Standardizing client-side API for Web payments?

Author: Stéphane Boyera (boyera@w3.org), W3C[1]

Introduction

Payment is an essential element of trade and commerce, and the explosion of e-commerce in the last two decades on the Web was made possible thanks to the remote payment facilities offered by network such as Visa, MasterCard or American Express. Beyond this “traditional” method of payments, numerous new payment systems have emerged over time. Pure on-line players such as Paypal have been joined by more recent innovations, such as Dwolla, or new crypto-currencies such as Bitcoin. This phenomenon has developed further and at higher speed with the rise of the mobile platform all over the world, with e.g. options such as direct carrier billing or app-store payments. Now even the millions of underprivileged people in rural regions of the developing world, who were excluded from the traditional banking systems, have access to electronic payment systems on their mobile phone (mobile money).

Unfortunately, the lack of interoperability between all these systems is creating major troubles for all the actors of the ecosystem. Merchants have to support a growing set of payment alternatives, and each new option requires specific development. Users have to manage multiple accounts on multiple payment systems, and ensure that each account has the required funds, etc. New innovative payment system providers (PSP) have to convince users and merchants to integrate their new solution, making it harder to see their solution adopted, and preventing innovation from small players. How can we change this situation and reach a point where merchants could author their payment pages once for all current and future payment systems, users could use whatever payments options they have access to, and new payment system providers emerge and be quickly usable on all ecommerce sites?

One key objective of this workshop is to identify specific area(s) for standardization and agreements between all players in the ecosystem that could ease payments for merchants, ease payments for customers, and enable new payment system providers to emerge and innovate.

This paper has the objective to identify potential areas for standardization in the domain of ecommerce and payments on the Web. It does not take into account use-cases related to in-store e-payments and related topics.

Challenges and Opportunities

The current situation and the evolution regarding payment systems on the Web are far from being optimal. Indeed, as more systems are blossoming in different regions of the world, the market is getting more and more fragmented. It is now almost impossible for a given merchant to offer all payments solutions. This situation leads now to an increasing number of missed opportunities for actors of the ecosystem.

[1] This work at W3C is supported by the European Union through the Seventh Framework Programme (FP7/2013-2015) under grant agreement n°611327 - HTML5 Apps

From the merchants' perspective, it is impossible to support all payment solutions, and to have an account and a specific payment page for 20 or 50 payment solutions. The result of this situation is that lots of business opportunities are missed due to people not being able to pay. Such a situation is particularly visible in developing regions, where people do not have access to traditional banks and credit cards, but do have mobile money subscriptions. As of today, they are not able to use this solution to buy anything on the Web, outside the very few local companies that implemented this option.

Related to this issue, merchants are in most cases offering credit card payment as the de facto option available to all. Apart from all the security issues related to current credit card payments, such methods, given the transaction fees, are inappropriate for micro-payments, leading again to missed opportunities for selling small items such as a press article.

From the users' perspective, the issues are very similar to the ones mentioned for merchants: users sometimes cannot buy what they want due to non-support by the merchant of the payment solution they have access to. They cannot also buy very small articles at very low cost. The second major issue is security. The current model of payments requires a very high level of trust from the user to the merchant. In most cases, you give to the merchant your own credit card information, and trust that, first, the merchant is honest, and, second, he is storing your details in a secure way. The recent stories in the news[2] are highlighting the growing risk with such approaches, and the inability for users to know whether a given merchant is at risk or not.

From the PSP perspective, the current situation is also problematic, because innovations face a very high adoption barrier. Numbers of innovations are now showing-up, like direct carrier billing, loyalty programs, coupon-based payments, crypto-currencies, mobile money etc., but the uptake of these payment solutions is very slow due to the effort required from the user and merchant perspective to add these payment solutions and trust them.

For the Web as a whole, the lack of unified payment mechanism is also a weakness compared to mobile native applications that are already offering an integrated approach in application stores. Working towards a unified approach is critical to ensure that HTML5 offers a competitive alternative to native apps.

It is therefore clear that it is time to solve the challenges, and to have an ecosystem approach where all actors work together to create a level playing field that will be beneficial to all. Given the importance of mobile platforms, the emergence of developing regions and their specificities with regards to e.g. mobile money, the emergence of dozens of new payment solutions all over the world, now is the time to work towards an overall model and a set of standards that will ease transactions on the Web, and leverage innovation by PSP.

[2] See e.g. Target security leak (http://explorernews.com/news/article_aac9ef22-68f4-11e3-a3f3-001a4bcf887a.html) or South Korea credit card data stealing (http://www.economist.com/news/finance-and-economics/21595059-enormous-data-heist-may-dim-koreans-love-affair-credit-cards-card-sharps)

Vision

Our long-term vision is a world where:

- A web developer will be able to develop a generic payment page template for any merchant:
    - Being agnostic on the set of payment solutions the merchant that will use this template will support
    - Being agnostic on the set of payment solutions the customers of the merchant will use to pay
- A merchant could use a generic payment template and add a set of payment solutions he supports, being agnostic on the set of payment solutions his customers will use. The merchant should be able to add and remove payment solutions very easily without changing their site(s).
- A user could install a set of payment solutions on his device/browser, and pay with them. A user should be able to install and remove payment solutions at any time.

In terms of process, a schematic application flow would look like this:

- When a user is ready to pay a merchant, he should be presented a set of payment options that is the merchant-compatible subset of all his installed payment solutions. Each payment solution may have its own individual extra-cost.
- When the selection is made by the user, he should then "be transferred"[3] to the PSP. The PSP will receive a set of information from the merchant and the user. The exact details of the information provided by the merchant to the PSP might be largely PSP-specific, but at least the protocol for exchanging information as well as the generic description of the request (at least item description, price, currency) needs to be standardized. The authentication and all related matters should be managed between the PSP and the user, and does not need to be standardized. It could be part of a separate independent standardization initiative[4].
- When the interaction between the PSP and the user is completed, the PSP returns results to both the merchants (merchants' server) and the user's UA.

[3] « Transferred » in this context can have different meaning depending on the payment solution. E.g. a native app is launched, an iframe opens an URI at the payment system provider site, a secure frame is open, etc.

[4] It is essential to manage authentication separately. Today each and every PSP has its own requirements on authentication and identities, uses different mechanism such as second factor authentication or mobile SIMcard authentication etc. While authentication and identities are critical topics for the Web in general, and have a larger scope than just in the case of payments, it is critical to ensure that standardization in the payment domain is not bound to finalization and acceptance of standards in authentication and identities that may takes long time.

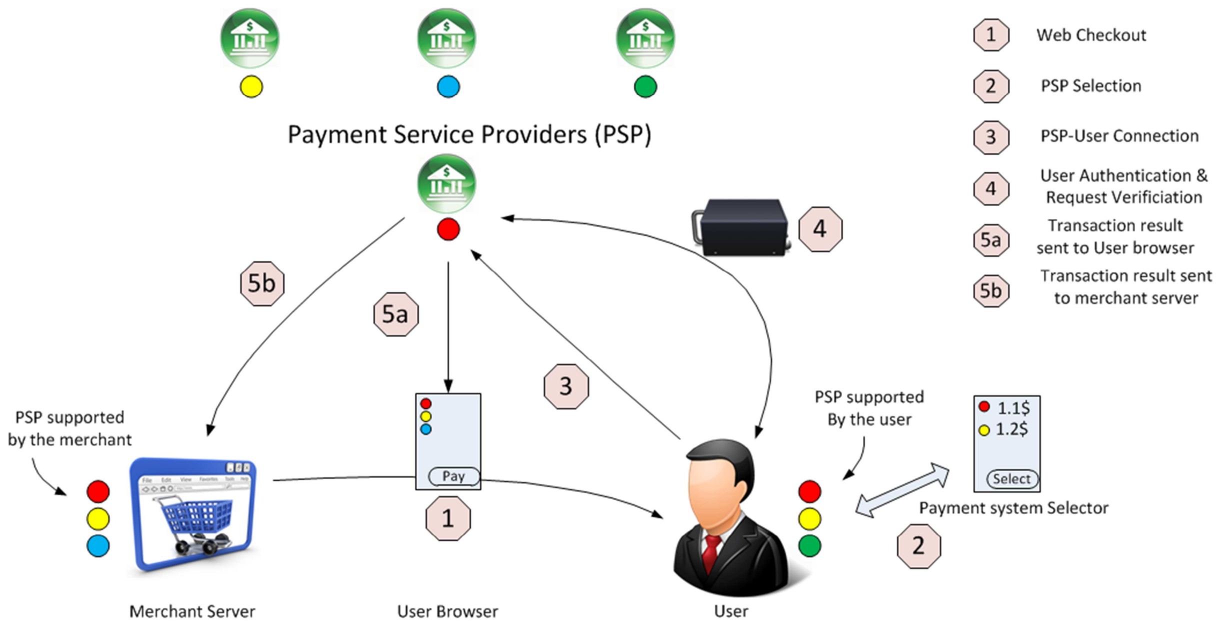
Payment Service Providers (PSP)
1 Web Checkout
2 PSP Selection
3 PSP-User Connection
4 User Authentication & Request Verificiation
5a Transaction result sent to User browser
5b Transaction result sent to merchant server
5b
5a
3
4
PSP supported by the merchant
PSP supported By the user
Pay
1.1$
1.2$
Select
Payment system Selector
1
2
Merchant Server
User Browser
User

Figure 1: Basic payment Architecture

The figure 1 represents the most basic version of the architecture. It can clearly be extended with e.g. (cloud-based) intermediaries that will help the selection. It is also possible in the future to consider more advanced PSP selection mechanism that is not based on 1-to1 matching.

It is important to note that this model does not mention, on purpose, issues related to cross-border trade challenges, tax and local regulations and all related topics. This is currently addressed (or not) by merchant sites before reaching the final payment stage. There is surely a need to address these issues in a global way, but this can be discussed separately, in the same way as authentication and identities can be decoupled from the application flow. It is essential to separate the space of standardization challenges in small independent elements to ensure quick progress.

The case currency conversion falls in the same approach. In a simplistic way, it is possible to consider that currencies conversion between what the merchant requests and what the user has access to could be a function of the PSP. It is the easiest model. A more complex one would be to have multiple currencies offered by the merchants through his PSP and have a selection/matching engine that is able to identify the possible options. This topic, while important given the international nature of the Web, should be considered as a future work item.

How far are we from the vision?

The model described in the previous section is not very new and already used by many PSP. For instance, all the in-app payment solutions we studied (e.g. google wallet, the Mozilla mozPay API, iOS) are using a model that is similar. Other PSP are also moving towards the same direction, e.g. PayPal and its recent release of a restful API that could be used client-side. It is interesting to note that among all PSP, Mozilla mozPay is the first and only one offering a way to manage multiple PSP. The steps for PSP selection are included in mozPay, while it is not in all other cases.

Looking into more details, each solution differs at different places:

1. Token formats and content: what the different fields in the token, how it is signed, how the request is described etc. Here to make quick progress, it would be critical to explore a

standardized format for the token globally and the description of the request, but allowing each PSP to add all the fields they want. E.g. illustrated in figure 1 stage 3, the request info should surely be commonly defined between PSP, but the merchant info (including application information) while standardized in the envelop of the info, may be specific to each PSP.

2. Client-side API: today almost all APIs have the same functionalities, but using different names for functions. It should be relatively easy to reach agreement on the client-side API.
3. Installation of Payment solutions by users: this probably one of the points that need more work. Today, as only mozPay has started to explore a possible way for this feature, it would be important to find more solutions that would fit all PSP and accepted/implemented by browser vendors.
4. PSP matching between merchants and users and user selection mechanism. Like for the previous point, no real work has happen yet on this feature. Protocols for negotiations are always tricky, and in this case, it is easy to add complexity such as e.g. ensuring that user PSP have enough credit etc. In order to make quick progress, it would be important perhaps to adopt very basic policy such as 1-to-1 matching (as e.g. illustrated in figure 1) as a first option, before exploring further this item later.
5. Client-side PSP behavior: One of the key elements in the overall framework is how the UA transfers the user from the merchant site to the PSP for authentication and validation. As of today, two cases are usually implemented: some systems are launching a native app on the device and some systems are redirecting users to the PSP web site using an iframe. How to describe a PSP within the UA and the behavior of this PSP is related to the point 3. To make quick progress, a first solution could be to limit options to a few set of alternatives that would be agreeable by most players. However, how the authentication is completed should let to each PSP (as illustrated with a black box in figure 1).
6. Protocol and Messaging between PSP and merchant server, and PSP and UA after completion of the payment transaction between the user and the PSP. All PSP today have a way to notify the results of the transaction to the merchant server and the UA (in the case of in-app, client side PSP). Having a standardized messages and protocol would be critical to ensure interoperability between PSP.

Given commonalities that exists with different systems, we believe that focusing a standardization activity on the points mentioned above, with basic functionalities first particularly for point 3 (e.g. starting with the user agent as the host for PSP installation) and 4 (like e.g. 1 to 1 matching) should lead to important results on relatively short term, making a significant step forward in the domain. Obviously, on longer term, it would be essential to enhance the overall architecture, and integrate e.g. a more distributed model, where the installation of payment solutions by the users and the merchants could be down at an intermediary level in the cloud.

Conclusion

The overall ecosystem of payments on the Web is very complex. It involves lots of building blocks (protocols, messages, APIs, authentication, identity, etc.), and this complexity is a major barrier for interoperability. This lack of interoperability and the growing number of PSP is an important threat for the Web as a global market place. It is therefore essential to work, as soon as possible, towards a set of standards that will be beneficial to all actors, and enable even further innovation in the domain.

However, given the complexity of the architecture, standardizing and building consensus on each and every block will be a long process. As a first step, it is critical to identify key bottlenecks, and quick wins that could impact the domain in the very near future.

As of today, one area looks quite promising: client-side API. A standardized client-side API that would support multiple PSP, a standardized messaging protocol and a minimal agreement on token description would be a major step towards global interoperability. Such model will have multiple benefits:

- Each PSP can decide what a safe authentication is for them (SSL, mobile native apps, SIMcard auth., etc.).
- Each user can decide the PSP he trusts, and ensure that all his personal information is only shared between him and his PSP
- Each merchant can support very easily a growing number of PSP at very low-cost, addressing the needs of all potential customers around the world.

Given the approach taken by many PSP today, an initiative around these items could deliver results and impact in a very near future.