Interview with Enrico Colombini

(http://www.spagmag.org/archives/backissues/spag51.html)

In 1982 Enrico Colombini wrote and sold the first text adventure in Italian, *Avventura nel Castello* (*Castle Adventure*), for the Apple][.
In 1985 he published two books about writing text adventures which were hugely successful in popularizing the genre. Along with the books came a tool written in BASIC, named Modulo BASE (BASIC Module), and more games: *L'Astronave Condannata* (*The Doomed Spaceship*) and *L'Anello di Lucrezia Borgia* (*Lucrezia Borgia's Ring*), for five different microcomputers. Later, in 1988, he revised one of his books and made Modulo BASE more powerful.
A new game for MS-DOS, *L'Apprendista Stregone* (*The Sorcerer's Apprentice*), showed off the new features. He developed another small example game, *Il Drago delle Caverne* (*Cave-Dragon*), for a course in BASIC in 1989.
These programs were updated and refined until 1999, when they were released as freeware along with the book *Avventure per MS-DOS* (*Adventures for MS-DOS*). In 2000 he released another tool, Idra (Hydra), for writing Choose Your Own Adventure-style games in JavaScript. In short, he's a living legend, but don't mention that to him or he will scold you.

TDF: Enrico, I wrote a short introduction but it isn't enough. Could you tell the readers a little about yourself?
ENRICO: As a teenager, I guess I could be classified as the quintessential nerd: from Meccano to electronics, I was in full control of technology... and of little else. I read a lot, though, so I had the seeds of my redemption in me.
Now I'm married, we have a son and I just lead a quiet life: I've always been a quiet type, but I never ceased being a nonconformist and an idealist, whatever the price (and it can be quite high, at times).
I'm sitting at my desk, but my mind is still adventuring out there, as always. What never ceases to amaze me is that, of the many things I designed in electronics, software and publishing, only adventure games survive in the collective memory: they probably happened to be born at the right time (I suppose there's a humbleness lesson in that).

TDF: People still remember you today mainly because of *Avventura nel castello*, even outside of Italian IF circles. Let's talk about it. How did you decide to write your first text adventure? I'd like to hear about the design phase, too. You credit your wife, Chiara, as co-author... Lastly, why did you choose a (Scottish) castle? Can you recall any source of inspiration?
ENRICO: As I wrote (in Italian) on my site, the inspiration came from *Adventure*, i.e. *The Colossal Cave*, more exactly from the 350-point 1980 version we played on my brand new (well, new for me, but actually quite used) Apple][.
We (Chiara, my friends and I) played it a lot: it was fascinating and, above all, it was something utterly new. However, the "no save" feature was irritating; I couldn't get around it by copying the save file between game sessions, because the disk was protected, so I decided to look into it. I wrote a primitive disk analysis utility ("DAN"), found out the protection system (it was just a different sector coding scheme), removed it and added the save/restore commands. I also had a look at the code, of course, but I wasn't particularly impressed with what I saw.
More or less at this point we said "This idea is great, why don't we do something like it in Italian? We can do it better". So the design phase started. Now, "design phase" is a rather pompous expression: in fact, we just thought about it from time to time, came up with ideas and discussed them. That's my standard way of designing when I'm not under pressure, and it usually works well because my subconscious mind does all the work: I just have to be patient and wait for the results. Chiara had an important role in the design: she discussed, corrected, often rejected my ideas, besides of course contributing with original ideas of her own (we can't really remember which one of us had which idea).
As soon as we had a minimal map and some puzzles in place, I started coding in the evenings. I was the coder, because Chiara wasn't really interested in programming on personal computers (she worked in assembly language on microprocessors at the time).
The first program was rather primitive, sort of many big IF...THEN...ELSE with just PRINTs for output, but it worked and it was promising enough to push us forward. We went on for a couple of months adding locations and puzzles, until I hit the memory barrier: the texts filled the 48 kB (actually, rather less in practice) available on my Apple][.
So I learned to effectively use the floppy disks and moved all texts to an indexed file to free RAM, then redesigned the program to be more table-driven and, generally speaking, saner. As a side-effect of the program's growth, I encountered two new problems: the extreme slowness of BASIC's garbage collection (it could hang the machine for many minutes at unpredictable times) and the relatively long time it took to look for a word (parsed from user input) in the dictionary. So I did a good thing and a bad thing.
The good thing was studying the insides of Applesoft BASIC and reading around a lot, which lead to a simple but very effective way of partitioning strings in two areas: "collectable" and "non-collectable". As the vast majority of the strings were constant, this approach did away with the garbage collection problem completely.

Of the bad thing I did I'm still a bit ashamed: having had no exposure to computer science and algorithms, I was naively doing a linear search. A simple binary search would have solved the problem with minimal effort but, being unaware of it at the time, I did what I knew how to do: I recoded the linear search in assembly language. It worked, of course, but it's still a dark stain on my (otherwise almost decently clean) programming history scroll.

The game progressed and evolved, design and implementation going hand-in-hand, and a few more months went by while new ideas and puzzles were added. We certanly had many sources of inspiration, first of all the large number of games I played: I'm sure some ideas were stol... er, inspired by early adventure games, but it's difficult to remember what came from what (for example, somebody pointed out the similarity of the plane-crashing introduction with *Cranstor Manor*, which I vaguely remember playing, but I've no idea when I played it).

The "graphic" fall from the plane came from *Adventure*'s chasm, of course, while the maze was patterned after that of Umberto Eco's *The Name of the Rose*; being tired of senseless mapping, I had long been thinking about a non-conventional maze and the library in that book gave me the right idea. Speaking of non-conventionality, the whole game was designed to be a challenge to "standard" ways of problem-solving in games (such as: "go and kill 'em all") and we're rather proud of the results.

Testing played a very important role too: during many months, I looked at friends playing and took note of everything they wrote, however strange or unexpected (especially if strange or unexpected!) then added most of it to the game. I think this should actually be considered part of the design: in fact we were using other people's minds in addition to our own.

About the choice of a Scottish castle background, I really have no idea: perhaps it was the influence of Stevenson's books, or Poe's, or some gothic novel... but when (many years later) we actually went to visit the wonderful country of Scotland, we were happy to realize that we'd been rather accurate in our settings.

TDF: You managed to sell *Avventura nel castello*, you're one of the few Italian authors who made some money with this kind of software. I know you began by selling it yourself (and this reminds me of Roberta & Ken Williams). What problems did you face? Was it hard to find a software house, later? Could you inform readers about the Italian software market in the 80s? Your game had a long life: three commercial editions, and the third was for MS-DOS. At the end of it, did you get rich?

ENRICO: At the beginning, it was just a favour a couple of friends did me: they had this computer shop (possibly the first one in my town) where we exchanged knowledge and tools, and they sold... well, they sold 12 copies around the end of 1982, according to my records.

Next year, the publisher I had begun writing technical articles for (Gruppo Editoriale Jackson) started a software marketing division (J.Soft), so I was able to propose them a couple of games, including *Avventura nel castello* (which had just won 1st prize at the first Italian computer game contest, *Computer Play 83*).

With the support of their computer magazines, they sold about 600 copies and, more importantly, managed to reach an agreement with Apple Computer Italy to have the games bundled with the new Apple //c, so a lot of people was able to play them. The other two games (my board game *Melopoli* and a friend's well-designed strategy game, *Signori della Galassia* [*Lords of the Galaxy*]) made a less-lasting impact, though.

The software business didn't prove to be a stellar success, due to hostile conditions in Italy, i.e. few computers around, lack of technical culture, and widespread piracy (often done in full daylight by the resellers themselves and sometimes tolerated if not encouraged by some hardware vendors - Commodore comes to mind).

I still hoped to be able to live by designing and selling games, but it proved to be impossibile. In the meantime, other countries were starting a real computer game industry; I even made a half-hearted attempt to contact a French publisher, but to no avail (I've never been good at marketing).

The nail in my ambition's coffin came when Apple declared it wanted no games for the Macintosh (I was developing one at the time). I continued to earn my bread (and butter too) with computer courses and encyclopedias; as for the games, alas, I had to content myself with playing them, usually on the IBM-compatible PC that was fast becoming the new standard after Apple's marketing suicide (but this is another story).

Anyway, I wanted people to be able to play my games, so I made an MS-DOS version. It was a complete redesign, based on a specialized language I had been designing, and it taught me a lot. For example, I learned that using a specialized language to write IF is not necessarily a good idea, at least when the author is also a decent programmer (later, I got much better results by using a hybrid approach).

Ah, the MS-DOS version sold about 100 copies through Hi-Tech (for which I was writing on a magazine for Apple users); a much, much larger number of copies was undoubtely pirated, but at that point I cared more for diffusion than for income.

About getting rich... well, I made millions! Unfortunately I got the timing wrong: the Euro wasn't there yet, so they were million liras, to be scaled by about a 2000:1 factor. But, technically speaking, text adventures made me a millionaire.

You mention Roberta & Ken Williams: they were undoubtely pioneers, and I enjoyed some of their early graphic adventures, but the IF authors I loved were in the Infocom camp, Steve Meretzky above all but many others also (by the way, *Enchanter* gave me the basic idea for *L'Apprendista Stregone*). Most of their adventures had good stories, good prose, good ideas and good care of detail. It was good, while it lasted.

TDF: After your first game, you hit bookstores with two books, *Avventure* (*Adventures*) and *Scrivere un gioco d'avventura* (*Writing a Text Adventure Game*). The first included an audio tape or diskette with three programs: two example games (*L'Astronave Condannata* and *L'Anello di Lucrezia Borgia*) and the tool you used to write them (*Modulo BASE*). It was quite a plain tool and you chose BASIC. Did you have a model in mind, i.e. Scott Adams' adventures? Did you look for a wider audience? Now I can say you had a deep impact on (nearly all) Italian IF developers: everybody strove hard to add features, and someone (i.e. Bonaventura Di Bello) even sold games which had your tool as backbone.

ENRICO: The only model I had in mind was the engine of *Avventura nel castello*; Scott Adams' interpreters were designed to save every bit of RAM in really small machines, while I worked in comparative luxury and had no such need for data compression. However, my engine was too complex for beginners to handle and for me to explain in a decent way, so I made it simpler by cutting off features; for example, objects couldn't have states anymore (e.g. a bone that could be whole or broken), but had instead to be replaced by a different object (a whole bone, a broken bone).

To my amazement, the 'reduced' engine proved in some respects better than the original, and certainly easier to use. Redesigning after a bit of experience can yield better results, especially when the aim is to distill and preserve the essence, discarding redundant junk.

Later, the second version of Modulo BASE reintroduced some useful concepts, such as indexed files on disk and a few (often-used) commands embedded in messages, reaching a good balance (as I see it) between power, flexibility and ease of use.

Most people, however, were contented with the capabilities of the first version. It was simple code (at places rather primitive), but I had thought about the underlying concepts for years, so it was an useful tool. Others, such as Bonaventura Di Bello you mentioned, exploited it to the core and beyond: one Sunday morning he called me (waking me up) asking, if I remember well, how to ease some of the program's intrinsic limits (for example, the maximum number of different words, which wasn't as simple as it sounds). Version 2 didn't exist yet, so I gave him some suggestions that he put it to good use: he released a string of acclaimed games for the newsstand, some of which used my tool. At last, that's how I remember it; I hope I'm not confusing him with another power user... you know, old age and all that... I'm sure about the phone call, though. 🙂

About Modulo BASE, the program had a simple 2-word parser but I still think, after all these years, that a more complete and 'realistic' parser and world model don't necessarily imply more enjoyable games, even if they would certainly be more interesting from an AI (artificial intelligence) perspective and in view of (always-almost-here-but-never-quite) real speech recognition. I feel that's very easy to fall in love with technology and forget playability.

TDF: *L'Apprendista Stregone* is your favourite, and I like it very much too. It's an ambitious work though you claim you wrote it in a fortnight. Did anybody help you? You had it well planned in advance, right? I guess you had a deadline you couldn't miss: do you work better under pressure? It has an iffy vibe I can spot also in your previous example games: you paid particular attention to the story. What about characters (human or not) and setting? Where did you find the magic system idea?

Enrico: Chiara contributed, as usual; her classical knowledge was very useful, even if I too know, er, should know, a tiny bit of Latin (but, sadly, no Greek). Choosing appropriate names for the spells was an amusing exercise.

The claim that it was written in a fortnight is true... the trick is in the "written". The design took much longer, as usual: we let ideas float and slowly take form, not unlike crystals (with or without flaws, it's for the players to decide).

The forest, for example, came from a trip to Saltzburg: we admired it while comfortably traveling by coach, and wondered about it. On the other hand, we carefully avoided putting in the game the incongruous gnome-miner that sat in the famed salt mines of that beautiful town. I suppose he's there for American tourists to admire, or at least I hope so. But I digress.

The not-to-be-missed deadline suddenly appeared when my publisher asked me to add something for the new edition. I really cared to see *L'Apprendista Stregone* published, so I put in long hours for a couple of weeks.

I don't know if I actually work better under pressure: the only sure thing is that I work more. 😉

The magic words idea was unashamedly lifted from Infocom's *Enchanter*, but left for the player to discover, while the narrative approach was of course a design choice: I wanted people to enjoy the story and the settings without having to draw complex maps or to solve fiendish puzzles. The challenges I chose to put in were mostly of the 'lateral thinking' type and I'm quite satisfied of the result, even if I'd have liked some extra time for refinements (but then I'd surely have asked for more).

For the main characters, the old mage Artemio and his young apprentice (the player), I can think of no definite source; it's a common theme, after all. But we had fun placing ourselves in the game, even if there's little resemblance with the originals: for example, Chiara does not do fortune telling but writes programs... uhm, actually, now that I think of it, unpredictabilty plays a big part in both jobs. As for myself and the illusionist... after all, games are a sort of illusion, aren't they? By the way, my math professor scolded me for dropping out of the University, so I made her do the same in the story, disguised as an old wizard.

Lastly, I liked the name I chose for 'my' character, so I adopted "Erix" as my signature on the Net, in those misty pre-Web times, and I'm still happily using it.

TDF: With Idra you steered the wheel towards CYOAs. Could you briefly introduce this tool to readers? Why did you develop it? You didn't release any CYOA, indeed the two examples included in the package are by other authors. I can assume you like this form, and maybe you read similar books in the past or shared this liking with friends.

ENRICO: Enrico: I'm not sure about Idra being exactly a Choose Your Own Adventure tool: it can certainly be used to that effect (in fact, it's the easier way to use it) but, in the hands of a good programmer, it could be quite flexible. I wished to write a complex adventure to show off its capabilities, but that implied complex planning... and time... and resolve... in short, I never got around to do it.

Idra was born from a question: does most people avoid text adventures because they have no wish to read, or because they have no wish to write? So I wrote a simple HTML/Javascript tool that, in a sense, emulated point-and-click graphic adventures, but with no graphics.

The results, I should say, were inconclusive: yes, more people accepted to play the games (as compared to people willing to play text adventures where writing was required), but on the other hand they were easily bored by long texts (I should say, by non-infinitesimal texts). So, in the end, it must be a combination of factors (like having grown up with more books than TV) that controls interest in the written page, be it a book or a game.

By the way, the first time I encountered a Choose Your Own Answer book, it wasn't a game at all. I still have it: it's an "Introduction to genetics" from the *Tutor* series, 1967 (a few years before the WWW craze...). It posed a question and redirected the reader to another page according to the answer, to explain the mistake or to reinforce the learning. It was well designed. Much later I bought game-books and found their design rather disappointing and primitive... even if I played them anyway. 🙂

Back to Idra, I recently started another project: a full DHTML/CSS engine for writing text adventures, with a few interesting twists and a more "pseudo-graphical" approach. I learned DOM and CSS, wrestled with compatibility problems for a couple of months, wrote a library, proved beyond doubt that it was feasible... and then I abandoned it, as usual.

It goes almost always like this: once I have gained the knowledge I'm interested in, an actual application is only wasted work... unless, of course, somebody is really interested in it (or, God forbid, even paying for it).

I have my physical, digital and mental drawers full of such experiments, in more or less advanced completion stages, many of them exploring text adventures approaches and possible innovations in a variety of languages (Basic, C, Prolog, Perl, Dylan, Bash script, Lua...) or just as thought exercises. Unfortunately, the road from basic proof to finished product is long and boring, especially for a nitpicking type like myself. If sometimes a complete work such as Idra sees the light, I'm afraid it's a sort of accident. But, in this imperfect world, accidents happen.

TDF: Tell us more about your experiments with programming languages. It's no secret you think Lua very suitable for developing IF or an IF tool. What's the matter with other languages?

ENRICO: Enrico: Well, after *Avventura nel castello* I had the illusion that you could build a world model: a perfect, logical representation of the world. Those were the years the object-oriented concept was coming into fashion and I was an early convert, so I tried building an object model in C, or maybe it was objective-C (using the Manx compiler I just bought for my Macintosh) complete with objects (in the adventure sense), attributes and so on.

My code worked, but I soon realized many things: first, that building a complete (or even passably realistic) library was impossible because of the combinatorial explosion and the refusal of the real world to be rigidly classified; second, that the more complex the library, the more rigidly and predictably the game interface behaved; third, and most important, that there was no relation at all between world-model complexity and player's fun.

For example, if you _can_ look under an object and behind an object, you _must_ look under all objects in the game and behind all objects in the game, and that's extremely boring and no fun at all.

Around the same time, I also experimented in a different direction: I tried Prolog, a declarative language (you don't write ordered statements: you write assertions, or whatever the correct term is, and the system takes care of the rest). Expert systems were supposed to be very powerful, so I was interested in trying out something of that sort.

I don't know how well it would have played, but I know that I almost immediately exhausted the 640 kbytes of my M24 PC when I tried to port Avventura nel castello to Borland's Turbo Prolog. Too bad.

But I digress; let me return to the point: in my opinion, to make an enjoyable adventure game you don't have to accurately simulate a world, or to simulate a world at all: you just have to answer the player's commands, answer them in a sensible way, and avoid being repetitive or too predictable. It doesn't matter if the answer is logical or not, it just has to be appropriate.

From an implementation standpoint, that translates into a different approach (with respect to the world model): instead of trying to have a logical understanding of what the player says, which would imply a very large number of unspoken assumptions, you can just do a simple (sometimes not so simple) pattern matching. It's the way a child learns a language, after all, and it's probably the way our brain usually works: pattern recognition.

So I don't really care what the words actually mean, but I care that certains expressions, or words, imply certain expectations, so I can give them a meaningful answer.

In practice, that means doing away with almost all logical and syntactical classifications, and centering the data around what the player could actually say and expect. For example, given a button and a carrot, I could expect the player to say something like:

- push the button
- eat the carrot
- push the button with the carrot

but, of course, that is far from exhausting all that the player could say, so I'll introduce wildcards:

- push $1
- eat $1
- push $1 with $2
- push button with $1
- push $1 with carrot
- $1 button
- $1 carrot
- $1 $2 with $3

Note that the system has no idea (at least at this stage) what the words actually mean, for example that "push" is a verb or thet "carrot" and "button" are objects: the answers are tailored to the patterns. Given adequate precedence rules, such a system could be perceived as much more 'real-life' than a full world-model library... at least for a given game.

Two drawbacks can be easily seen: this system is labour-intensive and it does not lead to a general-purpose, reusable library. But that's exactly my point: a good adventure game is a work of art, not an industrial product. If you build a reusable library, most of your game will be predictable; in other terms, you'd have done good engineering... and bad art.

I should say that I over-simplified the matter for space reasons, and also made black and white of things that actually have infinite gray shades. I also perhaps exaggerated a bit the difference between fixed world model and pure pattern matching, just to illustrate the different underlying concepts, even if practical implementations (mine included) may borrow from both schools.

Speaking of actual implementations, I'd like to do away with dictionary, map and such useless things. My ideal adventure definition text file would only contain patterns, that is complete or wildcarded sentences, and answers: all the rest (dictionary tables, map, etc.) should be done by the program. That's a bit utopic, I know, but I think it's a target worth aiming at.

Anyway, working in that direction I soon realized that statically-typed languages (as much as I like C for other uses) are a real hindrance in this field. Dynamically-typed languages are much better suited to such a task; for example they can create 'properties' on the spot, they allow you to mix code and data, or to pass any type of data around, even one you've just created.

All that, you'll say, amounts to asking for trouble, and you'd be correct. Unless, that is, all this wild freedom is framed in a well-tought-of framework. My Idra is a very limited example of such a framework, but I think it proves the point: a fully dynamic language (Javascript, in this case) can be tamed and put to good use to build a flexible tool, where the framework gives a clear direction and control but the 'dynamic' part leaves endless possibilities open. (well, Idra is not exactly a text adventure interpreter, but the principle is the same).

By the way (and on the same line) I hate compilers: with the computing power we have today on our desks, there's really no need at all for such a nuisance. All table-building, cross-referencing and analysis can be done in a very short time at program startup, without the player even noticing.

So, if you're still awake after all these digressions, I can at last answer your question: why do I like Lua? It's not a religion (after all, I used many languages), it's just a tool I find very well suited for this task because, among other things: it's very compact, simple but powerful, totally dynamic (in practice it's very similar to Scheme - a Lisp dialect - but in a more convenient form), has no syntax quirks (Python comes to mind...), can be easily integrated with C and, last but not least, it's free and wonderfully supported. Being also a sort of toolkit for creating new languages, it's very convenient when designing specialized interpreters such as those that power adventure games.

TDF: Ok, you don't like libraries very much, but sometimes you can change or rewrite them. What do you think about specific programming languages for IF, such as Inform, Tads, Hugo? Did you try their latest incarnations? C'mon, you can't dismiss them too easily... most people write IF this way. You said games would be predictable if everybody used libraries; there's some truth in this statement but my question is: are authors up to the challenge of writing good IF? Remember what the Z-Machine gave you back in the 80s... and I think we still get some good games now!

ENRICO: I'll start by confessing my greatest sin: I've not played most of contemporary IF... at least yet. The main reason is that I find cooperative play much more fun than playing alone, but none of my friends is interested in IF anymore.

About tools and languages, I browsed Inform's manual and found it way too formal and static (that's my impression, it could be wrong). It seems to me its main value is in the library, but (as I said before) libraries are a double-edged sword: on one hand they make the author's work a lot easier; on the other hand they make the player's life a lot more repetitive, especially when they are widely reused. Part of this limitation is implied in the established interface used for IF, of course: most of player interaction has assumed 'canonical' forms and it's therefore highly predictable (no, I've not a better idea to propose, not a coherent one anyway).

I've not tried out the other tools you mentioned. I guess they're a wise choice for 'pure' authors who have no interest in getting their hands dirty with technical problems, but I'm like those old painters who preferred to grind their own colors. I'm not lacking programming experience and I like challenges, so building a framework (especially on unexplored paths) is for me a significant part of the fun. Besides, using an existing tool means traveling along an existing road, and that's not what I'm mostly interested in.

TDF: If I were nasty I'd ask you to name the few titles of contemporary IF you played, but I'm not going to compel you. 🙂 What about your favourite pieces? You said you appreciate Meretzky's work. If you could steal an idea from his games (or any game, if you prefer), what would it be?

ENRICO: Having played a very small sample of contemporary IF (chosen more or less at random), it'd be pretty unfair of me to speak about it: I'd inevitably tend to judge it against my own (possibly inflated) expectations, not against the background of other works. So I'll pass on this question.

I've played a number of graphics adventures recently, but that's probably not the wise thing to say in this company... anyway, there's always something to learn about narrative techniques and tricks, and about interfaces, even when the player doesn't have to communicate in writing. I think that IF could benefit from radical interface innovations.

I appreciated Meretzky both for his witty style and for his narrative surprises, notably *Planetfall*'s Floyd character (the #1 idea I would have liked to steal!), even if sections of his games could sometimes be rather empty and dull. But of course he wasn't the only author I liked, even if his name stuck in my memory.

Michael Berlyn's *Suspended*, for example, was a masterpiece of innovation with its six robots acting as individual senses of the hibernated player. Douglas Adams's *Hitchhiker's Guide to the Galaxy* was extremely funny (even if somewhat difficult and at times unfair towards the player), also thanks to the cooperation of... oh, Steve Meretsky again.

I've recently been re-playing *Enchanter* (by Marc Blank and David Lebling) just to see if I was fondly remembering those games just because they happened to be published in the golden age. It seems to me that I wasn't: *Enchanter* is as fun to read and play now as it whas then.

Modern authors, of course, face much tougher problems than historical pioneers did, the foremost one being that they have to work on established tracks, so it's quite difficult to come up with something really new. Getting out of the tracks gets harder and harder as time gets by, so those who can attain it are really talented.

In a sense, having to work within an established framework, it's probably easier to tell an original story than to devise original puzzles, so readers who prefer a more novel-oriented approach are easier to satisfy than those searching for mind challenges (I place myself somewhat in the middle).

As I said, I think that radical interface innovation could do wonders to rejuvenate IF. I know about Inform's new 'natural language' design: it's certainly an interesting approach, even if it vaguely reminds me of Apple's Hypertalk and Applescript 'easier languages' and that sends shivers down my back. But I could be greatly mistaken: Inform 7 is young, so it's only fair to wait and see what comes out of it. Even if it should not pay out, failed experiments are always better than stagnation.

In the meantime, I'm idly thinking along different lines. For example, a 1-word parser (hugely dependent on context) that mimicks what we actually often say in a real-world conversation between friends. Think about it: how often do you speak complete sentences? Or, along a different line, hybrid systems combining written text with clicking and dragging. I've seen some experiments, but nothing remarkable yet (I may have looked in the wrong places, though). I was working on one such experiment a few months ago, but then I found a new job that took up most of my time.

TDF: You haven't released anything for quite some time now. Are you writing a game? You know Italian players would like to see a new piece of IF by Colombini... You said building a framework is fun: are you hinting at something? Any work in progress?

ENRICO: I knew this question was coming. 🙂

Now, I hate to disappoint my readers, but my mind works in a strange way: when I have work to do I'm almost totally absorbed by it (let's say in "on" state) and when I haven't I'm "off" and it's difficult for me to find enough energy to bring a full project to completion (as opposed to a simple study, or to sketching down ideas).

So I'm a bit like Sherlock Holmes, with the difference that I use computer games in place of cocaine to keep my mind from rusting: it's cheaper, more fun (I suppose, I'm not interested in trying it out) and definitely safer.

In fact, most of my games were written as work under publisher's deadlines (or at least in view of publication), with the notable exception of *Avventura nel castello*. But that one, being an exploration of unknown territory, was exciting enough in itself. As the probability of somebody paying me to write text adventures in the XXI century is low enough, I fear some time will have to pass before I could publish new IF (though I want to leave the door open to that possibility).

By the way, being unable to stand repetitive jobs is the main reason why I am sometimes unemployed, especially in this country where innovation and creativity are four-letter words. But I've just found an interesting job that promises to keep me in "on" state for some time, so maybe (I repeat: maybe) there's hope for an unpredictable future.

TDF: Last but not least... How do you foresee the future of the Italian IF community? Is IF as we know it and like it likely to survive as hobby or is it bound to extinction, without a real market? Will it move on the world wide web or on cell phones?

ENRICO: My crystal ball is temporarily out of order. However, I'm pleasantly surprised by the vitality of the Italian IF community relative to its size: most of its members give no sign of tiredness, even if creativity seems to come in bursts (as is to be expected); I also note some interest in trying out new concepts, both in literary form and in technology. The main problem of this community is probably its small size, even if sometimes a new face appears.

I guess (and hope) this community will survive, despite the full-color lure of graphics; of course the relatively small number of readers (with respect to the English-language IF community) means less feedback and thus less incentive for writers. It's impossible to overestimate the importance of feedback: somebody telling you "Hey, I really liked your work" can really make the difference between going ahead at full steam and abandoning one's project.

As for IF in general, I think it'll survive too, probably in the same niche it occupies today, even if 'normal' people don't seem very interested in reading anymore. On the other hand, the Web is a fascinating field to explore (I find it odd its potentiality has mostly been left unexplored by IF authors and programmers) and could definitely give birth to something of value. I'm less sure about cell phones, as they are traditionally connected to hurried, superficial use.

A technology that could revitalize IF is of course voice recognition, which is on the verge of arriving... as has been for about 20 years. Maybe IF could give it some help; I'm even more surprised to find very little experimentation here: I've been thinking about audio-controlled games since the '80s and IF was definitely on my list. You don't need full voice recognition to make an adventure game work; on the contrary, with a bit of imagination you could use recognition errors to your advantage.

Maybe one of the problems is the rigid separation between artists and authors, due to technology's ever-increasing complexity: mastering both creativity and technique, as an artist should, has become quite uncommon. It's difficult to create a work of art in cooperation, or to constrain creativity within a pre-built tool, but it's not impossible. So... I still hope to live to see a new creative age in IF.

TDF: OK, that's all. Thank you very much for your time, Enrico.

April 2008, torredifuoco