

In the long run

Just over a month ago Jesse Spears, the developer of Harpoon 3, made an announcement on the long-term future of the simulation (*for those who missed it, it is featured on the news section on our previous issue*). His statement was nothing less than groundbreaking, at least in the domain of the Harpoon series.

In no ambiguous terms, J.Spears stated that he plans to build and support an alternative version of the Harpoon 3 simulation, in addition to the existing one. The sole difference between the two versions will be the structure of the program: the existing version will continue to have all the program logic encapsulated in the single executable, while the alternative version will have the various program subcomponents (wargaming mechanics, interface, AI logic etc.) broken down into separate DLLs, whose source code will be available to the public for editing & recompiling. A central “closed” executable will still be present in order to prevent wholesale theft of the source code, but the rest of the program will be essentially reaped for modifications.

The main motive for this version was stated by the developer himself: *“The advantage is the same for any Open Source-style project: Lots of eyes looking at the code and lots of brains thinking about the code will find more problems than just one set. I’ll integrate bug fixes and significant changes back into the main source tree from time to time, but for the most part, I’ll stay away from [this version].”*

You’ll still need to buy a copy of the game to work on the Open Source project, since I’ll be the only one building the “Main” Application (everyone else will just build DLL’s), but if you have Modifications you think need to be in the game, you can work on them (and share them with everyone else, or not if you don’t want to). I will ask that all bug fixes be rolled back into the Open Source project (or at least given to me so I can roll them into the main source tree).”

These news gave us all here at the HQ plenty of food for thought. So, H3 is going for a semi-open source architecture. This is, without doubt, great news for a community who, in the past, has long been advocating greater participation from the part of the users on what gets into any of the Harpoon simulations. We have been asking a bazillion features every time a new version is in the works; now for the first time we will have the opportunity to examine the actual code and, if possible, add our wish-list items ourselves.

We have also been thinking something else. This transformation of H3, if indeed it materialises, will set a new paradigm in the non-classified combat simulation market. One that is worth examining, contrasting it with hitherto existing practices and considering its potential relevance for another project in the works: Harpoon 4.

Until now, in the computer entertainment industry, there have been three main paradigms with regards to after-market program modification:

1) The program is deliberately “sealed” in its structure. All supporting data (graphics, sound, separate code chunks etc.) are purposely protected from modification, either by encryption to a proprietary format or by being rolled-up into the main executable or by some other mechanism.

2) Most of the program is locked, but some bits and pieces of aesthetic purpose (3D models, textures, sound effects files etc.) are deliberately left open to modification. This is particularly popular in flight simulations, as it allows the creation of custom skins for aircraft and other virtual objects. Until very recently, user access to the program’s 3D models in simulations was a taboo issue, but the explosion of mod-making across the games industry in the late years has been beating hard on that barrier. Third Wire’s “Strike Fighters – Project 1” (not uncoincidentally the creation of T.Kawahito, a long-time advocate of empowering user mods) definitely points the way forward in this respect, allowing full user customisation of the simulation’s objects and their behavior.

3) A significant portion of the program material is left user-accessible, including data that determine how the program interacts with the user, as well as its internal mechanics. This also includes cases where the program is structured deliberately so as to allow user modifications to both its appearance and behavior. The old Microprose (R.I.P.) was a decisive advocate of this approach in its last years: Tank sim buffs may recall the public distribution of a Word document that described, in excruciating detail, how the text files that contained the battle scenarios of M1 Tank Platoon 2 could be modified (or new ones created from scratch), and how the behavior of the simulation objects could be determined to a fine degree by the same parameters. Falcon 4 raised the bar significantly in this respect, by allowing modifications on aspects such as operational doctrine, complete 3D shapes, OOBs, tactical AI behavior and myriad other details in addition to the more “traditional” custom skins and sound effects. (It was most unfortunate that Hasbro did not actively encourage the dissemination of information regarding these mod capabilities). Fleet Command, a program not overwhelmingly popular with certain members of the HHQ crew, had the almost-redeeming feature of modifiable doctrine files that could dictate the behavior of AI-controlled units. Harpoon 2, with its fully modifiable database as well as a mission editor that was essentially writing the book on AI behavior, was another example.

A semi-Open Source H3 would declare a new way of doing business in the sim/strategy genre: admitting without shame that the released product is imperfect to begin with, and actually encouraging users to work in modifying the

offered program, either to polish the rough ends or to adjust it to their own personal needs & preferences – or both. This is essentially what Microsoft has been doing for over a decade now with most of its modern applications (Office VBA anyone?), and the practice seems to be a resounding success.

In the entertainment software industry of these days, a product's post-release lifecycle can typically be measured in the span of a few months. How many games can you name that are being **actively** supported (a **free** patch/upgrade in the works, developer participation in discussion forums etc.) six months after having hit the shelves? Not many. However good the intentions of the development team, they have no choice but to support the title they released (after much effort on their part) just for a while and then move on to other projects, or get in serious trouble with their publishers. Such is the scheme of things.

Past experience however indicates that the products with the greatest degree of "openness" to outside tampering are the ones with the most long-term sales. A small selection of examples follow:

- Harpoon 3 started as a project after J.Spears, in his own words, got sick of having people contact him and ask where they could find a copy of H2. That was years after the release of the program, long after even the last copies had vanished from the bargain bins. H2's longevity can easily be attributed (apart from its inherent qualities that made it one of the best simulations ever) to the efforts of dedicated members of the Harpoon community.
- Falcon 4 has recently been observed being still in store shelves, for as much as 20GBP! That is a full four years after first release, and 3 years after the last official patch was released. People are still buying the simulation, install F4UT's SuperPak3 (freely downloadable from the web) and simply enjoy the finest non-classified air combat simulation of all time. In simple terms, Infogrames is still making money from a product that would normally not be even mentioned on the company's website.
- Fleet Command, until the recent eruption of interest over the expected release of H4, still attracted sales and interest as a result of NWS's series of upgrades. Not bad for a product that was initially released in a very poor condition in mid-1999.
- Microprose's Gunship!, (what should normally be called Gunship III), is still attracting sales as a result of a large modding endeavor by a team of Gunship-maniacs. Again, a product released in a premature state is having a second life thanks to outside efforts.
- The source code for AAA titles like MiG Alley, Battle of Britain and most recently Comanche-Hokum (among others) have been made available to the public in their entirety. Mod projects are already in full swing for all three titles and are expected to re-arouse interest around them. Interest that more than often translates to renewed sales.

The general underlying message of the customers to the producers & publishers is clear: Secrecy leads to reduced interest, and sales are limited to those that will buy the product no matter what its condition. Openness leads to sense of participation, which leads to increased and longer-term interest, which leads to more and longer-termed sales.

Now, let us think for a second how all this relates to the upcoming release of Harpoon 4.

There is every indication that H4 is going to be a superb product, crafted by a dedicated development team. Unless the developers at Ultimation are divine entities however (a thought that has entertained us occasionally ☺), there will be flaws, there will be bugs, and there will be things that people will wish they were just a little different.

Now, we have no doubt that Ultimation is committed to providing support and fixes to the product after its initial release, as they have with their previous titles. But for how long? 3 months? Six? How long until the market competition forces them to concentrate their efforts in other projects? What happens to the (patched) program then? Does it remain in the same form (remaining flaws and all) forever afterwards? Remember, the wargaming crowd is not going to turn to "the next best thing" after 6 months (unless it's really great). And like most niche groups, grognards tend to have a voice of influence well out of proportion of their market size.

This is where the question of open structure comes in. How "open" is Harpoon 4? Is most of the "meaty" stuff broken down neatly into individual-but-cooperating modules, or is everything rolled-up into a messy heap? (as reportedly was the case with the H2 code when J.Spears took over ☺). That, we don't know, as we have no knowledge of its source code whatsoever. If the structure of the program inherently discourages modifications and general tampering with part of the code or data, then the whole point is moot.

If, however, the hurdle is not in the technical domain, but in Ubisoft's belief in "keeping their IP to themselves", it might be wise to reconsider this approach.

Who gains from excessive secrecy over the code? No one. The product suffers as it grows "old", careful customers turn to other titles that offer greater openness, even the most dedicated followers rapidly lose their interest as they realise that they have no chance of ever having their program work the way they want it to. Eventually the product becomes "just another" abandonware title in a long line of products.

Who gains from extensive modability, not simply on data (textures, sounds etc.) but on the program internals themselves?

- The users, who get to have the program work their way.
- The developers, whose "to-do" list of fixes, improvements etc. shrinks dramatically as the users themselves do much of the work – and for free!
- The publisher, as sales increase both immediately and in the long term, and the title has a good chance of standing the test of time and building a strong reputation of excellence (*"From the makers of 'XXX'"*)

For these reasons, if the technical ability exists to enable post-release code modifications on Harpoon 4, we at the HHQ highly encourage Ubisoft's project manager(s) responsible for the title to provide their explicit permission for such developments. In our opinion, everyone stands to gain.

Harpoon HQ

By the Players for the Players