SI: Culture Digitally

SM+S
social media + society

Social Media + Society
July-December 2015: 1–11

DOI: 10.1177/2056305115603080
sms.sagepub.com
SAGE

The Platformization of the Web: Making Web Data Platform Ready

Anne Helmond

Abstract

In this article, I inquire into Facebook's development as a platform by situating it within the transformation of social network sites into social media platforms. I explore this shift with a historical perspective on, what I refer to as, platformization, or the rise of the platform as the dominant infrastructural and economic model of the social web and its consequences. Platformization entails the extension of social media platforms into the rest of the web and their drive to make external web data "platform ready." The specific technological architecture and ontological distinctiveness of platforms will be examined by taking their programmability into account. I position platformization as a form of platform critique that inquires into the dynamics of the decentralization of platform features and the recentralization of "platform ready" data as a way to examine the consequences of the programmability of social media platforms for the web.

Keywords

social media, platforms, Facebook, social network sites, Application Programming Interfaces

On 15 August 2006, Facebook introduced the Facebook Development Platform, giving third-party developers access to Facebook users' profiles, friends, photos, and events to extend the "Facebook experience" into external applications (Fetterman, 2006)—thereby turning Facebook into a developer environment. A year later, at the first f8 Developer Conference, Facebook launched Facebook Platform, officially marking Facebook's advancement as a platform. Facebook Platform provides developers with a set of tools for sending and retrieving data from and to Facebook and a deep integration with Facebook's "social graph," a mapping of the connections between people and objects, for building applications (Geminder, 2007; Hicks, 2010).

In this article, I inquire into Facebook's development as a platform by situating it within the transformation of social network *sites* into social media *platforms*. I situate this "platformization," or the rise of the platform as the dominant infrastructural and economic model of the social web and its consequences, in its historical context. Platformization entails the extension of social media platforms into the rest of the web and their drive to make external web data "platform ready." The specific technological architecture and ontological distinctiveness of platforms will be examined by taking one aspect of their medium-specificity (Rogers, 2013), their programmability, into account. In doing so, I follow Langlois, McKelvey, et al.'s (2009) call for a "platform-based perspective," which, according to Fenwick McKelvey (2011), should critically inquire into the programmability of platforms. Examining the decentralization of platform features into the web and the recentralization of platform ready data is a way to examine the consequences of the programmability of social media platforms for the web.

The new architectural model of the platform explicitly opens up websites by enabling their programmability with a software interface, an Application Programming Interface (API), for third parties. To comprehend this programmatic access, I draw on Alan Liu's (2004) notion of "data pours" to conceptualize platforms as pouring data systems that set up data channels to enable data flows with third parties. These data pours not only set up channels for data flows between social media platforms and third parties but also function as data channels to make external web data platform ready.

Material–Technical Perspective on Social Media Platforms

The term "platform" has become the dominant concept for social media companies for positioning themselves in the

University of Amsterdam, The Netherlands

Corresponding Author:
Anne Helmond, Department of Media Studies, University of Amsterdam, Turfdraagsterpad 9, 1012 XT Amsterdam, The Netherlands.
Email: a.helmond@uva.nl

market and addressing users, and it has been widely taken up by consumers and the press (Gillespie, 2010). Within new media studies, the platform concept has gained prominence to draw attention to the "discursive work" they undertake (Gillespie, 2010, p. 348) and to the role of software—which powers social media—in shaping participation and sociality (Bucher, 2012a; Hands, 2013; Langlois, McKelvey, Elmer, & Werbin, 2009; Van Dijck, 2013).

In one of the most central discussions on platforms, Tarleton Gillespie (2010) puts forward a rather open account of platforms by focusing on the different connotations of the term. In the computational sense, Gillespie (2010) defines a platform as an infrastructure to build applications on. However, Gillespie (2010) contends, Web 2.0 companies have introduced a broader meaning of the term "platform" that moves beyond its computational meaning:

> This more conceptual use of "platform" leans on all of the term's connotations: computational, something to build upon and innovate from; political, a place from which to speak and be heard; figurative, in that the opportunity is an abstract promise as much as a practical one; and architectural, in that YouTube is designed as an open-armed, egalitarian facilitation of expression, not an elitist gatekeeper with normative and technical restrictions. (p. 352)

Gillespie argues that this conceptual use enables platforms to bring various actors together. The computational meaning of platform speaks to developers, while the other connotations address actors such as users, advertisers, and clients (Gillespie, 2010). Gillespie is describing what in economics Jean-Charles Rochet and Jean Tirole (2003) call the business model of a "multi-sided market," in which a platform enables interactions between two or more distinct parties (p. 990). Facebook is an example of a multi-sided platform that connects users, advertisers, and third-party developers and experiences network effects where value increases for all parties as more people use it (Hagiu, 2014). Within this economics and management literature, Annabelle Gawer argues, platforms have often been theorized from two distinct perspectives: "economic theory conceptualizes platforms as markets (Rochet & Tirole, 2003)," whereas "engineering design theorizes them as 'modular technological architectures' (Baldwin & Woodard, 2009)" (Gawer, 2014, p. 1240). Bernhard Rieder and Guillaume Sire (2014) make an important call for bringing these perspectives together (p. 197): studying platforms as multi-sided markets, they argue, "can extend analyses of concrete configurations of power and identify control points, structural dynamics and crucial resources for argumentation" (Rieder & Sire, 2013, p. 208). Following such techno-economic outlook on platforms, in this article, I examine how the modular technical architecture of social media platforms connects to their economic model.

In his work on platforms, Gillespie (2010) emphasizes the participatory and economic aspects of platforms over their computational dimension by stating that "'[p]latforms' are 'platforms' not necessarily because they allow code to be written or run, but because they afford an opportunity to communicate, interact or sell" (p. 351). Other authors, such as Ian Bogost and Nick Montfort (2009), suggest a more narrow focus on platforms by foregrounding their computational aspect. In what follows, I am interested in developing such computational account of platforms further to examine the "work that platforms do" not in a rhetorical sense (cf. Gillespie, 2010) but from a material–technical perspective. Bogost and Montfort (2009) refute the idea that "everything these days is a platform" and call for taking platforms as computational infrastructures seriously. They see the platform, in its computational sense, as an understudied layer of new media (Bogost & Montfort, 2009). To address this blind spot, Montfort and Bogost (2009) introduce "platform studies," a call for a "technical rigor and in-depth investigation of how computing technologies work" (p. vii) to analyze "the connection between technical specifics and culture" (Bogost and Montfort 2009).

These connections have been explored by a number of authors engaging with a platform politics perspective to examine "the technological affordances of platforms in relation to their political, economic and social interests" as an important site where "platform politics" play out (Hands, 2013; Langlois & Elmer, 2013).[1] Platform politics approaches include critically interrogating the platform concept (Gillespie, 2010; McKelvey, 2011), analyzing the "technocultural logics" of platforms (Gerlitz & Helmond, 2013; Langlois, Elmer, McKelvey, & Devereaux, 2009; Langlois, McKelvey, et al., 2009), examining the role of the platform architecture in shaping networked sociality (Bucher, 2012a; Van Dijck, 2013) and analyzing the politics of APIs (Bucher, 2013) and platform data (Puschmann & Burgess, 2013) (see Renzi, 2011).

I am interested in the way platforms reformat the web according to the logic of social media. My approach is based on what Langlois et al. refer to as "disaggregation," critically examining social media platforms by taking them apart to inquire into their specific components (Langlois, McKelvey, et al., 2009). This contribution to platform studies and social media studies lies in a detailed material–technical perspective on the development and emergence of what we understand as social media platforms today. I will further develop this argument by focusing on Facebook, one of the largest and most visited social media platforms.[2]

Facebook: Social Network Site or Platform?

Before the platform concept gained prominence, social media platforms such as Facebook were often conceptualized as social network sites, defined by boyd and Ellison (2008) as web services in which users can create a profile and build and display a list of connections with other users in

the network (p. 211). However, Facebook has always carefully refrained from calling itself a social network (Arrington, 2008; Locke, 2007). Rather, over time, Facebook founder Mark Zuckerberg has framed Facebook as a "social directory" (Facebook Newsroom, 2006); a "social utility" (Facebook Newsroom, 2006); and a "platform" (Facebook Newsroom, 2007). In his book *The Facebook Effect* on the history of Facebook, David Kirkpatrick (2010) describes how Zuckerberg has always envisioned Facebook as a computational platform for other applications to run on, since its inception as Thefacebook in 2004 (pp. 215-217):

> He [Zuckerberg] wanted to do for the Web what Gates did for the personal computer: create a standard software infrastructure that made it easier to build applications— this time, applications that had a social component. "We want to make Facebook into something of an operating system, so you can run full applications," he [Zuckerberg] explained. (Kirkpatrick, 2010, p. 217)

In the Fall of 2004, Zuckerberg was working on another software project alongside Thefacebook called Wirehog, "a peer-to-peer content-sharing service" (Kirkpatrick, 2010, p. 44). The Wirehog application was integrated into Thefacebook to make use of its friendship connections to share content in Thefacebook with friends. Zuckerberg saw Wirehog as "the first example of treating Thefacebook as a platform for other types of applications" (Kirkpatrick, 2010, pp. 99-100). So, instead of a social network, Mark Zuckerberg has seen and designed Facebook as a platform from the beginning. Facebook's development as a platform should be perceived in the wider context of Web 2.0 as "the web as platform" (O'Reilly, 2005), in which the web was positioned as development platform.

Web 2.0: The Web as Platform

Social network sites are typically classified as one specific type of Web 2.0 application (Beer & Burrows, 2007) or type of social media (Van Dijck, 2013, p. 8). The term was popularized at the first Web 2.0 conference in 2004, when Tim O'Reilly defined Web 2.0 as the web as platform, a phrase used to situate the web as a "robust development platform" in which "websites become software components" (O'Reilly & Battelle, 2004). Web 2.0 or "the participatory web" is now understood as a wide set of services that foster collaboration and participation (Madden & Fox, 2006).[3]

O'Reilly put the computational meaning of the term "platform" at the center of the web as platform concept. With Web 2.0, O'Reilly (2005) no longer saw the web just as a medium for publishing information—which he retrospectively labeled Web 1.0—but as an infrastructure to build applications on, a distributed operating system that could deliver software services. Therefore, Matthew Allen (2013) argues, we should see Web 2.0 as "rhetorical technology" in which "the computing industry attempted to change the way we think of the internet" (p. 264), from publishing channel to software development platform.

However, this more computational definition of Web 2.0 as the web as platform did not catch on after the conference, Robert Gehl (2010) argues. Instead, Gehl (2010) claims, Web 2.0 was seen as a revival of the industry after the dotcom crash and, even more so within public and academic debates, as a revolution that would reshape the media landscape (pp. 26-37). Web 2.0 technologies were seen as blurring the boundaries between production and consumption (Bruns, 2008), giving rise to new forms of user participation as part of an online "participatory culture" (Jenkins, 2006). So, while the original definition of Web 2.0 implied making use of the web as a computational platform, it would be embodied in a more metaphorical sense (cf. Gillespie, 2010), as a platform for participation with the associated rhetoric of "empowerment" and "democratization" (Beer, 2009, p. 986).

From Social Network Sites to Social Media Platforms

To shift the focus from this broader conceptual notion of platforms back to a more narrow computational understanding to develop a platform critique of Facebook, I wish to further explore the technological development of software platforms on the web and in particular social media platforms. I do so by attending to another computational definition of platform, provided by Netscape founder Marc Andreessen (2007a) in a blog post discussing the launch of Facebook Platform:

> Definitionally, a "platform" is a system that can be reprogrammed and therefore customized by outside developers—users—and in that way, adapted to countless needs and niches that the platform's original developers could not have possibly contemplated, much less had time to accommodate.

For Andreessen (2007b), the key term in this definition of a platform is *programmable*, which eradicates the more conceptual uses of the term: "If you can program it, then it's a platform. If you can't, then it's not."

The programmability of Web 2.0 platforms, so McKelvey (2011) argues, offers a novel line of criticism within platform studies that starts with asking how a platform enacts its programmability. The notion of programmability has been key to understanding the logic of new media (Chun, 2011; Manovich, 2001)[4] and, by extension, figures centrally in examining the underlying logic of social media platforms.

To inquire into the specific preconditions of the programmability of social media platforms, I draw on Evans, Hagiu, and Schmalensee's (2006) definition of software platform as "a software program that makes services available to other software programs through Application

Programming Interfaces (APIs)" (p. vii). What follows from this definition is that in order to become a platform, a software program—or a website—needs to provide an interface that allows for its (re)programming: an API:

> An API is an interface provided by an application that lets users interact with or respond to data or service requests from another program, other applications, or Web sites. APIs facilitate data exchange between applications, allow the creation of new applications, and form the foundation for the "Web as a platform" concept. (Murugesan, 2007, p. 36)

Returning to O'Reilly's positioning of the web as a development platform for new services, not only the web as a whole but also websites themselves are now transformed into platforms by providing an API.[5] For example, Facebook is a platform because it offers an API that can be used by developers and webmasters to build new services on top of Facebook and to integrate their websites and apps with Facebook data and functionality.[6] Dating app Tinder is an example of an app that has been built on top of the Facebook platform: it requires users to login with Facebook and uses Facebook data such as "likes" and shared friends to match potential partners. Another way to integrate with Facebook is demonstrated by webmasters who have implemented Facebook functionality such as Like buttons into their pages.

In the web as platform websites can have two different interfaces: a user interface for human consumption (e.g. Facebook.com) and a software interface for machine consumption (e.g. Facebook Graph API). This software interface, the API, makes a website programmable by offering structured access to its data and functionality and turns it into a platform that others can build on. To extend this line of thinking further, I place APIs at the core of the shift from social network *sites* to social media *platforms*. The moment social network sites offer APIs, they turn into social media platforms by enacting their programmability. The API then becomes a key site to critically inquire into the consequences of this programmability.

Rise of Social Media APIs

Within the field of media studies, social media APIs have been understood as the technological glue of the social web in connecting services and enabling the sharing of content (Bodle, 2011; Bucher, 2013; Langlois, McKelvey, et al., 2009), as protocological objects (Bucher, 2013), as regulatory instruments that govern the relations between the platform and third parties (Puschmann & Burgess, 2013), as the business model of the social web[7] (Bodle, 2011; Bucher, 2013), and as tools that construct data for the data market (Vis, 2013). Most prominently, APIs have been used and discussed as "a method for data collection on social media platforms" (Lomborg & Bechmann, 2014). Less attention has been paid, however, to the history of social media APIs,[8] that is, their emergence on the web as part of the material infrastructure of social media platforms and their consequences for the adaptation of the platform model. One of the most comprehensive accounts so far has been by technology blogger Kin Lane (n.d.), who brands himself as "API Evangelist" and who has been studying "the business and politics of APIs" since 2010.

Lane (2012) traces the historical emergence of web APIs that targeted external developers back to the early 2000s, when Salesforce in 1999, eBay in 2001, and Amazon in 2002 started to offer APIs as business-to-business solutions for e-commerce. This first generation of web APIs, mainly provided by e-commerce companies, focused on exchanging data between different business applications to enable transactions and sales management (Lane, 2012). For example, Amazon's (2002) Web Services platform enabled third-party websites to search through their catalogue, display Amazon products, and earn referral fees from purchases from their own sites. In doing so, Amazon's API extended their e-commerce services into other websites.

In the mid-2000s, a new generation of web APIs, provided by social network sites, shifted the focus from sales transactions to access to user-generated content, user information, and their connections (Lane, 2012).

In 2003, social bookmarking site del.icio.us started offering programmatic access to its site, followed by Flickr in 2004, YouTube in 2005, Last.fm in 2006, Facebook in 2006, and Twitter in 2006, after which many other social network sites announced their APIs (DuVander, 2012; Lane, 2012). Robert Bodle (2011) describes how these sites made their content and functionality available as part of a business strategy in which third parties can add value to a platform by building new services on top of it (p. 325). He explains how Tim O'Reilly advocated businesses to pursue a platform strategy by opening up their valuable data to achieve platform lock-in (Bodle, 2011, p. 325). In his Web 2.0 manifesto, O'Reilly (2005) further encouraged the reuse of data with the recommendation to "design for 'hackability' and remixability" by offering third parties access to data and services. O'Reilly (2005) positioned data as the "building blocks" of Web 2.0. This access has given rise to the typical Web 2.0 practice of creating mashups—that is, building new applications by remixing data and functionality from existing sources using APIs (Benslimane, Dustdar, & Sheth, 2008). Web 2.0 has, therefore, also become known as "the programmable web" (Anderson, 2012; O'Reilly, 2005). In what follows, I explore the different types of programmability that social media platforms offer through their APIs, in order to formulate a platform critique of Facebook that foregrounds its distinct conditions of programmability and their consequences.

Levels and Conditions of Programmability

In a blog post on Facebook's new platform, Marc Andreessen (2007b) explained how the programmability of Internet-based software platforms can be facilitated on different levels, producing what he sees as three types of Internet platforms. These levels can also serve as a way to critically inquire into the role of the platform architecture.

According to Andreessen, most social media platforms provide a so-called Level 1 or "Access API." Here, external developers can access a platform's data and functionality by making API calls, which represent specific operations to perform a task, for example, read data, write data, or delete data (Andreessen, 2007b). The API is accessed "from outside the core system" which means that "the developer's application code lives outside the platform" (Andreessen, 2007b). Photo sharing service Flickr is an example of an Access API, where a developer can build a third-party application such as a slideshow viewer to show photos tagged with "sunset" using the Flickr API to request these data. In this scenario, the code of the application is located on an external server, and the application is hosted outside of Flickr. The programmability of a Level 1 platform is characterized by simple access to data and functionality. Developers can build new applications on top of the platform and integrate data and functionality into their external websites and apps but cannot reprogram the platform itself. That is, the programmability of Level 1 platforms is a way for platforms to expand outside of their platform boundaries.

The Level 2 "Plug-In API" allows developers to "build new functions that can be injected, or 'plug in,' to the core system and its user interface" (Andreessen, 2007b). Andreessen uses Facebook as an example of a Plug-In API since it not only allows developers to access data and functionality from Facebook to build new applications (Level 1 Access API), it also allows developers to load and use their application within the Facebook environment[9] through a so-called Canvas Frame. Canvas is "a frame in which to put your app or game directly on Facebook.com" in order to "deeply integrate into the core Facebook experience" (Facebook Developers, n.d.-a).[10] While the app runs within Facebook, the application code is located outside of the Facebook platform (Andreessen, 2007b).[11] Canvas apps enable users to customize their Facebook experience, drawing attention to McKelvey's (2011) reconsideration of John van Neumann's idea of "programming as an act of composition."

In the Level 3 "Runtime Environment" API, third-party applications run within the runtime environment of the platform itself (Andreessen, 2007b). Andreessen explains that this approach is most similar to "traditional" computing platforms, such as Windows operating system, where developers built applications to be executed within Windows itself (Andreessen, 2007b). The platform as runtime environment is the least common approach on the web, since it requires a more complicated technical framework for developers as well as database and storage management (Andreessen, 2007b).[12] As a consequence, the programmability of social media platforms is typically enabled through an Access API, Plug-In API, or both. More specifically, in Andreessen's terms, the most common type of social media platform is the Level 1 Access API (Twitter, Facebook, YouTube, Tumblr, and Instagram), followed by the Level 2 Plug-In API (Facebook).[13]

By distinguishing between different types of platforms, Andreessen offers a framework with which to evaluate individual platforms based on their conditions of programmability. Similarly, by drawing on Florian Cramer and Matthew Fuller's (2008) typology of interfaces, McKelvey (2011) argues that "[s]ince platforms have different interfaces, the line of critique allows for the comparison of how platforms facilitate programmability." That is, we can compare social media APIs to examine what kind of programmability these platforms envision, what they enable and constrain and what kind of data and functionality is made available for use and for whom.

Platformization of the Web

I use the term "platformization" to refer to the rise of the platform as the dominant infrastructural and economic model of the social web and the consequences of the expansion of social media platforms into other spaces online. Central to this is the offer of APIs, which turn social network sites into social media platforms. In order to understand these effects, I will explore how the distinct conditions of the programmability of social media platforms enable them to extend into the web and to employ these extensions to format external web data. That is, platforms enact their programmability to decentralize data production and recentralize data collection (Gerlitz & Helmond, 2013).

Websites have historically enabled their programmability through the exchange of data, content, and functionality with third parties in three ways, together providing the preconditions for the platformization of the web: first, the separation of content and presentation; second, the modularization of content and features; and, third, the interfacing with databases.

Separation of Content and Presentation

Most websites are created using the HyperText Markup Language (HTML), which describes the content and presentation of a web document. Since HTML is a presentation technology designed for human consumption and many HTML websites are ill-formatted, it is difficult for a machine to extract and process structured information from a website (Myllymaki, 2002, p. 635). The Extensible Markup Language

(XML) addresses these issues by separating content, structure, and presentation in a text-based format for machine consumption (W3C, n.d.).[14] This machine-readable and human-readable format enables the sharing of structured information between otherwise incompatible systems (Myllymaki, 2002, p. 635; W3C, n.d.). XML has been an extremely important development for the web by making website data machine-readable and interchangeable between different systems. It enables the structured formatting of data for transmission, forming the basis for various data exchange mechanisms that let website data flow out and into other websites.[15] Richardson and Ruby (2008) contend that XML is key to technologies such as RSS, XML-RPC, and SOAP which have "formed a programmable web, one that extended the human web for the convenience of software programs" (p. xviii).[16]

According to Liu (2004), the separation of content and presentation through XML informs the underlying technologic of the "post-industrial, transmission of information," which requires content be made "transformable," "autonomously mobile," and "automated" (pp. 57-58). This separation, Liu (2004) continues, makes content "transcendental," so that it can be poured from one container into another, moving from database to database on the web (p. 59). A. Liu (2004) describes how XML signals a shift from the first generation of self-contained HTML websites to a new type of website that is filled with content from external databases (p. 57). These new web pages employ what Liu calls "data pours" to pull in and display dynamic content from third parties. A data pour is code embedded in a web page demarcating a space or container on that page that transfers data from and to external databases (Liu, 2004, p. 59).

Published in the very early days of Web 2.0, Liu's (2008) idea of data pours can be read as an early reflection on the increasing modularity of the web, which he later updated as follows:

> My observations here about data pours apply with even more force in Web 2.0, where user-produced content flows both in and out of back-end databases through "template" Web pages that are often elegant, minimalist designs built around an all-powerful, blind aperture of parameterized code—like a reversed black hole—that sucks all content in and throws it out again. (p. 320)

These now commonplace data pours of Web 2.0, establish data channels for data flows between websites and external databases.

Modularization of Content and Features

In separating content from presentation, XML compartmentalizes web content by structurally describing each element on a web page and turning these into small modules of data that can be reused. The compartmentalization of content makes existing content available on the web for machine consumption and enables the circulation of content through modular elements. Modularization is a key aspect of modern software design that enables the management of complex systems by dividing them up into smaller modules and encouraging the reuse of these modules (Baldwin & Clark, 2000; Gehl, 2012; McKelvey, 2011). Within Web 2.0, Ullrich et al. (2008) argue, "services often disseminate their functionality by plug-in modular components, so called widgets." That is, "a platform architecture displays a special type of modularity, in which a product or system is split into a set of components" (Baldwin & Woodard, 2009, p. 25). These widgets enable the integration of a service's content and functionality into another website with a few lines of code that create a data pour. Widgets have become central, platform-specific objects for social media platforms to distribute their content across different web spaces and to extend themselves into the web.

An important development in this extension came from the video sharing site YouTube. On 7 July 2005, YouTube (2005) announced a new feature that enabled users to put a list of their YouTube videos on their own websites by copy-pasting the provided HTML code. This code embedded a YouTube widget showing a list of videos and thumbnails that linked to the videos on YouTube. A month later, YouTube announced a new widget that embedded a video player, so that YouTube videos could now directly be played from within any website (YouTube, 2005). The widget made it possible to distribute and view YouTube videos outside of YouTube's website. This video embedding feature is often seen as an important factor in the success of YouTube as it enabled YouTube to circulate videos across social networks, blogs, and other parts of the web by modularizing and decentralizing its platform features (Cheng, Dale, & Liu, 2008).

While YouTube created its own widgets to distribute content *outside* of its website, social network MySpace played an important role in popularizing the role of third-party widgets to share content *inside* of its network. In contrast to other social networks that were popular in 2005-2006—such as Friendster—MySpace allowed users to insert embed codes into their profile pages to add music players, photo albums, and videos. It was the first social network that had such an open architecture and with it arose a culture of profile customizing and accessorizing (boyd, 2007).

With the ability to insert embed codes into profile pages, third-party developers started to create widgets to enhance the look and functionality of MySpace. In November 2005, RockYou launched their first MySpace Flash widget to create and display photo slideshows. An important aspect of these early widgets is that, unlike YouTube's sharing widgets, they did not directly interface with MySpace's database. Users could not load their photos directly from MySpace into the widget because MySpace did not offer structured access (through an API or otherwise) to these photos. Instead, users had to upload their photos to the external image hosting website ImageShack within the RockYou widget first (Tokuda, 2009).

This lack of a direct interface with MySpace's database is what Gehl (2012) refers to as MySpace's "abstraction failure" to extract and monetize the content from its network (pp. 111-112). Whereas MySpace widgets were mostly oriented toward integrating and distributing content within its own network, YouTube's widgets were oriented toward the distribution of content and functionality outside of its network. As many Web 2.0 websites started to offer embed codes and widgets to distribute their content across the web, the approach of decentralizing platform features became central. A second important distinction is that, unlike MySpace widgets, YouTube widgets directly interfaced with the site's database. However, YouTube's database facing widgets were based on *one-way data streams*, on the dynamics of decentralization, where content is retrieved from the database and displayed on an external website. The next generation of widgets would be based on directly interfacing with databases to enable *two-way data streams*, on the dynamics of both decentralization and recentralization, to not only read data from the database but also to write new data to it.

Interfacing with Databases

Facebook's social plug-ins are a set of tools, or widgets, including the ubiquitous Like button, "that let you share your experience off of Facebook with your friends and others on Facebook" (Facebook Help, n.d.). The plug-ins function as modules to extend platform functionality into external websites (cf. Bodle, 2011). At the same time, Taina Bucher (2012b) argues, they function as "edge-creating devices," collecting data created by connections or "edges" outside of Facebook.com and sending it back to the platform's database (p. 6). Social plug-ins are an important part of Facebook's platform architecture, enabling the decentralization of platform functionality and data produced on the platform and the recentralization of data produced outside of the platform (Gerlitz & Helmond, 2013). By embedding a plug-in into their website, webmasters set up two-way data channels, data pours, in which data flows between the site and Facebook's database. Technically, a social plug-in functions as an API call (Helmond, 2013) and sends specific requests to Facebook's platform, for example, get the total number of people who liked this post or publish a new like after clicking the Like button.

Making Data Platform Ready

Before these plug-ins can interface with Facebook's database from an external website, webmasters need to make their websites compatible with Facebook's platform infrastructure. To do so, webmasters need to embed a piece of JavaScript code into their websites which sets up a data communication channel with Facebook's platform. This code initiates the Facebook Software Development Kit (SDK) for using social plug-ins, Facebook login, and making API calls to the database (Facebook Developers, n.d.-e). In doing so, webmasters are making their pages platform ready for data communication with Facebook. This notion of making external websites and web data platform ready extends Gillespie's (2014) idea of how data are made "algorithm ready" (p. 168) to highlight the role of the platform infrastructure in reconfiguring external data to fit the agenda of the platform.

Another important part of Facebook's platform infrastructure is Facebook's Open Graph, which is explicitly geared toward making external data platform ready. The Open Graph "lets you integrate apps deeply into the Facebook experience, which increases engagement, distribution and growth" (Facebook Developers, n.d.-d). To integrate an app, developers need to use the Facebook SDK and Facebook Login to set up relations between the app, Facebook, and the user (Facebook Developers, n.d.-d). This integration lets apps tell "stories" on Facebook such as "Mary ran 6 miles with MyRunningApp" (Facebook Developers, n.d.-d). Apps submit these stories to the Open Graph in a very structured manner, organized around four elements, for example: John (actor) is reading (action) The Odyssey (the object) on Goodreads (app). There are a number of predefined actions such as "like," "watch," and "read," but developers can also create their own actions. Bucher (2012b) describes these efforts from Facebook "as a way to build a semantic map of the Internet" (p. 5). The app integrations enable Facebook to collect external app data and activities in a very structured manner, send it back to the database, and connect it to a user or to other data. It further expands Facebook's data collection techniques into external applications and formats these data according to the logic of the platform, so that it can be put into new relations within the platform.

Webmasters can also make their websites platform ready by marking up their sites with Open Graph tags (Facebook Developers, n.d.-b). These meta tags provide Facebook's crawler with "structured info about the page such as the title, description, preview image, and more" and control how content appears on Facebook to "improve distribution and engagement" (Facebook Developers, n.d.-c). Similar to the search engine optimization (SEO) practices of webmasters, making websites "Facebook ready" can be seen as a form of social media optimization.

The Open Graph shows how Facebook strictly structures data flowing from apps and external websites to the platform in order to make it platform ready. While platforms position themselves as neutral intermediaries or utilities (Gillespie, 2010; Van Dijck, 2013), they (pre)format data passing through their infrastructure according to the logic of their underlying infrastructures.

Dual Logic of Platformization

The previous examples have shown how Facebook employs its platform as an infrastructural model to extend

itself into external online spaces and how it employs these extensions to format data for its platform to fit their economic interest through the commodification of user activities and web and app content. This platformization, I argue, rests on the dual logic of social media platforms' expansion into the rest of the web and, simultaneously, their drive to make external web and app data platform ready. As an infrastructural model, social media platforms provide a technological framework for others to build on, geared toward connecting to and thriving on other websites, apps and their data. At the same time, readying external data for their own databases is central to the economic model of social media platforms. These two processes of decentralizing platform features and recentralizing platform ready data characterize what I call the double logic of platformization. This double logic is operationalized through platform-native objects such as APIs, social plugins, and the Open Graph, which connect the infrastructural model of the platform to its economic aims. These elements serve as prime devices for social media platforms to expand into the web and to create data channels—data pours—for collecting and formatting external web data to fit the underlying logic of the platform.

By proposing a material–technical perspective on platforms, I have shown the "work that platforms do" not in a rhetorical sense (cf. Gillespie, 2010) but in a computational sense. The notion of platformization has been introduced as a way to critique the consequences of the programmability of platforms. This has been a first exploration in that area showing how social media platforms are enacting their programmability to reweave the web for social media.

Acknowledgements

I would like to thank the two anonymous reviewers and the guest editors, Tarleton Gillespie and Hector Postigo, for their valuable comments. Additionally, I would like to thank Richard Rogers, Carolin Gerlitz, and Bernhard Rieder for their feedback on earlier drafts.

Declaration of Conflicting Interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article.

Notes

1. "Platform Politics" is the title of a conference held at Anglia Ruskin University in Cambridge, United Kingdom, on 12-13 May 2011. The conference was organized by Josh Hands and Jussi Parikka and brought together a number of scholars studying the politics of platforms. Following the conference, *Culture Machine* published a special issue titled "Platform Politics" (2013).
2. According to Facebook, the platform had over 936 million daily active users on average for March 2015 ("Company Info," n.d.). Facebook.com ranks number 2 in Alexa's "top 500 sites on the web" which is "calculated using a combination of average daily visitors and pageviews over the past month" ("The top 500 sites on the web," 2015).
3. See Michael Stevenson's (2014) critique on the web's alleged "participatory turn."
4. In his seminal book titled *The Language of New Media*, Lev Manovich (2001) argues that all new media objects are numerical representations and that this makes media programmable. This programmability is key to the principles underlying new media: numerical representation, modularity, automation, variability, and cultural transcoding.
5. This draws our attention to the use of platform as a recursive concept, as put forward by software developer Dave Winer (1995), whom from early on saw the Internet as a meta-platform or "platform machine" which can be used to build new platforms on.
6. More than 30 million apps and websites have integrated with Facebook's platform (D. Liu, 2015).
7. The industry refers to this as the "API Economy" or "[t]he emerging economic effects enabled by companies, governments, nonprofits and individuals using APIs to provide direct programmable access to their systems and processes" (Willmott & Balas, 2013).
8. An exception within media studies is the work of Taina Bucher (2013) on "Objects of Intense Feeling: The Case of the Twitter API" in which she provides a historical background of the role of APIs in software engineering and briefly discusses early public web APIs.
9. On 25 March 2015, Facebook launched Messenger Platform which "enables developers to easily build apps that integrate with Messenger," Facebook's messaging app. Developers can plug their app into Messenger using the Messenger Platform, Facebook's new Level 2 platform.
10. In their developer documentation, Facebook explains how this works:
    Apps on Facebook are web pages loaded into a Canvas frame. The Canvas frame is simply a blank canvas within Facebook on which to run your app. You populate the Canvas frame by providing a Canvas URL and Secure Canvas URL that contains the HTML, JavaScript and CSS that make up your app. These will be used by users browsing Facebook over HTTP and HTTPS respectively. When a user loads your Canvas app on Facebook, we load the Canvas URL within an iframe on that page. This results in your app being displayed within the standard Facebook chrome. (Facebook Developers, n.d.-a)
11. The Canvas URL points to the external host where the app is located which is then loaded within an iFrame in Facebook.
12. Andreessen's examples of Level 3 Runtime Environment platforms include Salesforce which allows users to inject their own code and Andreessen's (2007b) own Ning platform "for creating and running social networking applications." Despite Andreessen's claim that all "platforms are good, period," he does state that "I call these Internet platform models 'levels', because as you go from Level 1 to Level 2 to Level 3, as I will explain, each kind of platform is harder to build, but much better for the developer." In this sense, he promotes Level 3 platforms, including his own Ning, as being the "best" platforms for developers.

13. Level 3 "Runtime Environment" platforms are mostly located in the business-to-business domain such as Salesforce or Amazon.
14. The structure of an XML document looks as follows:
<book category="Fiction">
<title lang="en">Emma</title>
<author>Jane Austen</author>
<year>1916</year>
</book>
15. XML is at the core of several important data exchange mechanisms on the web, including XML-RPC and SOAP. The XML-RPC protocol is based on the idea of remote procedure calls (RPC) to "provide for transfer of control and data across a communication network" (Birrell & Nelson, 1984, p. 39). It was developed in 1998 by Dave Winer from Userland and Microsoft to make requests to a remote computer and exchange data on the web (Laurent, Johnston, Dumbill, & Winer, 2001, p. x). Out of their work on XML-RPC came SOAP, Simple Object Access Protocol, a "lightweight protocol used to exchange XML-encoded information" (Laurent et al., 2001, p. 172). XML-RPC- and SOAP-based web services enable the exchange of structured data between different machines on the web by communicating via the HTTP transmission protocol. Recently, JSON has become the preferred format over XML to transmit data, as it considered a more lightweight format. In addition, the architectural style REST, Representational State Transfer, has gained prominence for building web services. For example, social media platform Twitter offers a REST-based API which returns data in JSON.
16. See previous footnote. RSS is a web syndication format for websites and blogs to publish a feed of their latest content. It is based on XML, see http://cyber.law.harvard.edu/rss/rss.html.

References

Allen, M. (2013). What was Web 2.0? Versions as the dominant mode of internet history. *New Media & Society*, *15*, 260-275. doi:10.1177/1461444812451567

Amazon. (2002, July 16). *Amazon.com launches web services* [Press Release]. Retrieved from http://phx.corporate-ir.net/phoenix.zhtml?c=176060&p=irol-newsArticle&ID=503034

Anderson, P. (2012). *Web 2.0 and beyond: Principles and technologies*. Boca Raton, FL: CRC Press.

Andreessen, M. (2007a, June 12). *Analyzing the Facebook Platform, three weeks in [Blog post]*. Retrieved from https://web.archive.org/web/20071002070223/http://blog.pmarca.com/2007/06/analyzing_the_f.html

Andreessen, M. (2007b, September 16). *The three kinds of platforms you meet on the Internet [Blog post]*. Retrieved from https://web.archive.org/web/20071002031605/http://blog.pmarca.com/2007/09/the-three-kinds.html

Arrington, M. (2008, September 15). *Facebook isn't a social network: And stop trying to make new friends there [Blog post]*. Retrieved from http://techcrunch.com/2008/09/15/facebook-isnt-a-social-network-and-dont-try-to-make-new-friends-there/

Baldwin, C. Y., & Clark, K. B. (2000). *Design rules: The power of modularity*. Cambridge, MA: MIT Press.

Baldwin, C. Y., & Woodard, C. J. (2009). The architecture of platforms: A unified view. In A. Gawer (Ed.), *Platforms, markets and innovation* (pp. 19-44). Cheltenham, UK: Edward Elgar Publishing Limited.

Beer, D. (2009). Power through the algorithm? Participatory web cultures and the technological unconscious. *New Media & Society*, *11*, 985-1002. doi:10.1177/1461444809336551

Beer, D., & Burrows, R. (2007). Sociology and, of and in Web 2.0: Some initial considerations. *Sociological Research Online*, *12*(5). Retrieved from http://dx.doi.org/10.5153/sro.1560

Benslimane, D., Dustdar, S., & Sheth, A. (2008). Services mashups: The new generation of web applications. *IEEE Internet Computing*, *12*(5), 13-15.

Birrell, A. D., & Nelson, B. J. (1984). Implementing remote procedure calls. *ACM Transactions on Computer Systems*, *2*, 39-59. doi:10.1145/2080.357392

Bodle, R. (2011). Regimes of sharing. *Information, Communication & Society*, *14*, 320-337. doi:10.1080/1369118X.2010.542825

Bogost, I., & Montfort, N. (2009). Platform studies: Frequently questioned answers. *Digital Arts and Culture*. Retrieved from http://escholarship.org/uc/item/01r0k9br.pdf

boyd, d. (2007, June 24). *Viewing American class divisions through Facebook and MySpace [Blog post]*. Retrieved from http://www.danah.org/papers/essays/ClassDivisions.html

boyd, d., & Ellison, N. (2008). Social network sites: Definition, history, and scholarship. *Journal of Computer-Mediated Communication*, *13*, 210-230. doi:10.1111/j.1083-6101.2007.00393.x

Bruns, A. (2008). *Blogs, Wikipedia, second life, and beyond: From production to produsage*. New York, NY: Peter Lang.

Bucher, T. (2012a). *Programmed sociality: A software studies perspective on social networking sites* (Doctoral dissertation). University of Oslo, Oslo, Norway.

Bucher, T. (2012b). A technicity of attention: How software "makes sense." *Culture Machine*, *13*, 1-13. Retrieved from http://www.culturemachine.net/index.php/cm/article/viewArticle/470

Bucher, T. (2013). Objects of intense feeling: The case of the Twitter API. *Computational Culture, 3*. Retrieved from http://computationalculture.net/article/objects-of-intense-feeling-the-case-of-the-twitter-api

Cheng, X., Dale, C., & Liu, J. (2008). Statistics and social network of YouTube videos. In *16th International Workshop on Quality of Service (IWQoS)* (pp. 229-238). New York, NY: IEEE. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4539688

Chun, W. H. K. (2011). *Programmed visions: Software and memory*. Cambridge, MA: MIT Press.

Company Info. (n.d.). Retrieved from http://newsroom.fb.com/company-info/

Cramer, F., & Fuller, M. (2008). Interface. In M. Fuller (Ed.), *Software studies: A lexicon* (pp. 149-152). Cambridge, MA: MIT Press.

DuVander, A. (2012, January 4). *Over 2,000 APIs added* in 2011: Social, telephony, open government [Blog post]. Retrieved from http://www.programmableweb.com/news/over-2000-apis-added-2011-social-telephony-open-government/2012/01/04

Evans, D. S., Hagiu, A., & Schmalensee, R. (2006). *Invisible engines: How software platforms drive innovation and transform industries*. Cambridge, MA: MIT Press.

Facebook Developers. (n.d.-a). *Canvas*. Retrieved from https://developers.facebook.com/docs/games/canvas/

Facebook Developers. (n.d.-b). *Facebook content sharing best practices*. Retrieved from https://developers.facebook.com/docs/sharing/best-practices

Facebook Developers. (n.d.-c). *A guide to sharing for webmasters*. Retrieved from https://developers.facebook.com/docs/sharing/webmasters

Facebook Developers. (n.d.-d). *Open Graph overview*. Retrieved from https://developers.facebook.com/docs/opengraph/overview/

Facebook Developers. (n.d.-e). *Quickstart: Facebook SDK for JavaScript*. Retrieved from https://developers.facebook.com/docs/javascript/quickstart/v2.2

Facebook Help. (n.d.). *What are social plugins?* Retrieved from https://www.facebook.com/help/103828869708800

Facebook Newsroom. (2006, May 3). *Facebook expands to include work networks*. Retrieved from https://newsroom.fb.com/news/2006/05/facebook-expands-to-include-work-networks-2/

Facebook Newsroom. (2007, May 24). *Facebook unveils platform for developers of social applications*. Retrieved from https://newsroom.fb.com/news/2007/05/facebook-unveils-platform-for-developers-of-social-applications/

Fetterman, D. (2006, August 15). *Facebook development platform launches*. Retrieved from https://www.facebook.com/notes/2207512130

Gawer, A. (2014). Bridging differing perspectives on technological platforms: Toward an integrative framework. *Research Policy*, *43*, 1239-1249. doi:10.1016/j.respol.2014.03.006

Gehl, R. (2010). *A cultural and political economy of Web 2.0* (Doctoral dissertation). George Mason University, Fairfax, VA.

Gehl, R. (2012). Real (software) abstractions on the rise of Facebook and the fall of MySpace. *Social Text*, *30*, 99-119.

Geminder, K. (2007, June 2). *Platform is here*. Retrieved from https://www.facebook.com/notes/facebook/platform-is-here/2437282130

Gerlitz, C., & Helmond, A. (2013). The Like economy: Social buttons and the data-intensive web. *New Media & Society*, *15*, 1348-1365. doi:10.1177/1461444812472322

Gillespie, T. (2010). The politics of "platforms." *New Media & Society*, *12*, 347-364. doi:10.1177/1461444809342738

Gillespie, T. (2014). The relevance of algorithms. In P. Boczkowski, K. Foot, & T. Gillespie (Eds.), *Media technologies* (pp. 167-194). Cambridge, MA: MIT Press.

Hagiu, A. (2014). Strategic decisions for multisided platforms. *MIT Sloan Management Review*, *55*, 71-80.

Hands, J. (2013). Introduction: Politics, power and "platformativity." *Culture Machine, 14*. Retrieved from http://www.culturemachine.net/index.php/cm/article/viewArticle/504

Helmond, A. (2013). The algorithmization of the hyperlink. *Computational Culture*, issue 3. Retrieved from http://computationalculture.net/article/the-algorithmization-of-the-hyperlink

Hicks, M. (2010, April 21). *Building the social web together*. Retrieved from https://www.facebook.com/notes/facebook/building-the-social-web-together/383404517130

Jenkins, H. (2006). *Convergence culture: Where old and new media collide*. New York: New York University press.

Kirkpatrick, D. (2010). *The Facebook effect: The real inside story of Mark Zuckerberg and the world's fastest growing company*. New York, NY: Random House.

Lane, K. (2012, December 20). *History of APIs*. Retrieved from http://history.apievangelist.com/

Lane, K. (n.d.). *About Kin Lane*. Retrieved from http://kinlane.com/about/

Langlois, G., & Elmer, G. (2013). The research politics of social media platforms. *Culture Machine*, *14*, 1-17. Retrieved from http://www.culturemachine.net/index.php/cm/article/viewArticle/505

Langlois, G., Elmer, G., McKelvey, F., & Devereaux, Z. (2009). Networked publics: The double articulation of code and politics on Facebook. *Canadian Journal of Communication*, *34*(3). Retrieved from http://www.cjc-online.ca/index.php/journal/article/viewArticle/2114

Langlois, G., McKelvey, F., Elmer, G., & Werbin, K. (2009). Mapping commercial Web 2.0 worlds: Towards a new critical ontogenesis. *Fibreculture*, issue 14. Retrieved from http://fourteen.fibreculturejournal.org/fcj-095-mapping-commercial-web-2-0-worlds-towards-a-new-critical-ontogenesis/

Laurent, S. S., Johnston, J., Dumbill, E., & Winer, D. (2001). *Programming web services with XML-RPC*. Sebastopol, CA: O'Reilly Media, Inc.

Liu, A. (2004). Transcendental data: Toward a cultural history and aesthetics of the new encoded discourse. *Critical Inquiry*, *31*, 49-84. doi:10.1086/427302/

Liu, A. (2008). *Local transcendence: Essays on postmodern historicism and the database*. Chicago, IL: University of Chicago Press.

Liu, D. (2015, March 25). *F8 2015: New ways to connect with the Facebook family of apps*. Retrieved from https://newsroom.fb.com/news/2015/03/f8-day-one-2015/

Locke, L. (2007). *The future of Facebook*. Retrieved from http://content.time.com/time/business/article/0,8599,1644040,00.html

Lomborg, S., & Bechmann, A. (2014). Using APIs for data collection on social media. *The Information Society*, *30*, 256-265. doi:10.1080/01972243.2014.915276

Madden, M., & Fox, S. (2006). Riding the waves of "Web 2.0." *Pew Internet and American Life Project, 5*. Retrieved from http://www.pewinternet.org/files/old-media/Files/Reports/2006/PIP_Web_2.0.pdf.pdf

Manovich, L. (2001). *The language of new media*. Cambridge, MA: MIT Press.

McKelvey, F. (2011). A programmable platform? Drupal, modularity, and the future of the web. *Fibreculture, 18*. Retrieved from http://eighteen.fibreculturejournal.org/2011/10/09/fcj-128-programmable-platform-drupal-modularity-and-the-future-of-the-web/

Montfort, N., & Bogost, I. (2009). *Racing the beam: The Atari video computer system*. Cambridge, MA: MIT Press.

Murugesan, S. (2007). Understanding Web 2.0. *IT Professional*, *9*(4), 34-41. doi:10.1109/MITP.2007.78

Myllymaki, J. (2002). Effective web data extraction with standard XML technologies. *Computer Networks*, *39*, 635-644.

O'Reilly, T. (2005). *What is Web 2.0: Design patterns and business models for the next generation of software*. Retrieved from http://oreilly.com/web2/archive/what-is-web-20.html

O'Reilly, T., & Battelle, J. (2004). *Opening welcome: The state of the internet industry*. Presented at the Web 2.0 Conference, Hotel Nikko, San Francisco, CA. Retrieved from http://web-2con.com/presentations/web2con/intro_tim_john.ppt

Puschmann, C., & Burgess, J. (2013). *The politics of Twitter data* (HIIG Discussion Paper Series No. 2013-01). Retrieved from http://papers.ssrn.com/abstract=2206225

Renzi, A. (2011). What is the politics of platform politics? *Television & New Media*, *12*, 483-485.

Richardson, L., & Ruby, S. (2008). *RESTful web services*. Sebastopol, CA: O'Reilly Media, Inc.

Rieder, B., & Sire, G. (2014). Conflicts of interest and incentives to bias: A microeconomic critique of Google's tangled position on the Web. *New Media & Society*, *16*, 195-211. doi:10.1177/1461444813481195

Rochet, J.-C., & Tirole, J. (2003). Platform competition in two-sided markets. *Journal of the European Economic Association*, *1*, 990-1029. doi:10.1162/154247603322493212

Rogers, R. (2013). *Digital methods*. Cambridge, MA: MIT Press.

Stevenson, M. (2014). Rethinking the participatory web: A history of HotWired's "new publishing paradigm," 1994-1997. *New Media & Society*. Advance online publication. doi:10.1177/1461444814555950

Tokuda, L. (2009, June). *Social applications: Growth, use, and monetization*. Presented at the Global ICT Summit 2009, Tokyo, Japan. Retrieved from http://www.ict-summit.jp/2009/pdf/report6_tokuda.pdf

The top 500 sites on the web. (2015, June 14). Retrieved from http://www.alexa.com/topsites

Ullrich, C., Borau, K., Luo, H., Tan, X., Shen, L., & Shen, R. (2008). Why web 2.0 is good for learning and for research: Principles and prototypes. In *Proceedings of the 17th International Conference on World Wide Web* (pp. 705-714). New York, NY: ACM. Retrieved from http://dx.doi.org/10.1145/1367497.1367593

Van Dijck, J. (2013). *The culture of connectivity: A critical history of social media*. New York, NY: Oxford University Press.

Vis, F. (2013). A critical reflection on Big Data: Considering APIs, researchers and tools as data makers. *First Monday*, *18*(10). Retrieved from http://firstmonday.org/ojs/index.php/fm/article/view/4878

W3C. (n.d.). *XML Essentials*. Retrieved from http://www.w3.org/standards/xml/core

Willmott, S., & Balas, G. (2013) *Winning in the API economy*. Retrieved from http://www.3scale.net/wp-content/uploads/2013/10/Winning-in-the-API-Economy-eBook-3scale.pdf

Winer, D. (1995, August 22). *What is a platform?* Retrieved from http://scripting.com/davenet/1995/08/22/whatisaplatform.html

YouTube. (2005, August 21). *August 2005*. Retrieved from http://youtube-global.blogspot.nl/2005_08_01_archive.html

Author Biography

Anne Helmond (M.A., University of Amsterdam) is a PhD candidate and lecturer of New Media and Digital Culture at the University of Amsterdam. Her research interests include software studies, platform studies, digital methods, and web history.