inside erin

The AIF newsletter

Volume 3, Number 3 March, 2007

Letter from the Editor

I am quite pleased, actually, to see February of 2007 go into the history books. It was a difficult month for me. I started it by working about a dozen straight 12-hour days finishing a project at work. During those days, the outdoor temperature didn’t rise above about 10°F and the insufficiency of insulation in my house was very evident. In one room, in fact, condensation formed inside electrical outlets which then froze (there was ice coming out of the outlets!), melted and stained the ceiling in the room below. Meanwhile, the football team in my city managed to win their way into the Super Bowl only to lose the game. My kids were diagnosed with strep throat and had to take yukky antibiotics for a week and a half, during which time they were miserable, didn’t sleep well, and were kept home

Gripe continued on page 5

Interview with Mike Roberts by A. Bomire

This month, it is my tremendous pleasure to present an interview with Michael J. Roberts, author of the Text Adventure Development System - better known as TADS. TADS was first released in 1993, and has been going strong ever since. In the fall of 2006, Mike Roberts released TADS version 3.

AB: How long did you work on TADS 3 before officially releasing it?

MR: I'm not sure exactly how long that was - several years, it's safe to say. I spent a while working on the core system components - the VM *[Virtual Machine]* and the compiler - before even starting to think about the IF side of it. But the library development was definitely the bulk of the work; that went on for several years before the first official release.

AB: Were you the only one developing TADS 3, or were there others who collaborated with you?

MR: It was a very collaborative project. Which is a big part of why it took a long time - and I don't mean that in a bad way; it's not a complaint by any means. It's not that the many contributors slowed things down; it's more that having all that additional brainpower both opened up the scope of what we were trying to accomplish, and caught a lot of my false starts before they became intractably entrenched in the system. The latter added time by making me re-do many things that I did wrong on the first attempt, but the trade-off is that we're not stuck with nearly as many bad design decisions now.

After I had the very early incarnation of the VM and compiler mostly working, I started talking to a few people about my plans. Initially, I contacted a few people who had been doing a lot with TADS 2, particularly people who had big ideas but were bumping up against the limits of the old system. I wanted

Continued on page 2

Contents

Mission Statement

Inside Erin is written and published by people who enjoy AIF. It is done for fun, but we also have some goals that we seek to achieve through the newsletter:

1. To encourage the production of more quality AIF games by providing advice from game developers, and by offering constructive criticism that is specifically relevant to AIF.

2. To encourage activity and growth in the AIF community. We aim to generate a constant level of activity so that there aren’t long periods in which people can lose interest in AIF.

3. To help document and organize the AIF community. This is done through reporting on games and events, as well as by helping to organize community-wide activities such as competitions and the yearly Erin Awards.

their perspectives on what I was doing with the new version, to make sure that it would actually address the kinds of problems people were having with TADS 2. It would have been a fairly pointless exercise to build a whole new version and find out that it still had the same limits as the original.

As time went on, more people started hearing about the new system, and I started getting email from people who wanted to know more about it. I wasn't really talking about it publicly at that point, but I also wasn't swearing anyone to secrecy. There's a long history in rec.arts.int-fiction of people grandly announcing The Greatest IF System Of All Time, talking it up for a couple of months, making all sorts of grandiose claims about how it will solve all known problems and even make delicious julienned fries - and then just disappearing from the group. I didn't want to become another vaporware salesman, so I restrained myself from talking publicly about the new system until I was sure I was going to be able to deliver it.

Even so, before long I was talking about it semi-privately with a bunch of people, and the "cc" list was becoming pretty unwieldy, so we set up a mailing list for the discussion. That proved to be a great approach. The list became an informal working group. Someone would bring up something they'd like the new system to do, and we'd discuss possible solutions; in most cases, a consensus would emerge on the best approach, and then I'd go implement it. A lot of the adv3 library got designed this way. I think this turned out to be a really good process - it gave us the benefits of having multiple people attack a problem, but without the "design by committee" drawbacks, in that ultimately it came down to one person doing most of the coding. I think the danger with a committee design is that there are inevitably a few things that a group - any group - can't agree on, and when those come up, committees tend to find really tortuous compromises that don't really make anyone happy but at least don't make anyone too mad. With our informal discussion group, when one of those intractable, consensus-defying problems came up, I kind of took it upon myself to be the tie-breaker, in that I'd just go implement the solution I liked best. I'm sure I often chose the wrong answer, but it at least kept us from bogging down too much on any one thing.

AB: Most programmers will tell you that there are always tweaks and changes they wish to make to any project. What was the biggest factor in deciding that TADS 3 was finally finished enough to take out of beta?

MR: I think it was just the accumulated weight of the time it had been in beta; it got to a point where enough was clearly enough.

I actually had been planning to release the first official version about a year earlier than it finally came out. I had already put together the first "release candidate," and my plan at that point was to turn all of my attention to writing a new Author's Manual. The new manual was really the last piece that we all agreed was needed before the official release. Like you said, there are always endless tweaks, large and small, that you want to add to software; but everyone on the list was pretty much in agreement that the software had enough features by that time. So I got the release candidate out and started on the manual - outlining, putting together tools, and so on. My ambition was to be able to produce a nice printed manual, like the TADS 2 Author's Manual, but also produce a well-designed HTML system from the same sources.

I made some progress at building the tool infrastructure for what I had in mind, but the content of the manual never really got off the ground. I had some other non-TADS things keeping me busy, and I also was spending a fair amount of time continuing to work on adv3 enhancements. Eventually I'd put enough new work into adv3 that we needed to go through another release candidate cycle.

Since so much time had gone by since that first release candidate - as I said, about a year - I realized that the new manual just wasn't going to happen as I'd hoped. I mean, the project as I conceived it was huge; it would probably have taken me another year. I didn't think it was reasonable to ask people to wait that much longer; many people were already actively using the system, and the "beta" label was a real problem for them. For one thing, they had to worry about whether I'd be making incompatible changes; but the bigger issue was that players were naturally less inclined to install a beta interpreter, so authors working with the new system might have seen their audiences artificially limited by my inability to get the final release out.

So I started thinking about how to deal with the missing manual. I looked around at what material was already out there - Eric Eve's books, the various notes and help files I'd put together, the "how to" articles on tads.org, the library comments... and I realized that there was actually a ton of material available, in large part thanks to Eric's prodigious efforts. In fact, taken together, the extant material covered just about everything I had put in my outline for the new manual. The only reason it seemed like the system was skimpily documented was that all this information was spread out into so many pieces. So I talked to Eric, and I talked to Ed Stauff, who wrote a program that generates Javadoc-style documentation by analyzing the adv3 code. I proposed assembling all of this material into a suite, and then bundling the suite with the system. Eric and Ed were quite positive on the idea, which meant we had a solution for the documentation problem, which meant we could finally produce the official release.

AB: I couldn't help but notice that TADS 3 was released from beta right around the same time frame that Inform 7 was picking up steam. Is that a coincidence, or possibly a little friendly rivalry?

MR: I think it was mostly coincidence. We probably felt a little extra pressure to get our show on the road, as it were, when people first started talking publicly about Inform 7 a while before its first beta release. But really, there was plenty of built-up pressure of our own, from the extremely long time TADS 3 had been in beta.

AB: With regards to Inform 7, I realize that TADS 3 and Inform 7 take almost diametrically opposed views to the creation of IF, but when you first heard about Inform 7 did you have a "I wish I thought of that" moment?

MR: Well, first, let me establish that I'm very impressed by Inform 7. It's a nicely put together system, and I think the approach will be very appealing to a lot of people. More than that, I'm impressed by the audacity of it. People have been idly kicking around roughly this idea for years, but the emphasis was always on "idly." It's one of those perennial blue-sky ideas that naive newbies are forever dreaming up and posting to the group, and which all of us wizened old experts are forever pouncing all over, dismissing out of hand because it's so obvious that it would never work once you got beyond the really basic stuff. But Graham actually sat down and worked it all out, not only how you could handle the obvious stuff like static room and object definitions, but also the real programming parts. Most people who have tried this before didn't think beyond toy games, but Graham is obviously someone who understands what's required of a first-class IF system, so he wasn't going to just stop at simple room definitions. I'm amazed that he was able to take it so far.

[AB: "Graham" is Graham Nelson, who created Inform and the most recent version, Inform 7.]

Now, to answer your question, did I wish I'd thought of it: no, not so much. As I said, lots of other people have thought of it, so I think the real question would be did I wish I'd pursued the idea with as much determination. And the answer to that is still no. As impressed as I am by what Graham's accomplished, it's just not the kind of thing I'm interested in building. As you said, it's a very different approach to programming. I've actually used a few other programming languages that aspired to be English-like, such as HyperTalk, and I've found they're not to my taste. My own experience with that sort of language is that they're great much of the time - but extremely frustrating some of the time, enough to outweigh the good parts. I don't know if Inform 7 specifically will suffer from that, but my negative experiences with other languages using the same rough approach have kind of soured me on the concept.

But despite all that, I can see how the English-like approach could be unusually appealing in this particular context. It's great that someone has decided to give it such a serious try.

AB: The tutorials, tour guides, quick reference charts, etc. for TADS 3 have all been developed by Eric Eve. They are a remarkable resource for any new author, or even existing authors. How closely did the two of you work on developing these?

MR: Eric gets all of the credit for those. He has an incredible ability to not just figure things out, but then to explain them to everyone else, to share what he's learned. Eric says that the Getting Started Guide, the Tour Guide, and the rest came out of his own early attempts to puzzle out how adv3 works by looking at the code and experimenting with it - I think he basically wrote the manuals he wished he'd had at the time. My only direct involvement in Eric's books was that I reviewed the early versions and offered a little feedback.

Eric's involvement in the documentation actually goes beyond his own books. When I approached him about my plan to create the documentation suite, Eric got very involved in that project - helping organize the overall bundle, reviewing new and old material I wrote, contributing new sections to the System and Technical manuals. Left to my own devices, I probably would have just bundled the old material as it was, but Eric felt - rightly so - that it would benefit greatly from a little work to tie the pieces together. Thanks to all his help, it came together as a much more coherent whole.

AB: Some of the complicated parts of AIF games are the ones dealing with layered clothing and body parts (we'll pretend we're talking about hair, hands, etc. for purposes of our discussion). Can you briefly outline the sorts of additions to TADS 2 within TADS 3 that may help with these constructs?

MR: Layered clothing is one "simulationist" area where the adv3 library doesn't really have any built-in support, although I think there's an extension that adds this. What the library does have, though, is a fairly comprehensive and extensible model for determine what's visible, what's in reach, and so on, and I think all of that is a key foundation for building a layered clothing system. In TADS 2, the sense-modeling system is very primitive; there are basic concepts of "visible" and "reachable," but it's often difficult to customize what those mean. The adv3 sensory system is extensible - you can add whole new senses if you want - and does a much better job of abstraction, so that you can customize exactly what it means to be reachable, for instance.

Another core feature of adv3 that should help considerably with layered clothing is the precondition system, which both imposes conditions on actions and also triggers implied actions to bring about those conditions. This would make it relatively easy to do things like automatically removing a coat before you could remove a shirt.

Finally, one of the subtle but really important improvements with tads 3 is that it's just a more powerful programming language. Something like layered clothing is a fairly complicated thing to think about, just in terms of the "mental model" - there's a lot to keep track of. Implementing a complex model is less work when the programming language is more powerful.

AB: The other complicated parts are the same for AIF as regular IF: dealing with character interaction. I'm vaguely familiar with ActorStates, but can you briefly outline the sorts of things within TADS 3 that can help with character descriptions and dialogue?

MR: That's a big subject. For anyone who really wants to get into the details, I wrote a whole article on the tads.org site about this - http://www.tads.org/howto/t3actor.htm. It's in three parts - parts 1 and 2 are mostly background information, analyzing the basic problems an IF author faces in implementing animated characters, and discussing the various known solutions; part 3 is specifically about how to implement some of those solutions in tads 3.

I'll try to give you a few highlights, though. Adv3 provides a lot of tools for two basic aspects of NPCs. The first is that characters have to be dynamic - they have to change over time. The tools for this aspect are embodied in ActorState and its subclasses. Basically, the old way - the TADS 2 way - is to define all of the changeable behavior directly in an Actor object, which means using a ton of conditionals ("if" and "switch" statements) spread out over lots of different methods. The adv3 way is to define only the static parts of an NPC in its Actor object, and then define the different "states" the actor can be in as separate objects. So, for example, rather than describing what the actor is doing as part of its Actor.desc property, you'd only describe there what the actor *always* looks like. Then, you'd describe what the actor is doing in each of its several ActorState objects. So you might have one state object representing "sitting at the bar" - that object would have a little add-on description string like "She's sitting on a bar stool sipping a drink." You'd have a separate state object for each other thing the NPC can be doing at any given time. As the NPC goes about her business, you switch to the correct state object at each time. Each state object is a little bundle of all of the customizations that apply to the state it represents.

The second set of NPC tools in adv3 is for conversation. The tads 2 approach was basically to foist the parsing of conversations onto each game author - the system would just give you the string from an ASK or TELL, and you'd pretty much have to do the rest. Adv3 gives you a lot of new tools. The key new tool is the notion of "topic databases" - rather than doing the parsing yourself in a giant "switch" statement, you create a little object for each conversational exchange you want to program. Each object is a topic response, and taken together, the responses form a database of topics. The cool thing is that you can make topics conditional on all sorts of things, ranging from physical conditions in the game to past conversation topics to what the NPC knows at any given time. The library keeps track of which topics are allowed and which aren't, and automatically picks the right one each time the player enters a conversation command. The library also has features for keeping track of the flow of a conversation, so that NPCs can act more naturally - they can greet you when you start talking, say goodbye when you leave, and so on.

There's a lot more, but that should give you some idea. The article I mentioned goes into a lot more detail.

AB: What is next on your agenda? Are you starting TADS 4?

MR: Well, not quite yet; there's a lot left to do on TADS 3. Since the release, I've been working on some Workbench for Windows enhancements, which are just starting to become available - there's an alpha release on the tads.org site for anyone who wants to take a look; just go to the tads 3 page and look for the alpha link. There are lots of additions, but the two big ones are integrated documentation search, and an integrated text editor. (Some time ago, back during the lengthy tads 3 beta period, I came across an open-source programmer's text editing component called Scintilla, and was quite impressed by it. I've been eager to integrate it ever since I discovered it, but I didn't want to add yet another delay to the beta, so I held off. Once the official release was out, I started on the integration project. I think it's turning out very nicely - with the addition of the editor, Workbench will *finally* be a real IDE.)

After that, I have a couple of things in mind, but nothing I'm ready to talk publicly about quite yet. That whole vaporware salesman fear again...

AB: You mention the new TADS IDE. Are there any plans on making such an animal backwards compatible with TADS 2?

MR: It's a possibility. I've actually roughed in most of that already - the TADS 2 and 3 versions of Workbench are mostly common code, so adding a feature to one tends to add it to the other more or less automatically. It's not completely automatic, of course, as there are a number of differences between the two. For example, the "project" list is only in Workbench 3, so any new features that tie into that need to be handled differently in Workbench 2. But if there's demand, I don't think it'll be a huge amount of extra work to get most of the new functionality into a Workbench 2 update. There'd be some testing and packaging work, but nothing insurmountable. The big new features - the integrated editor and the documentation search - would definitely be there.

Thanks again to Mike Roberts for taking the time to answer our questions. For anyone who is interested in learning more about TADS 3, please check out the home page for TADS (versions 2 and 3) at http://www.tads.org/.

Letter from the Editor (Continued)

from school several times. They got better, briefly, only to have the strep return for another round. I got a nasty cold. My wife got a nasty cold. Our microwave oven broke and we had to shell out for a new one. Is that all? Probably not, but it’s more than enough to give the idea.

So, what does all this complaining have to do with AIF? Well, that’s the point - it has nothing to do with AIF and neither have I since I finished the last edition of the newsletter. I plan on changing that. Look for me at least to participate far more frequently on the message boards in March. I’m looking forward to running the 2007 Mini-comp (see announcement in this edition). I may very well even enter it myself with a small game that I’ve been fiddling with.

Anyway, thanks for humoring me and putting up with my griping. I expect to have more relevant stuff to discuss next month. Oh, and please do enter the mini-comp. I was incredibly impressed by the quality of games entered last year and I really think it’s only going to get better.

Until next month, then.

Interview with Knight Errant by A. Ninny

AIF Community member Knight Errant contacted me recently asking to be interviewed for the newsletter. I agreed—and anyone who wants to participate in an interview like this simply has to ask. Thanks to KE for his interest in participating and for his thoughtful answers to my questions.

AN: You contacted us asking to be interviewed. So who are you, what brings you to AIF and what makes you a worthwhile subject?

KE: I'm a mid-20's biology lab technician in the USA. I've always enjoyed literary erotica, and I stumbled across the "Spellcasting" series on an abandonware website. For those of you who aren't familiar with that series, it's a graphic IF with several good softcore porn scenes. Then I found the old AIF archive and was hooked.

What makes me worth talking to? I suppose that's for the readers to judge, but I'd like to get more involved in the AIF community.

AN: How much did you participate in the recent Erin awards?

KE: The past Erins was the first time I voted. I also attended the award ceremony and generally made an ass of myself. I'm going to try and be more involved in the next Erins, though.

AN: What did you feel about the results? Were you as excited about GoblinBoy as the majority of voters seemed to be?

KE: GoblinBoy really caught my attention with his first game, *Camping Trip*. *Baron's Plot* was an excellent (and hilarious) follow-up, but like everyone I was totally blown away by *Key of Eternity*. That game was enormous and incredibly detailed, with excellent use of images. I have no idea how he found the time to write an epic like that. On the other hand, it's sad that in order to honor GoblinBoy for his efforts, so many other great authors had to be overlooked.

AN: You recently posted about having writers block beginning writing sex scenes for a game you're working on. Have you gotten past it yet?

KE: It's still a little frightening, but I'm starting to get descriptions into the system. I'm going to just do my best and see what my beta-testers think of it.

AN: How did you manage to break through?

KE: I went back to A Bomire's *Last Minute Gift* and took a look at his sex descriptions. It helped to just take a look at his source and read the text out-of-context. That helped me get an idea of how to approach it. It also helped to get the coding framework together, to see how the various parts of the SSS will work together once it's in the game. Finally, like any major piece of writing, the hardest part was just getting started. Once I filled out the body part descriptions, the foreplay didn't seem quite as scary. Now that the foreplay's coming together, the sex doesn't seem quite as scary. Still, I'm glad I can take a break and pound my head against disobedient code when I'm not in the mood for writing sex.

AN: Can you share anything about this work in progress?

KE: It's my entry for this year's mini-comp, a short TADS 3 game set in ancient Greece. I suppose technically it's a collaboration with Aristophanes, but since he's been dead for around 2400 years I'll have to accept any credit or blame on his behalf. It features a layered sex system, inspired by your *Parlour*, but not nearly as ambitious.

We wish Knight Errant the best of luck in his AIF endeavors and look forward to playing his Mini-comp game.

2007 AIF Mini-comp by A. Ninny

We are happy to announce the commencement of the **2007 AIF Mini-comp**! Ladies and Gentlemen, start your imaginations!

After conducting discussions with the community about different ideas for rules (such as adding optional themes and expanding limits), I have decided to hold the competition this year with rules basically identical to those from last year, though I did revise slightly the limits on characters. The reason for this decision is that I'd rather not take any action that would reduce the opportunities for authors to participate. The theme idea was an interesting one, but making them optional seemed rather pointless. Plus, the quality of games entered last year was generally excellent. Why fix what ain't broken?

So, without further ado - here are the rules and important dates for this year's mini-comp:

Mini-comp submission rules are as follows:

- Your game must have three or fewer rooms. Closets do not count as rooms so long as they're just places to store things. If your player is required to spend more than a couple of turns in a closet, it counts as a room.
- Your game may have no more than three characters, including the player-character(s). No more than two of those characters may participate interactively in sex scenes. **NOTE:** This is a expansion on previous years' rules and allows the PC to be a non-participant or voyeur while the two non-player characters have sex, and also allows the game to switch the PC from one character to another.
- Multimedia (images and sounds) are permitted, but may not add more than 150KB to the native (unzipped) size of the game file.
- No part of your game can have been released to the public before the deadline.
- Your game must be winnable (or at least it must have an ending that the player can reach).

Mini-comp submission procedures are as follows:

- The submission deadline is **9:00 a.m. CST Friday, May 11, 2007.**
- I will be available to help beta-test your game. Beta-testing is strongly encouraged but not required.
- I will collect the entries by e-mail and post the games on the AIF Newsletter web site. Send your entry to ninnyAIF@gmail.com.
- Authors should send a walkthrough with their entry. The walkthrough will be used by competition organizers to verify the game can be won and to provide hints for players.

Voting procedures are as follows:

- Everyone, including entrants, will be allowed to vote.
- Voters will have approximately two weeks to play all the games and vote. The voting deadline will be announced when the games are released.
- Voting will be conducted in a manner similar to that of the Erins: return a ballot marked in order of preference for each category. Votes will be counted using the instant-runoff method.
- Discussion of games (including requests for hints) will be forbidden during the voting period.
- Authors will not be permitted to post updates of their games during the voting period. They may post 'technical bulletins' with instructions as to how to work around bugs that are discovered. Technical bulletins must be approved by mini-comp organizers before being posted.

I am tweaking the judging this year. Voters will be asked to judge all the games in the following categories:

- **Concept.** Is it a good idea for a mini-comp game? Does it work well with the set limits? Does it feel complete or more like a game fragment?
- **Writing.** How well-written is it? Do the settings have the atmosphere that the author seems to be after?
- **Characters.** Do the characters 'come to life'? How sexy are they?
- **Sex.** How hot are the sex scenes? How well do the characters engage in sex?
- **Technical.** How many bugs are there? What neat tricks did the author invent?
- **Enjoyment.** How much did you like the game?

Good luck to all who plan to enter. I look forward to playing this year's crop of games and expect and hope that the quality will continue to improve.

Seven Seas of Thẽah - Episode 13 by Christopher Cole

OPTIONS: At the end of this story each month, you will be given a number of options. Choose the option that you like and vote in the poll at the Yahoo AIF Archive. The option that gets the most votes will determine how the story continues in next month's newsletter.

NOTE: You can read background information and other tidbits about this story here: http://ccole.aftermath.cx/theah.htm. ***The page has been updated for this episode.***

Magnus stood and was somewhat surprised that he appeared none the worse for wear. With the exception of his hair, he was no longer wet which meant he must have been unconscious for some time. The singing continued in the unknown distance but did not appear to be getting any closer, so he took the opportunity to examine his immediate surroundings.

The cave was not large. A few paces and he was at the far wall. He felt the wall and the water that dripped down its surface was cool. The rock was rough and hard. He moved around the cave in a circular pattern, moving around the pool at the center, and was soon back to his starting point. There didn't appear to be any way out of the cave, save perhaps for the pool itself.

As he was contemplating this dilemma, the singing began to rise in volume. He still could not make out where it was coming from. A slight edge of panic began to creep into Magnus' mind.

The dark water in the pool began to move, causing Magnus to jump. He backed against the wall and stared in fascination as ripples appeared on the surface of the pool. The ripples grew into tiny waves and then from within appeared a figure. It was a female with long, wavy brown hair. Its face was partially tattooed and it wore a circlet of beads and feathers. The female glided to the edge of the pool and rested upon the cool rocks. She was topless and her full breasts floated atop the water, pressing against the edge of the pool. Magnus was stunned, partially by her abrupt appearance, but mostly by her beauty.

She looked at him with a distant curiosity as he stood there, knees bent, ready to defend himself. She lifted herself out of the water and stood on the stones at the edge of the pool. Water dripped off her toned, nude form. In his dazed state, Magnus was expecting her to have the lower torso of a fish; a mermaid from legend. She was all human however, or at least appeared to be. There was definitely something not natural about her however, not the least of which was her sudden appearance from the pool.

She moved slowly towards him, her body gliding sensuously over the wet stones. Magnus did not move, but remained ready for anything. He was vaguely aware that the singing had stopped. She stopped in front of him and tilted her head as she looked at him. She cautiously placed her hands onto his chest, and Magnus could feel a warmth spread through his body. His manhood began to stiffen. She felt his muscled chest and pressed her breasts against it as she peered up into his eyes.

WHAT SHOULD MAGNUS DO?

1) Grab her by the arms and demand to know what is going on?
2) Flee by diving into the pool?
3) Take the woman sexually by force?
4) Move away and ask what is happening?
5) Return her touch?

R.A.G.S., the Rapid Adventure Game System, A Review By A. Bomire

A new authoring system was introduced to the AIF community this month. It is R.A.G.S, the Rapid Adventure Game System. From its website, R.A.G.S. is described as "a flexible and powerful tool designed to allow you to quickly and easily create fun, exciting games and distribute them to your players". Since I'm always interested in seeing what is new in game development (you never know, I might find something I like better than TADS someday), I decided to try out this new system. No, I didn't create a full game such as I did for Inform 7, but I did create a small test game just to explore the capabilities.

From a player's perspective, R.A.G.S. seems like a mixture of modern day text adventures and older graphical adventure games such as those produced by Sierra. Instead of typing commands, all interaction is done using the mouse. The player is presented with an array of panels, each representing a different part of the game. One panel displays the text describing the results of the player's actions. Another has a compass rose of available directions, active directions highlighted in green. Clicking on a direction moves the player. Other panels list all objects in the room, all visible characters, and the player's inventory. Right-clicking on any of these things brings up a menu of actions associated with that object: EXAMINE, TAKE/DROP, etc. These actions can change during the course of the game as new actions appear and other actions disappear. And all of this can be accompanied by pictures - pictures of the room you are in, objects within that room, characters, or whatever the author wishes to display.

That is the player's interface, or runtime. But what about creating games? Well, the system advertises itself as Rapid, and lives up to that name. Creating items, rooms and characters is very easy. You click on a "Add" button, and give the needed object a name and description and you are finished. To place a character or object in a room, select that room and you'll find a button to add the object to the room. Right clicking on the screen presents the author with context-relevant menus. Right click in the object window, and you are presented with a menu to add a new object. The same is true for characters and rooms. Joining rooms to each other is a simple matter of clicking on the exits tab for that room and clicking a drop-down menu of rooms for the required direction (very similar to ADRIFT). Adding pictures is just as easy. First, you add a picture or collection of pictures into the "picture library". From there, you can select a picture for each room, character, or object.

When you place these objects and characters into your game, by default there are no actions defined. You have to add any actions, including such obvious ones as EXAMINE, TAKE/DROP, OPEN/CLOSE, etc. This leads to the most important, and unfortunately most confusing, portion of the interface - the Actions section. New authors should search around on the R.A.G.S. website for the online documentation about actions before using them. Adding actions is as simple as adding anything else - right click on the Actions panel, and you are given the chance to add an action. There are several default ones listed or you can add a custom action. The name you give to the action is what appears on the action menu during the game. Once you add an action and name it, you have to add various commands to execute when that action is chosen. Those commands are broken down by "Commands on Success" and "Commands on Failure". These are meant to allow you to set conditions upon the actions, with different commands executed on success and failure, but it can be a bit confusing when there are no conditions set.

However, once you get used to the action interface, there are a broad set of commands you can execute. Some examples are "Display Text", "Display picture", move an item to a player or character inventory, move an item to a room, move the player or a character to a room, and change the picture for a room. All of these have drop down menus to allow you to select the various options available. Each action can have as many commands as you want. You can even activate/de-activate other actions, changing the menu of actions that appears for each character or object. A final bit of flexibility is that commands can be set to allow input from the player. (Okay, that bit where I said you don't have to type anything? Not entirely true.) Thus, you could set up a command to "ASK ABOUT..." and have the player input a specific topic. Or you could set up a command to "USE DILDO ON..." and have it prompt you with a list of available women.

As mentioned, all of these commands can have conditions placed upon them, with different commands specified for success or failure of that conditional check. Some of the conditions you can check for are if an item is held by a player or character, if an item is in a room, and if the player is in the same room as a character, to name a few. You can even define your own variables and test them for their value.

Lastly, each action can have many commands associated with it, all executing when the player clicks on the menu selection. This lets you perform many things seemingly at once - move objects or characters around, display pictures and text, turn menu selections on or off, etc.

By defining your own actions, you negate the main problem of text adventures - guess the verb. You have complete control over what actions are defined, and when those actions are available to the player. For the player, right clicking on any object, character or room reveals the possible actions available. Thus, there is never a need for a player to try to guess what commands can be used. However, the player can still sometimes be at a loss as there is no indication of when a new action has become "active" for an object. The player can be reduced at times to right-clicking on all objects, looking for the next possible action.

It should be said that the R.A.G.S. program is still in development. As such, you can find bits and pieces that don't really work as of yet. For example, there is an "Add Action Wizard" button in the actions portion of the program that didn't really do anything. Some of the commands didn't work like they were advertised, such as the "Change Character Description" command that didn't really change the character description. There is an option to prompt the player for a name, which didn't work either. Oddest is the player runtime, which doesn't do anything when you click on "File-Exit". You have to click on the "X" in the corner of the screen to close the game.

For authors who are unfamiliar with the system, it would be helpful to have connection to the Internet when working on a R.A.G.S. game, as there is no in-game help or any offline documentation. All documentation is currently available only on the R.A.G.S. website, in the form of a list of topics under the site's "Forum".

Other aspects of the system which may be offputting: it is Windows-based. Authors who like to use a Mac or *nix will have to find a Window's emulation program. It requires Microsoft's .NET framework, which adds 22MB to the download. If you already have .NET installed, then the downloads are much smaller. Finally, it has to be said that the game system is not at all friendly to the blind or visually impaired members of our community.

My final impressions: The player interface is certainly a new experience. The mixture of graphics and text take a little getting used to, but if you like the look and feel of graphics-heavy games such as "X-Clue", "Birthday" and "Plan 69" then you'll like this interface. Game development is indeed rapid and easy. It took me about a half hour to create a two room game with one NPC, and link several actions with pictures of that NPC. If you wanted to use this system to develop a game without any graphical elements, then I think it would actually be cumbersome and awkward - both to write and to play. However, if you are interested in developing a graphical game instead of a game based upon traditional text interfaces, then perhaps this system is of interest to you.

For more information about R.A.G.S., then check out the website at http://www.ragsgame.com/.

Game Reviews

Ron Weasley and the Quest for Hermione, a review by Grimm Sharlak

Game Info:	***Ron Weasley and the Quest for Hermione,* released 30th October 2006**
Author:	**captainc22**
Platform:	**ADRIFT 3.9**
Size:	**329KB (zipped)**
Content:	**m/f, m/f/f**
Game Type:	**Sex Romp**
Length:	**Medium**
Reviewed:	**February 2007**
Extras:	**Spells list**

Basic Plot:
You play the title character of Ron, as you make your way around the halls of Hogwart’s trying to woo the love of your life, Hermione Granger. This of course doesn’t stop you from trying to score every other female student on the way, but hey, you’re not with Hermione just yet!

Overall Thoughts:
Just a disclaimer, guys: I’ve never read any of the *Harry Potter* books or seen any of the movies, so I come into this self-proclaimed piece of fan fiction with little to no knowledge of the settings and or characters. On that note, I found this game to be your typical sex romp, with the characters being somewhat wooden and interchangeable, outside of their descriptions. This is just your typical “find the right thing, girl will jump your bones” type game, although you can occasionally lose the game by going to far with certain ladies.

Puzzles/Gameplay:
Puzzles are basically of the “talk to this person, get this item for them” variety, with the main quest of the game being based around obtaining a list of items for Hermione before you tell her how you feel about her.

There is a slight twist, though, due to the magical nature of the school and its students, you can actually cast spells with your wand to help solve in game puzzles. This is a neat little touch and for the most part which spell to cast is clear and actually casting it straightforward.

Sex:
Unfortunately the sex in the game is pretty repetitive, with the dialogue from the girls being very similar and very stilted outside of the game’s final scene. I swear to God every girl said “that’s messy” whenever they rubbed your cock. Ron himself doesn’t even sound that interested in

proceedings dialoguewise, and the actual scenes lack flair. Aside from the aforementioned final scene and one threesome scene (which was actually really good) the writing is lacking.

One neat twist with some of the sex scenes is, since you're on a quest for Hermione's love after all, having sex with certain girls could result in your game ending. Nice to see a little bit of realism in there, although to avoid this fate you can just do everything but penetrate the girls in question.

Technical:
The game is not very ambitious in a technical sense, and I found no bugs or issues (except for one where it wouldn't save your high score list, but I think that's more an interpreter problem than anything the author could have done). The only problem was with a lack of synonyms in casting spells. On two separate occasions, I tried casting spells at targets, using names that auto complete said were correct, but the author had only written the commands for a different alias (like a character's first name rather than their last). Since both of these scenes required you to cast the right spell or fail, it could be frustrating, but figuring out the correct name to use never took too long to figure out.

Intangibles:
While I'm certain this will appeal more to *Harry Potter* fans than to those who haven't seen or don't like the franchise, the writing is still fairly wooden and the characters lacking in individuality.

Final Thoughts:
Quest for Hermione is a solid little game, with some fresh ideas despite its basis in fan fiction. However, the writing really lets it down when it comes to the meat of the game, and the puzzles are too simplistic and clichéd to truly carry the game on its own. A bit more focus on making each sex scene more unique and the dialogue a bit livelier would go a long way with this one.

Rating: C+

O Erin!

O Erin! is on hiatus. Look for its return in a future edition. All previous comic strips can be viewed on the Newsletter web site.

AIF Wants You

If you can write game reviews, articles, opinion pieces, humorous essays, or endless blather, we want you. Contact the Editor for suggested content or just write what you want and send it to us.

Staff

Editor	A Ninny is an AIF player, author of three AIF games and frequent beta-tester. His *Parlour* received an Erin for Best "One Night Stand" game in 2004. His most recent game is *Malaise.*
Web Master	Darc Nite is a newcomer to the AIF scene. He is an avid gamer who heard the call for help with the AIF Newsletter.
Staff Writers	A Bomire is the author of several TADS AIF games, including *Dexter Dixon: In Search of the Prussian Pussy*, *Tomorrow Never Comes* and *The Backlot*. His games have won numerous awards and Erin nominations.
	BBBen is an AIF author. His games have received two Erin awards, numerous nominations and first place in A. Bomire's 2004 mini-comp.
	Grimm Sharlak is the author of two AIF games: *Breakout* and *Of Masters and Mistresses: Abduction*.
	Christopher Cole has written many popular ADRIFT AIF games, including *Camp Windy Lake*, *Gamma Gals,* and *Mount Voluptuous*. He is the 2005 winner of AIF's Badman Memorial Lifetime Achievement Award.
	Bitterfrost is a longtime IF/AIF player working on his first (and last) game, *How I Got Syphilix.*

Submitting your work to "Inside Erin"

Please direct all comments, articles, reviews, discussion and art to the Editor, A. Ninny, at aifsubmissions@gmail.com.

RATED M
M
18+ ONLY