Collaborative web extensions: a P2P approach

Rodolfo Gonzalez[1], Sergio Firmenich[1,2] , Alejandro Fernandez[1], Gustavo Rossi[1,2]

1. LIFIA, CIC, Facultad de Informática, Universidad Nacional de La Plata
2. CONICET, Argentina
{rgonzalez, sfirmenich, casco, gustavo}@lifia.info.unlp.edu.ar

Abstract. Web extensions are powerful software artifacts that allow end-users to adapt and enrich a website. These extensions run on the user's web browser as a single-user software that manipulates available third-party web contents. Many of them offer some collaborative features that depend on a web application. The need of two co-depending software artifacts (the web application as back-end and the web extensions as front-end) increases complexity, making the system harder to develop and maintain. In this paper we tackle this problem by proposing a P2P approach to build collaborative web extensions. The approach involves a middleware and a framework. On the one hand, the middleware serves to manage the resources offered by the browser so multiple P2P extensions can coexist. It ensures that the overall performance of the browser is not degraded by the collaborative web extension. On the other hand, the proposed framework is intended to allow developers without experience in P2P to create collaborative web extensions on top of the middleware. This paper discusses the main challenges of building P2P web extensions, presents the approach, and two case studies focused on the use of the framework for inexperienced developers.

Keywords: web extensions, collaboration, peer to peer.

1 Introduction

Web extensions [1], [2] are a popular mechanism to adapt third-party websites. Commonly, a web extension is made of a combination of Javascript, HTML, CSS, and configuration files. Web extensions can change the browser behavior, introduce changes to visited websites and also to provide new web pages delivered with the extension once installed in the web browser. Some well-known techniques such as mash-ups [3], [4] and web augmentation [5] rest on web extensions. Their purpose may be quite broad, such as integrating web contents [6], supporting repetitive users tasks [7], [8], improving accessibility [9], [10], and recommendation-based personalization [11] among other possibilities. Besides attracting the interest of the research community as shown by the previous examples, web-extensions are popular among web users that desire to adapt the web. The extensions repositories for Mozilla Firefox and Google Chrome show the popularity of these technologies, where thousands of extensions are available, many of them with thousands of users. Clearly, web extensions are nowadays the de facto standard to customize the web browser and consequently augment the user's experience with the web.

Web extensions can communicate with external services (via HTTP requests) which makes it possible to develop web extensions with collaborative features. A web service available on a server enables the communication among the same web extension installed in the browser of different users. When that is the case and regardless of the complexity of the functionality they offer, such networked extensions are designed in a client-server architecture. This means that they depend on a server-side component/service, which increases the technical skills required to create the web extension. In addition, depending on a server component complicates maintenance and increases the costs associated with deployment for production.

We argue that building web extensions in a P2P style opens new opportunities to augment the web, especially when collaboration is involved, by removing the need for a server component. Following the P2P philosophy, web extensions are designed to allow users to collaborate by sharing computation power, storage and networking capabilities of their browsers, and by explicitly solving tasks for one another. In this paper, we discuss the challenges that building P2P extensions presents and outline our proposed approach based on: i) a middleware that manages the resources offered by the browser so multiple P2P extensions can coexist, without degrading the browser's performance, ii) a framework for allowing developers experience in to create collaborative extensions that do not require a client/server architecture.

In this paper we are particularly interested in demonstrating that our approach enables developers without experience in P2P technologies to create P2P extensions with some collaborative features.

The paper is structured as follows. In Section 2 we present the technical aspects of our approach, including both the middleware and the framework. Sections 3 and 4 introduce two studies we have carried out to investigate if our framework is, in fact, useful for inexperienced developers. Section 5 presents the related works, and finally in Section 6 we give some conclusions and present future works.

2. The approach in a nutshell

To simplify development and to reduce development and execution errors, we separate different concerns into two supporting artifacts. First, a middleware that manages all P2P extensions installed in the browser, handling message exchange, and monitoring workload. Second, a framework that abstracts the key domain objects (such as message and peer), provides a clear interface to send messages and hides interaction with the middleware. Following this approach, developers do not require any other technical skill than those required to write any other web extension: JavaScript, HTML, and CSS; and they follow the same kind of deployment process, i.e. simply install the extension in a Web browser (no server deployment is required).

Figure 1 shows the overall approach. A web extension implements the P2P middleware and exposes the P2P API to any other web extension installed on the same browser. Other web extensions can directly execute functions in the middleware API via the message passing mechanisms [2]. However, the recommended way to interact with the middleware is via the P2P framework. Developers include the framework as a dependency of their extensions (it is a JavaScript library). As it may be appreciated, all the messages pass through the middleware and are later routed to the corresponding web extension. For a concrete web extension, a message is a JSON object with the features the developers desire. When the message passes through the middleware (using the behavior provided by our framework), it is encapsulated with the information required in the middleware layer (the type of message, timestamp, the extension's metadata among others) - see Figure 1, on the right. The P2P extension that delivers the middleware comes with a minimalist user interface, which allows the user to have control of the messages.

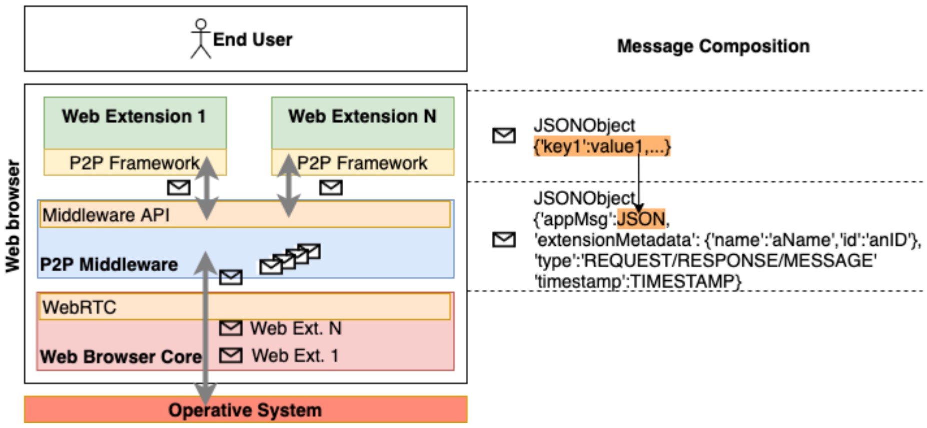


Fig. 1. The approach in a nutshell

Communication between peers is currently based on WebRTC. WebRTC brings real-time video and audio communication to the browser and can be used to transport other forms of content. It robustly solves the technical challenges in P2P communication. To establish a direct connection between peers in WebRTC, a discovery and negotiation method called signaling is used. It involves both parties connecting to a commonly agreed upon service to decide the mechanisms they will use to connect (as they may be located behind firewalls, in NAT'd networks, etc.). The signaling process can be implemented with any technology compatible with WebSocket/XHR. WebRTC depends on a commonly known signaling server that introduces a unique point of failure and turns the architecture into a hybrid P2P. However, it must be noted that our approach still removes the need for writing and deploying a specific server for each web extension. We consider this to be a good trade-off while we explore other alternatives.

2.1 Framework

The framework is packaged as a JavaScript library. It can be included in web extension projects that may be executed in desktop and mobile Web browsers. Once included in the web extension project, the user must create a class that represents the application (for instance, P2PNewsVisualization), and make this class inherit from the extension point offered by the Framework, which is called AbstractP2PExtension. This extension point lets developers specify the behavior of their web extensions considering two communication modes: (a) to send a message to another peer without expecting a response, (b) to send a request message for which a response is expected and must be managed by the peer that made the request when it arrives. The following list presents the main aspects to be considered for using the AbstractP2PExtension extension point. The developer must instantiate the concrete class and send to the new instance the connect() message which it inherits from the extension point. What connect() message does is:

- to send the initialize() message to the new instance. This is a method that developers must implement to set instance variables related to the extension's metadata (name and id) to uniquely identify the extension.
- to initialize the P2P communication mechanism for the web extension.
- to register the extension in the middleware.

Developers may use other inherited behaviors to look for peers, and to send messages/requests to other peers:

- getPeers(callback): obtains the peers currently connected. Since this method is asynchronous, a callback function must be passed as a parameter.
- sendMessage(msg, peer): sends a message (first parameter) to a specific peer (second parameter).
- broadcast(msg): send a message to all the peers available.
- sendRequest(msg, peer): send a request message (first parameter) to a specific peer (second parameter). It is expected to receive a response.
- sendResponse(msg, peer): send a response using a message (first parameter), and to a specific peer (third parameter). In this case, the msg (a JSON object) should be populated with further information about the original request.

To handle messages and requests according to its needs, the extension must implement some of the following methods (or all of them):

- receiveMessage(msg, peer): this method will be executed when a new message is sent to the extension. It is not expected to deliver a response. It receives the message as the first parameter and the peer who sent it as the second parameter.
- processRequest(msg, peer): this method will be executed when the extension receives a request. It is not expected to create and deliver a response during the method execution. This method is suitable for human (interactive) collaboration. Its response depends on the user's interaction which occurs asynchronously.
- automaticProcessing(msg, peer): this method will be executed when the extension receives a request and this request was marked as automatic (it is just a flag in the message). This method must return a JSON object intended to be used as a response, and the framework automatically delivers it when the method finishes. This method is specially designed for computing collaboration, that can be automated, i.e. without depending on user intervention.
- If the extension sends requests, it must implement the processResponse(msg, peer) method to manage the responses to the requests previously done.

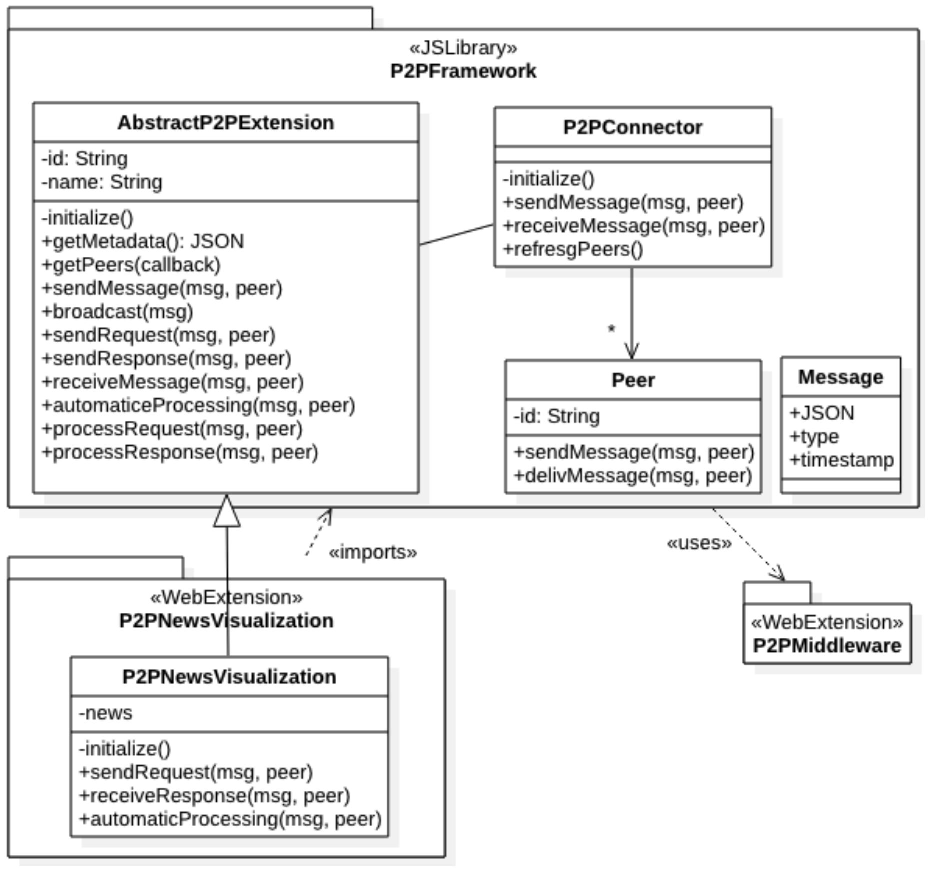


Fig 2. The framework and its extension point, the class AbstractP2PExtension

Figure 2 shows a simplified version of our framework plus another class showing how to inherit from the extension point, named AbstractP2PExtension. The P2PConnector class is the one that uses the middleware API. Two other classes provide simple abstractions for the peer and the message. The Message class is managed by the AbstractP2PExtension and the P2PConnector objects, meanwhile, the concrete class representing the web extension (P2PNewsVisualization) always works with the JSON Object defined by the developer.

3. Case studies

Two case studies were conducted to demonstrate the potential of P2P web extensions, to assess to what extent the proposed middleware and framework succeed in abstracting the complexity of P2P communication, and to identify strengths and weaknesses of the approach and its current implementation. The first case study focuses on building P2P extensions from scratch using our approach. It does so by asking several developers to build the same P2P extension while keeping a diary of activities. The second case study attempts to isolate the tasks that are directly related to the P2P networking functionality of the web extension. It does so by asking one developer to transform an existing single user web-extension (developed by himself) into a P2P web-extension. Following we describe both case studies.

3.1. Case study 1: Collaborative information access

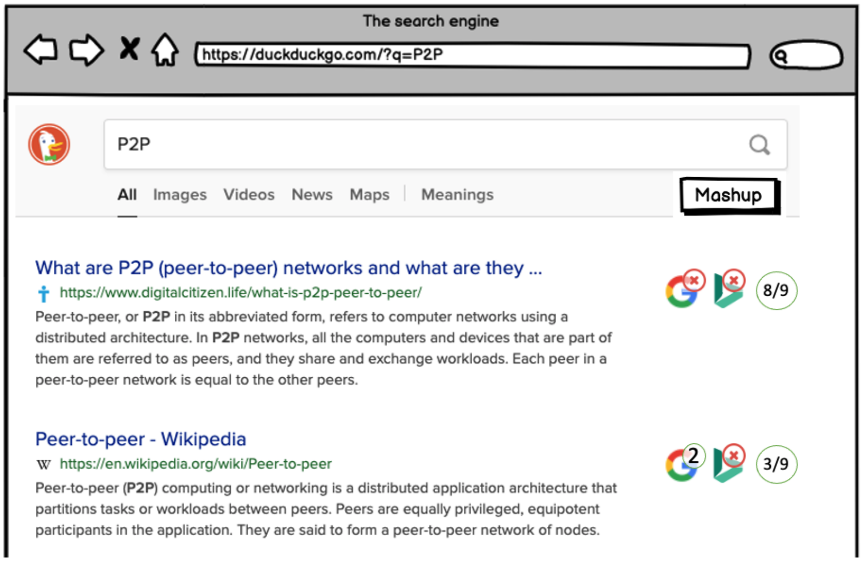
The first case study is focused on information retrieving, one of the most common tasks carried out by users when navigating the web. In this case we circumscribe the problem to web searching under the influence of the filter bubble [12]. Although the filter bubble problem has been addressed mostly in the context of social media, for the sake of simplicity, we will show a case study focused on main web search engines. However, it could be generalized to specialized search engines offered by social networks, e-commerce platforms, etc.

The main idea in this case study is to design a web extension that augments web search result pages with the results given for the same search but for other users, in a form of collaborative search. For instance, when a user searches something at Google, the resulting page will be augmented with the results for the same search but retrieved for all the users that have the same browser extension installed. In this way, the results will not be constrained by the user's profile information that the search engine considered when executing the search, reducing the filter bubble effect.

In order to be able to identify if our P2P middleware is understandable from the point of view of developers, we recruited several computer science students without experience in the creation of web extensions. The design directrices we gave to them were:

- To use our P2P middleware to provide an automatic collaboration. This collaboration implied asking peers to perform the same search (same search engine and same string search), extract the results, and send those results in JSON format to the user that performed the search in the first place.
- To augment three web search pages (including Google, Bing and DuckDuckGo), which includes:
    - To augment each search result to show how many peers got that result.
    - To provide a results mash-up including the ten most frequent results among all peers considering an average position, and thus to make visible the weight a result has for other users.

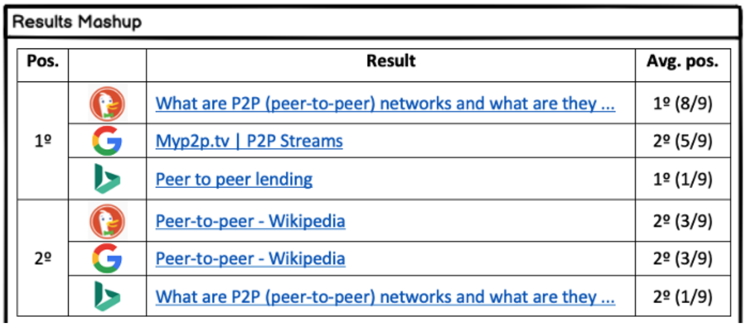
As part of the task definition, we gave to participants mockups of the desired augmenter behavior (mockups were similar to those shown in Figure 3 and 4). Each search result must be augmented with an icon indicating the position of the result in the other search engines. In Figure 3 (a search in DuckDuckGo for the term P2P) the first result is not present in the results provided by Google or Bing, whereas the second result is present in Google (in position 2), and not present in Bing. An additional widget should indicate for each result, how many peers obtained it. In Figure 3, the first result was obtained by 8 out of 9 connected peers. The web extension should also add a push button to the search engine's toolbar. The button opens the "Results mashup" shown in figure 4.


The search engine
https://duckduckgo.com/?q=P2P
P2P
All Images Videos News Maps | Meanings
Mashup
8/9
2
3/9

What are P2P (peer-to-peer) networks and what are they ...
https://www.digitalcitizen.life/what-is-p2p-peer-to-peer/
Peer-to-peer, or **P2P** in its abbreviated form, refers to computer networks using a distributed architecture. In **P2P** networks, all the computers and devices that are part of them are referred to as peers, and they share and exchange workloads. Each peer in a peer-to-peer network is equal to the other peers.

Peer-to-peer - Wikipedia
W https://en.wikipedia.org/wiki/Peer-to-peer
Peer-to-peer (**P2P**) computing or networking is a distributed application architecture that partitions tasks or workloads between peers. Peers are equally privileged, equipotent participants in the application. They are said to form a peer-to-peer network of nodes.

Fig 3. Augmentation mockup for DuckDuckGo search page



Results Mashup

Pos.		Result	Avg. pos.
1º		What are P2P (peer-to-peer) networks and what are they ...	1º (8/9)
		Myp2p.tv | P2P Streams	2º (5/9)
		Peer to peer lending	1º (1/9)
2º		Peer-to-peer - Wikipedia	2º (3/9)
		Peer-to-peer - Wikipedia	2º (3/9)
		What are P2P (peer-to-peer) networks and what are they ...	2º (1/9)

Fig 4. Results mashup

During the experience, participants completed an activity diary entry for each work session, considering the time used for the session, the consulted resources, difficulties and achievements. They could use any documentation available online to solve technical difficulties about web extensions and JavaScript. Documentation about our middleware and framework was also available as technical manuals. The activity diary was materialized in a Google Form that participants have completed for each work session.

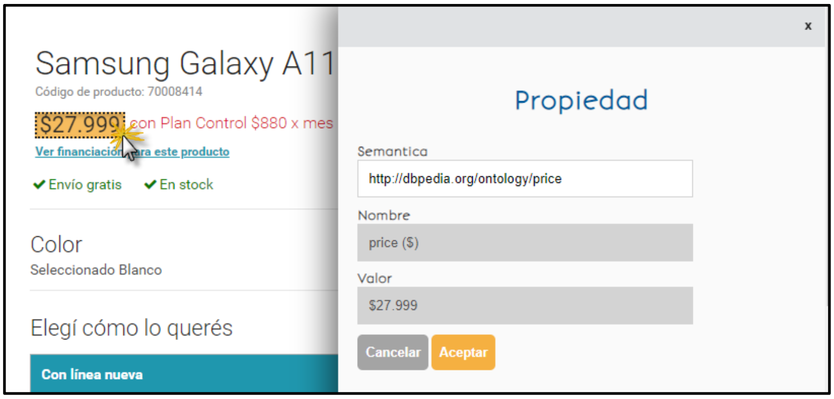
Ten participants have been able to use our middleware and framework to create the proposed P2P Web extension. They coudll use their preferred development environment. According to the activity diaries, most of the developers consulted StackOverflow for low level Javascript doubts and the Mozilla developer website for those aspects related particularly with Web extensions. All participants were able to read and understand our own documentation about the technologies presented in this paper. The most commonly reported difficulties had to do with the availability of the signaling service required by our P2P architecture, which was involuntary shut down in several opportunities.

As a conclusion, developers without experience on the creation of web extensions were able to create a collaborative one through the use of our approach.

3.2. Case study 2: Semantic extraction and information object retrieving

Data collection is an important part of learning, research, and decision making that frequently takes place on the web. There are web extensions that simplify data collection in the form of web-scraps, such as Evernote [13], or that help users to visually create web scraping templates to obtain structured content from the web, such as AnyPicker [14] and the WOA platform for data collection [15]. These tools normally rely on the existence of a centralized server. This second case study aims to explore the applicability of our P2P approach to build collaborative web extensions in the domain of structured data collection. Moreover, the goal of this case study is also to assess the difficulty of including P2P communication behaviour in a web extension using the facilities offered by our approach.

To conduct this study we recruited one junior web developer with no previous experience in building web-extensions or P2P applications. The study was organized in two phases. In the first phase, the developer had to create a web-extension similar to those offered by WOA and AnyPicker. Users of the extension can create extraction templates (or recipes) that map elements in the DOM tree of a webpage to properties of data items. Figure 5, shows the property definition dialog (on the right) where the user is defining the price property, and a web-page on the left where the user clicks on the DOM element that contains the price of the product. Templates, which are designed to work on all web-pages matching a certain URL pattern, can later be used to extract information items from those web-pages. Templates and extracted items are stored in the browser's local storage. The extension provides functionality to inspect the list of extracted items (i.e, the items repository). The items repository allows the user to filter items only matching a given URL pattern, or matching a given type of item (e.g., a phone, a car, a hotel, etc.). Items types and properties are expressed in terms of Schema.org classes and properties.


Samsung Galaxy A11
Código de producto: 70008414
$27.999 con Plan Control $880 x mes
Ver financiación para este producto
✓ Envío gratis ✓ En stock
Color
Seleccionado Blanco
Elegí cómo lo querés
Con línea nueva

Propiedad
Semantica
http://dbpedia.org/ontology/price
Nombre
price ($)
Valor
$27.999

Fig 5. Using the template editor to capture the price property of a product

In the second phase, the developer received basic training on the use of the middleware and framework presented in sections 2. He was asked to use the middleware and framework to modify the web-extension so it could work as a node in a P2P network. He was free to decide how nodes in the network would interact with each other. The developer's design decisions mainly impacted the way users extract items and inspect the list of extracted items (the repository of items).

Whenever the user attempts to extract items from a web-page, the web extension connects to all available peers and asks for matching templates (which could focus on different parts of the DOM tree to extract items with varying properties). Templates are presented to the user in a carousel, that indicates what items (with what properties) each template would extract. The user can choose to extract several items that are saved in the user's local storage. If the user extracts an item using a template obtained from a peer, the template is saved locally so it remains available if the peer disconnects. The repository of items was redesigned to include items available in the network. When the user opens the repository, a request is made to all available pairs for items matching the type and URL pattern filters. Moreover, an additional filter has been included to choose whether items from peers should be retrieved or not.

Peers can join and leave the network at any time, which means that the available templates and items vary in time. The absence of a central server results in a lower maintenance cost for the network which, in the worst case scenario of only one user, can work as a single-user web-extension. New peers can join the network to offer functionality that is different from template definition and item extraction. A new type of peer can, for example, offer functionality to compare items according to multiple criteria, or to crawl the web automatically extracting items along the way. The current design of the web-extension presents a serious limitation as all pairs in the network (regardless of their location, affiliation or interests) share templates and items. This design decision makes it unsuitable for realistic usage scenarios. However, the underlying framework allows for web-extension developers to define richer peer selection strategies.

After finishing the second phase, the developer was interviewed to learn about his experience using the middleware and framework to transform the single-user web extension into P2P. As positive points, the developers indicated that the framework succeeds in abtracting the technicalities of P2P communication, and provides flexible and comprehensive extension points. As the main criticisms, the developer reported that it is difficult to verify (understand) connection status of peers which complicated debugging applications as coding errors could not be told apart from connection errors. Moreover, he reported that although documentation is precise and complete, more varied examples are needed.

3.3 Discussion

The two case studies presented in this section demonstrate that our approach allows developers without experience on P2P technologies to create collaborative web extensions. In the first case study ten developers, i.e. the 100% of the participants, could develop a web extension for collaborative search. In the second case study, more qualitative data was acquired, which shows that migrating an existing web extension to support P2P communication is also possible without important constraints. We are aware that the kind of collaboration used in both experiences is simple, but demonstrates that it is easy to use our framework and middleware.

In the context of this work we use the terms collaboration and collaboration support to refer to a range of situations and supporting tools, following the early definitions provided by Bair [16]. A web extension that offers information, coordination, collaboration and/or cooperation support matches the software dimension in Lenz and Lenz [17] definition of groupware (i.e. intentional group processes plus software to support them). The case studies in this section cover only a small fraction of the universe of collaboration supporting tools. However, they demonstrate the key features of the proposed framework and middleware which are not limited to a particular form of interaction among users, nor to a particular domain (e.g., learning, work, leisure, etc.).

4. Related works

Although it is clear that underlying concepts about P2P are not new, since some years ago there is a trend in their use in new domains. Probably, one of the most well-known new applications of P2P is Blockchain, a technology that makes it possible to decentralize information in a secure way. In this context, applications (these are called Dapps, for Decentralized Applications) based on these blocks may be created, enabling its use for smart contracts and reaching a broad range of domains such as IoT, manufacturing systems, health systems, etc. [18]. There are mature technologies behind Blockchain, such as Hyperledger and Ethereum. In the context of the Web, Blockchain also may have an impact on its decentralization, giving more control to users about their own information, an aspect that is also being tackled by other approaches such as SOLID [19], as we explain below. However, none of these technologies have an impact on the use of the Web in itself, i.e. in the way users interact with contents and information.

In this paper we propose a novel use of P2P in the context of web navigation, creating a new technological enabler for making web extensions collaborative without requiring a centralized server. In this regard, and to the best of our knowledge, there are two well-known applications of P2P in web browsers. First, there are approaches to support collaborative computing. For instance, Pando [20] offers a platform in which a user must install a server and run it in his own machine. Then, other users may access this back-end application with their browsers to offer it for computing . On the other hand, it has been proposed to use the browser as a distributed platform for content delivery [21], [22]. In this line, Tindall [23] studies the use of a communication protocol that improves how to program over WebRTC. Jannes et al. [24] propose a generic distributed application server which is also currently supported by existing web browsers such as Beaker Browser [25]. Other approaches use P2P communication for specific aims, such as improving virtual environments [26]. Although these works show that decentralizing the Web is a current topic, these are far to be applicable to web extensions with the final goal of improving the overall user's web experience.

Server-side support for web extensions was already studied and analyzed [27], in which authors propose a Model-Driven Web Augmentation approach to model back-end requirements. Although the complexity for developing, deploying and maintaining the back-end component is clearly better than using an ad-hoc approach. We believe that a P2P approach based exactly on the same technology required for programming web extensions is a more suitable and convenient way, at the same time that it removes any need of a centralized server application.

Decentralizing the web is the main goal of the SOLID project [19], led by Tim Berners-Lee. With SOLID, application data is stored using RDF and Semantic Web technologies in personal online datastores (PODs) that are controlled by the user. PODs are web accessible data stores that the user can deploy on personal servers, or can obtain as a service from a company. The user can control, with various levels of granularity, which applications have access to which parts of the PODs. SOLID puts the user back in control of application data. The work presented in this paper shares the motivations of SOLID, and even borrows the idea of using Semantic Web vocabularies to model data (as in case study 2). However, the main difference with the SOLID approach is that our framework and middleware use the browser's local storage to store data. Moreover, each web extension developer is still responsible for the data that the extension generates, stores and shares. In a way, our approach is still not able to completely break the data silos created by the co-dependency of applications and the data they use.

5. Conclusions and Future works

External Web structures (i.e. "defining hypermedia structures externally of the involved documents" [28]) are software artifacts that improve the overall Web experience. Web extensions are the most common and convenient way to develop and deploy this kind of software. Without an intermediate server, a web extension cannot communicate to the same web extension installed in another user's browser. Even more, when some communication between different web browsers is required, new technical barriers appear (for instance, dealing with back-end technologies beyond HTML, CSS, and JavaScript). Server-side support has been very important for different reasons [27].

This paper presented an approach to build P2P web extensions, which aims to eliminate the need for a centralized server to communicate web browsers and users. A signaling back-end service has been designed and implemented. It may connect peers for any web extension or for a specific one without requiring changes on it, neither on the P2P web extensions source code because it was conceived as a generic single-purpose (to connect peers) platform.

In this paper we focused on the experience of developers while using the middleware and framework. The main idea was to study if developers without experience on P2P and on web extensions were able to create a collaborative extension without involving themselves in low level aspects of peer communication. In this regard we can say that inexperienced developers could achieve the proposed development task just by consulting available documentation of web extension development and of our approach.

Although we believe that our approach improves the potential of web extensions without requiring a centralized application, we still need to create and evaluate more scenarios. For instance, pervasive and distributed storage should be supported by the framework. However, we already could apply our approach in several scenarios. Besides future evaluations and experiments in this regard, it is also mandatory to study the power of a P2P web browser, as well as how to continuously measure and limit this kind of collaboration in order to not degrade the overall Web experience.

References

1. P. Mehta, *Creating Google Chrome Extensions*, 1st ed. edition. Place of publication not identified: Apress, 2016.
2. ‘Browser Extensions. Draft Community Group Report 28 January 2020’, Jan. 2020. Accessed: Mar. 17, 2021. [Online]. Available: https://browserext.github.io/browserext/
3. F. Daniel and M. Matera, *Mashups: Concepts, Models and Architectures*, 2014th edition. New York: Springer, 2014.
4. J. Wong and J. I. Hong, ‘Making mashups with marmite: towards end-user programming for the web’, in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, New York, NY, USA, Apr. 2007, pp. 1435–1444. doi: 10.1145/1240624.1240842.
5. O. Díaz and C. Arellano, ‘The Augmented Web: Rationales, Opportunities, and Challenges on Browser-Side Transcoding’, *ACM Trans. Web*, vol. 9, no. 2, p. 8:1-8:30, May 2015, doi: 10.1145/2735633.
6. O. Díaz and C. Arellano, ‘Sticklet: An End-User Client-Side Augmentation-Based Mashup Tool’, in *Web Engineering*, Berlin, Heidelberg, 2012, pp. 465–468. doi: 10.1007/978-3-642-31753-8_45.
7. O. Díaz, J. De Sosa, and S. Trujillo, ‘Activity fragmentation in the web: empowering users to support their own webflows’, in *Proceedings of the 24th ACM Conference on Hypertext and Social Media*, New York, NY, USA, May 2013, pp. 69–78. doi: 10.1145/2481492.2481500.
8. S. Firmenich, G. Rossi, M. Winckler, and P. Palanque, ‘An approach for supporting distributed user interface orchestration over the Web’, *International Journal of Human-Computer Studies*, vol. 72, no. 1, pp. 53–76, Jan. 2014, doi: 10.1016/j.ijhcs.2013.08.014.
9. J. P. Bigham and R. E. Ladner, ‘Accessmonkey: a collaborative scripting framework for web

users and developers', in *Proceedings of the 2007 international cross-disciplinary conference on Web accessibility (W4A)*, New York, NY, USA, May 2007, pp. 25–34. doi: 10.1145/1243441.1243452.

10. C. González-Mora, I. Garrigós, S. Casteleyn, and S. Firmenich, 'A Web Augmentation Framework for Accessibility Based on Voice Interaction', in *Web Engineering*, Cham, 2020, pp. 547–550. doi: 10.1007/978-3-030-50578-3_42.
11. M. Wischenbart, S. Firmenich, G. Rossi, G. Bosetti, and E. Kapsammer, 'Engaging end-user driven recommender systems: personalization through web augmentation', *Multimed Tools Appl*, vol. 80, no. 5, pp. 6785–6809, Feb. 2021, doi: 10.1007/s11042-020-09803-8.
12. E. Pariser, *The Filter Bubble: How the New Personalized Web Is Changing What We Read and How We Think*, Reprint edition. Penguin Books, 2012.
13. *Evernote*. Accessed: Mar. 22, 2021. [Online]. Available: https://evernote.com/
14. *AnyPicker*. Accessed: Mar. 22, 2021. [Online]. Available: https://anypicker.ryang-studio.com/
15. G. Bosetti, S. Firmenich, G. Rossi, M. Winckler, and T. Barbieri, 'Web Objects Ambient: An Integrated Platform Supporting New Kinds of Personal Web Experiences', in *Web Engineering*, vol. 9671, A. Bozzon, P. Cudre-Maroux, and C. Pautasso, Eds. Cham: Springer International Publishing, 2016, pp. 563–566. doi: 10.1007/978-3-319-38791-8_49.
16. J. H. Bair, 'Supporting cooperative work with computers: addressing meeting mania', in *Digest of Papers. COMPCON Spring 89. Thirty-Fourth IEEE Computer Society International Conference: Intellectual Leverage*, Feb. 1989, pp. 208–217. doi: 10.1109/CMPCON.1989.301929.
17. Peter Johnson-Lenz and Trudy Johnson-Lenz, 'Post-Mechanistic Groupware Primitives: Rhythms, Boundaries, and Containers', *The International Journal of Man Machine Studies*, vol. 34, pp. 395–417, 1991.
18. A. Bahga and V. Madisetti, *Blockchain Applications: A Hands-On Approach*, 1st edition. VPT, 2017.
19. E. Mansour *et al.*, 'A Demonstration of the Solid Platform for Social Web Applications', in *Proceedings of the 25th International Conference Companion on World Wide Web - WWW '16 Companion*, Montr&#233;al, Qu&#233;bec, Canada, 2016, pp. 223–226. doi: 10.1145/2872518.2890529.
20. E. Lavoie, L. Hendren, F. Desprez, and M. Correia, 'Pando: Personal Volunteer Computing in Browsers', in *Proceedings of the 20th International Middleware Conference*, New York, NY, USA, Dec. 2019, pp. 96–109. doi: 10.1145/3361525.3361539.
21. A. Kobusinska, A. Wolski, J. Brzezinski, and M. Ge, 'P2P Web Browser Middleware to Enhance Service Oriented Computing — Analysis and Evaluation', in *2017 IEEE 10th Conference on Service-Oriented Computing and Applications (SOCA)*, Kanazawa, Nov. 2017, pp. 58–65. doi: 10.1109/SOCA.2017.16.
22. C. Vogt, M. J. Werner, and T. C. Schmidt, 'Leveraging WebRTC for P2P content distribution in web browsers', in *2013 21st IEEE International Conference on Network Protocols (ICNP)*, Oct. 2013, pp. 1–2. doi: 10.1109/ICNP.2013.6733637.
23. N. Tindall and A. Harwood, 'Peer-to-peer between browsers: cyclon protocol over WebRTC', in *2015 IEEE International Conference on Peer-to-Peer Computing (P2P)*, Sep. 2015, pp. 1–5. doi: 10.1109/P2P.2015.7328517.
24. K. Jannes, B. Lagaisse, and W. Joosen, 'The Web Browser as Distributed Application Server: Towards Decentralized Web Applications in the Edge', in *Proceedings of the 2nd International Workshop on Edge Systems, Analytics and Networking - EdgeSys '19*, Dresden, Germany, 2019, pp. 7–11. doi: 10.1145/3301418.3313938.
25. *Baker Browser*. Accessed: Mar. 22, 2021. [Online]. Available: https://beakerbrowser.com/
26. T. Koskela, J. Vatjus-Anttila, and T. Dahl, 'Communication Architecture for a P2P-Enhanced Virtual Environment Client in a Web Browser', in *2014 6th International Conference on New Technologies, Mobility and Security (NTMS)*, Mar. 2014, pp. 1–5. doi:

10.1109/NTMS.2014.6814011.

27. M. Urbieta, S. Firmenich, G. Bosetti, P. Maglione, G. Rossi, and M. A. Olivero, 'MDWA: a model-driven Web augmentation approach—coping with client- and server-side support', *Softw Syst Model*, vol. 19, no. 6, pp. 1541–1566, Nov. 2020, doi: 10.1007/s10270-020-00779-5.
28. N. O. Bouvin, 'From NoteCards to Notebooks: There and Back Again', in *Proceedings of the 30th ACM Conference on Hypertext and Social Media*, New York, NY, USA, Sep. 2019, pp. 19–28. doi: 10.1145/3342220.3343666.