Interview with Kevin Klemmick – Lead Software Engineer for Falcon 4.0

March 12th, 2011
By Giorgio Bertolone

Falcon is, without any doubt, the most ambitious and realistic Air Combat Simulation ever created and, for this reason, many simmers all over the world still fly it regularly despite its aging graphics. Because of this success, I always hoped to have one day the opportunity to ask specific questions about its development.

I can't thank enough Kevin for agreeing to this interview and for answering in such a honest and professional way. This is a long interview that will reveal a lot of things that many simmers probably didn't know. I'm proud to make it available for everyone who, like me, considers Falcon as one the best simulators out there. There is a lot to read so sit comfortably and enjoy!

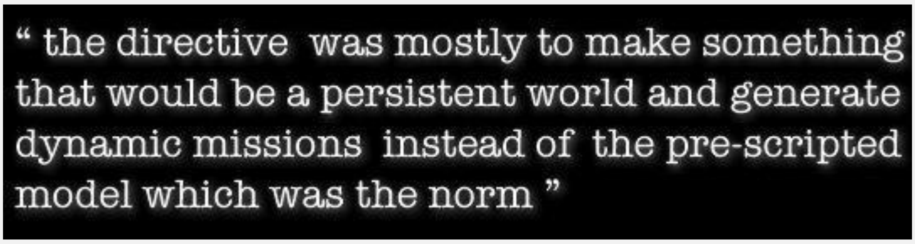
GENERAL NOTES FROM KEVIN: Keep in mind that all this happened about 15 years ago and my memory is definitely fuzzy. I pulled up what I could from there, but I will be the first to admit my memory isn't always accurate. I did no actual fact-checking on myself, so take all this with a grain of salt.

Let's start from the beginning. What can you tell us about your background and how did you find yourself working for MicroProse?

I had been studying Aerospace Engineering at Cal Poly when an opportunity came up to take an Internship at MicroProse (which was still Spectrum HoloByte at the time). Back in the 80s I had written several multiplayer games for a gaming BBS I ran in high school, and I found the job opportunity through those contacts. Because of my background with both gaming and aerospace it seemed like a good fit.

Could you describe your roles and responsibilities during those years?

Initially I was hired as an intern and asked to design and develop a dynamic campaign. For better or worse there wasn't a lot of direction on what that would entail – the directive was mostly to make something that would be a persistent world and generate dynamic missions instead of the pre-scripted model which was the norm. I'd written a few simple strategy games prior to this, so I approached it as designing and writing a strategy game. This was obviously a much bigger job than an intern could handle in a summer, so I eventually signed on full time. By the end of the project I'd ended up taking on much more, and was eventually lead programmer on the localization projects.



“ the directive was mostly to make something that would be a persistent world and generate dynamic missions instead of the pre-scripted model which was the norm ”

How many people were in the development team of Falcon 4.0

I honestly couldn't give you an exact number. Probably 50 or so over the course of the project. The thing is, the entire team turned over twice due to people leaving, layoffs or terminations. For a brief period of time I was the only programmer on the team. So, depending on how you count it this number can vary widely. For most of the project we had about 6 engineers and maybe the same number of artists.

Working on a simulator like Falcon 4.0 must have been an incredibly exciting and stressful job. How was the general atmosphere in the team?

This was my first job in the industry so I didn't really have anything to compare it to. At the time it seemed like we were really excited to build something cool, but in retrospect I realize there was a lot of stress, conflict and tension in the team. I take responsibility for a portion of that too. We all had strong opinions about what we wanted to do and there wasn't a strong management presence until the end (when Gilman Louie came in and filled this role personally), so things definitely went off the rails regularly.

Falcon's real-time Dynamic Campaign is one of the most impressive engine ever created in a sim. Could you talk specifically about its design, challenges and implementation?

I was given a pretty blank check in designing the Dynamic Campaign, so I approached it as I would a strategy game. The idea being that this game would be running in the background whether or not the player flew any missions. In fact, it could be played as a strategy game from the tool I wrote to monitor it. The AI was broken into three tiers, a strategic level, operational level and tactical level. Yet another level of AI would operate in the Simulation itself to drive the vehicles or aircraft.

The missions were generated as a byproduct of this AI, and in fact used real world planning techniques. For example, once a priority list of targets was determined, a package would be put together to time suppression of air defense, air superiority, refueling, AWACS, etc. All these missions would be timed out and planned much like a real world commander would, but were generated as a response to decisions made by the campaign's AI.

While my primary goal was to make something fun to play, we were very fortunate to get a lot of advice from military sources about how things work in the real world and I tried to match that as closely as possible while keeping the game play elements that I felt were important. However, all of this had to work within a very tiny slice of the CPU, which was a huge limitation given all the AI/planning work that was going on. That was probably the biggest challenge of this system.

" I was given a pretty blank check in designing the Dynamic Campaign, so I approached it as I would a strategy game "

How did you design and code the multiple scenarios? How did you manage to work on them without feeling overwhelmed by the immerse scale of these virtual conflicts?

We talked a lot about what theatres we wanted to use. I did some research about what was at the time thought to be the most likely future conflict zones. In the end we went with Korea because of a number of factors. I pushed hard to focus on one theatre in depth rather than do multiple theatres poorly, so we decided to do multiple scenarios in a single theater instead. The scenarios were based somewhat on historical situations in the Korean War, but also what could be likely situations given deployments at the time. The biggest problem with the scenarios wasn't feeling overwhelmed by them, it was testing them enough to feel comfortable that completely non-scripted AI would be able to play through them realistically.

What were the biggest technical problems that you had to face and solve in the other areas engineered by you (AI, Multiplayer, Coms, etc)?

The biggest technical challenge for me was doing everything I wanted to with the Dynamic Campaign in the CPU slice we budgeted, which I believe was something like 5% of the CPU. To really get AI to work well you need to do a lot of pathfinding and data crunching, all of which is CPU intensive. So there was definitely a lot of compromise in AI quality because of this.

Coms was the other big challenge I was a part of. We developed a very low cost protocol and spent a lot of time on the whole "player bubble" concept. This meaning mostly that events happening near the player were sent more often and with a higher level of detail than those far away. Outside of this bubble we updated very infrequently and with units in aggregate. For example, an entire battalion would pass a bitwise array of active vehicles, a formation and a location. All of which was just a few bytes of data.

Of course, there were plenty of other challenges as well, including simply organization of the various components. We had largely developed the various modules in isolation and when it came time to put them together this turned out to be the source of a lot of problems.

“ The biggest technical challenge for me was doing everything I wanted with the Dynamic Campaign in the CPU slice we budgeted, which I believe was something like 5% of the CPU ”

What part of your specific work on Falcon 4.0 are you most proud of and why?

Definitely the Dynamic Campaign. It’s the first and last time I was able to design and code a part of a game pretty much on my own, which had been my experience doing games as a hobby up until then. In the rest of the gaming industry you really don’t have very much input on the design of a game as a programmer. I was still pretty green at the time though and looking back I can see so much that could have been done better, but I am still quite proud of that.

Some parts of Falcon seem to have a modular design. Was this planned? Were you guys thinking ahead to future aircraft and terrain expansions?

Absolutely. In fact, we had different aircraft working in house very early on, but doing other aircraft to the level of detail we did for the F-16 just wasn’t possible given our resources. The Dynamic Campaign was initially designed to be able to be played as a separate game entirely, but in the end because very heavily intertwined with the rest of the game. Terrain sets and theatres were designed to be easily swapped out for future expansions (we had planned for an Iraq theatre). On the other hand, part of this modularization was due to different engineers working in isolation and became a problem later on. For example, three different modules ended up using 3 completely different coordinate systems, so communication between them required conversions.

It seems like the first release of Falcon 4.0 was rushed to the market in order to sell during the Christmas holidays. Was the code mature enough for this initial release?

I’d agree that the product was shipped in a pretty buggy state, but I couldn’t honestly say the first release of Falcon 4.0 was rushed. It took about 5 years to build and the last 9 months we were working 12-16 hour days (They had a hotel booked across the street, so my wife ended up staying there so that we could even see each other). It was a huge challenge to just finishing the thing; this was an incredibly complex product that really wasn’t planned out or managed well at all. Because of the complexity and lack of central design it became really difficult to find and fix the many, many bugs in the program. In the end we could have taken another year and still had open bugs, but at eventually you’ve got to get it out there. MicroProse was bleeding money at the time and Falcon already had the stigma of vaporware, so at some point we had to determine that it was good enough and then work hard on patching the problems.

“ It took about 5 years to build and the last 9 months we were working 12-16 hour days ”

You also worked as Lead Programmer for the post-release patch projects. What were your main priorities and which particular areas had to be fixed or improved?

I was actually a Lead Programmer on the localization projects, but I was involved in the patching process. The priorities there, to be perfectly honest, were to fix the problems that should have been fixed prior to shipping it. As I said, there were far more issues there than we had the resources to fix, but we tackled those that impacted the most users first and kept reducing the list. When I left I was still not happy with the stability of the game, which made it hard to leave it feeling unfinished, but I realized that to MicroProse this probably looked like a money pit.

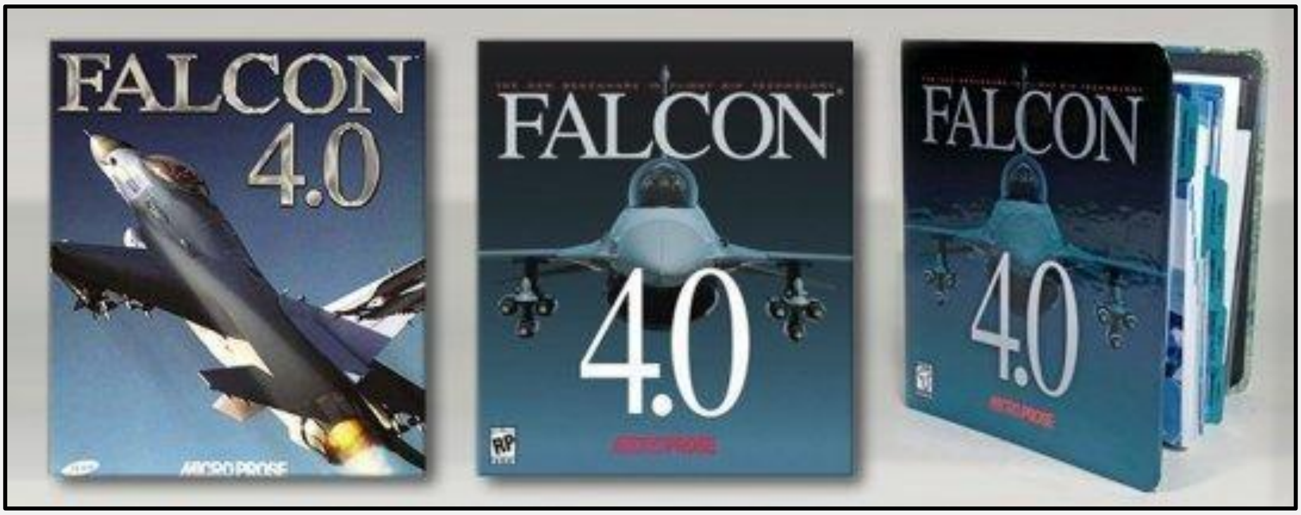
Ironically, moving to SEGA was exactly the opposite environment. We were doing arcade games which had to run with absolutely zero crashes and which are burned write only on EPROM chips. It was a completely different challenge to develop software that worked out of the box and that was unpatchable.

The last official patch was 1.08. Was there any plan for future patches after that? If yes, what would they have addressed

I had left for SEGA prior to then, so I don't know what the state of things were at that point.

Did the layoff of the entire development team come as a surprise or was it a predictable event after the acquisition of MicroProse by Hasbro?

Well, the development team had been hit with layoffs multiple times previously so the concept was pretty familiar by then. I saw the acquisition by Hasbro as a pretty negative thing, so left for SEGA prior to the layoffs. I don't think anyone was taken by surprise by that though.



Did you ever find out the cost of development of Falcon 4.0 (approximately) ?

I honestly don't know. I could make a guess given my industry knowledge but it would only be a guess. I suspect that in the end MicroProse did not make money on Falcon 4.0 however. This is not to say that flight simulators are entirely unprofitable, it's just that this one in particular had a much higher than average development cost.

I really liked the range of creative input I was allowed there. In retrospect maybe some of this wasn't so much allowed as assumed since I'd come from doing games as a hobby, but in any event it allowed me to go off and build something I thought was really cool. Unfortunately, it was exactly this approach that caused the development to take so long and coordination between engineers to be so difficult.

My worst memories are mostly of conflict between the team and the long hours. I was very young at the time (the youngest programmer on the staff) and was opinionated, overworked and had a fragile ego, so things got pretty tense at times.

Many recent simulators are released without even trying to code a Dynamic Campaign engine. Why do you think today's sim developers are so scared of what you guys were able to create more than a decade ag

Well, it's just really hard to do. Looking back on it, I think the only reason we took on what we did is because we were too inexperienced to know better. Knowing what I do now, even given my experience on Falcon, the cost to develop such an engine would be substantial. Since flight sims don't bring in that kind of revenue companies look at it from a cost to benefit standpoint and Dynamic Campaigns score pretty low in that regard. There is also the argument that scripted missions are more interesting which has some merit. I think if I were to do it over I would do a mix of scripted/generated missions, so that the player still feels like they're involved in the world, but there is also some variety thrown in to keep things interesting.

In 2000 the source code of Falcon 4.0 leaked out and after that groups of volunteers were able to make fixes and enhancements that assured the longevity of this sim. Do you see the source code leak as a good or bad event

Absolutely a good event. In fact I wish I'd known who did it so I could thank them. I honestly think this should be standard procedure for companies that decide not to continue to support a code base.

" Absolutely a good event (source code leak). In fact I wish I'd known who did it so I could thank them. I honestly think this should be standard procedure for companies that decide not to continue to support a code base "

I know that after MicroProse you moved on to other opportunities and important roles. But if asked, would you still consider working on a modern combat simulation?

I'd been approached about that a while back and expressed interest, but the team was being put together in Colorado I believe and I'm pretty tied to the San Francisco Bay Area at this point. I'm not interested from a flight sim perspective (I actually don't play them), but I would be from a Dynamic Campaign perspective. I'm much more interested in the strategy/persistent world aspect of it all.

You currently work as Technical Director for Gravity Bear and you are developing very interesting applications. Could you talk about your work on current and future projects?

I'm actually now working as a Technical Director for Electronic Arts, doing Sims projects (that "The Sims", not flight sims). Gravity Bear is a small company we created to do Facebook games, and I worked there for about 2 years. The entire company consisted of only 6 people and for much of that time I was the only programmer. I've been involved in a couple other startup attempts, largely because I would love to work on games I would actually enjoy playing and so much of what the big companies are doing these days are just remakes of old concepts, but I also have a family to support, so the stability of a company like EA and solid titles like "The Sims" is very attractive.