**Служба компоновки и создания фрагментов**

1. **Многофункциональность пользовательского интерфейса**

Сколько основных, готовых стандартных графических элементов или элементов управления интерфейса доступны для разработки? Как можно связывать данные и события с помощью этих элементов управления? Новые элементы управления должны быть удобны в использовании и иметь возможность простого подключения. Некоторые *RIA*-технологии предоставляют простые способы для добавления функциональности и информативности визуальному представлению, например, с помощью анимационных API на странице. Например, для обеспечения только одного нажатия пользователя на кнопке, можно добавить к кнопке анимацию так, чтобы она исчезла из представления.

1. **Сложность**

Разработчики используют существующие модели на основе страниц уже много лет благодаря их простоте и удобству, но эти технологии довольно грубы. *RIA*-технология должна быть простой для освоения, разработки и иметь возможности расширения. Она также должна взаимодействовать с существующими Web-технологиями.

1. **Гибкость и компонентное представление**

Важным фактором является гибкость для взаимодействия с различными межплатформенными компонентами. Взаимодействие должно обладать простотой компоновки и расширяемостью для создания новых пользовательских стандартных графических элементов интерфейса. После создания библиотек пользовательских стандартных графических элементов интерфейса их можно повторно использовать в приложениях.

1. **Обновление страницы**

Значительное преимущество заключается в возможности обновления блока всей страницы, так как обновление зависит непосредственно от сетевого трафика. Обновление блоков позволяет сделать приложение более быстрым и функциональным, и добавить больше визуализации для пользователей. Обновление блоков также упрощает обработку ошибок.

Предположим, пользователь выполняет действие или первую задачу на Web-странице, и данные в фоновом режиме пересылаются на сервер. Затем пользователь переходит к выполнения следующей задачи на этой же странице. Между тем приходит ответ на первую задачу и обновляет часть этой же страницы. Таким образом при разработке Web-страницы можно повысить эффективность работы и решения задач.

1. **Безопасность**

При выполнении настройки по RIAs убедитесь в отсутствии увеличения угрозы безопасности в сравнении с обычными приложениями. Проверьте безопасность окружающих сервер систем связи или модулей и расширений браузера, загружаемых с клиента.

1. **Поддержка основных Web-парадигм**

Технология должна поддерживать основные Web-парадигмы, существующие на текущий момент в Web-приложениях, например, *интернационализации*, независимости пользовательских устройств, независимости браузеров и поддержки передачи двоичных файлов для функций загрузки и выгрузки. Имеет значение также и зрелость технологии.

1. **Инструментарий**

Рассмотрим инструменты, доступные для разработчиков в виде интегрированных сред разработки (IDE) с функциями тестирования блоков и поддержкой отладки. Инструменты могут быть модулями, подключаемыми к существующим редакторам или поддерживаемые редакторами.

1. **Удобство использования**

Пользователи ожидают, что в браузерных приложениях будут работать функции обычных браузеров. В частности, в решениях на основе *Flash* не доступны такие функции, как сохранение изображений, использование клавиатурного сокращения Ctrl+F для поиска содержимого на странице и *функция* копирования-вставки. При разработке удобства использования *RIA* следует опираться на принципы взаимодействия человек-компьютер (*HCI*)..

Кастомизация корпоративных сайтов при помощи *RIA* может осуществляться посредством добавления интерактивных 3D-моделей, описывающих *ассортимент товаров*, видеороликов с демонстрацией услуг компании, а также различных *веб-форм* для управления сайтом на стороне пользователя (*авторизация*, управление профилем и прочее). Дополнительно в код самой страницы возможно включение специальных элементов, предоставляющих веб-браузерам информацию о наличии интерактивных сервисов, например, веб-фрагментов или провайдеров визуального поиска, транслирующих *RIA*, как это сделано, например, в IE8.



[увеличить изображение](https://www.intuit.ru/EDI/05_07_17_1/1499206907-9549/tutorial/635/objects/17/files/12_02.gif)

**Преимущества**

Несмотря на то что разработка веб-приложений для браузера имеет ограничения, более сложна и запутанна по сравнению с разработкой стандартных приложений, усилия обычно оправдываются, потому что:

* Не требуется установка приложения; обновление и распространение приложения — быстрый и автоматизированный процесс
* Обновление версий автоматическое
* Пользователи могут использовать приложение на любом компьютере, имеющем соединение с Интернет, причем обычно неважно, какая операционная система на нём установлена
* При работе веб-приложения компьютер пользователя гораздо меньше подвержен вирусному заражению, чем при запуске exe -файлов.

Поскольку *RIA* используют движок клиента для взаимодействия с пользователем, они:

1. *Богаче*. *RIA* предлагают пользовательский интерфейс, не ограниченный лишь использованием языка HTML, применяемого в стандартных веб-приложениях. Расширенная функциональность позволяет использовать такие возможности пользовательского интерфейса, как drag-and-drop, использование ползунка для изменения данных, а также возможность производить вычисления, которые не отправляются обратно на сервер, а выполняются прямо на машине пользователя (например, ипотечный калькулятор).
2. *Более интерактивные*. Интерфейсы *RIA* более интерактивны, чем стандартные интерфейсы веб-браузеров, которые требуют постоянного взаимодействия с удалённым сервером.
3. Наиболее сложные приложения *RIA* предлагают внешний вид и функциональность, близкие к настольным приложениям. Использование движка клиента позволяет добиться и других преимуществ в производительности:
4. *Сбалансированность клиент-сервера*. Использование вычислительных ресурсов клиента и сервера лучше сбалансировано. Поэтому сервер не должен быть "рабочей лошадкой", как в традиционных веб-приложениях. Это освобождает вычислительные ресурсы сервера, позволяя обрабатывать большее количество сессий одновременно за счёт одного и того же аппаратного обеспечения.
5. *Асинхронная коммуникация*. Движок клиента может взаимодействовать с сервером, не дожидаясь, пока пользователь совершит действие в приложении, нажав на кнопку или ссылку. Это позволяет пользователю просматривать страницу и взаимодействовать с ней асинхронно с помощью коммуникации между движком и сервером. Эта возможность позволяет разработчикам *RIA* передавать данные между клиентом и сервером без ожидания пользователя. В Google Maps эта техника используется для того, чтобы подгружать прилегающие сегменты карты, прежде чем пользователь пролистает, чтобы их посмотреть.

**Недостатки**

Основными недостатками и ограничениями *RIA* являются:

1. *"Песочница"*. Поскольку *RIA* загружаются в локальной *среде безопасности* "песочница", они имеют ограниченный доступ к системным ресурсам. Если права на доступ к ресурсам некорректны, *RIA* могут работать неправильно.
2. *Подключение скриптов*. Как правило, для работы *RIA* требуется JavaScript или другие скриптовые языки. Если пользователь отключил активные сценарии в своем браузере, *RIA* может не функционировать должным образом или вообще не работать.
3. *Скорость обработки клиентом*. Чтобы обеспечить платформенную независимость, некоторые *RIA* используют скриптовый язык на стороне клиента, например, такой как JavaScript, с частичной потерей производительности (серьезная проблема для мобильных устройств). Однако такая проблема не возникает при использовании встроенного языка, скомпилированного на стороне клиента, такого как Java, где производительность сопоставима с использованием традиционных встроенных языков, либо с Flash или с Silverlight, в которых программный код запускается непосредственно в *плагине* Flash Player или Silverlight соответственно.
4. *Время загрузки скрипта*. Даже если нет необходимости в установке скрипта, движок клиента *RIA* должен быть передан клиенту сервером. Поскольку большинство скриптов сохраняются в кэше, он должен быть передан хотя бы один раз. В зависимости от размера и типа передачи, загрузка скрипта может занять довольно много времени. Разработчики *RIA* могут уменьшить последствия этой задержки посредством сжатия скриптов, а также за счёт разбиения передачи приложения на несколько страниц.
5. *Утрата целостности*. Если приложение основано на X/HTML, возможны конфликты между целями приложения (которое, естественно, хочет иметь контроль над его представлением и действиями) и целями X/HTML (которое хочет отдать контроль). Интерфейс DOM для X/HTML делает возможным создание *RIA*, но это не даёт никаких гарантий, что оно будет работать корректно. Из-за того, что клиент *RIA* может изменять основную структуру приложения и переопределять его действия и представление, это может привести к ошибке приложения на стороне клиента. В конце концов, эта проблема может быть решена за счёт нового механизма клиент-сервер, предоставляющего клиенту *RIA* ограниченный доступ к изменению тех ресурсов, которые не входят в сферу его полномочий. Работа родного стандартного ПО не вызывает подобных проблем, поскольку они по определению автоматически обладают всеми необходимыми правами на локальные ресурсы.
6. *Утрата видимости для поисковых систем*. Поисковые системы могут оказаться не в состоянии проиндексировать содержимое приложения *RIA*.
7. *Зависимость от подключения к Интернету*. Идеальная замена для настольных приложений должна позволять пользователям подключаться к сети "эпизодически", покидая хот-*споты*, уходя и приходя в офис. Однако к 2007 году типичные приложения *RIA* требовали постоянного подключения.
8. *Доступность*. Известно множество проблем веб-совместимости с *RIA*. Одна из распространённых заключается в том, что пользователю, читающему текст с экрана, сложно выявлять динамические изменения (вызванные JavaScript) в контенте HTML.
9. *Отсутствие расширяемости*. *RIA* нельзя расширять плагинами и модами, как это возможно в традиционных приложениях.

**IE8 для RIA**

Если говорить о методах, которые помогли реализовать подобные возможности, а также расширили рамки для применения технологий *Web* 2.0, то в IE8 реализована *поддержка* нескольких платформ для *RIA*.

Во-первых, присутствует *AJAX*-*навигация*, которая улучшает *юзабилити* браузера за счет отмены повторных действий пользователя для ввода нового *URL* с целью перехода на новую страницу. Теперь *браузер* получает все события "на лету" и автоматически обновляет журнал посещений и адресную строку. Во-вторых, в IE8 улучшилась *поддержка* кросс-доменных запросов (*XDR*). Теперь *браузер* может выполнять, как прямые, так и анонимные запросы к общедоступным сервисам на других доменах (например, *авторизация* по *Windows* *Live* *ID*). Это позволяет сделать информацию доступной между доменами. При этом всего несколько строчек кода составляют обычный *XDR*, которые будет безопасным и быстрым - пользователи будут уверенны за свои данные, а владельцы вебсайтов определят, какую именно информацию они могут предоставить другим доменам и службам. Во-вторых, в новом браузере используется *функция* обмена сообщениями между документами (XDM) в соответствии с HTML5. За счет этого увеличивается *безопасность* взаимодействия объектов кода IFRAME и *DOM*, проблемы с которой являются распространенным явлением в существующих на рынке браузерных продуктах. В-третьих, с целью увеличения надежности работы с динамическими приложениями, было введено событие *Connection*, которое следит за состоянием подключения. Оно может найти применение в медиаплеерах и различных веб-формах, предоставляя посетителям сайта возможность работы с *RIA* в автономном режиме и в условиях медленного *интернет*-канала. Еще одно актуальное нововведение в области безопасности *AJAX* касается обработки *HTML*-кода. В IE8 сделаны серьезные улучшения безопасности, которые помогают очистить *HTML* и JavaScript (*JSON*) код от потенциально опасных исполняемых элементов сценариев.

OneRiot, компания, предоставляющая сервис по поиску в реальном времени новостей, блогов, видео и самых "горячих" событий в блогосфере, и компания Stimulant, специализирующаяся на разработке программ, использующих самые передовые технологии (мультитач) на Silverlight, выбрали IE8 для предоставления информации и сервисов своим клиентам. Для этих компаний наличие веб-фрагментов в IE8 играет большую роль, так эти веб-фрагменты могут автоматически обновляться и уведомлять клиента о наличии новой информации. На основе этих функций компания Stimulant создала для OneRiot два интересных веб-фрагмента. *Most* Shared, который показывает ссылки на самые свежие и горячие новости, записи в блогах и на социальных сетях. *Top* Videos позволяет выводить на экран 10 самых популярных видео с таких сайтов как YouTube, Hulu и Metacafe. В итоге, как отметили в OneRiot, пользователи, установившие их веб-фрагмент, заходят на *сайт* на 18% чаще, чем остальные. Для этих компаний наличие такой опции как веб-фрагмент было одним из основных критериев выбора IE8, так как это реальная возможность предоставить клиентам быстрый *доступ* к самым свежим данным. Для пользователя работать с веб-фрагментами очень просто, а для компании это хороший способ при минимальных инвестициях существенно увеличить *прибыль*. Еще один крупный игрок Интернета, компания Yahoo, используя новые решения в IE8, тоже смогла внедрить ряд интересных, полезных и привлекательных функций. Так пользователи, которые имеют *почтовый ящик* на Yahoo, теперь могут создать веб-фрагмент и всегда быть в курсе последних писем. Данная *функция* очень удобна, ведь теперь необязательно заходить непосредственно в почту, чтобы проверить новые письма. Все можно сделать за несколько секунд, в то время как просматривается любой другой *сайт*. Это экономит время, которого всегда мало. Применив ускорители IE8, компания Yahoo смогла обеспечить своим пользователям быстрый *поиск* по Yahoo Maps при выделении адреса. В данном случае нет необходимости копировать *адрес*, заходить на *сайт* Yahoo Maps и искать то или иное *место*. Простое выделение может сразу привести к нахождению необходимой локации.

Теперь в IE8 с помощью поддержки технологии ARIA (Accessible *Rich Internet Applications*) возможностей для тонкой настройки контента страницы, роли каждого элемента, доступа к нему стало больше. ARIA — это механизм, созданный для поддержки динамического содержимого страницы. Например, элементу DIV можно задать роль кнопки, поля для выделения или другую роль ARIA. Эта технология позволяет на базе *AJAX* разнообразить даже такой классический элемент как "древовидный *список*", используемый, например, при составлении карт сайта. Технология ARIA также открывает много возможностей для использования таких элементов как календарь, с возможностью выбрать конкретную дату, перетаскиваемые виджеты, *таблица*, с возможностью удобной навигации, *меню* для смены параметров (например, текста) прямо на странице, "ползунки" и многие другие. Теперь IE8 в полной мере обрабатывает ARIA объекты, правильно понимает их атрибуты и свойства, а также его разработчиками сделан ряд улучшения для поддержки стандартного синтаксиса, что упрощает создание страниц.

Модернизация веб-сайтов с помощью *RIA* и *адаптация* их под веб-стандарты в условиях современного бизнеса не может проходить без четкого экономического обоснования. Наличие поддержки этих технологий в браузере IE8 позволяет привлечь большее количество клиентов на веб-ресурсы компании, что повысит лояльность бренду и может увеличить доход фирмы.