Numero 9 – Giugno 2006

Indice

Editoriale

di Roberto Grassi

Benvenuti al numero 9 di Terra d'IF.
Scrivo l'editoriale di questo numero durante un afoso mese di giugno, che registra un importante momento per le avventure italiane. In particolare, la 'rinascita' della One Room Game Competition con 9 giochi di cui due inglesi, lascia ben sperare per il futuro della scena nazionale. Sul fronte internazionale è certamente da segnalare la conclusione della Spring Thing[1] e l'annuncio della IFComp 2006[2].

Veniamo ora ai contenuti di questo numero, dedicato agli aspiranti autori di Interactive Fiction.

La sezione articoli ospita quattro contributi originali dedicati alla programmazione, narrazione e game design di un gioco di narrazione interattiva. Ospitiamo inoltre la traduzione di un interessante post di Michael Coyne sul design di Risorgimento Represso. Ci auguriamo che questi articoli coprano lo spettro della maggior parte delle informazioni richieste periodicamente dagli aspiranti autori italiani di narrativa interattiva.

Il numero si chiude con un'interessante intervista a Jason Devlin, il vincitore della scorsa IFComp, le recensioni dei giochi della ORGC e la consueta rubrica di programmazione in Inform, che prende in esame l'implementazione di un ascensore.

Alla prossima...

1 http://www.springthing.net/2006/index.htm
2 http://ifcomp.org/comp06/index.html

Strumenti per lo sviluppo

di Paolo Lucchesi

La scelta degli strumenti giusti è sempre un momento cruciale in ogni progetto; ferma restando la nostra perizia, gli strumenti adatti possono farci risparmiare fatica e possono garantirci un miglior risultato, mentre lavorare con strumenti sbagliati potrebbe anche renderci incapaci di portare a termine il nostro lavoro.

La realizzazione di un'avventura testuale non fa eccezione a questa regola. Per fortuna, grazie al lavoro di molti appassionati, abbiamo un gran numero di linguaggi di programmazione e applicativi creati appositamente per scrivere narrativa interattiva. In quest'articolo cercheremo di fare una panoramica concisa ma il più possibile completa su questi strumenti.

I "Big Three"

Con questo termine anglosassone si indicano Inform, TADS e Hugo, tre sistemi che vanno per la maggiore e che hanno tra loro diverse similitudini (almeno fino a qualche tempo fa). Anzitutto linguaggi dedicati alla realizzazione d'avventure testuali classiche (anche se la loro flessibilità permette anche altre scelte stilistiche); poi sono tutti e tre linguaggi di programmazione completi procedurali e ad oggetti (come, ad esempio, Java o il C++), e per questo hanno quindi bisogno di un certo tempo d'apprendimento, soprattutto per i non programmatori; inoltre tutti e tre generano non programmi eseguibili, ma un codice che poi deve essere caricato in un interprete (e che quindi è indipendente dal sistema operativo); infine ormai hanno tutti capacità multimediali.

Inform

di Graham Nelson,
La versione 6 di Inform è stata negli ultimi anni lo strumento più usato, grazie anche alla quantità di documentazione (manuali, articoli, esempi) e alle estensioni disponibili, ed è l'unico strumento tra i "Big Three" ad essere adattabile all'italiano (grazie alle librerie Infit, di Giovanni Riccardi). Il 30 aprile 2006 è stata rilasciata una versione beta di prova della versione 7 di Inform, che utilizza un rivoluzionario linguaggio di programmazione modellato sul linguaggio naturale (ovvero, con Inform 7 le avventure si scrivono "in inglese"). Inform può creare avventure in formato Z-code (che poi sarebbe il vecchio standard della Infocom, solo testuale) o in formato Glulx (con capacità multimediali). Interpreti Glulx esistono per i principali sistemi operativi, e addirittura gli interpreti Z-code sono disponibili per praticamente ogni computer (dal C64 all'Amiga, dallo Spectrum 128 ai Pentium) e addirittura per palmari e smartphone.

Documentazione: ottima
Supporto: ottimo
Usabile in italiano: sì

Documentazione in italiano: buona
Licenza: freeware
Home Page: http://www.inform-fiction.org
Home Page Italiana: http://www.inform-italia.org

TADS

di Michael J. Roberts,
Attualmente la versione ufficiale di TADS è la release 2, anche se da ormai più di un anno è disponibile la versione beta della release 3, che è comunque usabile e usata (nonostante una certa carenza di documentazione).

TADS (soprattutto la release 3) è, tra i "big three" il linguaggio di programmazione più rigoroso e elegante, il più simile a linguaggi ad oggetti non dedicati come il C++, Java o il C#. È probabilmente il più complesso da imparare, ma per contro è quello che offre maggior potenza. Esistono interpreti per giocare ad avventure scritte con TADS per i più diffusi sistemi operativi. Al momento con TADS non è possibile scrivere avventure in italiano.

Documentazione: buona per Tads2, sufficiente per Tads3
Supporto: ottimo
Usabile in italiano: no
Documentazione in italiano: no
Licenza: freeware
Home Page: http://www.tads.org

Hugo

della General Coffee Company Film Productions,
Il terzo linguaggio della serie, Hugo, è forse il meno usato. Offre comunque caratteristiche simili agli altri due, ed è stato usato per produrre giochi di alta qualità (tra i quali Future Boy, uno dei pochi esempi di Avventura Testuale rilasciata commercialmente).

Come sintassi è più simile ad Inform 6 che a TADS, ed è attualmente l'unico strumento che permette di utilizzare dei filmati come contenuti multimediali. L'interprete per Hugo è disponibile su praticamente ogni sistema operativo.

Documentazione: buona
Supporto: buono
Usabile in italiano: no
Documentazione in italiano: sì
Licenza: freeware
Home Page: http://www.generalcoffee.com/hugo.html

Qualcosa di più semplice

I "Big Three" sono strumenti potenti e capaci di soddisfare qualsiasi esigenza, ma non sono facilissimi da imparare. Ci sono delle alternative interessanti che riscuotono un discreto successo tra gli autori di lingua anglosassone.

Adrift

di Campbell Wild,
Adrift è un programma che permette di realizzare Avventure Testuali non mediante un linguaggio di programmazione, ma con un interfaccia basata su icone e menu a discesa (in stile Windows), e risulta notevolmente più potente e flessibile rispetto ad altri programmi che condividono la stessa filosofia. È possibile inserire immagini e suoni. Sia il programma per creare avventure che quello per giocarle funzionano unicamente sotto Windows, ma esistono programmi alternativi per giocare le avventure scritte con Adrift anche sotto altri sistemi operativi. Esiste una libreria per realizzare avventure in Italiano con Adrift, ma a oggi non è mai stata usata.

A differenza della maggioranza degli altri sistemi (che sono gratuiti), il programma per creare avventure è shareware, la registrazione costa 18.95$. Senza registrazione il programma funziona, ma con limitazioni sul numero di stanze e oggetti.

Documentazione: buona
Supporto: buono
Usabile in italiano: sì, ma ancora non usato
Documentazione in italiano: no
Licenza: shareware (registrazione 18.95$)
Home Page: http://www.adrift.org.uk

Alan

di Thomas Nilsson,
Hugo è un altro linguaggio di programmazione ma, pur mantenendo una certa potenza, è discretamente più semplice di Inform, Tads o Hugo, e ha una sintassi che, per quanto verbosa, risulta molto più leggibile (soprattutto per un non programmatore). La versione 3 di Hugo (attualmente in alpha-testing) introduce i concetti di classi e oggetti, nonché elementi multimediali (immagini e suoni), e rappresenta così una possibilità interessante per chi non vuole imparare un linguaggio complesso come i “big three”. Compilatori e interpreti per Hugo sono disponibili per i maggiori sistemi operativi. Non esiste ancora la possibilità di usare Hugo per creare avventure in italiano.

Documentazione: sufficiente
Supporto: sufficiente
Usabile in italiano: no
Documentazione in italiano: no
Licenza: freeware
Home Page: http://www.alanif.se/

Prodotti “nostrani”

Anche in Italia sono stati realizzati strumenti per la creazione di Avventure Testuali.

Modulo Base

di Enrico “Erix” Colombini,

Il Modulo Base non è un sistema a se stante, ma si appoggia sul linguaggio di programmazione Basic (tipicamente il dialetto usato è il QBasic, ma non è difficile convertire il tutto per altri dialetti). Con il Modulo Base si possono realizzare Avventure Testuali "vecchio stile", simili a quelle dello stesso Colombini (da "Avventura nel Castello" in poi) o a quelle di Scott Adams. Generalmente le avventure realizzate con il Modulo Base funzionano sotto MS-Dos o sotto Windows. Le capacità multimediali dipendono dal linguaggio Basic scelto.

Documentazione: buona
Supporto: sufficiente
Usabile in italiano: sì
Documentazione in italiano: sì
Licenza: freeware
Home Page: http://www.erix.it/avventure.html

M.A.C.

di Paolo Lucchesi,
M.A.C. è un programma per scrivere Avventure Testuali "vecchio stile", basandosi su un linguaggio di definizione relativamente semplice da imparare ma abbastanza farraginoso a causa della compattezza delle istruzioni scelte. M.A.C. funziona sotto Windows e sotto Linux, e permette anche l'uso di immagini grafiche.

Documentazione: buona
Supporto: sufficiente
Usabile in italiano: sì
Documentazione in italiano: sì
Licenza: gpl
Home Page: http://mac.paololucchesi.it/

Altri Sistemi

Per quanto quelli descritti fino ad ora coprano la quasi totalità del mercato, esistono molti altri sistemi che potrebbero avere i loro lati interessanti.

Quest

Di Axe Software / Alex Warren,
Quest permette di usare due diversi metodi per creare un'avventura testuale: un linguaggio di definizione chiamato ASL, o un programma basato su menu a discesa chiamato QDK. La cosa più particolare di Quest è la possibilità di giocare le avventure in multiplayer, usando l'apposito programma QuestNet Server.

Quest funziona solo sotto Windows. Il programma per giocare le avventure è distribuito liberamente, ma il compilatore dei file ASL (senza di esso qualsiasi giocatore può leggere il listato del gioco), il programma QDK e il QuestNet Server per più di 3 utenti sono shareware o a pagamento (24.95$). Non permette di scrivere avventure in Italiano.

Documentazione: sufficiente

Supporto: scarso
Usabile in italiano: no
Documentazione in italiano: no
Licenza: shareware (registrazione 24.95$)
Home Page: http://www.axeuk.com/quest/

AAS

di "Roddy Ramieson",
AAS è un sistema composto da un programma di sviluppo/editor (AASide) e un interprete (AASexe), entrambi scritti in Java, basato su un linguaggio di definizione XML.

In realtà AAS, per quanto funzionante, è parte di un elaborato scherzo culminato il 1° Aprile 2003.

Documentazione: scarsa
Supporto: assente
Usabile in italiano: no
Documentazione in italiano: no
Licenza: freeware (shareware ai fini dello scherzo)
Home Page: http://ifarchive.giga.or.at/indexes/if-archiveXprogrammingXaas.html

AIEE!

di Mark Hughes,
Un sistema puramente interpretativo, basato su un linguaggio di definizione XML pensato per persone senza esperienza di programmazione. L'interprete è scritto in Java ed è quindi funzionante su ogni sistema operativo. Inoltre AIEE! permette l'uso di componenti multimediali (grafica e suono). Non è al momento possibile scrivere avventure in italiano.

Documentazione: sufficiente
Supporto: scarso
Usabile in italiano: no
Documentazione in italiano: no
Licenza: freeware
Home Page: http://kuoi.asui.uidaho.edu/~kamikaze/Aiee/

Creative Adventure Toolkit

di Philip Richmond,
Composto da un programma di sviluppo basato su icone e menu a discesa, e da un programma per eseguire le avventure, è un sistema semplice da apprendere ma anche abbastanza limitato. Per contro permette di usare grafica e suoni.

Funziona solo sotto Windows. È disponibile un'estensione per la lingua italiana.

Documentazione: scarsa
Supporto: scarso
Usabile in italiano: sì

Documentazione in italiano: no
Licenza: freeware
Home Page: http://www.richmond62.freeserve.co.uk/index.htm

IAGE

di Robin Rawson-Tetley,
Questo è un sistema per avventure testuali basato su di un architettura client/server, che permette anche di giocare in multi-utenza alla stessa avventura, su Internet o rete locale. L'avventura viene creata grazie ad un editor visuale, anche se la gestione di azioni e eventi deve essere fatta con un linguaggio di definizione. Tutti i componenti sono scritti in Java, e quindi funzionano su ogni sistema operativo. Il suo sviluppo sembra sospeso dal 2002. Non esiste un adattamento all'italiano.

Documentazione: scarsa
Supporto: assente
Usabile in italiano: no
Documentazione in italiano: no
Licenza: gpl
Home Page: http://ifarchive.giga.or.at/indexes/if-archiveXprogrammingXiage.html

IFML

di Jenny Schmidt,
Un sistema per AvvUn sistema per avventure testuali basato su di un linguaggio di definizione XML. Il sistema comprende un'interprete scritto in Java ed è pensato per essere eseguito (anche remotamente) attraverso un browser per Internet.

Documentazione: sufficiente
Supporto: scarso
Usabile in italiano: no
Documentazione in italiano: no
Licenza: zlib/libpng (open source)
Home Page: http://ifml.sourceforge.net/

JACL

di Stuart Allen,
Un altro sistema, puramente interpretativo, basato su di un linguaggio di definizione, e che può essere usato normalmente o come server HTTP, permettendo di giocare le avventure (remotamente) attraverso un browser per Internet.

Documentazione: sufficiente
Supporto: scarso
Usabile in italiano: no
Documentazione in italiano: no
Licenza: gpl o altra open source
Home Page: http://jacl.sourceforge.net/

JZuul

di Marcus Thiensen,
Un sistema per avventure testuali composto da un interprete e un editor per il linguaggio, entrambi scritti in Java (e quindi portabili su ogni sistema operativo), contraddistinto da buone capacità grafiche, un aspetto accattivante e la possibilità di giocare in multiplayer. Le avventure vengono scritte in un linguaggio di definizione XML. JZuul può essere usato per scrivere avventure in inglese o in tedesco (ma non in italiano).

Documentazione: scarsa
Supporto: assente
Usabile in italiano: no
Documentazione in italiano: no
Licenza: gpl
Home Page: http://www.jzuul.org/

OliText

di Jacek Olszak,
OliText è un insieme di routine e funzioni usabile per scrivere avventure testuali in Java. Le avventure così prodotte possono quindi funzionare su qualsiasi sistema operativo (basta che ci sia installato il JRE).

L'home page offre varia documentazione, ma in parte incompleta. OliText non può essere usato per scrivere avventure in italiano.

Documentazione: scarsa
Supporto: scarso
Usabile in italiano: no
Documentazione in italiano: no
Licenza: gpl
Home Page: http://olitext.org/www/

PAWS

di Roger Plowman,
PAWS è un sistema per creare avventure testuali in Python. Per questo le avventure scritte con PAWS possono girare su qualsiasi sistema operativo, ma soltanto se sul computer è stato installato Python. Per contro l'approccio programmativo di Python è decisamente interessante.

Al momento PAWS permette la realizzazione solo di avventure testuali in inglese.

Documentazione: sufficiente
Supporto: scarso
Usabile in italiano: no
Documentazione in italiano: no
Licenza: freeware (sorgenti distribuiti)
Home Page: http://members.nuvox.net/~zt.wolf/PAWS.htm

PHPAdventure!

di Steven Kollmansberger,
Un sistema basato sul web per scrivere avventure giocabili via browser (basta avere un sito internet con PHP a disposizione per ospitare le avventure). La cosa più particolare è che le avventure create non usano il parser, ma una meccanica point'n'click.

Purtroppo il sistema è ancora in alfa e abbastanza instabile, e non disponibile in italiano.

Documentazione: assente
Supporto: assente
Usabile in italiano: no
Documentazione in italiano: no
Licenza: gpl
Home Page: http://phpadventure.sourceforge.net/

Python Universe Builder

di Joseph J. Strout,
Un sistema per creare avventure testuali in Python, simile in pregi e difetti a PAWS. Per quanto ancora immaturo, ha caratteristiche interessanti come una limitata gestione in real time e la possibilità di gioco in multiutenza.

Documentazione: sufficiente
Supporto: scarso/assente
Usabile in italiano: no
Documentazione in italiano: no
Licenza: gpl / lgpl
Home Page: http://py-universe.sourceforge.net/

SUDS

di Andy Elliot,
SUDS è un sistema composto da due programmi, Constructor e Player. Il primo consente di realizzare un gioco con un'interfaccia visuale a menu e icone, mentre il secondo serve ovviamente a giocarle. La cosa più particolare è che le avventure realizzate con SUDS non hanno un parser ma piuttosto un'interfaccia grafica (pulsanti per i verbi e per gli oggetti visibili).

SUDS funziona solo sotto Windows.

Documentazione: sufficiente
Supporto: scarso
Usabile in italiano: no
Documentazione in italiano: no
Licenza: freeware
Home Page: http://www.btinternet.com/~sudslore/

Conclusioni

Abbiamo cercato di fare un elenco più completo possibile. E' quasi inevitabile che qualcosa sia stato dimenticato, ma riteniamo che questo elenco possa

offrire un buon colpo d'occhio globale sul mondo degli strumenti per scrivere avventure testuali. Infine, in questa lunga panoramica, abbiamo volutamente trascurato tutti i sistemi per creare avventure a scelte multiple, avventure ipertestuali o libri game. Contiamo di affrontare questo argomento al più presto.

Narrazione e avventure

di Francesco Cordella

Quanto conta la qualità della narrazione in un'avventura? E come si ottengono risultati soddisfacenti? Queste due domande sono alla base del presente articolo che affronta un argomento complesso, ostico: la qualità della narrazione nelle avventure. Per maggiore chiarezza, ho pensato allora di porre la questione sottoforma di autointervista.

Dunque, da dove cominciamo? Da una premessa importante. Normalmente le avventure vengono divise in due grandi categorie: avventure fondate sugli enigmi, le cosiddette vecchiascuola, e le avventure fondate sulla narrazione, quelle maggiormente in voga oggi. Faccio due esempi italiani. Avventura nel castello di Enrico Colombini, per esempio, non può definirsi un'avventura narrativa. Non ha la struttura di un racconto, non ha, per esempio, flashback che danno corpo alla storia, non ha personaggi con profondità psicologica, in realtà non ha una vera e propria storia: è un'appassionante caccia al tesoro con tanti enigmi da risolvere. Una classica avventura vecchiascuola.

Beyond di Mondi Confinanti (il trio Grassi-Lucchesi-Peretti) è invece un'avventura narrativa, con una storia corposa ed enigmi funzionali allo svolgimento di questa storia. Questo non significa, però, che l'una sia meglio dell'altra: sono semplicemente due modi diversi di concepire le avventure. Se vogliamo, quella tra avventure non narrative e narrative è una semplificazione: perché, in fondo, esistono solo belle avventure e brutte avventure. E per fare una bell'avventura serve una buona qualità di narrazione.

Sta dicendo che, paradossalmente, anche in un'avventura non narrativa conta… la qualità della narrazione? Proprio così.

Scusi, ma che cosa intende di preciso per qualità della narrazione? Per me, è data principalmente da tre fattori: una buona scrittura, una buona storia e una buona struttura della storia, anche perché una buona struttura sta alla base di buoni enigmi. Detto questo, è chiaro che anche le cosiddette avventure vecchiascuola possono avere questi elementi (storia, buona scrittura, buona struttura), che sono appunto elementi narrativi. Penso alle avventure di Scott Adams: The Count, Voodoo Castle eccetera, nonostante le descrizioni telegrafiche e il parser elementare (finanche stucchevole), sono a modo loro dei racconti. Anche Avventura nel Castello lo è. Poi, certo, chi vuole puntare più sulla narrativa in senso stretto va oltre: cura l'introspezione dei personaggi, inserisce flashback, offre magari più punti di vista con un cambio di personaggio. Ma un punto fermo resta: la qualità della narrazione è importante in qualsiasi gioco.

Insomma, semplificando, si può dire che secondo lei per scrivere una buona avventura servono, di base, tre cose: scrittura, storia, struttura. Partiamo dalla scrittura: se l'autore non ha una buona penna può comunque scrivere una buona avventura testuale? Direi proprio di sì. L'avere una buona penna conta fino a un certo punto: se la storia è buona e la struttura è ben costruita, il gioco sarà comunque di valore anche se chi la scrive non ha grandi doti di narratore.

Poi, ovvio che una buona scrittura valorizza un'avventura.

Può darci qualche consiglio per ottenere una buona scrittura? Be', questo dovrebbe chiederlo a uno scrittore. Io posso solo dirle che, come ripete chi scrive per professione, per imparare a scrivere bene, o migliorare la propria scrittura, bisogna innanzi tutto scrivere molto. E poi leggere, naturalmente. Se può servire, a volte, nei miei giochi, ho usato il metodo leggi-e-fuggi.

E che cos'è questo metodo leggi-e-fuggi? Quando ero alle prese con la descrizione di una stanza, o di un personaggio, o un dialogo, e non ero soddisfatto del risultato, mi fiondavo davanti alla mia libreria e cercavo qualcosa da cui trarre ispirazione. Che ne so: magari sfogliavo i Buddenbrook per vedere come Thomas Mann tratteggia un personaggio, Ellroy per la descrizione di un ambiente, Auster per entrare in un tono malinconico che mi serviva in quel momento.

Questo vale anche per la scrittura di un racconto. Ha qualche suggerimento specifico per la scrittura di un'avventura? Premessa: checché se ne dica, sono convinto di una cosa: giocare a un'avventura non è come leggere un racconto. L'avventura è un gioco e come tale va trattato. Non è detto che in un'avventura uno debba dare sfoggio delle sue capacità di scrittura. Detto questo, passiamo a qualche banale consiglio. In genere, nella scrittura di un'avventura non amo la verbosità: preferisco poche righe per la premessa, la descrizione delle stanze, degli oggetti, i dialoghi. Secondo me le descrizioni lunghe rallentano il ritmo, rischiano di annoiare. E le descrizioni sono importantissime, soprattutto quelle delle stanze".

Perché contano tanto le descrizioni delle stanze? Perché sono i biglietti di presentazione dei nuovi ambienti. Vanno perciò curate al massimo, rese appassionanti, ma allo stesso tempo non verbose. Non mi piace infarcire la descrizione di oggetti, meglio metterli a parte, separati da una riga dal corpo della descrizione stessa: nella descrizione vera e propria va invece creata l'atmosfera, magari inserendo dei "rumori" ("dal soffitto provengono lo zampettio di un cane e il fastidioso strisciare delle sedie che si muovono, segno che l'appartamento al piano di sopra non è disabitato come gli altri in questo palazzo").

E i personaggi? E' importante caratterizzarli, dare nella loro descrizione qualche indicazione, anche fugace, sul modo di porsi, sull'espressione, l'aria che hanno o anche i vestiti che indossano. Inoltre, è bene dare l'idea che siano "vivi": se resto per molte mosse in una stanza in cui c'è un personaggio, ogni tanto una frase tipo "Giorgio si versa un altro bicchiere di vino e beve", inserita così, tanto per dare colore e movimento, dovrebbe esserci. Altrimenti il personaggio rischia di risultare una buona statuina. E occhio al protagonista: in molte avventure vecchiascuola, da Colossal Caves a Zork, il protagonista (PC, player charachter) è un "blank hero", cioè una persona senza identità, una pedina-testadilegno senza passato né presente. Penso sia più utile, invece, dargliela, un'identità: dare una storia al protagonista, tratteggiandolo nella descrizione, in qualche flashback o in qualche frase buttata qua e là che aiuti a caratterizzarlo meglio.

Passiamo alla storia. La storia è tutto. Ma non deve essere necessariamente "I pilastri della terra", sia chiaro. L'importante è che ci sia. Avventura nel Castello, in fondo, ha una storia: un tizio intrappolato in un castello cerca una

via d'uscita. Va bene anche così se voglio ottenere un'avventura vecchiascuola-caccia al tesoro. Ma se voglio qualcosa di narrativamente più interessante, allora devo lavorare di più sulla storia, sviluppare l'idea di partenza. Io non sono di quelli che dicono: voglio scrivere un'avventura, ora cerco una storia. Io parto dalla storia, o almeno da un'idea. L'idea che poi ha portato alla "storia" di Flamel, ad esempio, me l'ha ispirata Trilogia di New York di Paul Auster, mi piaceva l'idea di sconvolgere la vita di un uomo solo, solitario, grigio. Sono partito da lì. La prima regola che cerco di seguire io è evitare situazioni stereotipate e personaggi-cliché, ovvero ciò che si è visto e rivisto, letto e riletto. Sia chiaro, però, uno può anche prendere una situazione stereotipata e trattarla in modo originale, l'importante è evitare il cliché puro, che annoia. Per dare corpo alla storia, inoltre, può essere utile inserire "sottostorie" rispetto a quella originale. Mi spiego. Mettiamo che io abbia una storia non troppo originale: il protagonista deve evadere da un carcere. Per valorizzare questa storia posso aggiungere delle sottostorie: per esempio, durante le sue peripezie il protagonista scopre che il cappellano del penitenziario è minacciato da un detenuto e devo aiutarlo. Oppure: scopro che la direttrice ruba soldi e voglio incastrarla. Queste sottostorie possono aggiungere interesse, sostanza, alla storia principale. E qui entra in gioco il terzo elemento della qualità narrativa: la struttura, importantissima anche per gli enigmi.

Ma che cos'è di preciso la struttura e come può valorizzare un'avventura? E' il modo in cui viene raccontata una storia, come viene costruita: lineare, a incastri, con flashback. Prendiamo Spider And Web di Andrew Plotkin, storia è quella di un uomo che deve "espugnare" un palazzo. Costruita come l'ha costruita Plotkin, con continui salti avanti e indietro nel tempo, la storia diventa ancora più accattivante. Altri esempi ancora di struttura: una storia con più finali; un cambio di personaggio. Storie con strutture così assumono valore. Nel cinema, pensate al film Pulp Fiction: se fosse costruito in modo lineare, senza la struttura a incastri, sarebbe altrettanto interessante? Naturalmente la struttura va pensata bene all'inizio, prima di mettersi a scrivere l'avventura.

Lei prima ha detto che una buona struttura sta alla base di buoni enigmi: si spiega meglio? Be', curando la struttura, inserendo sottostorie, flashback interattivi, un cambio di personaggio, nascono automaticamente nuovi enigmi. E nascono più facilmente enigmi legati alla storia. Ecco perché la struttura è così importante. Certo, creare una storia non lineare è più difficile, richiede più impegno, ma il gioco vale la candela.

A sentire lei, insomma, la qualità della narrazione ha davvero grande peso in un'avventura. Proprio così. Ha peso benché, come ho detto, le avventure sono e restano dei giochi, mentre la qualità della narrazione è un concetto più vicino alla letteratura. Tuttavia tra giochi-avventure e qualità della narrazione, come abbiamo spiegato, c'è un forte legame. Non a caso, tempo fa feci un esperimento, a partire dalla domanda: ma dai grandi romanzi, dai capolavori si possono trarre buona avventure? Per dare una prima risposta, creai un'avventura tratta da un episodio dell'Odissea: "L'avventura del Ciclope". E scoprii che il testo originale, di per sé, conteneva descrizioni di stanze che sono perfette per un'avventura e offriva anche buoni enigmi. Quando leggo un romanzo, mi chiedo sempre se, senza modifiche corpose, possa essere trasformato in un'avventura. E a volte la risposta è sì. Alcuni capolavori della letteratura potrebbero essere grandi avventure: dalla Divina Commedia al

Nome della Rosa, dalla Repubblica di Platone ai Pilastri della Terra di Ken Follett ci sono opere che potrebbero essere quasi prese pari pari e diventare narrativa interattiva. Insomma, la qualità della narrazione conta, eccome.

Fondamenti di IF Game Design

di Roberto Grassi

Questo numero di Terra d'IF si focalizza sugli aspetti della creazione di un gioco d'Interactive Fiction. In prima approssimazione, i tre aspetti fondamentali possono essere suddivisi in:

- Programmazione
- Narrazione
- Game Design

Gli altri due aspetti sono stati trattati nei corrispondenti articoli. Questo, invece, si soffermerà sulle problematiche di Game Design. In altre parole, si cercherà di rispondere alla domanda: "Sono in grado di programmare ed ho in mente una storia meravigliosa. Cosa mi manca per scrivere un gioco d'Interactive Fiction?" Cercherò di dare qualche risposta nelle righe seguenti, ben sapendo che non ci sono regole che funzionino in modo assoluto e che ognuno degli aspetti descritti di seguito potrebbe essere oggetto di una lunga trattazione.

Cos'è il Game Design

Iniziamo innanzitutto dal capire cosa s'intende per "Game Design". La parola inglese "design" significa progettazione ed indica un insieme concertato di conoscenze, azioni, metodologie e strumenti finalizzati al raggiungimento di uno scopo, che rappresenta l'aspetto fondamentale d'ogni attività di progettazione per l'industria. Si tratta di un processo completo e articolato che parte dalle primissime fasi d'esplorazione e generazione di un'idea (nota come "concept design") e si svolge fino alla definizione finale di un prodotto e la sua collocazione sul mercato[1]. E passiamo quindi al Game Design. Il "Game Design" è il processo della progettazione del contenuto, del background e delle regole del gioco. Un documento che descrive la progettazione di un gioco potrebbe anche essere chiamato "design document". I game designer professionali si specializzano in particolari tipi di giochi, come ad esempio i giochi da tavolo, i giochi di carte o i video games[2]. In poche parole, il Game Design è il processo di costruzione delle meccaniche e dei contenuti del gioco. Alcune regole di "buon" design sono comuni ai vari tipi di giochi, mentre altre sono specialistiche per un genere particolare.

Nozioni Fondamentali

In quest'articolo non considererò le fasi d'esplorazione di un'idea fino alla sua collocazione sul mercato (come richiede il significato della parola design, vista sopra) dato il carattere non commerciale dell'IF. Mi soffermerò, dunque, sulla fase di generazione e definizione finale.

1 Vedi Wikipedia: http://it.wikipedia.org/wiki/Design
2 Traduzione da Wikipedia: http://en.wikipedia.org/wiki/Game_design

Conoscenza del mezzo

E' fondamentale, a mio parere, essere 'padroni' del mezzo. E per mezzo intendo il genere di gioco e gli strumenti che sono usati sia per scrivere che per giocare. La padronanza la si acquisisce con il vecchio adagio caro a Paolo Lucchesi "Giocare, giocare, giocare... meglio se ai giochi migliori." Non si può che essere d'accordo. Solo giocando molto e osservando come gli altri autori hanno affrontato alcuni problemi ci si può rendere conto di alcune finezze di creazione. A questo scopo è anche interessante giocare lo stesso gioco con interpreti diversi, perché le possibilità di layout di un interprete o di una piattaforma potrebbero essere importanti. Adrift, ad esempio, ha un'interessante funzione di AutoMap che potrebbe essere utile considerare, in fase di generazione di un'idea. Un altro esempio potrebbe essere il fatto di voler decidere di utilizzare immagini e suoni. Oppure dei filmati. Non tutte le piattaforme consentono di fare le stesse cose, oppure con la stessa semplicità. Se non si dà un'occhiata alle diverse possibilità che esistono, si finisce col pensare in maniera schematica e si parte prima dall'applicativo che si intende usare e poi si scrive il gioco di conseguenza. Questo è sbagliato. Difatti, può essere che, in fin dei conti, la nostra idea sia meglio se viene realizzata come un racconto a scelte multiple, piuttosto che come narrativa interattiva tradizionale.

Pensare come il giocatore

Questo aspetto è da tenere in grande considerazione, perché potrebbe portare al successo o al fallimento il vostro gioco. Si trovano alcuni articoli interessanti sull'argomento (Grenade)[3], ma proverò qui a riassumere brevemente cosa s'intende. Due cose devono essere molto importanti. Gli obiettivi devono essere chiari da subito e tutti gli sforzi devono essere orientati ad aiutare il giocatore anziché complicargli la vita. Queste raccomandazioni sono così categoriche perché, in particolare nel nostro genere di giochi preferito, il giocatore tende "da solo" a perdere di vista gli obiettivi e a complicarsi la vita. Per questo motivo dobbiamo sempre tenere a mente questo aspetto ed essere sempre pronti a dare al giocatore le informazioni utili nel momento giusto o quando lui prova a cercarle. Il blocco-note dell'ispettore in Beyond s'ispira a questo principio e consente di tenere a mente gli aspetti rilevanti del gioco, in modo che il giocatore può, in un colpo solo, ricordarsi tutto ciò che è importante ricordare. Mi è stato ispirato da un gioco che non ha avuto molta fortuna, ma che implementava questo concetto in maniera semplice "Murder at the Aero Club" del 2004. Non so se si tratti di un'idea originale, ma a me è piaciuta.

Il Mondo

E veniamo ad un aspetto dei più controversi e dibattuti. Quello che riguarda gli aspetti di coerenza, mimesi e simulazione del mondo in cui sarà ambientato il nostro gioco. Credo che, per dare una risposta soddisfacente,

3 Vedi Brass Lantern: http://brasslantern.org/editorials/wrong.html

debbano fare da guida la storia e le interazioni più rilevanti che il giocatore può avere. Se il gioco è orientato agli aspetti narrativi, allora una simulazione spinta (anche relativamente al numero ed al dettaglio degli ambienti) sarebbe con ogni probabilità fuori luogo. Probabilmente sarebbe più opportuno approfondire gli aspetti di narrazione legati ai personaggi non giocanti (se ce ne sono) oppure ad aspetti della narrazione legati agli aspetti della conoscenza del personaggio nel mondo giocato (relazione con gli oggetti e con le stanze). In ogni caso, non si deve fare l'errore, una volta scelto il tipo di approccio 'narrativistico' o 'simulativo', di passare con disinvoltura da uno all'altro. Il giocatore si aspetta che il mondo di gioco sia dotato di una coerenza e consistenza interne. Se poco prima il giocatore è entrato in una meravigliosa stanza adorna di un mucchio di oggetti ma non poteva interagire con nessuno di essi, suonerebbe una nota stonata se all'improvviso dovesse interagire con un cassetto dettagliato fin nei minimi particolari. Non posso dimenticare - spero che Caldarola mi perdoni, in nome della nostra corregionalità - un armadio colmo di delicati bicchieri di cristallo nella casa di un rude cow-boy.

Best Practices

In questa sezione invito il novello autore a leggere, studiare ed informarsi facendo tesoro di chi prima di lui ha provato, con successo o meno, a scrivere giochi di Narrativa Interattiva. In questo senso, sarebbe certamente molto utile leggere i consigli di Graham Nelson[4], i "Crimini contro la mimesi"[5] , gli articoli di Emily Short[6] ed articoli sul game design in generale. Chi scrive quest'articolo cura una rubrica di "Design Review" che si sofferma sugli aspetti di design prendendo esempi dalla pratica dei giochi. Talvolta, infine, sul newsgroup RAIF nascono interessanti discussioni sugli aspetti di game design

Interazioni soddisfacenti

Qui arriviamo agli aspetti di game design che devono occuparsi di "rendere la sessione di gioco" qualcosa di significativo ed emozionante. In prima approssimazione, si tratta di tre tipi di interazioni:

1. Interazioni tra il giocatore ed il suo alter-ego. In questo senso, i comandi da gestire sono quelli del tipo "inventario", "x me" oppure quelli di vestirsi/spogliarsi. La corretta gestione di questi comandi dipende sostanzialmente dal tipo di gioco che vogliamo gestire. Nella maggior parte dei casi, il gioco non mi obbligherà a dichiarare TUTTI gli oggetti che voglio indossare (arrivando sino alle mutande, magari...). E' ovvio, tuttavia, che se un gioco esalta molto gli aspetti di simulazione ed in particolare quello sugli aspetti di vestizione/svestizione, deve tenere in forte considerazione interazioni e risposte soddisfacenti.
2. Interazioni tra il giocatore ed i personaggi non giocanti. Queste sono le più difficili da gestire, perché dipendono dalla cura e dal dettaglio

4 *Carta dei Diritti del Giocatore*. Di Graham Nelson. Tradotto in Italiano sul numero 7 di Terra D'IF, http://www.terradif.net/magazine/numero-7/carta-dei-diritti-del-giocatore

5 *Crimes Against Mimesis*. Di Roger Giner-Sorolla. Tradotto in Italiano da Francesco Cordella, http://www.avventuretestuali.com/inventario/crimes-against-mimesis

6 Vedi http://emshort.home.mindspring.com/

d'implementazione che desideriamo dare al personaggio non giocante. In molti casi, probabilmente la maggior parte, i personaggi non giocanti sono statici (nel senso che non cambiano di locazione) e rispondono come automi solo ad alcuni argomenti. Non hanno obiettivi e non sono dotati di vita propria. In altre parole, il personaggio non giocante potrebbe essere sostituito senza problemi da un 'manichino'. Naturalmente, nei giochi che hanno velleità più narrative un'implementazione di questo tipo stride ancora di più, ma anche nei giochi più orientati al 'gioco' ed alla 'simulazione' un personaggio non giocante implementato nel modo suddetto è assolutamente insoddisfacente.

3. Interazione tra il giocatore e gli oggetti del mondo simulato. Questi tipi di interazioni sono molteplici, ma sono in qualche modo più prevedibili. A differenza dei personaggi non giocanti, non ci si aspetta che gli oggetti siano dotati di vita propria ed abbiano obiettivi. Questo semplifica di molto il loro corretto trattamento. Non mi soffermo molto su questo argomento perché l'interazione con gli "oggetti" ha una lunga tradizione nel mondo della Narrativa Interattiva e quindi i requisiti minimi per un'interazione soddisfacente sono generalmente noti.

Enigmi

Questo aspetto deve discendere direttamente dall'approccio che stiamo dando al mondo. Gli enigmi devono essere 'ingranati' nel gioco. Non c'è cosa peggiore di un enigma appiccicato nel gioco con il solo scopo di prolungare la frustrazione del giocatore. Quando gli enigmi sono correttamente inseriti nel gioco, il giocatore tende ad approcciarli in maniera logica ed a risolverli con maggiore facilità perché, dell'enigma, intuisce gli obiettivi ed i meccanismi per risolverlo. In altre parole, quando si prendono in considerazione gli enigmi credo che debba valere il seguente adagio: "Metti solo gli enigmi che servono, quando servono e solo se sono logicamente inseriti nella storia e nella logica del gioco." In altre parole, se ci si accorge che il capolavoro che pensavamo di avere in testa si tramuta in un gioco d'IF "senza enigmi", non disperiamo. Scriviamolo comunque. Non ha nessun senso inserire un enigma in maniera forzata, per il solo gusto di far perdere tempo al giocatore.

Il design di Risorgimento Represso

Dietro le quinte di RR (postato su r.a.i.f. il 14/03/06)

di Michael Coyne
traduzione di Max "torredifuoco" Bianchi

Scrissi la seguente risposta ad una domanda sul mio blog, nella quale si chiedeva se fossi partito da un trascritto nella creazione di Risorgimento Represso, e pensai che avrebbe potuto interessare la gente del gruppo di discussione dal punto di vista del mestiere dell'IF... e mi sembra anche appropriato, dato il rilascio del codice sorgente e il terzo anniversario dall'inizio della scrittura di Risorg1...

L'ho scritta avendo in mente un pubblico più generico di r.a.i.f., di qui i riferimenti chiarificatori alla HitchHikers Guide to the Galaxy e agli enigmi del Babel Fish che tutti voi probabilmente conoscete già...

No, precipitai dentro a Risorg1 senza un trascritto.

Infatti, quando iniziai a programmare, non mi resi conto che stavo scrivendo un gioco. Cominciai con un racconto breve che avevo scritto (o cominciato a scrivere) in precedenza, e pensai di trasformarne la prima parte in un'avventura solo per apprendere Inform. Volli anche provare il codice per gli incantesimi alla Enchanter contenuto nel gioco esempio per Inform Balances.

Nella versione originale, si deve ritrovare la pergamena di un incantesimo Fitzmo e utilizzarla per sbloccare le tubature fognarie in cantina, recuperare gli occhiali di Ninario, fare in modo che ti rimandi a casa, e tutto finisce lì.

Quando terminai, decisi che non mi piaceva proprio avere delle pergamene per gli incantesimi in giro; mi sembrò troppo derivato e poco originale. Inventai quindi un modo diverso per sbloccare le tubature.

A quel punto, valutai la mini-avventura che avevo, e quanto fosse relativamente facile da produrre, e mi resi conto che avevo l'inizio di un'avventura propriamente detta. Mi sedetti e pensai un po' agli elementi che volevo nel gioco.

Volli qualche enigma multi-livello, come quello del Babel Fish in Hitch Hiker's Guide to the Galaxy, l'avventura che Douglas Adams scrisse in collaborazione

con la Infocom. Si può notare l'ispirazione al Babel Fish negli enigmi di Risorgimento Represso riguardanti il "passare l'orso" ed "entrare nella fattoria", dove ogni soluzione che si possa pensare ad un problema immediato conduce ad un problema nuovo ed imprevisto – sebbene allo stesso tempo sia un passo necessario verso la soluzione.

Volli lasciare lo stereotipato vecchio mago decrepito dentro; trasformare un personaggio così simile ad un cliché in qualcosa d'interessante e divertente fu una sfida che volli affrontare.

Non seppi che fare di Ninario per la durata dell'intera partita, e come liberare il giocatore senza lasciare Ninario tristemente seduto nel suo studio. Parimenti, non volli che Ninario vagasse solamente nel suo piccolo gruppo di locazioni... così, farlo rapire fu il modo più adatto per toglierlo di mezzo una volta che le sue possibilità come NPC fossero esaurite.

L'utilizzo non autorizzato della magia da parte di Ninario mi sembrò la spiegazione perfetta per il suo rapimento. Nel breve racconto originale, fu la ragione della sua fuga dalla sua fortezza con il protagonista, così si tradusse piuttosto facilmente all'interno del gioco. E quindi, la mano pesante della Gilda dei Maghi nei confronti di Ninario diede origine all'intera idea della Gilda che sopprimeva le innovazioni scientifiche; l'idea del risorgimento represso che è, naturalmente, il significato del titolo.

Volli un orso, per deferenza verso l'originale Colossal Cave (cioè Adventure).

Una volta che seppi che volevo un orso, ebbi la necessità di avere un posto dove metterlo... quindi, perché non un sentiero di montagna? E il luogo dell'orso sembrò un posto ideale per un enigma alla Babel Fish.

Naturalmente, se passare l'orso sarebbe stato un enigma pieno di significato e multi-livello, ci voleva qualcosa dall'altra parte [N.d.T. oltre l'orso, che funge da guardiano della soglia]. Una cima solitaria e spazzata dal vento, sembrò il luogo migliore per un eremita stanco del mondo. E, una volta che ebbi l'eremita, cosa sarebbe stato più naturale se non avere una bottiglia a chiusura ermetica?

Infine, sapevo di voler fare un gioco "elegante" [N.d.T. e con del formaggio, "cheesy" ha un doppio senso intraducibile] (vedi http://emshort.home.mindspring.com/cheese.htm).

Una volta deciso quello, seppi anche che Renaldo, il capo malvagio della Gilda dei Maghi, avrebbe dovuto essere intollerante al lattosio. Non è un pezzo gigantesco del gioco e c'è, infatti, solo una minuscola parte dove è possibile

scoprirlo, ma ha modificato la mia visione di quel personaggio. Di conseguenza decisi che il Negozio del Formaggio in Vechlee sarebbe stato un'altra vittima della Gilda dei Maghi, con un cartello affisso fuori recante l'avviso della sua chiusura. Apparentemente, sarebbe stato per l'introduzione di metodi di produzione più moderni, ma dietro le quinte, seppi che era l'intolleranza al lattosio di Renaldo.

Ricominciai con questi elementi indispensabili, insieme ad una mappa del mondo di gioco. Alcune cose che volli derivarono dal dover collocare alcune cose sulla mappa. Altre volte, il disegnare la mappa guidò la selezione degli elementi voluti.

Quando completai questa fase, ebbi una mappa, e un'idea piuttosto buona degli elementi necessari. Molti enigmi saltarono fuori in maniera naturale dalla costruzione della mappa. Ebbi la torre e la fortezza di Ninario, e capii che una qualsiasi struttura del genere avrebbe avuto delle porte corazzate e un cancello – specialmente se il suo abitante avesse vissuto nella paura di rappresaglie della Gilda dei Maghi.

L'esigenza delle porte fece nascere il cannone, che portò all'enigma della polvere da sparo. La necessità di un cancello portò alla macchina idrolitica contro-bilanciante [N.d.T. !!], che è, credo, l'unico enigma dell'avventura con tre soluzioni distinte. Entrare nel capannone giunge appena secondo con due soluzioni piuttosto diverse.

Per semplificarla, volli che la macchina idrolitica contro-bilanciante fosse inutilizzabile una volta che l'enigma fosse risolto; e così feci, in modo che il testo la descrivesse mentre si fermava con un tubo flessibile che saltava. Quindi, pensai, piuttosto che avere il tubo penzolante dalla macchina, perché non farne un oggetto che saltava via e che venisse utilizzato in un secondo momento della partita? Quello portò all'intera idea di salire al piano superiore della Gilda dei Maghi, il carrello portavivande, la seconda bottiglia di Wizstrip, e mi permise di portare il giocatore a conoscenza dei retroscena che avevo su Renaldo (la sua intolleranza al lattosio) come una divertente conseguenza del mandare uno dei tanti pezzi di formaggio al piano superiore con il carrello portavivande.

Fu perciò uno sviluppo del genere uovo-e-gallina, con ogni elemento ad ingrassare gli altri, che crebbe in un'avventura molto più estesa di quanto non avessi anticipato in principio.

Con Risorg2 sto, partendo da un trascritto, costruendo un ambiente molto più strutturato. Non sono ancora sicuro se ciò mi stia aiutando o ritardando. Mi sta sicuramente aiutando a rimanere in una "modalità" alla volta. Sono o nella modalità programmazione o nella modalità scrittura, e posso essere più concentrato su qualsiasi cosa stia facendo al momento, ma potrebbe derivarne

un gioco più limitato quando tutto sarà detto e fatto.

Aspettiamo e vediamo...

Jason Devlin

intervistato da Roberto Grassi

> Jason, parlaci di te

Nella schermata di “ABOUT” dei miei giochi e nelle altre interviste che ho fatto, dico che sono un biologo e studente di chimica. Credo, ora che sono vicino a questi traguardi, che dovrei iniziare a dire cosa sarò. Ho in mente d’iscrivermi ad un certo numero di scuole di medicina e programmi per il dottorato di ricerca, o magari qualcosa in comune tra queste cose. A parte questo, sono assistente ricercatore in un laboratorio di chimica analitica.

> Parlaci dei tuoi giochi

Alcune persone sono dei programmatori. Altre sono scrittori. Alcune sono entrambe le cose. Altre nessuna delle due. Tendo a pensare che rientro in questa categoria. Mi arrabatto nella programmazione e scrivere non è la mia passione. Quello che mi piace è il divertimento. Mi piace pensare che i miei giochi rappresentino questa visione; scrivo giochi che io stesso vorrei giocare, e spero che valga lo stesso per gli altri. Penso che sia questo il motivo per cui Vesper (e con minore fortuna ‘Sting of the Wasp’) hanno ottenuto una buona accoglienza. Potrebbero non essere dei capolavori di scrittura o di programmazione ma, per la maggior parte dei giocatori, sono stati dei giochi divertenti che li hanno impegnati per un’ora o due.

> Cosa ne pensi dei vecchi giochi d’interactive fiction?

Ora che ci penso, non credo di aver mai completato un gioco di quelli della vecchia scuola. Ne ho iniziati molti ma non sono mai andato avanti molto. Appartengo ad una generazione di giocatori più giovane e più pigra. Poiché non ho nemmeno il fattore della nostalgia, se un gioco dovesse risultarmi noioso (rimanere bloccato, poche descrizioni, storia assurda) semplicemente non lo giocherei più. Ci sono molti altri giochi da risolvere.

>Parlaci della scena attuale

Penso che Jimmy Maher abbia riassunto molto bene lo stato attuale dell’IF nel suo recente editoriale su SPAG. Il nuovo secolo ha visto emergere molti giochi sperimentali che avevano la caratteristica della giocabilità, ma ora stiamo

applicando quelle idee sperimentali per servire lo scopo dei giochi, piuttosto che produrre giochi che servissero allo scopo della sperimentazione. Chancellor è un ottimo esempio di questo: il concetto di mondi 'interconnessi' è già stato fatto, ma di solito sono solo una vetrina che l'autore vuole mostrare. In Chancellor essi sono entrambi necessari alla storia. Blue Chairs e Snatches sono altri due giochi recenti che mostrano l'integrazione d'idee sperimentali con i giochi: entrambi con un ottimo successo.

> Come vedi il futuro dell'interactive fiction?
Inform 7 sarà rivoluzionario sotto molti aspetti, e sono sicuro che con esso saranno prodotti molti giochi di alta qualità. Non ho avuto modo di guardare in dettaglio la documentazione, ma sembra un ottimo sistema che consentirà di poter fare molte cose in maniera più semplice e con meno errori (non vedo l'ora di vedere i giochi dell'IFComp di quest'anno scritti con questo sistema). Ad ogni modo, i giochi scritti con I7 continueranno ad essere l'IF come noi la conosciamo attualmente. Ed è una buona cosa; spero infatti che l'IF convenzionale ed amatoriale continui indefinitamente. I7, magari, renderà l'IF più avvicinabile da parte dei non programmatori e forse alcuni scrittori famosi di fiction lineare decideranno di usarlo. Credo tuttavia che se l'IF dovesse fare breccia nel main(er)stream, si dovrebbe considerare un cambio radicale. Bene o male, la gente gradisce la grafica e molte persone potrebbero non interessarsi all'IF perché non ha figure. Penso che la versatilità del parser, combinata con un motore grafico, renderebbe l'IF più appetibile a molte di esse.

> Parlaci dei tuoi giochi futuri
Al momento, ho due progetti: uno in collaborazione insieme ad uno da solo. Il progetto solitario è molto diverso dai miei giochi precedenti e non credo che piacerà a molti. Penso che non piacerà alla maggior parte delle persone, ma quei pochi che lo apprezzeranno lo riterranno davvero buono, ed è questa la strada che voglio seguire questa volta, piuttosto che rilasciare un gioco che possa piacere alla maggior parte dei giocatori.

One Room Game Competition 2006

Voti e recensioni di Paolo Lucchesi

Frankenstein III

Prova di avventura testuale monocamerale.
(c) Corvo di Odino

Inguardabile e ingiocabile, tanto da far pensare ad un qualcosa di volutamente malfatto, da far pensare che l'autore aspiri al molto discutibile onore dell'ultimo posto.

A parte la brevità e l'esilità della trama, l'avventura è irrisolvibile senza guardare la soluzione: la "cabina" non viene mai nominata esplicitamente, la "grossa leva" rimane innotata fino al momento giusto e i due comandi fondamentali sono sbagliati ("ripara fusibile" invece di "sostituisci fusibile", "accendi leva" invece di "tira leva").

Inoltre la stanza è mal descritta, priva di qualsiasi oggetto nominato o supposto: dove sono i macchinari elettrici, le candele, il tavolo? e soprattutto, dov'è la creatura?

Infine a una prosa che lascia molto a desiderare di per sé si aggiungono errori di battitura e una grammatica che mal regge e che culmina nel sacrificio rituale di un congiuntivo proprio nell'ultima frase.

La cosa che si nota, in conclusione, è che quest'avventura non ha subito il minimo playtesting, probabilmente nemmeno da parte dello stesso autore. E questo è il suo problema principale.

Voto: 1; Aggettivo: "teribbile"

De Reditu

Avventura Testuale italiana.
Copyright (c) 2006 da Massimo Corso.

Ci sono molte piccole cose che non vanno, in quest'avventura. Qualche maiuscola di troppo, una prosa corretta ma spesso pesante a leggersi (e che di tanto in tanto ricorda i Britanni di Asterix), una giocabilità un po' macchinosa e dalla causalità abbastanza opinabile, i png un po' troppo monodimensionali e un po' poco reattivi. Si tratta di difetti minori, anche se mi ha particolarmente disturbato la disonestà di un oggetto che compare miracolosamente in un contenitore precedentemente vuoto. Certo, possiamo considerarli peccati veniali che spero vengano corretti in versioni future. Ma sicuramente inficiano la qualità globale del gioco.

C'è anche del buono in "De Reditu"; è ottima l'ambientazione (che poteva essere resa meglio, ma questo è secondo me uno dei limiti delle ORG) e anche l'idea di partenza; non è facile mettersi a giocare con la storia. E il finale, pur non originalissimo, non è affatto da buttare via.

In generale "De Reditu" si aggiunge alla lista delle buone idee sviluppate in

modo non eccepibile.

Voto: 5; Aggettivo: imperfetta

Lo Sforacchiato Giallo

Un'avventura testuale di Veronica Auretta e comp.

L'avventura non è particolarmente originale, e l'abitacolo in genere è già stato usato più volte come ambientazione one-room. Però l'ambientazione storica è particolare ed è ben resa, e il ritmo è frenetico al punto giusto.

Gli enigmi sono della giusta difficoltà (considerando che sono quasi tutti da risolvere "a tempo") e ben contestualizzati; per contro anch'essi difettano un po' di originalità (a meno di non considerare tutto il sistema di guida un enigma).

In generale tutta l'avventura è realizzata con sufficiente cura del dettaglio e non pare presentare grossi problemi. Ha la brevità e la limitatezza classiche delle avventure monocamerali ma, giocandola, si spera che Veronica produca presto qualcos'altro, qualcosa di più ampio respiro.

Voto: 7; Aggettivo: promettente

Il Diavolo a Venezia

Un crimine in maschera di Lorenzo Carnevale.
(Nota: sono stato betatester per questo gioco)

Non è male, quest'avventura. Intrigante nella trama e negli enigmi, sufficientemente ben scritta, programmata con competenza e senza grossi problemi, e con un protagonista deliziosamente e decadentemente amorale.

Per contro il gioco non è nemmeno esente da difetti: l'ambientazione veneziana è quasi impercepibile (in parte per la claustrofobica limitazione della singola stanza) e lo svolgimento degli eventi, nel suo volersi ispirare ad un'opera già esistente (o, peggio, volerla seguire nella trama), è nel finale abbastanza meccanico e poco intuibile da parte del giocatore.

In conclusione l'avventura lascia intuire le potenzialità di Lorenzo come autore, ma non le lascia esprimere a pieno.

Voto: 6; Aggettivo: intrigante

It's Easter, Peeps!

Copyright (c) 2006 by Sara Brookside

Inesorabilmente l'aggettivo che meglio definisce quest'avventura è "carina". Nel tema festivo e favolistico, nello stile allegro (ma non umoristico) e leggermente surreale, nei puzzle semplici e gradevoli (anche se la loro inclusione nel contesto è un po' debole), nell'inclusione di "feelies" estremamente pasquali e a loro volta "carini". E anche nella doppia versione z-code/adrift.

Da un punto di vista meramente tecnico, l'avventura è realizzata molto bene, con molto gusto del dettaglio e con molta cura, ed è in questo sicuramente

superiore a gran parte delle avventure nostrane iscritte alla ORGC2006. Per contro le si può imputare solo il fatto che, per sintassi e per oggetti simili, spesso obbliga il giocatore ad inserire comandi abbastanza verbosi.

Per chiudere, non si deve cercare qualcosa di sconvolgente o di straodinariamente originale in quest'avventura, ma solo qualcosa di piacevole da leggere e da giocare.

Voto: 8; Aggettivo: "carina"

Lazy Jones e l'ultima crostata!

Un avventura di Gabriele Lazzara e Carmelo Paterniti

È innegabile che le avventure di Lazzara e Paterniti siano in molti modi inconfondibili; nel bene e nel male (ma soprattutto nel male) hanno un loro stile. E ovviamente anche quest'ultima loro fatica non fa eccezione.

Su di una trama esile e che malamente si regge in piedi, i due autori costruiscono situazioni ancora più assurde e paradossali e ostacolano il giocatore con enigmi dalla logica quanto meno opinabile (senza mai preoccuparsi dell'esistenza di possibili soluzioni alternative e magari maggiormente logiche). Il tutto è narrato in un italiano a tratti "particolare" e in uno stile comico e umoristico, ma di una comicità facile e caciarona e che spesso non riesce affatto a divertire.

Anche la cura del prodotto è abbastanza approssimativa, con qualche errore di battitura, di grammatica (ah, questi congiuntivi) e di programmazione (perchè mai il frisbee dovrebbe essere chiuso?). Come in Italia succede spesso, la carenza di playtesting è evidente. Per contro anche questi problemi in fondo rimangono "in stile".

Voto: 3; Aggettivo: sconclusionata

Galeotto fu il Canotto (Tre modi per buttare l'ancora)

Un'avventura testuale di Andrea Rezzonico

Per la seconda volta Andrea Rezzonico ci propone una sua rilettura interattiva di un pezzo di storia della musica leggera italiana. Il pregio principale di quest'avventura è il suo stile narrativo: leggero, umoristico, scanzonato, con quell'aria da commedia nera che è difficile non apprezzare. Molti passaggi sono di per sé piacevoli a leggersi. Inoltre le citazioni precise dal testo (una per tutte, il contenuto del cestino) sono una ciliegina sulla torta. L'unica mancanza, forse, è un finale a sorpresa.

Per contro il difetto principale del gioco è, come spesso accade nelle trasposizioni da opere già esistenti, la seria carenza di interattività: per procedere il giocatore dovrebbe essere in grado di leggere la mente dell'autore (o la soluzione); tra comandi particolari e sequenze di azioni rigorose, risulta veramente ostico, se non impossibile, terminare l'avventura senza aiuto.

Tecnicamente e linguisticamente l'avventura è sufficientemente ben fatta, ma all'autore è sfuggito un bug estremamente fastidioso (che si manifesta dopo aver conversato e che può rendere impossibile la conclusione del gioco).

Voto: 5; Aggettivo: umoristica

Forma Mentis

Avventura a stanza singola di Paolo Maroncelli

In generale questa è un'avventura decisamente ben riuscita. L'ambientazione è ben resa, scritta in una prosa corretta e scorrevole (fatta eccezione, purtroppo, per l'introduzione), il gioco è ben implementato (senza grossi errori), la trama è appassionante. Inoltre il giocatore si trova di fronte ad una serie di enigmi che ruotano attorno all'utilizzo di "macchinari fantascientificamente tecnologici"; ebbene, questi enigmi sono molto ben pensati e contestualizzati, stimolanti, hanno giusta difficoltà, necessitano di una certa dose di ragionamento e infine sono gratificanti per il giocatore che li risolve.

Certo, si tratta di un avventura molto "di genere", sia per l'impostazione da FS classica ma anche per la presenza forte dei suddetti enigmi, ma questo non rappresenta certo un demerito; potrei arrivare a dire che nel panorama internazionale degli ultimi anni questo possa, ovviamente in un'opera ben costruita, costituire piuttosto un punto a favore.

C'è anche qualche difettuccio, certo, che lascia intuire una certa "immaturità" dell'avventura (nel senso che sembra rilasciata troppo presto); piccole sviste e un png un po' troppo limitato. Ma tutto sommato è poca cosa.

Voto: 9; Aggettivo: appagante

Final Selection

An Interactive Text Adventure

Final Selection è, nel suo distaccarsi dai canoni odierni della narrativa interattiva, un'opera estremamente peculiare e interessante.

Il punto più debole, volutamente debole, è rappresentato dalla trama esile, pressoché inesistente. Chiaramente essa è solo un pretesto per catapultare il giocatore in una stanza piena fino all'inverosimile di oggetti più o meno comuni (bilance, mappamondi, quadri e altro ancora) e sottoporgli una "diabolica" serie di enigmi e problemi.

Tali enigmi sono comunque interessanti, mai scontati, e anche se all'inizio sembrano esageratamente difficili, esplorando si possono sempre trovare e raccogliere indizi che possono aiutare a raggiungere la soluzione.

La realizzazione è pregevole, curata nel dettaglio, quasi esente da errori, con una scrittura semplice ma efficace e adatta al tipo di gioco. In particolare sottolineo la gestione degli appunti del personaggio, il comando "WHERE IS" e, soprattutto, la gestione della "focalizzazione" rispetto al punto della stanza dove si trova il personaggio (utilissimo, vista la densità di oggetti e di azioni da compiere).

Final Selection è dunque un'avventura quasi completamente priva di parte narrativa, e per questo potrà non piacere a tutti, ma la solidità degli enigmi e l'implementazione la rendono un prodotto di indubbia qualità.

Voto: 9; Aggettivo: cerebrale

INFORM: la gestione di un ascensore

di Vincenzo Scarpa

Un importante prerequisito: questi articoli sono riservati a coloro che già conoscono le basi della programmazione in Inform. Per i principianti di questo linguaggio consigliamo i seguenti siti:

- Guida a Inform per Principianti[1]
- Un manuale su Inform e Glulx[2]

La gestione di un ascensore è un altro "classico" delle avventure testuali. Ecco allora come funziona l'esempio da me proposto:

> **Ascensore**
> Un semplice esempio che gestisce un ascensore a quattro piani
> di Vincenzo Scarpa
>
> Versione 1 — Numero di serie 050704
> Inform v6.30 — Libreria 6/11 — Infit v2.5 S
>
> **Piano terra**
> Sei al piano terra.
>
> Puoi vedere delle porte scorrevoli qui.
>
> >

All'inizio abbiamo davanti a noi delle porte scorrevoli. Naturalmente, occorre aprirle per entrare poi nella cabina:

> >esamina le porte
> C'è un pulsante accanto alle porte.
>
> >premi pulsante
> Le porte si aprono silenziosamente.
>
> >entra
>
> **Cabina dell'ascensore**
> Sei nella cabina dell'ascensore.
>
> Puoi vedere un pannello di controllo con quattro tasti numerati da zero a tre.
>
> >

1 Vedi http://www.inform-italia.org/docs/gip
2 Vedi http://www.casalengo.it/vscarpa/inform/manuale

Eccoci dunque dentro la cabina del nostro ascensore. Se non soffrite di claustrofobia, potete allora decidere a quale piano andare premendo l'apposito pulsante:

>premi 2
Le porte si chiudono e la cabina si mette in movimento.
Improvvisamente, le porte si aprono di nuovo.

>esci
Non appena esci dall'ascensore, le porte scorrevoli si chiudono silenziosamente alle tue spalle.

Secondo piano
Sei al secondo piano.

Puoi vedere delle porte scorrevoli qui.

>

Una volta arrivati a destinazione, potete uscire dalla cabina quando volete, ricordandovi però di riaprire le porte se dovete usarla di nuovo:

>entra
Le porte sono ancora chiuse: credi forse di essere un fantasma?

>premi il pulsante
Le porte si aprono silenziosamente.

>entra

Cabina dell'ascensore
Sei nella cabina dell'ascensore.

Puoi vedere un pannello di controllo con quattro tasti numerati da zero a tre.

>

A questo punto, il suo utilizzo dovrebbe ormai essere abbastanza chiaro. Occorre però ancora notare che, se il giocatore preme il tasto corrispondente al piano su cui si trova l'ascensore, non accade nulla:

>premi 2
Non accade nulla.

>

come d'altra parte avviene in tutti gli ascensori che si rispettino. Per quanto riguarda invece il codice, occorre fare alcune considerazioni. Iniziamo dalla funzione Muovi_cabina:

```
[Muovi_cabina x y;
    Ascensore.piano_asc = x;
    if (parent(Porte_scorrevoli) ~= y) {
        move Porte_scorrevoli to y;
        move Pulsante_chiamata to y;
        print_ret "Le porte si chiudono e la cabina si mette in
            movimento. ^Improvvisamente, le porte si aprono di
            nuovo.";
    }
    else print_ret "Non accade nulla.";
];
```

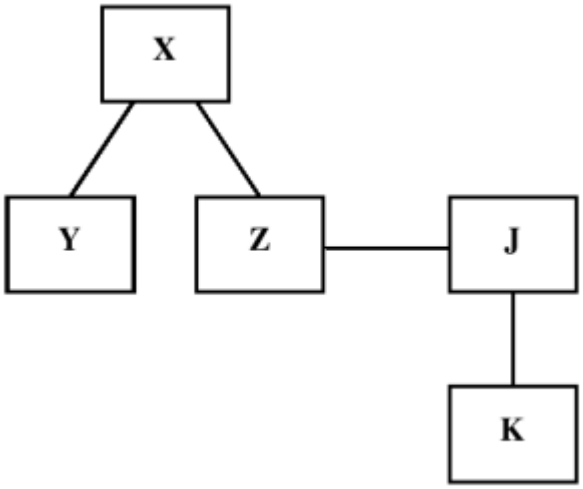
I parametri che devono esserle passate sono rispettivamente il numero del piano che l'ascensore deve raggiungere (x) e il nome dell'oggetto al quale corrisponde il piano stesso (y). È facile allora capire che una chiamata del tipo:

```
Button -> Pulsante_2 "terzo pulsante"
    with name 'terzo' 'due' '2//',
         before [;
            Push: Muovi_cabina(2, Piano_2);
            rtrue;
         ];
```

fa sì che, se il giocatore preme il pulsante due, viene effettuata una chiamata alla funzione `Muovi_cabina` con, come parametri, 2 per la variabile locale x e Piano_2 per la variabile locale y. A questo punto, viene dapprima assegnata alla variabile locale `piano_asc` il valore contenuto in x, utile all'oggetto Ascensore per sapere a quale stanza deve indirizzare il giocatore quando questi esce dalla cabina:

```
Object Ascensore "Cabina dell'ascensore"
    with description "Sei nella cabina dell'ascensore.",
         out_to [;
            < >;
         ],
         n_to [;
            give Porte_scorrevoli ~open;
            print "Non appena esci dall'ascensore, le porte
scorrevoli si
                chiudono silenziosamente alle tue spalle.^";
            if (self.piano_asc == 0) return Piano_terra;
            if (self.piano_asc == 1) return Piano_1;
            if (self.piano_asc == 2) return Piano_2;
            if (self.piano_asc == 3) return Piano_3;
         ],
         piano_asc 0;
```

Segue poi, una condizione che verifica se il nome dell'oggetto al quale appartiene `Porte_scorrevoli` è o meno uguale al nome dell'oggetto passato come parametro alla funzione stessa. Se lo è, allora non accade nulla (il giocatore ha cioè premuto il tasto corrispondente al piano stesso in cui si trova la cabina dell'ascensore), altrimenti vengono spostati gli oggetti `Porte_scorrevoli` e `Pulsante_chiamata` al relativo piano. Per capire meglio quanto ho appena detto, occorre rifarsi al concetto di albero; Inform, cioè, raggruppa gli oggetti con uno schema di questo tipo:



dove gli oggetti Y e Z appartengono all'oggetto X (e sono di conseguenza i child di X), mentre quest'ultimo è il parent di Y e Z. L'oggetto J è il sibling di Z (e non appartiene a X) ma anche parent di K e così via.

Sembrerebbe tutto a posto, ma c'è un'ultima considerazione da fare. Definendo il primo pulsante come segue:

```
Button -> Pulsante_1 "secondo pulsante"
    with name 'secondo' 'uno' '1//',
         before [;
            Push: Muovi_cabina(1, Piano_1);
                  rtrue;
         ];
```

tutto sembra essere a posto. Tuttavia, quando il giocatore preme il pulsante 1:

> **Cabina dell'ascensore**
> Sei nella cabina dell'ascensore.
>
> Puoi vedere un pannello di controllo con tre tasti numerati come zero, uno e due.
>
> >premi 1
> (il pannello di controllo)
> È fisso al suo posto.
>
> >

Inform assegna il nome 1 al pannello di controllo anziché all'oggetto `Pulsante_1`. Questo bug, che non so se imputare a Inform stesso o a Infit, è possibile evitarlo con un po' d'astuzia. Iniziamo a esaminare l'oggetto Pannello_controllo:

```
Object -> Pannello_controllo "pannello di controllo"
    with name 'pannello' 'controllo',
         description "Lo osservi attentamente, ma non noti nulla di
            speciale.",
         initial "Puoi vedere un pannello di controllo con quattro
tasti
            numerati da zero a tre.",
    has static;
```

La prima modifica che occorre fare è l'inserimento dell'azione Push (che in questo caso deve essere analoga a quella dell'oggetto `Pulsante_1`):

```
Object -> Pannello_controllo "pannello di controllo"
        with name 'pannello' 'controllo',
             description "Lo osservi attentamente, ma non noti nulla
di
                speciale.",
             initial "Puoi vedere un pannello di controllo con
quattro tasti
                numerati da zero a tre.",
             before [;
                Push: Muovi_cabina(1, Piano_1);
                      rtrue;
                ],
        has static;
```

Agendo in questo modo, l'ascensore si può spostare al primo piano:

> **Cabina dell'ascensore**
> Sei nella cabina dell'ascensore.
> Puoi vedere un pannello di controllo con quattro tasti numerati da zero a tre.
>
> >premi 1
> (il pannello di controllo)
>
> Le porte si chiudono e la cabina si mette in movimento.
> Improvvisamente, le porte si aprono di nuovo.
> >

Come potete però vedere, prima che l'ascensore si sposti viene anche stampato il nome dell'oggetto, che non è ovviamente il pannello di controllo. Occorre allora fare una seconda modifica:

```
Object -> Pannello_controllo "secondo pulsante"
    with name 'pannello' 'controllo',
         description "Lo osservi attentamente, ma non noti nulla di
            speciale.",
         initial "Puoi vedere un pannello di controllo con quattro
tasti
            numerati da zero a tre.",
```

```
        before [;
           Push: Muovi_cabina(1, Piano_1);
                 rtrue;
        ],
    has static;
```

Cambiando il nome dell'oggetto come secondo pulsante, abbiamo risolto anche il secondo problema:

> **Cabina dell'ascensore**
> Sei nella cabina dell'ascensore.
>
> Puoi vedere un pannello di controllo con quattro tasti numerati da zero a tre.
>
> >premi 1
> (il secondo pulsante)
> Le porte si chiudono e la cabina si mette in movimento.
> Improvvisamente, le porte si aprono di nuovo.
>
> >

Rimane tuttavia un altro problema; se il giocatore prova a premere il pannello di controllo:

> **Cabina dell'ascensore**
> Sei nella cabina dell'ascensore.
> Puoi vedere un pannello di controllo con quattro tasti numerati da zero a tre.
>
> >premi pannello
> Le porte si chiudono e la cabina si mette in movimento.
> Improvvisamente, le porte si aprono di nuovo.
>
> >

Inform continua a eseguire il codice relativo all'azione Push associata (la stessa, cioè, dell'oggetto `Pulsante_1`). Per risolvere questo terzo problema, bisogna ricorrere a una soluzione drastica:

```
Object -> Pannello_controllo "secondo pulsante"
    with name 'pannello' 'controllo',
         initial "Puoi vedere un pannello di controllo con quattro
tasti
            numerati da zero a tre.",
         parse_name [;
            if (NextWord() == 'pannello' or 'controllo') return 0;
         ],
         before [;
            Push: Muovi_cabina(1, Piano_1);
                  rtrue;
         ],
    has static;
```

Utilizzando la proprietà parse_name in questo modo, “obbligo” Inform a ignorare totalmente i nomi “pannello” e “controllo”:

> **Cabina dell’ascensore**
> Sei nella cabina dell’ascensore.
>
> Puoi vedere un pannello di controllo con quattro tasti numerati da zero a tre.
>
> >premi il pannello
> Non vedi nulla del genere.
>
> >premi 1
> (il secondo pulsante)
> Le porte si chiudono e la cabina si mette in movimento.
> Improvvisamente, le porte si aprono di nuovo.
>
> >esci
> Non appena esci dall’ascensore, le porte scorrevoli si chiudono silenziosamente alle tue spalle.
>
> **Primo piano**
> Sei al primo piano.
>
> Puoi vedere delle porte scorrevoli qui.
>
> >

Certo, la risposta di sistema non è delle migliori (il pannello di controllo infatti esiste, nonostante Inform affermi il contrario), ma d’altra parte non si può sempre avere tutto.

Segue ora il listato completo dell’esercizio[3]:

```
Constant Story "Ascensore";
Constant Headline "^Un semplice esempio che gestisce un ascensore
a quattro piani^di Vincenzo Scarpa^^";

Include "Parser";
Include "VerbLib";
Include "replace";

!-----------------------------------------------------------------------!
Class Button
 with name 'bottone' 'pulsante' 'tasto',
 has static scenery concealed;

Class Floor
 with s_to [;
        if (Porte_scorrevoli has open) {
            PlayerTo(Ascensore);
            rtrue;
```

3 Il listato completo dell’esercizio (tratto da Un manuale su Inform e Glulx potete anche trovarlo nel file Capitolo4.zip(esercizio 4.7.inf).
Vedi http://www.casalengo.it/vscarpa/inform/manuale

```
        }
        else print_ret "Le porte sono ancora chiuse: credi forse
            di essere un fantasma?";
      ];

!-------------------------------------------------------------------!
Floor  Piano_terra "Piano terra"
  with description "Sei al piano terra.";

Object -> Porte_scorrevoli "porte scorrevoli"
  with name 'porta' 'porte' 'scorrevole' 'scorrevoli',
       description "C'@`e un pulsante accanto alle porte.",
       before [;
           Open: "Non puoi aprirle.";
       ],
       react_before [;
            Enter: if ((noun == nothing) && (self has open)) {
                        PlayerTo(Ascensore);
                        rtrue;
                   }
                   else print_ret "Le porte sono ancora chiuse:
                            credi forse di essere un fantasma?";
       ],
       door_to Ascensore, door_dir s_to,
  has static door pluralname female;

Object -> Pulsante_chiamata "pulsante di chiamata"
  with name 'pulsante' 'bottone' 'tasto' 'chiamata',
       before
       [; Push: if (Porte_scorrevoli hasnt open) {
                    give Porte_scorrevoli open;
                    "Le porte si aprono silenziosamente.";
                }
                "Le porte sono gi@`a aperte.";
       ],
   has static scenery concealed;

!------------------------------------------------------------------- !
Object Ascensore "Cabina dell'ascensore"
  with description "Sei nella cabina dell'ascensore.",
       out_to [;
          < >;
       ],
       n_to [;
          give Porte_scorrevoli ~open;
          print "Non appena esci dall'ascensore, le porte scorrevoli
                  si chiudono silenziosamente alle tue spalle.^";
          if (self.piano_asc == 0) return Piano_terra;
          if (self.piano_asc == 1) return Piano_1;
          if (self.piano_asc == 2) return Piano_2;
          if (self.piano_asc == 3) return Piano_3;
       ],
       piano_asc 0;

Object -> Pannello_controllo "secondo pulsante"
  with name 'pannello' 'controllo',
       initial "Puoi vedere un pannello di controllo con quattro
            tasti numerati da zero a tre.",
       parse_name [;
           if (NextWord() == 'pannello' or 'controllo') return 0;
       ],
```

```
      before
      [;
         Push: Muovi_cabina(1, Piano_1);
               rtrue;
      ],
  has static;

Button -> Pulsante_0 "primo pulsante"
  with name 'primo' 'zero' 'terra' '0//',
      before
      [;
         Push:   Muovi_cabina(0, Piano_terra);
                 rtrue;
      ];

Button -> Pulsante_1 "secondo pulsante"
  with name 'secondo' 'uno' '1//',
      before
      [;
         Push:   Muovi_cabina(1, Piano_1);
                 rtrue;
      ];

Button -> Pulsante_2 "terzo pulsante"
  with name 'terzo' 'due' '2//',
      before
      [;
         Push:   Muovi_cabina(2, Piano_2);
                 rtrue;
      ];

Button -> Pulsante_3 "quarto pulsante"
  with name 'quarto' 'tre' '3//',
      before
      [;
         Push:   Muovi_cabina(3, Piano_3);
                 rtrue;
      ];

! ---------------------------- !
Floor  Piano_1 "Primo piano"
  with description "Sei al primo piano.";

Floor  Piano_2 "Secondo piano"
  with description "Sei al secondo piano.";

Floor  Piano_3 "Terzo piano"
  with description "Sei al terzo piano.";

! ---------------------------- !
[ Initialise;
    location = Piano_terra;
    give player light;
    give Porte_scorrevoli ~open;
];

[Muovi_cabina x y;
    Ascensore.piano_asc = x;
    if (parent(Porte_scorrevoli) ~= y) {
        move Porte_scorrevoli to y;
        move Pulsante_chiamata to y;
```

```
        print_ret "Le porte si chiudono e la cabina si mette in
movimento.
                  ^Improvvisamente, le porte si aprono di nuovo.";
        }
    else print_ret "Non accade nulla.";
];

! ———————————————————— !
Include "ItalianG";

Extend 'entra' first * -> Enter;
```