The Death of Flash

A Research Paper submitted to the Department of Engineering and Society

Presented to the Faculty of the School of Engineering and Applied Science
University of Virginia • Charlottesville, Virginia

In Partial Fulfillment of the Requirements for the Degree
Bachelor of Science, School of Engineering

Calvin Kuo

Spring 2023

On my honor as a University student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments.

ADVISORS

Joshua Earle, Department of Engineering and Society

1. Introduction

In computer science, some technologies have succeeded and become ubiquitous standards, while others saw even their own backers give up and abandon them. The internet provides a particularly interesting example of this, since servers and clients must mutually understand a protocol for it to be useful. Some of these technologies have so far had staying power (such as IP[1], HTTP[2], HTML[3], CSS[4], JS[5]), while others have fallen into obscurity (ActiveX[6], Java applets[7], IRC[8], Usenet[9]). Adobe Flash Player was one of the technologies that found itself in the latter category. Unlike other technologies which simply failed to catch on or survive as a mere shadow of their heyday, Flash dominated the web for nearly a decade before being discontinued, with many historical sites that were never updated no longer being accessible.

In order to determine what factors played a role in Flash's demise, this paper analyzes the social and technological factors that initially carried Flash to success, as well as the changes in these factors that led to Flash's decline. In the methods section, I explain the STS methods that I have used, as well as the social groups whose actions I considered. In the background section, I provide a history of the web and the role that Flash played from 1995 to today. In the discussion

[1] The Internet Protocol (IP) defines the rules for transmitting data across the internet. Almost all applications that use the internet ultimately rely on IP at some level.

[2] The Hypertext Transfer Protocol (HTTP) is an application of the internet that focuses on transferring documents that link to each other.

[3] HTML5 is the latest version of HTML (Hypertext Markup Language), which describes the content of a web page. It introduces new features such as videos and canvases.

[4] Cascading Style Sheets (CSS) describes the layout and styling of elements with a web page, such as colors, borders, backgrounds, and positioning.

[5] JavaScript (JS) is a programming language that runs in web browsers and is what makes many websites interactive.

[6] ActiveX was a software framework developed by Microsoft that ran in the Internet Explorer browser. It was discontinued in 2013.

[7] Java applets were a way to run Java code within a web browser using a plugin. They no longer run in web browsers as of 2017.

[8] Internet Relay Chat (IRC) is an instant messaging system. It has declined in popularity with the rise of alternatives such as AOL Instant Messenger (AIM), Skype, Facebook Messenger, Slack, Discord, and other social media.

[9] Usenet is a distributed discussion system where users can subscribe to newsgroups and read and post messages. It has declined in popularity with the rise of bulletin board systems (BBS), forums, Reddit, and other social media.

section, I comment on how Flash suffered from technological stasis, how Apple's "walled garden" philosophy affected Flash, and how the internet has shifted from being more distributed to more centralized. In the conclusion, I address the ultimate cause of Flash's decline and what steps Adobe might have been able to take to prevent it.

2. Methods

2.1 Social construction of technology

To determine what the underlying causes of Flash's success and subsequent decline, I chose to primarily use the social construction of technology (SCOT) approach of Bijker, Hughes, and Pinch (2012). Social constructivism here means that humans collectively shape technology through consensus. Once a society agrees that a technological problem has been solved, then it is solved, and scientists and engineers will move on to other problems. This process is called technological closure.

SCOT contrasts with technological determinism, which argues the opposite, stating instead that technological evolution is inevitable. Marx and Smith described determinism as how "advancing technology has a steadily growing, well-nigh irresistible power to determine the course of events" (1994). In this theory, superior technology outcompetes inferior technology, and as a result, societies naturally come to adapt improved technologies.

In this paper, I use SCOT to analyze how Flash Player initially achieved technological closure and market dominance, as well as the factors that caused the tables to turn on Flash, ultimately leading to its discontinuation.

2.2 Other methodologies

In addition to SCOT, I chose to analyze Flash's rise and fall through the lenses of social influence, technical code, and technological stasis. The role of social influence is particularly important due to the relationship between servers and their many clients, especially as it pertains to virality and trends (Vannoy & Palvia, 2010). The technical code perspective on SCOT, extended by Flanagin et al., emphasizes how the values in the design of the system influence the social outcomes of the system. The technological stasis perspective used by Plennert examines how stagnation can occur with standards (2009; 2018).

To guide my research, I began with the article "The rise and fall of Flash, the annoying plugin that shaped the modern web" by Will Bedingfield and the book *Flash: Building the Interactive Web* by Anastasia Salter and John Murray (2019; 2014). These led me to various historical events and turning points in the history of Flash, which I then corroborated with other primary and secondary sources. Additionally, I have made extensive use of market share data collected by various groups including Statcounter to understand the makeup of internet users and how those trends changed over time.

2.3 Relevant social groups

In this section, I identify the relevant social groups, as they are the ones who must come to a consensus that the technological problem has been solved in the SCOT framework. We can divide the internet users during the heyday and decline of Flash up into several groups based on the following characteristics: device type, operating system, and web browser. Device type refers to the physical form factor of the internet-connected device, such as desktop, laptop, phone, and tablet. Operating system refers to the software that the device runs, such as Windows, Mac, iOS, and Android. Web browser specifically refers to a program used to browse the internet, such as Internet Explorer, Mozilla Firefox, Google Chrome, and Safari. These comprise the major device categories, operating systems, and web browsers during this time period (Statcounter, 2022).

Instead of considering end users directly, I view their actions through the lens of the developers targeting them, divided into three major groups. The first group is the people at Microsoft, Google, and Apple who worked on these web browsers (Internet Explorer, Mozilla Firefox, Google Chrome, and Safari) and operating systems (Windows, Mac, iOS, and Android). The second group is the developers who created the websites that people were visiting, primarily in the commercial, enterprise, and entertainment spheres. The third and final group is the developers at Adobe, who maintained Adobe Flash Player.

3. Background

3.1 The first browser wars

From 1995 to 2001, Netscape Navigator and Internet Explorer (IE) fought against each other in the first browser wars. Netscape had over 80% of the web browser market share when Microsoft released Internet Explorer 1.0 in August 1995 (Borland, 2003). While Navigator was a paid product, Microsoft used their income from Windows to fund IE's development and released IE 2.0 as a free download. At first, both browsers competed fiercely by introducing new features in their respective browsers, often copying each other to provide partially compatible scripting, styling, and plugin[10] support.

Microsoft would begin bundling Internet Explorer with Windows starting in Windows 98 with IE 4.0 (Lash, 1997). Most end users felt no need to install another web browser when their computer already came with one. By 2001, IE reached almost 90% market share as Microsoft released version 6.0 with Windows XP, relegating Netscape to the dustbin of history (Borland, 2003; Baldazo, 2003).

[10] Plugins are pieces of software which add additional features to an existing program. Different browsers may refer to plugins as "add-ons" or "extensions" instead.

Over the next five years, Internet Explorer would receive no major feature updates, as Microsoft struggled to work on the successor to Windows XP. This opportunity would lead to the second browser wars, which began with the launch of Mozilla Firefox, the spiritual successor to Netscape Navigator, in 2004. By 2006, IE6 still controlled over 80% of the market, with Firefox in second place with about 10%, and Apple's Safari in a distant third with about 2% (OneStat.com, 2006; TheCounter.com, 2006).

3.2 Flash Player

In May 1996, FutureWave Software released a browser plugin called FutureSplash Animator, which was compatible with both Netscape Navigator and Internet Explorer (FutureWave). By December, Macromedia acquired FutureWave and rebranded FutureSplash as Flash Player (Coale, 1997). Originally designed to allow users to easily create animations, it grew increasingly complex as Macromedia added scripting to allow for interactivity. This animation-first approach made Flash easier to learn than traditional programming languages, so many amateur games and websites were made in Flash (Salter & Murray, 2014).

At the time, browsers had no native support for features such as animations, interactivity, vector graphics, or 3D rendering. To get around this, web developers developed sites using plugins such as Flash Player. Web pages would prompt users to install Flash in order to get the full experience. This began a virtuous cycle of social influence: as more people installed Flash Player, more web developers would create sites with Flash.

By 2005, it was nearly impossible to browse the web without Flash. Many sites used Flash for their navigation menus, and sites such as YouTube were reliant upon it for core features such as viewing videos. Adobe acquired Macromedia in December 2005, rebranding it as Adobe Flash Player (Flynn, 2005). Indie game developers and multinational corporations alike were using Flash to design modern, interactive websites.

3.3 Launch of the iPhone

As part of a keynote address on January 9, 2007, Steve Jobs introduced the iPhone as a combination of a cell phone, an iPod, and an internet communications device (Apple, 2007). Although this last point received the least applause from the audience, this turned out to be the groundbreaking feature of the iPhone (Apple Events, 2007, 27:37–28:34). Most phones that advertised internet access prior to this primarily supported text-only emails or special, stripped-down versions of websites.

Web designers for mobile devices faced with a conundrum: how should they redesign websites for this new platform? Most websites were still designed for desktop computers, for users with a keyboard and mouse, with a standard desktop or laptop monitor resolution of 1024 × 768 pixels (W3Schools, 2021). However, the original iPhone had a screen with a resolution of only 320 × 480, on which these sites would display extremely zoomed out, requiring users to pinch and scroll in order to display text at a readable size. The primary approach at the time was to make a separate mobile site or web app, tailored to the physical dimensions and vertical screen of the smartphone, with buttons and links large enough to tap easily (Marcotte, 2010).

However, Safari did not support browser plugins on the iPhone. This included Adobe Flash Player, which Apple explicitly refused to support for several reasons, which Steve Jobs would outline in a blog post titled “Thoughts on Flash.” Apple did, however, partner with YouTube, the most popular site reliant on Flash Player, to make a native app for the iPhone, portending what was to come.

3.4 Replacements for Flash

By 2008, Firefox managed to increase their share of the desktop market up to approximately 30% (W3Counter). The major turning point in the second browser wars was Google's launch of its own browser, Google Chrome. Like with Microsoft, Google used its dominance in the market for web searches to advertise its own browser to users of other browsers. By 2012, Chrome would come to surpass Internet Explorer and Mozilla Firefox as the dominant web browser (*Statcounter,* 2022).

As the mobile device market diversified, it became infeasible to have a single mobile site tailored to the iPhone. Android phones had a variety of different screen sizes depending on the model and manufacturer, and tablets had screen sizes between those of smartphones and laptops. Combined with the rise of standards-abiding browsers such as Firefox, Chrome, and IE8, this led to web developers migrating towards cross-platform open standards and technologies, such as HTML5, CSS, JavaScript, SVG[11], and WebGL[12]. Eventually, responsive design took hold as the primary web design philosophy, where a single web page automatically adjusts to fit the size of the window or screen (Marcotte, 2010).

In November 2011, Adobe discontinued Flash Player for Android in favor of supporting HTML5 across all mobile devices. In their announcement, Adobe specifically noted that on some major mobile devices, HTML5 was exclusively supported (Winokur, 2011). This, of course, is in direct reference to Apple's policy of not allowing Flash on their mobile devices. On December 31, 2020, Adobe ended support for Flash outside of China, and on January 12, 2021, major browsers and the Flash Player plugin blocked all Flash content (Adobe, 2021).

[11] Scalable Vector Graphics (SVG) is a file format which stores images as lines and shapes, rather than pixels.
[12] WebGL (Web Graphic Library) is a way for web pages to take advantage of 2D and 3D graphics rendering.

4. Discussion

4.1 The browser wars and technological stasis

The lifespan of Flash spanned two browser wars: the first, when Internet Explorer toppled Netscape Navigator, and the second, when Google Chrome dominated IE and Firefox. During both periods, we see one of the major benefits of competition: rapid innovation.

During the first browser wars, IE and Netscape would independently add new features, while also copying each other's features, such as when IE adopted JScript to compete with Netscape's JavaScript. Flash benefited from this when Netscape added support for NPAPI plugins, which spurred Microsoft to add ActiveX plugin support to IE to match. This let Flash Player be a plugin for both browsers, allowing the developers of Flash to innovate and fill the niches of animation, vector graphics, video, and interactivity while the browsers continued to compete in other areas.

By contrast, during the second browser wars, what we see instead are browsers attempting to reclaim the niches that Flash Player occupied. Founded in 2004, The Web Hypertext Application Technology Working Group (WHATWG) currently consists of members from Apple, Google, Microsoft, and Mozilla who work on developing new HTML and web standards (WHATWG, 2004). Google released frequent updates for Chrome in an attempt to create a faster and more standards-compliant browser. By utilizing standards bodies such as the WHATWG and the World Wide Web Consortium (W3C), the browser developers collectively agreed upon new standards such as HTML5, CSS, JS, SVG, WebGL, and HLS[13]. This allowed them to act as a cartel of sorts and exclude Adobe from the conversation regarding the future of the web, while still supporting open standards.

[13] HTTP Live Streaming (HLS) is a video streaming protocol developed by Apple in 2009. It has since become a standard for video streaming across browsers.

Flash found itself on the right side of the first browser war by helping IE and Netscape both gain access to new forms of content without the browser developers having to work on the functionality themselves: it was in the right place at the right time. But it was on the wrong side of the second browser war, as Flash was now showing its age. It was now stuck in technological stasis, since it was much more difficult for Flash to add radically new, competitive functionality while still fixing bugs, patching exploits, and maintaining compatibility with old Flash projects. This gave browser developers the opportunity to make Flash Player redundant by defining new, "modern" standards for browsers to meet.

4.2 Apple's "walled garden" philosophy

With the iPhone, Apple created a "walled garden," where users can only install apps that Apple has approved for publication in the App Store. This provides some level of security and quality guarantees, but prevents developers such as Epic Games from publishing apps which contravene Apple's policies on in-app purchases, use of undocumented APIs[14], etc. (Statt, 2020; Eadicicco, 2015).

At the time, many commentators remarked that Apple was abusing its market power by preventing competitors, such as Adobe, from creating software with their own tools on Apple's platforms. In response, Apple CEO Steve Jobs wrote a blog post in April 2010 titled "Thoughts on Flash" (Jobs). In it, he describes six reasons Apple refused to allow Flash on iPhones. We can consider the public release of these points as an attempt on Apple's part to set the record straight on how it would define technological closure with regards to Flash. Of them, only the third point, that Flash has poor security and performance on mobile devices, and the fourth point, that Flash

[14] An Application Programming Interface (API) allows two programs to communicate with each other. In this case, the apps used APIs that Apple did not publicly reveal to communicate with the operating system. Only Apple's own apps and features are supposed to use those APIs, and Apple bans apps by third-party developers which use them.

video requires software encoding that is power-hungry and lowers battery life, directly claim that Flash is technologically inferior to the alternatives.

The most intriguing point is the fifth point, which states that since developers would need to rewrite their site to support touchscreens anyway, they might as well rewrite it with HTML5, CSS, and JavaScript. This presupposes several cultural shifts that were occurring around this time: first, that developers are the ones making websites; second, that developers must continually maintain and rewrite their websites; and third, that it is the responsibility of a web developer to cater to what the browser developer demands of them.

4.2.1 Democratization and centralization

At first, one of the fundamental principles of the internet was the end-to-end principle: every computer connected to the internet could act as both a server and a client. However, as the internet grew in the 1990s, it began to near its design limit of four billion addresses, and network engineers implemented solutions such as dynamic IP address allocation[15], NAT[16], and port forwarding[17] to allow it to keep growing. However, these came at the cost of breaking the end-to-end principle, making it more difficult for non-technically savvy users to set up their computer as a server. This changed the assumptions baked into the internet's technical code from allowing any computer to act as a server to requiring a substantial amount of investment and setup.

Nevertheless, as the internet boomed in the late 1990s and early 2000s, interested individuals and amateur programmers still wanted to create their own websites. Free web hosting platforms such as GeoCities popped up, and democratized the web hosting experience by

[15] Dynamic IP address allocation is when internet service providers (ISPs) assign a customer a temporary address from a pool of addresses, rather than a single, fixed IP address. This makes it difficult to host a website from a home internet connection, as the IP address for visitors to connect to will keep on changing.

[16] Network address translation (NAT) allows multiple devices on a local network to share a single public IP address.

[17] Port forwarding directs connections to a particular port on a public-facing IP address toward a particular device within the local network, allowing that device to act as a public-facing server.

allowing anyone to create their own website without having to set up their own server. Amateurs often used Flash to create interactive websites and games due to it being comparatively easier to learn than traditional programming languages like Java, especially for creative-minded people such as artists (Salter & Murray, 2014). These canvasses were unrestrained by the borders of the Facebook post, tweet, or Reddit comment that we see on the internet today.

While social media apps and platforms have indeed allowed far more people to participate in online spaces than the earlier web ever did, it has simultaneously limited their capability for expression. The shift away from a proprietary but easy-to-use language toward standardized, open-source languages designed by and for programmers changed the balance of power across the web. Now, it is professional developers who create and maintain apps, and the users instead merely generate content within the confines of the platform.

4.2.2 Software as a service and continual maintenance

This brings me to the second implication: that developers must continually maintain their websites. This is a hallmark of the modern approach to software as a service (SaaS). Consider many modern video games, such as *Minecraft*, *Fortnite*, *Fall Guys*, or *Pokémon GO*. Their developers have continually updated these games since their release, funding their continued development largely through microtransactions and in-app purchases. Other examples include Adobe Creative Cloud and Office 365, where enterprise and individual subscriptions fund their continued development.

Contrast this to how sellers historically sold software, where buyers paid a one-time fee for a physical floppy disk, cartridge, or CD which contained the software. With these, there was no way to update it without buying a new version once it came out a few years later. This is much more like how physical books have always been published: the consumer pays for the content once, and can fully enjoy it from that point on. Even if the publisher releases a new edition of the same book, it does not invalidate or remotely update previous editions, or force the consumer to pay to keep their book. The authors who created many of the Flash animations, art, and experiences during its heyday did it under this model. They invested their time and effort into creating and publishing their work once, and did not intend to continually maintain and update it. The shift to SaaS disadvantaged independent creators and small businesses who did not have sufficient resources to rewrite their code to support modern browsers.

This leads me to the third implication: that developers on a platform must continually meet the changing demands of the developers of that platform. Unlike with Microsoft Windows, which maintains an almost extreme level of backwards compatibility with software written for older versions, Apple has largely taken a path of continually making older applications no longer usable on new software and hardware. The Mac line has undergone two relatively recent changes in their processor lines: switching from PowerPC to x86 from 2005-2006, and from x86-64 to the M1/M2 chips starting in 2020. Both changes used software named Rosetta to translate older software designed for the older processor to run on the newer processor. A few years after switching to x86, however, Apple completely stopped supporting Rosetta on Mac OS X, meaning that subsequent versions could no longer run older software (AppleInsider, 2011).

Apple has taken a similar approach with the App Store, removing apps from sale if their developer has not updated them recently, even if the apps are still fully functional (Roth, 2022). For them to even submit an update to the App Store, developers must build the app with the latest versions of Xcode, Apple’s development environment for iOS, and the latest software development kit (SDK). This means that they will still need to rewrite their old code to meet Apple’s new standards, even to submit an update that changes nothing.

* * *

In summary, Apple’s practice of continually breaking older software has served their interests in maintaining control over their platform, largely by allowing them to freely drop efforts to be backwards-compatible with older software. This, in turn, has contributed to the centralization of the internet around a few tech companies and platforms, as independent creators find it increasingly difficult to create websites and experiences without the burden of continually maintaining them.

4.3 Shift from desktop to mobile

The worldwide market share of mobile devices has outnumbered desktop devices since 2017, having increased rapidly from near zero in 2009 (*Statcounter*, 2022). When Adobe discontinued Flash Player for Android in November 2011, they effectively surrendered to Apple by giving up their claim on the growing market of users on mobile phones. Up until that point, the fact that Android phones and tablets supported Flash was a selling point, and put pressure on Apple for being the lone holdout, who had refused to allow Flash Player on their mobile devices.

By limiting themselves to the desktop market, Adobe ended up forcing their own customers to abandon Flash for mobile as well. Since maintaining two codebases, one with Flash for desktop and one without for mobile, was often more expensive than migrating it to a single codebase that could run on both desktop and mobile, it shouldn’t come as a surprise that many companies would end up jumping ship from Flash when even Adobe themselves could no longer commit to it as a universal platform.

5. Conclusion

Did Adobe make any fatal missteps that directly led to Flash’s demise? The easiest target is Adobe’s decision to discontinue Flash for Android, since that choice let Apple and Google alone decide what the future of the internet on mobile would be. But by that point, Apple had already been supporting work on implementing replacements for each of Flash Player's core functionalities, and standardizing them to ensure that Mozilla, Google, and Microsoft would follow suit. Adobe was dependent on the browser developers to keep supporting Flash Player, and they would only do that if Flash Player was sufficiently important to their users, both corporate and individual.

Today, Adobe still maintains a chokehold over the photography, graphic design, and layout industry with their Creative Cloud offerings such as Photoshop, Illustrator, and InDesign, all of which are proprietary software. By contrast, Flash's dominance in the interactive web content space could be threatened by a disruption in how we access the web (namely, the smartphone) and an adversary willing to keep them out (namely, Apple). The social power Adobe had was through the web developers and creators already using and making Flash animations, which if they had been large enough, might have convinced Apple to relent and allow Flash.

But the tide was already turning against independent creators and small businesses that relied on Flash. The rise of social media and cloud computing had pushed the internet towards increased centralization, focused around all-encompassing platforms and algorithms of Google, YouTube, Facebook, Twitter, and Instagram. In exchange for discoverability, users willingly gave up complete creative control and editorial freedom.

Perhaps the fundamental problem was never with Flash, but with the design of the internet. Breaking the end-to-end principle meant that the internet was no longer merely interconnected computers on a level playing field, but hierarchically separated into clients and servers, guests and hosts, or users and providers. Rather than going through all the trouble of setting up your own server and website, why wouldn't you take the path of least resistance, and just post on Facebook instead?

Works Cited

Adobe. (2021, January 13). Adobe Flash Player EOL General Information Page. *Adobe*. Retrieved October 11, 2022, from https://www.adobe.com/products/flashplayer/end-of-life.html

Apple. (2007) Apple Reinvents the Phone with iPhone. *Apple Inc.* Retrieved February 24, 2023, from https://www.apple.com/newsroom/2007/01/09Apple-Reinvents-the-Phone-with-iPhone/

Apple Events. (2007) Macworld San Francisco 2007 Keynote Address. *Apple Inc.* Retrieved February 24, 2023, from https://podcasts.apple.com/us/podcast/macworld-san-francisco-2007-keynote-address/id275834665?i=1000026524322

AppleInsider. (2011) Inside Mac OS X 10.7 Lion: Missing Front Row, Rosetta and Java runtime. *AppleInsider*. Retrieved February 24, 2023, from https://appleinsider.com/articles/11/02/26/mac_os_x_lion_drops_front_row_java_runtime_rosetta.html

Bedingfield, W. (2019). The rise and fall of Flash, the annoying plugin that shaped the modern web. *WIRED UK*. Retrieved October 27, 2022, from https://www.wired.co.uk/article/history-of-macromedia-flash

Bijker, W. E., Hughes, T. P., & Pinch, T. (2012). *The Social Construction of Technological Systems, anniversary edition: New Directions in the Sociology and History of Technology (The MIT Press)* (Anniversary). MIT Press.

Baldazo, R. (2003). Microsoft Internet Explorer 6.0 review: Microsoft Internet Explorer 6.0. *CNET*. Retrieved February 24, 2023 from https://www.cnet.com/reviews/microsoft-internet-explorer-6-0-review/

Borland, J. (2003). Browser wars: High price, huge rewards. *ZDNet*. Retrieved February 19, 2011, from https://web.archive.org/web/20110219142631/https://www.zdnet.com/news/browser-wars-high-price-huge-rewards/128738

Coale, K. (1997) Macromedia Rides the FutureWave. *WIRED*. Retrieved November 9, 2020 from https://web.archive.org/web/20201109033154/http://www.wired.com/1997/01/macromedia-rides-the-futurewave/

Eadicicco, L. (2015) Hundreds of apps have been banned from Apple's App Store for spying on your personal information. *Business Insider*. Retrieved February 24, 2023, from https://www.businessinsider.com/apple-removes-apps-youmi-sdk-personal-information-2015-10

Flanagin, A. J., Flanagin, C., & Flanagin, J. (2009). Technical code and the social construction of the internet. *New Media & Society*, *12*(2), 179–196. https://doi.org/10.1177/1461444809341391

FutureWave Software (1996). Download FutureSplash^{TM} Player 1.1. FutureWave Software, Inc. Retrieved November 5, 1996 from https://web.archive.org/web/19961105022644/http://www.futurewave.com/fssetup/dlfs_manual.htm

Flynn, L. J. (2005). Adobe Buys Macromedia for $3.4 Billion. *The New York Times*. Retrieved May 29, 2015 from https://web.archive.org/web/20150529190713/https://www.nytimes.com/2005/04/19/technology/adobe-buys-macromedia-for-34-billion.html

Jamieson, J. (2016). Many (to platform) to many: Web 2.0 application infrastructures. *First Monday*. https://doi.org/10.5210/fm.v21i6.6792

Jenkins, H., Ford, S., & J., G. (2013). *Spreadable Media: Creating Value and Meaning in a Networked Culture on JSTOR*. NYU Press. http://www.jstor.org/stable/j.ctt9qfk6w

Jobs, S. (2010). *Thoughts on Flash*. Apple Inc. Retrieved May 1, 2010, from https://web.archive.org/web/20100501010616/http://www.apple.com/hotnews/thoughts-on-flash/

Lash, A. (1997). Memphis is Windows 98. *CNET*. Retrieved February 20, 2016 from https://web.archive.org/web/20160220095758/https://www.cnet.com/news/memphis-is-windows-98/

Marcotte, E. (2010) Responsive Web Design. *A List Apart*. Retrieved on February 24, 2023, from https://alistapart.com/article/responsive-web-design/

Marx, L., & Smith, M. R. (1994). Does Technology Drive History? The Dilemma of Technological Determinism. The MIT Press. Retrieved April 14, 2023 from https://search.ebscohost.com/login.aspx?direct=true&db=nlebk&AN=28215&site=ehost-live&scope=site

McCarthy, D. R. (2010). Open Networks and the Open Door: American Foreign Policy and the Narration of the Internet. *Foreign Policy Analysis*, *7*(1), 89–111. https://doi.org/10.1111/j.1743-8594.2010.00124.x

OneStat.com (2006). Microsoft's Internet Explorer 7 global usage share is 3.06 percent according to OneStat.com. *OneStat.com*. Retrieved December 11, 2006 from https://web.archive.org/web/20061211150013/http://www.onestat.com/html/aboutus_pressbox49-microsoft-internet-explorer-7-usage.html

Plennert, M. (2018). The social construction of technological stasis: The stagnating data structure in OpenStreetMap. *Big Data & Society*, *5*(2), 205395171879059. https://doi.org/10.1177/2053951718790591

Roth, E. (2022). Apple App Store appears to be widely removing outdated apps. *The Verge*. Retrieved on February 14, 2023, from https://www.theverge.com/2022/4/23/23038870/ apple-app-store-widely-remove-outdated-apps-developers

Salter, A., & Murray, J. (2014). *Flash: Building the Interactive Web*. The MIT Press. https://doi.org/10.7551/mitpress/9833.001.0001

Statcounter Global Stats. (2022). StatCounter. Retrieved October 11, 2022, from https://gs.statcounter.com/

Statt, N. (2020) Apple just kicked Fortnite off the App Store. *The Verge*. Retrieved on February 24, 2023, from https://www.theverge.com/2020/8/13/21366438/apple-fortnite-ios-app-store-violations-epic-payments

TheCounter.com (2006) Browser Stats. Retrieved December 9, 2006, from https://web.archive.org/web/20061209034515/http://www.thecounter.com/stats/2006/December/browser.php

Vannoy, S. A., & Palvia, P. (2010). The social influence model of technology adoption. *Communications of the ACM*, *53*(6), 149–153. https://doi.org/10.1145/1743546.1743585

W3Counter. (2008). Browser & Platform Market Share: January 2008. Retrieved February 24, 2023, from https://www.w3counter.com/globalstats.php?year=2008&month=1

W3Schools. (2021). Browser Display Statistics. Retrieved February 24, 2023, from https://www.w3schools.com/browsers/browsers_display.asp

WHATWG. (n.d.) FAQ. *WHATWG*. Retrieved on February 24, 2023, from https://whatwg.org/faq

Winokur, D. (2011). *Flash to Focus on PC Browsing and Mobile Apps; Adobe to More Aggressively Contribute to HTML5*. Adobe Featured Blogs. Retrieved on November 9, 2011, from https://web.archive.org/web/20111109161903/http://blogs.adobe.com/conversations/2011/11/flash-focus.html