


# Postmortem: Pinball-RPG hybrid Rollers of the Realm

---

 [gamasutra.com/view/feature/233340/postmortem\\_pinballrpg\\_hybrid\\_.php](http://gamasutra.com/view/feature/233340/postmortem_pinballrpg_hybrid_.php)

PRINT 

By Sean Thompson, Tony Walsh, Ericka Evans, David Evans



Phantom Compass was founded back in 2008 as a service company developing branded games for kids TV producers. Our first four years were highly productive, but we couldn't grow the company on the basis of low-budget Flash games.

In late 2011, a local triple-A studio laid off dozens of staff. Many of them had years of industry experience, but few or no shipped titles and needed to build their portfolios before finding work. Phantom Compass founder, Tony Walsh and production head, Ericka Evans, saw an opportunity in this and brought some of the best industry vets into a meeting with our tiny indie team.

The triple-A side came to us with high production values and experience with shipping large titles. On the indie side, we had rapid development know-how, agility, and experience running distributed teams (Phantom Compass operates out of Toronto and nearby St. Catharines). Our mission was to bring both sides together in creating a small game for Xbox Live Indie Games -- a six-month project where we could learn from each other, provide our triple-A friends with a unique portfolio piece, and shift Phantom Compass into high gear.

We took a big risk in combining two distinct teams and fittingly chose an unlikely mash-up of pinball and role-playing games. What we ended up with three years, two funding rounds, and nearly 50 extended team members later exceeded everyone's expectations. Here are the high scores and gutter balls of that adventure.

## What Went Right:

### 1. Being a Bit Weird

The original idea for *Rollers of the Realm* came out of a group brainstorming meeting between Phantom Compass and our laid-off triple-A friends. We needed an idea that was easy to get behind and small enough in scope to produce over 3-6 months. Our friend, Tom Beirnes, brought up *Pinball Quest* as something that hadn't really been done in a while. *Pinball Quest* was an RPG pinball game for the NES in the early '90s. "RPG + Pinball" instantly excited us.



### ***The early XNA version of the game***

This offbeat high concept opened a lot of doors. We managed to pull together a 2D XNA vertical slice prototype of the game in time for GDC 2012 and got ready to show our weird little game to the world. We set up meetings with potential publishers and barked "RPG pinball" to any passers by with a press badge. Almost all were immediately intrigued by the concept, yet couldn't wrap their heads around how it would work. This turned out to be a great recipe to start dialogue and get to know publishers, press, and other like-minded devs.

Bolstered by the interest and encouragement from that first GDC showing, we decided to pursue production funding from two major Canadian funds, the CMF Experimental program and the OMDL Interactive Digital Media Fund. Both funders value innovation: being weird paid off, and we received production funding to turn our small 2D portfolio project into our first major 3D game.

## **2. Quirky, Appealing Characters**



Once we caught people's attention with "RPG pinball," our characters sustained interest. The game narrative was purposely kept simple and familiar to place focus on our odd array of misfit playable characters. Our pinball heroes were pulled from the margins: a young girl, an old drunk, an elderly woman, a mother, an Arab-esque fighter, an African-style mystic.

The playable characters were given a lot of attention in production and we think they came together nicely on all levels. The number one comment we received when demoing Rollers was "I love the character art!" The 2D art was supplied by a talented group of young contractors who were willing and able to take the time to develop interesting characters fitting the quirky nature of the game.

We were equally lucky with our casting call for VO. Some very experienced voice actors were able to take on multiple parts and some naturally talented non-actors filled in the ethnic backgrounds we needed. This required extra time in casting and recording but it was worth it. We listened to over 200 voice submissions before finding Maya Wolosyn, who stars as our tough but lovable Rogue. Our game director was a film director in a past life, so he was able to get great performances out of everybody... except himself. He recast himself.

### 3. Making Sure the Game was Fun

Is it fun? Yes, it is. How do we know? We tested it. The core mechanics (namely pinball combat) proved to be universally fun to those attracted to the game. This was evident from our first prototypes all the way to the final product. Even some of our harshest critics don't deny that the core gameplay is fun.

With an odd genre blend to prove out, we engaged in public playtesting earlier and more often than we had with other products. We hosted public "Test Fests" at our office, took the game to festivals, conferences, demo nights, trade shows and competitions. Consequently, we had hundreds of testers from ages 3 to 83 play the game throughout production.

We discovered that people were drawn to the game for different reasons which gave us some clues as to who our core audience was or wasn't -- some were attracted to art style or high concept, some were pinball fans, some were RPG fans, and some were both.





### ***The final game***

Once they were drawn in, they would often spend 20-30+ minutes with us and the game before moving on -- this gave us invaluable feedback that we able to incorporate into the game. Watching a stranger play your work in progress can point out your design mistakes pretty quickly. There were times where we thought a playfield design was worked out or the instructions were clear, and then we would demo it and watch 20 strangers in a row struggle with same issues. This process was so useful that we will be keeping it up with all of our future projects.

In hindsight, we did however get stuck with a public testing problem. As simple as it sounds, *Rollers of the Realm* has a bit of learning curve and 3/4 of Chapter 1 is the tutorial. It also has a linear story that we didn't want to totally spoil for potential consumers, so we most often only demoed the first two chapters of the game.

Full game playtesting was limited to our small group of Closed Beta Testers, our development team and only one of our hosted "Test Fest" events. Our small group of closed Beta testers played remotely from their homes and provided great written feedback, but we weren't able to benefit from the "watching as you play" experience that live demoing gave us. Consequently later chapters didn't benefit as much from playtesting as the first two chapters. In the future we will consider having our Closed Beta Testers to privately stream their testing sessions with us.

## **4. Choosing Unity 3D for Development**

When we first started work on our portfolio prototype in the fall of 2011, Phantom Compass was just considering switching from being primarily a Flash shop to Unity 3D shop. Although we were well versed in C#, we had only been experimenting with Unity and our triple-A friends had no experience with it at all.

Our target platform for Rollers at the time was Xbox Live Community Games -- with the stretch goal of hitting Xbox Live Arcade if we could polish things up. Our timeline was only 3-6 months. Everyone involved was already familiar with XNA, which meant we could get up and running faster, so we moved forward with a 2D XNA plan. Our artists were primarily 3D artists, and planned to model the environments and characters in 3D and then paint them over to make our textures and sprites.

In the fall of 2012, we got production funding from CMF and OMDC in Canada and part of our vision for a larger, more robust game was adding the third "D" to Rollers. We scrapped our 2D XNA prototype and started coding it from scratch in C# for Unity 3D. With a few months of actual development experience with Unity under our belts,

our new goals were to focus on PC development, with an eye towards Steam and when (and if) the time came, we would take advantage of Unity's multi-platform publishing abilities to take the game further platforms.

Unity 3D was easy to pick up, even for the designers and artists who had no experience with it, and we were able to rapidly re-prototype the game and go into full production quickly. We were able to reuse some of the 3D environment and character assets that we had created for the original prototype to give us a head start on asset creation. Our designers were able to remap most of our 2D playfield designs in 3D space, playing with Unity's built-in physics engine.

Unity's large, supportive, and active community definitely made development easier. Help on any question we had nearly always proved to be just a forum search away. The asset store saved us time/money more than once, with solutions that would have taken us days or weeks to produce on our own. An asset store rock looks just as nice and works just as well as a homemade rock, for example. We did however get burned more than once by paying for a solution instead of investing in a custom one. iTweens, for example, didn't play nice with the PS Vita.

When Atlus approached us to bring the game to PS4 and PS Vita, Unity 3D's multi-platform publishing was a great selling point. This however was tempered in practice by a few points that might better fit in the "what went wrong" section. The closer we got to this cutting edge of Unity's tech, the harder it was to find the support we needed. There was great documentation and lively forums for PS Vita, but apparently there just aren't that many Unity devs who have developed games for PS4, so that lovely big community support network we were used to got a whole lot smaller. We suspect *Rollers* was the first Unity game to have cross-buy and cross-save for PS4 and PS Vita (either that or the other devs were particularly quiet about it).

## 5. Taking a Big Risk to make a Big Step Forward

When we first conceived of the relationship between our teeny four-man indie studio and 20+ laid-off/between-jobs developers, we knew it would be challenging and risky on many levels. To get the prototype started we had more than quadruple the company size and onboard people who were used to a traditional office environment into our distributed office environment -- communicating primarily through Skype, Gmail, Google Docs, Dropbox, SVN and Assembla, our cloud-based code and task tracking software.

We had to be very accommodating around hours of work and time commitments and be ready for it all to change, as people rolled off of the project when they found full-time work. We had to be agile and resourceful to keep the game moving forward even though the team behind it shifted constantly.

*Rollers of the Realm* also tackled a risky design challenge -- taking on a relatively unproven, unusual genre-blend was a risk we took on because we just loved the idea so much. We were very fortunate that our creative director, Dave Evans, was a constant. He championed and drove the project from day one, which definitely took some pain out of those early days.

So what came out of that risk? So much goodness. We have built great working relationships with experienced and talented developers. We've grown our company to a small but mighty 10 employees. We've been nominated and even won awards, got Greenlit on Steam, and negotiated our company's first international publishing deal. We've shipped our first console title. We now have super-fans who love our game (and we love them right back!) All huge steps for an unknown indie -- *Rollers of the Realm* has taken us further than we dreamed.

---

## What Went Wrong

### 1. Systems Failure

As it was originally conceived, *Rollers* was a more straightforward arcade-style game, but we veered from that path pretty early on as we dreamt about a larger-scope project. We experimented and added features and unique scripts one at a time, playing with them and throwing out ones that weren't as fun.

We ended up with a wide variety of character stats and abilities that sat on the separate character objects. When we got production funding, we planned to beef up stats and abilities, but never took a step back to look at what we had -- we were planning to add and work them all into a cohesive externalized system. This made design, implementation, testing and balance more difficult, but we only realized our mistake when it was too late to change it.

This even haunted our UI process -- it made our Port (where you hire characters and upgrade them) UI/UX a challenge, and we went through several major versions. Our final version is the best of the bunch, but we're still not 100 percent happy with it.

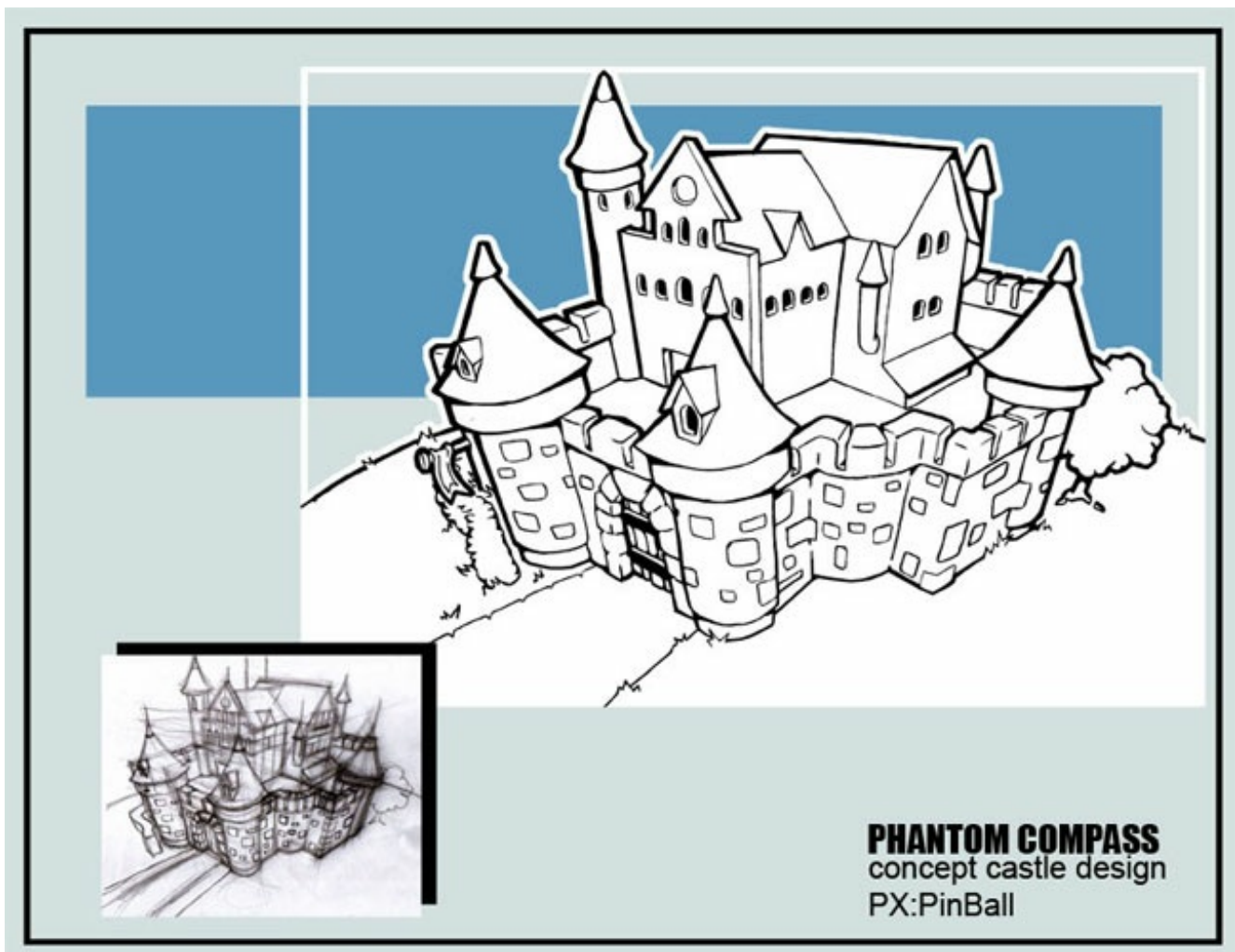
Most of the game's major systems aren't wired up to external data sources, and are pretty hard-coded, so we were never able to implement a difficulty system or tweak game balance remotely. Future games won't get past prototyping without a nice neat external data system.

## 2. Optimization Underestimation

When we agreed to add the PS Vita as a launch platform, no one on the team had ever developed for the PS Vita before and we vastly overestimated its abilities. We originally estimated that 30 percent of our time in the last stretch would go to Vita-specific optimization. The reality was that about 80 percent of our time went to optimizations, from graphics to physics and other systems. This was even after we transferred extra resources onto the project to help.

Choosing to develop with Unity, while great for so many reasons, also hurt here... because we didn't have access to the Unity source code, this meant that when we needed to comb through the game looking for deeper and deeper optimizations, Unity itself was a black box.

As a result, we missed the mark on polish (all platforms) and Sony hardware-specific features. We didn't end up having time to fully exploit the capabilities of each platform, so our PC and PS4 experiences are not as visually robust as we would have liked (we're using Vita-friendly shaders and effects on all platforms), even if the performance is excellent due to extreme optimization.



### 3. Audience and Balance

We took a big risk in combining two very different game forms. We didn't expect to satisfy hardcore pinball or hardcore RPG fans -- we figured the game would appeal to mid-core gamers. Based on early reviews, opinion differs greatly (we've got superfans, haters, and everything in between). We're not sure if trying to find a new audience with this mash-up was the best way to go for our first major title.

As mentioned in our "what went right" section, our playtesting provided valuable insight for balancing and audience appeal, but the testing was mostly limited to the first few chapters. We misjudged the game's difficulty curve and the willingness of players to solve some of the game's strategy puzzles or manage party members and their upgrades.

On its face, the game appears to be similar enough to pinball -- many people believe anyone can play pinball, and this is true -- but mastery is another story. Our game doesn't look like it requires a lot of skill, but it does.

Our final boss battle was, for most players, punishingly hard. To add insult to injury, failing the last boss in the series could cost about 30 minutes of time and a replay of the entire battle. This left a sour taste for many players and reviewers. We added checkpoints in the final battle in an early patch for Steam and hope to bring this and other improvements to the Sony platforms soon.

Our "System Failure" didn't make it easy on ourselves to tune the overall balance, let alone do it remotely (via external data, which would have allowed us to nerf a few things without patching). We won't be making this mistake in future projects (this is something we've already institutionalized).

### 4. Too Much Too Soon

We were eager to move from our original 6-month prototype into full production -- maybe too eager. We front-



loaded resources to try to overcome our potential Unity learning curve, but everyone picked up Unity so quickly we ended up with a bottleneck in programming. Our designers were waiting to port over workable ideas to the Unity project from our XNA prototype and our creatives had to wait to test content before the basic framework was in place.

We pushed ahead with design before the core set of scripts and combat system was complete, which contributed later to awkward workarounds (both in design and programming) that were difficult to QA -- many of these were redone from scratch when the system was completed, which ultimately was a waste. Obviously some better production planning and resourcing could have helped here.

Throughout the course of the entire project, we created a number of demos related to industry events such as GDC, and sometimes ended up rushing features that we later wouldn't have time to properly polish. In the future, we'll plan our production schedule around key demos.

On the design side, our team worked through the game in chronological order, which meant the first half of the game got the most polish and finesse (as well as the most audience testing / feedback) whereas the last half of the game didn't get as much love. We might have solved some of this by picking the "star" levels from each chapter and working on those first, then filling in the gaps later.





## 5. Time Not Exactly on Our Side

It took almost exactly three years from conception to launch worldwide on all platforms. While a long production cycle proved necessary, it also gave time for issues to bloom.

We had planned to launch the PC version of *Rollers* in late 2013 / early 2014, but we pushed this when we partnered with Atlus USA and added two additional launch platforms (PS4 & Vita). Having the production timeline stretch out meant *Rollers* rubbed against the work-for-hire projects that we lined up because we need to keep the studio lights on.

We needed to split our resources into two project teams. Phantom Compass often works on multiple projects at once, and in many ways this is a strength, but in this case it caused more strain on finishing up *Rollers* than we would have liked and hurt our ability to polish.

Staffing was also challenging: From our original six-month concept through to finding financing for the full game we lost many of our original team members as they found full-time industry jobs. Onboarding dozens of part-timers and contractors in the early days of our prototyping was an expected, but nevertheless a strain.

Our lead programmer position was also a bit of a revolving door, with four lead programmers coming on and off the project over the development cycle. This cost time in bringing each replacement up to speed and resulted in legacy code kicking around that they were wary of removing.

We really couldn't avoid the long development cycle for this project, but there is value in keeping them shorter when possible.

## Conclusion

Was it worth it? Definitely. This game gave us many "firsts." We have all learned so much. We love this game and feel grateful that we are able to share it with others now who might love it too.

What will we miss the most now that production is over? Having the word "balls" come up hundreds of times per day.

What's next? We have a three-year plan to slowly wean ourselves off of work for hire projects and transition to creating on our own IP 100 percent of the time. We love the *Rollers* characters and universe a lot, and would like to keep them alive in another game... maybe a sequel, maybe something else. But for now we'll keep on rolling.

## Data Box:

Developer

- Phantom Compass

Publisher

- Atlus

Release Date

- Nov 18th, 2014 (Steam, PS4 and PS Vita SCEA)
- Nov 26th, 2014 (PS4 and PS Vita SCEE)

Platforms:

- Steam (Windows)
- PS4

- PS Vita

#### Number of Developers

- 3-7 full time
- 30 part-time contractors over time
- 15 voice actors, including Willow the Dog

#### Length of Development

- 2-D Prototype ~ 6 months
  - First Meeting Nov 9, 2011
  - Prototype ready for GDC March 2012
  - Polished up for submission for Funding spring 2012
- Unity 3D Full Production ~ 2 years 2 months
  - September 2012- November 2014

#### Budget

- Approx. \$700,000

#### Lines of Code

64225 Lines of Phantom Compass Written Code

+ 56680 Lines of Code from Plugins

= 120905 Total Lines of Code (ignores shaders)

#### Development Tools

- Unity 3D

[Return to the full version of this article](#)

Copyright © 2016 UBM Tech, All rights reserved