienza nell'archiviazione e generazione delle definizioni 53
3.3.2 Formato di archiviazione, prestazioni e sicurezza 54
3.4 DIFFICOLTÀ DI GIOCO 55
3.5 INTERFACCIA GRAFICA E CASI D'USO 56
3.5.1 Interfaccia grafica 1: dimostrazione 57
3.5.2 Interfaccia grafica 2: vista utente 58

CONCLUSIONI 61

BIBLIOGRAFIA I

Introduzione

Lo studio della matematica nelle scuole ha sempre rivestito un ruolo di primaria importanza nella formazione dei giovani [LAF14]. Questa, infatti, da un lato trova applicazione nella vita di tutti i giorni, dall'altro è la porta di accesso per molti studi avanzati di livello universitario. Ciò nonostante, specialmente negli ultimi decenni, si è assistito ad una sorta di disaffezione di massa da parte dei giovani studenti nei confronti dello studio della matematica [VAL15].

Alcuni dati concreti riguardanti questo fenomeno provengono dal numero di nuove iscrizioni alle facoltà scientifiche, in progressiva diminuzione nel corso degli anni, soprattutto per quanto riguarda Matematica [GEN05].

La causa non è di immediata individuazione, anche se alcuni studi suggeriscono che sia da ricercare nell'aumento frenetico dei ritmi di vita, che colpisce soprattutto i giovani e giovanissimi: questi, infatti, a causa delle molte attività che svolgono, starebbero progressivamente perdendo la capacità di concentrarsi a lungo su un problema e di ragionare per trovare una soluzione [INT01]. Oltre a questo aspetto concorre anche l'abuso o uso problematico della tecnologia, che da un lato fornisce risposte veloci a problemi di varia natura, mentre dall'altro disincentiva chi la utilizza a fare affidamento sulle proprie forze e costituisce una fonte di distrazione. In particolare la recente diffusione di smartphone anche tra i giovanissimi porta facilmente a situazioni di uso problematico di Internet e delle tecnologie digitali [PAN01].

Al di là della tecnologia, alcuni studi evidenziano come questa disaffezione sia causata dalla percezione di un senso di inadeguatezza verso la materia, correlata all’esistenza di preconcetti [GEN05].

In questa sede tuttavia non si desidera individuare le ragioni e le cause che hanno portato a questo fenomeno, quanto piuttosto di valutare e implementare

meccanismi per riavvicinare i ragazzi alla matematica.

Per raggiungere questo obiettivo si è pensato di sfruttare la passione dei giovani per le nuove tecnologie, indirizzandola verso lo studio della matematica. Per questo motivo si è deciso di utilizzare la presenza pervasiva di smartphone, tablet e computer nella vita dei ragazzi come piattaforma interattiva target sulla quale sviluppare. Inoltre, poiché molti giochi sono basati sulla matematica e/o richiedono una conoscenza della matematica più o meno avanzata, si è pensato di veicolare la matematica attraverso giochi. Questo concetto si avvicina a quelli di "Gamification" e "Serious games", che saranno descritti nel dettaglio nel corso della tesi.

Alla luce della problematica individuata, nell'ambito di questa tesi si cercherà di realizzare un'applicazione di gioco finalizzata al riavvicinamento dei giovani al mondo della matematica. In particolare, verrà realizzato un cruciverba numerico, ovvero un classico cruciverba all'interno del quale non sono presenti le lettere dell'alfabeto, ma cifre decimali. Inoltre, le definizioni saranno costituite da calcoli matematici da svolgere, più o meno complessi, basati sui programmi scolastici.

La tesi si inserisce all'interno di un progetto più ampio, che prevede la realizzazione di una serie di giochi matematici, racchiusi all'interno di un'applicazione contenitore che ne coordinerà le attività. In questa sede, si implementerà una WebApp, ovvero un'applicazione realizzata tramite tecnologie web ed eseguibile all'interno di un browser, all'interno della quale sarà implementato uno dei giochi, che successivamente verrà integrato nella macro-applicazione, che sarà realizzata in seguito. La WebApp sarà realizzata in ambiente Apache Cordova, al fine di creare un'applicazione ibrida installabile sui dispositivi degli utenti.

La scelta di implementare l'applicazione tramite tecnologie web piuttosto che native delle singole piattaforme è giustificata dall'esigenza di massimizzare la portabilità dell'applicazione sui molteplici dispositivi posseduti dagli utenti, siano questi computer desktop o portatili, siano questi dispositivi mobili come smartphone e tablet, in maniera indipendente dal sistema operativo utilizzato.

La tesi è strutturata in tre capitoli, contenenti gli argomenti secondo la seguente suddivisione.

- Nel primo capitolo sarà trattato l'utilizzo dei videogiochi nella didattica e verranno presentate e descritte le principali tipologie di gioco legate a questo ambito: in particolare si farà inizialmente riferimento alle meccaniche di Gamification e al loro impiego, e successivamente ai "serious games", con attenzione alle sfumature tra queste due categorie. Si parlerà inoltre della monetizzazione all'interno dei giochi ad opera del produttore, evidenziando come in alcuni casi queste scelte possano diventare controproducenti ai fini del successo e della diffusione dei giochi stessi. Si passerà poi ad un'analisi dei dispositivi mobili, la cui diffusione è notevolmente aumentata nel corso degli ultimi anni, che si candidano a piattaforme di gioco per il "casual gaming", avvicinando al mondo dei videogiochi anche persone che precedentemente ne erano estranee. Infine, si descriveranno le problematiche riguardanti privacy e sicurezza, con particolare riferimento ai dispositivi mobili, sensibilmente meno sicuri rispetto ai computer, e allo stesso tempo ancora più pericolosi nella pratica, essendo utilizzati con leggerezza da qualunque tipologia di utente, non necessariamente sensibile alle tematiche di sicurezza e protezione dei dati.
- Nel secondo capitolo si passerà ad un'analisi delle specifiche generali di progetto, comuni non solo all'applicazione realizzata nell'ambito di questa tesi, ma anche a tutte le altre nell'ambito di questo progetto. Verranno anche dettagliatamente descritti i processi che hanno portato alla definizione di tali specifiche. Si inizierà con un'analisi dei sistemi operativi desktop e mobili presenti sul mercato, e se ne individueranno le percentuali di diffusione. Successivamente verranno analizzate le varie piattaforme e linguaggi di sviluppo nativi all'interno di ognuno di questi sistemi. Si passerà poi alla considerazione delle WebApp, in quanto si propongono l'obiettivo di essere compatibili con ogni piattaforma. Dopodiché, si studieranno le applicazioni ibride come via intermedia tra le applicazioni native e le WebApp, in quanto raccolgono aspetti positivi sia dalle une che dalle altre, con particolare riferimento al framework Apache Cordova come caso di studio. Alla luce delle analisi svolte e in base ai requisiti, si definiranno delle specifiche comuni per il

progetto, scegliendo le tecnologie più adatte. Infine, verrà proposta una possibile architettura di rete dell’applicazione, basata sull’utilizzo di un server centrale.

- Nel terzo capitolo verrà descritta l’implementazione vera e propria del progetto di questa specifica tesi, ovvero il cruciverba numerico. Nello specifico si descriverà il funzionamento dell’algoritmo di generazione dei cruciverba, con particolare attenzione riguardo agli aspetti legati all’efficienza e alla rapidità di esecuzione, giustificando le scelte implementative. Questo aspetto è particolarmente importante specialmente sui dispositivi a prestazioni ridotte, come gli smartphone di fascia bassa. Saranno poi eseguiti alcuni test per valutare prestazioni dell’algoritmo e qualità degli schemi generati. Si passerà quindi all’analisi delle possibili alternative riguardanti la gestione e la generazione delle definizioni da inserire all’interno dei cruciverba creati, anche qui con particolare riferimento all’efficienza. In seguito, verranno descritte le meccaniche di gioco, derivanti dal mondo della Gamification, che consentiranno di creare livelli di difficoltà multipli, variando alcuni parametri di generazione dei cruciverba. Infine, verranno mostrati due casi d’uso dell’applicazione, e verrà presentata l’interfaccia grafica.

1 Il gaming nella didattica

In questo capitolo saranno descritti l'ambito del progetto realizzato e le strategie adottate per far fronte alle necessità evidenziate nelle specifiche. Verranno illustrati i concetti di Gamification e di “serious game”; si parlerà poi del fenomeno dell’esplosione dei dispositivi mobili negli ultimi anni e se ne analizzeranno le potenzialità. Infine, si parlerà di alcuni aspetti legati alla sicurezza, con particolare riferimento ai dispositivi mobili.

1.1 La Gamification

Per Gamification (termine coniato dal game designer Jesse Schell nel 2010 [VIO11]) si intende l’utilizzo di elementi derivanti dal mondo del gioco al di fuori di questo contesto [DET11], con il fine di trasformare un'attività utile, ma di per sé noiosa, in un'attività divertente, introducendo l'elemento gioco. In questo modo gli utenti sono incentivati a svolgere volentieri una stessa attività che fino a prima tendevano ad evitare. In figura 1.1 è presente uno dei loghi relativi alla Gamification.

Le tecniche di Gamification possono essere utilizzate per scopi socialmente utili, come ad esempio mappare l'accessibilità urbana. In questo caso l'obiettivo è quello di coinvolgere nel progetto il maggior numero di persone possibili, anche se non tutte sono direttamente influenzate dal risultato finale. Ad esempio una persona ipovedente è già interessata direttamente dalla presenza di semafori acustici in città, mentre tutte le altre persone non hanno un riscontro diretto dei benefici derivanti dall'introduzione di semafori di questo tipo. Tuttavia, attraverso l'uso di tecniche di Gamification è possibile trasformare in un gioco l'attività di mappatura dei semafori, in modo tale da estendere l'utenza dell'applicazione al di fuori dei confini specifici del

dominio di partenza.

Figura 1.1 - Logo Gamification

Uno studio del 2011 di Jane McGonigal sostiene l'efficacia dell'introduzione dell'elemento gioco al fine di sensibilizzare l'utenza verso problematiche sociali [FAT12]. Ulteriori studi, realizzati attraverso l’analisi dei dati generati dai software di gioco relativi al comportamento degli utenti internamente al gioco stesso, hanno dimostrato come l’impiego di meccaniche di videogioco si riflettano sugli utenti sviluppando un comportamento attivo, che è preferibile ad uno puramente passivo, in quanto il messaggio veicolato viene associato dall’utente all’azione stessa, venendo racchiuso nel contesto dell’esperienza di gioco. É possibile utilizzare la Gamification anche per fini commerciali, con lo scopo di fidelizzare i clienti ad uno specifico marchio e di profilarli, consentendo analisi di mercato finalizzate all'individuazione di target e tendenze [ALI10].

In generale la Gamification è uno strumento utile ogni volta che si vuole veicolare un messaggio, sia di pubblico interesse, sia commerciale. A questo proposito, i primi utilizzi del gioco applicato alla vita quotidiana sono state individuate negli anni 80 sia da soggetti pubblici (statali), sia privati (aziende che desiderano pubblicizzare e diffondere i propri prodotti) [VIL11].

1.1.1 Cenni storici

Gli aspetti di fidelizzazione del cliente della Gamification affondano le loro

radici alla fine del Settecento, quando comparvero i primi esempi storici di fidelizzazione. Alcuni negozianti americani introdussero dei gettoni di rame, ovvero una moneta virtuale, utilizzabile per acquistare prodotti reali presso quello specifico negozio, in modo da premiare i clienti che sceglievano di fare acquisti in quel negozio a scapito di altri. Questo sistema divenne sempre più articolato, fino ad arrivare alle raccolte punti e ai *Frequent Flyer Program* degli anni ottanta del Novecento, per la realizzazione dei quali furono coinvolti anche dei veri game designer. Infine, negli anni novanta, con il diffondersi dei videogiochi anche in ambiente domestico, si poté assistere ai primi esempi di in-game advertising, principalmente ad opera di multinazionali che inserivano i loro marchi all'interno dei giochi in cambio di sponsorizzazioni.

1.1.2 Meccaniche della Gamification

Le meccaniche sfruttate nella Gamification derivano dal mondo dei videogiochi, ed ereditano le innovazioni di quel settore [ALT10]. Quelli principali sono:

- un sistema di gioco organizzato in livelli, eventualmente con difficoltà crescente, che rappresentano dei traguardi da raggiungere;
- l'assegnazione di un punteggio in base ai risultati ottenuti nelle partite, al quale può essere legato il raggiungimento di traguardi specifici;
- l'attribuzione di premi speciali virtuali al conseguimento di specifici traguardi, come ad esempio il raggiungimento di un determinato punteggio o livello, attraverso i quali l'utente può personalizzare e potenziare l'esperienza di gioco;
- la competizione con gli altri utenti, attraverso un sistema di confronto del punteggio e dei premi ottenuti.

In particolare, l'utente è motivato ad incrementare il proprio punteggio compiendo le necessarie azioni in gioco, anche se il punteggio è puramente virtuale e non associato ad un valore economico. Il punteggio può essere considerato come una valuta interna al gioco che l'utente può spendere per acquistare benefici all'interno del gioco, che possono essere funzionali all'avanzamento nel gioco (potenziamenti) oppure puramente estetici (temi grafici e oggetti da indossare). Al raggiungimento di ogni livello, inoltre, il giocatore vede il proprio profilo arricchito con badge che testimoniano l'avvenuto progresso, aumentando la competizione con gli altri giocatori. Si viene a creare una classifica degli utenti, che ogni giocatore è spinto a scalare.

Possono essere presenti più classifiche basate su parametri diversi, come ad esempio il punteggio o il livello raggiunto, la quantità di tempo trascorsa all'interno del gioco, la rapidità nel raggiungere i traguardi, e così via. Il desiderio di prevalere sugli altri giocatori costituisce una forte spinta motivazionale all'utilizzo del gioco. Infine, la compravendita di oggetti virtuali interna al gioco motiva l'utente a investire ulteriori energie al fine di acquisire un particolare bene.

Grazie ad un’efficace implementazione di queste meccaniche l'utente è incentivato a proseguire nel gioco per il gioco stesso, a prescindere dallo scopo reale per cui è stato creato, dando origine ad un circolo virtuoso che porta beneficio sia alla causa che ha portato alla creazione del gioco, sia all'utente che si diverte. È provata inoltre l'esistenza di un meccanismo psicologico che associa la sensazione di soddisfazione dell'utente durante il gioco e le azioni che compie fisicamente, tale per cui inconsciamente l'utente è portato a considerare positiva l'azione che compie, anche al di fuori del gioco stesso [ALI10]. Riprendendo l'esempio precedente sull'accessibilità urbana, l'utente che utilizza il gioco viene maggiormente sensibilizzato riguardo alle problematiche che questo tenta di risolvere. Un ulteriore esempio può essere legato alla sensibilizzazione dei cittadini sul tema della raccolta differenziata: attraverso l'esperienza del gioco l'utente sarà più incline a metterla in pratica, non vivendola quindi come una costrizione, ma come un gesto spontaneo, avendolo interiorizzato.

1.1.3 Applicazioni della Gamification

Un primo esempio di applicazione del concetto di Gamification è stato presentato nella prima parte del paragrafo 1.1. Ad ogni modo, essa si presta particolarmente non solo al settore dell’urbanizzazione e dell’accessibilità cittadina da parte di chiunque, ma più in generale a una serie di ambiti meno specifici, che possono trovare applicazioni particolari a seconda dei singoli casi che si presentano, quali ad esempio:

- E-Commerce: secondo uno studio molti videogiocatori tendono ad essere anche utenti di negozi online. In particolare i giocatori hanno il 20% di probabilità in più di essere anche utenti di e-commerce rispetto alle persone non giocatrici. Non solo: tra la totalità degli utenti e-commerce, i giocatori tendono ad acquistare di più, sia come quantità, sia come frequenza. Per questo è possibile applicare l’utilizzo di meccaniche di gioco allo shopping, al fine di motivare

ulteriormente l'utente a fare acquisti. Infine, lo stesso studio afferma che i giocatori abituali hanno mediamente il 50% di probabilità in più di convincere i loro conoscenti non giocatori quando si tratta di fare acquisti. I giocatori sono quindi da considerarsi una risorsa polivalente dai gestori di negozi online, e l'impiego di meccaniche di Gamification contribuisce alla fidelizzazione del cliente [STE10].

- Scuola e educazione: l'impiego di giochi nella scuola non è inedito. È possibile individuare tre categorie all'interno delle quali catalogare i vari giochi [COR10]:
    - i classici giochi *edutainment* (termine ottenuto dalla contrazione di "education", educazione, ed "entertainment", divertimento). Storicamente sono stati i primi a comparire nella scuola, e proprio per questo i bambini tendono a darli per scontati, essendo abituati alla loro presenza. Questa tipologia di gioco è spesso ben fatta, ma non centra pienamente l'obiettivo educativo che si prefigge, oltre ad avere spesso una grafica poco appagante. Gli studenti tendono quindi ad annoiarsi velocemente, anche a causa della ripetitività. Questo non significa che siano inutili, ma purtroppo tendono a costituire un'occasione sprecata.
    - i giochi sviluppati dagli studenti. Questa categoria è successiva alla precedente, e probabilmente è nata anche per rimediare ai punti in cui la prima era carente. In questo caso l'idea è quella di insegnare agli studenti come sviluppare il loro gioco personale, utilizzando strumenti come *Scratch* (logo in figura 1.2), sviluppato dal Lifelong Kindergarten Group presso il Media Lab del MIT, oppure *Kudo*, sviluppato da Microsoft, che non richiedono conoscenze avanzate di programmazione, rendendo così questo mondo accessibile anche ai bambini. L'esperienza è quindi più coinvolgente e motivante. Con Scratch inoltre i giochi realizzati sono soggetti a feedback degli altri utenti, incentivando quindi lo studente che li realizza ad impegnarsi per poi ottenere gratificazione. Questo approccio ha anche il vantaggio collaterale di avvicinare i bambini verso il mondo degli sviluppatori, anche in un'ottica di una possibile professione futura. Tuttavia, affinché questo metodo sia efficace si richiede che lo studente sia interessato ed appassionato

all'argomento, magari sviluppando anche al di fuori delle ore strettamente scolastiche. Se uno studente non è motivato, difficilmente potrà trarre giovamento da questo approccio, a causa dell'impegno personale richiesto, difficile da gestire senza una passione di fondo.

  - i giochi realizzati con meccaniche di Gamification, descritte dettagliatamente del paragrafo 1.2.1. A differenza della tipologia precedente, non fa affidamento sulla motivazione degli studenti, ma sulle meccaniche di competizione e assegnazione di ricompense ad ogni avanzamento di livello. In questo modo, uno studente inizialmente non motivato può acquistare interesse proprio grazie a queste meccaniche, e collateralmente realizzerà lo scopo dell'applicazione, ovvero di insegnare. La Gamification è ancora poco esplorata in ambito scolastico, e potrebbe riservare interessanti sorprese negli anni a venire.

- Addestramento lavorativo e militare: l'apprendimento non si limita solo alla scuola, ma anche a qualunque tipo di lavoro per il quale sia richiesto di acquisire conoscenze specifiche, al fine di rendere più piacevole ed efficace l'apprendimento. In questi ambiti l'accoglienza dell'utente può essere molto varia: da un lato la novità è accolta in modo molto positivo, dall'altro esiste anche una serie di lavoratori che oppongono resistenza, preferendo metodi più tradizionali [HUL10].
- Social Learning, ovvero l'apprendimento e la creazione di valore realizzato internamente ad una rete attraverso scambi formali e informali, basato sulle linee di base che determinano il comportamento individuale e un modello teorico sui processi alla base del comportamento. Anche qui la Gamification può costituire una spinta motivazionale per l'apprendimento in questa direzione [SLB11].
- E-Learning, ovvero l'utilizzo di Internet e dei sistemi multimediali al fine di ottenere un miglioramento nell'apprendimento, attraverso una migliore accessibilità ai servizi, quali ad esempio i servizi di condivisione e creazione di dati collaborativi [VIL11].
- Salute e benessere, al fine di sensibilizzare gli utenti a specifiche problematiche medico-sanitarie o renderli a conoscenza dell'esistenza di servizi. [VIL11].

Figura 1.2 – Logo di Scratch

1.2 Monetizzazione

Un'estensione del meccanismo di compravendita dei beni in gioco è costituita dai giochi cosiddetti *free-to-play*, che consentono di comprare la valuta interna al gioco con denaro reale. Questo modello di vendita ha conosciuto una forte espansione negli ultimi anni. Tende ad avere successo in quanto punta a catturare inizialmente l'interesse dell'utente offrendo un gioco gratuito, per poi passare a proporre contenuti a pagamento una volta che l'utente ha imparato a conoscere il gioco. In questo modo il produttore aumenta il proprio margine di guadagno rispetto ad un rilascio di tipo gratuito. Allo stesso tempo, il fatto che il gioco di base sia gratuito è in grado di catturare un pubblico maggiore, in quanto molti utenti si potrebbero avvicinare al gioco anche senza una forte motivazione iniziale.

Quando si sceglie di utilizzare questo modello di business, il rischio da tenere sempre presente è che la possibilità di utilizzare denaro reale porti squilibrio all'interno del gioco tra giocatori paganti e non: se ciò dovesse avvenire, questi ultimi si allontanerebbero progressivamente dal gioco, risultando quindi controproducente. Questo è particolarmente vero per i giochi in cui gli utenti sono in diretta competizione l'uno con l'altro non solo sul confronto del punteggio, ma anche internamente al gioco. In particolare si assiste a questa meccanica specialmente nei giochi di guerra.

Una soluzione per evitare questa situazione può essere quella di dividere gli

utenti in base al livello, in modo che ogni giocatore possa competere direttamente solo con quelli di pari livello: così facendo, un eventuale utente che dovesse scegliere di scalare rapidamente la classifica pagando denaro reale, si ritroverebbe a competere con persone di pari livello, le quali potrebbero essere giunte a quel punto tanto pagando quanto gratuitamente. Un'altra possibilità è quella di consentire l'acquisto solo di beni che non condizionano il bilanciamento del gioco, ma esclusivamente aspetti grafici o poco rilevanti. Se questa meccanica non viene implementata con cura, un giocatore di basso livello non avrebbe speranze di progredire e si disaffezionerebbe rapidamente al gioco.

Il rischio di degenerare in forme di gioco autodistruttive non è da sottovalutare, in quanto molte software house rischiano continuamente di ricaderci, sacrificando l'esperienza dell'utente sull'altare della monetizzazione facile. Esiste un’ampia categoria di giochi *free-to-play* che sono chiamati dispregiativamente dai giocatori *pay-to-win*: il riferimento è a quei giochi che si presentano come *free-to-play* ma in realtà diventano ingiocabili nel giro di poco tempo, a meno di non spendere denaro reale. Questo può essere realizzato ad esempio aumentando la difficoltà in modo non lineare con i potenziamenti che l’utente acquisisce normalmente, rendendo impossibile procedere oltre i primi livelli senza pagare.

Esistono addirittura degli studi che spiegano come realizzare queste meccaniche, in modo da spremere agli utenti il massimo dei soldi possibili. Si tratta di creare uno squilibrio tra la moneta virtuale guadagnata e quella effettivamente necessaria per procedere nel gioco [MAB15].

1.3 Serious games

L'impiego degli elementi di gioco a fini educativi è precedente alla Gamification: un’altra categoria di gioco è quella dei *serious games*, in cui si uniscono aspetti educativi ad altri puramente di gioco. La differenza principale rispetto alla Gamification è che si tratta di giochi completi e ben definiti nel loro percorso. Anche in questo caso, l’idea è quella di istruire gli utenti del gioco rispetto ad una tematica specifica. Anche qui l’aspetto videoludico è fondamentale per catturare l’attenzione, a prescindere dall’argomento trattato, che può essere variegato.

Al contrario della Gamification, in cui la distinzione tra aspetti didattici e puramente videoludici è ben distinta, nei *serious games* il confine è più sfumato: non è

immediato distinguerli in maniera precisa dagli altri giochi che hanno come unico scopo l'intrattenimento. Infatti, in molti casi uno stesso gioco può entrare od uscire da questa categoria in base all'utilizzo dell'utente. Un esempio lampante è costituito dai simulatori, utilizzati abitualmente dai professionisti dei vari settori per prepararsi agli scenari reali. In particolare, per i piloti di aerei si tratta di un'esperienza fondamentale: solo dopo un lungo numero di ore trascorse al simulatore potranno passare a pilotare un vero aereo.

Figura 1.3 - Schermata di Flight Simulator X – Cabina di pilotaggio Boeing 737-800

D'altro canto, un simulatore di volo può essere utilizzato anche da un utente qualunque, non necessariamente pilota, che vuole semplicemente divertirsi senza imparare realmente tutti i dettagli tecnici. Per questo motivo molti software di simulazione, non solo di volo, ma anche di corse automobilistiche, sono configurabili nel livello di realisticità in modo da adattarsi ad entrambe le tipologie di giocatore: un pilota che deve completare il suo addestramento vorrà mantenere un'esperienza il più possibile vicina alla realtà, e apprezzerà la possibilità di gestire ogni singolo aspetto del simulatore in maniera manuale; al contrario, un giocatore generico che vuole divertirsi molto probabilmente non conoscerà nel dettaglio ogni aspetto tecnico relativo al

dominio applicativo, e rimarrebbe spaesato di fronte ad una quantità di personalizzazioni elevata. Di conseguenza preferirà un software meno configurabile ma più semplice da usare, in modo da concentrarsi solo sugli aspetti principali e meno tecnici, sacrificando una quantità più o meno grande di realisticità. Un esempio di software di questo tipo è lo storico *Flight Simulator X* di Microsoft, rilasciato nel 2006 (figura 1.3).

1.3.1 Cenni storici

Storicamente le origini dei *serious games* sono associate ai *Kriegsspiele*, le simulazioni di guerra organizzate agli inizi del Settecento per l'addestramento dell'esercito prussiano. In realtà è improprio catalogarli tra i *serious games*, anche se costituirono effettivamente la base per l'utilizzo dei giochi a scopo formativo. Per arrivare ad una concezione di gioco più moderna si dovette attendere i primi giochi da tavolo del Novecento come lo storico Monopoly [ADA12].

L'utilizzo del computer, e quindi di software, in questo ambito ebbe inizio negli anni Cinquanta del Novecento, quando gli Stati Uniti, tramite la Johns Hopkins University, ne iniziarono l'uso per realizzare simulazioni militari [ARM08].

Solo a partire dagli anni Ottanta i videogiochi cominciarono a diffondersi su larga scala. Grazie a questa forte espansione di mercato, anche i *serious games* trovarono terreno fertile per svilupparsi, specialmente nei paesi nordici, dove divennero ben presto argomento di dibattito e ricerca scientifica, sia da un punto di vista ingegneristico che psicologico [TIG14].

Fu lo sviluppatore di giochi militari Clark Abt a coniare nel 1971 il termine *serious game*. Nel suo libro "Serious Games" definisce i giochi educativi, a prescindere dal supporto digitale o non, descrivendoli come giochi con un esplicito e ben strutturato scopo educativo, il cui scopo non è direttamente il divertimento dell'utente, pur non escludendolo [ABT70].

Ai nostri giorni vengono definiti *serious games* quei videogiochi educativi che, secondo la definizione di Sawyer, costituiscono un qualunque uso significativo di elementi videoludici o provenienti dall'industria dei giochi, il cui scopo non è però il divertimento [SAW07].

1.3.2 Pedagogia dei serious games

Inizialmente i *serious games* erano basati sulla psicologia comportamentale, metodo sviluppato dallo psicologo John Watson agli inizi del Novecento, il quale sosteneva che solo i comportamenti espliciti di un individuo sono analizzabili scientificamente, attraverso la relazione tra gli stimoli ambientali e i comportamenti associati. Successivamente si diffusero ulteriori teorie pedagogiche, in particolare l'apprendimento esperienziale, secondo il quale le informazioni, le esperienze e le emozioni vissute restano impresse a lungo, consentendo al giocatore di migliorare le proprie abilità attraverso un utilizzo prolungato del gioco. Si tratta del cosiddetto "learning by doing" [ARN12].

L'aspetto pedagogico da tenere in considerazione è che in questo modo l'utente ha svolto in prima persona delle azioni (per progredire nel gioco), arricchendo il proprio bagaglio conoscitivo attraverso l'esperienza diretta, la quale è più efficace rispetto ad assistere passivamente ad una lezione frontale. Inoltre, tanto più l'esperienza di gioco sarà simile alla realtà, allo stesso modo aumenterà la fiducia dell'utente rispetto alle proprie possibilità.

Infine, l'ambiente di gioco è protetto: l'utente non viene giudicato e ha la possibilità di ripetere l'esercizio per migliorarsi. La ripetizione aumenta la padronanza e la serenità con cui affronterà il problema quando si presenterà realmente, anche al di fuori dell'ambiente simulato.

1.3.3 Applicazioni dei serious games

L'utilizzo di *serious games* è presente in molti ambiti, alcuni più specifici, altri che entrano maggiormente in concorrenza con la Gamification [ARN12]:

- nella formazione dei dipendenti, sia per addestrarli come nell'esempio dei piloti nel paragrafo 1.3, sia per simulare situazioni in cui è necessaria un'esperienza diretta (come vendite telefoniche o colloqui di vario genere);
- ogni qualvolta si vuole veicolare un messaggio positivo per sensibilizzare le persone relativamente ad una specifica tematica;
- in ambito commerciale, per pubblicizzare un prodotto;
- in ambito militare, per addestrare sia i soldati che gli operatori tecnici.

Esistono anche casi di videogiochi non creati come *serious games* e senza pretese di essere educativi, ma che sono stati in grado di ritagliarsi un settore anche in questo ambito: è il caso dei giochi di guerra o avventura che sono ambientati in specifici periodi storici, ricreati fedelmente nell’ambiente e nelle meccaniche di gioco, e per questo possono avere una valenza didattica, per quanto lo scopo di intrattenimento sia preponderante [ORL07].

1.4 Smartphone, tablet e social network

Negli ultimi anni l'esplosione del fenomeno smartphone e tablet ha fornito nuove opportunità di sviluppo per questa tipologia di applicazioni, e non solo. La presenza ormai pervasiva di dispositivi portatili costantemente connessi ad Internet ha reso possibile l'origine di una nuova categoria di applicazione, che non è semplicemente un'estensione di un computer, ma consente di accedere ad una serie di elementi hardware precedentemente non diffusi: GPS, fotocamera, accelerometro, giroscopio, e qualunque altro sensore potenzialmente integrato in un dispositivo portatile.

I dispositivi utilizzati non sono nati esplicitamente per il gioco, essendo originariamente indirizzati al solo settore della telefonia mobile, ma possono essere sfruttati anche per questo scopo. Questo è possibile grazie alla performance hardware elevate rispetto ai dispositivi di vecchia generazione e al diffondersi di sistemi operativi veri e propri, quali Android di Google, iOS di Apple e Windows Phone di Microsoft (loghi presenti in figura 1.4), indirizzati all’ambito mobile. Non si tratta quindi più di un firmware chiuso e proprietario, gestito totalmente dal produttore, ma di un ecosistema più ampio che lascia spazio agli sviluppatori. Il gaming su piattaforma mobile nel tempo è passato in alcuni casi da funzione secondaria a primaria: in questo senso trovano giustificazioni smartphone e tablet particolarmente potenti, che per un utilizzo standard sarebbero ampiamente sovradimensionati, ma che possono trovare una ragione per l’utente che intende eseguire applicazioni che richiedono un’elevata potenza computazionale, come ad esempio alcune categorie di gioco.

Figura 1.4 – Loghi di Android, iOS e Windows Phone

Inoltre, la diffusione massiva dei social network ha aperto la strada ad un nuovo modo di interazione tra gli utenti: è possibile coniugare l'esperienza di gioco di un'applicazione all'elemento “social”, consentendo all'utente di pubblicare i risultati conseguiti all’interno del gioco sui social network. Questo ha una doppia valenza: da un lato esalta le performance del singolo utente, che si sentirà quindi orgoglioso dei progressi raggiunti, dall'altro inciterà gli amici di quell'utente ad utilizzare a loro volta il gioco, al fine di battere il record dell'amico. Questa motivazione è molto più forte grazie all'utilizzo di social network, poiché il giocatore non si confronta solo con un'infinità di altri giocatori senza volto, ma con i propri conoscenti, rendendo l'esperienza competitiva più stimolante. I maggiori social network, Facebook in primis, si impegnano per favorire l'integrazione con i servizi esterni di terze parti rilasciando le proprie API che consentono agli sviluppatori di accedere in maniera immediata alle funzionalità social [FAC15].

Non solo: sono comparse meccaniche di Gamification anche nel settore dell’editoria online: la testata giornalistica *The Huffington Post*, forte di un numero di lettori pari a ventitré milioni solamente negli Stati Uniti, ha introdotto nel 2010 una serie di badge conquistabili dai lettori e visibili a tutti gli utenti: in questo modo ogni utente veniva spronato a raggiungere quei traguardi per prevalere all’interno della community, aumentando come conseguenza il traffico del sito e il numero di interazioni tra utenti, ad un costo trascurabile per il giornale. La presenza pervasiva di smartphone ha consentito un aumento importante del numero di commenti agli articoli online, in quanto chiunque può accedere in qualunque momento, non solo da casa o in presenza di un computer. I badge erano quelli visibili in figura 1.5 e descritti qui:

- Networker: badge inserito per incentivare gli utenti a collegare il proprio account con i social network Twitter e Facebook, al fine di aumentare la viralità e la formazione di connessioni tra gli utenti, con lo scopo di fidelizzarli a quella piattaforma (un utente è restio a spostarsi da una piattaforma in cui ha una certa reputazione e popolarità ad una nuova). Come gli altri badge, è composto da più livelli, e gli utenti che raggiungono il secondo livello ottengono la possibilità di scrivere in rosso (e non solo in nero).
- Superuser: badge che incentiva a commentare ogni contenuto inserito sulla piattaforma, aumentando quindi il tempo speso sul portale, il coinvolgimento dell'utente rispetto ai contenuti pubblicati e la quantità di commenti inseriti da ogni utente.
- Moderator: badge che incentiva gli utenti a segnalare i contenuti inappropriati. Più contenuti vengono segnalati ed effettivamente rimossi dai moderatori "reali", più si sale di livello nella classifica del badge. Questo badge è stato introdotto per facilitare il compito dei moderatori, travolti da un'enorme quantità di commenti da controllare.

Figura 1.5 – Badge di "The Huffington Post"

1.4.1 L'espansione del pubblico

L'avvento di tecnologie hardware portatili come gli smartphone e tablet, e di tecnologie software come i social network hanno aumentato notevolmente il pubblico di videogiocatori. Di conseguenza una grande quantità di non giocatori ha avuto modo di avvicinarsi a questo mondo nei ritagli di tempo. Il pubblico di riferimento dei videogiochi non si è espanso quindi solo come numero, rimanendo all'interno di uno specifico target di sesso ed età. In particolare si è passati da un pubblico composto principalmente da giovani uomini ad una suddivisione più eterogenea: una serie di studi

il numero delle donne videogiocatrici è in aumento, avvicinandosi al 50%. Questa diffusione è stata possibile anche grazie ad una tipologia di gioco poco impegnativa sia come complessità, sia come quantità di tempo impegnato: le meccaniche di gioco infatti sono semplici e raramente richiedono un notevole sforzo intellettuale, e il gioco è spesso organizzato in partite di breve durata.

Non a caso le statistiche relative ai negozi ufficiali delle piattaforme mobili la categoria dei giochi si colloca al primo posto sia come fatturato generato, sia come numero di applicazioni presenti [TAG11].

1.5 Privacy e sicurezza

Sebbene l'avvento di nuove piattaforme mobili performanti abbia portato una ventata di aria fresca al settore del gaming, sono sorte altre problematiche di privacy e sicurezza. Infatti, se da un lato grazie a questi dispositivi molte persone si sono avvicinate al mondo dei giochi (e della tecnologia più in generale), dall'altro è altrettanto vero che molte di queste persone hanno competenze informatiche nulle o quasi. L'analfabetismo informatico ancora oggi è a livelli impressionanti. Si parla di analfabetismo informatico intendendo l'incapacità di utilizzare i mezzi informatici (non necessariamente computer) in maniera attiva e consapevole. Questo non significa necessariamente non essere in grado di eseguire qualche operazione di base (in quel caso si parla di analfabetismo informatico assoluto), ma denota l'incapacità generale ad affrontare attivamente situazioni variabili (analfabetismo informatico relativo o funzionale).

Secondo un rapporto ISTAT del 2009, in Italia oltre un milione e settecentomila giovani tra i 15 e i 29 anni non hanno mai utilizzato un computer nell'ultimo anno. Inoltre, secondo una ricerca realizzata da Nielsen Media Research nello stesso anno, solo il 55% degli italiani utilizza Internet, e solo il 34% lo utilizza in maniera consapevole e critica. Le cause sono da ricercare da un lato nella carenza di infrastrutture per l'accesso a Internet e al *digital divide*, mentre dall'altro nella scarsa diffusione di conoscenze minime e di figure di intermediazione in grado di insegnare alle nuove generazioni le basi dell'informatica [DEL11].

Moltissime di queste persone, tuttavia, pur non possedendo delle basi informatiche, acquistano e utilizzano dispositivi mobili, acquisendo le conoscenze strettamente necessarie a fare ciò di cui hanno bisogno, che nella maggior parte dei casi

equivale a navigare su Facebook (o altri social network) e utilizzare un software di messaggistica istantanea (primo tra tutti Whatsapp). Tuttavia, poiché non conoscono nulla del funzionamento del loro dispositivo, non sono in grado di rilevare i pericoli della rete. Inoltre, proprio a causa della mancanza di conoscenza, hanno un'elevatissima probabilità di cadere in trappole che qualunque informatico considererebbe banali, ma che loro non sono in grado di individuare, non disponendo degli strumenti critici e tecnici. Non solo non individuano le minacce: le sottovalutano, rimanendo vittime di truffe o peggio. Il comportamento tipico è quello di ignorare qualunque segnale di allarme finché non si verifica qualcosa di abbastanza grave da non consentire più di continuare ad ignorare il problema.

Il diffondersi di smartphone e tablet ha aperto la strada anche ai malintenzionati, incentivati ancora di più dal fatto che la percentuale di utenti inesperti è maggiore rispetto ad altre piattaforme. Le varie tipologie di attacco possono essere genericamente racchiuse in due categorie:

- derivate dall'ambito desktop: rientrano in questa categoria tutti gli attacchi che non sfruttano elementi specifici delle piattaforme mobili, ma che si presentano quasi uguali su computer. Si tratta soprattutto di pubblicità ingannevoli, che inducono l'utente ad aprirle, più o meno consapevolmente. Una volta aperte tenteranno di infettare il dispositivo tramite vulnerabilità note, o tentando di convincere l'utente ad installare un *trojan horse* mascherato da applicazione legittima.
- specifiche per la piattaforma: rientrano in questa categoria tutti gli attacchi che fanno uso di caratteristiche specifiche dei dispositivi e del sistema operativo. Si tratta per la maggior parte di *trojan horse* che richiedono il consenso dell'utente per essere installati. Per giungere a questo scopo vengono inseriti dai malintenzionati anche direttamente sugli store ufficiali delle piattaforme, spacciandoli per applicazioni legittime (specialmente software di pulizia e giochi). Una volta installati, nel migliore dei casi si limitano ad invadere l'utente di pubblicità indesiderata, altrimenti eseguono il furto dei dati che l'utente conserva sul dispositivo (numeri di telefono, fotografie, dati personali, credenziali di accesso).

L’obiettivo principale dei malintenzionati è guadagnare denaro alle spalle degli utenti: per questo motivo sono molto diffusi i software che visualizzano pubblicità e che eseguono il furto di dati sensibili. Tutto ciò che è monetizzabile è potenzialmente un bersaglio. Esistono anche gli attacchi mirati ad opera di esperti, ma rappresentano una percentuale infinitesima rispetto al totale, ed è piuttosto improbabile che un utente si ritrovi a fronteggiare una minaccia di questo tipo. Inoltre, in quest’ultimo caso non sarebbero sufficienti degli accorgimenti di base, ma sarebbero opportune conoscenze più avanzate, decisamente fuori dalla portata dell’utente medio.

È importante anche parlare di un aspetto molto significativo ma spesso sottovalutato: la sicurezza dei sistemi operativi mobili è estremamente scarsa. Come in qualunque sistema operativo, anche quelli dedicati a smartphone e tablet non sono immuni dalla presenza di vulnerabilità software, che vengono periodicamente individuate dai ricercatori. In questi casi il rilascio di aggiornamenti correttivi è nelle mani del produttore del dispositivo. Di fatto, però, gli aggiornamenti non sempre vengono rilasciati: è prassi comune della maggior parte dei produttori abbandonare i dispositivi rilasciati dopo due o tre anni dalla messa in commercio, specialmente nel mondo Android, in cui è presente un’elevata frammentazione (esistono concorrentemente più versioni del sistema operativo, e ogni produttore applica al firmware originale rilasciato da Google le proprie personalizzazioni). Inoltre, anche nel caso in cui gli aggiornamenti siano effettivamente rilasciati, sono talvolta necessari periodi di tempo molto lunghi, anche nell’ordine dei mesi, prima che questi giungano agli utenti finali. A questo proposito, lo schema in figura 1.6 evidenzia la percentuale di dispositivi Android vulnerabili. Degni di nota sono i grandi dislivelli che si creano nel grafico quando vengono scoperte nuove vulnerabilità, a cui segue un periodo molto lungo prima che il livello medio di sicurezza sia ripristinato in seguito al rilascio di aggiornamenti correttivi [ANV15]. Questo specifico problema è mitigato nei sistemi operativi di Apple (iOS), in cui la frammentazione è minore rispetto ad Android, e su base annuale i dispositivi vengono quasi tutti aggiornati, e Microsoft (Windows Phone), in cui gli aggiornamenti sono rilasciati tempestivamente da Microsoft stessa. Su Android l’unico modo di avere aggiornamenti tempestivi è installare un firmware personalizzato, diverso da quello installato dal produttore. Questa operazione tuttavia non è praticabile dall’utente medio, lasciando in questo modo un enorme numero di dispositivi vulnerabili.

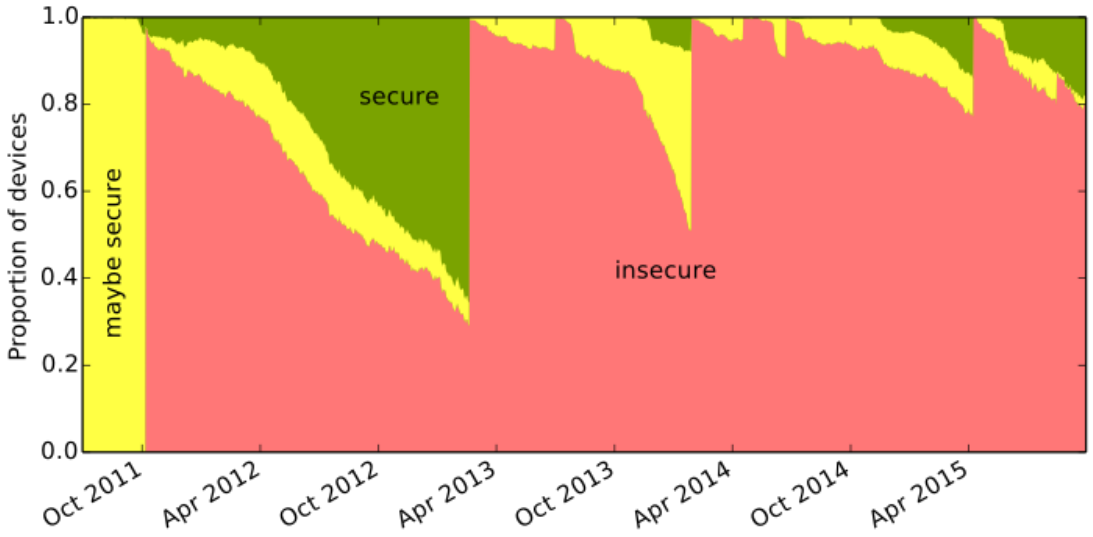

Figura 1.6 – Dispositivi Android vulnerabili [ANV15]

2 Progetto: specifiche, architettura e tecnologie coinvolte

In questo capitolo verranno descritte le tecnologie utilizzate per la realizzazione del progetto, motivando le scelte in relazione agli obiettivi preposti. Si partirà da un analisi dei sistemi operativi e piattaforme maggiormente diffusi sul mercato, passando poi alle tecnologie di sviluppo disponibili su ogni piattaforma. Infine, date le possibili scelte, si stabiliranno delle specifiche unificate per il progetto e una possibile architettura di rete.

2.1 Introduzione

Come precedentemente accennato, l'obiettivo del progetto è quello di coinvolgere maggiormente i giovani nello studio della matematica. Per raggiungere questo risultato è fondamentale innanzitutto determinare le modalità con le quali i giovani d'oggi si rapportano alla tecnologia. Nel paragrafo 4 del primo capitolo si è evidenziato come la tendenza in questo ambito sia a favore dei dispositivi portatili quali smartphone e tablet piuttosto che tradizionali computer.

Se da un lato quest'informazione consente di imprimere al progetto una direzione apparentemente univoca, dall'altro è bene considerare i pro e i contro di uno sviluppo basato su piattaforme mobili. Inoltre, sebbene la tendenza dei giovani sia di adottare principalmente dispositivi mobili, questo non significa che non utilizzino anche il computer, seppure in misura minore. Non solo: sebbene non rappresenti la maggioranza, esiste una categoria di potenziali utenti che, possedendo sia uno

smartphone che un computer, preferiscono utilizzare quest'ultimo per svariate ragioni, quali ad esempio la maggiore versatilità, le migliori prestazioni, le maggiori dimensioni dello schermo e la disponibilità di dispositivi di interazione quali mouse e tastiera, più precisi rispetto allo schermo touch screen di uno smartphone. Infine, esiste una categoria di utenti che non possiede uno smartphone: questo può avvenire per una scelta specifica del ragazzo, o a causa di un'imposizione dall'alto (un numero non trascurabile di genitori rimanda l'acquisto di smartphone per i figli a quando saranno più grandi).

A causa di questo panorama variegato, nel quale emerge una grande varietà di dispositivi target, sicuramente sarà importante sviluppare per piattaforme mobili, ma con un occhio riguardo anche ai dispositivi desktop. Verranno quindi analizzati i principali sistemi operativi presenti sul mercato, per individuare pattern di diffusione, e in seguito stabilire se convenga sviluppare su alcune piattaforme piuttosto che altre.

2.2 Market share dei sistemi operativi e versioni supportate

In ambito mobile i sistemi operativi più diffusi sono Android di Google (declinato nelle personalizzazioni dei vari produttori), iOS di Apple e Windows Phone di Microsoft; in ambito desktop sono diffusi principalmente i sistemi Windows di Microsoft (nelle varie versioni), ma una minore porzione di utenti utilizza una distribuzione Linux o Mac OSX di Apple.

Tendenzialmente attraverso versioni diverse dello stesso sistema operativo viene mantenuta la compatibilità delle applicazioni. In alcuni casi specifici è tuttavia necessario apportare delle correzioni, ma, poiché per la realizzazione del progetto non si prevede di utilizzare funzionalità a basso livello del sistema, si presume che non si avranno problemi di questo tipo. Ad ogni modo non saranno ufficialmente supportati sistemi operativi obsoleti, quali ad esempio Windows XP, il cui supporto è terminato l'8 aprile 2014 [MIC14] e versioni di Android precedenti alla 4.1 (Jelly Bean), la cui diffusione è ormai limitata [ADD15]. Questa decisione è stata presa al fine di non rendere caotico lo sviluppo e non appesantire eccessivamente il codice dell'applicazione con eccezioni di compatibilità, rese necessarie dalle specificità di ogni sistema operativo. Inoltre, solo una minoranza degli utenti utilizza versioni obsolete dei sistemi operativi, ed è auspicabile che eseguano un aggiornamento quanto prima. Le

informazioni relative alle percentuali di diffusione dei sistemi operativi sono state raccolte da agenzie come Gartner e a partire dai contatori statistici utilizzati nelle pagine web, che rilevano il sistema operativo utilizzato dai client [GAR15]. La figura 2.1 è tratta dall'analisi dei dai prodotti da questi studi [NMS15]. Ad ogni modo, non si esclude a priori il funzionamento dell'applicazione su questi sistemi operativi, ma non lo si garantisce. L'utente che utilizzerà l'applicazione su sistemi obsoleti lo farà a proprio rischio e pericolo, e in caso di bug o malfunzionamenti non potrà ricevere supporto ufficiale.

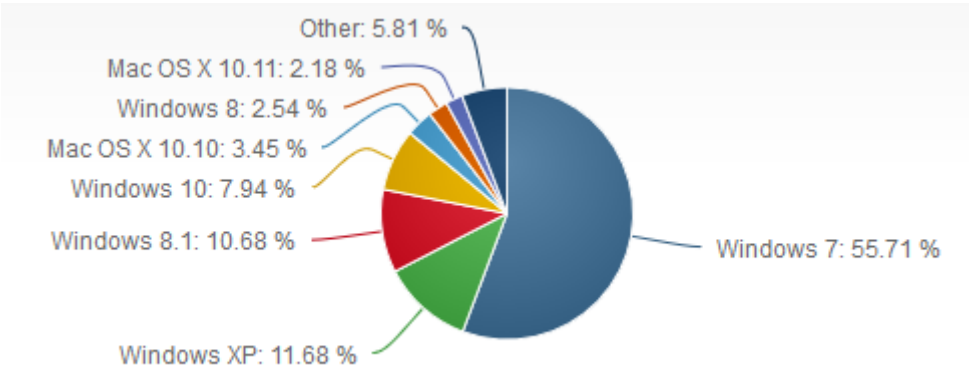

Figura 2.1 - Quote di mercato dei sistemi operativi desktop [NMS15]

2.3 Analisi delle tecnologie di sviluppo native

Esistendo sistemi operativi di produttori differenti, si avranno come conseguenza piattaforme di sviluppo e linguaggi di programmazione dipendenti dalla piattaforma. Verranno ora analizzate le specificità di ogni sistema mobile, evidenziando le tecnologie disponibili.

- Android: le applicazioni vengono scritte prevalentemente in Java, avvalendosi di strumenti quali Android Studio (rilasciato più recentemente) o Eclipse (software preesistente per la programmazioni in Java, che consente anche di sviluppare applicazioni Android) [AND15].
- iOS: la più recente tecnologia di sviluppo è rappresentata dal linguaggio Swift, creato da Apple nel 2014, in abbinamento all'ambiente di sviluppo Xcode; precedentemente si utilizzava il linguaggio Objective-C, sempre in ambiente Xcode [APP15].
- Windows Phone: Microsoft ha creato il concetto di Universal Windows Platform (UWP), ovvero un tipo di applicazione che può essere eseguita

indistintamente tanto dai sistemi operativi mobili quanto dalle versioni desktop di Windows più recenti (da Windows 8 in poi). I linguaggi utilizzati sono principalmente C# e XAML, oltre a linguaggi provenienti dal Web, quali Javascript, HTML e CSS. L'ambiente di sviluppo è Visual Studio [WIN15].

Analogamente, per i sistemi operativi Desktop è possibile sviluppare applicazioni native. In questo caso è possibile utilizzare una grande varietà di linguaggi di programmazione e ambienti di sviluppo, in quanto i vincoli sono molto meno stingenti rispetto ai sistemi operativi mobili. Una menzione speciale è tuttavia opportuna per Java, che consente di creare applicazioni multi-piattaforma in esecuzione all’interno di una macchina virtuale (Java Virtual Machine, JVM). Anche la piattaforma .NET di Microsoft segue una filosofia simile, ma al momento è disponibile solo per Windows. In realtà la situazione potrebbe cambiare in futuro, in quanto Microsoft sta iniziando a sviluppare parte di .NET (per ora solo le componenti dedicate ai server) anche su Linux [LAR15].

Analizzando queste tecnologie emerge come non esista un linguaggio universale per tutte le piattaforme desktop e mobili. Le tecnologie che coprono il maggior numero di sistemi operativi sono, nell'ordine, Java (disponibile su tutti i sistemi desktop e utilizzabile su Android) e Microsoft UWP (disponibile sia nelle versioni desktop che mobili di Windows). Risulta chiaro come sviluppare applicazioni native per tutte queste piattaforme richiederebbe inevitabilmente il coinvolgimento di almeno tre linguaggi di programmazione, rendendo necessario portare avanti in maniera parallela lo sviluppo di più versioni della stessa applicazione, realizzate con tecnologie diverse. In alternativa si potrebbe decidere di sviluppare solo su una piattaforma specifica, abbandonando le altre. In questo caso probabilmente si privilegerebbe Android, in quanto consentirebbe di coprire il maggior numero di utenti. Inoltre, lo sviluppo su Android è possibile da qualunque sistema operativo desktop, mentre lo sviluppo su piattaforme Microsoft o Apple è possibile solo utilizzando sistemi operativi dello stesso produttore.

D'altro canto, la realizzazione di applicazioni native consente di ottenere le massime prestazioni e la migliore efficienza energetica su ogni dispositivo, traducendosi in un migliore gradimento dell'utente, in particolare dai possessori di dispositivi poco performanti. La maggiore complessità del progetto derivante dallo sviluppo di applicazioni parallele, tuttavia, è un fattore molto importante da tenere in

considerazione. Infine, l'applicazione che si intende realizzare non richiederà performance elevate, rendendo non critico lo sviluppo di un'applicazione nativa.

2.4 WebApp

All'estremo opposto rispetto alle applicazioni native si hanno le WebApp. Con il termine WebApp si intende un tipo di applicazione realizzata utilizzando principalmente tecnologie web, in particolare HTML, CSS e Javascript (loghi in figura 2.2). Conseguentemente l'ambiente di esecuzione è il browser dell'utente, esattamente come per qualunque pagina web. Utilizzare le tecnologie web per lo sviluppo di applicazioni può sembrare un controsenso, ma si tratta di una scelta che porta anche dei benefici, oltre ad alcuni svantaggi.

Figura 2.2 - Loghi di HTML5, CSS3 e Javascript

Rispetto ai tempi della nascita del Web all'inizio degli anni Novanta, i linguaggi dedicati allo sviluppo di siti web si sono sviluppati passando prima dal supporto di contenuti statici a quello di contenuti dinamici, inizialmente con l'introduzione di tecnologie server-side come CGI, e successivamente con l'introduzione di Javascript, che funge da linguaggio di scripting client-side. Negli anni successivi CGI è stato in gran parte soppiantato da PHP, JSP e ASP. Già con l'introduzione di queste tecnologie si è reso possibile sviluppare piccole applicazioni all'interno del browser,

eventualmente anche accedendo ai dati di un database esterno. All'epoca però regnava ancora la mancanza di standard universalmente riconosciuti, facendo sì che ogni browser si comportasse in modo diverso a fronte di istruzioni uguali. Questo rendeva complicato lo sviluppo di siti web, obbligando gli sviluppatori a realizzare workaround per ogni specifico browser, o dedicandosi allo sviluppo su un solo browser, tagliando fuori la compatibilità con gli altri. La definizione di specifiche comuni e la standardizzazione avvenne grazie al W3C (World Wide Web Consortium). Nel 1998 venne definito XML, ponendo le basi per il web semantico.

Con le ultime versioni dei linguaggi HTML5 e CSS3 la standardizzazione si può dire quasi completa: tutti i browser moderni supportano questi standard, con l'eccezione delle specifiche ancora in via di sviluppo, che non sempre sono supportate correttamente. Ad ogni modo, fintanto che gli sviluppatori si attengono agli standard, possono essere ragionevolmente sicuri che il loro sito web verrà visualizzato in maniera corretta su tutti i browser. Infatti, gli sviluppatori di browser web sono molto più sensibili rispetto al passato alla definizione di standard comuni. HTML5 e CSS3 non rappresentano solo un notevole passo avanti nella standardizzazione, ma pongono le basi per lo sviluppo di applicazioni web complesse: in primo luogo grazie alla separazione tra informazioni semantiche (di cui si occupa HTML) e di stile (di cui si occupa CSS), in secondo luogo grazie all'introduzione di funzionalità complesse quali ad esempio le animazioni, precedentemente realizzabili solo programmando funzioni personalizzate in Javascript, e ora utilizzabili nativamente con CSS, con un notevole risparmio di forze da parte del programmatore. Inoltre, il supporto nativo da parte dei browser rende queste operazioni avanzate molto meno costose dal punto di vista computazionale, riducendo la quantità di codice Javascript personalizzato. Infine, le prestazioni di Javascript stesso sono notevolmente migliorate nel corso degli anni, grazie all'introduzione all'interno dei browser di motori Javascript sempre più performanti ed efficienti.

Negli ultimi anni sono state sviluppate anche librerie come WebGL [WEB15] che consentono alle applicazioni web di accedere a funzionalità di rendering 3D dell'host, migliorando notevolmente la qualità grafica ed espandendo le possibilità. Sono stati realizzati anche alcuni giochi di esempio utilizzando questa tecnologia, con ottimi risultati.

Grazie alle potenzialità di questi linguaggi sviluppare applicazioni web non solo è diventato possibile, ma anche sensato: sebbene le prestazioni di un'applicazione web saranno sempre inferiori rispetto a quelle di un'applicazione nativa, in tutti i casi in cui non sono richieste performance elevate, queste offrono comunque prestazioni accettabili per l'utilizzo. È bene però ricordare che un browser è un ambiente protetto che non può accedere direttamente alle funzionalità fisiche del dispositivo. Una WebApp quindi non potrà ad esempio accedere ai sensori dello smartphone e al file system. Esistono comunque alcune eccezioni notevoli, quali i cookie e l'utilizzo della fotocamera, che è stato recentemente implementato. Se l'applicazione che si vuole sviluppare richiede l'accesso a queste funzionalità, la via della WebApp non è percorribile.

Dal punto di vista dello sviluppatore, realizzare le stesse funzionalità in un'applicazione web può essere più complesso rispetto ad un'applicazione nativa, ma porta l'indubbio vantaggio di scrivere l'applicazione una sola volta, quando altrimenti sarebbero necessarie tante applicazioni native quante sono le piattaforme su cui si vuole sviluppare. Questa scelta quindi può rivelarsi vincente quando si vuole realizzare un'applicazione funzionante sul maggior numero di dispositivi possibili, minimizzando i costi di sviluppo. Inoltre, si ha un vantaggio collaterale relativo agli aggiornamenti: nel momento in cui un aggiornamento è pubblicato, raggiunge immediatamente tutti gli utenti, rendendo il più veloce possibile la distribuzione di patch correttive, e dando allo sviluppatore la garanzia che tutti gli utenti stiano utilizzando l'ultima versione disponibile, rendendo in questo modo superfluo il supporto di versioni precedenti.

Dal punto di vista dell'utente, come accennato precedentemente, potrebbero verificarsi maggiori problemi prestazionali con una WebApp piuttosto che con una nativa. Inoltre, per utilizzarla, l'utente dovrà collegarsi ad un sito web, non essendo l'applicazione installata sul dispositivo, al contrario delle applicazioni native. Se da un lato questo elimina il problema degli aggiornamenti, dall'altro rende meno immediato l'utilizzo per l'utente medio. Inoltre, poiché tutti i dati sono online, sarà necessaria una connessione ad Internet costante, a meno di non eseguire una forma di caching in locale da parte del browser (e quindi non facilmente prevedibile dallo sviluppatore). Per alcuni utenti questo potrebbe essere un problema, specialmente se non dispongono di una connessione ad Internet stabile. Infine, anche per gli utenti sempre connessi ad Internet, si avrà un maggiore consumo di traffico dati, il che potrebbe essere un

problema per tutti gli utenti che utilizzano connessioni a consumo: in Italia vale a dire la quasi totalità degli utenti mobili, dal momento che gli operatori offrono esclusivamente connessioni mobili a consumo. Potrebbe non essere un problema per gli utenti che accedono da linea telefonica fissa, sia tramite computer che tramite smartphone o tablet, se collegati ad una rete Wi-Fi.

2.5 Applicazioni ibride

Come evidenziato nei precedenti paragrafi 2.3 e 2.4, sia le applicazioni native che le WebApp hanno notevoli punti sia a favore che contrari. Scegliere una qualsiasi delle due soluzioni richiede necessariamente di scendere a compromessi, o dal punto di vista delle prestazioni, o da quello della complessità di sviluppo. Viene quindi naturale chiedersi se esistano soluzioni intermedie, ovvero che prendano da entrambe qualche aspetto positivo, e solo una parte degli aspetti negativi. La risposta a questa esigenza è data dalle cosiddette applicazioni ibride.

Un'applicazione ibrida è fondamentalmente una WebApp "avanzata". L'idea di base è quella di estendere le funzionalità delle WebApp che, come spiegato nel paragrafo 2.4, dispongono di un accesso molto limitato alle funzionalità del sistema operativo sul quale sono eseguite. Si tratta quindi di realizzare applicazioni native, utilizzando però i linguaggi di sviluppo web (soprattutto HTML, CSS e Javascript) invece di quelli specifici per le singole piattaforme [FER09]. Tuttavia, nei paragrafi 2.3 e 2.4 sono state analizzate le varie tecnologie di sviluppo, sottolineando come il mondo delle applicazioni native sia sostanzialmente separato da quello delle WebApp. Appare quindi ossimorica la commistione di linguaggi che le applicazioni ibride si propongono di concretizzare.

L'integrazione di questi mondi così diversi richiede l'ausilio di un framework intermedio, che funga da mediatore. Esistono diverse soluzioni sul mercato: la più utilizzata è Apache Cordova, che è gratuita e open-source, ma esistono anche versioni commerciali che offrono funzionalità aggiuntive, come ad esempio IBM Worklight, basato su Cordova. In questa sede verrà analizzato Cordova come caso esemplificativo per la realizzazione di applicazioni ibride.

2.5.1 Apache Cordova

Apache Cordova (logo in figura 2.3) è un framework per lo sviluppo di applicazioni ibride, creato originariamente da Nitobi con il nome di PhoneGap, e successivamente acquistato da Adobe [ADO11]. Nel 2011 è diventato open-source ed ha assunto la denominazione attuale (Apache Cordova). Attualmente esiste anche una versione personalizzata di Cordova chiamata con il vecchio nome "PhoneGap" e mantenuta da Adobe.

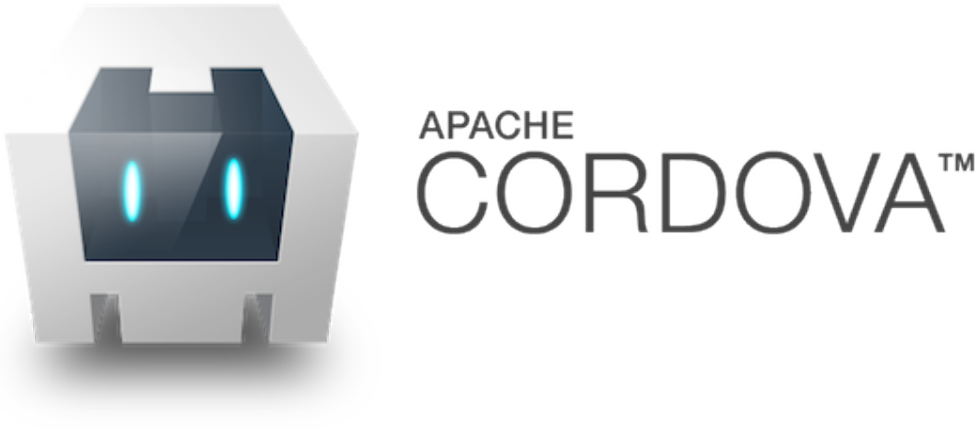

Figura 2.3 - Logo di Apache Cordova

Come accennato nella prima parte del paragrafo 2.5, Cordova consente di scrivere applicazioni mobili utilizzando i linguaggi web piuttosto che quelli nativi per le singole piattaforme. Non solo consente di utilizzare i linguaggi web, ma li estende con funzionalità specifiche dei sistemi operativi sottostanti, mascherando allo sviluppatore le specificità dei singoli sistemi. In questo modo il programmatore può utilizzare un'interfaccia unificata per accedere alle funzionalità a più basso livello del sistema in maniera trasparente. Questo è il motivo per cui le applicazioni realizzate con questa tecnologia sono dette "ibride": costituiscono una via di mezzo tra le WebApp e le applicazioni puramente native.

Un'applicazione sviluppata in Cordova utilizza HTML5 e CSS3 per il rendering dell'interfaccia grafica, in maniera web-like, mentre in Javascript è implementata la logica dell'applicazione. In particolare, un'applicazione Cordova è a tutti gli effetti un'applicazione nativa, e quindi installabile dall'utente sul proprio dispositivo, che al suo interno esegue una sorta di browser potenziato, all'interno del quale viene eseguito

il rendering grafico dell'applicazione e vengono eseguite le funzioni Javascript che possono utilizzare API implementate da Cordova, rendendo così accessibili funzioni non normalmente utilizzabili all'interno di un browser. Essendo l'ambiente di esecuzione Cordova stesso, viene superato il problema della frammentazione: soprattutto nei dispositivi mobili l'utente può installare browser diversi, che non implementano tutti necessariamente le stesse funzioni; utilizzando Cordova invece è garantita la compatibilità con Cordova stesso, per cui lo sviluppatore non deve preoccuparsi delle differenze specifiche dei dispositivi [INT12]. Cordova include funzionalità di base per l'accesso a sensori quali accelerometro, giroscopio, microfono, fotocamera e file system. È possibile l'utilizzo di plugin che implementano funzioni aggiuntive richiamabili tramite Javascript. Frequente è l'uso della libreria Javascript JQuery, che semplifica la stesura del codice.

Cordova supporta al momento una vasta gamma di dispositivi e sistemi operativi, per cui si può pensare che un'applicazione realizzata con questa tecnologia sarà eseguibile sulla quasi totalità dei dispositivi in commercio.

Poiché di fatto un'applicazione Cordova è installata sul dispositivo, nascono anche dei vantaggi collaterali per l'utente: a differenza delle applicazioni web, un'applicazione ibrida può essere interamente scaricata sul dispositivo, riducendo i trasferimenti di dati, che possono essere problematici sulle connessioni a consumo e/o a basse prestazioni. Inoltre, la disponibilità dell'applicazione direttamente negli store ufficiali delle varie piattaforme facilita all'utente l'installazione e l'applicazione di aggiornamenti. In realtà, per quanto riguarda quest'ultimo punto, l'immediatezza di aggiornamento non è comunque al livello di una WebApp, che si aggiorna necessariamente all'ultima versione ad ogni visita in maniera trasparente all'utente. Questo può portare alla presenza contemporanea di versioni diverse dell'applicazione sul mercato. Naturalmente lo sviluppatore non è tenuto a supportare tutte le versioni obsolete, e teoricamente può forzare l'utente ad installare sempre l'ultimo aggiornamento, pena la negazione dell'accesso all'applicazione. Questa scelta drastica può tuttavia essere controproducente, in quanto molti utenti vedono gli aggiornamenti come una scocciatura, e non come un miglioramento.

La realizzazione di applicazioni utilizzando questo framework, tuttavia, hanno prestazioni sensibilmente peggiori rispetto ad un'applicazione nativa, analogamente alle WebApp, anche se in misura minore. Per questo motivo l'impiego di tecnologie

ibride non è consigliabile in tutti quei contesti nei quali siano necessarie prestazioni elevate, specialmente dal punto di vista grafico. In realtà la perdita prestazionale è percepibile anche in applicazioni basilari, specialmente su dispositivi poco performanti, ma non solo: un utente esigente dal punto di vista delle prestazioni riuscirà a notare rallentamenti anche molto leggeri. Per questo motivo, nonostante le applicazioni ibride si avvicinino maggiormente alle applicazioni native dal punto di vista prestazionale, la loro adozione deve essere attentamente valutata in fase progettuale, in quanto i minori costi e complessità del progetto si pagano in termini di esperienza dell'utente. Addirittura, nel caso dei dispositivi Apple, il cui store presenta rigide regole sulla pubblicazione di applicazioni, le applicazioni ibride possono essere rifiutate qualora non raggiungano un livello minimo di prestazioni [TRI12].

Lo sviluppo di un'applicazione tramite Cordova non richiede l'utilizzo di un ambiente di sviluppo specifico: è sufficiente installare gli strumenti di Cordova (a riga di comando) e un qualunque editor di testo che supporti i linguaggi Web. In realtà l'uso di un IDE è consigliato in quanto consente di accedere ad un'organizzazione meglio strutturata del progetto, oltre a funzionalità che semplificano la stesura del codice. È possibile sviluppare anche in maniera esclusivamente testuale, ma è consigliato solo nel caso in cui si voglia portare avanti lo sviluppo su molteplici piattaforme in maniera parallela.

2.6 Scelta per il progetto

Nei precedenti paragrafi paragrafi del capitolo 2 sono stati analizzati la diffusione dei sistemi operativi e le possibili strade per lo sviluppo di un'applicazione: nativa, web, o ibrida. Alla luce delle osservazioni fatte si cercherà ora di identificare la soluzione migliore tra quelle disponibili ai fini del gioco specifico che si intende implementare.

Come analizzato nel primo capitolo, lo scopo del progetto è la realizzazione di un'applicazione rivolta agli studenti delle scuole medie. Si è visto come la diffusione di dispositivi mobili come smartphone e tablet sia stata considerevole negli ultimi anni, rendendone la presenza ormai pervasiva. Sebbene i dispositivi adatti all'esecuzione di un gioco siano ormai onnipresenti, il mercato è estremamente frammentato tra i vari dispositivi e sistemi operativi dei vari produttori, rendendo impossibile focalizzarsi solo su una piattaforma (o un insieme di esse) senza tagliare fuori gran parte dei potenziali

utenti. Questo, unito al fatto che le risorse per lo sviluppo del progetto sono limitate, propende a far scartare la strada delle applicazioni native. Inoltre, non essendo il gioco particolarmente esigente dal punto di vista grafico, non richiedendo rendering intensivo, indirizza ancora di più il progetto verso la strada delle WebApp e delle applicazioni ibride.

La principale differenza tra applicazioni ibride e WebApp, come visto nei paragrafi 2.4 e 2.5, è la non accessibilità da parte delle WebApp a molte funzionalità del dispositivo (specialmente sensori e archiviazione), oltre all'ambiente di esecuzione differente (un'applicazione Cordova piuttosto che un browser web generico). L'applicazione che si intende realizzare in questa sede non richiede l'accesso a funzionalità specifiche dei dispositivi sui quali viene eseguita, per cui una WebApp non sarebbe limitante da questo punto vista. Dopo un'attenta analisi si è tuttavia deciso comunque di percorrere la strada dell'applicazione ibrida, in quanto più facile da gestire per l'utente e in grado di garantire prestazioni leggermente migliori. Inoltre, realizzare l'applicazione come ibrida può consentire in futuro di aggiungere funzionalità attualmente non previste. Infine, in questo modo non è necessario testare la compatibilità dell'applicazione con i diversi browser dei diversi produttori.

2.7 Specifiche comuni a tutti i giochi

Il gioco realizzato nell'ambito di questa tesi non costituisce il progetto nella sua totalità. Quanto implementato in questa sede rappresenta una parte di un progetto più ampio, realizzato in maniera modulare e sviluppato nell'ambito di tirocini e tesi da parte di più studenti. Come accennato nell'introduzione, l'applicazione nella sua totalità sarà composta da un insieme di giochi matematici, ognuno dei quali realizzato da uno studente diverso, che verranno uniti all'interno di una macro-applicazione. Sebbene i giochi siano in gran parte indipendenti l'uno dall'altro, è opportuno definire alcune specifiche comuni a tutti, in modo da semplificare l'integrazione, sia dal punto di vista implementativo che dell'esperienza utente. Infatti, è importante mantenere alcuni aspetti di continuità tra un gioco e l'altro, rafforzando l'organicità del progetto. Questo da un lato consentirà un'unione dei giochi quanto più possibile indolore, dal punto di vista della compatibilità tanto a basso livello (sistema di passaggio e memorizzazione dei dati) quanto ad alto livello (interfaccia grafica simile). Le specifiche di base concordate sono le seguenti:

- tutti i giochi saranno configurabili tramite parametri che consentano di agire sulle seguenti variabili:
    - difficoltà della partita: regola il livello di gioco della singola partita, agendo su parametri quali ad esempio: complessità dei calcoli da eseguire, numero di possibili combinazioni di azioni eseguibili dall'utente, dimensione del campo di gioco. Il modo in cui il livello di difficoltà viene gestito è dipendente dal tipo di gioco stesso: ad esempio, nel caso del cruciverba numerico la maggiore difficoltà potrà essere data dalle maggiori dimensioni della griglia e dalla maggiore complessità dei calcoli matematici che l'utente dovrà risolvere; nel caso del Tangram dal numero di pezzi a disposizione dell'utente, dalla complessità della figura da costruire e dalle possibili rotazioni dei pezzi forniti (nel livello più semplice tutti i pezzi potrebbero essere già orientati correttamente, mentre procedendo con i livelli potrebbe essere necessario ruotarli); nel caso dei giochi di logica potrebbe aumentare progressivamente il numero di variabili di cui tenere conto, alcune delle quali potrebbero essere anche non utili al raggiungimento della soluzione.
    - punteggio dell'utente: come spiegato nel capitolo 1, una delle meccaniche di gioco utilizzate nella Gamification è un sistema di punteggio che costituisca una moneta virtuale all'interno del gioco, spendibile dall'utente per comprare benefici internamente al gioco. Di base potrebbe essere possibile anche acquistare questa valuta anche con denaro reale, come avviene nei giochi *free-to-play* (descritti con maggiore dettaglio nel paragrafo 2 del primo capitolo), ma l'obiettivo di questa applicazione non è la monetizzazione, quanto invece di incentivare i giovani allo studio della matematica. Introdurre una meccanica di tipo *free-to-play* si rivelerebbe con ogni probabilità controproducente.
    - set grafico e contesto di gioco: per rendere più accattivante il gioco implementando meccaniche di Gamification, si intende offrire la possibilità di eseguire gli stessi giochi con temi grafici

diversi. Saranno quindi previsti diversi set di immagini per comporre dinamicamente l'ambiente di gioco. Inizialmente si utilizzerà un set a tema zombie, ma in futuro se ne potranno aggiungere di qualunque tipo, eventualmente anche in seguito al primo rilascio dell'applicazione. In questo modo l'utente potrà scegliere il tema che più si addice ai suoi gusti, offrendo un'esperienza di gioco più appagante. Si potrebbe anche rendere alcuni temi non disponibili al momento dell'installazione dell'applicazione, ma sbloccabili in seguito al raggiungimento di determinati traguardi all'interno dei giochi. Anche questo aspetto si ricollega alla meccanica di Gamification basata su un sistema di ricompense, che incentivino l'utente a proseguire nell'utilizzo del gioco.

- tutti i giochi saranno sviluppati in ambiente Apache Cordova. In realtà si tratterà di WebApp che non fanno un uso di funzionalità specifiche dei singoli dispositivi, per cui sarebbe possibile eseguirle anche nativamente all'interno di un browser web. Si è scelta comunque questa via per i motivi evidenziati nei paragrafi 2.5 e 2.6. Si ipotizza inoltre che nella versione finale dell'applicazione possano essere utilizzate alcune di queste caratteristiche, in particolar modo l'archiviazione di dati sul dispositivo, per memorizzare i progressi degli utenti o altri parametri di configurazione. Ad ogni modo al momento non si è discussa nel dettaglio questa possibilità. La scelta di sviluppare il progetto come applicazione ibrida, tuttavia, rimane aperta a questa possibilità, rendendone semplice l'implementazione, qualora si scegliesse di sfruttare questa funzionalità.
- elaborazione eseguita interamente o quasi dai client. In questo modo non si rende necessario un server centrale potente, e si minimizzano i tempi di caricamento dovuti alla latenza e alle prestazioni della rete. D'altro canto l'esecuzione di algoritmi complessi lato client deve fronteggiare il problema della grande varietà di prestazioni dei dispositivi utilizzati dagli utenti, rendendo necessario sviluppare algoritmi sufficientemente performanti su ogni tipo di dispositivo. L'utilizzo di Javascript, inoltre,

gioca a sfavore delle prestazioni, in quanto le prestazioni saranno necessariamente inferiori rispetto al software nativo.

- scambio e passaggio di dati: ogni gioco dovrà gestire dati in input (ricevuti dall'applicazione principale al momento dell'avvio di una partita) e in output (restituzione in un formato standard dei punteggi ottenuti dall'utente e della moneta virtuale spesa). Questo dovrà essere realizzato in modo tale da consentire l'interoperabilità dei dati prodotti dai singoli giochi. L'applicazione principale si occuperà eventualmente di normalizzare i dati sia provenienti dai singoli giochi, sia direzionati verso essi. Un'ipotesi caldeggiata in fase di definizione delle specifiche è quella di utilizzare un RDBMS (Relational Database Management System) per gestire in maniera centralizzata i dati relativi agli account e ai punteggi dei singoli utenti. In questo modo si avrebbero parecchi vantaggi, primo fra tutti l'integrità dei dati: un database gestisce automaticamente gli accessi concorrenti mantenendo l'atomicità delle transazioni. Di conseguenza non si verrebbero a creare problemi di inconsistenza per le variabili ad accesso condiviso in scrittura (ad esempio se la quantità di monete di un utente variasse in un breve lasso di tempo da due dispositivi diversi, l'utente si ritroverebbe ad utilizzare un numero di monete diverso da quello effettivamente in suo possesso). Questo problema diventerebbe urgente qualora venissero implementate partite non singole (associate a non più di un solo giocatore contemporaneamente), ma condivise tra un gruppo di studenti, in cui il punteggio e i bonus vengono condivisi tra i membri del gruppo. In quel caso, se i valori relativi al punteggio e alla quantità di moneta disponibili venissero passati al gioco semplicemente all'avvio della partita, si creerebbe un lungo intervallo di tempo (tutta la durata della partita) all'interno del quale ogni membro del gruppo, ognuno con il proprio dispositivo, disporrebbe dell'intero ammontare di "denaro". In questo modo ognuno potrebbe spenderlo interamente, costituendo un abuso. In alternativa si potrebbe dividere in parti uguali l'ammontare di denaro in gioco tra i membri del gruppo all'inizio della partita. La soluzione sarebbe tuttavia meno elegante, e mancherebbe di scalabilità futura. Un

altro vantaggio nell'utilizzo di un database è quello di garantire un livello di sicurezza ed affidabilità maggiore rispetto alla memorizzazione dei dati puramente sui dispositivi degli utenti: i progressi degli utenti sarebbero salvati su un server remoto, rendendoli persistenti in seguito ad un cambio di dispositivo da parte dell'utente. Inoltre, in questo modo sarebbe possibile giocare da più dispositivi a seconda delle proprie preferenze (ad esempio da smartphone in mobilità e da computer a casa), proseguendo sempre la stessa partita, e non iniziandone una nuova per ogni dispositivo. In fase di definizione delle specifiche si è ipotizzato anche di centralizzare i temi grafici, inserendoli sempre all'interno di un database. In questo caso però i benefici sarebbero limitati, e dovrebbero far fronte anche ad una serie di svantaggi. In primo luogo i temi grafici sono costituiti da immagini, ovvero dati non strutturati, che mal si sposano con i database relazionali; in secondo luogo, se i temi fossero memorizzati su un server remoto, questo obbligherebbe l'utente a scaricarli sul proprio dispositivo con una frequenza piuttosto elevata, aumentando il traffico dati generato dall'applicazione e il tempo di caricamento, con riferimento ad uno degli svantaggi delle WebApp (paragrafo 2.4). È preferibile quindi memorizzare questi dati all'interno dell'applicazione stessa, in modo da renderli disponibili localmente. In questo modo l'introduzione di nuovi temi richiederebbe necessariamente un aggiornamento dell'applicazione dallo store, e non potrebbe essere veicolato in maniera trasparente all'utente. Tuttavia, poiché si tratta semplicemente di aspetti grafici, e non di funzionalità dell'applicazione, il mancato aggiornamento da parte dell'utente non costituisce un problema serio, in quanto nello scenario peggiore quell'utente avrebbe semplicemente meno temi grafici a disposizione. Comunque, gli aggiornamenti delle funzionalità devono essere necessariamente veicolati tramite store a causa della scelta dello sviluppo come applicazione ibrida, per cui è auspicabile che periodicamente l'utente esegua l'aggiornamento.

2.8 Architettura di rete dell'applicazione

Basandosi anche sulle specifiche comuni descritte nel paragrafo 2.7, è stato realizzato uno schema all'interno del quale è rappresentata una possibile implementazione dell'architettura di rete (figura 2.4). Non è al momento certo che questa sarà del tutto coincidente con l'implementazione finale.

Dal punto di vista dei client, ovvero gli utenti dell'applicazione, l'accesso è possibile da un'ampia gamma di dispositivi, a causa della scelta di realizzare l'applicazione come ibrida. Nello schema sono presenti tre tipologie di client: computer desktop, computer portatile e smartphone. Per utilizzare tutte le funzionalità del gioco sarà necessaria una connessione ad Internet, anche se per specifiche modalità di gioco potrebbe non essere necessario (ad esempio per una partita singola senza salvare i progressi su un server remoto). All'utente è lasciata massima libertà sul tipo di connessione, in quanto l'applicazione non fa un uso intensivo del traffico dati: potrà collegarsi alternativamente tramite connessione cablata, Wi-Fi, o tramite la rete mobile 3G o 4G dell'operatore.

Dal punto di vista del server, in questo schema si è valutato lo scenario in cui è presente un database remoto che gestisce i dati degli utenti (credenziali per l'autenticazione, punteggi, progressi e quantità di moneta interna al gioco disponibile). Il database sarà eseguito sotto forma di applicazione all'interno di un server sempre attivo. A questo scopo non è necessario che il server sia particolarmente performante, in quanto dovrà gestire una quantità limitata di transazioni per unità di tempo, almeno inizialmente. Potrebbe essere quindi sensato affittare un server virtuale tramite un provider che offra questo tipo di servizio, come ad esempio DigitalOcean [DIO15] o Amazon AWS (Amazon Web Services) [AMA15], con riferimento principalmente al servizio Elastic Compute Cloud (EC2), che offre macchine virtuali. Inoltre, utilizzando un provider di servizi cloud, in caso di necessità si potrebbero potenziare le risorse allocate, per seguire le necessità future dell'applicazione, come l'aumento del numero di utenti e l'introduzione di nuove funzionalità.

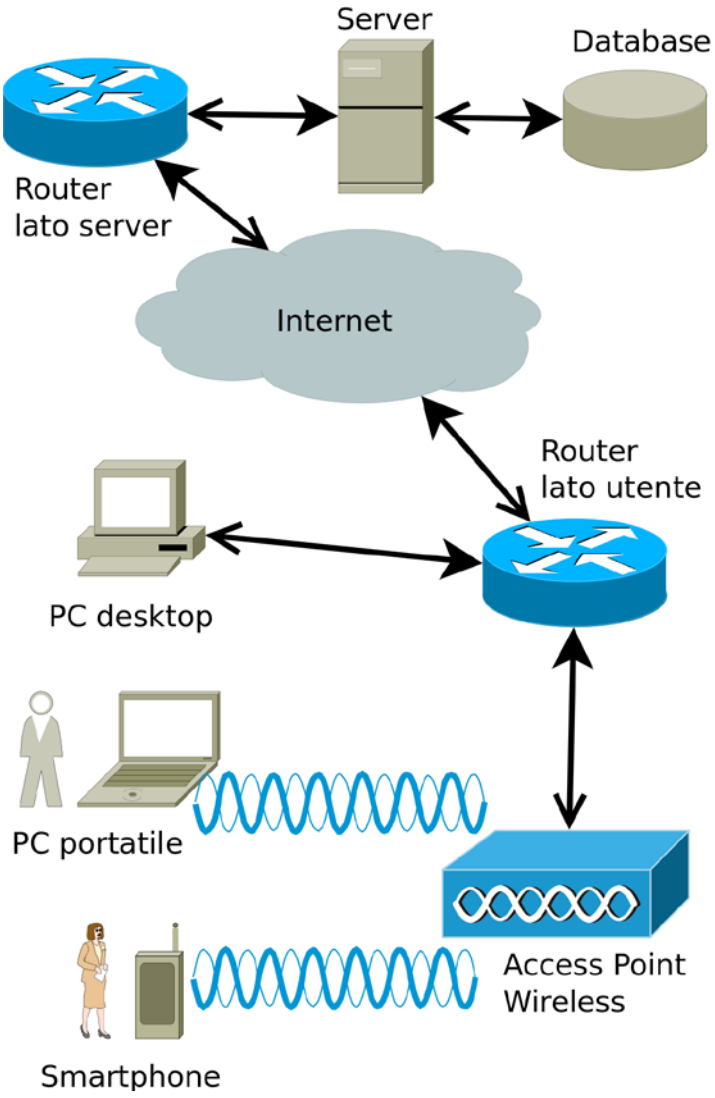

Figura 2.4 - Architettura di rete dell'applicazione

3 Implementazione

In questo capitolo verranno descritte le scelte implementative del progetto, basate sulle specifiche descritte nel capitolo 2. Verranno inoltre discusse alcune esigenze implementative sorte successivamente alla fase progettuale, principalmente per quanto riguarda l'efficienza e i costi computazionali.

3.1 Progetto implementato

Nel capitolo precedente sono stati definiti alcuni parametri generici sul funzionamento del macro-progetto. In particolare, la macro-applicazione raccoglie un insieme di giochi matematici e ne coordina le attività comuni (lancio dei giochi, raccolta di dati). Nell'ambito di questa tesi è stata implementata una sola parte del tutto, ovvero uno dei giochi matematici. Il gioco che sarà implementato è un cruciverba numerico, ovvero un classico cruciverba composto da una griglia all'interno della quale inserire dei termini, aiutandosi con le definizioni fornite. Tuttavia, i termini sono costituiti esclusivamente da cifre decimali, al posto delle lettere dell'alfabeto.

Lo schema del cruciverba sarà composto da definizioni orizzontali e verticali, e la generazione sarà automatica tramite un algoritmo, il quale attingerà ad un database delle definizioni in base ad alcuni parametri, e produrrà uno schema risolvibile e corretto. L'elaborazione, come da specifiche descritte nel paragrafo 2.7, avverrà interamente client-side, ovvero sul dispositivo dell'utente, rendendolo di fatto un gioco stand-alone, minimizzando il carico su un eventuale server centrale.

Come dettagliatamente spiegato nel capitolo precedente, il gioco sarà realizzato sotto forma di applicazione ibrida utilizzando il framework Apache Cordova. Di conseguenza sarà realizzato utilizzando quasi esclusivamente tecnologie web, quali HTML5, CSS3 e Javascript.

Poiché la logica dell'applicazione dovrà essere implementata in Javascript, nell'implementazione verrà utilizzato anche codice di terze parti, in particolare la libreria Javascript JQuery [JQU15], che fornisce una lunga serie di utilità che semplificano la stesura del codice. Sarà utilizzata anche parte di un progetto open source di nome *js-xwords* disponibile su Github e rilasciato con licenza Apache versione 2.0 [WAY14], specialmente per quanto riguarda la parte algoritmica relativa alla generazione degli schemi dei cruciverba.

3.2 Algoritmo di generazione cruciverba

L'algoritmo di generazione dei cruciverba è interamente implementato in Javascript, ed è in gran parte basato sul progetto open source *js-xwords* [WAY14], già citato nel precedente paragrafo. Si è cercato di utilizzare quanto più possibile il paradigma orientato agli oggetti, in modo da rendere il codice il più possibile chiaro e mantenibile. In particolare, si è fatto un uso intensivo del concetto di incapsulamento, in modo da massimizzare il riuso per l'implementazione di molteplici funzionalità, eventualmente anche in futuro. Si è inoltre cercato di seguire quanto più possibile il pattern object-oriented "Model View Controller" (MVC).

L'algoritmo di generazione degli schemi prende in input parametri quali:

- la dimensione della griglia del cruciverba da produrre (regola la grandezza dello schema generato e del font), rilevante soprattutto per la visualizzazione su schermi di differenti dimensioni e formato;
- il numero di righe e di colonne, da cui dipendono direttamente il numero di definizioni inseribili nello schema e il formato dello schema stesso (nel caso siano coincidenti sarà quadrato, altrimenti più o meno rettangolare);
- la percentuale di facilitazione, ovvero il numero di caselle già svelate al momento della generazione.

L'algoritmo gestisce il cruciverba come una griglia, rappresentata come un array di array monodimensionali, e accessibile con doppio indice. I termini da inserire nello schema sono memorizzati come array di "parole". Le parole sono oggetti personalizzati creati per memorizzare ogni termine da inserire, contenenti il termine stesso e un insieme di proprietà associate, quali il posizionamento all'interno dello schema.

Una serie di metodi pubblici espone alla procedura chiamante (in questo caso l'applicazione principale) i dati sugli schemi generati, quali il numero di parole presenti nello schema, le parole inutilizzate (nel caso non esista un modo per inserirle all'interno di uno specifico schema), lo schema stesso (per accedere ai singoli valori delle celle, utile in fase di risoluzione del cruciverba), le funzionalità di creazione degli schemi. Tutti i metodi che non necessitano obbligatoriamente di essere chiamati dall'esterno sono stati raccolti a parte, su modello dei metodi privati.

3.2.1 Ottimizzazioni ed efficienza

La generazione di tutti i possibili schemi per un cruciverba, dato un insieme di definizioni, è un'operazione la cui complessità computazionale aumenta in maniera esponenziale con il numero e la lunghezza delle definizioni stesse. Di conseguenza la generazione potrebbe richiedere molto tempo nel caso di schemi di medie e grandi dimensioni. Questo può essere un problema specialmente sui dispositivi portatili come smartphone e tablet, che dispongono di norma di prestazioni molto più limitate rispetto ai computer: se un esecuzione dell'algoritmo su un computer performante può richiedere diverse decine di secondi, su uno smartphone di fascia bassa potrebbe richiedere minuti. Un tempo nell'ordine delle decine di secondi è comunque troppo lungo anche su computer, in quanto l'utente percepisce in modo molto negativo le attese. Inoltre, non si tratta solo di un problema di attesa dell'utente, ma anche di consumo energetico, particolarmente rilevante nei dispositivi mobili. Per questi motivi, i tempi di esecuzione dell'algoritmo rivestono un ruolo fondamentale all'interno del gioco, in quanto influenzano in maniera diretta e marcata la qualità complessiva percepita. Essendo da specifiche l'elaborazione eseguita interamente lato client, non è possibile mitigare il problema acquistando hardware potente lato server. Di conseguenza è opportuno mettere in pratica alcuni accorgimenti per velocizzare l'algoritmo.

Innanzitutto, una semplice osservazione che può essere avanzata è la seguente: ai fini di una singola partita, non sarà necessario individuare tutti i possibili schemi ottenibili dato un insieme di parole in input, in quanto ad una partita è associato un solo schema. Di conseguenza sarà sufficiente generare un solo schema, purché rispetti alcuni vincoli di qualità. In particolare, è possibile che iterando l'algoritmo una sola volta si giunga ad uno schema in cui non tutte le definizioni in input vengono utilizzate.

Si preferirà quindi iterare l'algoritmo più volte al fine di generare non una singola soluzione, ma un insieme, dal quale sarà possibile scegliere la migliore, ovvero una di quelle in cui tutte le parole sono state inserite oppure, nel caso non esista una soluzione di questo tipo, una di quelle in cui il numero di parole non inserite è minimo. Rimane il problema di stabilire il numero di esecuzioni dell'algoritmo, in quanto iterandolo un numero limitato di volte si potrebbe ottenere una soluzione poco soddisfacente, mentre iterandolo troppe volte si otterrebbe da un lato una soluzione migliore, ma dall'altro comporterebbe un aumento inaccettabile dei tempi. Si tratta quindi di scendere ad un compromesso, dove da un lato si hanno il tempo di esecuzione e il consumo di risorse, e dall'altro la rapidità di esecuzione e l'esperienza dell'utente.

Per generare un buon cruciverba non sarà obbligatorio individuare la soluzione ottima (questo richiederebbe di calcolare tutte le possibili combinazioni), ma ne è sufficiente una che sia abbastanza vicina. Questo è particolarmente vero nel caso specifico di questa applicazione, in cui il cruciverba è di tipo numerico, e la presenza di una definizione in più o in meno non influisce né sull'esperienza dell'utente, né sulla qualità dell'apprendimento. In questo modo la probabilità di individuare una soluzione considerata valida ai fini del gioco entro pochi secondi dall'inizio dell'esecuzione dell'algoritmo è molto alta. Questo genere di problema è trattato anche nell'ambito della ricerca operativa: quando si deve risolvere un problema la cui soluzione ottima richiederebbe costi e tempi eccessivi, in alcuni casi si può preferire limitarsi ad una soluzione buona (ma non ottima), che però richiede molte meno risorse per essere calcolata.

Un altro aspetto che condiziona direttamente il tempo richiesto prima di trovare una soluzione valida è il rapporto tra il numero di definizioni e la dimensione dello schema: fissato un numero di definizioni da inserire nello schema, quanto più la dimensione massima dello schema sarà alta, tanto più saranno gli schemi validi generati; al contrario, riducendo la dimensione dello schema, ma mantenendo lo stesso numero di definizioni, sarà progressivamente sempre più difficile posizionarle tutte. Si raccomanda quindi di consentire la creazione di schemi sufficientemente ampi da ospitare tutti i termini che si desidera inserire. In ogni caso, è bene non impostare dimensioni eccessive, in quanto se da un lato l'esecuzione dell'algoritmo sarà più rapida, dall'altro le maggiori dimensioni dello schema potrebbero diventare poco gestibili su smartphone, caratterizzati da schermi di piccole dimensioni (rispetto ad un

tablet o ad un computer). Inoltre, utilizzare uno schema sovradimensionato porterà alla generazione di un cruciverba in gran parte vuoto, peggiorando l'estetica del risultato.

Resta un ultimo dettaglio da definire, ovvero il criterio di interruzione dell'algoritmo. Si è notato empiricamente come le probabilità di generare uno schema valido entro pochi secondi dal lancio dell'algoritmo siano superiori al 99%, a patto di rispettare i vincoli indicati sul rapporto tra il numero di definizioni e la dimensione dello schema. Si è quindi deciso di limitare le iterazioni dell'algoritmo ad un tempo di cinque secondi (stabilito empiricamente). Allo scadere di questo tempo, viene scelto lo schema migliore tra quelli generati, indipendentemente da quanti ne siano stati effettivamente prodotti. Su un dispositivo performante, quindi, la qualità degli schemi prodotti sarà leggermente migliore per tutti quei cruciverba generati sfruttando interamente il tempo a disposizione di cinque secondi. Il tempo impostato pare comunque adeguato anche sui dispositivi con prestazioni ridotte. In ogni caso, qualora ci si rendesse conto che per ottenere schemi migliori è necessario aumentare questo tempo, lo si potrebbe facilmente modificare.

Esiste potenzialmente un altro tipo di ottimizzazione: sfruttare la potenza computazionale inutilizzata del dispositivo per generare in anticipo una serie di schemi e memorizzarli in una cache. In questo modo non si avrebbe il vincolo di tempo per la generazione, in quanto sarebbe eseguita in maniera asincrona rispetto alle attività dell'utente, e al momento del lancio di una nuova partita sarebbe sufficiente caricare un cruciverba già generato. Tuttavia, questa soluzione, al fine di essere efficace, richiederebbe il calcolo di un elevato numero di schemi, per fare fronte a tutti i possibili tipi di partita che l'utente potrebbe voler giocare. Di conseguenza, un gran numero degli schemi generati rimarrebbe inutilizzato, avendo così sprecato potenza computazionale ed energia (situazione da evitare soprattutto sui dispositivi alimentati a batteria). Per questo motivo si è deciso di scartare questa soluzione.

Diverso sarebbe stato il discorso se i cruciverba venissero generati lato server, e poi passati ai client attraverso la rete: in questo caso, la generazione in anticipo degli schemi potrebbe essere un'ottima soluzione, in quanto l'elevato numero di utenti dell'applicazione farebbe sì che ogni schema venga utilizzato in tempi ragionevolmente brevi dal momento della generazione. Inoltre, questa soluzione, avrebbe consentito di ridurre il carico di picco sul server, ad esempio pianificando la generazione dei cruciverba nelle ore notturne, quando il carico è minore.

3.2.2 Test dell'algoritmo

Verranno ora generati degli schemi a partire da un insieme prefissato di definizioni, ma variando i parametri di generazione quali dimensioni orizzontali e verticali dello schema e percentuale di facilitazione. Verranno inoltre generati più cruciverba con gli stessi parametri, per mostrare come di volta in volta lo schema generato sia diverso. Le definizioni da inserire nello schema in questo test saranno prefissate (si ricorda che ogni definizione è composta dalla risposta da inserire nello schema e dall'indizio, separati da una virgola; ad ogni riga corrisponde una definizione):

```
29,3^3+2
146,85+37+24
9864,12*15*42+2304
276,523-247
436,5232/12
4500,50*90
130,70+64+27-31
2506,1253*2
7409,74*1000+9
1134,324*7/2
101,10000/1000*10+1
512,2^9
```

Notare come nelle definizioni non sia presente alcun riferimento alla loro posizione all'interno dello schema, nemmeno per quanto riguarda l'orientamento orizzontale o verticale, in quanto queste informazioni sono stabilite di volta in volta dall'algoritmo di generazione dei cruciverba, e fissarle a priori ridurrebbe sensibilmente la varietà degli schemi generati.

3.2.2.1 Test numero 1

In questo primo test si imposterà una griglia di dimensioni 12x12, volutamente molto grande in relazione all'esiguo numero di definizioni, e si itererà più volte l'algoritmo per visualizzare i diversi schemi generati. Per ogni iterazione sarà mostrato a sinistra lo schema in bianco (pronto per essere compilato), e a destra quello contenente la soluzione.

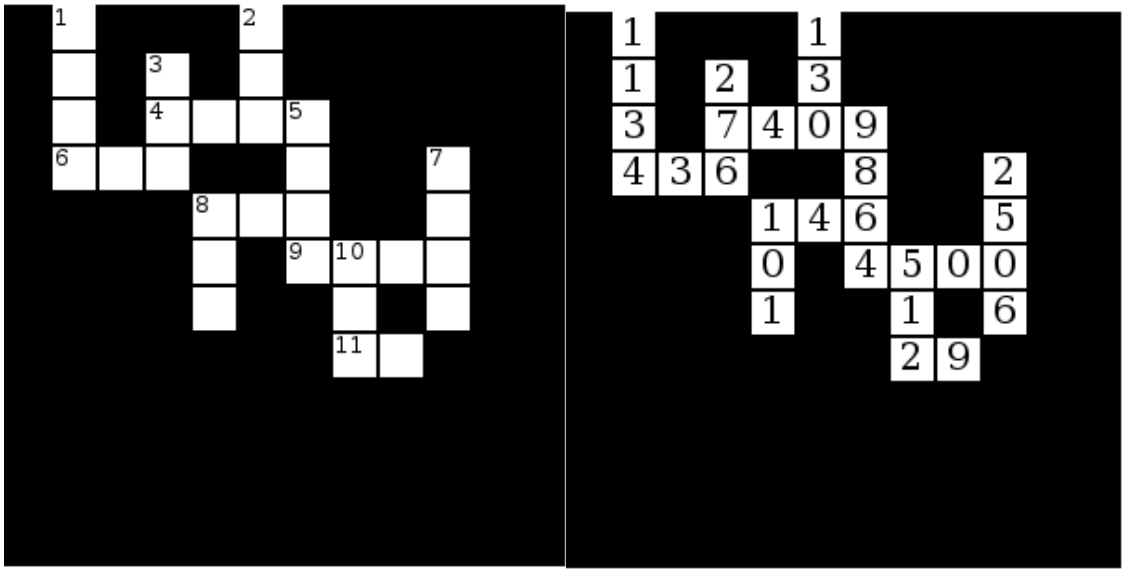

Figura 3.1 - Test algoritmo 1 – prima iterazione

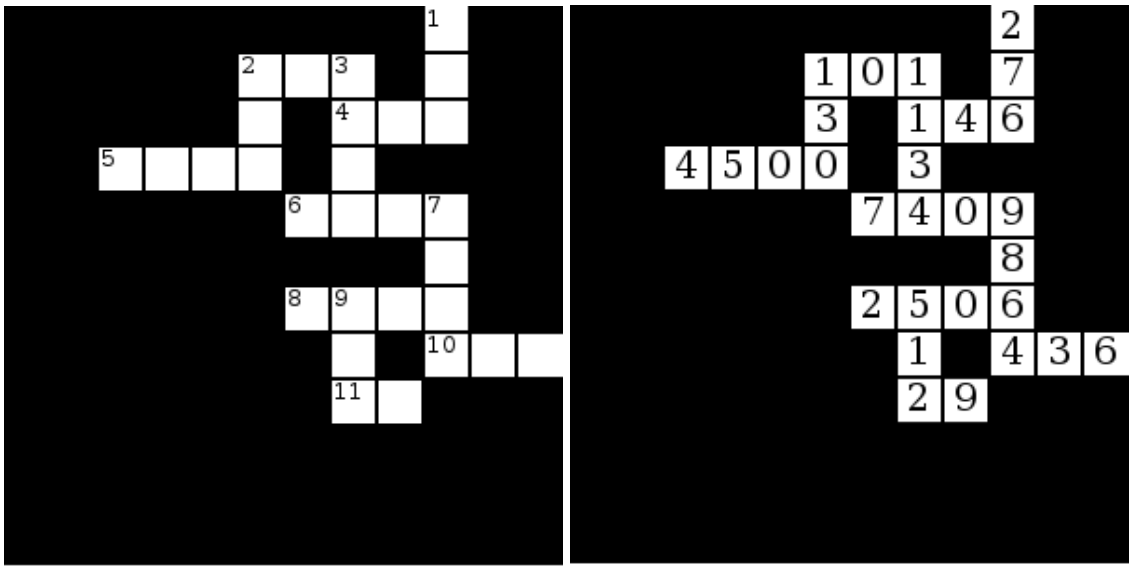

Figura 3.2 - Test algoritmo 1 – seconda iterazione

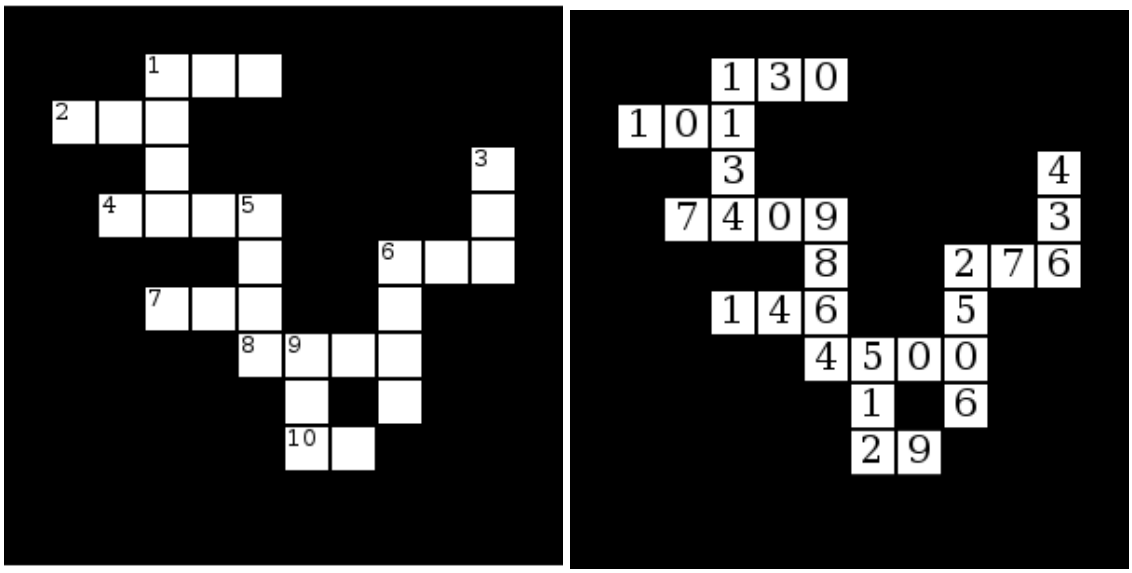

Figura 3.3 - Test algoritmo 1 – terza iterazione

A scopo esemplificativo sono state mostrate tre iterazioni dell'algoritmo. I possibili cruciverba generabili sono tuttavia un numero maggiore. Poiché le dimensioni dello schema sono elevate rispetto al numero di definizioni da inserire, l'algoritmo non incontra particolari difficoltà nel collocamento delle definizioni all'interno dello schema. Per questo motivo, la generazione del cruciverba è molto rapida.

3.2.2.2 Test numero 2

In questo test verranno ridotte progressivamente le dimensioni massime della griglia, sia orizzontali che verticali, per analizzare come l'algoritmo si comporta al restringersi dei vincoli.

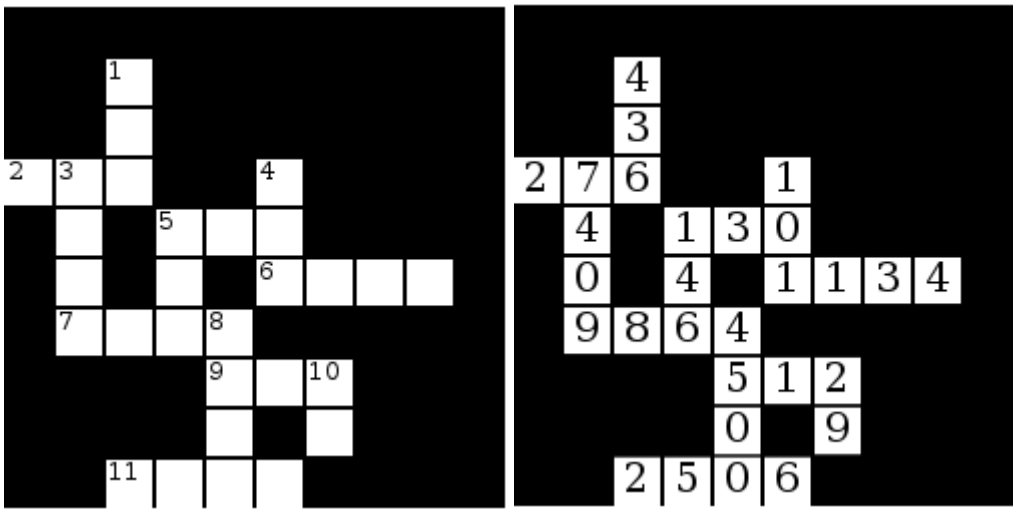

Figura 3.4 - Test algoritmo 2 – Griglia 10x10

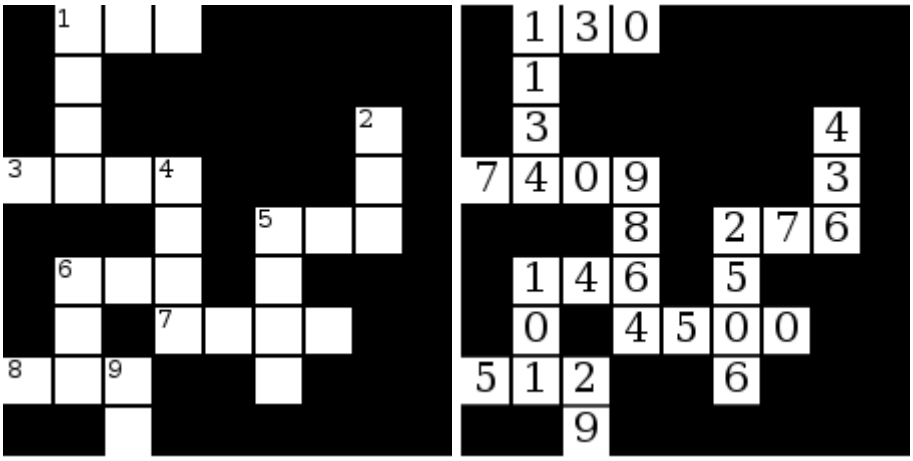

Figura 3.5 - Test algoritmo 2 – Griglia 9x9

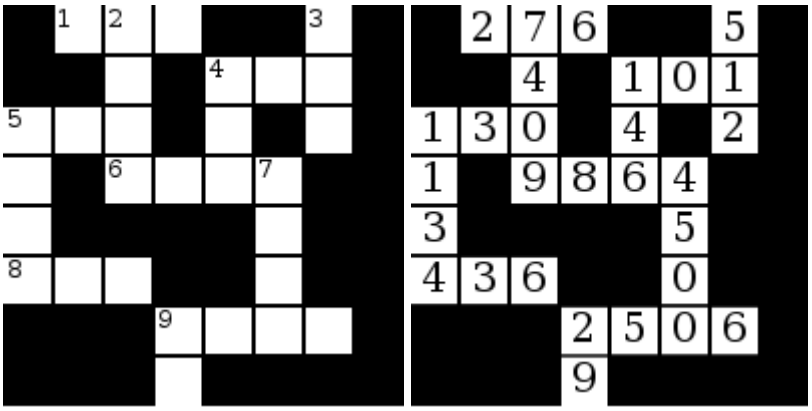

Figura 3.6 - Test algoritmo 2 – Griglia 8x8

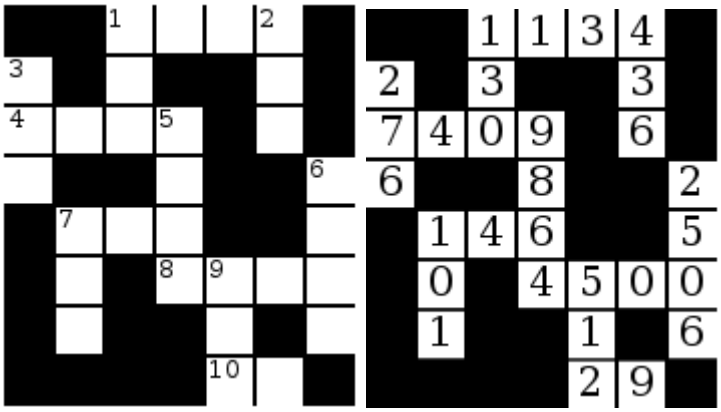

Figura 3.6 - Test algoritmo 2 – Griglia 7x8

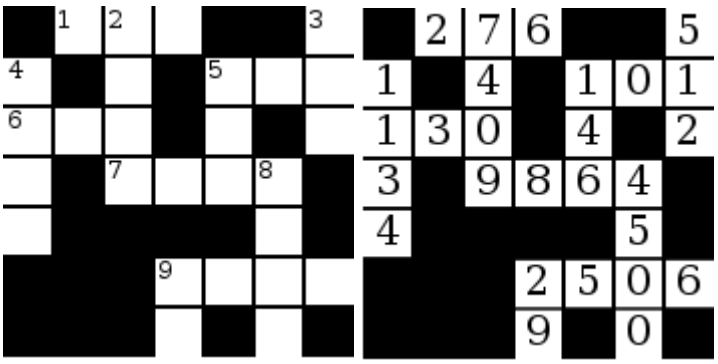

Figura 3.6 - Test algoritmo 2 – Griglia 7x7

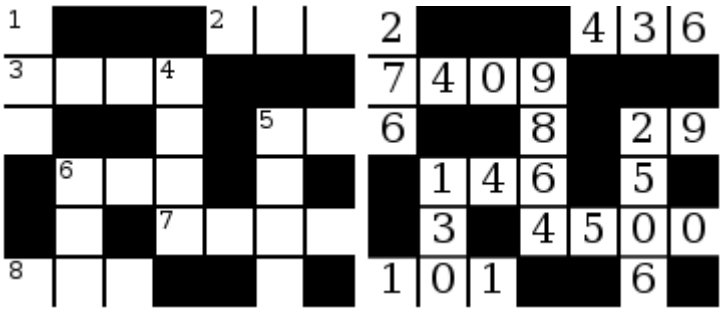

Figura 3.7 - Test algoritmo 2 – Griglia 7x6

Con il diminuire delle dimensioni della griglia, si nota come il tempo necessario all'algoritmo per generare uno schema valido aumenti progressivamente. Tuttavia, il tempo in più richiesto rimane sotto controllo, ed entro limiti accettabili, fintanto che esiste un modo per inserire tutte le definizioni in uno schema della dimensione data.

Diverso è, invece, se non è materialmente possibile inserirle tutte: in questo caso l'algoritmo itererà alla ricerca di una soluzione ottima finché non scadrà il tempo stabilito di cinque secondi, al termine del quale sceglierà una delle soluzioni migliori tra quelle trovate. È questo il caso degli schemi generati in figura 3.6 e 3.7: nei casi precedenti è sempre stato possibile inserire tutte le definizioni, ma non negli ultimi due, nei quali è stato necessario scartare rispettivamente tre e cinque definizioni. Questa situazione è considerata indesiderata. Sarà quindi necessario calibrare le dimensioni dello schema in relazione al numero di definizioni, al fine di non incorrere in tempi di elaborazione lunghi e soluzioni che non considerano tutte le definizioni (pur essendo corrette).

È interessante sottolineare come nel caso in cui l'algoritmo non individui una soluzione comprendente tutte le definizioni entro il tempo limite, questo non significa necessariamente che una soluzione di questo tipo non esista. Infatti, nel caso non sia

possibile testare tutte le possibili combinazioni di collocamento delle definizioni entro il tempo limite, l'algoritmo terminerà comunque e considererà valida la migliore soluzione parziale trovata. Tuttavia, come osservato nel paragrafo 3.2.1, l'attesa è percepita in maniera molto negativa da parte dell'utente, per cui si preferisce privilegiare la rapidità di esecuzione. Inoltre, se le dimensioni dello schema selezionate non sono eccessivamente ridotte, si potrà ragionevolmente affermare che una soluzione ottima sarà individuata prima dello scadere del tempo limite di cinque secondi.

Infine, ridurre eccessivamente le dimensioni dello schema, anche nei casi in cui una soluzione ottima viene individuata, comporta una notevole riduzione di entropia, in quanto le possibili combinazioni sono più limitate. Di conseguenza, gli schemi generati saranno meno diversificati.

3.2.2.3 Test numero 3

In questo test saranno applicati differenti livelli di facilitazione, ovvero si varierà la quantità di numeri già svelati, ad uno schema di dimensione 8x8.

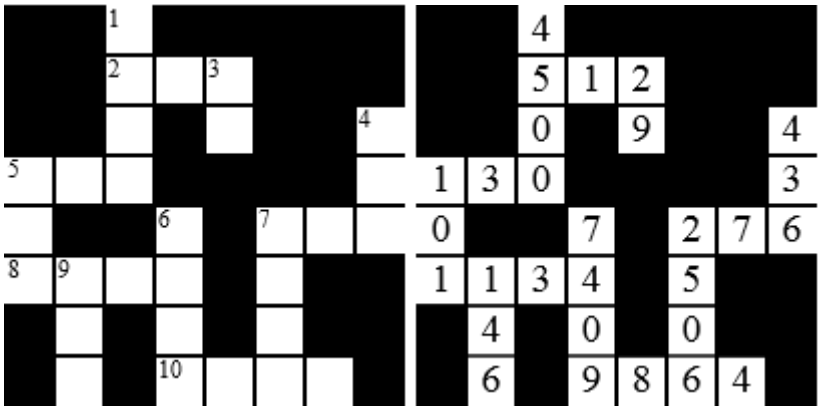

Figura 3.8 - Test algoritmo 7 – Facilitazione 0%

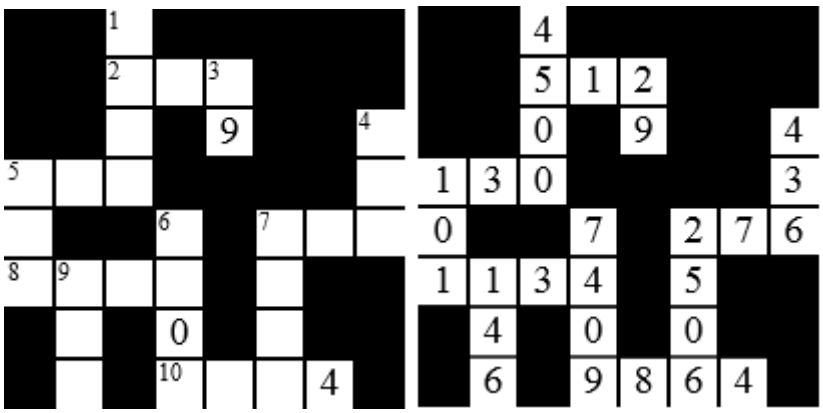

Figura 3.9 - Test algoritmo 7 – Facilitazione 10%

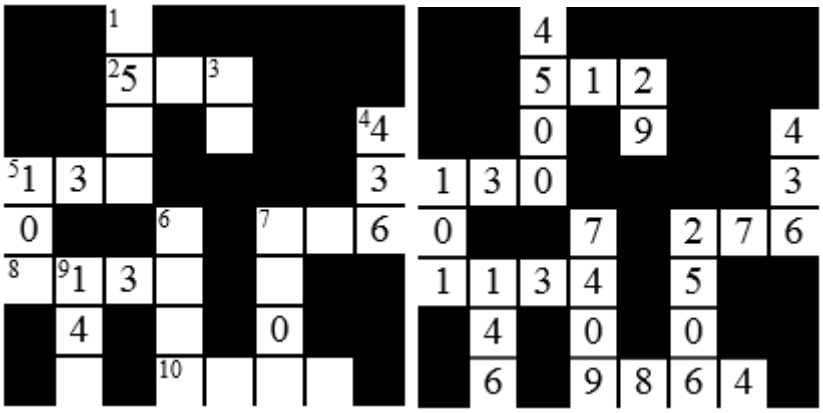

Figura 3.10 - Test algoritmo 7 – Facilitazione 30%

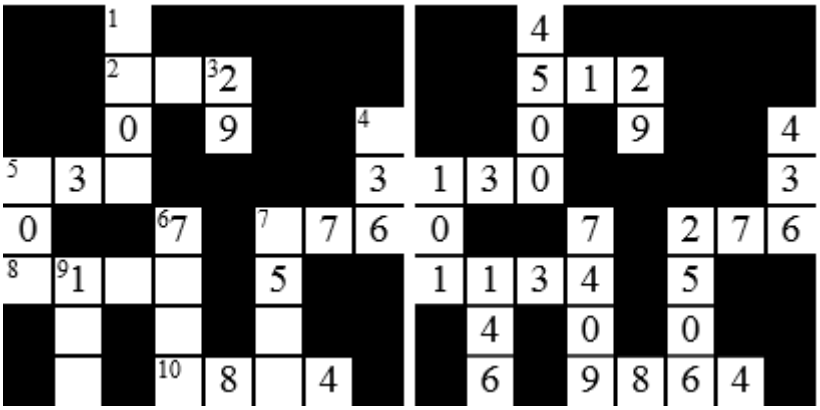

Figura 3.11 - Test algoritmo 7 – Facilitazione 50%

I caratteri svelati sono scelti casualmente dall'algoritmo, per cui anche applicando la stessa percentuale di facilitazione allo stesso schema si ottengono risultati diversi di volta in volta. Questo contribuisce all'entropia generale dell'algoritmo, generando schemi variabili non solo nel posizionamento delle definizioni, ma anche dei suggerimenti.

3.3 Gestione delle definizioni

Descritto l'algoritmo, verrà ora illustrato lo schema di memorizzazione delle definizioni dei cruciverba. Con il termine "definizioni" si intende l'insieme delle coppie risposta-domanda che compongono uno schema. Essendo questo un cruciverba di tipo numerico, le risposte saranno composte da cifre decimali, mentre le domande da operazioni matematiche.

Essendo il gioco rivolto ai ragazzi delle scuole medie, le definizioni saranno inerenti a temi trattati dai loro programmi scolastici. Lo schema è tuttavia estendibile per includere qualunque tipo di operazione, rendendo in questo modo possibile

ampliare il pubblico di utenti anche ai bambini delle scuole elementari (inserendo calcoli matematici più semplici) o ai ragazzi delle superiori (potenzialmente ogni definizione potrebbe richiedere di risolvere un'equazione, e così via). Non solo, l'algoritmo di generazione dei cruciverba supporta tutti i caratteri alfanumerici, rendendo possibile anche la generazione di cruciverba tradizionali basati sulle lettere dell'alfabeto. Potrebbe essere quindi utilizzato anche per l'apprendimento di altre materie, oltre alla matematica.

3.3.1 Efficienza nell'archiviazione e generazione delle definizioni

Essendo il cruciverba di tipo numerico, è possibile percorrere due strade nella generazione di uno schema per quanto riguarda il reperimento delle definizioni. Entrambe presentano benefici e costi.

- Memorizzare un grande insieme di definizioni all'interno di un archivio e, al momento della generazione di un nuovo schema, sorteggiarne un sottoinsieme casuale, a partire dal quale verrà generato lo schema. Questa soluzione consente di risparmiare notevolmente sul costo computazionale, in quanto le definizioni sono date a priori, non rendendo necessario generarle e calcolarle ogni volta. Scegliendo questa modalità, la varietà degli schemi è data dal modo in cui le definizioni vengono sorteggiate e da come vengono inserite all'interno dello schema di volta in volta. Questo metodo è efficace se il numero di definizioni presenti nell'archivio è sufficientemente elevato da introdurre negli schemi generati abbastanza entropia da rendere praticamente impossibile all'utente riconoscere uno schema ricorrente nei cruciverba generati. In altre parole, l'utente non deve percepire la presenza di un algoritmo "meccanico" che genera cruciverba troppo simili.
- Generare le coppie risposta-domanda in maniera dinamica, senza memorizzare dati in locale, utilizzando un apposito algoritmo. In questo caso non si pone il problema di preparare un database di definizioni a priori. Tuttavia, la complessità ricade interamente sull'algoritmo di generazione delle definizioni. Affinché sia efficace, deve essere estremamente articolato, in modo da generare schemi sempre diversi, sia

nelle definizioni contenute, sia nella resa grafica (ovvero il modo in cui le definizioni vengono incrociate tra loro). Questa seconda opzione è decisamente più complicata da gestire per il programmatore, e comporta un notevole aumento del costo computazionale per il client. Questo potrebbe essere un problema critico specialmente sui dispositivi a basse prestazioni, in quanto il tempo di elaborazione dovuto alla generazione delle definizioni si aggiungerebbe a quello già esistente per il collocamento delle definizioni nello schema (descritto più dettagliatamente nel paragrafo precedente). Esiste un ulteriore aspetto negativo: le definizioni generate, essendo casuali, non saranno necessariamente significative, ovvero alcune operazioni potrebbero essere eccessivamente complesse o, al contrario, troppo semplici, rendendo molto difficile stimare il grado di complessità di un cruciverba generato, e di conseguenza sottoporre all’utente uno schema della complessità richiesta.

Analizzati gli aspetti a favore e contrari di entrambe le soluzioni, si è deciso di implementare la prima soluzione in quanto, se correttamente realizzata, consente comunque di ottenere schemi molto variegati e difficilmente distinguibili da quelli prodotti adottando la seconda strategia. A scopo di ricerca si tenterà comunque di realizzare un prototipo di algoritmo per la seconda opzione, al fine di valutarne fattibilità e prestazioni, confrontate con la prima soluzione.

3.3.2 Formato di archiviazione, prestazioni e sicurezza

Poiché la logica applicativa del gioco è implementata in Javascript, anche le definzioni saranno archiviate in file Javascript. Ogni definizione viene memorizzata in una riga contenente prima la risposta e poi la domanda, separate da una virgola. Quando l'algoritmo di generazione cruciverba viene lanciato, verrà prelevato un sottoinsieme di definizioni in base ai parametri scelti. Poiché il numero di definizioni in archivio potrà essere elevato, ai fini di migliorare le prestazioni è necessario mettere in pratica alcune ottimizzazioni. Infatti, memorizzare tutte le definizioni in un unico archivio comporterebbe un carico di lavoro elevato, e richiederebbe una maggiore quantità di memoria RAM. Appare quindi sensato dividere le definizioni in più parti,

raggruppandole per caratteristiche, allo scopo di eliminare la necessità di controllare tutte le definizioni ad ogni generazione di un nuovo cruciverba.

Si può ipotizzare di raggruppare le definizioni per livello di difficoltà (vedere anche paragrafo 3.4): in questo modo nella generazione di uno schema semplice sarebbe necessario attingere solo ad un sottoinsieme delle definizioni; analogamente per uno schema di livello medio o difficile. Un ulteriore raggruppamento potrebbe essere fatto in base al tema delle definizioni che, nel caso di questo cruciverba matematico, corrisponde agli argomenti del programma scolastico. Così facendo sarebbe possibile generare cruciverba sui soli argomenti trattati a lezione, e aggiungerne di nuovi a mano a mano che vengono svolti.

Esiste una possibile vulnerabilità minore di sicurezza derivante dalla memorizzazione delle definizioni in un archivio: essendo Javascript un linguaggio interpretato, e non compilato, ed essendo le definizioni memorizzate come testo all'interno di uno o più file, è teoricamente possibile che un utente con buone conoscenze informatiche sia in grado di estrarre dal pacchetto dell'applicazione il database delle soluzioni, e di sfruttarlo per “barare” nella risoluzione dei cruciverba. Tuttavia, una situazione di questo tipo è decisamente improbabile per più motivi: da un lato, è improbabile che molti dei ragazzi target del gioco possiedano le conoscenze necessarie per realizzare questo scenario; dall'altro, le definizioni presenti non saranno particolarmente complesse, per cui il tempo richiesto dallo svolgimento dei calcoli in maniera corretta non potrebbe essere ridotto più di tanto dalla conoscenza del contenuto del database; infine, sarebbe necessario lavorare contemporaneamente su due dispositivi, l'uno con il gioco in esecuzione, l'altro con il database aperto in un editor di testo con funzioni di ricerca (preferibilmente un computer).

3.4 Difficoltà del gioco

Poiché il gioco implementerà meccaniche di Gamification, parte del gioco sarà costituito da un avanzamento nei livelli, in maniera analoga ad una saga. Di conseguenza sarà prevista sia la possibilità di eseguire una partita estemporanea, sia di eseguire partite multiple con difficoltà crescente. Per raggiungere questo obiettivo si andrà ad agire su più parametri, in parte descritti nel paragrafo 3.2.

- Numero di caselle orizzontali e verticali abbinate ad un crescente numero di definizioni nello schema. Questa meccanica si basa sul

presupposto che, a parità di difficoltà delle definizioni, uno schema di dimensioni maggiori sarà più difficile da risolvere, o perlomeno richiederà più tempo, di uno di dimensioni minori.

- Percentuale di facilitazione: nei livelli iniziali si può prevedere un'elevata quantità di caselle svelate in partenza, rendendo gli schemi più semplici da risolvere. Questa potrà essere ridotta, fino ad eliminarla completamente, procedendo con i livelli successivi, aumentando così la difficoltà.
- Complessità delle definizioni: memorizzando le definizioni in un archivio (come descritto nel paragrafo 3.3) è possibile raggrupparle per complessità. In questo modo al momento della creazione di uno schema sarà possibile scegliere di generarlo utilizzando definizioni più o meno facili. È ipotizzabile la creazione di più livelli di difficoltà variando la percentuale di definizioni “facili” e “difficili” all'interno degli schemi. A parità di dimensioni dello schema, uno schema contenente un'elevata percentuale di definizioni che richiedono calcoli complessi sarà più difficile da risolvere rispetto ad uno che contiene molte definizioni semplici.

La presenza di molti parametri rende possibile la creazione di molti livelli di difficoltà diversi, a seconda delle combinazioni dei parametri elencati. Sarà possibile creare una serie di “preset” (ad esempio un livello di difficoltà facile, uno medio e uno difficile), o lasciare la massima libertà nella generazione dei cruciverba.

3.5 Interfaccia grafica e casi d’uso

Sono state realizzate due interfacce grafiche per l’applicazione: la prima ha scopo dimostrativo, e consente di illustrare e presentare le funzionalità dell’applicazione; la seconda rappresenta la schermata di gioco visualizzata dall’utente, e contiene meno dati. Nella prima i parametri di generazione dei cruciverba sono impostati tramite interfaccia grafica; nella seconda sono nascosti, in quanto si prevede che le partite saranno lanciate automaticamente dalla macro-applicazione, selezionando i corretti parametri di dimensione dello schema, difficoltà e percentuale di facilitazione, in maniera trasparente all’utente. In questo modo sarà possibile organizzare il gioco in maniera dinamica: l’utente potrà partecipare a partite singole con parametri

personalizzati, o potrà affrontarne una serie, generate con difficoltà crescente da parte del sistema, come indicato anche nel paragrafo 3.4. Per la generazione di queste immagini è stato utilizzato lo stesso set di definizioni specificato nel paragrafo 3.2.2, con uno schema di dimensione fissata ad 8x8.

3.5.1 Interfaccia grafica 1: dimostrazione

In questo esempio verrà generato un nuovo cruciverba, e sarà visualizzato sia lo schema in bianco da compilare, sia lo schema risolto. Saranno inoltre stampate tutte le definizioni orizzontali e verticali. Qualora non fosse possibile inserire tutte le definizioni all'interno dello schema, verranno visualizzate anche queste in un elenco a parte. Saranno inoltre visualizzati a video i controlli per l'inserimento delle definizione e per la regolazione dei parametri di generazione del cruciverba: dimensione della griglia e del font, dimensione orizzontale, dimensione verticale, e percentuale di aiuto. Una volta impostati tutti i parametri del cruciverba, sarà sufficiente cliccare il pulsante "Genera cruciverba" per avviare l'algoritmo. La figura di riferimento è la 3.12. Durante l'esecuzione verrà mostrata un'immagine animata di caricamento.

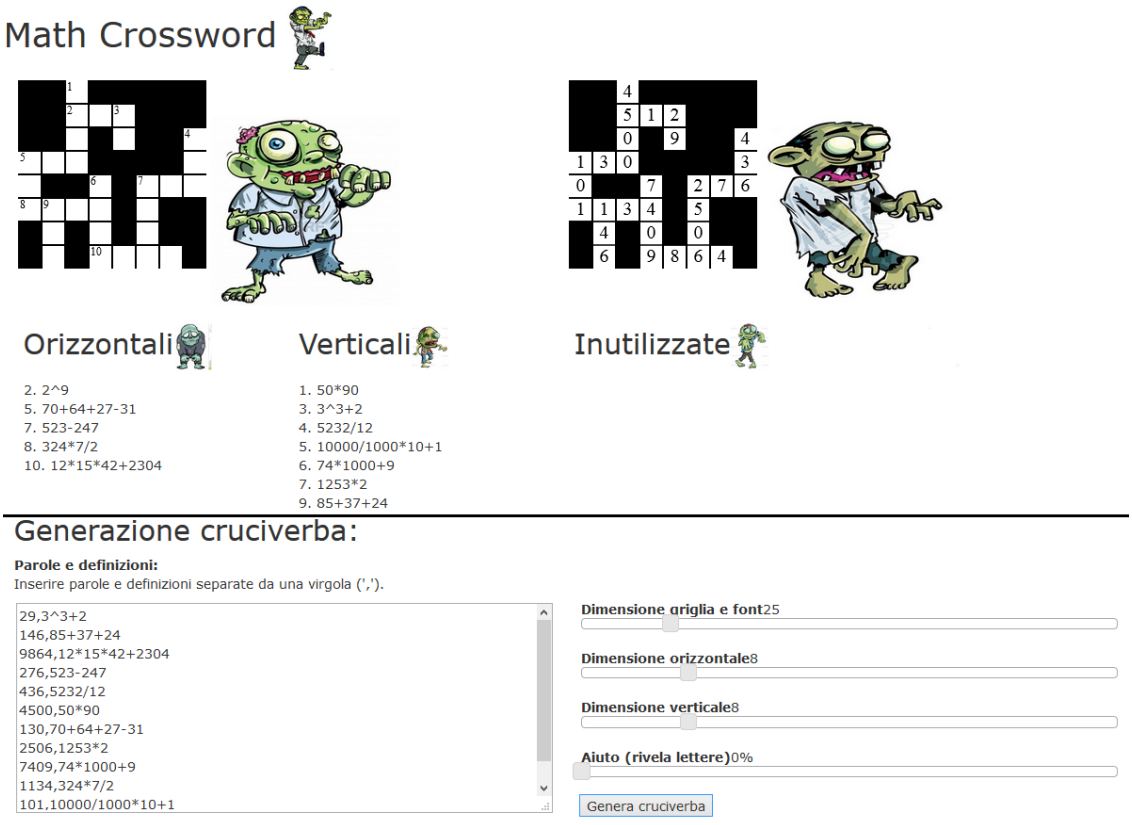

Figura 3.12 – Interfaccia di generazione

Notare come in questo primo esempio sia stata introdotta una prima tematizzazione in stile zombie. Come precedentemente accennato, saranno disponibili progressivamente altri temi, in modo di adattarsi alle preferenze degli utenti e agli specifici ambiti di utilizzo. Inoltre, come ipotizzato nel capitolo 2, si potrebbe rendere disponibili alcuni temi grafici con il conseguimento di determinati traguardi, quali ad esempio aver risolto un cruciverba particolarmente complesso, oppure averne risolti correttamente un numero elevato.

A causa dell'elevato contenuto informativo all'interno di questa visualizzazione, se ne raccomanda l'utilizzo da dispositivi che dispongano di uno schermo sufficientemente grande, in quanto in caso contrario sarebbe necessario scorrere continuamente da un lato all'altro dello schermo, oppure di impostare un livello di ingrandimento tale da rendere difficilmente leggibile il contenuto.

Sebbene questa interfaccia grafica sia stata creata a scopo dimostrativo, potrebbe essere anche utilizzata da un utente che vuole generare i propri cruciverba personalizzati, ad esempio per riutilizzarli anche al di fuori dell'applicazione. È questo il caso di un docente che vuole proporre ai suoi studenti uno schema contenente definizioni ben precise, in relazione agli ultimi argomenti trattati a lezione.

3.5.2 Interfaccia grafica 2: vista utente

Dopo aver presentato le principali funzionalità dell'applicazione con il primo caso d'uso, verrà ora mostrata l'interfaccia grafica che sarà utilizzata dall'utente durante il normale utilizzo dell'applicazione. Il cruciverba utilizzato è lo stesso del caso d'uso precedente. La figura di riferimento è la 3.13.

In questo caso su schermo vengono mostrate molte meno informazioni rispetto al primo caso d'uso. Questa interfaccia, infatti, è stata progettata per meglio adattarsi a dispositivi di varia natura. Essendo progettata in verticale, è meglio adattabile agli smartphone, che non a caso sono i dispositivi con schermo più piccolo tra quelli a cui l'applicazione è indirizzata. Su smartphone con schermo sufficientemente grande, e relativamente a cruciverba di dimensioni non eccessive, è possibile visualizzare interamente e contemporaneamente sia lo schema che le definizioni, in modo tale da avere sempre tutti i dati in vista, migliorando l'esperienza d'uso. Per questo motivo si raccomanda di evitare la generazione di schemi troppo grandi per il dispositivo target di volta in volta.

Ad ogni modo, l'interfaccia è utilizzabile anche da computer senza comportare significative alterazioni nella visualizzazione: la dimensione verticale di uno schermo da computer è di norma sufficiente a contenere tutto lo schema e le definizioni associate. In alternativa, potrebbe essere sensata una visualizzazione alternativa, nella quale le definizioni non sono mostrate sotto allo schema, ma accanto, per meglio sfruttare gli schermi con rapporto 16:9.

Per quanto riguarda i tablet, l'applicazione può essere eseguita sia in modalità *portrait*, sia in modalità *landscape*. Principalmente si suggerisce la prima modalità, in quanto le dimensioni verticali meglio si abbinano a quel tipo di visualizzazione. Qualora si implementasse anche un'interfaccia grafica *landscape*, in maniera simile a quella alternativa proposta per computer, l'utilizzo di una modalità piuttosto che dell'altra sarebbe a discrezione dell'utente.

In questa modalità l'utente inserisce le risposte ad ogni definizione al di sotto della definizione stessa. In un'ipotetica modalità *landscape*, l'area di testo potrebbe trovarsi a fianco della definizione, riducendo lo spazio occupato verticalmente a favore della dimensione orizzontale.

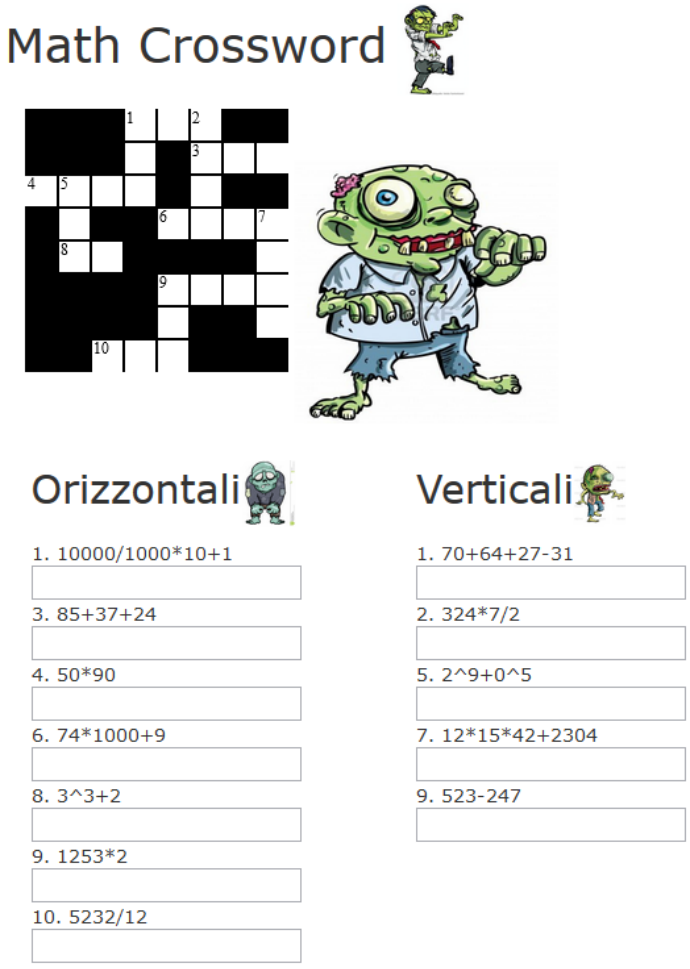

Figura 3.13 – Interfaccia utente

Conclusioni

Nell’ambito di questo progetto è stato realizzato un gioco sotto forma di WebApp, ovvero un cruciverba numerico. In questa tesi sono stati analizzati gli utilizzi dell’elemento gioco in ambito didattico e le possibilità di impiego delle nuove tecnologie mobili, quali smartphone e tablet, come piattaforme di gioco. Sono inoltre state analizzate le tecnologie e le piattaforme di sviluppo sui vari sistemi operativi e le applicazioni multipiattaforma passando dalle WebApp alle applicazioni ibride. Per ognuna di queste piattaforme sono stati analizzati aspetti positivi e negativi, e la decisione finale sul tipo di applicazione da implementare è stata presa alla luce di ogni aspetto, come dettagliatamente descritto nel capitolo 2.

Nel corso della realizzazione del progetto, ci si è resi conto di come alcune delle specifiche inizialmente previste non fossero direttamente realizzabili, se non a fronte di alcune modifiche. In particolare, non si pensava che la componente prestazionale dell’applicazione risultasse critica, come invece si è rivelata. Per questo motivo è stato necessario ripensare l’algoritmo di generazione dei cruciverba e le modalità di gestione delle definizioni degli schemi. Infatti, dal momento che parte delle specifiche richiede che le elaborazioni siano svolte interamente dai client, si ha come conseguenza diretta la necessità di far fronte a dispositivi caratterizzati da una capacità elaborativa limitata: l’applicazione, infatti, deve essere eseguibile in maniera corretta sia su dispositivi performanti, come i computer, sia su dispositivi a prestazioni ridotte, come gli smartphone. In alternativa, sarebbe necessario spostare l’onere computazionale su un server remoto, che potrebbe essere dimensionato secondo le necessità, comportando però una maggiore dipendenza dalla disponibilità di una connessione ad Internet da parte dei client. Poiché nel caso dei dispositivi degli utenti non è pensabile che questi

siano acquistati con potenza sempre maggiore in tempi rapidi, l’applicazione dovrà essere pensata in modo tale da essere funzionante con prestazioni accettabili anche sui dispositivi meno performanti, che rappresentano il caso d’uso pessimo.

Per questo motivo, si è dovuti passare da un algoritmo auto-iterante, potenzialmente un numero illimitato di volte, ad un algoritmo contenente criteri euristici, che non necessariamente porta alla generazione di un cruciverba perfetto, ovvero contenente tutte le definizioni richieste e posizionate nella maniera più compatta possibile, ma che comunque sia qualitativamente buono e non sensibilmente peggiore rispetto alla soluzione ottima.

Per fare fronte a questa situazione si è scelto di limitare il numero di esecuzioni dell’algoritmo, non in base ad un numero preimpostato, ma al tempo massimo di esecuzione, che è stato fissato empiricamente a cinque secondi, al termine del quale verrà scelta la migliore soluzione trovata. Al fine di generare schemi della migliore qualità possibile, sono inoltre state indentificate delle specifiche consigliate che regolano il rapporto tra il numero di definizioni all’interno di uno schema e la dimensione del cruciverba stesso.

Sono state necessarie anche ottimizzazioni per quanto riguarda il formato di memorizzazione delle definizioni, dal momento che archivi di dimensioni eccessivamente elevate risultano difficilmente gestibili da dispositivi a basse prestazioni. Si era inizialmente ipotizzato di generare le definizioni maniera dinamica, e non memorizzandole all’interno di un database. Tuttavia, questa scelta implica un ulteriore aumento del carico di lavoro, e come soluzione è stata scartata in favore della gestione tramite archivio già contenuto all’interno dell’applicazione.

Nell’ambito di questa tesi si è approfondito l’utilizzo delle tecnologie web, con particolare riferimento al linguaggio Javascript. Infatti, in una WebApp l’utilizzo di Javascript è decisamente più marcato rispetto, ad esempio, ad un sito web standard. La causa di ciò è data dal fatto che in una WebApp la logica applicativa è realizzata tramite Javascript, mentre in un sito web più semplice l’utilizzo di Javascript è limitato e fornisce funzioni accessorie.

Tuttavia, sebbene le potenzialità di Javascript si siano espanse nel tempo, resta un linguaggio meno potente rispetto a molti altri linguaggi di programmazione tradizionali, quali ad esempio Java o C#, che sono eseguiti direttamente sul sistema

piuttosto che all'interno di una sandbox quale è un browser. La realizzazione di applicazioni ibride attraverso l'utilizzo di framework quali Apache Cordova consente di ridurre il gap funzionale, ma non modifica Javascript stesso. La minore espressività di Javascript rende più complesso un utilizzo rigoroso dei pattern object-oriented, causando una maggiore difficoltà nel riuso del codice e nell'incapsulamento. Di conseguenza, anche il debug del codice scritto risulterà meno immediato.

Ad ogni modo le WebApp e le applicazioni ibride rappresentano un trend in crescita, proprio grazie all'estrema portabilità, irraggiungibile con i linguaggi di programmazione tradizionali. Infatti, realizzare un singolo progetto come applicazione nativa richiede una versione differente per ogni piattaforma, comportando un aumento dei costi e delle risorse necessarie allo sviluppo, a meno di non scegliere a priori un insieme ridotto di dispositivi target e sviluppare esclusivamente per quelli, eliminando le altre piattaforme, e di conseguenza perdendo un numero considerevole di potenziali utenti.

Gli algoritmi e le scelte implementative effettuate nella realizzazione del progetto consentono di estendere l'utilizzo dell'applicazione anche al di fuori dell'ambito di partenza, ovvero lo studio della matematica per i ragazzi delle scuole medie: è possibile allargare il pubblico anche ai bambini delle scuole elementari o ai ragazzi delle superiori, proponendo cruciverba di difficoltà adeguata alla loro preparazione. Inoltre, essendo l'algoritmo di generazione dei cruciverba non legato al caso specifico del cruciverba numerico, è possibile riutilizzare il gioco anche per altri contesti differenti da questo progetto di tesi.

Bibliografia

[LAF14] “L’importanza del Calcolo e della Geometria nella Scuola Primaria” http://www.ilsussidiario.net/News/emmeciquadro/Emmeciquadro-n-54/2014/10/1/SCIENZ-SCUOLA-L-importanza-del-Calcolo-e-della-Geometria-nella-Scuola-Primaria/526449/

[VAL15] “Perché la matematica scolastica è noiosa e così poco interessante?” – Mario Valle. http://mariovalle.name/matematica/problemi.html

[GEN05] “La Matematica e i giovani: un rapporto conflittuale superabile? Resoconto di una esperienza.” – Giuseppe Gentile http://ww2.unime.it/mathesis/La%20Matematica%20e%20i%20giovani.pdf

[PAN01] “Uso e ‘abuso’ della rete” - Ilbo http://www.unipd.it/ilbo/content/uso-e-abuso-della-rete

[VIO11] “Cosa è e cosa non è Gamification” – Gameifications.com http://www.gameifications.com/gamification/cosa-e-e-cosa-non-e-gamification/

[DET11] Deterding, Sebastian, et al. "From game design elements to gamefulness: defining gamification." Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments. ACM, 2011.

[FAT12] “Quando il gioco «fa» salute” – Marketingsociale.net http://www.marketingsociale.net/download/Quando_il_gioco_fa_salute.pdf

[ALI10] “Gamification e obiettivi principali” - Gamification.it http://www.gamification.it/gamification/gamification-e-obiettivi-principali/

[ALT10] “Meccaniche e dinamiche della gamification” - Gamification.it http://www.gamification.it/gamification/meccaniche-e-dinamiche-della-gamification/

[VIL11] “Guida alla gamification” http://www.slideshare.net/akiraa/lezione-di-gamification

[STE10] “New Report Says Online Shoppers Can be Motivated by Gaming” – Auctionbytes.com http://www.auctionbytes.com/cab/cab/abn/y10/m11/i04/s04

[COR10] "The 'Gamification' Of Education" - Forbes.com
http://www.forbes.com/2010/10/28/education-internet-scratch-technology-gamification.html

[HUL10] "Gamification: Turning Work Into Play" – Hplusmagazine.com
http://hplusmagazine.com/2010/03/25/gamification-turning-work-play/

[SLB11] "Unire la Gamification al Social Learning" - Social Learning Blog
http://www.sociallearning.it/unire-la-gamification-al-social-learning

[ADA12] Adams E., Dormans, J.(2012), Game Mechanics: Advances Game Design, Berkley: New Riders Games, 2012, pp. 272-274.

[ARM08] "History of Military gaming" - The United States Army
http://www.army.mil/article/11936/History_of_Military_gaming/

[TIG14] "The International Journal on Serious Games, a scientific Open Access Journal", first issue January 2014. – Seriousgamessociety.org
http://journal.seriousgamessociety.org/

[ABT70] Abt, C. C. (1970), Serious Games, Viking Press: "Games may be played seriously or casually. We are concerned with serious games in the sense that these games have an explicit and carefully thought-out educational purpose and are not intended to be played primarily for amusement. This does not mean that serious games are not, or should not be, entertaining."

[SAW07] Sawyer, B. (2007). The "Serious Games" Landscape. Presented at the Instructional & Research Technology Symposium for Arts, Humanities and Social Sciences, Camden, USA.

[ARN12] Arnab S., Berta R., Earp J.,de Freitas S., Popescu M., Romero M., Stanescu I. and Usart M., "Framing the Adoption of Serious Games in Formal Education" , in: Electronic Journal of e-Learning, Volume 10, Issue 2, 2012, pp. 159-171.

[ORL07] Kyle Orland, Dave Thomas, Scott Steinberg, The videogame style guide and reference manual, Power Publishing, 2007, p. 26 ISBN 978-1-4303-1305-2

[MAB15] "Virtual currency: Sources and Sinks"
http://android-developers.blogspot.it/2015/10/virtual-currency-sources-and-sinks.html

[FAC15] "Facebook Developers" - Facebook for Developers
https://developers.facebook.com/

[TAG11] "Gamification & Editoria Online"
http://blog.tagliaerbe.com/2011/06/gamification-editoria.html

[DEL11] "Analfabeti digitali. L'Italia che ignora Internet" - LaRepubblica.
http://www.repubblica.it/cronaca/2011/11/03/news/analfabeti-24332485/

[ANV15] AndroidVulnerabilities.org
http://androidvulnerabilities.org/

[MIC14] "Il support per Windows XP è terminato" – Microsoft.com
http://windows.microsoft.com/it-it/windows/end-support-help

[ADD15] "Android Developer Dashboard"
https://developer.android.com/about/dashboards/index.html?utm_source=suzunone

[GAR15] "Gartner Says Tablet Sales Continue to Be Slow in 2015", 5 January 2015. Retrieved 6 March 2015. http://www.gartner.com/newsroom/id/2954317

[NMS15] “Market share statistics for Internet Technologies” - Netmarketshare https://www.netmarketshare.com/operating-system-market-share.aspx?qprid=10&qpcustomd=0

[AND15] Android Developer https://developer.android.com/index.html

[APP15] Apple Swift https://www.apple.com/it/swift/

[WIN15] “Introduzione alle Windows app” – Microsoft https://dev.windows.com/it-it/getstarted

[LAR15] “Microsoft Launches Its .NET Distribution For Linux And Mac” – Techcrunch.com http://techcrunch.com/2015/04/29/microsoft-launches-its-net-distribution-for-linux-and-mac/

[WEB15] “WebGL Wiki” https://www.khronos.org/webgl/wiki/Main_Page

[FER09] Jose Fermoso (April 5, 2009). "PhoneGap Seeks to Bridge the Gap Between Mobile App Platforms". GigaOM. Retrieved 2012-04-07. http://gigaom.com/2009/04/05/phonegap-seeks-to-bridge-the-gap-between-mobile-app-platforms/

[ADO11] "Adobe Announces Agreement to Acquire Nitobi, Creator of PhoneGap" - Adobe.com. 2011-10-03. Retrieved 2012-04-07. http://www.adobe.com/aboutadobe/pressroom/pressreleases/201110/AdobeAcquiresNitobi.html

[INT12] "The Development of Mobile Applications using HTML5 and PhoneGap on Intel Architecture-Based Platforms". 2012-06-22. Retrieved 2013-02-17. https://software.intel.com/en-us/articles/the-development-of-mobile-applications-using-html5-and-phonegap-on-intel-architecture-based

[TRI12] "PhoneGap advice on dealing with Apple application rejections" - Adobe Systems. 2012-10-29. Retrieved 2013-02-17. http://www.adobe.com/devnet/phonegap/articles/apple-application-rejections-and-phonegap-advice.html

[DIO15] DigitalOcean https://www.digitalocean.com/

[AMA15] Amazon Web Services (AWS) https://aws.amazon.com/it/

[JQU15] JQuery https://jquery.com/

[WAY14] Progetto js-xwords – Github https://github.com/waystoskinacat/js-xwords