## ТЕХНИЧЕСКИЕ НАУКИ

***Скачихин Алексей Анатольевич***
инженер-программист, бакалавр
ООО "Техартгруп", Белорусский государственный университет
информатики и радиоэлектроники
г. Минск, Республика Беларусь

# АНАЛИЗ МЕТОДОВ РАСШИРЕНИЯ ФУНКЦИОНАЛЬНОСТИ WEB-БРАУЗЕРОВ ДЛЯ ДОСТУПА К API ОПЕРАЦИОННОЙ СИСТЕМЫ

***Аннотация****: в статье кратко рассматриваются методы и средства расширения функциональности Web-браузеров, выявляются и анализируются достоинства и недостатки того или иного метода; дается краткий анализ существующих технологий для реализации плагинов. Как результат, представлено наиболее эффективное, по мнению автора, решение для расширения функциональности Web-браузера, которое позволяет получить доступ к API ОС.*

***Ключевые слова****: расширение Web-браузера, Web-стандарты, ОС, NPAPI, PPAPI, JavaScript.*

В наше время World Wide Web является одной из самых популярных сред распространения цифрового контента [1]. Web-браузер становится универсальным инструментом получения, распространения и взаимодействия с контентом. Поведение современных Web-браузеров описывается набором стандартов, разрабатываемых в рамках организации World Wide Web Consortium. Чёткое следование стандартам в определённой степени гарантирует переносимость исходных кодов и прозрачность процесса развития Web-технологий. В свою очередь сам процесс стандартизации занимает продолжительное время и, как результат, общие темпы развития программных и аппаратных средств значительно превосходят темпы развития Web-браузеров. В связи с этим актуальной становится задача расширения функциональности Web-браузера возможностями, которые находятся вне плоскости существующих стандартов.

Расширения Web-браузеров, или так называемые «технологии на стыке», впервые получили свою популярность в конце 90-х годов. В то время технология Macromedia Flash, которая по своей сути является расширением Web-браузера, предоставляла кроссбраузерный способ воспроизведения цифрового контента (аудио, видео) [2]. Аналогичный стандарт, HTML5, получил своё начало лишь в 2010 году, и до сих пор его реализации имеют существенные ограничения по сравнению с Macromedia Flash. Ещё одним примером успешного применения расширений Web-браузера является расширение Google Voice, которое позволяется участвовать в онлайн конференциях с использованием аудио- и видеоустройств. Аналогичный по функциональности стандарт, WebRTC 1.0, на данный момент находится в разработке (Working Draft), что делает Google Voice более приемлемым вариантом с точки зрения корректности и качества при применении его для разработки программных средств [3].

Можно выделить два типа расширений для Web-браузера:

1. Плагин (Plugin) – независимо разрабатываемый и компилируемый модуль, расширяющий возможности Web-браузера по отображению и взаимодействию с контентом определённого типа. Плагины, как правило, исполняются в контексте Web-браузера и имеют доступ как к возможностям самого Web-браузера, так и к возможностям операционной системы. Чаще всего для разработки плагинов применяются такие языки программирования, как C и C++. Разработка и сопровождение плагина является сложной задачей, т.к. ошибки в коде программы могут привести к полной остановке работы Web-браузера.

2. Дополнения (Extension) – независимо разрабатываемый модуль, который вносит изменения в пользовательский интерфейс Web-браузера. Дополнения значительно легче разрабатывать, чем плагины, т.к. дополнения не могут напрямую взаимодействовать с операционной системой, а лишь посредством абстрактного, очень ограниченного кроссплатформенного интерфейса, представленного на высокоуровневом языке программирования, обычно – JavaScript. Ошибки в коде программы не приводят к полной остановке работы Web-браузера, а высокоуровневый язык программирования JavaScript лучше подходит для

задач управления пользовательским интерфейсом и задач алгоритмизации, где время разработки является более приоритетным фактором по сравнению с производительностью.

Таким образом, можно сказать, что дополнения являются более предпочтительным выбором, если доступ к возможностям операционной системы не требуется. Дополнения являются более безопасными расширениями Web-браузера с точки зрения пользователя и простыми в разработке с точки зрения разработчика. Тем не менее, именно необходимость получить доступ к операционной системе является главной причиной разработки расширений. Рассмотрим, какое место занимают плагины в типовой упрощённой архитектуре Web-браузера (рис. 1).

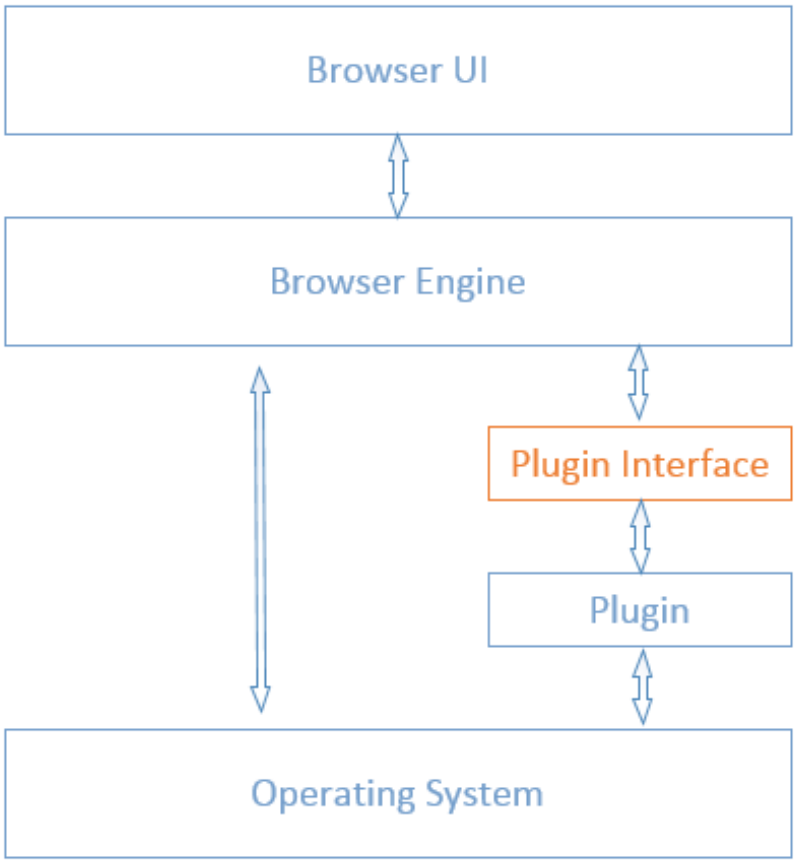


Рис. 1

Плагины взаимодействуют с операционной системой посредством системных вызовов, ссылаясь на динамические библиотеки ОС статически (на момент компиляции) или динамически (в момент исполнения) [4]. В свою очередь с

Web-браузером плагины взаимодействуют через специальный интерфейс. Важно отметить, что интерфейс между плагином и Web-браузером должен быть стандартизирован, иначе реализация плагина будет непереносима между Web-браузерами.

Для того, чтобы уменьшить степень влияния самого плагина на Web-браузер, современные реализации используют более сложную схему, где каждая вкладка браузера и плагин исполняются в отдельном процессе, а взаимодействие между ними происходит посредством IPC (сокр. от. англ. Inter-process Communication – «Межпроцессное взаимодействие») [4]. С точки зрения программиста такая реализация означает, что предположения о процессе или потоке исполнения Web-браузера являются заведомо некорректными. На момент публикации браузеры Google Chrome, Mozilla Firefox и Internet Explorer уже реализовали схему «одна вкладка – один процесс».

Рассмотрим стандартизированные интерфейсы, применяемые при программировании плагинов.

NPAPI (сокр. от. англ.  Netscape Plugin Application Programming Interface – «Программный интерфейс подключаемых модулей Netscape») – первый кроссплатформенный интерфейс плагинов, поддерживаемый всеми популярными браузерами (Mozilla Firefox, Google Chrome, Apple Safari) за исключением Internet Explorer [5]. Этот интерфейс лежит в основе двух наиболее значимых расширений Web-браузера на момент публикации: Adobe Flash, Microsoft Silverlight. Плагины на NPAPI чаще всего программируются на C, C++.

Достоинства интерфейса:

1. Неограниченный доступ к возможностям ОС.

2. Устоявшийся стандарт.

3. Простой процесс установки плагина в систему.

Недостатки интерфейса:

1. В силу неограниченности доступа к ОС представляет потенциальную угрозу безопасности пользователя. По статистике, плагины, реализованные на NPAPI, являются основным вектором атак на Web-браузеры.

2. Из-за проблем с безопасностью, Google анонсировал, что в 2015 году NPAPI интерфейс не будет поддерживаться в Google Chrome. Распространённые плагины, такие как Adobe Flash, Microsoft Silverlight, сохранят свою работоспособность на неопределённое время.

PPAPI (сокр. от англ. Pepper Plugin API) – интерфейс, разработанный компанией Google, как более безопасная альтернатива NPAPI. Принципиальное отличие PPAPI от NPAPI заключается в том, что взаимодействие плагина с ОС происходит не напрямую, а через PPAPI, таким образом, все вызовы, выполняемые из плагина, проходят через стандартизированный интерфейс [6]. Для того, чтобы разработчики не могли обойти ограничения PPAPI, плагины должны компилироваться с помощью специальных C/C++ компиляторов, предоставляемых PPAPI SDK. Плагины, компилируемые под PPAPI, могут быть представлены в двух форматах:

1. NaCL (сокр. от англ. Native Client) – модуль, содержащий платформенно-независимый байт-код, который транслируется браузером в момент исполнения.

2. PNaCL (сокр. от англ. Portable Native Client) – пакет модулей, содержащий платформенно-зависимый машинный код. Выбор конкретного модуля из пакета происходит в момент исполнения.

Интерфейс PPAAPI даёт неявный доступ к следующим API ОС [6]:

1. Операции файлового ввода/вывода.

2. OpenGL ES 2.0.

3. Audio/Video API.

4. Управление потоками ОС.

Достоинства интерфейса:

1. Высокий уровень безопасности.

2. Простота использования – PNaCL модули не требуют установки и распространяются как самостоятельный тип контента.

Недостатки:

1. Проприетарность – интерфейс принадлежит компании Google и основан на деталях специфичных для Web-браузера Google Chrome. По этой причине интерфейс не поддерживается в других Web-браузерах.

2. Доступ к API ОС ограничен операциями, изначально заложенными в PPAPI.

Рассмотрев методы расширения Web-браузера, автор пришёл к следующему выводу: наиболее эффективным методом расширения Web-браузера с целью получения доступа к API ОС является метод разработки плагина на основе NPAPI, т.к. этот метод единственный на данный момент позволяет получить доступ к API ОС вне зависимости от используемого Web-браузера. Альтернативный метод, плагин на основе PPAPI, является хоть и более безопасным, но крайне ограниченным, т.к. применим только в рамках браузера Google Chrome, что противоречит идеи переносимости Web-приложений. Метод, основанный на разработке дополнения к Web-браузеру, не позволяет осуществлять доступ к API ОС и предоставлять эти результаты Web-приложению. Результаты сравнительного анализа также позволяют утверждать, что стандартизация API ОС комитетом W3C в большинстве случаев является более предпочтительным решением относительно любого другого метода в силу переносимости, безопасности и прозрачности реализации. Стоит так же отметить, что дальнейшим продолжением исследования может служить анализ методов, основанных на взаимодействии Web-приложения непосредственно с компонентами ОС посредством посылки сетевых HTTP или WebSocket запросов.

***Список литературы***

1. W3C Consortium // [Electronic resource] – Mode of access: http:// http://www.w3.org/Consortium/. – Date of access: 20.11.2014.

2. The Life, Death and Rebirth of Adobe Flash // [Electronic resource] / C. Warren – Mode of access: http://mashable.com/2012/11/19/history-of-flash/ – Date of access: 1.12.2014.

3. Google, Skype, and WebRTC // [Electronic resource] / D. Michels – Mode of access: http://www.nojitter.com/post/240160776/google-skype-and-webrtc – Date of access: 13.11.2014.

4. Grosskurth, A. A reference architecture for Web browsers / A. Grosskurth – IEEE, 2005. – c. 661-664.

5. Plugins – Mozilla // [Electronic resource] – Mode of access: https://developer.mozilla.org/en-US/Add-ons/Plugins – Date of access: 17.11.2014.

6. Native Client – Technical Overview // [Electronic resource] – Mode of access: https:// developer.chrome.com/native-client/overview – Date of access: 20.11.2014.