5. GCompris

5.1 Una suite per piccoli utenti

"GCompris is a full featured educational application that propose different activities to children from 2 to 10 years old. In french GCompris is pronounced "j'ai compris" and means I Have Understood. This software is released under the GPL license and is an official GNU Package. GCompris runs on GNU/Linux and various flavors of Unix."[1]

Così si apre la descrizione del programma GCompris. Dopo un primo utilizzo ci si rende subito conto delle potenzialità di questa suite, davvero molto vaste in questa versione. Uno dei punti di forza è che le varie attività vengono gestite con un sistema completo e modulare: chiunque può scrivere la sua 'board' – così vengono chiamati i moduli di GCompris – e inserirla all'interno del programma. Molti sviluppatori di tutte le nazionalità hanno contribuito ad estendere le funzionalità del programma con giochi ed attività sempre nuovi.

Dal punto di vista didattico troviamo molte esperienze

1Da http://www.ofset.org/GCompris/about.html "GCompris è una applicazione completa che propone differenti attività ai bambini dai 2 ai 10 anni. In francese GCompris si pronuncia "j'ai compris" e significa 'ho compreso'. Questo software è rilasciato sotto licenza GPL ed è un pacchetto ufficiale GNU. GCompris gira su sistemi GNU/Linux e vari tipi di Unix." (trad. mia)

multimediali, tutte con vari livelli di difficoltà sequenziali – una volta finito il primo si passa automaticamente al secondo e così via – ed anche dei giochi più semplici adatti a pause di divertimento.

Ad esempio, per i bambini fino ai quattro anni troviamo una serie di attività per scoprire l'utilizzo del computer: muovere il mouse, scoprire l'utilizzo delle lettere e dei numeri sulla tastiera, ricostruire puzzle di quadri famosi o semplice grafica vettoriale; per i bambini più grandi (dai 7 anni in su) possiamo invece trovare delle applicazioni complesse per imparare l'algebra (a più cifre) oppure relative al sistema monetario europeo con il quale il bambino familiarizza con l'euro in tutti i suoi tagli, acquistando virtualmente alcuni oggetti.

Sono disponibili molte altre attività, per tutti i tipi di età e la mailing list degli sviluppatori e utilizzatori di GCompris è molto attiva, in un continuo scambio di idee e di impressioni che portano a rapidi miglioramenti nel corso del tempo. Paradossalmente questo fiorire di applicazioni didattiche all'interno della suite è stato un ostacolo per la ricerca di possibili boards ancora da sviluppare.

5.2 Requisiti tecnici per l'utilizzazione e lo sviluppo

Le applicazioni per i sistemi GNU/Linux ci hanno abituato a requisiti minimi per quanto riguarda un utilizzo medio del computer ben lontani da quelli richiesti dalle più blasonate applicazioni commerciali. GCompris non fa eccezione, tanto che per utilizzarlo basta un Pentium 150 Mhz o equivalente, 32 MB di memoria e 30 MB di spazio sul disco fisso. Per quanto riguarda il software al momento GCompris gira su sistemi GNU/Linux, MacOS X, FreeBSD e Unix[2] – vi è anche in corso un porting per

2Tutti i nomi di aziende, marche e prodotti sono marchi registrati o marchi delle rispettive società.

Microsoft Windows[3] – e data la sua intrinseca multimedialità necessita anche del sistema grafico X-Window. Al momento GCompris è pacchettizzato per tutte le distribuzioni più diffuse, e comunque sono disponibili i sorgenti per costruire il programma binario – tutto il pacchetto di GCompris è open source, rilasciato sotto licenza GPL – quindi virtualmente è possibile effettuare il porting su qualsiasi tipo di architettura.

Per quanto riguarda lo sviluppo di nuovi giochi, e quindi anche la compilazione dei sorgenti per poi ottenere il pacchetto binario compilato, le richieste a livello software sono naturalmente maggiori.

Come si legge nel file INSTALL contenuto all'interno della directory principale dei sorgenti di GCompris:

```
GCompris requires gnome-libs, xmlib2, gdk-pixbuf-gnomecanvas and
gdk-pixbuf.
GCompris also requires the gnuchess package for the chess
activity to run.
GCompris includes several activity written in python. To enable
them, you need to have python installed on your system with the
following packages:
libpython
python
gnome-python-canvas
libpython2.2-devel
gnome-python
python-base
Now that the sound is managed internally, you also need:
libvorbis libao libogg libao-devel libvorbis-devel libogg-devel
```

3 Idem c.s.

Si tratta dei requisiti standard per lo sviluppo di una applicazione multimediale che si appoggia sia sul linguaggio Python sia sul linguaggio C.

5.3 Localizzazione in lingua italiana delle voci

Essendo il progetto GCompris basato sul lavoro di volontari che dunque non sempre possono dedicarsi a tempo pieno al suo sviluppo, le voci che guidano il bambino durante i giochi di lettura o di geografia non erano ancora disponibili nella traduzione italiana.

Alcuni dei traduttori italiani avrebbero preferito tradurre le voci con una voce di bambino e non di adulto per far sentire più a loro agio i piccoli giocatori. Questo però non è sempre conveniente in termini di "pulizia" della dizione delle voci: in molti giochi di lettura il riconoscimento del suono è cruciale e non è quindi pensabile che un bambino riesca ad articolare il suono a livello tale da poter far distinguere le differenze sonore per esempio tra una 'd' e una 'b'.

Per questo ho scelto di registrare le voci con una voce di una doppiatrice professionista che potesse ben articolare e scandire i suoni, in modo tale da mettere in grado il bambino di rispondere correttamente alla domanda.

Dato che lo sviluppo è comunque al vaglio continuo della comunità non si è ancora deciso se questa traduzione rimarrà quella ufficiale oppure se verrà rimpiazzata con quella di un bambino.

Sicuramente ora i bambini della materna possono giocare con molte delle board di livello più semplice (come per esempio il riconoscimento dei colori di base).

Dal punto di vista tecnico ho preferito usare il programma 'rec' da linea di comando per poter più agevolmente gestire la registrazione sequenziale dei suoni:

```
rec -c 1 -r 44000 -s w nomedelfile.wav
```

Una volta registrati i suoni li ho compressi tramite il programma 'oggenc', seguendo le direttive del file HOWTO_ENCODE presente nella directory gcompris/boards/sounds :

```
- In the directory where WAV files are run
 oggenc -c "Copyright 2002 (name of author). This file is distributed
under the terms of the GNU General Public License, either version 2 or (
at your option) any later version. See the file COPYING for details."
*.wav
```

Per rendere attive queste voci all'interno di GCompris, è stato necessario modificare i vari file gcompris_xxxx_yy.assetml.in contenuti in ogni sottodirectory di gcompris/boards/sounds/it – uno per sottodirectory – quali: alphabet, colors, geography e misc. La parte yy del nome del file identifica la lingua dei suoni – nel nostro caso italiano, quindi it – mentre invece la parte xxxx prende il nome della directory alla quale il file .in fa riferimento. Naturalmente ho modificato anche i vari file Makefile.am all'interno di queste sottodirectory per indirizzare il processamento ai nuovi file creati: è bastato inserire nella voce assetml_in_files= i nomi dei file creati (ad esempio gcompris_alphabet_it.assetml.in).

5.4 La creazione di un nuovo modulo per GCompris

Per inserirmi in modo funzionale nella suite GCompris, ho pensato di restare in ambito europeo e aggiungere un modulo sul riconoscimento

dei segnali stradali più comuni. Lavorando con le insegnanti della scuola G. Marcati, abbiamo effettuato una selezione dei segnali più adatti allo scopo. Anche se fin dalla scuola materna i bambini sono stimolati al riconoscimento di segnali generici grazie alla loro forma o colore, questo modulo è sicuramente più adatto ad alunni della scuola elementare.

Dal punto di vista implementativo si è scelto di partire dal codice sorgente direttamente scaricato dal CVS[4] per lavorare sulla versione più aggiornata possibile dello stesso. La versione online del codice è soggetta a cambiamenti molto di frequente – apportati da tutti gli sviluppatori associati al progetto; tutte queste patch vengono vagliate da Bruno Coudin, il fondatore di GCompris, che ha adottato il sistema per gestire il codice utilizzato anche dal progetto GNOME (fisicamente il codice di GCompris risiede sul server CVS di questo). Il linguaggio utilizzato è il C o il Python, a seconda della board. Prima GCompris faceva uso anche dei widget Canvas di GNOME ma nella versione 5.0 il fondatore del progetto ha preferito abbandonarli in favore dell'utilizzo esclusivo delle librerie GTK. Per un programma educativo il codice sorgente è molto vasto, basti pensare che occupa ben 90 MB di spazio sul disco fisso, una volta compilato questo spazio scende sui 28 MB, che restano comunque molti per un programma didattico. Questo è dovuto in gran parte all'intensivo utilizzo di grafica e di suoni, per rendere tutta la suite più accattivante e coinvolgente per il bambino. Inoltre non bisogna dimenticare che all'interno di GCompris sono tante e tali le applicazioni presenti che non è possibile paragonarlo ad un semplice programma

4CVS acronimo per Concurrent Versions System: da http://www.openlabs.it/regcorsi/schedule.php?cid=1 "il principale sistema di controllo di versione usato per gestire lo sviluppo di prodotti creati dal singolo sviluppatore come per progetti seguiti da più team distribuiti su tutto il pianeta. E' un semplice e potentissimo strumento per risolvere la gran parte delle problematiche relative alla produzione, evoluzione e manutenzione di progetti scritti a più mani che permette di storicizzare la crescita di ogni componente di sistema (un programma, della documentazione, progetti, ...)." Cfr. http://www.cvshome.org/ .

didattico con uno scopo unico.

Grazie alla modularità estrema della suite, costruire una nuova tavola per il riconoscimento dei segnali è stato relativamente facile a livello di mole del linguaggio di programmazione. La board costruita fa parte di una tipologia di gioco chiamata da Bruno Coudin 'shapegame' – gioco delle forme appunto – e varie attività si basano su di essa (babyshapes, imagename ecc). Tutta la suite oltre ai veri e propri linguaggi di programmazione fa molto uso di XML[5] per descrivere le varie attività. L'XML è molto usato anche per completare la parte implementativa di alcune board dato che molte di queste all'interno della sezione lettura usano le indicazioni testuali oltre che le immagini e i suoni per guidare il bambino verso il coretto svolgimento dell'esercizio.

La suite di GCompris è utilizzata in tutto il mondo grazie a una fitta schiera di traduttori che si avvicendano a mano a mano che vengono rilasciati nuovi giochi per fornire le traduzioni nella loro lingua di appartenenza. Di base le varie board sono scritte in lingua inglese, e anche per i nomi dei file si cerca di attenersi a questa regola utilizzando sempre nomi significativi ma in inglese.

Durante la creazione del mio modulo ho preferito sviluppare di pari passo la creazione con la traduzione, poiché essendo il testo parte integrante dell'esercizio era necessario controllare livello per livello che tutte le indicazioni testuali fossero perfettamente leggibili. Purtroppo questo controllo è stato possibile solo per la lingua inglese e quella italiana. I traduttori per le altre lingue potranno comunque inserire il codice '/n' per mandare il testo a capo e spezzarlo su due righe.

5XML: Extensible Markup Language. “È un metalinguaggio che permette di creare dei linguaggi personalizzati di markup. [...] Si caratterizza per la semplicità con cui è possibile scrivere documenti, condividerli e trasmetterli nel Web.”Cfr. http://www.html.it/xml/guida/origine.htm

Per la traduzione delle indicazioni testuali ho seguito le linee guida dell' IT-GNU-Translator-HOWTO[6], riferendomi specificatamente al punto quattro, quello che tratta i messaggi del programma. L'implementazione avviene tramite i file con estensione `.po` - le due lettere che formano il nome del file identificano la lingua in cui si è effettuata la traduzione – nel caso della lingua italiana sarà quindi `it.po`. È necessario solamente un file per ogni traduzione, che contenga tutti i messaggi del programma. La sintassi utilizzata è relativamente semplice: con la parola chiave `msgid` si identifica la parte del testo da tradurre racchiusa tra virgolette doppie – deve essere identica a quella contenuta nel file originale – mentre con `msgstr` la traduzione, contenuta anch'essa in una coppia di virgolette doppie.

5.5 Il codice sorgente e le modifiche ai Makefile preesistenti

Per costruire il modulo ho dovuto tenere conto della struttura e dei tool usati per mantenere il codice CVS. Prima di poter utilizzare il codice sorgente è necessario – oltre all'avere il sistema CVS installato sulla macchina – configurare delle variabili d'ambiente:

```
export CVSROOT=':pserver:anonymous@anoncvs.gnome.org:/cvs/gnome'
```

quindi bisogna registrarsi al servizio con:

```
cvs login
```

6Cfr: http://www.linux.it/~md/IT-NLS-HOWTO. Le linee guida per i traduttori italiani scritto da Marco d'Itri <md@linux.it>, il coordinatore del team italiano dei traduttori GNU.

non vi è nessuna password per scaricare il codice, quindi alla richiesta che verrà fatta dal server basterà premere invio per mandare una password vuota. Infine è possibile ottenere il codice di GCompris vero e proprio con il comando:

```
cvs -z3 checkout gcompris
```

L'opzione -z[n] specifica il livello di compressione dei dati. I valori consentiti vanno da 1 (alta velocità, bassa compressione) a 9 (compressione disabilitata) – di solito viene usato il valore 3 per la maggior parte dei progetti. Come impostazione di default tutto il materiale verrà scaricato all'interno della directory corrente, il manuale consiglia quindi di crearne una apposita dove depositare tutto il codice sorgente. Il modulo GCompris di per sé creerà una directory `gcompris` nella quale vi saranno tutti i file del progetto.

All'interno di questa cartella vi sono delle parti comuni a quasi la totalità dei progetti:

- in ogni directory vi è la sottodirectory CVS che contiene tutte le informazioni riguardo i numeri di versioni, da dove è stato scaricato il codice sorgente ecc (questa cartella può essere ignorata ai fini dello sviluppo vero e proprio, ma non va assolutamente cancellata pena il fallimento delle varie operazioni di sincronizzazione);
- il file `ChangeLog` dove vengono registrati tutti i cambiamenti effettuati al codice sorgente da parte dei vari sviluppatori; il file `README` dove si trovano delle indicazioni generali riguardo il modulo: un indirizzo email al quale riportare eventuali errori di compilazione, i programmi richiesti, i termini e le licenze di utilizzo ecc;
- il file `MANTEINERS` che contiene i nomi e gli indirizzi dei responsabili della manutenzione giorno per giorno del modulo;

? infine nella directory principale vi è lo script di collegamento `autogen.sh` che gestisce tutte le funzionalità di altri sotto programmi quali gettextize, intltoolize, libtoolize, aclocal, autoheader, automake & autoconf. Tramite questo script è possibile iniziare la compilazione del modulo vera e propria con il comando:

```
./autogen.sh –prefix=/usr/local
```

dove con `/usr/local` si indica il percorso nel sistema locale dove andranno installati una volta compilati i file binari.

Nello specifico, la struttura di Gcompris è costituita da una serie di `Makefile` che vengono generati con lo script suddetto a partire dai `Makefile.in` contenuti in ogni directory tramite i quali si istruisce il compilatore riguardo tutte le direttive per ottenere i file finali delle varie board e di tutte le parti generiche del programma.

Una volta lanciato lo script – a patto che non si siano ottenuti errori di compilazione o di dipendenze di librerie non soddisfatte – basta lanciare il comando:

```
make && make install
```

per ottenere i binari e quindi l'installazione di questi nella directory specificata con l'opzione `–prefix=/usr/local` (in questo esempio abbiamo installato il modulo in `/usr/local`).

Per costruire la nuova board vera e propria sono partito da una preesistente, come consigliato nella guida per gli sviluppatori di GCompris. Dato che era mia intenzione costruire una tavola per il riconoscimento dei segnali stradali, il modulo più adatto era sicuramente `imagename` ovvero il gioco di 'immagini e nomi' che si propone un

obiettivo concettualmente diverso ma tecnicamente simile: saper associare il nome di alcuni oggetti con la oro rappresentazione grafica.

Partendo dal codice sorgente all'interno della sottodirectory boards/imagename ho costruito un'altra sottodirectory all'interno della stessa boards chiamata roadsigns nella quale ho messo i file relativi alle varie tavole – una per ogni livello – contenenti le descrizioni e le funzionalità della board. Per i file ho usato l'XML, seguendo lo stile e le parole chiave utilizzate dall'autore di imagename (lo stesso Bruno Coudin, fondatore del progetto GCompris stesso). I nomi dei file scelti seguono una semplice numerazione sequenziale: boardX_Y.xml.in dove X è il numero del livello (da 1 a 6) e Y è la categoria di appartenenza della tavola (0 per tutte in questo caso, date che appartengono allo stessa tipologia di gioco).

Il contenuto di questi file è molto simile ne riporto qui uno solo, è possibile comunque trovare gli altri in appendice:

```
<?xml version="1.0" encoding="UTF-8"?>
<ShapeGame>
 <Shape name="1" pixmapfile="gcompris/gcompris-shapelabel.jpg"

     type="SHAPE_BACKGROUND"    x="405"    y="490"    zoomx="1"    zoomy="1"
position="0"/>
 <Title  x="394" y="490" justification="GTK_JUSTIFY_CENTER">
 <_name>Drag and Drop the items above their written name</_name>
 </Title>
 <Shape name="E" pixmapfile="gcompris/misc/1_no_entry.png"
        x="194" y="90" zoomx="1" zoomy="1" position="0"/>
 <Title x="194" y="200" justification="GTK_JUSTIFY_CENTER">
  <_name>no entry for all vehicles</_name>
 </Title>
```

```
 <Shape name="F" pixmapfile="gcompris/misc/1_no_horns.png"
        x="194" y="340" zoomx="1" zoomy="1" position="0"/>
 <Title x="194" y="450" justification="GTK_JUSTIFY_CENTER">
  <_name>use of horns prohibited</_name>
 </Title>
 <Shape name="B" pixmapfile="gcompris/misc/1_parking.png"
        x="394" y="90" zoomx="1" zoomy="1" position="0"/>
 <Title x="394" y="200" justification="GTK_JUSTIFY_CENTER">
  <_name>parking</_name>
 </Title>
 <Shape name="D" pixmapfile="gcompris/misc/1_right.png"
        x="394" y="340" zoomx="1" zoomy="1" position="0"/>
 <Title x="394" y="450" justification="GTK_JUSTIFY_CENTER">
  <_name>direction to be followed</_name>
 </Title>
 <Shape name="M" pixmapfile="gcompris/misc/1_stop.png"
        x="594" y="90" zoomx="1" zoomy="1" position="0" />
 <Title x="594" y="200" justification="GTK_JUSTIFY_CENTER">
  <_name>halt sign</_name>
 </Title>
 <Shape name="A" pixmapfile="gcompris/misc/1_traffic_signal.png"
        x="594" y="340" zoomx="1" zoomy="1" position="0"/>
 <Title x="594" y="450" justification="GTK_JUSTIFY_CENTER">
  <_name>traffic signals ahead</_name>
 </Title>
</ShapeGame>
```

Come visto in precedenza per le indicazioni di testo all'interno dei file con estensione .in ho usato la lingua inglese – lo standard per GCompris – e poi ho modificato il file it.po aggiungendovi le relative

traduzioni in italiano.

Una volta costruito il modulo vero e proprio ho dovuto istruire il compilatore riguardo i nuovi file creati: è bastato modificare a catena tutti i Makefile all'interno delle varie directory. La struttura di GCompris è molto modulare, e bisogna porre attenzione in realtà a che il modulo con i suoi file grafici si trovino nel punto giusto di questa stessa struttura. Come già accennato la directory principale del modulo è gcompris/boards/roadsigns e contiene al suo interno tutti i file delle boards con il relativo Makefile.am:

```
xmldir = $(pkgdatadir)/@PACKAGE_DATA_DIR@/roadsigns

xml_in_files = \
    board1_0.xml.in \
    board2_0.xml.in \
    board3_0.xml.in \
    board4_0.xml.in \
    board5_0.xml.in \
    board6_0.xml.in

xml_DATA = $(xml_in_files:.xml.in=.xml)

@INTLTOOL_XML_RULE@
```

Nella sua struttura sono descritti i file da processare delle varie tavole, e la directory dove il compilatore le andrà a cercare.

All'esterno di gcompris/boards è stato necessario creare un altro file XML, roadsigns.xml.in che contenesse le specifiche del gioco (il nome, il tipo, la sezione, l'icona, la difficoltà, l'autore e la directory nel quale è contenuto) e altre notizie informative che verranno poi

visualizzate tramite i menù nel gioco stesso (la descrizione, il titolo, i prerequisiti, l'obiettivo finale e il modo di giocare):

```
<?xml version="1.0" encoding="UTF-8"?>
<GCompris>
 <Board
     name="roadsigns"
     type="shapegame"
     section="/reading/."
     icon="boardicons/roadsigns.png"
     difficulty="3"
     author="Francesco Agrusti (f.agrusti@email.it)"
     boarddir="roadsigns">
     <_description>Drag and Drop the road signs above their written
name</_description>
     <_title>Road Signs</_title>
    <_prerequisite>Reading and basic knowlegde about european road
signs</_prerequisite>
    <_goal>Road signs recognition, vocabulary and reading</_goal>
    <_manual>In the main board area, a set of red plots associated
with work are printed. In the vertical frame - at the left of the
main board area, a set of road signs are represented, each of these
road signs match with one short description in the main board area. The
right
association must be find. This is achieved by dragging the signals to the
right
red plot in the main area. There are six different levels of difficulty.
</_manual>
```

Come in precedenza è stato necessario segnalare la presenza

del nuovo modulo all'interno del `Makefile.am` proprio della directory `boards` aggiungendo nell'elenco della variabile d'ambiente `SUBDIRS` la directory `roadsigns` e poi nell'altro elenco, quello dei file XML con estensione `.in` – sempre contenuto nello stesso file – il file appena creato `roadsigns.xml.in`.

5.6 L'utilizzo del nuovo modulo

La nuova board ha come obiettivo il riconoscimento dei segnali stradali considerando come già acquisite le capacità di lettura e una minima conoscenza dei segnali stradali di base.

Figura 1 – *Presentazione della board*

Dalla schermata principale è necessario selezionare le attività di lettura e poi all'interno della tabella l'attività nominata 'segnali stradali' identificata da una icona del cartello stradale del divieto di sorpasso, come si può vedere nella figura 1.

Il bambino deve associare i segnali alla loro funzione in una tabella come quella mostrata nella figura 2. Naturalmente il giocatore deve avere già una certa familiarità con il mouse e più in generale con il sistema di trascinamento e rilascio delle icone, che poi è l'obiettivo di altri moduli della suite.

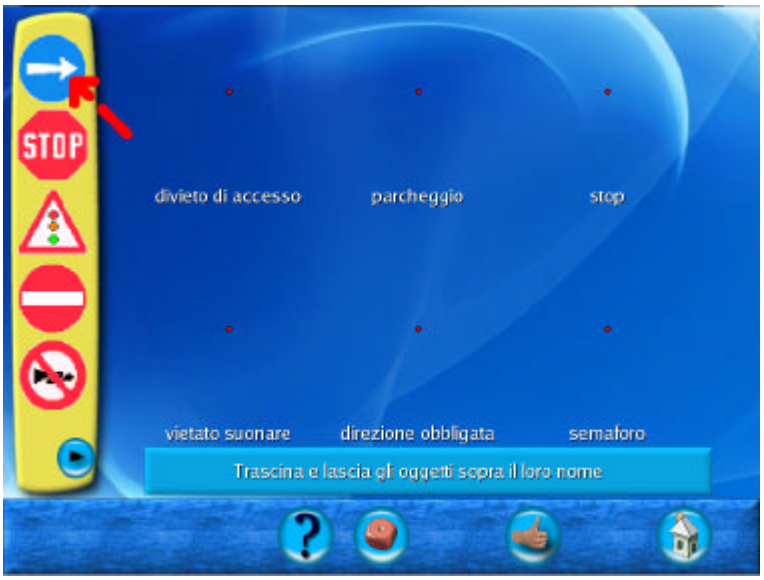

Figura 2 – *Primo livello della board*

Per lo stesso motivo non vi è alcun timer nel gioco, senza che quindi vi sia alcun premio per chi svolge il gioco in minor tempo. Come

anche non sono state prese in considerazione le mosse che vengono effettuate per raggiungere la soluzione – sia questa corretta o meno.

Una volta proposta una soluzione, l'alunno può confermare cliccando sul bottone con il disegno della mano. Nel caso la soluzione fosse errata il programma lo segnala con un suono e con la grafica visibile nella figura 3:

Figura 3 – *Primo livello della board errato*

Una volta modificata la soluzione – mettendo nella configurazione corretta le icone dei segnali – l'alunno la può sottomettere nuovamente. Come si vede in figura 4 il programma segnala la soluzione corretta e passa automaticamente al livello successivo (figura 4).

I segnali stradali sono stati scelti - tra la rosa di quelli ufficiali usati

all'interno della comunità europea - insieme al corpo docente della scuola elementare G. Marcati, con la consapevolezza che anche per i bambini delle classi più avanzate è necessario comunque l'intervento di una maestra che guidi l'alunno oltre che nella risoluzione corretta del gioco anche nell'apprendere quei segnali più ostici che, magari a tentativi, sono stati posizionati nel giusto posto all'interno della tavola. Questo modulo è stato pensato come completamento di alcune ore di educazione stradale, a modello di una verifica finale di un itinerario didattico appositamente programmato.

Figura 4 – *Primo livello della board corretto*