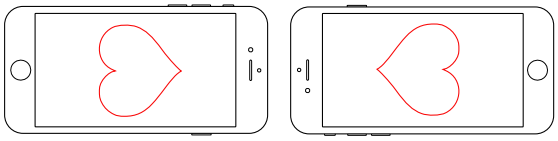
The Fibreculture Journal
DIGITAL MEDIA + NETWORKS + TRANSDISCIPLINARY CRITIQUE

issn: 1449-1443

issue 25: Apps and Affect.



FCJ–181 There's a History for That: Apps and Mundane Software as Commodity.

doi: 10.15307/fcj.25.181.2015

Jeremy Wade Morris and Evan Elkins
University of Wisconsin-Madison

Abstract:

With global app sales estimated at $25 billion in 2013 and thousands of software developers marketing all manner of services and products as apps, it is hard to deny the importance of software applications for smartphones and other mobile computing devices as an economic and cultural platform. While apps provide many functions previously possible with software, apps represent a new way of producing and packaging software. This article traces a lineage of the term app within the context of the software commodity's longer history. We argue apps, as mundane software, represent a particular affective and contextual experience of software that expands the potential uses of software but also embeds it more deeply in everyday practices.

Introduction

I Am Rich was first released in August 2008 in Apple's iOS App store for $999 (Milian, 2008). The program's only function, other than displaying an image of a jewel, was a self-congratulatory message that read 'I am rich, I deserv it, I am good, healthy & successful [*sic*].' It was reportedly purchased 8 times before Apple removed it from the store. *I Am Rich* could be read as clever commentary on conspicuous consumption (Veblen, 1965) in an era of digital and ephemeral goods or dismissed as a trivial piece of software – a novelty joke like an electric shock pen or a whoopee cushion (both of which, incidentally, you can also find as apps). However, these readings of the app's limited functionality and ostentatious pricing miss a much larger story about the way apps have radically reconfigured the presentation and pricing and potential purposes of the software commodity. *I Am Rich* is both a meditation on the value and worth of digital cultural goods and a realisation of what we might call the mundane abilities of software.

Individual apps like *I Am Rich* might at first glance seem meaningless but, collectively, this new format of the software commodity represents a significant emerging economic market and cultural platform. *I Am Rich* is just one of nearly 1.5 million apps available across half a dozen app stores (AppAnnie, 2014b; Ingraham, 2014). With over 100 billion app downloads, global app store revenues were around $25 billion last year, up 62% from 2012 (Lessin and Ante, 2013). These are projected to rise to nearly $77 billion in the next 3 years (Gartner, 2014). The app subsector involves thousands of independent and established developers, from hundreds of countries, and the software they create finds its way into smartphones, mobile computing platforms, cameras, televisions, cars, game consoles, and a host of other technologies.

Accordingly, this article explores 'apps' as a distinct format for the delivery and distribution of software and places apps within the longer history of the software industries. Through an analysis of computer and technology trade journals, as well as historical archives (e.g. Computer History Museum, Charles Babbage Institute), we trace the lineage of the term 'app' and the evolution of software as a distinct commodity. Thinking both about the political economic dimensions of the production of software as commodity and the experiential aspects of encountering software in the form of app, we consider 4 key characteristics (micro-control, micro-transactions, micro-functional and micro-material) that are reshaping software's commodity form. On the one hand, apps draw on software's historical liminal status between free, shared or hacked good and paid product to advance various economic experiments for selling software in the digital age. On the other, the presentation of software as app has been key to embedding it more deeply into leisure, commercial, educational, interpersonal and other spheres of everyday activity.

While research in media and software studies has noted the importance of video games, enterprise software, and other popular genres of programs, we look at apps as a particular economic and aesthetic presentation of code to ask what insights reveal themselves if we take apps seriously, if we seek meaning in the mundane?

Mundane Commodities

Apple's iOS store debuted in July 2008 with around 500 apps for its iPhone and iPod Touch devices and saw close to '10 million applications downloaded in just three days' (Apple, 2008). Competitors such as Google, Amazon and RIM followed suit with app stores of their own. The growth in the number of apps and size of revenues from these stores has prompted observers to note we now inhabit a 'Planet of the Apps' (Bruno, 2010) and that each passing year is the 'Year of the App' (Dowell, 2010; O'Malley, 2013; Smith, 2014). With around 58% of U.S. adults using smart phones, and over 42% with tablets, apps are becoming one of the most frequent ways users interact with software (Olmstead, 2014). Games are the most popular type of app, though spending on non-game apps, like news, weather, maps, entertainment, lifestyle, banking, social networking and communication, increased 2.3 times during the last two years (AppAnnie, 2014a; Purcell, et al., 2010).

Discussions of the recent rise of apps, however, ignore the app's place in a longer history of software as a commodity. Apps represent, first and foremost, a particular model for presenting, marketing and distributing software that stands in relation to how software has typically been positioned and sold. This section looks at the history of the term app to trace some of the social and cultural values this particular mode of distribution holds for everyday interactions with software.

We are particularly interested in the app as a specific manifestation of the software *commodity*, since the form of the software commodity holds deeper social, cultural and economic insights about digital goods more generally. Marx called the commodity a strange and mysterious thing, 'abounding in metaphysical subtleties and theological niceties' (1978: 319), but he also grounded commodities in the materials of their production, the labour it took to produce them, and the expectations they held about their eventual consumption. As Lee argues, the commodity is a 'vital touchstone in any analysis of the social relations of capital' because it embodies the social relations of its production and points us beyond the objects themselves to the cultures in which they are made and consumed (1993: 120).

The commodity form is not just a means of selling goods, it is also 'an objectification of a mode of production at a given phase of [capitalism's] development' (Lee, 1993: 120). Moreover, changes to the commodity-form over time become important inflection points for understanding the changing dynamics of capitalism (Lee, 1993). Each phase of capitalism has a corresponding and identifiable 'ideal-type commodity form' that embodies the most powerful economic, technological, and cultural forces at work at any given historical instant. Cars and suburban housing during Fordism, for example, were not simply goods that became key industries, but ideal-type commodities around which a whole set of social practices and values formed (Lee, 1993: 129). Kline et al. draw on Lee to argue video games represent an ideal-type commodity form in late capitalism, given the ways games fuse and fuel 'technological innovation, cultural creativity, and mediated marketing' (2003: 29). Similarly, Snickars and Vondereau suggest the iPhone is an ideal-type commodity form for the digital age, a 'media *dispositif*' that encapsulates a radical break in the relationships among technologies, subjects, and modes of communication and interaction (2012: 5–7). In each case, major shifts in the production or distribution of goods involves an ensuing shift in the commodity form.

Rather than add apps to a list of ideal-type commodities – though they do seem to function as such – we instead want to examine what shifts to software's commodity form, as apps, tell us about changing notions of software, media consumption and digital goods more generally. We argue apps represent an ideal delivery mechanism for what we are calling *mundane software*: software that spreads out beyond the computer and into a vast range of everyday routines and activity. Mundane software is mundane because it is relatively unremarkable: a to-do list app, a bird-watching guide, an app that mimics the sound of a zipper, etc. This is not to suggest mundane means useless or boring. Mundane software can be necessary, affective, and enjoyable, even if it is for everyday tasks like doing groceries or tracking your exercise. But given the diversity and wide-range of functionality apps provide that seem to defy categorisation or unification, we suggest their mundaneness is precisely what allows for the incorporation of software into a range of everyday practices. This is also not to suggest mundane software is coextensive with apps; there have been mundane programs before apps just as there will likely be mundane software in future iterations of the software commodity. Rather, mundane refers to the everydayness of the features and functions of the software, the unremarkability of the contexts in which it gets deployed, and the very material ways it insinuates itself into banal routines. The favourable economics of production and distribution behind apps, as well as the variety of devices to which they can be extended make apps a particularly effective commodity form for presenting, distributing and using mundane software.

Previous research in management and organisational studies has primarily focused on enterprise software (Rettig, 2007) while media studies has analysed video games social media and other communication programs (Consalvo and Dutton, 2006; Dyer-Witheford

and de Peuter, 2005; Murthy, 2013; Wolf and Perron, 2003). The recent turn to software studies has diversified the kinds of software under analysis (i.e. Chun 2013; Montfort and Bogost, 2009; Fuller, 2008), but there are still few frameworks for addressing the multiplicity of software products available as apps. Juul's (2010) work on casual games and Anable's (2013) ideas around small games address some of the more mundane encounters users have with software, noting that it is not so much content that makes games casual, but features, functions, and context of use (Juul, 2010: 50). Similarly, we do not define mundane software strictly by its content, but rather as an intersection of its form, mode of delivery, the functions it provides (or, in some cases doesn't), and the everydayness of practices to which it can be applied. Games can be mundane, just as productivity applications can be. But mundane also points us to hundreds of genres and styles of applications that do not fit easily within traditional definitions like 'games' or 'media software' or 'enterprise applications'. Mundane software is a way to make sense of the explosion in the types and purposes of software apps represent; a way of finding commonalities in software that individually seem insignificant, but collectively, represent a significant cultural platform for encountering software. Apps represent a particularly effective commodity form for selling and distributing mundane software, for packaging otherwise unremarkable and routine tasks into problems solvable by software.

A History of the Software Commodity

Given the seeming abundance of apps as a now-dominant mode for software delivery, it is easy to forget that software has not always been a distinct commodity. In the early 1960s, computers were so large, expensive and technically unwieldy, that they either came with programs installed by the manufacturer, or contract programmers were hired to create highly specific applications for completing a task (Campbell-Kelly, 2003; Johnson, 2002). Even though IBM and other manufacturers had been using punch cards in ways that resembled later methods of selling software, the future of software as a distinct sector was not evident (Johnson, 2002; Pugh, 2002). Software's viability as a commodity was also challenged by a vibrant hobbyist scene that freely traded code and programs in an effort to build a public knowledge base for working with computers. Hobbyist groups like DECUS or SHARE pooled resources and skills to avoid the per-instruction fee many manufacturers charged users for programming support. [1] There was, in short, little market for software in the early 1960s and little impetus to treat it as an independent commodity (Johnson, 2002: 14).

In 1969, IBM announced its intent to 'unbundle' its software from its hardware. The company divided its programming business into two categories: Systems Control Programming (i.e. free maintenance of IBM operating systems) and Program Products

(i.e. language compilers, conversion aid programs, general purpose utilities, and industry applications) which would be offered under a license agreement (IBM, 1969: 9). Initially a response to a pending antitrust suit, IBM's unbundling helped turn a product that previously seemed unsellable into a value-generating commodity. Given its clout at the time, the IBM announcement is often cited as the origin point of the software industry. More accurately, IBM's decision crystallised a number of practices already taking place in the late 1960s (Campbell-Kelly, 2002; Pugh, 2002) at companies like Applied Data Research (that had developed and released a flowcharting program Autoflow in 1965) and International Computer Program (that published a catalog throughout the 1960s of software packages and vendors) (Campbell-Kelly, 2003: 99).

The software sector grew in the 1970s but it remained largely limited to corporate and organisational uses. The exception, of course, was games. As Dyer-Witheford and DePeuter note, it was in the early 1970s that the roots of gaming emerged as hacks or subversions of military-purposed software (2005: 7–10). Freed from their original military-industrial context, and not originally considered commodities, games later served as ideal-type commodities; they were key sources of growth for a global industry of gaming companies. With roots in hacker culture but also as multi-billion dollar industry, games represent both a threat to, and the epitome of, informational capitalism (Kline, et al., 2003; Dyer-Witheford and de Peuter, 2005). Games, like software more generally, continue to waffle between these poles, between free, shared and hacked good, as well as a packaged commodity (Dyer-Witheford and de Peuter, 2005: 32). Software is simultaneously a commodity and an inspiration for all kinds of hacks, mods, and open source ideas that question the regime of value associated with software. In this light, the move towards apps signifies a concerted effort to capitalise on the perception of software as free and to integrate conceptions of "free" into the very commodification process. In stark contrast to free and open source software movements for desktop applications, apps are content to push free as in free beer, rather than free speech.

As personal computing matured, and games started migrating to computers, a corresponding infrastructure for software products was gradually established. Early word processing programs, for example, which were sold by mail order in the mid 1970s were, by the end of the decade, available in retail franchises like ComputerLand, and Software City (Campbell-Kelly, 2003: 209). The period from 1975–1983 saw a dramatic rise in personal computer software, culminating with around 35,000 programs from about 3,000 vendors in 1983 (2003: 208). These included categories like productivity (e.g. spreadsheets, word processors and database programs), industry (e.g. ledger and accounting programs), and consumer and educational applications (e.g. games, typing tutorials, home accounting/management software), which were available between $50 and $500 as disks in shrink-wrap packaged boxes (2003: 208). This relative explosion

in the kinds of software being developed and its capabilities played a significant role in 'personalising' the personal computer revolution (Friedman, 2005; Turner, 2006; Streeter, 2010). This software was not yet what we define as mundane above, but it was moving beyond enterprise and engineering uses, and beyond video games. These new kinds of software helped convince users computers could be devices that, beyond making cold calculations, could help with making grocery lists, home accounting or entertaining children.

This marked shift in the market for software was, fundamentally, 'a moment when contemporary computer technology created a business opportunity for a new mode of software delivery' (Campbell-Kelly, 2003: 3). Given the openness of the personal computer to third party applications, it was a 'gold-rush' period for software development and helped industrialise particular roles within the software industry, like developers, publishers, and retailers (Boudreau, 2012; Campbell-Kelly, 2003: 203, 209). Software's potential as intellectual property also began to be more thoroughly exploited. As personal computing matured through the 1980s, many early software companies failed or were acquired. Mundane software might have flourished had the conditions that created the 'gold rush' continued, but the ensuing years saw much greater barriers to entry for software developers and the growing dominance of a few major publishers and manufacturers (e.g. Microsoft, IBM).

This period also marked the consistent appearance of the word 'app', usually as a shortened version of 'application' meant to conserve space in job postings in computing and IT trade journals (Holwerda, 2011b). Holwerda (2011b) argues that by 1981 'app was already a widespread term [in the tech industry]' and points to software from 1985 that used the term, like the Apple's programming toolkit MacApp, or the Apps menu in Ashton-Tate's office suite Framework II (see also Benson-Allott, 2011: 10). By 1989, tech reporters starting writing about 'killer apps' [2] and through the early 1990s, the term 'app' was regularly used in trade headlines about 'multimedia apps' and 'enterprise apps' (Bozman, 1993: 10; Jarvie, 1991: 83). By the end of the decade, app was creeping into popular usage, traveling beyond tech publications to more general fare, like *USA Today*'s Internet 100 index listing of 'software/apps' companies (Krantz, 1999: 3B).

The rise of the web in the 1990s encouraged another gold rush era for developers, via the growth of web-based applications that ran through a user's browser or were available for direct digital download. Given the web's open architecture the emergence of Flash animation programs, media software, email programs, map applications and casual games once again diversified and expanded the capabilities of software beyond desktop publishing and business applications (Boudreau, 2012). Mundane software was starting to

flourish but it was spread out across a variety of developer sites, aggregation hubs, service platforms and file sharing networks. Even though Bill Gates was predicting a potentially 'disruptive…sea change' in online software delivery thanks to the move toward internet- and cloud-based computing platforms during the mid-to-late 2000s, the web offered few cohesive venues in which to experience the many emerging types of software (Gates qtd. in Linn, 2005).

This changed with the announcement of the iPhone Software Development Kit (SDK) and Apple's App Store in March 2008. The original iPhone and iPod Touch devices (launched in 2007) came pre-loaded with Apple-built software such as apps for weather, stocks, calendar, etc. But Apple, long a company that has relied on third party software development for its platforms, encouraged developers who wanted to build programs for these rapidly popular devices to build "web apps" that could be accessed through the native Internet browser on the devices rather than native apps. However, web apps couldn't make use of any of the core functionality of the devices, and the apps often performed slowly or more poorly compared to Apple's native's apps (Ritchie, 2013). Out of frustration, users and developers quickly found work arounds by "jailbreaking" their iOS devices and installing unsigned code (i.e. third party apps) on them (discussed further below).

Apple's 2008 announcement, then, was as much reactionary as it was innovatory. Part concession to developers, part realisation of an opportunity, the App Store signalled Apple's opening up of its platform to third party developers. In Steve Jobs' keynote launch speech, he characterised the App Store as a boon for iPhone users and software developers alike (Jobs, 2008). In addition to the convenience of the app store for users, Jobs promised that the platform would solve the problem of software developers who wanted to 'reach every iPhone user' but did not have the distributional resources to do so. As Apple had done a few years earlier with music – when users were looking for music across multiple file-sharing sites, competing formats and limited-content e-commerce stores – the App Store aimed to consolidate software distribution and installation. Rather than having to download software from each developer independently, and then go through variable experiences installing programs, users and developers had one point of contact: the App Store. For developers, they traded access to a potential audience of millions for a split of the software revenues (i.e., Apple takes 30% of all software sales made through the store). The App Store officially launched on July 10, 2008, with apps across a variety of genres (e.g. news, travel, health, games, etc.). The event solidified 'app' as its own, discrete form of software delivery, and provided a hub for the distribution and presentation of all kinds of mundane software. Within months, Google launched its Android Market, and RIM announced an 'application storefront' for Blackberry devices (Nuttall, 2008). Microsoft and Amazon later joined the trend, prompting trademark battles over who owned the term 'app store' (Holwerda, 2011a). [3]

As the central unit of transaction and display in the store, 'app' became more than just a piece of industry lingo; it was a culturally coherent explanation for a new kind of software delivery and distribution. While third party software for mobiles existed before the App Store (as we discuss below), the App Store helped regularise apps as part of the user's experience of those devices. The word 'app' is as much a triumph of marketing as a technical designation (Holwerda, 2011b). A term formerly used as an exigency of publishing has become a thing in itself. 'App' has mutated from industry jargon into the latest iteration of the software commodity imbued with a new sense of software's values, uses, and functions.

So what does the app tell us about the current role of software in everyday life? Bogost argues it is not simply an abbreviated or slang version of the term application, but 'it's also the application itself that's shortened and slanged' (2011). Apps are apps, rather than applications, because they splinter the software commodity into various sellable components and because they break down previously multi-purpose programs into an array of individual, specialised functions. This is partly a business strategy; app stores provide a new market for low priced software and a unified platform for finding and installing software. Apps aim to be the killer app for distributing software. But apps also signify a re-thinking of software's more essential characteristics. Apps present microcosms of designs, practices, and experiences that extend software's capabilities and embed it, and the opportunities to purchase it, ever deeper into everyday routines. They represent an echo of the early 1980s and the late 1990s, when hundreds of new kinds of software (and developers) emerged. But they also limit and control software's long-standing status as a free and hacked good and give it a more standardised articulation as commodity. Apps represent a shift in the software's commodity form, and a shift in the meaning and definition of users' relationships with software.

The term app, then, is more than mere abbreviation. It is also more than what a rigid definition based on size (e.g a relatively small self-contained program), on device (e.g. available on smartphones, tablets and other such hardware), on acquisition method (e.g. for download from one or more of the app stores), or on purpose (e.g. for specialised tasks like playing a game, checking the weather, writing a list, etc.) could provide. Rather, we define app as a moment in the history of the software commodity when the form, distribution model and economics behind software production have shifted to encourage a proliferation of mundane software and an intensified integration of software into everyday routines.

The App Commodity

I Am Rich stands out in the app store because of how exceptional its price is relative to other apps. It was, at least in the iOS Store, the maximum price at which an app could be sold. Compared to earlier forms of software, apps are supposed to be small, cheap, and plentiful. Like gumballs in the dispenser machines at malls or grocery store exits, apps are trivial, easily consumable, and designed for impulse purchases. Apps fracture the price of software yet offer no standard price, though the most successful are only a few dollars, or free. By countering this trend, by playing with notions of value, price and use, *I Am Rich* asks the same question currently facing a host of software developers, publishers and retailers in these new venues: What precisely is the form of the software commodity and exactly how much is it worth? With the longer history of the software commodity in mind, we turn now to examine a series of apps that, like *I Am Rich*, may not answer this question directly, but speak toward larger shifts taking place in software's commodity form. We focus on four key vectors that characterise the transition to apps: micro-control, micro-transactions, micro-functionality and micro-materiality. Although these have, arguably, shaped software since the mid 1960s, we use the prefix 'micro' to suggest what is novel about apps is not only their small and mundane nature, but how the shifts taking place at the micro level have allowed apps to excel at commodifying practices that were previously outside the realm of software, and established a commodity form that increasingly governs the selling of other cultural goods and digital media as well.

Micro-control

The Cydia app, which is not available in any official app store, is 'an alternative to Apple's App Store for 'jailbroken' devices, [...] specialising in the distribution of all that is not an "app"' (Freeman, 2011). [4] In this case, 'all that is not an app' refers to software that Apple or Google will not allow in their stores. Jay Freeman, the program's developer, claims it has been installed on over 30 million jailbroken devices since its launch in March 2009 (Freeman, 2011). The Cydia store generated close to $10 million and netted Freeman a personal profit of about $250,000 in 2011 (Shapira, 2011). Cydia is not the first or the only alternative method for installing and finding apps; the Installer.app offered similar functionality in 2007, a full year before Apple's iOS Store was announced, while Rock Your Phone and other similar stores launched shortly after Cydia (Shapira, 2011). These 'apps' function like regular apps, though they can usually only be accessed after installing special software to jailbreak or bypass the manufacturer's (i.e. Apple, Google) restrictions.

The Cydia store, the apps it displays and sells, and the jailbroken devices that depend on such alternative stores, are all responses to what some users and developers see as too much control imposed by manufacturers like Apple over their devices. For some, the "appification" of software (and the web more broadly) signals a new kind of computing that moves away from generative and open platforms (i.e. the PC, the web) to more closed and tethered relationships (Zittrain, 2008). Although third party access development has generated millions of apps, app stores still employ a variety of different micro-control strategies for coordinating how developers can configure, present and sell their apps, through code suites, SDKs, and other developer literature. Beyond this, some app stores exert even further control through the outright refusal or banning of certain apps. Apple, for example, has a lengthy list of reasons for rejecting apps, including pornography or other objectionable material, apps that don't work or conform to Apple's design guidelines, copycat apps, or apps that directly mimic or compete with Apple's own software (see for example the stories of Kafasis, 2008; Ojeda-Zapata, 2008). Google aims to be more accommodating by making its Android software platform more open source and accessible than Apple's iOS, but their Play store still enforces a thorough app content policy and bans apps that fail to adhere to it. For those that do get published in the stores, developers are then beholden to a revenue split with the retailers – usually around 30% of an app's retail price – and any use of in-app transactions must be done through the store (thus, giving the retailers a cut of micro-transactions as well).

Through these kinds of micro-control, app stores represent a fusion of the role of software publisher and retailer. As with the other shifts we examine, this is not entirely novel; publishers and retailers have previously exerted similar amounts of control over the production and sale of software. Nintendo, for example, as a reaction to the crash in the games industry in the early 1980s pioneered techniques to control who was able to manufacture games for their platform (Campbell-Kelly, 2003: 285; Dyer-Witheford and de Peuter, 2005: 85; O'Donnell, 2009). Control, in this instance, was largely focused on ensuring that software met particular technical considerations, rather than aesthetic ones, though hackers and modders quickly and regularly found ways to circumvent the protection measures (Dyer-Witheford and de Peuter, 2005: 85). The difference with micro-control is that, as both publisher and retailer, the manufacturers behind the app stores shape the functions, features, as well as the look and feel of the apps that appear on its platforms. The app stores control the way in which payments take place, the kinds of goods that can be purchased, and the features of the goods being purchased.

The conflation of the publisher and retailer positions app stores as key intermediaries in the presentation and sale of software. It also allows the manufacturers behind the app stores to divest themselves from a significant amount of risk that accompanies the production of software. By controlling the platform on which content is built and the outlet

through which that content is sold, Apple, Google and other app store owners invest relatively little in app software development and succeed in presenting a wide variety of apps. App stores profit despite how many failed apps populate the stores, because any success or failure still renders the manufacturer a portion of the profits. App store owners can also integrate the more successful software ideas they witness into their own products. Apple, for example, borrows from – and then obviates the need for – many apps with each update (i.e. they have in recent iterations integrated flashlight functionality, voice messaging, and disappearing messages, all of which originated in other apps like *Flashlight*, *What's App* and *SnapChat*). The question is less about whether Apple is wrong to borrow functionality it sees as successful in the apps that populate its store, but that developers take on the cost of production and testing themselves, only to be potentially undermined by the owners of the platform.

This arrangement – the exchange of developer labour for the promise of future potential success through exposure to an audience much larger than that which developers could reach independently – exemplifies what Kuehn and Corrigan describe as hope labour (2013: 9). The aspirations of developers, in this relationship, create a powerful enough dynamic to warrant shifting the costs and risks of development onto these individual hope labourers (Kuehen and Corrigan, 2013: 21). App stores and the various app-building programs (e.g. AppMakr, iBuildApp, Good Barber, AppMachine) that allow users to design their own apps with little or no knowledge of programming all preach a rhetoric that anyone can build an app, distribute it and find fortune in doing so. Apple, for example, claims that over 90% of all its apps are downloaded at least once a month, and refers in its press releases to testimonials from independent developers who've gone from amateur coders to globally successful entrepreneurs (Apple, 2013). Proof of 'long-tail' economics in action (Anderson, 2006), Apple uses these numbers to incentivise its growing pool of developers who hope for the success of their app, and find comfort in the knowledge that even in a marketplace of millions, the majority of apps are being tested and tried (Lee, 2013).

But this promise of democratised software development is undercut by banned apps and by the economics of app stores, which favour larger companies with a healthy portfolio of software. It also stands in contrast to other research that suggests that 68% of smartphone owners used five or fewer apps on a weekly basis, with many purchases being forgotten almost instantly (Purcell, et al., 2010; Austin, 2013). Some analytics firms even suggest that a majority (close to 60%) of the apps in the app stores are 'zombie apps' – apps that receive a download or two every now and then, but 'never appear in any of the thousands of charts published by Apple' or other app stores (Lee, 2013). [5] Although it's hard to know precisely how many zombie apps might exist, it is clear that the ability to get noticed on the app store is increasingly a difficult task. As a result, 'only 2% of the top 250 publishers in

Apple's App Store [and 3% in Google's Play store] are "newcomers"' (Rubin, 2013).

So while app stores may have sparked the creation of thousands of pieces of software, they have not entirely shifted the balance of power in the software industries. Only a small percentage of app developers (12%) are successful while a large number (34%) make less than $15,000. Developing products for app stores relies on a significant amount of hope labour and suggests that long tail theory of cultural consumption is not necessarily economically viable for many at the long end of the tail. There's been little change in the developer profile as well, which remains overwhelmingly male (94%) and American (54%) (Austin, 2013). Cydia, and various jailbreak tool kits, attempt to counteract some of the instances of micro-control; they list any app in their stores and do so in a much quicker period of time than the official stores. Because there aren't as many users with jailbroken phones, these stores also promise better visibility for developers and their apps. However, these alternatives to more closed and controlled systems still take 30% of the sale of any app. They also serve as a grey market space for testing innovation for manufacturers. Many of the features that emerge on the Cydia store, for example, serve as the basis for future versions of Apple's various software and hardware capabilities, like the ability to record video, to send pictures via text message, to customise home screens, etc. (Heath, 2012, Wortham, 2009). In other words, even though these alternate stores (or, we would argue, open source platforms like the Ubuntu app directory) attempt to specialise 'in the distribution of all that is not an app', they still exert micro-control and are subject to the commodity constraints that come with apps.

Micro-transactions

This level of control over the store manifests itself at the more immediate level of the transaction. *69 Positions* is a racy "lifestyle" app that offers descriptions and silhouette illustrations of sex positions to help you 'brush up on all the classics, while adding many new and exciting positions to your repertoire'. The app – which reportedly has 7 million downloads – was created by college sophomore Michael Karr, who had little prior knowledge of software programming but had gathered the positions 'from research and personal experiences' (Slattery, 2011). Despite the name, the full version of the app, available for 99¢, offers 113 positions. The free, lite version has 26 positions.

Karr's app is one of hundreds of apps that offer in-app purchases. Linked to a credit card account, in-app purchases allow users to make contextual purchases from within apps without having to exit the software (in Karr's case, the in-app upgrade unlocks

more positions). Other apps feature much more complex and layered in-app purchases: new levels or weapons (e.g. *Lego Star Wars*), maps (e.g. *Garmin*), yearly subscriptions to fitness coaching advice (e.g. *Argus)*. Apple's developer SDK defines four types of in-app purchases: consumable (e.g. extra health points or experience points), non-consumable (e.g. extra chapters in a book, extra songs on a music app), non-renewing subscriptions (e.g. limited term purchases or location service subscriptions) and auto-renewable subscriptions (e.g. newspapers and magazines). Each of these micro-transactions integrates the purchase process directly into the software experience. Apps represent a commodity form that seamlessly blends commerce and use.

Apps are micro-transactional entities most obviously because they are largely available for a small price. In 2012, the average price paid for an app was just over $3.00 (Austin, 2013), though if free apps are factored in, this number drops drastically to less than 20¢ (Gordon, 2013). The low price opens up a previously underserved market for software; only the most organised shopper would pay $30 - $50 for a grocery list application in boxed software, but at 99¢, software becomes a potential solution for this kind of mundane task. More significantly, apps are micro-transactional because they parcel up previously physical commodities into their component pieces and disperse the purchase process across multiple payment opportunities. Rather than sell a game or software product as a finished and completed good, in this case, the content is divided up into multiple parts, each of which can be assigned its own cost or own manner of access. Apps are not the first expression of software to rely on upgrades, subscriptions and iterative purchases, but the nature of these micro-transactions splices the software commodity in much finer-grained saleable layers than previously possible. Just as digitisation has amplified the disaggregation of the music album into various singles (Bogost, 2011), or even into single musical bed tracks (e.g. Radiohead's 'Nude' remix contest), apps break software up into its most microscopic expressions of value.

This hyper-versioning strategy further establishes software as part of what Vincent Mosco calls a 'pay-per' society, where media and communications commodities are continually parsed into seemingly infinite iterations (1989). Take, for example, the *Biophilia* app, which included songs and games related to Bjork's 2011 album of the same name. Originally listed as a free app that came with one song/game, the app later sold individual songs for $1.99 each, before eventually listing the app at $12.99 for all the songs. The staggered release schedule meant that those who bought the tracks as soon as they were released later had to pay for the full app, despite already owning 3 or 4 songs. This kind of repeat purchasing is not uncommon. Apple's support page for in-app purchases, for example, notes how in-app purchases do not necessarily equate to full ownership: 'you can't sync or transfer non-renewing subscriptions and consumable In-App Purchase to another iOS device' and if you lose or accidentally delete apps, you would also lose in-app purchases (Apple, 2014).

This on-going tethered relationship between the user and the software allows producers to extract more value from the same commodity and to exert greater control over the distribution of software and its contents (Zittrain, 2008). Micro-transactions may open up new ways of selling software but they also offer new ways of controlling its flow.

This is true even for free apps which make up close to 90% of all apps (Gordon, 2013). The process for obtaining free apps is transactionally similar to the process for obtaining paid apps; users require an account and download the software in the same ways. Free apps act like advertisements for other paid apps, or like shareware that came before it. But by integrating free software as a legitimate and significant part of the software buying process, the app stores integrate free goods into the larger sphere of commodity purchases. Free apps set the ground for freemium models (where the initial download is free and extra services/features are purchased), and even apps that remain free for the duration of their use rely on data collection, advertising and other very non-free forms of surveillance and monitoring (Andrejevic, 2013). In other words, micro-transactions do not simply raise issues about the value and worth of the software commodity, they also incorporate 'free' goods into a commodified transaction.

Micro–functionality

The *Yo* app describes itself as the 'simplest & most efficient communication tool in the world'. It allows users to send a text alert to friends that simply says 'yo'. Reviews of the app declared it the 'dumbest' or 'stupidest' app of all time, and 'easily the worst piece of software' even as they admitted to being slightly enamoured with the app's simplicity in an era of information overload (Cush, 2014; Shapiro, 2014; Shontell, 2014). While the app has inspired philosophical reviews about its deeper social and cultural meanings, *Yo* is not a meta-commentary on the changing nature of communication in the digital age. Rather, it is doing what so many other apps do regularly: distilling from the complex routines, patterns, and activities of everyday life partitionable processes that can be converted into software solutions. More than anything else, *Yo* is an example of micro-functional software: single or limited-purpose apps that split up and break down complex activities and assign them each a new technology to help.

Micro-functionality is related to micro-transactions. Given the constant ability to update, and the low price point for many apps, developing an app that has small and mundane functions is economically more feasible than what was previously possible in the era of boxed software. While 'home management software' in the 1980s that allowed computers

to run home appliances, operate lighting, and control security devices and other mundane software existed, the genre 'sunk without a trace' in the 1980s since the market cohered around more successful products, like word processors (Campbell-Kelly, 2003: 226). But now with apps, as Bogost notes, multi-function software suites are giving way to programs that are 'purpose-built for a specific function… or just as often, for no function at all' (Bogost, 2011, n.p.). While there are plenty of multi-function apps available, apps have also fostered an abundance of software programs designed to meet minor, micro and mundane tasks. This is partly what is at work in apps like *Hold On*, an app that lets users put their finger on a button to see how long they can hold it for, or *Taxi Hold 'Em*, a button-free app helps users hail a cab by displaying a flashing, yellow 'TAXI' sign and producing a whistling sound. Apple's marketing slogan – 'There's an App for That' – is not just a self-congratulatory tagline but a condensed manifesto for a world in which all problems have solutions in software.

The kinds of devices on which apps currently work also further the development of micro-functionality. Apps are spreading out to watches, TVs, cars, fridges, and a variety of other devices, as these devices start to provide the ability to check the weather, play music, or connect to social networks (Cheng, 2012). With 'smart appliances' fuelling a wider discussion on 'ubiquitous computing' and the 'Internet of Things' (see for example Wasik, 2013), apps emerge as the central interface through which these science fictions meet the consumer in the realm of the possible. Futurists envision scenarios where fridges can automatically replenish groceries, or homes can light and heat themselves based on the movements of their occupants (Wasik, 2013), but these fantasies rely on the micro-functionality of a variety of context dependent software and devices. If ubiquitous computing is the next step in computing, mundane software is what it will run on.

Micro-materiality

As developers see apps as a successful emerging model for the distribution of the software commodity, they begin reimagining how other interactions with software and digital media can take place. *Coast*, an app produced by Opera Software, highlights a phenomenon tech critics call "appification", where the functional and design aspects of apps become increasingly central to user experiences with all kinds of software and technology (Aggarwal, 2014; Hay, 2010). *Coast* 'appifies' mobile web browsing by offering a visual interface that presents different sites and platforms like YouTube, Facebook, and Gmail as individual, app-like icons. As one report suggests, *Coast* 'feels more like a dedicated content consumption app than a Web browser' (Campbell, 2014). Opera's app relies heavily on a kind of app aesthetic as a primary design strategy for its service. It suggests apps

aren't just limited to apps, but they represent a larger mode of organising and presenting information.

Coast's attempts to make web browsing more like using apps points to the micro-materiality of the software commodity. Given its composition as code and information, software has long hovered between immaterial and material good. Despite their seeming inconsequentiality and evanescence though, apps express their materiality through their underlying code, their interface design, and their use of hardware affordances to provide particular haptic and physical experiences. Rather than immaterial, apps are instead, like software more generally, 'differently material, tenuously material, almost less materially material' (Berry, 2012, n.p.).

As the building blocks of function and design, the app's code (usually in programming languages like Objective-C or Java), is the most basic element of its materiality. Galloway's (2004) dictum that "Code = Praxis" reminds us that, while SDKs and programming suites (such as Apple's xCode) promise to make software development for mobile distribution platforms both relatively easy and open, they also standardise and control the kinds of functions software can provide and shape the aesthetics of the finished product. Developer guides from Apple and Google highlight the expectations the companies have for any program that will be available through their platform, and these stated expectations are often embedded in the tools that accompany the SDK.

The code's materiality manifests in an app's interface and design, both of which are crucial elements of app development. Apps appear as icons on the home screens of devices which follow the icons-on-desktop graphical user interface familiar to computing, but their chicklet-sized covers and square rounded edges are evidence of an influential design strategy that has, as the example of *Coast* suggests, moved beyond app stores to web designers more broadly. The app icon gives way to the app's own interface, which is a combination of developers' desire to establish their own style while still working within the confines of a larger pre-coded environment. There are, of course, distinctions between the various app platforms. Google, for example, follows a 'simple, beautiful, useful,' cross-platform aesthetic (Sapnar Ankerson, 2014) while Microsoft relies on multicoloured, differently sized, dynamically changing app icons. Developers must adjust their applications to suit the aesthetics of each store and to account for the particular affordances that occur across devices (e.g. smartphones, tablets, game consoles, TVs, etc.). These micro-material aesthetic decisions are no small matter; recent reports suggest that design matters as much as price, usability, and brand loyalty in determining app use (Bjoran, 2010), and that an app's icon has a significant effect on user psychology when it comes to finding and purchasing apps (Kai-Chun and Chen-Heng, 2013).

The micro-materiality of apps is also evident in the hardware platforms on which they run. While most apps invite material interaction through touchscreen taps, swipes, or voice recognition, some apps draw on other affordances to make unintended uses of a phone or tablet's hardware. For example, apps that turn your phone into a flashlight (using either the camera's flash or the phone's display) are common across all platforms and apps like *Pocket Heat*, *High Tech Hand Heater*, or *Handwarmer* activate several CPU functions to heat up the phone. The *Blower* app blows air through an iPhone's speaker to blow out on-screen candles, while games such as *Bubble Blower* deploy the same functionality to create a virtual bubble. By capitalising on the limitations of a mobile phone or tablet's physical capabilities, these apps emphasise the materiality of this form of software.

Mundane apps may seem ephemeral and transient in their form, then, but these examples illustrate how they are part of a much longer chain of materials that stretches from their code to the aesthetics of their interfaces to the interactions with their hardware. This chain does not stop simply at the material interactions with the software we list above. Brophy and de Peuter's (2014) recent work, for example, examines the circuits of labour that coalesce to create the various materials that make up the digital devices that supply apps. They trace the flow of materials from the extraction of precious materials and metals like coltan in militia-led mines in countries like Congo to the assembly materials and questionable labour practices in Chinese electronics factories to the immaterial labour of the vast network of call centre support workers, as well as the materials for coding and creating among dispersed developers (Brophy and De Peuter, 2014). Similar to the activist app *Phone Story*, their argument makes clear that underneath the sheen and the minimalist design of devices that seem to deny their materials in a quest for lightness, smallness and speediness lies a very real material path (or a path of materials). Apps remind us that the interfaces with which we interact depend heavily on the materiality of their platforms.

The App Imaginary

In an art exhibit curated by Paul Miller and Svitlana Matviyenko (2013), various artists, technologists and other contributors present a series of 'Imaginary Apps': apps that respond to the question 'What is the most desirable, terrifying, smart, ridiculous, or necessary app that has not been and, possibly, will never be released?'. One app, called *The Ultimate App*, describes itself as follows:

One tap launches *The Ultimate App*, immediately initiating an in-session payment. Upon completion of the payment, the app elegantly closes itself. With seamless in-app payments,

you can be finished using the app in seconds. No fuss, no muss - an app that performs its own finitude quicker than most. [...] For the low per-use cost - only $0.99, €0.99, £0.99 (regionally).

For these imaginary app designers, their artistic intervention boils app usage down to its most essential function, the transaction. *The Ultimate App* is a commentary on how, stripped of some of their other dressings and distractions, apps are an incredibly efficient form for the delivery of mundane digital commodities. Apps split software up into multiple component pieces, into micro-functions that extend the layers at which transactions and purchases can occur. Through their micro-materials they instill a particular affective relationship with software that is highly personalised and individualised, and extends the potential uses of software into a host of mundane activities. Their incorporation of micro-transactions allow for an iterative commodity (i.e. one that keeps supplying purchase opportunities) that legitimises and captures 'free' into an economics of the transaction. Software, in this case, may have little or no price but it still operates according to logics of commodification and micro-control. *The Ultimate App*'s embrace of the transaction as both routine and essential speaks to how significantly mundane software in the form of apps has altered the shape of software's commodity form.

If part of what makes apps mundane is a micro-functionality, materiality and transactionality that enables them to be easily and subtly incorporated into daily routines, an app that does nothing takes such mundanity to its logical conclusion. But *The Ultimate App*'s critique of digital capitalism ignores many of the other useful and affective responses users have towards apps, and some of the possibilities apps might represent for extending not only the kinds of software available, but also the kinds of developers engaging in software production. As Anable (2013) notes about casual games, apps can be 'affective systems that mediate relations between players and devices, workers and machines, and images and code (and our feelings about those relations).' Mundane software like the *Yo* app or *Coast* may seem trivial or banal, but it is within the routine, everyday interactions that such programs manifest that software's commodity form is being re-imagined. Software began its history as one with the machines and platforms on which it was available, until developers, manufacturer, publishers and retailers found value in treating it as a distinct commodity. Although software came to represent an ideal-type commodity for informational capitalism, it never fully left its liminal state between priced product and hacked resource. This tension remains, though apps represent new attempts to capture perceptions of free and to put them in service of further commodification.

It is through their specialised functions, their iterative transactions, their novel materialities that apps have been able to so pervasively extend the reach of mundane software. This

has offered new means to control the flow and the shape of the software commodity. It has also helped software spread beyond computers and mobile devices to a whole host of technologies, practices and aesthetics.

Biographical Notes

Jeremy Wade Morris is an Assistant Professor in Communication Arts at the University of Wisconsin-Madison. His research focuses on the digitisation of cultural goods and commodities and it has appeared in *New Media & Society*, *Critical Studies in Media Communication*, and *First Monday*.

Evan Elkins is a doctoral student in Communication Arts at the University of Wisconsin-Madison. His research interests include digital media technologies, media industries, and cultural studies of media globalisation. He is currently completing a dissertation about regional lockout in digital entertainment media technologies.

Notes

[1] DECUS was a support group for DEC computer users who freely shared software under the motto ‘Steal from your Friends’ (Richards, 1965). SHARE was founded in 1954 to discuss how to use, install and market a programming language for IBM Systems called Mark IV(Campbell-Kelly, 2003 31–34)

[2] Killer apps are usually considered to be software that is essential enough to a particular technology or operating system that it could drive sales of that platform. The longer ‘killer application’ had been used to refer to the same idea before 1989.

[3] Following Apple’s long-standing practice of sending cease and desist letters to others using the term ‘app store’ (Holwerda, 2011a), they filed suit against Amazon’s ‘Appstore’. Apple eventually withdrew the suit, but the incident brought a fair amount of discourse regarding the meaning of ‘app’ and who had ownership over it.

[4] Jailbreaking allows users root access to their mobile devices and allows them to alter their phones in ways unintended by the manufacturer. It was originally treated as an illegal activity, and Apple would send out updates that rendered a user's jailbroken phone unusable. Jailbreaking has since been ruled a legal practice.

[5] This study was later called into question for methodological reasons; the company was a firm that sold developers on a service for helping their apps get noticed. It was in their best interest to portray the app store as a chaotic mix of apps with little hope for indie developers to ever get their app seen or noticed.

References

Aggarwal, Raj. '25 Years after Its Birth, the World Wide Web Becomes Appified', *Wired* (2014), 14 March (2014), http://www.wired.com/2014/03/25-years-birth-world-wide-web-becomes-appified/

Anable, Aubrey. 'Casual Games, Time Management, and the Work of Affect', *Ada: A Journal of Gender, New Media and Technology* 6.2 (2013), http://adanewmedia.org/2013/06/issue2-anable/.

Anderson, Chris. *The Long Tail: Why the Future of Business Is Selling Less of More* (New York: Hyperion, 2006).

Andrejevic, Mark. *Ispy: Surveillance and Power in the Interactive Era* (Lawrence, Kans.: University Press of Kansas, 2007).

AppAnnie. 'Digital Content Report 2013: White Paper Global Shift Driven by Dramatic Growth of Apps', App Annie and IHS Technology (2014a).

———. 'Mobile App Advertising and Monetization Trends 2012–2017: The Economics of Free', IDC: International Data Corporation (2014b).

Apple. 'Apple's App Store Marks Historic 50 Billionth Download', (Cupertino, CA: Apple, 2013).

———. 'iPhone App Store Downloads Top 10 Million in First Weekend', (Cupertino, CA: Apple, 2008).

———. 'iTunes Store: About in-App Purchases', Apple, Support Website http://support.apple.com/kb/ht4009

Austin, Scott. 'The Surprising Numbers Behind Apps', *Digits from the Wall Street Journal*, 11 March (2013), http://blogs.wsj.com/digits/2013/03/11/the-surprising-numbers-behind-apps/

Benson-Allott, Caetlin. 'Cinema's New Appendages', *Film Quarterly* 64.4 (2011): 10–11.

Berry, David M. (ed.). *Life in Code and Software: Mediated Life in a Complex Computational Ecology* (England: Open Humanities Press, 2012).

Bjoran, Kristina. 'User Expectations with Mobile Apps – Catching up with Effective UI', 21 December (2010), http://www.uxbooth.com/articles/12207/

Bogost, Ian. 'What Is an App? A Shortened, Slang Application', Personal Blog 12 January (2011), http://www.bogost.com/blog/what_is_an_app.shtml

Boudreau, Kevin J. 'Let a Thousand Flowers Bloom? An Early Look at Large Numbers of Software App Developers and Patterns of Innovation', *Organization Science* 23.5 (2012): 1409–27.

Bozman, Jean S. 'Gupta Offers Tool Kit for Enterprise Apps', *Computerworld*, 18 January (1993): 1.

Brophy, Enda, and Greig De Peuter. 'Labours of Mobility: Communicative Capitalism and the Smartphone Cybertariat', in Andrew Herman, Jan Hadlaw, and Thom Swiss (ed.) *Theories of the Mobile Internet: Materialities and Imaginaries* (New York: Routledge, 2014).

Bruno, Antony. 'Planet of the Apps', *Billboard*, 18 December (2010): 32.

Campbell, Mikey. 'Opera "Coast" for iOS Brings the Enjoyment Back to Mobile Web Browsing', *Apple Insider*, 24 April (2014), http://appleinsider.com/articles/14/04/24/opera-coast-for-ios-brings-back-the-enjoyment-to-mobile-web-browsing

Campbell-Kelly, Martin. *From Airline Reservations to Sonic the Hedgehog: A History of the Software Industry* (Cambridge, Mass.: MIT Press, 2003).

———. 'Software as an Economic Activity', in Ulf Hashagen, Reinhard Keil-Slawik, and Arthur Norberg (eds.) *History of Computing: Software Issues* (Berlin: Springer, 2002), 185–202.

Cheng, Roger. 'How Apps for Your Appliances Represent the Next Opportunity', *CNET*, 16 January (2012), http://www.cnet.com/news/how-apps-for-your-appliances-represent-the-next-opportunity

Chun, Wendy Hui Kyong. *Programmed Visions: Software and Memory* (Cambridge: MIT Press, 2013).

Consalvo, Mia, and Nathan Dutton. 'Game Analysis: Developing a Methodological Toolkit for the Qualitative Study of Games', *Game Studies* 6.1 (2006), http://www.gamestudies.org/0601/articles/consalvo_dutton

Cush, Andy. 'Yo, the World's Dumbest App, Hacked by Three College Students', *Gawker*, 20 June (2014), http://gawker.com/yo-the-worlds-dumbest-app-hacked-by-three-college-stu–1593737429

Dowell, Andrew. 'The Rise of Apps, iPad and Android', *The Wall Street Journal*, 27 December (2010).

Dyer-Witherford, Nick, and Greig de Peuter. *Games of Empire: Global Capitalism and Video Games* (Minneapolis, MN: University of Minnesota Press, 2005).

Freeman, Jay. 'Weclome to Cydia', https://cydia.saurik.com/.

Friedman, Ted. *Electric Dreams: Computers in American Culture* (New York: New York University Press, 2005).

Fuller, Matthew. *Software Studies: A Lexicon* (Cambridge: MIT Press, 2008).

Galloway, Alexander R. *Protocol: How Control Exists after Decentralization* (Cambridge: MIT Press, 2004).

Gartner. 'Predicts 2014: Apps, Personal Cloud and Data Analytics Will Drive New Consumer Interactions', Gartner (Stamford, CT: Gartner, 2014).

Gordon, Mary Ellen. 'The History of App Pricing, and Why Most Apps Are Free', *Flurry*, 18 July (2013), http://www.flurry.com/bid/99013/The-History-of-App-Pricing-And-Why-Most-Apps-Are-Free#.VI8KcCdHZi8

Hay, Timothy. 'The "Appification" of Everything', *Wall Street Journal*, 4 October (2010), http://blogs.wsj.com/digits/2010/10/04/the-appification-of-everything/

Heath, Alex. 'The History of Jailbreaking', *Cult of Mac*, 26 September (2012), http://www.cultofmac.com/192850/the-history-of-jailbreaking-feature/

Holwerda, Thom. 'Apple Threatens Open Source Amahi Project with Legal Action', *OS News*, 22 June (2011a), http://www.osnews.com/story/24872/Apple_Threatens_Open_Source_Amahi_Project_with_Legal_Action

———. 'The History of "App" and the Demise of the Programmer', *OS News*, 24 June (2011b), http://www.osnews.com/story/24882/The_History_of_App_and_the_Demise_of_the_Programmer

IBM. 'New IBM Pricing Policy', (Mountain View, CA: Computer History Museum, 1969).

Ingraham, Nathan. 'Apple Announces 1 Million Apps in the App Store, More Than 1 Billion Songs Played on iTunes Radio', *The Verge*, 22 October (2013), http://www.theverge.com/2013/10/22/4866302/apple-announces–1-million-apps-in-the-app-store

Jarvie, Barbara. 'The Sound of Music - Midi Spurs Crescendo in Multimedia Apps', *Computer Reseller News*, 18 November (1991): 1.

Jobs, Steve. 'Steve Jobs introduces the App store - iPhone SDK Keynote', 13 March (2008), https://www.youtube.com/watch?v=xo9cKe_Fch8

Johnson, Luanne. 'Creating the Software Industry: Recollections of Software Company Founders of the 1960s', *IEEE Annals of the History of Computing* 24.1 (2002): 14–42.

Juul, Jesper. *A Casual Revolution: Reinventing Video Games and Their Players* (Cambridge, Mass.: MIT Press, 2010).

Kafasis, Paul. 'A Bridge Too Far', *O'Reilly Digital Media*, 13 September (2008), http://blogs.oreilly.com/iphone/2008/09/a-bridge-too-far.html

Kai-Chun, Hou, and Ho Chen-Heng. 'A Preliminary Study on Aesthetic of Apps Icon Design', Paper presented at the IASDR 2013 5th International Congress of International Association of Societies of Design Research, Tokyo, 26 August - 30 August (2013).

Kline, Stephen, Nick Dyer-Witheford, and Greig De Peuter. *Digital Play: The Interaction of Technology, Culture, and Marketing* (Montréal: McGill-Queen's University Press, 2003).

Krantz, Matt. 'How the Internet 100 Index Has Been Changed', *USA Today*, 1 October (1999).

Kuehen, Kathleen T. and Thomas F. Corrigan. 'Hope Labor: The Role of Employment Prospects in Online Social Production', *The Political Economy of Communication* 1.1 (2013): 9–25.

Lee, Dave. 'App Store "Full of Zombies" Claim on Apple Anniversary', *BBC*, 10 July (2013), http://www.bbc.com/news/technology–23240971

Lee, Martyn J. *Consumer Culture Reborn: The Cultural Politics of Consumption* (New York: Routledge, 1993).

Lessin, Jessica E. and Spencer E. Ante. 'Apps Rocket toward $25 Billion in Sales', *The Wall Street Journal*, 4 March (2013).

Linn, Allison. 'Microsoft's Gates Warns of Major "Sea Change"', *USA Today*, 8 November (2005).

Marx, Karl. 'Capital. Volume 1, Part 1', in C. Tucker Robert (ed.) *The Marx-Engels Reader* (New York: Norton, 1978), 302–29.

Milian, Mark. 'Apple Removes $1,000 Featureless IPhone Application', *Los Angeles Times*, 7 August (2008).

Miller, Paul and Svitlana Matviyenko. 'The Imaginary App', (London, Ontario: Museum London, 2013).

Montfort, Nick and Ian Bogost. *Racing the Beam: The Atari Video Computer System* (Cambridge, Mass.: MIT Press, 2009).

Mosco, Vincent. *The Pay-Per Society: Computers and Communication in the Information Age* (Toronto, ON: Garamond Press, 1989).

Murthy, Dhiraj. *Twitter: Social Communication in the Twitter Age* (Malden, MA: Polity, 2013).

Nuttall, Chris. 'Rivals Take Aim at IPhone's App Store', *Financial Times*, 21 October (2008).

O'Donnell, Casey. 'Production Protection to Copy(Right) Protection: From the 10NES to DVDs', *IEEE Annals of the History of Computing* 31.3 (July-September 2009 2009): 54–63.

O'Malley, Gavin. '2012 Doubles as Year of the App', *MediaPost: Online Media Daily*, 2 Janu-

ary (2013), http://www.mediapost.com/publications/article/190211/2012-doubles-as-year-of-the-app.html

Ojeda-Zapata, Julio. 'Local Software Developers Struggle to Meet Apple's IPhone Standards,' *Twincities.com*, 21 September (2008), http://www.twincities.com/ci_10516619

Olmstead, Kenneth. 'Mobile Apps Collect Information About Users, with Wide Range of Permissions', Pew Research Center, Press Release, 29 April (2014), http://www.pewresearch.org/fact-tank/2014/04/29/mobile-apps-collect-information-about-users-with-wide-range-of-permissions/

Pugh, E. W. 'Origins of Software Bundling', *IEEE Annals of the History of Computing,* 24.1 (2002): 57–58.

Purcell, Kristen, Roger Entner, and Nichole Henderson. 'The Rise of Apps Culture', PEW Research Center, 14 September (2010), http://www.pewinternet.org/2010/09/14/the-rise-of-apps-culture/

Rettig, Cynthia. 'The Trouble with Enterprise Software', *MIT Sloan Management Review*,* *1 October (2007), http://sloanreview.mit.edu/article/the-trouble-with-enterprise-software/

Richards, Mark. '"Steal from Your Friends" Decus Button', Computer History Museum (1965), http://www.computerhistory.org/revolution/mainframe-computers/7/172/697

Ritchie, Rene. 'App Store Year Zero: How Unsweetened Web Apps and Unsigned Code Drove the iPhone to an SDK, *iMore*, 10 July (2013), http://www.imore.com/history-app-store-year-zero

Rubin, Ben Fox. 'The Dirty Secret of Apps: Many Go Bust', *The Wall Street Journal*, 7 March (2013).

Sapnar Ankerson, Megan. 'Google's Aesthetic Turn: One Simple Beautiful Useful Google', *Antenna*, 13 January (2014), http://blog.commarts.wisc.edu/2014/01/13/googles-aesthetic-turn-one-simple-beautiful-useful-google/

Shapira, Ian. 'Once the Hobby of Tech Geeks, iPhone Jailbreaking Now a Lucrative Industry', *The Washington Post*, 7 April (2011).

Shapiro, David. 'A "Yo" Is Lovely to Receive', *The New Yorker*, 23 June (2014), http://www.newyorker.com/online/blogs/culture/2014/06/a-yo-is-lovely-to-receive.html

Shontell, Alyson. 'The Inside Story of Yo: How a "Stupid" App Attracted Millions of Dollars and Rocketed to the Top of the App Store', Business Insider, 21 June (2014), http://www.businessinsider.com/the-inside-story-of-yo-there-isnt-actually–1-million-in-the-bank–2014–6

Slattery, Caroline. 'Michael Karr: Creator of 69 Positions, a #1 Sex App', *SMU: Her Campus Media*, 15 November (2011), http://www.hercampus.com/school/smu/michael-karr-creator–69-positions–1-sex-app–0

Smith, Cassidy. '2013: The Year of the App in Review', *StartApp*, 10 January (2014), https://web.archive.org/web/20140705060218/http://blog.startapp.com/2013-year-app-review/

Snickars, Pelle, and Patrick Vonderau. *Moving Data: The iPhone and the Future of Media* (New York: Columbia University Press, 2012).

Streeter, Thomas. *The Net Effect: Romanticism, Capitalism, and the Internet* (New York: NYU Press, 2010).

Turner, Fred. *From Counterculture to Cyberculture: Stewart Brand, the Whole Earth Network, and the Rise of Digital Utopianism* (Chicago: University of Chicago Press, 2006).

Veblen, Thorstein. *The Theory of the Leisure Class. 1899* (New York: A.M. Kelley, Bookseller, 1965).

Wasik, Bill. 'In the Programmable World, All Our Objects Will Act as One', *Wired*, 14 May (2013), http://www.wired.com/2013/05/internet-of-things–2/all/

Wolf, Mark J.P. and Bernard Perron. *The Video Game Theory Reader* (New York: Routledge, 2003).

Wortham, Jenna. 'Unofficial Software Incurs Apple's Wrath', *The New York Times*, 13 May (2009): 1.

Yared, Peter. '2012: The Year of the App-Ocalypse?', *CNET* 23 March (2012), http://www.cnet.com/news/2012-the-year-of-the-app-ocalypse/

Zittrain, Jonathan. *The Future of the Internet and How to Stop It* (New Haven, CT: Yale University Press, 2008).