Postmortem: 11 Bit Studios' Anomaly Warzone Earth

PRINT

By Paweł Miechowski

The popular "inverted tower defense" game was the first created by 11 Bit Studios, a team of Polish veteran developers. Here, Paweł Miechowski, the studio's senior writer, talks about the difficulties and successes of booting up and building a game at the same time.

Anyone who has ever worked on a project in the entertainment industry -- whether it's a movie, a music album, or a game -- knows well the feeling of being proud when the moment of finalizing the project arrives. This pride is some unique mixture of the joy of finishing the work and something that parents feel when their kid has learned to walk or has been given their first A in school. Of course, parents (usually) don't celebrate this moment with barrels of beer, as often happens in the game development industry.

If this comparison is applied to the game industry model, then our kid has a teacher or a supervisor (the publisher), who has his own ideas for teaching the kid to walk. Sometimes it works fine, but sometimes the kid falls over on its own feet, much to frustration of the parents. Sometimes the teacher takes care of teaching the child how to walk as well as the parents, but later, in kindergarten, he doesn't pay much attention at all, and our kid is left on his own, not integrating with the others, and no one is interested in him.

And sometimes the teacher is flat broke or mean as a Scrooge, and he doesn't even get the kid a bus ticket to the kindergarten (well, that's what parents should do, but this is the hypothetical model, right?) Then no one will befriend with the kid, he'll fade into obscurity and end up as abandonware. While we want him to have a lot of friends and not only be given As in school, but also win competitions and awards. (Sometimes the teacher really takes care of a kid, but that's something for a different story.)

That's why one day we decided to quit the jobs we had at the time and found a new studio to be -- from that point -- completely independent. Awareness of the future difficulties or a lack thereof did not matter.

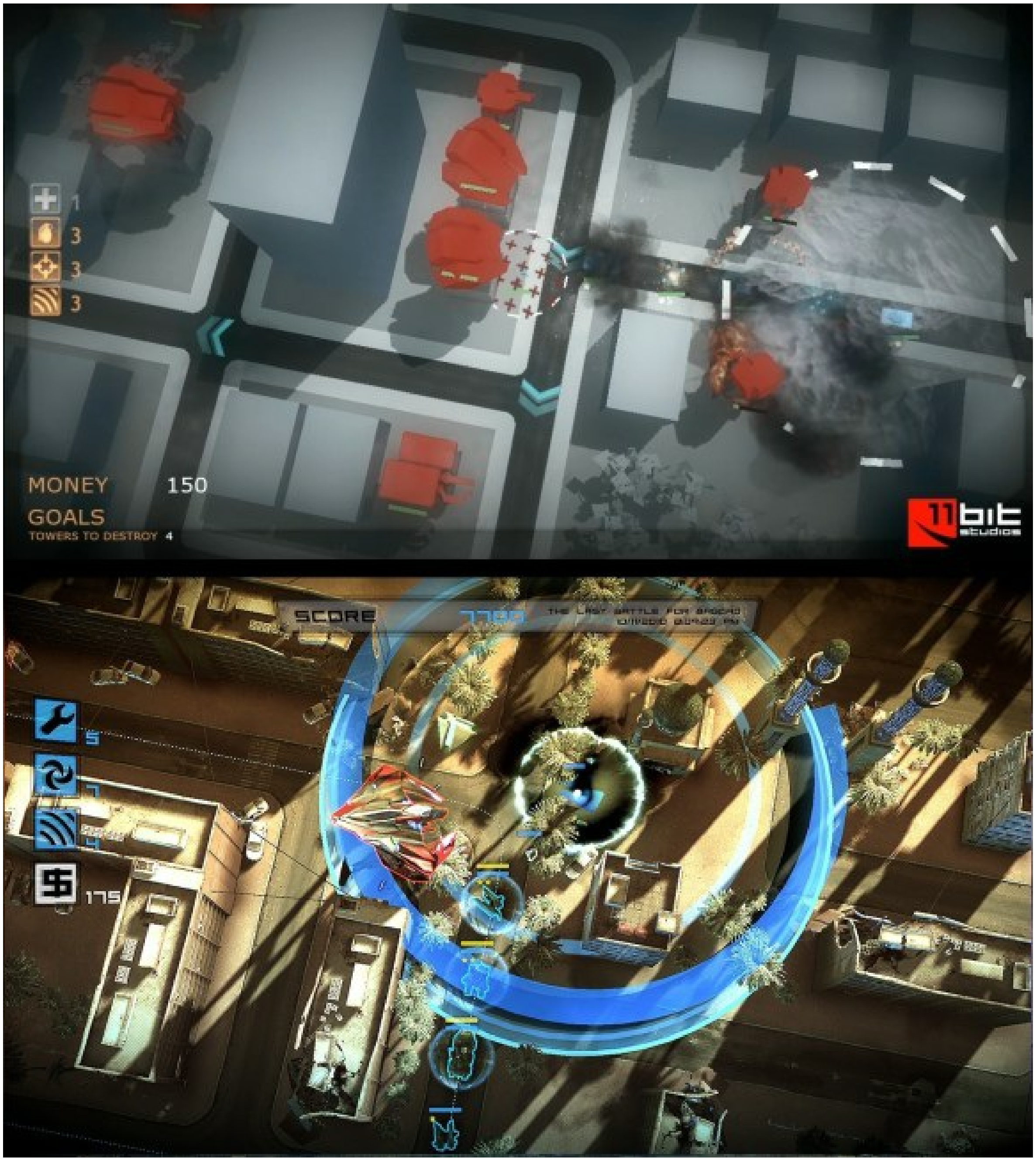
We wanted to become parents, who are the only people responsible for their kid, because we believed -- and still believe -- that no one would take care of the project better. We wanted to make absolutely personal games, completely in our own way, and use the blessed opportunity of digital distribution, where the creator is also a publisher That's how 11 Bit Studios was born.

Apart from business independence, several thoughts were fundamental when the studio was created:

- The leading theme of the first game will be gameplay. We won't make a story-driven game, because it's not our strong point. Experience prompted us to make a gameplay-driven game.

- The concept has to be original, innovative, and fresh.

- In the past we used to work pretty efficiently on our own engines, so at the start we decided to create the base fundaments for a new proprietary engine chiefly adapted to our needs.



***Anomaly Warzone Earth* (prototype, top; final game, bottom)**

What Went Right

1. Multiplatform Development on Our Own Engine

The Liquid Engine has been in development by the lead programmer Bartek Brzostek, in consultation with the designer Michal Drozdowski (back then, the only designer, now design director) and with artist Przemek Marszał (back then, the only one, now art director) since December 2009.

In April 2010, the base allowed us to build the first raw level. Fast? At the time, Bartek had over 10 years of experience in engine and advanced tools creation for building graphics and the logic of a game, so the decision to build our own engine was pretty natural -- let the guy do the engine if he's been making engines for years. He needed only time to prepare the fundaments of the new engine.

Indies are often advised to not waste time on the creation of proprietary tools and just quickly go into production on some external engine. This, however, has some serious drawbacks we wanted to avoid.

Firstly, no engine supports all platforms. For us, it was extremely important to port the game to as many of them as possible to enable multiple revenue streams. Secondly, using a third-party engine provides you with certain tools and features but no more. If you need something really custom, you're screwed; the engine developer won't deliver custom features for a small indie startup. Sometimes there is an option to acquire the source code and develop these features on your own; however, in most cases, it's not free. In essence, by using an engine out of the box, you give up a lot of flexibility and involve third-party risk. Most startups have no real alternative to this. We, however, did.

From the outset, our team was very strong in the field of technology (frankly speaking, it was very strong in all areas). We had a lead programmer with deep knowledge of PC and Xbox 360, including both high-level engine design and down-to-the-metal optimization. We had one of the best Polish GPU programmers with PlayStation 3 experience as a contractor. We had an extremely experienced gameplay programmer. And all those people had worked together for a couple of years in the past. This was a real asset, and we decided to use it to our advantage.

We limited the feature and tool lists to the absolutely necessary -- locations created out of XSI prefabs, top-down view, game logic created in Lua scripts, single player only, with a light pre-pass renderer for the sake of simplicity.

Everything that remained, however, we implemented at a state of the art level. We implemented a scene preprocessor to combine prefabs and optimize draw call count. We developed a smart multithreading architecture with one thread designed to do barely anything but feed GPU with data, and the other one processing game logic and nothing more -- the two most CPU-heavy tasks.

We optimized our pre-pass renderer extensively, fighting for the last shader instructions and GPU memory bandwidth. We implemented our math library with the use of vector units of all CPUs we supported. We designed a nice resource system with overlapped data loading, decompression, uploading to the GPU memory, and dealing with console memory fragmentation. We implemented a proprietary RTTI system to deal with serialization and developed a localization system with full support for Unicode.

The key to quick and efficient development was to enable gameplay programmers to work on the actual game almost from the start. So Lua integration came online as soon as the engine was capable of displaying a simple wireframe mesh. This way, while the engine was being developed, the game prototyping went on.

In terms of tools, we went with level editor, material editor, particle editor, sfx editor and localization editor -- mostly the tools supporting the visual side of the project. What we decided to skip were gameplay-related tools. So all game logic, cutscenes, UI flow was programmed in Lua scripts. It was enough to create a good game; however, it was a bit tiresome at the end. When the project went gold, we decided to work on additional tools to make gameplay flow creation more efficient.

Looking back, our decision to develop our own engine was 100 percent right. When we wanted to add PlayStation support, we could. iOS, Android, Linux, Blackberry -- same story. Sure, it was a lot of work. Each platform was a challenge. However it was also extremely rewarding. And it left us in total control. If one day we decide to develop *Anomaly Warzone Earth* for a mind-controlled refrigerator -- we will be able to.

---

2. Organized Production (No Crunch!)

Knowing from the past that chasing milestones could leave us with heavy crunch time, which was something that no one wanted at all, we made a production plan with solid buffers after each milestone, just in case we would slip.

Believe it or not, right up to the moment of release, we ran into only one crunch period, when preparing a master build for Steam; in all of 2011, there were only a few days of crunch time.

However, these buffers were not the only factors that made the plan work. Firstly, we were experienced. If we decided to implement a multi-threaded resource system or local contrast post-processing shader, we didn't have to research the area. We just knew how to do it -- and the same about graphics and design. We knew exactly what assets are needed to make the game look good, and how to proceed with gameplay iterations.

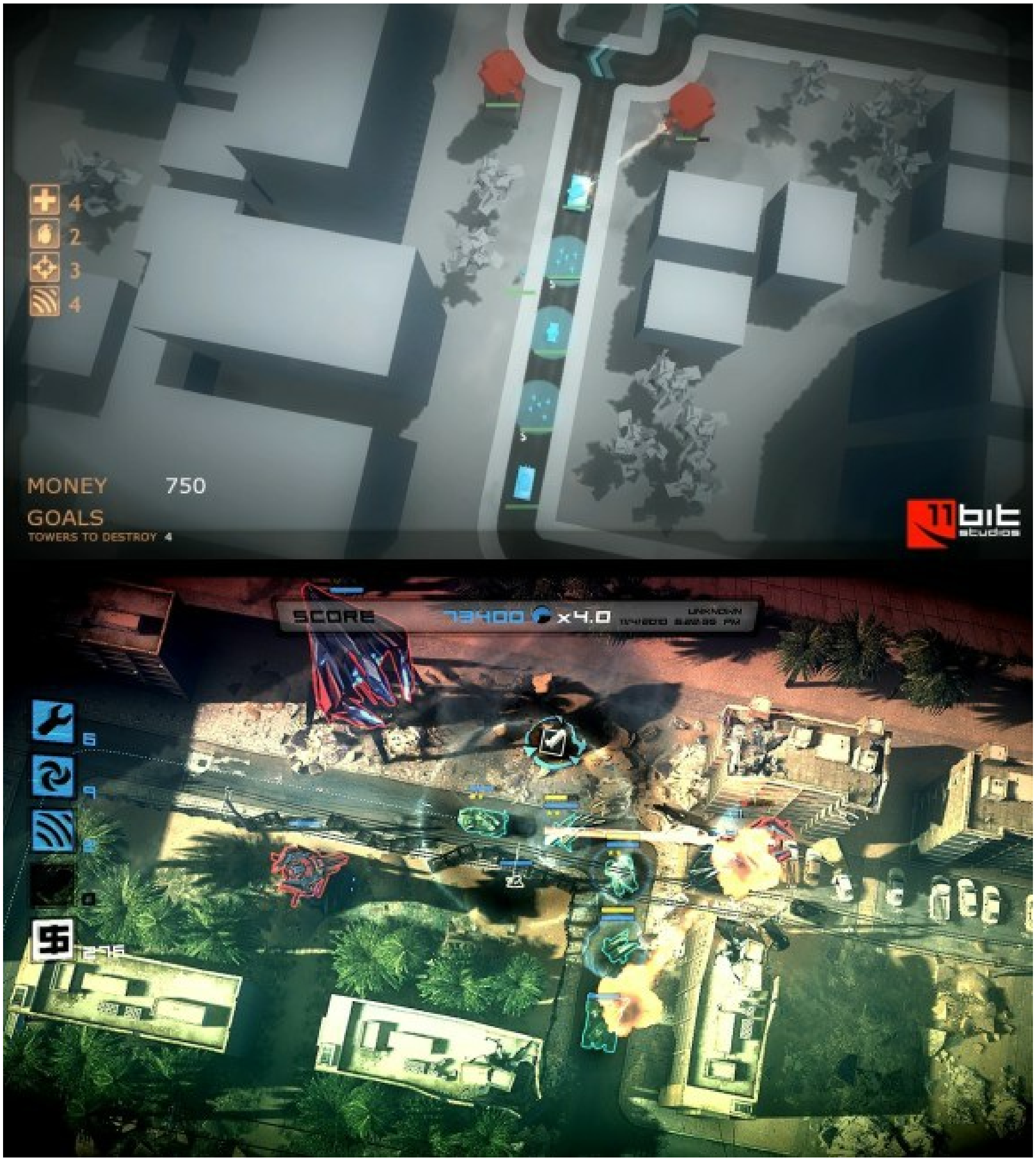
Secondly, the core team had worked together in the past -- some people for more than 10 years. We've already been through good and bad together. We were a real team.

Thirdly, it was the first time we worked without a publisher -- and the first time we didn't have to pretend we created a triple-A title to grab publisher's attention. It brought a lot of positive energy to the team. We felt we created the game we liked, the way we liked.

Fourthly, we were aware that we had no money for delays, or to create a second game. We had just one shot. The game had to be made on time, had to be good, and had to sell -- so we stayed fully focused. No business meetings, no publisher requests, no other options: just the game and the plan. Vertical slices for presentations to possible publishers eat a lot of your time. Think about what you really need, and what kind of development should consume your time in the end.

Fifthly, our plan was a good one. We avoided feature creep, and we scheduled enough iterations. We had a full time pessimist onboard to question every detail of the plan until it was rock solid. If you are planning buffers, double them, and get the worst pessimist to look at the plan and then add more buffers to make that pessimist satisfied.

Despite that, we were not perfect here -- initially the game was supposed to have 21 levels and two additional modes. Eventually, at an early stage, we deleted seven levels, and thanks to this, the work went on quite well. Making games is cool, but not for 15 hours a day!



***Anomaly Warzone Earth* (prototype, top; final game, bottom)**

3. Cooperation with PR

It's obvious that in order to get press attention for an indie production, you need to have a good game on hand. But having the good game, you need to know how to create some buzz around it and present it to people. Word of mouth is one thing, but visibility in game press is the other -- and the second thing can surely enlarge the first one. To this end, we decided to work with Evolve PR, because we're friends with director Tom Ohle; we had a couple of beers together, and we know he's a decent guy and does his work well.

We honestly don't know if using external PR is the right choice for every developer (certainly, some prefer to do PR on their own), but it surely was good for *Anomaly Warzone Earth*. To a large extent, the solid visibility of the game was achieved thanks to good cooperation between 11 Bit Studios and Evolve PR. We were too novice on the promotional side to do this all on our own.

When journalists saw the game, it caught their attention -- and this is surely due to the game itself. But the fact that it reached gaming press to a wide extent was mostly due to Tom and his people. I guess it's not common to point out a good PR in a postmortem, but, well, here we are.

Thanks to our external PR agency we didn't have to handle communications by ourselves. At the time the team was pretty small, and it actually takes a lot of time to spread the word about the game. Additionally, we didn't have any contacts at major outlets, so our emails would have landed in the spam box. Tom has those contacts, so he helped to reach a lot of outlets, including major media from America and Europe.

The game got a lot of reviews, gaining the coverage necessary to inform the gaming community that *Anomaly* is out there. If you're a newbie developer producing a hardcore game, I'd recommend a good PR agency to help to spread the word. If you are creating a casual game, this is pretty different, and I do not know a good solution.

4. Gameplay Innovation

Since the beginning, our core idea was to make an original game offering gameplay never seen before. Our design director -- who's pretty much addicted to strategic, tactical, and card games -- came up with the idea of reversing the popular tower defense genre and making out of it not only a strategy game, but one also heavily packed with action.

Development of this concept brought us, finally, to a game with original gameplay based on unique mechanics, which at the same time was a cohesive experience with ample tactical possibilities (as a strategy game) and the dynamic gameplay you see in an action game.

At the very start, we had one goal -- to reverse tower defense. With such a goal, Michal (design director) and Przemek (art director) created a few initial ideas about how the game could look. Those ideas were then presented to the rest o the team and heavily discussed. The team's questions and doubts allowed us to find quickly the holes in design and kill those concepts that did not arouse any enthusiasm. Thanks to this, they were able to create the general game scheme in a fast and efficient way and then quickly go to prototyping.

Before programming took off, we knew that the player would control a squad made of a few units, would have free choice in choosing their path, and move through a maze of streets defended by towers. With these general assumptions and with our base mechanics written down, we started to build the prototype. We wanted to build a dynamic game in which arcade elements would be as important as tactical ones, so testing a playable version was key.

Thanks to this, we could precisely analyze the feel and improve core gameplay ad hoc. We created a few versions with placeholder graphics one by one, modifying each subsequent prototype with new solutions. In the following iterations the Tactical View was added (that's where player sets a path for the troops), and then the on-field Commander (a little dude controlled by a player to run around the battlefield and use area-of-effect abilities), and then the possibility to get the abilities after destroying towers. During this part of the process, we created general rules for map size, level length, and tower density. There's a fact worth mentioning: from the very start, we were testing the game playing with both mouse and gamepad to ensure a proper user experience.

Becuase *Anomaly* is an inverted tower defense game, we played a lot of the genre -- both classics and twists on it. One major inspiration was surely the amazing *Defense Grid*. I need to note that we have played some simplified inverted TD games (mostly Flash games available via browsers), however none of them fully utilizes the idea of tower offense gameplay; in *Anomaly*, we have the hero, his so-called special abilities, and RTS elements -- like resource gathering, and management to buy more troops or upgrade current ones.

Later, in reviews and user feedback, this emerged as one of the key values of the game; the feedback underlined the freshness of the game -- "unlike anything I've ever played before." It's worth noting that the lead designer was determined and consistent in the implementation of his vision. Conclusion: innovation pays off (in this context, gameplay, but I'd suggest that any innovation will).

There's something I was asked at a game dev event in Kraców, Poland: "How would you recommend to work on new gameplay ideas?" What I said was: "Get your favorite game genre and flip it over, or add an unexpected element. Say you love FPS games. Get your hero's weapon and change it into magic wand, so you can cast spells on enemies to make them fly instead of being shot down." For example -- look at one of my favorite games, *Puzzle Quest*. They took the match-3 idea and added lots of RPG elements. It's an unexpected, weird connection, yet a genius one and brilliantly executed.

---

5. Quality of Execution

Another thing we put a lot of focus on was the quality of execution. We focused on high quality visuals, and went through a lot of iteration. If you make a game, it's worth iterating extensively, because you not only notice bugs or shortcomings, but you'll also do something colloquially called "polishing." Of course, it requires time and work, but in this industry -- in our opinion -- polishing is absolutely necessary. As a result, in users' and critics' feedback, there was often mention of the game being "incredibly polished" or even "better polished than some triple-A titles." The iOS version reflects that clearly -- its Metacritic score is 94!

Until a game is a real breakthrough, with one particular feature bringing everyone to their knees, the quality perceived by the player is often defined by the quality of its weakest component. A game with beautiful graphics crashing every three minutes, or a super smooth game with crappy audio, won't be perceived a quality title. Because it's risky to rely on creating a genius game, if you're hungry for good reviews, you have to include a constant fight for quality as part of your development process.

The first step in assuring the quality was scoping the project correctly; for example, we didn't include multiplayer because we didn't feel confident we could do a good job with the time and resources available to us. We decided to do less, but make it better.

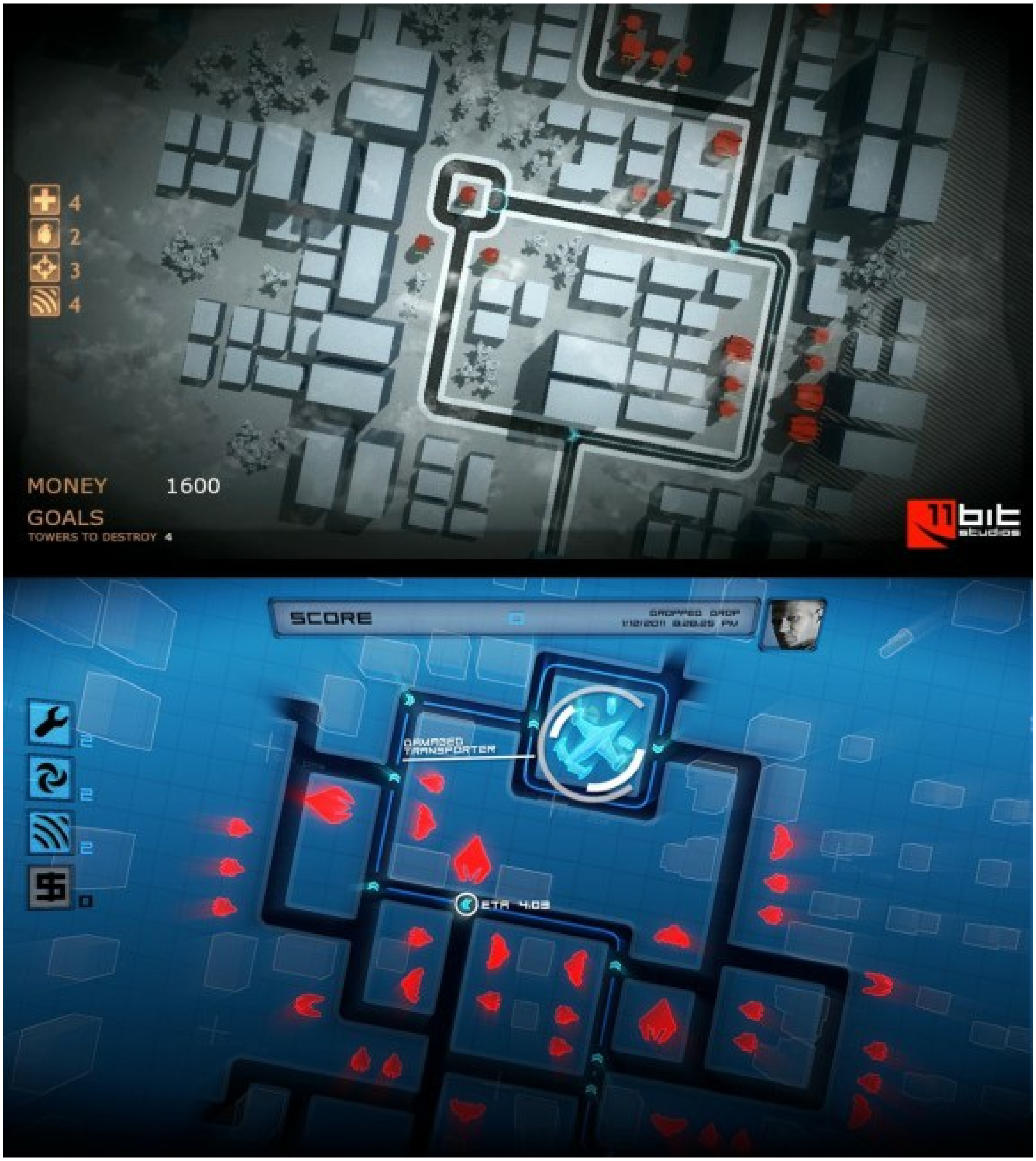
The second was having the right art director. Let's face it; a quality title needs a high quality, coherent visual style. A good art director can develop that style and lead the team to follow it. This is exactly what our art director did.

Gameplay-wise, it takes a lot of prototyping and playtesting. When we started an "inverted tower defense" project, we created the first prototype. It was boring. So was the second. The third one did the trick. Same with each of the gameplay mechanics: they were prototyped, played by us, and then by testers. We judged them by our feelings and our observations of others playing, and then decided what to do with them. If a mechanic was okay, we made it good; if it was good, we made it even better. If it was ugly, we threw it away.

So we iterated. A lot. And we constantly improved. Plus we paid a lot of attention to the details. Everyone strived to improve the quality of his area of the game. The programmers fought for the last frame per second, the artists for UI to show up nicely and smoothly, the audio engineer to balance the sound levels, and so on. We made tons of small touches here and there to make the game look expensive and feel pro. If you don't do it, expect feedback saying your stuff is "undercooked." It sounds obvious, but some indies don't do that.

Additionally...

In the early startup phase, our capabilities were limited. Luckily, we knew a few skilled programmers and graphic artists we'd worked with in the past. We outsourced different parts of the code and graphical assets to those people, which allowed the core team to focus on the big picture of the game. In this case, the outsource model worked really well, because we knew our external contractors well. From our experience, outsourcing is very risky when you don't personally know the contractors.



***Anomaly Warzone Earth* (prototype, top; final game, bottom)**

What Went Wrong

1. Dialogue

The script was extremely simple. This is a gameplay-driven game, and the story serves mainly as a natural initiation for new targets and missions. And that worked well -- no one complained about the story itself, despite the fact that it was based on that ancient theme of "Aliens invading Earth." Nonetheless, too often we encountered opinions about the audio dialogues being "cheesy but tolerable."

What's interesting is that the complaints came from the UK (the actors and our proofreader are British), while in U.S. and the rest of the world, we saw the opposite opinion, as some gamers and critics expressed their approval of the British voice acting.

Anyway, this is one thing that didn't go too well. The story may be simple, but the dialogue needed more attention, so it would sound more natural, especially considering the fact that we're from Poland and none of us is a native English speaker. Conclusion: not enough testing of dialogue, not enough iterations, not enough work to make them sound natural.

2. Additional Modes for PC and Mac

Most of the players were interested only in the story campaign mode. Steam stats show that only about 15 percent of players launched additional modes (like our Squad Assault mode, which is something like a survival mode). It means that most players got what they wanted out of the main campaign and didn't want to jump back into the game quickly.

What's interesting here is the stats are pretty different for the iOS version. The additional mode in that version had more-or-less similar stats, but another that we added sometime later as a free update brought a bigger number of primary users, and again increased average daily sales. Conclusion: a well-done base campaign is enough of a satisfying experience for a player. The player is willing to jump back into the game, but only after some time -- when he or she is offered new content.

3. No Updates to the PC / Mac Version

This point follows on from the previous one. By launching additional modes with the first release, we ran out of content that could have been added to the game later. Being busy with other platforms, we didn't have manpower and time to prepare updates with new missions for PC and Mac (additional content was brought to Steam during the Holiday Sale).

For this reason, the interest in *Anomaly* on PC rose when the game was on sale on Steam, while our experience with the iOS version shows that it could have bounced back again when new content was delivered.

New content doesn't have to be a new level; for example, it could be enhanced graphics for iPad 2 and iPhone 4S (both devices utilize the A5 chip, which allowed us to render much more demanding effects and animations for 3D objects) or enabling iCloud features, which increased sales on the App Store, too.

---

4. PC Version Balance (a.k.a. "Bloody Difficult Mission 11")

For some reason -- no one wants to take responsibility for this decision today -- we changed the initial version of the 11th level, feeling that it was too easy for this part of the game. The entire story campaign mode has 14 levels, and the difficulty curve initially was kept correct -- the last two levels were the biggest challenges, and in the last one, of course, the toughest was the mega-boss.

In the 11th mission, we introduce a new enemy -- the Energizer -- which is capable of respawning destroyed enemies. So it's a pretty tough enemy, but one is able to beat it fairly easily by bombarding with an airstrike. Beating this mission with airstrikes is still properly challenging for a later level.

And suddenly -- oops! -- we took airstrike from the player, and Mission 11 become extremely difficult. We didn't notice that, because we knew the game mechanics so well that we knew how to beat this mission easily, and we thought that the 11th level should be more difficult to maintain a proper difficulty curve.

Players, on the other hand, had problems with it. Some stopped playing on that level, because they could not beat it. A part of those who finished entire game have admitted that the 11th level was the toughest. It's a solid mistake, perhaps caused by the fact that the final tests were done by people who knew the game too well. The conclusion is simple -- until the very end, you need to involve some people in testing who are completely fresh and new to the game.

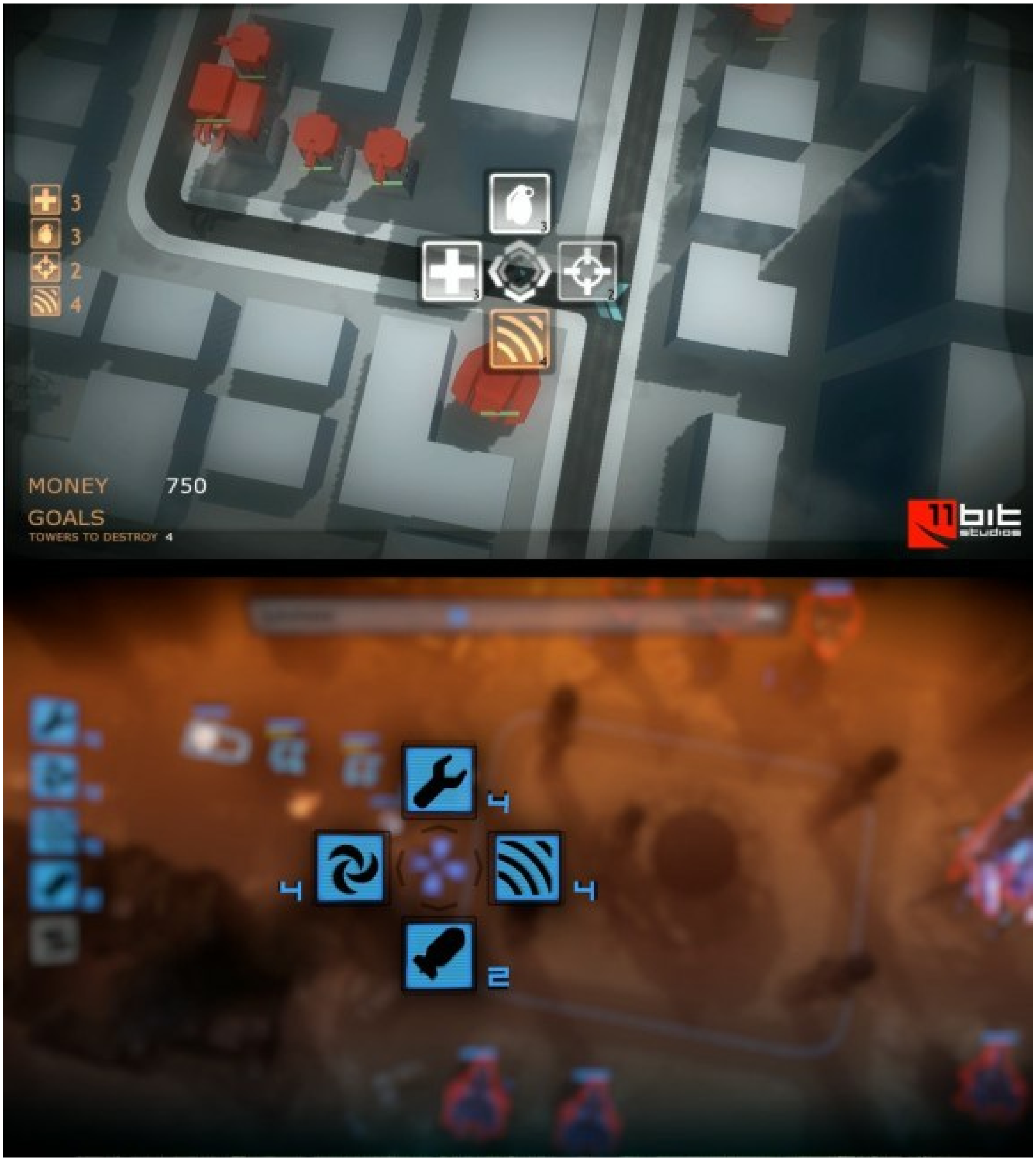
5. Bad Start on Google Play

We totally didn't have any experience, nor were we properly prepared to launch a game on Android Market (now Google Play). We learnt everything from scratch on the back of our own mistakes. It began with a not-so-good toolset - we used Visual Studios and the VS_Android plugin. Unfortunately, we could not conjure up a debugger. Eventually we used the console log.

Secondly, many devices heavily utilized bugged graphics drivers. Overcoming those bugs required blind guessing. Unfortunately there is no any easy way to communicate to a user to install the latest drivers. Thirdly, it's impossible to test the game on all possible devices. There are over 1,000 listed on Google Play. Quite often -- and only from players -- we received feedback about problems on various devices. We don't have any idea about testing these devices other than to just buy the most popular ones and learn through feedback from users whether bugs disappear on their devices too.

Furthermore, we had isseus with the limitation for APK size. *Anomaly* consists of about 150MB of data. The market limits APK size to 50MB. To solve this, one has to arrange hosting of additional data. This approach completely kills the point of refunding money to user within 15 minutes if they're not satisfied with the application. Within 15 minutes, many people were unable to download all the game data, especially when they connect to the internet via mobile network.

Finally, an interesting fact -- Google Play did not allow you to sell apps from Poland. In Poland, there are several hundred developers of apps for mobile devices. We had to get around this by cooperating with a foreign company. However, a little later, the store expanded to allow apps from Poland and the Czech Republic.



Anomaly Warzone Earth **(prototype, top; final game, bottom)**

Conclusion

Despite numerous iterations, we couldn't avoid mistakes in balancing and testing. The gameplay came out fantastically, but the story and dialogue could have been better. In future projects, we're going to put more attention toward these elements.

At an early stage, we quickly managed to create our own engine, which we constantly improved with new tools. Now we have our own toolset adjusted to our needs and to multiplatform development. We expect to have slightly faster tempo for future projects; however, the "no crunch time" rule is still in force!

Anomaly Warzone Earth did a good job on PC and Android (10,000 downloads were reached in about a week and 50,000 in a bit more than one month). On Mac and iOS it did terrific. The Linux version was launched with a Humble

Bundle, which sold 150,000 bundles.

The Xbox version, though, had a serious problem at the start. During first two weeks, the game page on XBLA was missing trailers, screenshots, and artwork -- almost everything except the description. So if anyone was looking for the game to buy and download, they were able to see only the text and nothing else. We and the team on Microsoft side struggled with this system bug for two weeks before it was fixed; the initial sales was rather bad, but in the end the console version paid off -- however things didn't go as well as the PC or mobile version. For us, the console market is the toughest one, and we rather recommend hitting Steam or Google Play unless you already have a *Minecraft*-size brand on your hand.

Let me sum it up. We created a new studio, in which the experience of veterans of the Polish game industry, and the creativity of new members gave birth to a game that gained respect from young strategy fans and old hardcore gamers as well. Often it was praised for its uniqueness of gameplay, originality of the concept and high level of polish. Now, we want to keep this level in future games -- gamers expect this from us, and we expect this from ourselves!

Return to the full version of this article