

This is a pre-copyedited version of an article accepted for publication in *Journal of Fandom Studies* following peer review. The definitive publisher-authenticated version is available through Intellect.

This article was published in *Journal of Fandom Studies* Vol. 6, No. 1 (March 2018). The published version is available here: https://doi.org/10.1386/jfs.6.1.7_1

What Were Minecart Boosters?

***Minecraft*, Digital Distribution, and Preservative Labor**

Over the past half-decade, studies of popular media reception and fan culture have been rife with morbid biological metaphors. Whether turning their gaze to the ‘deaths’ of television series (good, or otherwise), or the phenomenon of ‘zombie fandom’, works by figures such as Rebecca Williams, C. Lee Harrington, Natasha Whiteman, and Joanne Metivier have all examined the termination of the life cycles of pop culture artifacts, and the precarious position of the post-object fandoms left in their wake (Harrington, 2013; Whiteman and Metivier, 2013; Williams, 2011). Absent or dormant cultural objects have become, in recent years, a popular lens through which the destabilization of fan communities can be observed and documented.

Of course, while the definitive ending of a pop culture object (for instance, the finale of a television show) can provide a compelling platform from which to examine fan community transitions, it is important to not lose sight of the smaller destabilizations that occur at more localized conclusions (for instance, the death of a character on a show). To extend our biological metaphor, we should keep in mind that living things are always dependent upon their dynamism, and that healthy organisms undergo constant deaths on the cellular level (the processes of apoptosis and autophagy, if we want to get technical) to keep themselves metabolically fit. Death is, in fact, not always the end—sometimes, it is just *an* end, necessary to keep the larger whole in proper working order.

In the twenty-first century, this is perhaps nowhere more evident than in the lifecycles of digitally distributed videogames. Based on cycles of growth, maturation, and death that are quite

dissimilar from pop culture objects such as television series, digitally-distributed videogames likewise present an opportunity to examine forms of fan engagement—including forms of destabilization, dissatisfaction, and mourning—that stray from the definitive endings that characterize ‘post-object’ fandom, into more the more subtle ‘inter-object’ negotiations that arise from loving a volatile text. Employing our metaphor, the finalization of a game’s assets and code (‘going gold,’ to use the now-ancient parlance of disc-based distribution) was once seen as its ‘birth.’ Within the new mold of contemporary distribution strategies, finalization has become death. Digitally distributed games remain vital and ‘alive’ through their constant regeneration—which includes, along the way, a constant level of little cellular deaths, as bugs and under-developed features atrophy, molt, and are cleansed or consumed to make way for the new attractions that arrive with new updates. While these cellular deaths typically pass by unnoticed to most players, they can present more serious setbacks to serious fans pursuing more esoteric strategies within a game’s possibility space, sometimes representing a significant ending to a cherished practice.

This article investigates the fallout of one such cellular death in *Minecraft* (Mojang, 2011). Due, in part, to its tremendous popularity (as of July 2015, the game has sold 70 million copies), *Minecraft* stands as the most internationally visible harbinger of a new mode of digital distribution of games (Tyson, 2015). Throughout its first year of availability, the game was sold in an explicitly ‘unfinished’ state, enticing buyers through the lure of novelty and the promise of the game’s imagined future, rather than a set list of a stable set of features. The game’s massive success virtually ensured that other developers would follow suit, and the intervening years have seen the rise of paid beta and “early access” sales as a major revenue stream for independent game developers. Game distribution is undergoing a sea change, with the previously assumed stability of the single-player videogame making way for an all-out embrace of the volatility digital distribution allows. This sea

change, in turn, creates new opportunities for miniature endings and losses to affect dedicated fan communities.

My choice of *Minecraft* as a case study, and my interest in the vital dynamism of digitally distributed games in general, is animated by two central concerns. First, *Minecraft* presents an important and necessary opportunity to study emerging forms of awareness on the part of players of games' status as *software*—as collections of ever-expanding lines of code, updates to which promise a stream of new delights, but can also hold the danger of crashes, of compatibility problems, or the occasional bout of outright brokenness.¹ This manner of conceiving of games has a profound effect on conceptions of the temporal boundaries of games as objects—of when they emerge into existence, of when they fade away, and to what extent they can change, ship of Theseus style, while still being considered a coherent object. Game studies in general, and game fan culture studies in particular, would be wise to take note of such shifts. Theories of player engagement have come a long way since early over-reliance on blanket terms such as 'immersion,' developing more complex models that can account for more phenomena. What has gone under-theorized, however, is the ways in which an informed awareness of a game's code, and the volatility of that code, can be comfortably concomitant with a deep engagement with the game's attempts at world-building.

Scholars such as Noah Wardrip-Fruin, who sit on the boundary between the genre of game studies and the more specialized methodological corner of software studies, have provided a helpful template for considering the ways in which even relatively non-technically minded players understand games as software. 'A gardener doesn't need to understand chemistry', Wardrip-Fruin writes, 'and a *SimCity* player doesn't need to understand programming language code, yet both can come to grasp the elements and dynamics of complex systems through observation and interaction' (Wardrip-Fruin, 2009, p. 310). The careful respect accorded here to *SimCity* players' powers of observation is appreciated, but this line of inquiry can—and, indeed, must—be pushed further. In

today's era of digital distribution, the dividing line between casual player and more dedicated fan is often located in the latter's ability to maintain a diachronic awareness of a games' code, to understand the changes brought by the shifting flow of software updates, the birth and death cycles of features and mechanics, and to adjust practices of play and community sharing accordingly.

Secondly, the dynamism of an object such as *Minecraft* presents an unusually nuanced arena for examining conflict in the fan-producer relationship. Given that fans, from the very outset, have been prepared to accept *Minecraft* as a fluid and unstable text—indeed, that is part of its very appeal—responses to sudden change tend to be more muted than, for instance, the reactions of “betrayal” that Will Brooker has traced among fans that perceived shifts in tone and quality in the *Star Wars* franchise with the arrival of *The Phantom Menace* (Brooker, 2002). Although anger and resentment over changes do sometimes pollute the fan-producer relationship of *Minecraft*, the game's fan community also shows a striking willingness to forgo venom in favor of various modes of what I term *preservative labor*: the work of keeping aspects of the past alive, even as software updates march on.

Gaming fan communities have long demonstrated a strong commitment to preservative labor. For instance, the proliferation of emulators (virtual machines that ensure the continued playability of games far beyond the life cycle of their original platforms) evinces not only a punk refusal of the game industry's technological fickleness, but also a striking willingness on the part of fans to anonymously donate hours of highly skilled technical labor to community preservative projects. The phenomenon of digital distribution, meanwhile, illustrates what happens when this drive towards preservation slips from the realm of platform to the realm of the individual game. If, previously, the war waged against obsolescence was one that took place after the death of a platform, today its battles play out within the lifecycles of games, as fans take up the regiments of upkeep required when residing within a volatile virtual landscape. Due to reasons ranging from the

game's mechanics to its developer's philosophies, *Minecraft* is especially well-positioned to encourage this sort of in-game preservative labor, a fact which adds a unique flavor to its fan-producer relationship.

***Minecraft*, digital distribution, and dynamism**

On 30 June 2010, Swedish game developer Markus 'Notch' Persson released alpha version 1.0.0 of his sandbox mining, building, and crafting game *Minecraft* to the public. This was not the first version of the game released for public consumption. Persson had been distributing even rougher versions of the game for over a year, first publically sharing a version referred to as '0.0.11a' via a link on his blog on 17 May 2009. It was, however, the first time that Persson had asked players to put down money for the game. For a single payment of US \$13, those interested could download a copy, with the promise that they would receive a continuous stream of updates during the game's alpha and beta stages. Standing as something of an inversion of the old shareware model that had helped make id Software's *DOOM* a hit back in 1993, in which players can first download a free portion of the game before putting down money for the rest of it, this model paired an upfront charge with the promise of additional content later on—as well as an opportunity for a first-hand glimpse into the process of iterative game design as he gradually added features, and the possibility of shaping the development of the game. The gameplay of *Minecraft*, which finds players exploring a procedurally generated landscape of cubic blocks, collecting resources so as to build structures and defend themselves against monsters that emerge at night, proved to be a good match for this distribution scheme. The game emerged as a breakout hit, with millions of players putting money down to play the game during its alpha and beta stages of development: Between 20 June 2010 and 6 April 2011, still seven months away from its official 'full' commercial release, the game generated approximately US \$32 million in revenue for Persson and his fledgling company Mojang Specifications (Johnston, 2011).

Fully exploiting the possibilities of digital distribution, *Minecraft* pushed dozens of updates on its players during this period: Between 15 September 2010 and 15 September 2011, for instance, the game's alpha and later beta went through forty-six officially numbered versions. During this time, upon any given boot-up of the game, players could potentially be greeted with an 'updating' screen, beyond which would lie their familiar virtual homesteads, now replete with new in-game entities, new rules, and new possibilities. This could include new flora, new fauna, new building materials, or new items and tools that savvy players would quickly learn how to assemble. It could also take the form of changes in simulated weather patterns, AI behavior, or the propagation of animal populations. One of the key pleasures of *Minecraft* is exploration, and the game's steady stream of updates ensured that players booting up the game would have a good chance of seeing something completely new over their next horizon.

[Figure 1 near here]

Before *Minecraft*, paid alphas and betas were exceedingly rare. In the wake of its success, the practice of asking players to pay upfront for an unfinished game has joined crowdfunding as one of the major methods for securing capital in the contemporary independent game development scene. In March of 2013, Valve Corporation's mammoth PC game sales platform Steam unveiled its 'Early Access' release option, specifically intended for the sale of games still in active development. What was once a highly unusual payment scheme has, in the intervening years, become commonplace, especially among small independent development teams.

The benefits of such distribution schemes for game developers are obvious: Much like services such as Kickstarter (which first appeared in April 2009, just one month before *Minecraft*), early access purchases give them access to a revenue stream that can serve as an alternative to traditional investor interest, providing them with financial resources to complete their games. But what sort of expectations does it create for players? Here, a danger emerges. If the central appeal of

games distributed early in their production cycle is, in fact, their promise of continued novelty, then players are inevitably going to be let down once stability is reached. Regular updates to a game can continue past its hitting official release version 1.0, of course, but updates such as these present an opportunity cost to developers, who may want to begin work on a new project, rather than spend resources on an eternally mutable game with an ever-expanding list of features.

Minecraft has sidestepped this problem so far, by providing its players with regular content updates well beyond its ‘full’ commercial release on 18 November 2011. Other developers, however, lacking the resources of Mojang, have had no choice but to eventually settle on a final, stable version of their product—and have suffered the wrath of fan ire because of this. When Andrew ‘Redigit’ Spinks, the lead developer of *Terraria* (Re-Logic, 2011), announced in February 2012 that he would stop pushing out regular updates for his game—fully ‘released’ nearly one year prior—to spend more time with his infant child, player response ranged from devastated to outright aggressive. The response of Jesterhead, a poster on the official board that Spinks made his announcement of, reveals the accusatory rage that can emerge when an implicit promise of constant novelty is broken:

‘As far as i am concerned this game was bought and i came to love for its open-ending-creative world - which also included periodically updated and increased content (long discussed and promised). Considering that, in order to achieve such updates, you need perhaps 5 working days per month it is absolutely disgusting - your offer of one final bugfix as a thank you for thousands of supporting (read: paying) customers.

Leave the game in the hands of community, release the complete coding, make the payments go to charity.

You ungrateful f***’ (Redgit 2012: post #6).

One might ask: If this is the now the fan reaction a developer can expect for ceasing work on a game, then why should developers ever consider ‘completing’ a game’s development, at all? And, indeed, there are some signs that the game distribution landscape is shifting in the direction of this mindset. Steam’s Early Access FAQ page includes the following scare-quote heavy disclaimer:

‘Its up to the developer to determine when they are ready to “release.” Some developers have a concrete deadline in mind, while others will get a better sense as the development of the game progresses. You should be aware that some teams will be unable to “finish” their game’ (Valve Corporation 2013).

Unable—or, perhaps, unwilling. After all, what is the benefit of ever emerging from a paid beta stage, of ever hitting a stable release version of 1.0, if such a development is only going to be greeted by fan outcry that the game is now ‘dead’? If, today, a game is alive only as so long as it is a dynamic entity, then to be a finished product is to be, quite simply, *finished*. And so, in the age of Kickstarter and the endless parade of stretch goals it allows, some developers have gleefully promised to *never* ‘finish’ their games—a move which some game critics have responded to with rejoice (Farockhmanesh 2014, Kaiser 2014).

Such a development creates some headaches for those who study game history and fan cultures. Previously, genres with a dominant social online component, such as the massively-multiplayer online game, had carved out a unique space within game studies in which version numbers and expansions had to be methodologically accounted for, precisely to address fan reactions to the inevitable loss that comes with change.² Now, not even single player games possess anything quite so quaintly stable as a ‘release date.’ Instead, they exist as dynamic slices of time. Areas such as software studies have already begun accounting for such complications: Adrian Mackenzie, attempting an analysis of the Java Virtual Machine, has commented on the difficulties presented by software that ‘continues to be in formation,’ proposing that such new media texts are perhaps ‘more productively approached as a project,’ rather than as a single, isolatable object (MacKenzie 2006: 96). Game studies, it seems, need to catch up.

Quite aside from professional game historians, such developments also create difficulties for game fans. Although the general trend may be toward an embrace of game volatility and the novelty it allows, sometimes seemingly small changes can blossom into significant losses for fans pursuing particularly esoteric in-game practices, instigating bursts of both mourning and preservative labor. In

the remainder of this article, I undertake a small and focused case study of one such change in *Minecraft*: a change in the physics behavior of minecarts and minecart tracks, two virtual entities that players can assemble within their *Minecraft* game, which resulted in a significant shift in fan behavior in 2011.

The life and death of minecart boosters

As previously mentioned, *Minecraft* is a game built upon the activities of exploration, mining, building, and crafting. The landscape of *Minecraft* is constructed from stacks of cubic blocks, texture mapped differently but otherwise uniform in shape and, for the most part, size. Given the proper tools, players can destroy these environmental blocks, producing a number of resource blocks that can then be added to the player's inventory. Destroyed trees beget wood blocks, destroyed stone blocks beget cobblestone, and so on. These resource blocks can be used to *build*: they can be placed back into the environment, arranged by players into an astoundingly varied array of architectural forms, limited only by the cubic shape of the blocks themselves.

[Figure 2 near here]

Resource blocks can also be used to *craft*—that is, arranged in a specialized configuration on a grid within the player's inventory screen to assemble a new tool, such as a pickaxe, a torch, or a furnace for smelting ores. Crafting can only be accomplished via specific 'recipes', pre-programmed arrangements that blocks must be drag-and-dropped into in order to produce a new useable item. In a design decision that ultimately shaped the contours of the game's fan community, the PC version of *Minecraft* comes with no tutorial or digital instruction booklet to cue players in to what these 'recipes' are. Players are left, on the one hand, with arduous trial and error, or, on the other hand, a wealth of player-assembled online knowledge, in the form of beginner's guides, YouTube 'let's play' videos, and the official *Minecraft* wiki. The vast majority of *Minecraft*'s player base are dependent on this type of player-generated documentation for enjoyment of the game, enabled by a much smaller,

self-selected group of wonkish fans willing to put in the time to discover, decipher, and publicize the game's more esoteric items and systems.

On 18 June 2010, when the game was still in its pre-alpha stage, 'Notch' Persson introduced two game-changing craftable objects to the landscapes of *Minecraft*: *minecarts*, and their attendant *minecart tracks*. Although nominally intended for transporting materials out of the game's titular mines, the fact that players could hop into minecarts and use them as a mode of personal transportation helped ensure that industrious players soon found a wider panoply of uses. This included rail lines that cut down arduous commutes across virtual landscapes, and roller coasters exploiting the vertiginous thrill of the illusion of bodily movement first-person games such as *Minecraft* can provide.

When minecarts first debuted, they could be set rolling using the momentum of a downhill incline, but otherwise there was no official way of granting them velocity. This was rectified on 30 July 2010, when alpha version 1.0.14 introduced powered minecarts—carts that contained a furnace within them, and could act as locomotives, pushing non-powered carts, while consuming coal, an in-game resource players had to mine for. Persson, however, had underestimated two things: the industriousness of fans within the forty-two day gap he created between introducing minecarts and introducing their official propulsion system, and the stinginess of players when it came to the time and energy needed to collect an in-game resource.

We see the offspring of this mixture of industriousness and stinginess in a video uploaded onto YouTube on 19 August 2010, by *Minecraft* tutorial creator The Carlz0r, titled '*Minecraft* Alpha – Two types of cart “boosters”'. In it, The Carlz0r creates a downhill slope of minecart track lasting the span of only one block—hardly enough to establish any momentum whatsoever. Next, he extends this length of minecart track up a steep incline—normally, a death-knell for the momentum of minecarts, and an especially impossible climb for a cart with as little velocity as we would imagine

a cart on this track having. From here, things get curious. The Calz0r lays down a length of track that runs parallel to his initial track for five blocks, before terminating at a wall. He places a minecart on this parallel track, and then jumps into a minecart on the original track. ‘Now,’ The Calz0r intones on the video’s soundtrack, ‘when this minecart comes down off the starting point and goes down this angled slope, it’s gonna hit this minecart kinda strangely, and they’re gonna glitch each other out, and push each other up the hill.’

[Figure 3 near here]

Persson’s rapid work on the development *Minecraft*, it turns out, had resulted in some sloppy coding in the area of the game’s physics simulation. Soon, thanks to tutorials created by dedicated and keenly observant fans such as The Calz0r and his fellow YouTuber AlmtYBob, the knowledge expanded outward into the general player community that when two minecarts ran on adjacent tracks, they displayed some unusual behavior. First, they matched speeds, with one seeming to drag the other along. Then, more curiously, they rapidly accelerated: Their dragging motion did not create friction, but instead seemingly reversed it, with the two moving objects briefly reinforcing each other’s velocity by sharing it.

This movement behavior was nothing but a physics glitch: a bug. Within the culture of programming, bugs are considered a nuisance at best, a PR disaster at worst. They can fundamentally break a piece of software (and financially break the company that produced it); they are something to be caught during quality assurance, or, as a worst-case scenario, fixed in a speedy patch. Players, however, can have quite different relations to bugs than those available to software developers. For players, bugs can be amusing, endearing, or even, in certain rare cases, quite productive. During its heyday, the ‘booster’ physics bug of *Minecraft*’s minecarts was a bug with a following, with the keen observations of a few fans eventually inspiring no small level of player industriousness.

Notably, players were not in complete alignment when it came to the ‘booster’ bug. A gentle rift emerged within the player community: Some users gleefully embraced the possibility of the glitch, while others restrained themselves. Sometimes, this restraint proceeded out of a staid moral opposition to glitch-exploitation. *Minecraft* Forum user Magix mulls over this viewpoint in a post from 19 November 2010, three months after the initial upload of The Calz0r’s video:

‘As I understand Notch didn't actually intend this to happen but since it has so great use, many people use it in their worlds. But now I've seen quite a few people QQing that it's abusing bugs and that they don't use this because of that. What do you think? Would you rather Notch implemented some sort of a block or a way to boost your cart ‘legally’?

To be honest, I wouldn't want that to happen. I like the current system. ... Who cares that it's a glitch. I hope it never gets fixed’ (Magix 2010: post #1).³

For *Minecraft* players at this time, the question of what constituted ‘cheating’ was very much an open one. Games scholar Mia Consalvo has mapped out several categories of player attitudes toward cheating, including players who consider any exploitation of bugs (even when no other players are being taken advantage of), or any use of external sources such as web-based guides, to be cheating (Consalvo, 2007, pp. 87–92). For a game in active development, however, it is not always entirely clear what counts as a glitch. Further complicating the matter is the fact that *Minecraft* had no official in-game tutorial at this time, meaning that fan wikis and YouTube videos were how the majority of players learned to play the game. Clearly, *Minecraft* was rendering the player attitudes chronicled by Consalvo out-of-date, and new fault lines were emerging as these attitudes evolved.

Other players, however, avoided minecart boosters not out of high-minded morality, but simply out of caution, knowing that the glitch would likely be fixed in an upcoming update). Later down in the same thread, fellow forum user Hoodwink stresses the importance of players always keeping in mind boosters’ clandestine and unofficial nature, ‘just in case they get ‘fixed’ somehow, rendering some very creative and highly impressive minecart systems totally defunct’ (Magix 2010: post #5).

Prescient warnings such as Hoodwink's, however, could not stop the march of technological progress. Over the course of months, fans' careful trial-and-error observation of cart behavior resulted in a burgeoning scene of folk engineering, all in pursuit of the dream of crafting virtual perpetual motion machines. By November 2010, *Minecraft* Wiki editors such as CrazyTerabyte and Michamus were cataloguing and providing instruction on elaborate feats of user-generated engineering, offering handy visual guides to industrious rail line creators looking to design such elaborate propulsion mechanisms as the 'two-way corner auto-reset,' the 'uphill auto-reset,' and the 'auto start.'

And then, on 26 May 2011, beta version 1.6 was released, and the dream abruptly died. The physics bug was fixed. Overnight (or, more accurately, over the course of a download of an updated *minecraft.jar* Java archive file), the physics of *Minecraft* underwent a revolution, forever changing the landscape of players' worlds. Elaborate booster-powered rail systems suddenly failed to function, the basic physical principles underlying their engineering having been utterly erased. Although the user-built structures still remained, they instantly became vestigial, monuments to the lost opportunities and forgotten possibilities of a physical universe constantly in flux.

Here, we have an example of a software update as the source of strain in the fan-producer relationship. Persson was well aware of fan behavior, and was frankly apologetic about the change he enacted, giving ample warning to his followers on Twitter ahead of time: 'From my changelist: "* Fixed minecarts next to each other causing extreme velocities (sorry!)"' (Persson 2011). An apologetic god, however, is still a god: There is no way around the fact of Persson's omnipotent grip on the physics of *Minecraft*, and the fact that he caused, in a series of keystrokes, a cellular death—the sudden (if long-suspected) end to a cherished fan practice.

Upon the release of beta 1.6, *Minecraft* players who had placed their bets on the continued efficacy of the minecart booster system found themselves faced with the task of infrastructure

upkeep, of saving their rail lines from obsolescence. Other, more reliable means of propulsion—powered minecarts, or the powered rails introduced in beta 1.5, specifically designed to be functionally isomorphic to the booster system (another magnanimous gesture toward booster-users)—had to be adopted. A central overseeing body had set new standards for construction: not a state government with the power to levy fines, but instead a game studio, with the power to change physics by pushing updates through. The matter of compliance was left up to the individual player. Legacy systems were instantaneously broken by uncooperative physics, and, in order to be salvaged, they had to be repaired, restored, refurbished—‘brought up to code,’ to put it punningly.

Some players were resistant. The construction of powered rails required the acquisition of in-game resources, and, being designed from the ground up to perform a specific function, the rails were also less susceptible to perpetual-motion exploits. Of course, this did not stop industrious fans from trying to find such exploits. MCFUser 472924, a restless denizen of the *Minecraft* Forum, gleefully announced only July 2, 2011 the discovery of ‘a replacement for the bug boosters that were fixed,’ one that incorporated booster rails into a mechanism providing momentum that ‘(i think) is infinite, or will last a very long time’ (MCFUser472924 2011: post #1).

Skepticism toward such bug-based solutions, however, had set in among the community. ‘No, it’s a glitch,’ responds user Mardoc, ‘A glitch that will be fixed’ (MCFUser472924 2011: post #9). Fans were becoming increasingly consciousness of the finitude of anything in *Minecraft* not officially sanctioned by Mojang, and of the futility of founding engineering endeavors upon game ‘features’ that were actually bugs. As Mojang became a larger company with better quality assurance, and as *Minecraft* barreled toward its full commercial release and eventual ascent to becoming one of the best-selling videogames of all time, the fan community became more cautious and less ambitious about such matters.

Building, Dwelling, Preserving

Enraptured engagement and technological awareness: often, these two phenomena are seen as occupying opposite ends of a spectrum. Those who understand how a given media artifact works—particularly, a new media artifact, with its underpinnings in code—are taken to be enlightened, approaching their objects with a demystified scrutiny. Those who surrender themselves to the pleasures of a piece of entertainment are assumed, on the other hand, to possess questionable media literacy, to be precariously close to being uncritically ‘fooled’ by a particular artifact.

This dichotomy finds elegant expression in the writings of science fiction author and free software advocate Neal Stephenson, whose writing has become a frequent touchstone in new media studies, particularly within software studies. Thrown into a state of shock and horror upon encountering imperfect backwards compatibility when upgrading to Microsoft Word version 6.0, Stephenson pinpoints this personal moment as one of the central turning points in his turn away from GUI-based operating systems. Once one becomes fully aware of the fragility of one’s work in the face of compatibility problems, Stephenson poses, how can one ever again be seduced by graphical representation (Stephenson 1999)? Glitches, bugs, and software updates that lead to compatibility problems: these are precisely the sorts of things that are supposed to pull users out of their experience, disrupt their flow (whether that term refers to work or play), and put the breaks on enraptured engagement.

As has been shown throughout this article, however, sometimes things are more complicated. The acute awareness of the ways in which objects and spaces presented graphically on a screen are determined by code *can* lead one to reject all they see as mere metaphor, mere illusion, as Stephenson suggests. But such awareness can also be negotiated, can coexist with other forms of understanding and practices. For a certain breed of fan, pouring over patch notes and discovering ways to re-engineer constructions has no ill effect on their engagement with *Minecraft*. Yes, the fluid physics of *Minecraft* throughout its version changes present an opportunity for annoyance. But such

changes also presented an opportunity for fans to take charge, to engineer new solutions and generously share them with less-inquisitive players, to prove their dedication to the game by gracefully embracing the world-affirming labor of maintenance.

In his famous essay ‘Building Dwelling Thinking,’ Martin Heidegger traces an etymology of the Old English and High German word *ban*, mapping out a history of the interconnection between the concepts of dwelling and of building, both of which spring from this original linguistic root. ‘Dwelling,’ Heidegger insists, *is* building, but ‘genuine building’—building considered broadly, outside of the narrow work of construction and architecture. Here, Heidegger does in part mean ‘the raising up of edifices,’ but also wants to include human labor that aims to ‘cherish and protect,’ to ‘preserve and care for.’ Heidegger includes here the ‘nurturing’ work of agriculture, ‘to till the soil, to cultivate the vine’ (Heidegger 1971: 157). Subsequent commentators such as Iris Marion Young have added the work of keeping things ‘cleaned, dusted, repaired and restored’ (Young 2005: 142). The routines of *upkeep* can be a lifelong project granting years of meaningful labor—can be, in Young’s words, ‘fundamentally world-making’ (145).

Much of players’ activity in *Minecraft* bends towards the ‘building’ end of Heidegger’s spectrum, and away from the ‘preservation’ end. The game’s main pleasures are grounded in its well-honed cycle of exploration, resource exploitation, and ever-more-ambitious construction. On the face of it, care and preservation seem to be rarer pursuits: Nothing in the pristine, cubic world of *Minecraft* ever needs dusted or cleaned.

However, things do occasionally need repaired and restored. According to Young, the work of preservation ensures that things are ‘protected from the careless neglect or accidental damage caused by those who dwell among and use them’ (Young 2005: 142). In *Minecraft*, we can add another category of protection: Things must be protected against accidental damage caused by the

code that hums beneath them. As updates render what was formerly in working order functionless, players must devote time to reparative labor.

Sometimes, this labor takes the form of toiling *with* the game, as a piece of software. If a player finds that a new update breaks a civil engineering project, sometimes the best course of action may be to roll back that update. In 2011, when the boosters died, this process required disabling the game launcher's updating functions, finding a 'snapshot' of a previous version archived online on a fan site, searching through one's computer files to find the location of the 'minecraft.jar' file, and replacing it. If they so desire, players can stand athwart history and yell 'stop' (successfully!), forgoing the thrill of a constantly progressing game for the stability of a familiar version. (On 1 August 2013, Mojang added the officially sanctioned ability to launch prior versions via a drop-down menu in the games' software launcher—a clear indication that they were aware of some fans' desires for stability and preservation tools over constant novelty.)

Some players, though, embrace progress, despite the havoc it wrecks. For some players, rolling back to a prior version is tantamount to giving up. The more satisfying alternative is to work *within* the game, to re-engineer their devices, to embrace the zen-like pleasures of preservative maintenance. Yes, this means the most dedicated fans must mark their old manuals as obsolete (as The Carlz0r did, annotating his YouTube videos to reflect the fact that certain tips no longer work), and draw up new schematics. But dwelling within the world of a game distributed while in active development means eking out what stability one can in a restless universe. Counterbalancing the flood of new features, there will inevitably be the occasional small disappearance, the occasional cellular death, pockmarking this brave new landscape with monuments to obsolescence.

The alternative, of course, would be to return to an era in which games are 'finished,' in which their code and assets are eventually finalized, and cease to grow and change. And, to a certain generation of gamers, such circumstances look a lot like death.

Acknowledgements

I would like to thank Will Carroll, Patrick Jagoda, Mikki Kressbach, James Lastra, Jordan Schonig, Tyler Schroeder, and Zhang Ling for their helpful input throughout the writing of this article, as well as Justin Schumaker and Peter McDonald for the opportunity to present an early version of this material at SLSA 2014.

Notes

1. Of course, we can map a similar awareness onto the fan communities of analog games, whose opinions and feedback sometimes influence changes to published rule sets, for instance through the distribution of errata inserts or new editions of rulebooks (Švelch 2014).

2. Game studies has been rife with quality studies of change, disruption, and endings within MMO communities. Standouts here include Celia Pearce's study of players displaced by the closing of the MMO *Uru: Ages Beyond Myst* (Cyan Worlds, 2003), and Henry Lowood's overview of fan-made machinima reactions to changes in MMOs both small and large (Lowood 2011, Pearce and Artemesia 2009).

3. I have chosen to use the *Minecraft* Forum as my primary archive of player reactions to the 'booster' glitch, due to the fact that, having been initially created on 16 June 2009, less than one month after the game was made publically available (and over a year before alpha version 1.0.0 would first be made available for purchase), it is the oldest forum for *Minecraft* users, and the one that was most active during the game's beta stages, before it achieved more mainstream popularity in the wake of its 'full' release.

References

- AlmtyBob (2010), 'Bob's *Minecraft* Tutorial 1: Minecart Boosters',
<https://www.youtube.com/watch?v=gPls6LXlkGI>. Accessed 2 December 2014.
- Brooker, W. (2002), *Using the Force: Creativity, Community and Star Wars Fans*, New York: Continuum.
- Carlz0r, The (2010), '*Minecraft* Alpha - Two types of cart "boosters"',
<https://www.youtube.com/watch?v=LzGcirn15to>. Accessed 5 December 2014.
- Consalvo, M. (2007), *Cheating: Gaining Advantage in Videogames*, Cambridge, MA: MIT Press.
- CrazyTerabyte (2010), 'Tutorials/Minecart booster: Revision as of 15:05, 3 November 2010 by CrazyTerabyte',

- http://minecraft.gamepedia.com/index.php?title=Tutorials/Minecart_booster&oldid=2408
7. Accessed 11 October 2015.
- Farokhmanesh, M. (2014), 'Why *Star Citizen* May Never Truly Be Complete'
<http://www.polygon.com/2014/6/13/5805052/star-citizen-chris-roberts-interview>.
Accessed 10 October 2015.
- Harrington, C.L. (2013), 'The Ars Moriendi of US Serial Television: Towards a Good Textual Death'. *Int. J. Cult. Stud.*, 16, pp 579–595.
- Heidegger, M. (1971), 'Building Dwelling Thinking', in A. Hofstadter (tran.), *Poetry, Language, Thought*, New York: Harper & Row, pp 145–161.
- Johnston, C. (2011), 'Minecraft to Exit Beta November 11 with \$32 Million in Revenues',
<http://arstechnica.com/gaming/2011/04/minecraft-to-exit-beta-november-11-with-32-million-in-revenues/>. Accessed 29 November 2015.
- Kaiser, R. (2014), 'Early Access Exposes the Lie That the Best Games Should, or Even Can, Be Finished', <http://www.polygon.com/2014/1/24/5338478/early-access-exposes-the-lie-that-the-best-games-should-or-even-can>. Accessed 9 October 2015.
- Lowood, H. (2011), 'Video Capture: Machinima, Documentation, and the History of Virtual Worlds', in H. Lowood and M. Nitsche (eds.), *The Machinima Reader*, Cambridge, MA: MIT Press, pp 3–22.
- Mackenzie, A. (2006), *Cutting Code: Software and Sociality*, New York: Peter Lang.
- Magix (2010), 'Minecart booster discussion',
<http://www.minecraftforum.net/forums/archive/alpha/alpha-survival-single-player/816591-minecart-booster-discussion>. Accessed 2 December 2014.
- MCFUser472924 (2011), 'Minecart bug booster replacement!!!!',
<http://www.minecraftforum.net/forums/minecraft-discussion/discussion/140138-minecart-bug-booster-replacement>. Accessed 2 December 2014.
- Michamus (2010), 'Tutorials/Minecart booster: Revision as of 04:41, 9 November 2010 by Michamus',
http://minecraft.gamepedia.com/index.php?title=Tutorials/Minecart_booster&oldid=2637
3. Accessed 11 October 2015.
- Pearce, C. and Armesia (2009), *Communities of Play: Emergent Cultures in Multiplayer Games and Virtual Worlds*, Cambridge, MA: MIT Press.
- Persson, M. (2009), 'Minecraft 0.0.11a for public consumption',
<http://notch.tumblr.com/post/9290455106/minecraft-0011a-for-public-consumption>.
Accessed 10 October 2015.
- Persson, M. (2011), 'Markus Persson on Twitter: "From my changelist: "* Fixed minecarts next to each other causing extreme velocities (sorry!)"',
<https://twitter.com/notch/status/68693850062995456>. Accessed 2 December 2014.
- Redigit (2012), 'My Fellow Terrarians', <http://www.terrariaonline.com/threads/my-fellow-terrarians.81455/>. Accessed 9 October 2015.
- Stephenson, N. (1999), *In the Beginning ... Was the Command Line*, New York: HarperCollins.
- Švelch, J. 'Regarding Board Game Errata', <http://analoggamestudies.org/2014/12/regarding-board-game-errata/>. Accessed 10 November 2016.
- Valve Corporation (2013), 'Introducing Early Access',
<http://store.steampowered.com/earlyaccessfaq/>. Accessed 10 October 2014.
- Wardrip-Fruin, N. (2009), *Expressive Processing: Digital Fictions, Computer Games, and Software Studies*, Cambridge: MIT Press.

- Whiteman, N., Metivier, J. (2013), 'From Post-Object to "Zombie" Fandoms: The "Deaths" of Online Fan Communities and What They Say About Us', *Particip. J. Audience Recept. Study*, 10, pp 270–289.
- Williams, R. (2011), 'This Is the Night TV Died': Television Post-Object Fandom and the Demise of *The West Wing*, *Popular Communication*, 8, pp 266–279.
- Young, I.M. (2005), 'House and Home: Feminist Variations on a Theme', in *On Female Body Experience: Throwing Like a Girl and Other Essays*, New York: Oxford University Press, pp 123–154.