A QoE Perspective on HTTP/2 Server Push

Torsten Zimmermann, Benedikt Wolters, Oliver Hohlfeld
Communication and Distributed Systems, RWTH Aachen University
{zimmermann,wolters,hohlfeld}@comsys.rwth-aachen.de

ABSTRACT

HTTP/2 was recently standardized to optimize the Web by promising faster Page Load Times (PLT) as compared to the widely deployed HTTP/1.1. One promising feature is HTTP/2 *server push*, which turns the former pull-only into a push-enabled Web. By enabling servers to preemptively push resources to the clients without explicit request, it promises further improvements of the overall PLT. Despite this potential, it remains unknown if server push can indeed yield human perceivable improvements.

In this paper, we address this open question by assessing server push in both *i)* a laboratory and *ii)* a crowdsourcing study. Our study assesses the question if server push can lead to *perceivable* faster PLTs as compared to HTTP/1.1 and HTTP/2 without push. We base this study on a set of 28 push-enabled real-word websites selected in an Internet-wide measurement. Our results reveal that our subjects are able to perceive utilization of server push. However, its usage does not necessarily accomplish perceived PLT improvements and can sometimes even be noticeably detrimental.

CCS CONCEPTS

• **Networks** → **Application layer protocols**; **Network measurement**; • **Information systems** → *World Wide Web*;

KEYWORDS

Quality of Experience; HTTP/2 Server Push; User Study

ACM Reference format:
Torsten Zimmermann, Benedikt Wolters, Oliver Hohlfeld. 2017. A QoE Perspective on HTTP/2 Server Push. In *Proceedings of Internet QoE '17, Los Angeles, CA, USA, August 21, 2017,* 6 pages.
DOI: https://doi.org/10.1145/3098603.3098604

1 INTRODUCTION

With the advent of HTTP/2 as a replacement of HTTP/1.1, the Web is currently experiencing a major protocol shift. Given its simplicity, HTTP/1.1 (H1 for the remainder) has become the first choice [17] for realizing a plethora of desktop and mobile applications, which is reflected by traffic shares of $\geq 50\,\%$ in Internet's core [1]. Despite its widespread use, the increasing complexity of the Web lets the 20 year old H1 suffer from inefficiencies impacting the Page Load Time (PLT), especially with regard to parallelization, e.g., Head-of-Line Blocking (HOL Blocking). As a result, techniques such as *domain sharding*, i.e., distribution of content to different servers to allow for connections in parallel, or *resource inlining*, i.e., merging resources into the page itself to save requests, emerged. However, even those techniques introduce problems, such as under-utilized connections or not providing the possibility to cache inlined resources [9].

To tackle the inefficiencies of H1, its successor HTTP/2 (H2) [2] was standardized in 2015, with the reduction of PLT as its primary goal. New components of H2 are request and response *multiplexing* into parallel streams, stream *prioritization*, and *header compression*. It has been shown that these components can lead to (perceivable) PLT improvements, but also to PLT degradation [3, 21, 22]. One key feature is H2 *server push*, which enables servers to speculatively push objects to the client without an explicit request. By saving request-response round-trip times for these objects, it has the potential to reduce the overall PLT. Despite this potential, it remains unclear if push can indeed lead to human *perceivable* PLT improvements.

The goal of this paper is therefore to address the following open questions in a user study: *i)* Can H2 server push yield perceivable PLT improvements relative to H1 and H2 without server push in a pairwise comparison? *ii)* How is the perceived PLT improvement in line with underlying existing technical metrics, e.g., SpeedIndex? *iii)* Can server push be the differentiator of faster perception between H1 or H2? Answering these questions is relevant to assess whether current engineering and standardization efforts are indeed sufficient to optimize the web browsing experience. The contributions of this paper are as follows:

(1) We provide the first assessment of H2 push on end-user perception. By assessing if push-enabled websites are also perceived to load faster than their H1 or plain H2 counterparts, we answer the question on whether push is an engineering effort worth pursuing to optimize the Web. Therefore, we conducted a controlled laboratory (28 participants) and a crowdsourcing study (323 participants). These pave the way for follow-up quality assessment-studies.
(2) Our findings indicate that H2 push is no silver bullet to optimize the Web. When used properly, it can result in websites that are perceived to load faster. However, we also identified situations in which push has opposite effects. These findings provide the baseline to derive guidelines on proper push usage to optimize end-user experience.

We present information on the history of H2 and its characteristics in Section 2. Subsequently, we summarize related work regarding the global H2 adoption, Web QoE, and the current use of H2 server push in Section 3. Moreover, we describe how we selected websites as the basis for our study in Section 4 and discuss our study design in Section 5 in detail. We present and analyze the study results in Section 6 and conclude our work in Section 7.

Internet QoE '17, August 21, 2017, Los Angeles, CA, USA

2 BACKGROUND: H2 AND SERVER PUSH

H2: The initial starting point and some of H2's key features are rooted in Google's attempt to replace H1 by SPDY. Meanwhile, SPDY has converged to H2, which has been finalized and standardized in May 2015 [2]. The major difference between H2 and its predecessor H1 is that it is a binary instead of an ASCII protocol, thus enabling easier framing and more efficient processing [9]. In addition, H2 is able to *multiplex* requests and responses on parallel *streams*, identified by stream IDs, over a single TCP connection. These streams can be prioritized, depending on their importance for the browser's rendering process. H2 preserves the H1 paradigm of a stateless protocol, i.e., header information for the same connection can be repetitive. To reduce this redundancy overhead, H2 enables *header compression* [16]. Although not explicitly required by the standard, all major implementations use H2 over TLS [9] for security reasons, which increases protocol overhead.

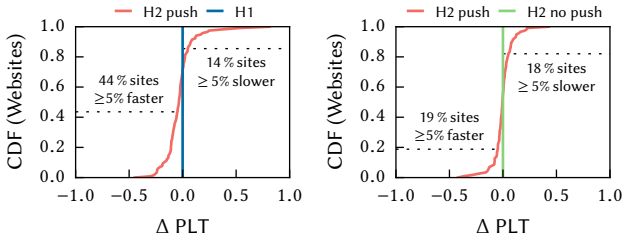
Server Push: Aiming to optimize the user experience by reducing the overall latency, H2 adds the *server push* feature. It enables servers to preemptively *push* resources to the clients without explicit request, e.g., also send a stylesheet upon index request, thus saving request round-trip times. This is in contrast to the traditional pull-only Web, where the browser would first request the body page, parse it, and then explicitly request all embedded objects. Thus, the potential to improve the PLT and the overall web experience render server push a key feature of H2. Its importance motivates us to assess if push can indeed yield perceivable PLT improvements.

3 RELATED WORK

H2 Adoption & Performance. The first empirical understanding of H2 and SPDY adoption for *popular* domains was provided by scanning the Alexa Top 1M list [22]. These scans indicate a growth of H2 adoption from 1.4 % in November 2014 to 3.6 % in October 2015 [22]. We assess this adoption in a follow-up study [24] and find only few push-enabled websites. Besides this adoption analysis, a much larger body of related work focused on the performance of H2 (or its predecessor SPDY) compared to H1 in various network setups or real world traces [6, 8, 22, 23]. All of these works utilized PLT as *objective metric* to compare website performance. Performance increases and even decreases (expressed in PLT) are observed, e.g., depending on the website's structure or object sizes.

Web QoE. Approaches to map PLT to *user perceived* quality estimates are the focus of many Web QoE studies (see e.g., [18]) and the ITU-T standard G.1030. Beyond, studies have investigated time-dependent properties of PLT in Web QoE [12] and identified situations in which the PLT is insufficient to capture the Web QoE (see e.g., [10, 14]). Particularly for H2, it has been shown that *i)* H2 can yield noticeable QoE improvements [21] but *ii)* that technical metrics such as PLT do not always capture QoE and thus new models are needed [3]. Beyond the PLT, the websites' visual appeal [19, 20] has been shown to be a key influence factor.

H2 Server Push. While the assessment of H2 as defined in the standard [2] has recently received attention, the implementation and usage of push as key web performance optimization feature is not (yet) defined. Thus, the critical question on *which* resources to push *when* (push policies) is left to developers and system administrators. In the absence of standardization, recent works attempt at filling this gap by providing first push policies. With an additional signaling of the client's current cache state to the server, server push can yield bandwidth savings in cellular networks [11, 15]. In the absence of such an additional signalling channel, the server can select pushable objects based on their size [5]. Other works proposed to use push for quickly delivering objects with high priority at an early stage of the web request, based on tracking object dependencies [4, 23]. Despite these first efforts, the question on when and how to use push remains far from being settled. Importantly, it is also unknown if push as key feature indeed is a silver bullet to optimize the web experience, motivating our study.



(a) H2 vs H1. **(b) H2 push vs. H2 w/o push.**

Figure 1: Speedup of H2 with *push* vs. H1 and H2 *without* push for 526 Websites using H2 server push

4 SELECTING PUSHING WEBSITES

Since it is unknown which push policies are currently used in practice and to base our study on a realistic set of push-enabled websites, we start by performing Internet-wide measurements to identify H2 enabled websites using push.

Internet-wide measurement. To probe the Internet for H2 support, we scanned domains from the *Alexa 1M* and the complete *.com/.net/.org* set, and additionally scanned the entire IPv4 space using *ZMap* [7]. As of January 2017, this provided us with a set of 5.38 M sites supporting H2. Subsequently, we visited each site using nghttp2—an H2 capable C library—and identified 7 K websites pushing resources on their landing page. Further inspections showed that ≈ 6.5 K sites belong to a domain parker, which uses the same template and push behavior on each site, leaving us with 526 *unique* websites. For more specific information about the measurement methodology, we refer to [24].

To turn our 1 Gbit/s campus network into a more typical home user environment, we use the Linux tc utility to limit the network bandwidth to 16 Mbit/s downlink and 1 MBit/s uplink. We additionally added 50 ms symmetric delay. Finally, we automate Chromium 58 using the Selenium framework to visit these websites and measure their PLT in an end-user browser. We measure the PLT as the difference between the `navigationStart` and `loadEventStart` events obtained from the W3C Navigation Timing API. The aforementioned procedure is repeated for different protocol settings, e.g., *H1*, *H2 with push*, and *H2 without push*, for 31 repetitions each. Since push is enabled in Chromium by default, we modified the browser to disable/enable push via the `SETTINGS_ENABLE_PUSH` protocol flag. Moreover, each individual run starts with a fresh browser profile, a cold browser cache, and QUIC disabled (to enforce TCP as the same transport protocol for both H1 and H2).

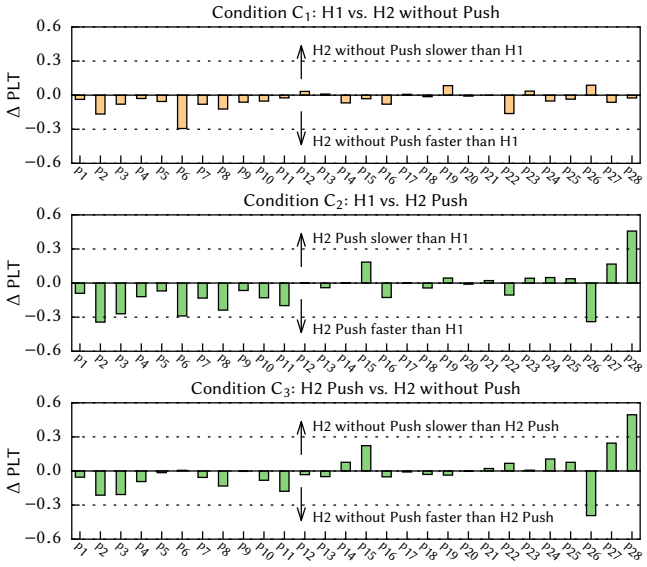


Figure 2: Relative difference in PLT (Δ PLT) for the websites (p_1-p_{28}) selected for the user study.

Figure 1 shows the measured PLT as the relative median increase ($\Delta PLT < 0$ on the left) and median decrease ($\Delta PLT > 0$ on the right) for the three conditions. We select a threshold of 5 % PLT difference to decide, if a website was faster or slower than its respective counterpart. These results already indicate that some websites benefit from push, while other detriment—expressed by *slower* PLTs. Assessing if these increases or decreases are perceivable is the core of our study.

Website selection. Out of the overall set of 526 push-enabled sites, we selected 28 for our study as follows: We *i)* exclude websites where the PLT measurements exhibit low stability, i.e., overlapping deviations from the mean. Moreover, we *ii)* select websites with a reasonable loading time, i.e., PLT < 10 s. Finally, we *iii)* grouped websites into three groups: *(a)* websites that either exhibit no or only marginal PLT improvement (i.e., < our 5% threshold), and websites that *(b)* exhibit an increase or *(c)* a decrease in PLT for various protocol comparisons. The resulting set consists of 28 websites from various categories including online shops, entertainment, personal blogs, or news. Figure 2 depicts the relative median PLT difference across the different conditions. Following [3], we inspect each website to look complete and correct (e.g., no missing objects) and that no offensive or adult content is included.

5 STUDY DESIGN

To assess *perceivable differences* in the loading times of the same websites transferred via two protocol variants (e.g., H2 push vs. H2 without push), we design a *pair-wise comparison study* following [21]. That is, we display the loading process of both variants of a website as video side-by-side and ask the subject "Which version loaded faster?" with the options "Left", "No Difference", and "Right". This pair-wise comparison allows to detect even subtle differences in the loading process by push. Which, if exist, can then be assessed in a follow-up quality assessment to derive a QoE model.

Condition	Definition
C_0	*Control* (same video, e.g., H1 vs. H1)
C_1	*H1 vs. H2 without push* (or vice versa)
C_2	*H1 vs. H2 with push* (or vice versa)
C_3	*H2 w/o push vs. H2 with push* (or vice versa)

Table 1: Tested study conditions

Study	Users	Gender	Age	Expertise	Online [h]
		♀, ♂, ○	<25, 25-31, >31	−, ∅, +	<4, 4-8, >8
Lab	28	6, 21, 1	6, 20, 2	0, 9, 19	7, 11, 10
Crowd	323*	72, 246, 5	143, 119, 61	7, 95, 221	86, 130, 107

Table 2: Overview of study participants. *Resulting set *after* applying all filter levels.

Conditions. The study is based on our selection of 28 websites (see Section 4), which were requested via different underlying protocols, i.e., *H1*, *H2 with push*, and *H2 without push* resembling our study conditions C_1–C_3 (see Table 1). Our additional control condition C_0 shows the same loading process on the left and on the right side and we expect the participants to select *No Difference*.

Loading process displayed as video. To ensure that every subject rates the same stimuli, we record videos of the websites' loading and rendering process. This way, we fix the rated loading process to guarantee that it is *not* influenced by any network characteristics out of our control. To record the videos, we utilize the browsertime[1] framework to capture the browser's window. We capture each website in each condition C_1–C_3 31 times and select the video corresponding to the median PLT over all visits. Finally, we generate *single* videos of side-by-side combinations of this set of three videos following the Cartesian product, i.e., we create *one* video showing *two* website loads side-by-side to guarantee a synchronized playback. Thereby, we obtain a total of 9 videos per website, i.e., 6 videos showing condition C_1–C_3 and 3 videos showing the control condition C_0 for each of the three protocol variants. Furthermore, we provide the option to *replay* a video before submitting a vote.

Study procedure. First, we show a screen explaining the study procedure. Second, we obtain the following demographic data: *i)* Gender, *ii)* level of Internet expertise, *iii)* age, and *iv)* time spent online per day. The last two items are presented as pre-defined intervals (e.g., age "25–31"). Last, we show the videos, one video per-screen.

Implementation. We implemented our study as a website using the TheFragebogen.de framework, which is a HTML5-based javascript framework to realize paperless questionnaires. Following crowdsourcing best practices [13], we instructed the framework to capture additional metadata: *i)* time spend on each rating screen, *ii)* focus changes (e.g., leaving the browser window), *iii)* screen resolution, *iv)* stalling events or loading errors, *v)* number of user-triggered replays per video. Moreover, we save information about the browser and the timezone that was configured at the client. Our study is available at https://userstudy.comsys.rwth-aachen.de

5.1 Laboratory Study Design

Environment. To obtain ratings from a controlled environment complementing the (potentially noisy) crowdsourcing study, we first conducted a laboratory study at our institute located in Aachen (Germany). The study was performed in an office environment on

[1] https://github.com/sitespeedio/browsertime

a fixed desktop computer with a 19" TFT screen and a mouse to complete the questionnaire. To limit the effect of changing light conditions from outside, we lowered the blinds and switched on the ceiling lights. The examiner was present during the study to answer questions and notice anomalies in the procedure. We interviewed the participants afterwards, if they noticed something special or if there were any problems in participating in the study.

Design. Our study involved 28 unpaid participants, summarized in Table 2. For *age* and *time spent online per day*, we show the median selected interval and merge the remaining selections. The study design follows a latin-square within-subject design. That is, we selected one condition per website that was assessed by every subject in random order. The assessed 28 different websites include all conditions C_0–C_3 with a particular focus on C_3 (16 websites) to assess the impact of push. The median study duration per participant was 273.72 s. Note that we assess the full range (i.e., all conditions per website) in our crowdsourcing study.

5.2 Crowdsourcing Study Design

To obtain a larger user base and to assess all conditions for each website, we additionally performed a crowdsourcing study. The study was advertised via social networks to reach a diverse set of participants. We further targeted the e-mail list of students attending lectures at our institute. In contrast to the laboratory study, the participants now only rate a random subset of 8 websites to keep the study duration short (median duration per participant 114.37 s). Out of these, one website (chosen at random) always serves as control condition C_0. The remaining 7 stimuli show websites subject to conditions C_1–C_3 (website and condition chosen at random).

Filter levels. Following best practices [13], we use filter levels to remove unreliable participants. We *i)* remove participants with corrupted or incomplete data. Next, we *ii)* remove participants that changed the focus away from our study website, and we *iii)* remove all participants that voted but did not watch the video until the first visual change. In the last—and most conservative—step, we *iv)* remove all participants that provided wrong answers (i.e., not selecting *No Difference*) to the control condition C_0. We summarize information on participants remaining in the crowd study after filtering in Table 2 (82 out of 405 participants removed).

6 RESULTS

Notation. We refer to individual web pages in our set as p_i, where $i \in [1, 28]$ denotes the page number. To measure the agreement, we define *verdict* as the average over all votes per condition. We encode *each* vote as follows: −1 for left, 0 for no difference, and +1 for right. As a result, a verdict of 1 (−1) indicates that the right (left) video is perceived faster (100% of all votes). A verdict of 0 indicates either no difference (i.e., 100% of all votes) or disagreement (e.g., 50% of all votes for left and 50% for right).

6.1 General Observations

We begin by comparing the vote distribution of lab vs. crowd participants. Therefore, we compare the verdict for every condition tested in our lab study to the same conditions in our crowdsourcing study in Figure 3. We remark that a larger number of conditions per website was tested in our crowdsourcing study and thus only

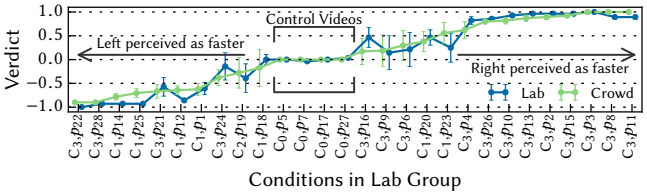


Figure 3: Voting verdict lab vs. crowd study. Error bars show the agreement as relative share of votes not following the trend of the verdict, e.g., if the verdict is < 0, votes ≥ 0 count as error = disagreement.

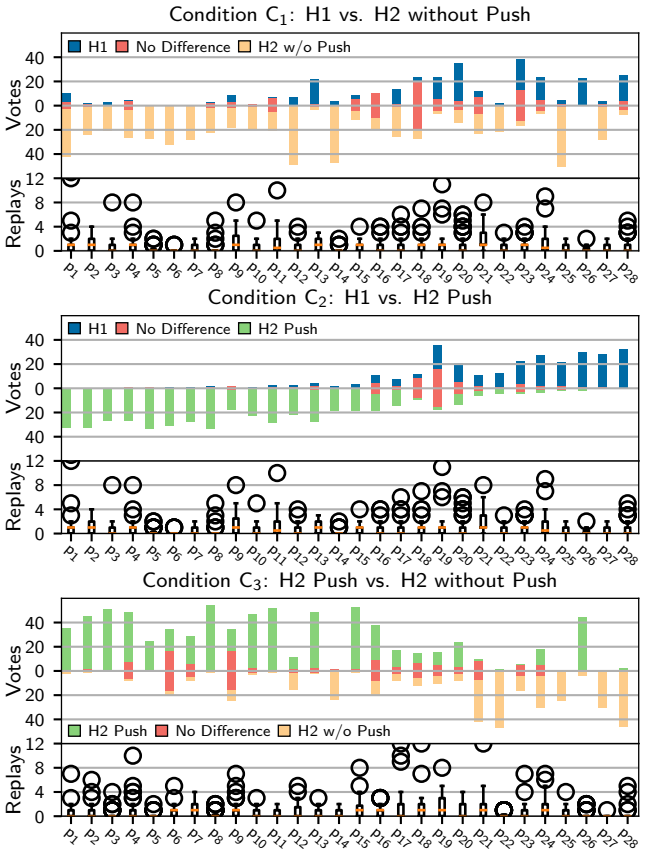


Figure 4: Combined results (Lab + Crowd) for all 28 websites and 3 conditions. The y-axis shows support for both protocol variants above or below 0 and is centered at *No Difference* (0). Moreover, we show how often users replayed the video.

a subset can be compared. A similar voting distribution in both participant populations can be observed (median error between both groups is 0.095 ± 0.085), suggesting that votes of our crowd participants (after filtering) are as reliable as of our lab participants. Thus, we combine both populations in the remainder of this section to reason on a larger population.

Further, we observe agreement among participants for a larger number of tested conditions, indicated by verdicts of ≈ −1 (≈ 1) and of 0 for the control condition. We therefore analyze these conditions in detail and start by depicting the absolute number of votes for

each page p_i for every condition C_1–C_3 in Figure 4 for the combined data set. Since the websites and conditions in the crowd study were chosen at random for every participant, their number of votes differs with an average of 34.83 votes and a minimum of 17 votes. Moreover, the average number of user-triggered video replays per user correlates slightly ($r = 0.58$) to the verdict, i.e., videos with a low verdict presenting cases that are more challenging to distinguish (in particular no difference scenarios) have slightly higher replay counts. Moreover, we observe agreement in participants' verdict for a large set of scenarios, i.e., in 58.33% (84.52%) of scenarios the vote is more than 75% (50%) in favor of one video. However, this level of agreement differs by condition.

6.2 H1 vs. H2: Is Push the decisive Factor?

H1 vs. H2. Comparing user votes between each individual condition allows us to evaluate whether push is the determining factor in the user-perceived performance between H1 and H2. Thus, we investigate whether there are differences between votes for H1 and H2 (cf. Figure 4, *top* vs. *middle*), when push is enabled or disabled.

First, we observe that H2 push was perceived *faster* for pages p_1–p_{16} by $\geq$ 59.09% of votes compared to H1 (C_2, green bars). However, considering the case H1 vs. H2 *without* push (C_2), we see that those websites already had a vote towards H2 (> 50%) and likely already benefit from other H2 properties than push. Only p_{13} and p_{15} exhibit changes in their votes in favor of H2, when push is enabled. In this case push contributes to the faster perception of H2. In the following, we discuss possible effects causing this resulting perception for p_{13} in detail.

Case Study 1: p_{13} is a sports website with only 13 resources. Here we find H1 to perform well since we observe only little HOL Blocking (< 10 ms). When using H2, the server does not adhere to the stream priorities specified by the browser. As a result, some requested resources are not delivered in the order requested by the browser, impacting the rendering performance and diminishing the overall H2 performance (H1 is faster). However, with H2 push the server pushes two render critical stylesheets, that contain links to font files. Without push the browser would have to request the stylesheets explicitly and subsequently after retrieving the stylesheets request the missing font files. Thereby, pushing the stylesheets reduces the overall time (H2 push is faster than H1).

Moreover, some pages do not benefit from push, when compared to H1: we observe that for p_{21}–p_{28} H1 was perceived faster (>50 % of votes) than H2 with push (C_2, cf. Figure 4, *middle*). This observation can be attributed to push, e.g., for p_{22}, p_{25}, or p_{27} the use of push is perceived as slower than H1 while H2 without push is perceived faster than H1.

H2 vs. H2 push. When presented with C_3, i.e., H2 push vs. H2 without push, 10 (15) pages were perceived to load faster *with* push by > 80% (> 50%) of the votes (green bars in Figure 4, *bottom*). Here, websites benefit from using push. However, push can also yield detrimental effects: 5 (9) pages were perceived faster *without* push by > 80% (> 50%) of the votes (yellow bars). Here push lets the participants *not* perceive the web faster. In the following, we discuss the possible technical reasons for one exemplary website.

Case Study 2: p_{22} is an example of harmful push usage. Six render-critical resources are pushed, i.e., the resources are necessary to render above-the-fold content. However, those pushed resources do not only arrive in an suboptimal order for the browser to process, they also delay the initial HTML response, hindering the browser to discover additional resources beyond the pushed resources (e.g. 3rd party content) required to render. When comparing H1 and H2 with push, the *firstRender* is delayed by 277 ms in the case of push, i.e., the page is rendered partially sooner but incomplete. Interestingly, although the page starts to render later in the H2 case the overall rendering process is faster using H2 push compared to H1, still 70.59 % subjects voted in favor of the earlier H1 scenario.

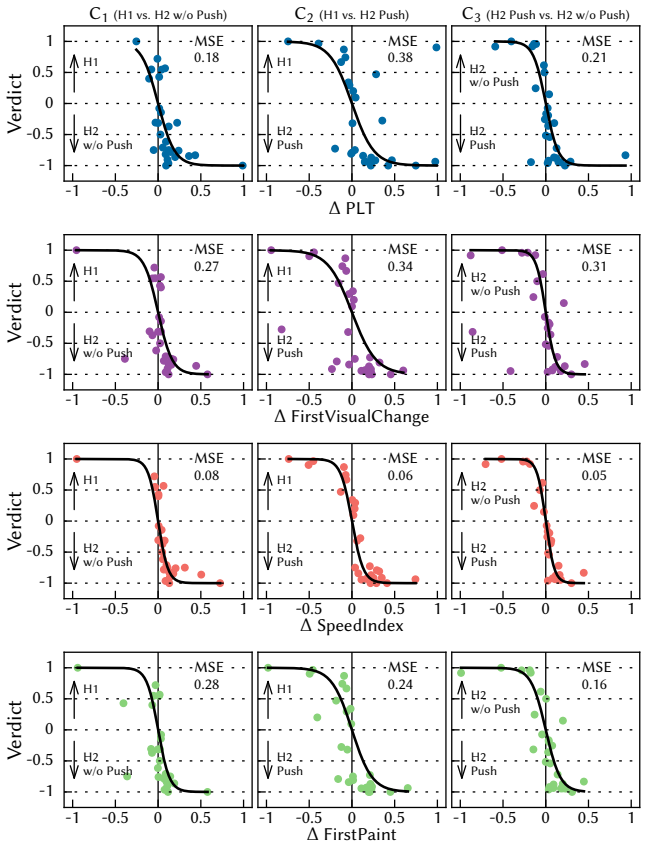


Figure 5: Subjective user ratings vs. technical metrics. Goodness of the fitted functions indicated by MSE. We show the absolute normed difference for the metrics.

Comparing the results for C_3 with C_2 further allows us to identify cases where detrimental push usage does not change the overall perception between H1 and H2. Exemplary, for p_{14} push is perceived as slower. However, this does not change the faster perception of H2 in general for this page. Conversely, p_{26} indicates that even though H2 push was perceived faster than H2, H1 is still perceived faster than H2 push.

6.3 Metrics vs. Perception

Typically, websites are assessed by technical metrics (e.g., PLT, Google's SpeedIndex, Time of FirstVisualChange / FirstPaint). Hence, we are interested if these are correlated to the subjects' votes (i.e., verdict), as our initial measurement identified pages where

push has positive as well as negative effects on the PLT (recall Figure 1(b)). Therefore, we obtain several technical metrics from the browser and plot these against subjects' votes in Figure 5. We observe that the vote is correlated to the relative difference in the metric, following a fitted sigmoid function. Moreover, we note a tendency towards one scenario when the intra-scenario difference between the underlying metrics is high (e.g., large PLT differences). In other words, the higher the difference in the objective metric, the higher the likelihood that the difference is being perceived by our participants. Still, we observe few outliers where the relative difference in the metric *disagrees* with the subject's vote. Exemplary, this is the case for PLT (C_2, upper right quadrant) and FirstVisualChange (C_3, lower left quadrant) in Figure 5. For the case of web engineering, these outliers can, however, still be relevant. This is in line with related work that focused on the mapping of technical to subjective metrics [3, 10, 14], indicating that technical metrics often fail to predict human perceived quality. As an example, we provide two case studies for concrete outliers in the following.

Case Study 3: For p_{15} the median PLT of H2 push is 345 ms *slower* than H2 without push, however, the participants still prefer the slower H2 push (94.44 % of votes). This is a case where the underlying metric significantly diverges from the user perception. Examining the page, however, we find that the time of the FirstVisualChange and firstPaint is significantly faster in the H2 push case than the PLT. In the push case Chrome spends more time processing the pushed resources before requesting an analytics javascript that does not contribute to the visual progress of the page.

Case Study 4: For p_8 the time of the FirstVisualChange is 618 ms slower in the push case. Still, 96.43 % voted in favor of H2 push. We find that the page consists of 6 resources including a small image and two fonts. The majority of page content is textual and thus the fonts are required to display the text. By inspecting the captured video, we find that in the H2 case without push the small image is partially displayed sooner but the text renders significantly later. In the push case stylesheets and fonts are pushed, thereby enabling the browser to render the main content of the page faster, but the time of FirstVisualChange is delayed.

These examples highlight the complexity of proper push usage that should optimize subjective perception by humans rather than technical metrics. Moreover, during our analysis, we noticed that technical metrics do not necessarily correlate, further motivating a human-centred rather than technical optimization of websites.

7 CONCLUSION

The goal of our work is to elucidate the influence of H2 server push on the human perceived loading times of websites. Based on our results, we argue that push is no general silver bullet to optimize the Web. When used properly, it can lead to websites that are perceived to load faster than their H1 or plain H2 counterparts. However, we also observe cases in which push leads to slower loading times and to websites that are not perceived to load faster. Here, the usage of push is detrimental for the website operator and we find reasons for this to be rooted in numerous page-specific aspects.

Further, we find that only optimizing technical metrics (e.g., PLT) does not generally result in websites that are also *perceived* to load faster. Currently, little is known on *optimal* server push usage. Hence, we posit that a deeper understanding of server push is needed to provide guidelines on appropriate usage. As illustrated in the analysis of our study, these guidelines need to not only incorporate technical metrics, but also account for end-user perception.

ACKNOWLEDGEMENTS

We thank our shepherd Amogh Dhamdhere and the anonymous reviewers for their insightful comments and suggestions. Moreover, we would like to thank Peter Hedenskog and Tobias Lidskog for their effort in developing browsertime and thus supporting our study. This work has been funded by the DFG as part of the CRC 1053 MAKI and the European Union's Horizon 2020 research and innovation program 2014–2018 under grant agreement No. 644866 ("SSICLOPS"). It reflects only the authors' views and the European Commission is not responsible for any use that may be made of the information it contains.

REFERENCES

[1] B. Ager, N. Chatzis, A. Feldmann, N. Sarrar, S. Uhlig, and W. Willinger. Anatomy of a Large European IXP. In *ACM SIGCOMM 2012*.
[2] M. Belshe, R. Peon, and M. Thomson. 2015. Hypertext Transfer Protocol Version 2 (HTTP/2). RFC 7540. (18 Nov. 2015).
[3] E. Bocchi, L. De Cicco, M. Mellia, and D. Rossi. The Web, the Users, and the MOS: Influence of HTTP/2 on User Experience. In *PAM 2017*.
[4] M. Butkiewicz, D. Wang, Z. Wu, H. V. Madhyastha, and V. Sekar. KLOTSKI: Reprioritizing Web Content to Improve User Experience on Mobile Devices. In *NSDI 2015*.
[5] I. N. de Oliveira, P. T. Endo, W. Melo, D. Sadok, and J. Kelner. Should I Wait or Should I Push? A Performance Analysis of Push Feature in HTTP/2 Connections. In *ACM LANCOMM Workshop 2016*.
[6] H. de Saxcé, I. Oprescu, and Y. Chen. Is HTTP/2 really faster than HTTP/1.1?. In *IEEE INFOCOM (WKSHPS) 2015*.
[7] Z. Durumeric, E. Wustrow, and J. A. Halderman. ZMap: Fast Internet-wide Scanning and Its Security Applications. In *USENIX Security 2013*.
[8] J. Erman, V. Gopalakrishnan, R. Jana, and K. K. Ramakrishnan. 2015. Towards a SPDY'Ier Mobile Web? *IEEE/ACM Transactions on Networking* 23, 6 (2015), 2010–2023.
[9] I. Grigorik. 2013. *High Performance Browser Networking*. O'Reilly.
[10] D. Guse, S. Schuck, O. Hohlfeld, A. Raake, and S. Möller. Subjective quality of webpage loading: The impact of delayed and missing elements on quality ratings and task completion time. In *QoMEX 2015*.
[11] B. Han, S. Hao, and F. Qian. MetaPush: Cellular-Friendly Server Push For HTTP/2. In *Workshop on All Things Cellular 2015*. ACM.
[12] T. Hoßfeld, S. Biedermann, R. Schatz, A. Platzer, S. Egger, and M. Fiedler. The memory effect and its implications on Web QoE modeling. In *ITC 2011*.
[13] T. Hoßfeld, C. Keimel, M. Hirth, B. Gardlo, J. Habigt, K. Diepold, and P. Tran-Gia. 2014. Best Practices for QoE Crowdtesting: QoE Assessment With Crowdsourcing. *IEEE Transactions on Multimedia* 16, 2 (Feb 2014), 541–558.
[14] C. Kelton, J. Ryoo, A. Balasubramanian, and S. R. Das. Improving User Perceived Page Load Times Using Gaze. In *NSDI 2017*.
[15] J. Khalid, S. Agarwal, A. Akella, and J. Padhye. Improving the performance of SPDY for mobile devices. In *ACM HotMobile Poster 2015*.
[16] R. Peon and H. Ruellan. 2015. HPACK: Header Compression for HTTP/2. RFC 7541. (31 Dec. 2015).
[17] L. Popa, A. Ghodsi, and I. Stoica. HTTP as the narrow waist of the future Internet. In *ACM SIGCOMM HotNets 2010*.
[18] D. Strohmeier, S. Egger, A. Raake, T. Hoßfeld, and R. Schatz. 2014. Web Browsing. In *Quality of Experience: Advanced Concepts, Applications and Methods*, Sebastian Möller and Alexander Raake (Eds.). Springer, 329–338.
[19] M. Varela, T. Mäki, L. Skorin-Kapov, and T. Hoßfeld. Towards an understanding of visual appeal in website design. In *QoMEx 2013*.
[20] M. Varela, L. Skorin-Kapov, T. Mäki, and T. Hoßfeld. 2015. QoE in the Web: A dance of design and performance. In *QoMEX 2015*.
[21] M. Varvello, J. Blackburn, D. Naylor, and K. Papagiannaki. EYEORG: A Platform For Crowdsourcing Web Quality Of Experience Measurements. In *CoNEXT 2016*.
[22] M. Varvello, K. Schomp, D. Naylor, J. Blackburn, A. Finamore, and K. Papagiannaki. Is the Web HTTP/2 Yet?. In *PAM 2016*.
[23] X. S. Wang, A. Balasubramanian, A. Krishnamurthy, and D. Wetherall. How speedy is SPDY?. In *NSDI 2014*.
[24] T. Zimmermann, J. Rüth, B. Wolters, and O. Hohlfeld. How HTTP/2 Pushes the Web: An Empirical Study of HTTP/2 Server Push. In *IFIP Networking 2017*.