Open
Web Advocacy

OWA - Mobile Browsers and Cloud Gaming - Response to Remedies Working Paper

VERSION 1.0

Open Web Advocacy
contactus@open-web-advocacy.org

1. Table of Contents

2. Introduction

We would like to thank the Market Investigation Reference (MIR) team for investigating these critical topics and for the thoughtful analysis that they have published. Much of the work done by the MIR is groundbreaking as they explore topics, some of which have never before been publicly investigated by any other regulator. We are grateful for both the detailed analysis in working papers 1 - 6 and the remedies being discussed in working paper 7.

Both browser and Web App competition is currently being undermined on iOS and Android to the detriment of UK consumers and businesses.

We strongly support almost all of the proposed remedies. These remedies collectively strike at the heart of the lack of browser competition on mobile devices in the UK.

> *"We all rely on browsers to use the internet on our phones, and the engines that make them work have a huge bearing on what we can see and do. Right now, choice in this space is severely limited and that has real impacts – **preventing innovation and reducing competition from web apps**. We need to give innovative tech firms, many of which are ambitious start-ups, a fair chance to compete."*
> Andrea Coscelli - Chief Executive of the UK's Competition and Markets Authority
> (emphasis added)

We are however deeply concerned that without additional remedies and the strengthening of particular existing remedies that the innovation and competition from Web Apps discussed in the opening statements of the MIR will not be achieved. In particular we believe that the following are needed in addition to the remedies proposed by the MIR:

- Obligation to allow browsers to install and manage Web Apps with their own engine.
- Obligation to implement install prompts for iOS Safari.
- Obligation to allow third-party browsers to provide feature parity to Web Apps.
- Prevent effective ban of third-party browser engines via unfair contracts.
- Direct install for browsers on iOS and Android.
- A strengthening of the A3 API remedy to cover any API that a browser vendor might reasonably need to implement performance, stability, functionality, security or privacy.

We would like to again thank the MIR team for allowing all parties to see these potential remedies laid out in such detail and the opportunity to raise these concerns.

The Web, while dominating on desktop has been prevented from achieving its true potential on mobile to the advantage and profit of a small number of gatekeepers. These remedies, collectively, have the potential to change the trajectory of both mobile browsing and software development, not just for the UK, but for the entire world. These changes will lead to more competition, higher quality, cheaper, more private and more secure software for all consumers and businesses in the UK.

3. Additional Required Remedies

3.1. Allow Browsers to install and manage Web Apps with their own Engine

In order to allow effective Web App competition a number of remedies are needed. The first two, allowing browsers to use their own engines on iOS and allowing access to needed OS software and hardware APIs have been correctly identified by the MIR team.

In the case of Google not sharing WebAPK Minting, this may be sufficient as the system they have created internally relies on Android Custom Tabs which supports other browsers using their own engine, out of the box. Google simply hasn't shared it with any other browsers. While Google may require code updates to share this, this does appear to be well covered by the API rule.

In Apple's case however, their Web App solution is hardwired to the version of WebKit they bundle with iOS, it contains none of the plumbing required to allow other browsers using their own engines to install or manage Web Apps. Worse, Apple has indicated via their public interactions in relation to the EU's Digital Market Act that they do not intend to allow third party browsers to use their own engines.

> *"This support means Home Screen web apps continue to be built directly on WebKit and its security architecture, and align with the security and privacy model for native apps on iOS."*
>
> Apple's statement

This means that Apple could plausibly allow third-party browsers to use their own engines on iOS and then simply share the existing share menu with them. That is, if a user has installed for example Firefox (using its own engine), when the user attempts to install a Web App, that Web App would run using the existing WebKit code. Any functionality, performance, stability, security, privacy that Firefox had added to their browser would not be available in the Web App.

What is needed here is an explicit legally binding directive for Mobile OS gatekeepers to allow third party browsers to install and manage Web Apps using their own engines. While Google is likely covered under the API access rule, it would be sensible to include them anyway to prevent future problems such as them entirely removing Web Apps, or updating their implementation to hardcode it to their Blink WebView.

As such, we believe the following remedy is appropriate, justified and proportionate to restore competition in provision of functionality, performance, stability, security, privacy to Web Apps via third-party browsers. We ask that the MIR team consider including it:
"Mobile OSes shall allow third-party browsers to install and manage Web Apps using their own browser engine."

Additionally, Apple should not be able to introduce any friction to the install process for Web Apps and each browser should be allowed to implement their own version of install prompts (including for example Web App stores).

3.2. Implement Install Prompts for Safari

In order for Web Apps to fairly contest the iOS App Store, they need to be as visible as native apps. Apple has progressively added features to iOS Safari to push users towards native apps on their app store, however refuses to add the equivalent feature for Web Apps: Install Prompts, despite incredible pressure from developers over many years.

Install Prompts are the essential missing feature for Web Apps, without which Web Apps will not be able to compete with native apps.

Currently the installation of Web Apps in Safari is completely hidden away under an awkward multi-step process which the user must access through the “share” menu. The install process for Web Apps are so obscured in Safari that the majority of users are unaware of their existence.

In order for Web Apps to be able to fairly contest Apple’s native app ecosystem, and to enable businesses to take over the extensive advantages Web Apps have to offer (including Security/Interoperability/Cost) **Apple must be compelled to provide an effective implementation of Install Prompts from within Safari.** Apple will not provide this without regulatory intervention.

As such, we believe the following remedy is appropriate, justified and proportionate to allow Web Apps to compete with the OS’s apps, app store, services and apps delivered via their app store. We ask that the MIR team consider including it:
"A requirement for Apple to implement Install Prompts for iOS Safari.”

We cover this in more detail in section “3.3.1. Install Prompts” in our previous submission “OWA - Mobile Browsers and Cloud Gaming - Response to Working Papers 1-6”.

3.3. Obligation to allow Third-Party Browsers to provide Feature Parity to Web Apps

Apple has repeatedly claimed that Web Apps are the alternative distribution method for apps in their mobile ecosystem.

> *"QUESTION: Apple is the sole decision maker as to whether an app is made available to app users though the Apple Store, isn't that correct?*
> *REPLY: If it's a Native App, yes sir, if it's a Web App no."*
>
> *Tim Cook - Speaking to US Congress*

> *"Web browsers are used not only as a distribution portal, but also as platforms themselves, hosting "progressive web applications" (PWAs) that eliminate the need to download a developer's app through the App Store (or other means) at all."*
>
> Apple Lawyers - Court Filing in Australia

> *"For everything else there is always the open Internet. If the App Store model and guidelines are not best for your app or business idea that's okay, we provide Safari for a great web experience too."*
>
> Apple App Store Guidelines

Even if the CMA and other regulators compel Apple to allow third party browsers using their own engine to power Web Apps, only Apple can really build and maintain the integration into the many surfaces that apps on iOS inhabit.

These surfaces include (but are not limited to):

- The Web App icons, text, badging on the homescreen
- The settings/permissions page of each Web App
- Notifications from the Web App
- Menus to uninstall or move the Web App
- Various search menus
- Various defaults (i.e. default email client, default map)

Only Apple (the developers of iOS) have control over each of these surfaces. We are concerned that if Apple became nervous as to the potential for Web Apps to undermine its app store that Apple might take steps to lessen Web Apps appeal via these surfaces. That is, while allowing competition in browser engines powering Web Apps solves most issues, i.e. functionality that takes place within the Web App, it is important that Apple is not allowed to make Web Apps second tier citizens via their control and design of the integration into these surfaces.

The aim here is to allow browsers to facilitate Web Apps being true substitutes and competitors to native apps distributed via Apple's app store. This should not be undermined by Apple.

The CMA in their mobile ecosystems study noted that Apple has an incentive to prevent Web Apps from competing and as far back as 2011 Apple executives were concerned about the threat the Web presents to their app store.

With all of these facts in mind, we believe it is reasonable and proportional to have a general remedy that:
"Where feature parity between Web Apps and native apps is possible, Apple must technically enable it and it should not be artificially prevented either by OS rules or OS design. Apple must not self-preference their own apps, apps sold via their app store or their own services over Web Apps."

3.4. Prevent Effective Ban of Third-Party Browser engines via Contract

Throughout most of the world, Apple explicitly bans third party browsers from using their own browser engine.

> *"2.5.6 Apps that browse the web must use the appropriate WebKit framework and WebKit JavaScript. You may apply for an entitlement to use an alternative web browser engine in your app. Learn more about these entitlements."*
>
> Apple App Store Guidelines
> (Recently Updated Text)

Recently, Apple has updated the wording of rule 2.5.6. On first glance it might appear they have allowed eligible browsers to globally be updated with their own engines. However they have in actual fact just moved the ban inside their browser engine entitlement contract which makes it explicit that only browsers within the EU are eligible.

This is due to the fact that the app store rules were specifically updated in order to be compliant with the Digital Markets Act which prohibits such a ban. However, as we extensively argue in our paper, they have an effective ban still in place via making the contract for the APIs required by browsers using their own engines so unreasonable, restrictive and scary that no browser vendor would dare sign it for a production browser.

Some non-exhaustive examples of problems with the contract are:
- Vendors must abide by all rules in "Apple Materials", a vague term encompassing thousands of documents that Apple can update at any time with no notice.

- Any breach of the rules (no matter how minor) grants Apple the right to remove all of the apps of the browser vendor from every operating system that Apple controls.
- Browser vendors can not update their existing browser and must ship a new EU only browser, thus losing all their customers.
- Apple can remove the browser vendor's browser for any reason.
- Apple can break or remove any API (including from a single browser vendor) at any time at its sole discretion.
- Browser vendors are required to rebuild their browsers on particular iOS components such as scroll. (this one was recently silently removed by Apple by changing one word in API documentation)

What is key here, is that even if the UK mandates that Apple can no longer have such a ban, we are concerned that Apple will try a similar strategy within the UK where they instead switch to a quasi-ban by unreasonable contract terms. The solution is to keep Apple strictly to terms which relate to protecting security and to closely scrutinize the necessity or proportionality of any term. Terms that would severely undermine browser competition should be struck out.

We would propose the remedy that *"Apple and Google can only apply strictly necessary, proportionate, and justified security measures to browsers using their own engine. All security rules and their justifications must be publicly published. All security rules for browser vendors should be available in a single public up-to-date document. Changes to these rules should be subject to regulatory scrutiny"*.

3.5. Direct Install for Browsers on iOS and Android

Another important avenue to improve browser competition on iOS and Android is to allow browser vendors to also offer their browsers outside the app stores. Essentially browsers that have been granted the browser entitlement, would be able to offer their browsers directly from their own websites, where users could be prompted to install them. This is already obligated in the EU under the Digital Markets Act. Thus Apple and Google will simply need to turn on this functionality for the UK or globally.

This is important as there is great fear in the industry that Apple wields app store review as a weapon to punish competitors. This fear appears to be well justified.

On March 5th, Spotify submitted an update to Apple that puts links to Spotify's website, along with pricing information for different subscription options, directly in the EU version of its app, without using Apple's payment system. Despite being fined 1.8 billion by the EU on this very topic, Apple refused to let the update pass app review even though more than 2 weeks had passed since the update was submitted.

> *"Moreover, Apple has demonstrated its ability to use its smartphone monopoly to impose fee structures and* ***manipulate app review to inhibit aggrieved parties from taking advantage of regulatory and judicial solutions imposed on Apple*** *that attempt to narrowly remedy harm from its conduct."*
>
> DOJ - Case 2:24-cv-04055
> (emphasis added)

> *"Specifically, Apple sets the conditions for apps it allows on the Apple App Store through its App Store Review Guidelines. Under these guidelines, Apple has sole discretion to review and approve all apps and app updates. Apple selectively exercises that discretion to its own benefit, deviating from or changing its guidelines when it suits Apple's interests and allowing Apple executives to control app reviews and decide whether to approve individual apps or updates.* ***Apple often enforces its App Store rules arbitrarily. And it frequently uses App Store rules and restrictions to penalize and restrict developers that take advantage of technologies that threaten to disrupt, disintermediate, compete with, or erode Apple's monopoly power.****"*
>
> DOJ - Case 2:24-cv-04055
> (emphasis added)

> *"there are endless horror stories around curation of the store. Apps are rejected in arbitrary, capricious, irrational and inconsistent ways, often for breaking completely unwritten rules."*
>
> Benedict Evans - Technology Writer

> *"There's a lot of talk about the 30% tax that Apple takes from every app on the App Store. The time tax on their developers to deal with this unfriendly behemoth of a system is just as bad if not worse"*
>
> Samantha John - CEO Hopscotch

Given that third-party browsers using their own engine will likely need to comply with extensive security rules in order to be allowed access to the relevant APIs, there is no security justification for any additional scare screens or warnings.

Currently, only a few browser vendors offer their browser through the macOS app store. It seems plausible that browsers will want to abandon the iOS app store at some point in the future to avoid having to deal with any unreasonable app store guidelines that Apple might impose.

We would propose the remedy: *“Apple and Google shall allow browsers meeting strictly necessary, proportionate and justified security conditions the ability to be installed directly from their own websites. Taking advantage of this should not impose penalties on the browser vendor nor block them from offering their apps on Apple and Google’s app stores or financially penalize them (e.g. ‘core technology fee’).”.*

Browser vendors should be allowed to opt to distribute the direct distribution version of their browser in the choice screen (without penalty from Apple or Google).

Browser vendors need the appropriate APIs and access to be able to effectively push automatic updates (without unreasonable delay) via a direct distribution channel.

4. Review of Proposed Remedies

4.1. Issue 1 – Apple's WebKit restriction

4.1.1. A1 - Allow Alternative Browser Engines on iOS

> *"A1 - Requirement for Apple to grant access to alternative browser engines to iOS."*
> Browsers and Cloud MIR - WP7

We strongly support this remedy.

There is effectively no browser competition on iOS. This is due to the fact that Apple has banned third party browsers from choosing or modifying their own engines.

Apple has done this by rule 2.5.6 of their app store guidelines:

> *"Apps that browse the web must use the appropriate WebKit framework and WebKit Javascript."*

This very innocuous rule in fact forces the entire content area of a browser and the majority of functionality important to Web Apps to be exclusively controlled by Apple. This essentially makes all browsers on iOS reskinned versions of Safari. What Apple is saying with this clause is, *"Throw away the browser you have spent 100,000+ hours working on, and build a 'thin' user interface shell around Safari which we have total control over and that you can't modify".*

> *"**Apple has a browser monopoly on iOS**, which is something Microsoft was never able to achieve with IE"*
> Scott Gilbertson - The Register
> (emphasis added)

> *"because **WebKit has literally zero competition on iOS, because Apple doesn't allow competition**, the incentive to make Safari better is much lighter than it could (should) be."*
> Chris Coyier - CSS Tricks
> (emphasis added)

> *“Today, you can download what look like "alternative" browsers on iOS, like Chrome and Firefox, **but these browsers are mostly just skins overtop of Apple's Safari engine**. iOS app developers aren't actually allowed to include their own browser engines, so everything uses Safari's WebKit engine, with a new UI and settings and sync features layered on top.”*
>
> Ron Amadeo - ArsTechnica
> (emphasis added)

While this does allow some minor features to be implemented such as bookmarks and sync, browsers are unable to compete in performance, stability, functionality, security or privacy in almost all aspects. Consumers are mostly unaware of this and thus Apple has maintained this mirage of competition for over a decade.

> *A mobile browser engine is the core software component of a mobile browser that handles the rendering and display of web content. The browser engine is responsible for processing HTML, CSS, and JavaScript code, and rendering websites into the visual format that users see on their mobile devices. **In practical terms, this means the main reason that websites may look, load and work differently in different browsers is their browser engines.**”*
>
> Browsers and Cloud MIR WP1
> (emphasis added)

This is critically important as both developers and consumers have no recourse if Safari is missing a feature, has a severe bug or security vulnerability. Consumers can not switch to a better browser as all other browsers are forced to use Safari’s core, the WebKit WKWebView.

This lack of ability to switch means that there is no real risk of Safari losing users due to bugs, missing features or security issues. As such Apple faces no “effective competition” to improve, beyond the bare minimum not to cause embarrassment. This removes all incentive to heavily invest in Safari to improve features and stability.

Some commentators point to Android as a source of browser competition. The argument being, Android has no such ban on third party browser engines, and as such if a consumer were unhappy with the fact that Apple has effectively banned their favorite browser, they could switch to Android where the browser would be available.

This unfortunately does not work. Mobile ecosystems have tremendous lock-in, consumers have low awareness of the mechanics of how browsers work and even fewer consumers are aware that Apple has effectively banned third party browsers.

This was highlighted in the consumer research done by Verian:

> *"These findings are consistent with qualitative consumer research conducted by Verian for this market investigation, which noted the topic of browsers on users' smartphones was a low salience topic that had rarely been considered, if noticed at all, by respondents. This supports the view that the availability of specific browsers and browser engines on a mobile device and their quality likely plays a limited role in users' decisions to purchase mobile devices (and in driving competition between mobile ecosystems)."*
>
> Browsers and Cloud MIR WP1

The WP1 report correctly assesses this, and states that *"On this basis, it is our emerging thinking that the supply of mobile browsers on iOS and Android should be considered as two separate product markets."*. OWA agrees with this assessment.

For all of the reasons above we support this remedy. This remedy is needed in combination with other remedies in order to allow effective browser and Web App competition.

4.1.2. API Access for Third-Party Browsers on iOS

> *"A2 - Requirement for Apple to grant equivalent access to iOS to browsers using alternative browser engines."*
>
> Browsers and Cloud MIR - WP7

> *"Under this option, Apple would also have flexibility about how to implement the requirement. In order to provide equivalent access, Apple could choose to create new APIs replicating the functionalities and features made available to WebKit and Safari or it could choose to give access to some of the existing private APIs that exist as internal interfaces within iOS.*
>
> *Creating new APIs which replicate the access granted to WebKit (and Safari) may provide a better option than granting access to existing APIs for third-party browser engines. This is because the significant integration that exists at present between WebKit, Safari and iOS could make extending existing APIs to third-party browsers insufficient to achieve equivalent access."*
>
> Browsers and Cloud MIR - WP7

The primary distinction between A2 and A3 appears to be that under A2, Apple would be able to keep various APIs private to Safari/WebKit on the condition that they produced equivalent APIs or functionalities and made them available to third-party browsers.

We believe that A3 is a far stronger remedy and is required to allow fair browser competition on iOS. In order to remove the ability of Apple to provide preferential treatment to their own browser, it is important that third-party browsers can make use of all of the same APIs as Apple. Allowing Apple to make and dictate custom APIs for third-party browsers to use would be a nightmare for any regulator to oversee due to the vast number and complexity of the games a company seeking to slow or undermine browser competition could play.

Regulators need only look to Apple's compliance with the DMA for examples of such games.

Given that Apple is already required to implement a stronger version of A3 in the EU under the Digital Markets Act, the additional cost for implementing it in the UK is zero. This is an important fact when considering proportionality.

As such, we strongly prefer A3 to A2.

4.1.3. API Access for Third-Party Browsers on iOS (No Private APIs)

> *"A3 - Requirement for Apple to grant equivalent access to APIs used by WebKit and Safari to browsers using alternative browser engines."*
>
> [Browsers and Cloud MIR - WP7]
>
> *"Apple would be required to eliminate its use of private APIs for WebKit and Safari without degrading currently available functionality made available for WebKit and Safari.*
>
> *5.33 Apple previously noted that it does not have a means of estimating the extent of investment required to ensure that third-party developers have access to the same WebKit and iOS functionality available to Safari, [].*
>
> *5.34 As noted above, it is unclear at this stage whether extending access to currently private APIs to third-party browsers would be the best option of enabling access to third party browser engines."*
>
> [Browsers and Cloud MIR - WP7]

As discussed in the previous section we support this remedy. It is critical to allow third-party browsers the ability to compete fairly.

We also believe that scoping it to the features Safari/WebKit have access to is problematic. While it is important that they have that as a minimum, browsers have a high likelihood of needing APIs outside of what Safari/WebKit use, in order to support features that Safari/WebKit do not support.

At a minimum, browsers should additionally have access to all the APIs that standard native apps on iOS have access to.

Browsers should then be able to request and have the right of access to any software or hardware API that they can justify a clear need of to support particular functionality, performance, stability, security or privacy.

We are not asking for browsers to have unrestricted access. APIs can and should be designed to grant only the access that is actually needed. These APIs can be subject to strictly necessary, proportionate and reasonable security measures.

In the event that Apple has not locked down some of the private APIs that its own apps use, upgrading them to more carefully lockdown access will also improve the security of Apple's own apps.

A hypothetical example would be to allow browsers access to an API to secure enclave for the purpose of implementing better payment handling. This would not mean that browsers could access other app's data stored in secure enclave, rather this API would allow the browser to store and retrieve its own data.

4.2. Issue 2 – API Access for Browsers

4.2.1. Equivalent access to APIs used by Chrome.

> *"A4 - Requirement for Google to grant equivalent access to APIs used by Chrome."*
>
> Browsers and Cloud MIR - WP7
>
> *"The aim of the remedy for Google would be to provide equivalent access and functionality to third party browsers including the ability to configure and customise these features. This is similar in nature to the aim of remedy Option A2 for Apple which aims to achieve equivalence of access to WebKit and Safari but does not prescribe the means by which Apple (or Google in the case of Option A4) would achieve this. As Google already makes most APIs public, there is no equivalent to remedy Option A3.*
>
> ...

> *Based on the evidence to date, it may be sufficient that Google enables access to the WebAPK minting functionality, which is essential for implementing PWAs, for third-party browsers to address the issue set out in 'WP3 - Access to browser functionalities within the iOS and Android mobile ecosystems' paper."*
>
> Browsers and Cloud MIR - WP7

We support this remedy.

To our knowledge WebAPK minting is the only functionality on Android devices that are not available to other browser vendors (excluding Samsung). This prevents these third-party browser vendors from competing in the provision of Web App functionality, performance, stability, security and privacy.

WebAPKs remaining exclusive to Chrome is anti-competitive, restricts browser competition and damages the viability of Web Apps because:

1. Other browsers can not compete to provide better functionality for Web Apps
2. Provides an unfair advantage to Chrome on Android through increased engagement from Web Apps.
3. Damages adoption of Web Apps by making them not work properly in other browsers.

Google publicly stated 7 years ago that they were working on sharing this functionality with other browsers but there has been no progress. We believe that they should be forced to share this functionality with other browsers. Google should be able to set strictly necessary, proportionate and justified security conditions attached to obtaining access to this API.

It is essential that Google allows competing browsers to properly install Web Apps. The simple, timely solution to this is for Google to make the existing Play Services API available to third-party browsers, allowing them to use the same WebAPK minting service that Chrome enjoys access to.

Google is already obligated under the Digital Markets Act to make this functionality available to third-party browsers. As such there is no additional cost for them to make such a change within the UK.

While we are pleased that Google has indicated to the MIR that they are working on sharing this functionality, given that this has been their posture of 7 years with no progress, we recommend that the MIR either force them to share this functionality or obtain legally binding commitments with set time-frames to do so.

Importantly, given this service is provided by Google Play Services, Google should be able to provide this on almost all existing Android devices (Play Services currently goes back to Android 5.0 - 2014, so includes around 99.6% of users). We believe it is reasonable that Google complete this change within the next 6 months.

4.3. Issue 3 – Apple preventing all rivals from offering remote tab IABs on iOS

> *"B1 - A requirement for Apple to enable remote tab IABs for WebKit-based browsers. B2 - A requirement for Apple to enable remote tab IABs for browsers wishing to use alternative browser engines."*
>
> Browsers and Cloud MIR - WP7

> *"Since remote tab IABs are not currently offered to any third-party browsers on iOS, Apple submitted that iOS []. Apple has also submitted that [].*
>
> *Remote tab IABs are available on Android devices and we consider should be technically feasible on iOS, but we have limited evidence to suggest the types of mitigations that can be put in place to address any security concerns in extending this functionality."*
>
> Browsers and Cloud MIR - WP7

We support both of these remedies.

We believe that SFSafariViewController should be upgraded such that it invokes the default browser (regardless of whether that browser uses WebKit or another engine) rather than, as it currently does, ignore the user's current choice of default browser.

Android Custom Tabs on Android already operates in this fashion and is a mature and stable technology.

4.4. Issue 4 - Alternative WebViews for IABs on iOS

> *"Issue 4 – Apple preventing rival browser engines from offering nonWebKit based webview IABs, including bundled engine IABs to app developers on iOS"*
>
> Browsers and Cloud MIR - WP7

> *"B3 - A requirement for Apple to allow alternative webviews to Apple's iOS WKWebView."*
>
> Browsers and Cloud MIR - WP7

We do not believe this remedy is necessary, and are concerned it could encourage behavior that is against consumers interests. In particular as we highlight in our paper, some in-app browsers from ByteDance and Meta (that do not use the default browser) have been caught injecting effectively keyloggers and other extensive tracking into third party websites.

If a company or organization wishes to compete in the supply of in-app browsers then the ideal method from both a consumer and competition perspective is for them to build a dedicated browser app, and convince users to set it as their default browser. That browser should then be invoked when users do in-app browsing in any app. For example when users click on a link in a messaging app.

This aligns incentives correctly as users are gained not by controlling a popular social media app and silently replacing the users default browser but by creating an excellent browser and convincing users to use it by creating better features, performance, stability, security or privacy.

In order for this to function properly, the other remedies the CMA has proposed (in particular B4) need to be enacted.

4.5. Issue 5 - Alternative WebViews for IABs on Android

> *"Issue 5 – on Android, default settings and preinstallation of Android WebView make it difficult for app developers to use IABs based on alternative webviews"*
>
> Browsers and Cloud MIR - WP7

As with the previous section, we do not believe any remedy is necessary for the same reasons as on iOS.

4.6. Issue 6 – Apple’s and Google’s IAB policies

4.6.1. Remote-Tab IABs to use default browser

> *“B4 - A requirement for Apple and Google to implement remote tab IABs using the default browser."*
>
> [Browsers and Cloud MIR - WP7]

> *"Option B4 remedy would require that where the app developer utilises remote tab IAB, then the app would call on whichever browser a user has set as their default dedicated browser – although would still allow app developers to change which browser they use for in-app browsing if needed."*
>
> [Browsers and Cloud MIR - WP7]

We support this remedy.

Specifically we believe that SFSafariViewController should be updated to invoke the default browser. In the rare event that the default browser does not support this, it can fall back on a system default, this should be designed to work even if the system pre-installed browser has been uninstalled due to remedy C9.

Android Custom Tabs already supports this functionality.

We believe all non-browser apps on both iOS and Android should respect the users choice of default browser and should either be prohibited from overriding it or at a minimum be required to ask for explicit permission. We cover the permission example under remedy B6.

4.6.2. Disclosure of In-App Browser

> *“B5 - A requirement for Apple and Google to make users aware of being in an IAB by implementing changes to the interface or implement disclosures.”*
>
> [Browsers and Cloud MIR - WP7]

> *“Option B5 would aim to increase user awareness of the in-app browsing experience to facilitate more informed user choice when browsing in native apps.*
>
> *6.38 A potential remedy option would be to require an ‘information screen/ disclosure’ to be presented to a user when opening a link to third-party content in*

> *an IAB that would provide an alert about the presence of an in-app browsing feature (for example, this may read 'This browsing experience is provided within the app andis not your default browser')."*
>
> Browsers and Cloud MIR - WP7

We support this remedy, specifically in the form of an operating system permission prompt.

Importantly, we do not believe it is necessary to alert the user they are in an in-app browser if it is invoking the users default browser under the hood. Where in-app browsers are problematic is where they have silently usurped the users chosen default browser and created additional steps and friction to open the link in the users default browser.

We believe that the CMA should either outright ban this practice or as a softer approach require the non-browser apps to gain explicit permission from the user to not use their default browser as the in-app browser.

Importantly these measures should apply to all large companies that currently override the users choice of default browser including Apple, Google, Meta and ByteDance.

We describe such a permission in the next remedy.

4.6.3. Opt-out Setting for In-App Browsing.

> *"B6 - A requirement for Apple and Google to implement opt-out settings for in-app browsing."*
>
> Browsers and Cloud MIR - WP7

We support this remedy.

One important distinction is that this should be specifically targeted at in-app browsers that override the users choice of default browser. That is this setting should only disable in-app browsers that **do not** use the user's default browser.

In section 5.5 of our paper "Digital Markets Act - Interventions - In-App Browsers" we proposed the following more detailed remedy:

Users should be allowed to opt-out of using the apps own in-app browser in two ways.

First is a per-app permission for using the app's own in-app browser to manage http/https links to non-cooperating third-party websites.

A possible message could be:

> *Allow "Instagram" to use its own in-app browser instead of your default browser "Firefox":*
> *Allow Once*
> *Allow*
> *Don't Allow*

Second, a global user opt-out blocking apps from asking for this permission to stop every new app bothering the user if they have already decided they wish to always use their default browser for http/https links to non-cooperating third-party websites.

A possible settings page could be:

> ***Allow Apps to request to use its own in-app browser [Toggle Button]***
> *Allow Apps to request to use their own in-app browser instead of your default browser.*
>
> *When this is off, all new requests by apps to use their own in-app browser instead of your default browser will be denied.*
>
> *Apps that have asked for permission to use their own in-app browser will appear here.*

This per-app opt-out paired with a global opt-out has a precedent on iOS with the app tracking transparency settings that iOS introduced in iOS 14.0 on September 16, 2020.

Operating system gatekeepers would then need to enforce respecting these settings via app store rules and to abide by it for their own non-browser apps.

4.7. Issue 7 - Choice Architecture in Factory Settings

4.7.1. Pre-installed Third-Party Browsers

> *"C1 - A requirement for Apple and Google to ensure that multiple browsers are pre-installed, using defined criteria."*
>
> Browsers and Cloud MIR - WP7

> *"Option C1 would require Apple on iOS and Google on Android to pre-install a certain number of browsers in factory settings based on pre-determined criteria, though we note that the* ***criteria would need to be carefully considered.****"*
>
> Browsers and Cloud MIR - WP7
> (emphasis added)

This is a fascinating idea and we support this remedy.

We assume this would likely take the form of browsers or a browser folder appearing on the first homescreen. The key advantage to this idea is that it will make ambient choice clear and available to end users.

While this will be straightforward for Apple to implement (single OS under their sole control, typically higher end phones and tablets), there may be challenges to this on Android which is dominated by OEMs. We ask that the CMA take a practical approach that may potentially differ between high-end phones (with high bandwidth and space) and low end phones.

Carefully designed with proportionality in mind, this remedy could help push browser competition on both iOS and Android.

The CMA should also consider whether this will apply to existing devices and how this interacts with devices sync and restores on new devices.

4.7.2. Browser Choice Screens at Device Set-Up.

> *"C2 - A requirement for Apple and Google to ensure the use of browser choice screens at device set-up."*
>
> Browsers and Cloud MIR - WP7

We support this remedy.

Given that both Apple and Google have already implemented choice screens for the EU, we ask that the CMA consider them as starting points for their own choice screens. Many of the flaws in their initial design have been ironed out and recently Apple have implemented a number of our recommendations.

Importantly the CMA is not constrained by the proscriptive language of the Digital Markets Act and can make some key improvements.

1. Should be shown on device-setup and should be unskippable.

2. The gatekeeper's browser, if not selected, should be uninstalled from the device or virtually uninstalled, i.e. hidden from the user, if the gatekeeper can demonstrate that is not practical because of technical limitations. We cover this idea in more detail in Section 4.3 of our OWA - Mobile Browsers and Cloud Gaming - Response to Working Papers 1-6.

4.7.3. Grant Default Browser the Hotseat at Device Setup

“C3 - A requirement for Apple and Google to ensure the placement of a default browser selected by the user in the ‘dock’ / ‘hot seat’ or on the default home screen at device set-up.”

Browsers and Cloud MIR - WP7

We support this remedy.

Currently being selected as the default browser does not grant the hotseat. This means it is entirely possible, and in fact likely that many existing users who have set a third party browser as their default browser will still have Safari or Chrome in the hotseat.

While users can change this setting manually by dragging the item out and another in, Apple and Google have added significant friction by not making this automatic.

"Only about half (52%) of people understand that their default browser is opened when they, for example, click on a link in an email or document.
[..]
over half (53%) also erroneously believed that their default browser would automatically be pinned to their task-bar."

Mozilla - “Can browser choice screens be effective?” paper

We believe that setting the default browser should automatically place it in the hotseat if the factory default is still in the hotseat. That is this remedy should be strengthened to not only apply at device setup but should also apply for the non-device setup choice screen and when a user installs and sets a new browser as default.

4.7.4. Respect Default Browser Choice

“C4 - A requirement for Apple and Google to ensure that a user’s choice of default browser is always followed across all browser access points.”

Browsers and Cloud MIR - WP7

> *"Option C4 relates to the default setting for browsers. This would aim to ensure that the user default is respected as such across all other access points*
> …
> *Our preliminary view is that Option C4 would ensure that effectiveness of users choice of default browser (via 'choice screen' in relations to Options C2 and C5) is not restricted only to a direct use of browser. This remedy could also incorporate Option B4 (see paragraph 6.31 above) that mandates that remote tab IABs use default browser."*
>
> Browsers and Cloud MIR - WP7

We wholeheartedly agree with this remedy. User's choice of default browser only matters if it is respected by the operating system and apps on the operating system. The default browser is the user's chosen agent to view the wider web, it should be the default in all situations where the user is viewing third-party content on the Web.

Third-party content in this case means any website that has not explicitly consented to be shown within the non-browser app. In our paper "OWA - DMA Interventions - In-App Browsers" we proposed a mechanism whereby websites could opt-out of being shown in non-default in-app browsers and only displayed in the users default browser. This mechanism could help make the dividing line between 1st, 2nd and 3rd party content clear.

4.8. Issue 8 - Choice Architecture after Device Set-Up

4.8.1. Recurring Browser Choice Screens

> *"C5 - A requirement for Apple and Google to ensure the use of browser choice screen(s) after device set-up."*
>
> Browsers and Cloud MIR - WP7

> *"7.34 This remedy can be viewed as similar to Option C2 in respect to the design and framing, with the key difference being that this remedy would be shown to existing smartphone owners, as opposed to Option C2, which concerns a choice screen at set-up for new devices. In this case, a choice screen of browser options would prompt users to select their default browser.*
>
> *7.35 As with Option C2, the timing, frequency and the design of a choice screen (eg one-off choice screen prompt for existing users only to avoid duplication with Option C2) would need to be considered carefully; together with whether this could be implemented as a part of an operating system update."*

Browsers and Cloud MIR - WP7

We support this remedy for both iOS and Android.

Apple has recently agreed to show a browser choice screen for all devices each major software update (presumably meaning once per year). We have yet to see a demo of Apple's latest choice screen but it appears that many of the issues we identified with their previous choice screen have been fixed.

4.8.2. Easier to Change Default Browser

> *"C6 - A requirement for Apple and Google to make adaptations to the user journey for changing their default browser."*
>
> Browsers and Cloud MIR - WP7

> *"Option C6 seeks to address frictions in the user journey for changing default browser settings by mandating that Apple and Google reduce the number of steps to switch default browser on their devices. This option would seek to ensure that the journey for users who, unprompted, want to change their default browser, is not unnecessarily burdensome.*
> ...
> *This remedy may consider either a single centralised location in the settings for changing default browser, regardless of what browser is currently set as default and/or that the user journey for changing default browser should be identical regardless of which browser is set as default."*
>
> Browsers and Cloud MIR - WP7

We support this remedy.

Changing defaults should be available in a centralized and prominent location in the settings. There should be nothing about the design that indicates that the operating system's apps are preferred or preferenced.

Searching for "default", "default browser" or "browser" should link to this page for both the general search and the settings search.

Making it difficult to change the default only advantages large players who have via contracts or control of the operating system pre-installed and set their browser as default.

This centralized default location (which is standard on most operating systems) would also have the advantage of alerting users that they can change defaults. For example when changing the default email client, a user may notice they can change the default browser.

4.8.3. Sharing if set as Default Browser

> *"C7 - A requirement for Apple and Google to share user data on default browsers settings with browser vendors."*
>
> Browsers and Cloud MIR - WP7

We support this remedy.

This is important to allow browser vendors the ability to monitor if competition solutions are working and to design sensible prompts to users (i.e. don't bother users who have already set the browser as the default).

Allowing browsers to indicate to users that they can be set as the default and making it easy for users to do so is an important part of browser competition. While we share the CMA's concerns about excessive prompting we believe this, if necessary, could be covered by the next remedy.

Browsers typically do not want to annoy their users which might explain why this has not been a problem on other operating systems.

4.8.4. Limit on Default Browser Prompts

> *"Option C8 seeks to ensure that third-party browser vendors use the same volume and frequency of prompts as Safari or Chrome currently do, suggesting that users change their default browser."*
>
> Browsers and Cloud MIR - WP7

> *"Limiting the frequency of prompts would seek to level the playing field for other providers, ensuring that neither Apple nor Google can leverage their control of their respective operating systems to self-preference in relation to prompts and notifications regarding default browser status. Users might benefit from the limit on the use of such prompts as it would remove the potentially 'nagging' nature of prompts that users are exposed to and ameliorate the fatigue that an overload of prompts and notifications can produce. However, using prompts is an important tool for third-party vendors as it is one of the main mechanisms through which they can obtain a foothold in the market."*
>
> Browsers and Cloud MIR - WP7

One important point here is that Safari typically does not need to prompt users to set them as the default as they are the default in factory settings. To our knowledge Safari never asks to be set as the default. As such we suggest changing the language to make the limit a reasonable number of requests.

Browsers vendors that are in a significant position of power by control of operating systems or via complex placement/revenue sharing deals should not have access to a greater quality or number of prompts to switch default browsers. They should not be able to place these prompts in parts of the operating system that are not equivalently available to third-party browser vendors.

We also agree that prompts to set a newly installed browser are an important way for smaller browser vendors to gain traction on a platform but that browsers should not be able excessively nag users to set them as the default.

Our proposal would be a one-click system prompt that the browser could invoke displaying for example:

> *"Would you like to set Vivaldi as your default browser?*
> *YES | NO ".*

If the browser is already the default then no prompt is shown. The operating system could limit the frequency with which the browser could ask to a reasonable number, perhaps 3 times, then once per a year thereafter.

As a side-point we believe that setting the browser as the default should replace the pre-installed factory settings operating systems browser in the hotseat.

4.8.5. Make Safari and Chrome Uninstallable

> *"C9 - A requirement for Apple and Google to allow users to uninstall Safari browser app on iOS and Chrome on Android devices."*

Browsers and Cloud MIR - WP7

We support this remedy.

> *"The inability to uninstall Safari and Chrome further limits user control and choice over the customisation of their device, and could appear to create an implicit endorsement and self-preference Safari on iOS and Chrome on Android in comparison to other browsers."*

Browsers and Cloud MIR - WP5

We agree with the assessment that making particular browsers impossible to uninstall signals to users that these are the preferred browsers for the operating system.

The WKWebView, SFSafariViewController, Android WebView and Android Custom Tabs should be treated as system components, and it should not be possible to uninstall them. These components are used in a wide variety of native apps. Many native apps use the webviews as a convenient way to render first/second party content and any remedy which would break these apps would be unreasonable, disproportionate and counter-productive. Other apps use SFSafariViewController or Android Custom Tabs when the user clicks on an external http/https link and should be allowed to continue to do so, provided Apple commits to making SFSafariViewController respect the user's choice of default browser.

We would also support it not being possible to uninstall the default browser until a new default browser has been selected. An appropriate and neutral error message should be displayed if the user attempts to do so.

As such both Safari on iOS and Chrome on Android should be uninstallable.

4.9. Issue 9 and 10 - App Store Cloud Gaming Remedies

> *"D1 - A requirement for Apple to review and amend its Guidelines to remove the specific restriction identified as restrictive and a prohibition on Apple introducing new restrictions with equivalent effect.*
>
> *4.9.2. D2 - A requirement for Apple to enable cloud gaming native apps to operate on a 'read-only' basis (i.e. with no ingame purchases or subscriptions) so that games do not need to be re-coded and no commission would therefore be payable to Apple).*
>
> *4.10.1. D3 - A requirement for Apple and Google to allow CGSPs to incorporate their own or third party in-app payment systems for in-game transactions."*

While we are generally supportive of moves to open up competition these remedies are outside the scope of our organization's mandate.

5. Toward A Brighter Future

OWA believes that the Web's unmatched track record of safely providing frictionless access to information and services has demonstrated that it can enable a more vibrant digital ecosystem. The web's open, interoperable, standards-based nature creates an inclusive environment that fosters competition, delivering the benefits of technology to users more effectively and reliably than any closed ecosystem.

OWA's goal is to ensure that browser competition is carried out under fair terms, that user choice in browsers matters, and that web applications are provided equal access and rights necessary to safely contest the market for digital services.

The MIR team has a critical opportunity to fix key issues that have undermined both browser and Web App competition for over a decade to the benefit of both UK consumers and UK businesses. This will improve interoperability, contestability, and fairness leading to lower priced and higher quality apps — not only for the UK but for the entire world.

OWA believes competition, not walled gardens, leads to the brightest future for consumers, businesses, and the digital ecosystem.

6. Open Web Advocacy

Open Web Advocacy is a not-for-profit organization made up of a loose group of software engineers from all over the world, who work for many different companies and have come together to fight for the future of the open web by providing regulators, legislators and policy makers the intricate technical details that they need to understand the major anti-competitive issues in our industry and potential ways to solve them.

It should be noted that all the authors and reviewers of this document are software engineers and not economists, lawyers or regulatory experts. The aim is to explain the current situation, outline the specific problems, how this affects consumers and suggest potential regulatory remedies.

This is a grassroots effort by software engineers as individuals and not on behalf of their employers or any of the browser vendors.

We are available to regulators, legislators and policy makers for presentations/Q&A and we can provide expert technical analysis on topics in this area.

For those who would like to help or join us in fighting for a free and open future for the web, please contact us at:

Email	contactus@open-web-advocacy.org
Web / Web	https://open-web-advocacy.org
Mastodon	@owa@mastodon.social
Twitter / X	@OpenWebAdvocacy
LinkedIn	https://www.linkedin.com/company/open-web-advocacy