International Journal of Computer Information Systems and Industrial Management Applications.
ISSN 2150-7988 Volume 5 (2013) pp. 060-068
© MIR Labs, www.mirlabs.net/ijcisim/index.html

On the Development of Browser Games – Current Technologies and the Future

Juha-Matti Vanhatupa

Department of Software Systems,
Tampere University of Technology,
P.O.Box 553, FIN-33101 Tampere, Finland
juha.vanhatupa@tut.fi

Abstract: **Browser games are played directly in web browsers. Consequently they do not need software installation. A modern browser game is a sophisticated combination of client and server software. Nowadays there is a wide range of different technologies used to implement browser games. Traditional implementation technologies have gained new competitors, browser plugin players, which allow sophisticated graphics. In this paper we present technologies that are used to implement browser games and also accessories used in those. Especially we concentrate on use of game engines in browser game development, and give thoughts what kind of features such game engine should possess. Towards the end of the paper we discuss the future of the browser game genre, and also the future possibilities of a web browser as a gaming platform.**

Keywords: browser games, multiplayer games, web technologies, web browser, application runtime, server-side scripting.

I. Introduction

Browser games are computer games played directly in the web browser and do not need software installation. This accessibility has made those extremely popular and browser games have lured many players who would never buy a regular computer game to start playing.

In the recent years, the evolution of browser games has been fast and a lot of new ones have been launched. There is also been evolution on technologies used in browser games. Traditional implementation technologies, PHP, ASP and Java jsp-pages have gained new competitors in the form of browser plugin players, which allow use of sophisticated 3D graphics.

In addition to implementation technologies, different kinds of accessories for browser games have appeared. The games can be located inside an application platform, for example Facebook social utility [16]. Different browser plugins are used to enchant play experience either to process game data for the player, or to modify the game graphics. Game engines have a strong position in regular computer game development and recently those have appeared into browser game development also. Most likely eventually those will establish same position in browser game development also.

In our previous paper [53] we presented a definition for a browser game, different browser game types, and financial opportunities related to the genre. The paper was later extended with community aspect and published in [54]. In this paper, we present technologies used to implement browser games and technologies used in browser game accessories. The rationale is that because of different software platforms (Web vs. binary), a game engine for browser games needs different kind of features than a traditional game engine. Currently there is no common understanding what features are important for such game engine. In this paper we present a proposal for these feature sets. We also discuss the future of the browser game genre as a whole. This paper is based on earlier conference article [55], but it has been extended with numerous technical details and with game engine specifications aspect.

The rest of the paper is structured as follows. In Section 2 we present our definition of a browser game and background from our previous article. In Section 3 we present technologies used in browser games. In Section 4 we discuss accessories used alongside browser games. In Section 5 we go through background of game engines, and in Section 6 we discuss game engines for browser game development. In Section 7 we focus on use of web browser as a gaming platform. In Section 8 we discuss the future of the browser games. In Section 9 we present related work, before concluding remarks in Section 10

II. Background

In this paper, we are interested and discuss browser-based games, which fulfill properties presented in our earlier paper [53]. Our category is quite similar to the category of long-term browser-based games, often referred as persistent browser-based games, presented in [45]. The main difference is that our definition is stricter, for example the game must be a multiplayer game to belong our browser game genre. There are also lots of other browser-based games, since for-example a solitaire played in Internet does not fulfill multiplayer property and by no means is an always running persistent world necessary for a browser-based game. In this article, we use term browser game and are interested about browser-based games, which fulfill following properties:

1) the game is a multiplayer game,
2) the game can be played directly in web browser (no separate game specific software installation),
3) the game is always on,
4) the game duration is long or eternal and
5) each player has an account, which is used to play the game.

Properties 3 and 4 together with the fact that results of several actions – for example moving troops or reallocating resources – can be seen only after real world time has passed, creates an unique type of gameplay to browser games. The player can make a few actions requiring only a moment of real world time and return later to see the results of these actions. Whereas regular computer games and massively multiplayer online role-playing games (MMORPGs) require hours long continuous play. The study [30] among over 8200 players of strategy browser game Travian [48] suggests that flexibility, in essence easy-in easy-out gameplay, was one of the two primary reasons to play browser games. The other primary reason found was social relationship involved in the game play. Browser games are social games, and huge online communities have birth alongside browser games [54]. The online comminute gathered alongside the game can be a huge motivating factor for playing, and when considering to quit playing, the community might be the reason why players keep playing [45].

Originally browser games appeared at 1990s, for example the Earth 2025 [15] was released in October 1996. It is said to be the first one of unique, interactive games designed to be played directly on the web. Nowadays web access has become almost standard in modern houses and modern mobile phones are able to access the web. This development has lead to the explosion of the number of browser games. One important motivator in recent growth of browser games is the popularity of application platforms like Facebook social utility [16]. Facebook lets users to program their own browser games, and supports several programming languages.

MMORPGs are also popular multiplayer games. Those are separated from browser games, because those games have installation software, which must be bought. However, the gameplay of a role-playing browser game, for example RuneScape [44] can be very close a regular MMORPG as it is can also be played in hours long intensive sessions. Thus the different genres easily merge to gain the best properties of both genres.

III. Browser Game Technologies

A browser game consists of several components. The server-side application runs the game and is executed at the server. A web browser used to access the game acts as the client-side application. Databases are used in the server-side to store huge amount of data needed in browser games. Components of a browser game are presented in Figure 1.

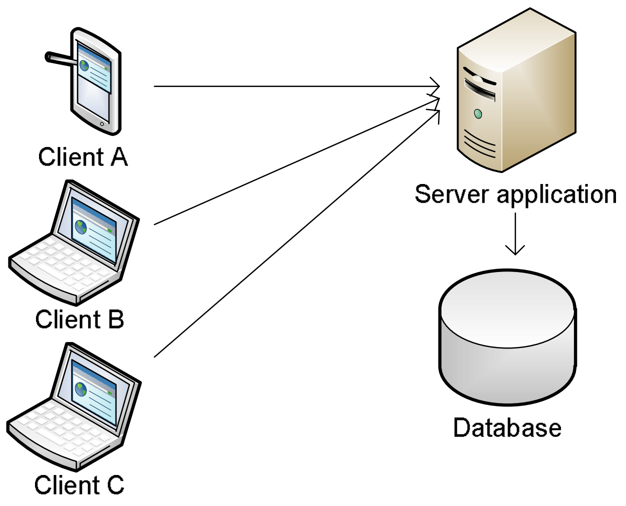


Figure 1. Browser game components.

A. Server Application

The server side applications must be robust and fast reacting. Main technologies are application runtimes, which are installed into web browser as plugins and server-side scripting. In addition to these two main technologies there are others used to implement the user interface and communication with server, and these are listed as a third group.

1) Application runtimes

Adobe Flash player [2] is a cross-platform browser-based application runtime. It is usually used for dynamic games with 2D graphics. Flash player combines animation engine with programming environment. Flash is built on an object-oriented language called ActionScript [1]. Flash Player has advantage in running graphics when compared to server side-scripting as PHP. However, 3D graphics are not as good as 2D graphics in Flash. Flash Player has small system resource requirements and can be run in Windows, Linux and Mac operating systems. A good example of browser game using Flash Player is Dark Orbit [13], a strategy / shooter game with over 35 million registered users. The player controls his starship and fights against other players. Dark Orbit also offers chance to win real money.

Adobe Shockwave Player [3] can be used 3D games with rich graphic environment. However, at the moment it is mostly used for single player games played in web browsers instead of multiplayer browser games. Shockwave Player is not available for Linux operating system.

Microsoft Silverlight [36] is a development platform that can be used to create applications for web, desktop and mobile devices. Silverlight applications can be written in any .NET programming language and .NET development tools can be used in programming Silverlight applications. At the moment there is not yet many browser games developed using Silverlight, but it is a potential environment for those games. There is an open source implementation for Linux named

Moonlight [39], which is developed by Novell in cooperation with Microsoft.

Java applets [27] are usually small applications run in the Java runtime in web browser. However, it can be used for full-scale browser games, RuneScape [44] is a very popular game launched already 1999 is done using a Java applet.

Unity web player [50] can run games built using Unity game engine [51]. The games build on top of Unity have smooth graphics. For example, Battlestar Galactica Online [9] launched at February 2011(beta test), gathered two million registered players in three months [10], and at the moment has almost nine million registered players. However, Unity web player is not available on Linux and in general it has not yet spread wide.

Modern application runtimes can run many different operating systems. Available operating systems for application runtimes are presented in Table 1.

Table 1. Application runtimes in different operating systems.

Application runtime	Windows	Linux	Mac
Flash Player	Yes	Yes	Yes
Shockwave	Yes	-	Yes
Silverlight	Yes	(Yes) 1	Yes
Java	Yes	Yes	Yes
Unity	Yes	-	Yes

1) Silverlight has an open source implementation for Linux.

2) *Server-side Scripting*

PHP [42] is a general purpose scripting language, especially suited for web development. PHP scripts can be embedded into HTML [23] documents. When a client browser requests file from the server, PHP is first interpreted in the server to HTML and then delivered to the browser. PHP is widely used in text-based browser games. It is more popular in browser games than its competitors, for example Perl or Python.

Perl programming language [41] used with CGI [11] are widely used for web development. Perl has powerful text processing facilities, and those are also useful when dealing with SQL. It is made to be easy to use, efficient and complete; however, using Perl, it is also easy to make unsecure web application.

Python [43] is another widely used programming language for web applications. It has a wide collection of web development tools and framework. However, those are not designed for web game development.

JavaServer Pages (JSP) [29] is a technology used to create dynamic web pages, which separates the user interface from content generation. JSP technology uses XML-like tags to encapsulate the logic that generates the content for web page. The page accesses server-side resources such as JavaBeans through those tags. JavaServer Pages is an extension of Java Servlet technology, which makes easier to combine fixed or static templates with dynamic content. Hyperiums [25] is a strategy browser game running since 2001 build on Java technologies. Hyperiums is purely text-based.

Active Server Pages (ASP) [34] is Microsoft's first server-side scripting environment. It allows creating and running dynamic, interactive web server applications. Usually VBScript is used with ASP, but it can be programmed with any programming language, which supports ActiveX scripting. ASP.NET [35] is a successor of ASP and no further versions of ASP are planned.

Server-side JavaScript (SSJS) [12] refers to JavaScript [28] run in the server-side, not downloaded by the application. In addition to client-side JavaScript, server-side JavaScript allows an application to communicate with a database, sharing information between the users and perform file manipulations on a server. Server-side JavaScript web pages can also contain client-side JavaScript code.

3) *User Interface and Communication*

In addition to application runtimes and server-side scripting other technologies are used too. However, a browser games are rarely implemented using only those, commonly those are used as a combination with mentioned ones. For example JavaScript can be used to handle mouse clicks, and AJAX to transfer data to PHP script acting as a game logic.

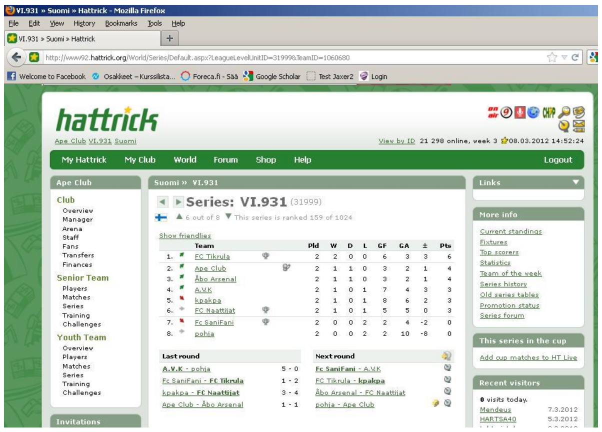
JavaScript [28] is a scripting language that can be used to add dynamic content to web pages. It can be used to implement web games and simple graphics. JavaScript was originally created to run in the client-side, but since SSJS is now available, a term client-side JavaScript (CSJS) is also in use. WebGL [57] is an extension of JavaScript that allows use of 3D graphics in a web browser. Currently, WebGL is supported by Google Chrome, Mozilla Firefox 4, Safari and Opera browsers.

AJAX [7] is a combination of web development technologies used in the client-side to create interactive web applications. These technologies communicate with the server without interfering with the shown web page. JavaScript is used to combine these technologies. Since AJAX is a combination of technologies already exists in browsers, AJAX requires no plugins or other additional components. However, AJAX/JavaScript combination has difficulties to handle differences between web browsers and need explicit code to cover those.

B. *Client Application*

The only client application needed to run a browser game is usually a web browser. The applications are also independent from the operating system of the client computer. Figure 2 shows a typical user interface of browser game, Hattrick football manager game [20].

However, each operating system cannot run every browser, and this sometimes limits playing of a certain browser game. Similar limitations relate also to other software in different operating systems, for example, at the moment iPad is lacking Adobe Flash Player support and browser games using it cannot be played with iPad.



http://www92.hattrick.org/World/Series/Default.aspx?LeagueLevelUnitID=31999&TeamID=1060680

21 298 online, week 3 08.03.2012 14:52:24

Series: VI.931 (31999)

6 out of 8 This series is ranked 159 of 1024

	Team	Pld	W	D	L	GF	GA	±	Pts
1.	FC Tikrula	2	2	0	0	6	3	3	6
2.	Ape Club	2	1	1	0	3	2	1	4
3.	Åbo Arsenal	2	1	1	0	3	2	1	4
4.	A.V.K	2	1	0	1	7	4	3	3
5.	kpakpa	2	1	0	1	8	6	2	3
6.	FC Naattijat	2	1	0	1	5	5	0	3
7.	Fc SaniFani	2	0	0	2	2	4	-2	0
8.	pohja	2	0	0	2	2	10	-8	0

Last round		Next round
A.V.K - pohja	5 - 0	Fc SaniFani - A.V.K
Fc SaniFani - FC Tikrula	1 - 2	FC Tikrula - kpakpa
kpakpa - FC Naattijat	3 - 4	Åbo Arsenal - FC Naattijat
Ape Club - Åbo Arsenal	1 - 1	pohja - Ape Club

0 visits today.

Mendeus	7.3.2012
HARTSA40	5.3.2012

Figure 2. Hattrick Browser game user interface (Copyright Hattrick Ltd, used with permission).

Modern mobile phones can execute web browsers quite easily and those have increased popularity of browser games. Many browser games offer mobile interfaces, which are simpler versions of actual ones. Those have fewer graphic and game user interface is structured for narrow screens of mobile phones. Special installable versions of browser games for mobile phones have also been made. These special versions use the same resources as original browser games.

C. Databases

Browser games need reliable and fast databases. Since any database errors and breakdowns can affect the gaming experience of thousands players, reliability is an important requirement. However, it is not extraordinary that a game situation in a browser game needs to be reversed because of a database error. Usually a game is shut down for some time and reversed to previous valid situation.

Knowledge about which database is actually used is usually secret for security reasons. Most likely choices for database management systems in browser games are for instance MySQL [40] or similar. For example, Hattrick football management browser game mentioned above is believed to run Microsoft SQL [37] on a Hitachi server park; however the actual implementation of the game database system is known only by the Hattrick Team [22].

D. Architecture

In [26] Häsel introduces two architectural choices for browser games. The first, client-server architecture is traditionally used for online games. The server keeps clients up to data and client requests are delivered to the server, which responds and sends data to the other clients.

The second, peer-to-peer architecture relies on peers that have same responsibilities. Each peer has also own copy of the game state. Peer-to-peer architecture uses less bandwidth and can be more effective than client-server architecture. In this architecture cheating can be hard to prevent as each player has own copy of the game state, which can be vulnerable to abuse.

IV. Accessories

In addition to the browser game itself, other applications can be used in when playing the game or in the development phase. The game can be located inside an application platform, for example in the Facebook social utility [16]. Modern downloadable games are usually developed using game engines, but those are yet rarely used in browser game development. Browser plugins can enchant play experience, either processing game data for the player or by modifying the game graphics. Traditional installable application can also be used for some specific tasks, where installation as a browser plugin would not give any advantage, e.g. complicated calculations, which are made weekly in real time.

A. Application Platforms

Browser games located in separate application platforms, for example Facebook platform, can be programmed in several different programming languages. The application code itself is run in a separate server. Separate application platform can offer lot of publicity for the application. However, if function calls are linked through application platform API to the application, it can downgrade the performance.

Application platform can be an advantageous environment for a browser game, since the player has already registered to the platform and he can start playing simply clicking the game. This further lowers the starting threshold for a browser game. Other benefit is that the players attract other players to play certain games. A word about good games spreads rapidly inside an application platform, either by a platform specific way, like invitations in Facebook, or by using forums inside it. This kind of spreading is natural since many application platforms are social networking platforms, and therefore reflect the social relationships of their users [33].

B. Game Engines

Game engines [18] are the most important tools in traditional computer game development, but those have not established in browser game development. There are potential for game engines in browser game development, since browser games have many common features, which could be implemented in a product platform instead of the game itself. At present, some game engines have been developed for browser game development.

Unity game engine [51] can be used to develop browser games, which run on Unity web player [50]. The game engine is originally developed for regular computer game computer games. Unity offers support for networking, scripting, physics, lightning and advanced graphics. Unity can be scripted using three programming languages, JavaScript, C#, or dialect of Python named Boo.

Unusual Engine [52] is a game engine for browser-based games. It is build on a standard Adobe Flash [2] and ActionScript 3 [1]. Features of Unusual Engine include networking, 2D and 3D graphics, physic engine, sound, scripting, animation. Unusual Engine also has Facebook support. Game engines are discussed in more detail in the following sections.

C. *Browser Plugins*

A browser game can offer separate interfaces for browser plugins. Good example of separate interface for browser plugins is in Hattrick [20]. Application developers can develop programs and get those approved by the Hattrick team; those programs can be noticed from CHPP (Certified Hattrick Product Provider) logo. There are general guidelines on what a CHPP program may do and or not. For example, a CHPP program may not track opponent's player formation changes, injuries or sold players. The system also makes it possible to arrange friendly matches through Facebook.

Graphic packages installed as browser plugins can modify browser game graphics. There can be several different themes for game. These graphic packages do not change the rules of the game and use of those has no in-game benefits. For example there are several graphic packages for Travian. Those change the appearance of buildings, maps, units, or any other graphics item available in Travian.

Browser plugins are rarely used to implement basic functionality of a browser game. However, for example The Nethernet [47] is played through a special Firefox toolbar. The story of game is about an Internet war between Order and Chaos. The players browse the web and leave traps and other tools to the websites, which affect others visiting same websites. The Nethernet toolbar needs to be installed to play the game, and it is installed to the web browser as plugin and not as a traditional software installation.

D. *Other Accessories*

Separate installable applications are used as accessories, when playing a browser games. Those can be more suitable than browser plugins if the operation is only needed from time to time and it would be overkill to launch the application every time the browser game is played. For example, Hattrick Manager [21] calculates optional line-up for the player's team, tracks players training progress, and offers many ways to sort and present game data.

There are no limitations for programming languages used in those external applications. However, the application needs to be able to call functions in the browser game API if it wants to access it. Usually these external applications must access the browser game API to load the game data, which is then used in the application.

V. Traditional Game Engines

A modern game engine is a massive collection of software components. Typical components of a traditional game engine are presented as layer architecture in Figure 3. The core system refers to the most essential systems required by the game engine, for example memory management, IO access, and possibly other components required by the used programming language and environment. In most game engines the game loop, which calls other components, is located into the core system.

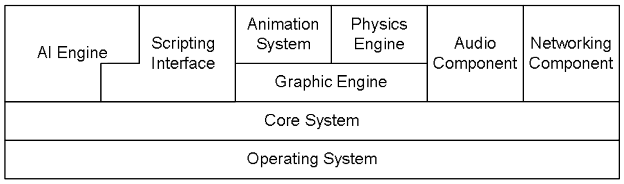


Figure 3. Game engine components.

In addition to the run-time component of a game engine, several separated tools are used when creating the game. Most of these tools relate to content creation, for example, game world editor, 3D modeling tool (for creating game characters and items), audio tool. In theory, for those tools it does not matter is the game browser-based or a regular computer game. However, at the moment, browsers are not capable of executing similar quality computer graphics as can be used in traditional computer games. Although the situation is changing and browsers are becoming more powerful application platform.

VI. Game Engines for Browser Games

A game engine for browser games needs different kind of features than a traditional game engine used to build installable computer games. We have divided the features into mandatory and optional categories. Mandatory features are useful for all browser games. Optional features are features that game engine for browser-based games can benefit, however those features i) are not useful for every type of browser games or ii) the benefit of having those is rather small. In addition to these groups we have listed few features that are common in traditional game engines, but are not significantly useful in when building browser games.

A. *Mandatory Features*

Registration is the only mandatory step needed to start playing a browser-based game. Since every browser-based game needs registration system, the functionality could be easily moved to the game engine. An application platform can also be used to automate the registration process, and therefore lower the starting threshold for playing the game. Though, if application platform handles the registration, registration functionality of the game engine is still needed to execute player data to the game database.

Networking component is responsible for handling network connections. The game engine would also ensure that game data relates to correct players. In the game code the players are seen only as a game object and no information about how the player is connected is present, that information is handled in the game engine.

Security features should be implemented in the game engine, and in addition also in the game itself, because security is cross-cutting aspect. Building security for browser-based game starts from design of the game application logic. Since the data send from the client can be manipulated, the client should not be able directly to command the server: (add points to this player or move this player to …). The server should always check what the game status is and ensure validity of client commands.

Database access is essential for every browser-based game. It suits well for implemented in game engine level. A game engine can offer easy access interface for using the database, in this way the SQL statements are used only in the game engine. The game code calls the game engine and data that needs to be updated into the database is, for example delivered as a parameter.

Player messaging system lets players send messages to each others. It works as in-game email system. This kind of slow messaging is suitable for most browser-based games. However, many browser games based on the player's fast reflexes, for example shooter games, are too fast proceeding that the players has no time for writing messages during matchers. In those games, the messaging system could be used between matches. In general, a player message system is very useful feature for a game engine.

Graphic interface offers functions for easily presenting graphics. It is necessary for all other types than text-based games. Publishing text-based games to be played in browser is truly a minor activity nowadays, therefore it can be generalized that graphic interface is a mandatory feature. Quality of a graphic interface can vary a lot from a few functions capable of creating 2D graphics into full-scale 3D interface.

B. Optional Features

Real-time player chat is a real-time chat between players. Need for instant messaging depends heavily on the type of the game. It might not be realistic to offer instant open chat in a browser-based game with a thousand players. However, in such game, a clan might need an internal communication channel, and for example IRC is commonly used for such.

Audio component is also most useful in fast proceeding action browser-based games. Many strategy browser-based games do not include sound effects or music any kind. The game flow of those games is slow; therefore sounds are not needed for confirmations to user actions. In addition strategic browser-based games are commonly played in public places for example, in workplaces; the players would mute the sounds anyway.

Animation system refers to ready-made interface offering possibility to add animations to graphic objects in the game. Benefits of animation interface are linked with level of graphics needed by the game. Usually benefits of possibility to animate graphics are very limited in development of strategic browser-based games. However, a ready-made animation interface could be very useful in fast-proceeding action games.

Facebook support is one potential feature for a game engine at the moment. Facebook social utility [16] is currently the most popular website in the United States; it is very popular in elsewhere too. An application platform can be advantageous environment for browser-based game, since those automate the registration process and can offer publicity for the game.

Artificial intelligence (AI) engine is useful in most browser-based games. However, all of them do not include computer controlled players. For most browser-based games AI engine does not need to be very sophisticated, basic functionality that enables simple AI enemies to be implemented easily is enough.

Physic engine is a common feature in traditional game engines. However, its usefulness is questionable in a game engine for browser-based games, because only a small amount of those use a kind of 3D graphics that benefits from a physic engine. Although there are many non-browser, installable games using heavily physic engine. For example Angry birds [6], is currently very popular, because modern touch screen devices have made new forms of gaming possible. There are also browser-based conversions of these games; however, usually those are not multiplayer games and do not fill other properties also.

C. Left out

A scripting interface is a very common feature of traditional game engines. It can speed up the development of game rapidly and can be left open, to provide an interface for players to build their own modifications from the game. However, many browser-based games are already built using a script language; therefore there is no use of building a separate interface for another scripting language. In addition, because a server-side part of the browser-based game is located to a server, it would be unwise to let players execute their own script code to the game.

VII. Web Browser as a Gaming Platform

The Web has gone through a long evolution from a simple document browsing and distributing environment to a rich software platform [8]. These early roots of Web are still evident and traditionally it has been difficult to build sophisticated, interactive applications without using plugins. This situation has hindered browser game development; earlier browser games were purely text-based. Recently plugins have spread and quality of graphics and other features of browser games have improved.

Forthcoming technologies will improve the situation even further, those aim at improving the use of the Web as a real application platform. The forthcoming HTML5 standard [24] adds new features to existing HTML standard. From several new features, some affecting game development are: ability to embed video and audio directly to web pages, drag-and-drop capabilities, graphic features and offline storage database. This offline storage database possibly can be used to allow the player make his moves in browser game even the network connection is not available, and update the moves when the network reconnects.

WebGL [57], already mentioned in the technologies section, is a cross-platform web standard for hardware accelerated 3D graphics. It allows use of 3D graphics natively in the web browser without plugins. Because this eliminates the need to install separate plugins for gaming, it can have great impact to browser game development and increase the popularity of browser games.

In general, the future of web browser as a gaming platform seems bright. Currently new browser games are already in quality close to their installable cousins and forthcoming technologies will shorten the gap between those even further.

VIII. Future of Browser Games

The software industry is currently experiencing a paradigm shift towards web-based software [46]. The majority of new software applications intended for desktop computers are web applications. Web-based software requires no installation or manual upgrades and their distribution is superior to conventional desktop-style applications.

We believe that the trend includes computer games as well as other software applications, and playing directly from the web without installation on the local computer, is finally the future of all computer games. Playing can be also paid directly on web. At the moment, almost all browser games are free to play, but when the regular computer games transforms to be played from the web, the profit must be created by monthly payments. However, the gameplay of browser games and regular computer games are different: short sessions often and constantly on versus hours long intensive play. Therefore they remain as separate game genres; even if both would be played using web browser.

There are also signs that people use more time using applications, which use the web directly, but less time on browsing the web [4]. If this trend continues and external applications are created to connect web resources, it helps the user to access the browser game without selecting the game from his bookmarks or typing the web address. Although the player would use a platform specific, external application (native app) when accessing the game from his computer or mobile phone, the game is still accessible from public computers through a web browser. This battle between browser-based and native applications [38] will settle in the future. However, as presented for success of browser games it does not matter if native apps are created to access those. Of course, it can be argued if a browser game accessed by native app is truly a browser game, because native app is installable software, but this is discussion is very close to nit-picking.

In the future mobile phones are even more close to personal computers. This opens more market for browser games, because people carry gaming device all the time, instead of that gaming is limited to the time they are at front of a computer.

Although, game engines have not established position in browser game development, it is probable that those will be important tools in the future in the field. Currently application platforms automate the registration process in browser games, but there are lots of common development issues, for example networking, database access, player chats, graphics, which can be implemented in a game engine. A game engine for browser games would also add security features like authentication of browser clients.

In addition to technologic issues, the success of browser games depends on social aspect. The nature of browser games is highly addictive, and usually the time spent on a browser game increase daily. When the player starts playing a new browser game, the game might need only a minute at the first day. Then two minutes at the second day, and so on. In strategy games the controlled territory expanses and the player needs more time just to play at same level as before. Even in manager games, where the basic functionality of the game, for example matches played per week, is not expanding, time used for searching new players and other supportive actions increases as the player advances in the game. The amount of real world time that the game demands for the player to be affective in the game world and interference with the gamers' personal life, are two important reasons why people quit playing online games [5]. However, the social pressure can be smaller and constant playing more acceptable, if the browser game is located inside a popular application platform, which the player's family and friends use a lot. These social reasons to quit playing are difficult to overcome by new technology innovations.

IX. Related Work

Development of sophisticated web applications has become increasingly complex. Kuuskeri and Mikkonen [32] aim at implementing "fat clients" and running application mostly on the client. They have created a set of guidelines how applications should be divided between the server and the client. By following these directives application developers can address the issues of complexity that are common in modern web development. Their implementation is done using server-side JavaScript. "Fat clients" could reduce the server load and make a browser game run smoother as the client could handle most user events. Security issues are intentionally left out of the paper. However, those are very important in browser games and secure environment is one requirement for successful browser game.

The main drawback for games to use a peer-to-peer architecture is lack of a central authority that regulates access and prevents cheating. Hampel et al. [19] present a peer-to-peer architecture based on an overlay network using distributed hash tables with support for persistent object storage and event distribution. They present concept of sets of controlled peers that supervise each others. This kind of redundancy can prevent cheating and improve stability by eliminating single point of failures. Goodman and Verbrugge [17] present a design for a cheat-resistant, scalable hybrid network model for massive multiplayer online games. In their design, the gamestate is still controlled by a centralized server, acting both a login point and arbiter of client behavior. Computations are distributed between clients to reduce computational load and thus increase scalability of the central server. Calculations are verified through a simple peer auditing scheme. They present that cheating can be effectively controlled in a semi peer-to-peer system with good scalability and acceptable overhead.

Performance is one important requirement for implementation technology of a browser game. PHP is a widely used scripting language in text-based browser games. Trent et al. [49] present performance comparison of PHP and JSP as server-side scripting languages. They present that under high loads JSP tends to perform better than PHP, however, if a 5%-10% difference in throughput and performance is acceptable, then implementers of a web system can achieve similar results using either PHP or JSP.

In addition to already presented game engines for browser games, Unity [51] and Unusual Engine [52], there are also other game engines for browser game development.

Vision [56] is a commercial multi-platform game engine by Trinigy. Vision was originally released in 2003. By using WebVision, projects made using Vision game engine can be ported to be the browser-based. WebVision is free for anyone licensing Vision game engine.

Aves is a HTML5 game development framework for JavaScript that does not use plugins to run. At the moment Aves is owned by Zynga and it is not available for market, however there has been discussion about releasing the gaming component of Aves as open source. Most likely we will see more game engines using JavaScript/HTML5 combination in the future as the forthcoming standard HTML5 becomes general.

Lively Kernel [31] is a platform for developing and hosting client side applications implemented in JavaScript only. Although Lively Kernel is not designed to be a game engine, there are game applications present also in it, and it implements some features required for a game engine. For example user chat and user identification (registration) are present in the latest version.

Facebook [16] platform already mentioned earlier contains wide range of game applications. Di Loreto and Abdelkader [14] show that in the Facebook context playfulness is linked to blending of personal aspects and social aspects. The presence of friends in Facebook is the lever to push players return to use the application. In the end, the users themselves create playfulness.

X. Conclusions

Since web access has become common and modern mobile phones can also be used to play browser games, people have more opportunities to play. Generally, the future of browser games looks bright and it is probable that new technologies, for example specific game engines for browser games, allow even more sophisticated browser games. However, non-technical issues are also affecting success of browser games. Browser games are always running, therefore they are very addictive and need a lot of time.

In this paper, we presented current technologies used to implement browser games. Evolution has brought browser games in quality closer to their regular counterparts. Most likely this trend continues when game engines for browser games become common and produce same quality improvement as they did to the regular computer games. New forthcoming web standards and technologies, such as HTML5 will also make the Web better gaming platform, improving the quality of browser games.

Based on recent development towards web-based apps, we claim that finally all computer games are played directly in the web and not installed to the local computer. The trend will cause improvement on web technologies, since almost all software development is then web application programming and the current wide collection of different web technologies evolves into more sophisticated technologies and standards. Possibly with the evolution of web technologies, the number of technologies used to implement browser games, will decrease, and some set of technologies and game engines, instead of the current wide collection, will have a dominant position in the browser game development.

References

[1] ActionScript Technology Center, http://www.adobe.com/devnet/actionscript.html

[2] Adobe Flash Player, http://www.adobe.com/products/flashplayer/

[3] Adobe Shockwave Player, http://www.adobe.com/products/shockwaveplayer/

[4] C. Anderson and M. Wolff, "The Web is Dead. Long Live the Internet". *Wired*. pp. 118-127, 2010.

[5] D. Anderson, "The Dark Side of MMOGs: Why People Quit Playing". In *Proceedings of CGAMES'2009*, pp. 76–80, 2009.

[6] Angry Birds game, http://www.angrybirds.com/

[7] AJAX, http://www.adaptivepath.com/ideas/essays/archives/00 0385.php

[8] M. Anttonen, A. Salminen, T. Mikkonen and A. Taivalsaari. "Transforming the Web into a Real Application Platform: New Technologies, Emerging Trends and Missing Pieces". In *Proceedings of the 2011 ACM Symposium on Applied Computing* (SAC 2011), pp. 800-807, 2011.

[9] Battlestar Galactica Online, browser game, http://us.battlestar-galactica.bigpoint.com/

[10] Bigpoint company press release, http://bigpoint.net/2011/05/battlestar-galactica-online-blasts-past-two-million-registered-players-in-three-mo nths

[11] Common gateway interface, http://www.w3.org/CGI/

[12] CommonJS, a project defining APIs for JavaScript outside the browser, http://www.commonjs.org/

[13] Dark Orbit browser game, http://www.darkorbit.com/

[14] I. Di Loreto and G. Abdelkader, "Facebook Games: Between Social and Personal Aspect". *International Journal of Computer Information Systems and Industrial Applications*, III, pp. 713–723, 2011.

[15] Earth 2025 browser game, http://games.swirve.com/earth

[16] Facebook social utility, http://www.facebook.com/

[17] J. Goodman, C. Verbrugge, "A Peer Auditing Scheme for Cheat Detection in MMOGs". In *Proceedings of 7th ACM SIGCOMM Workshop on Network and System Support for Games* (Net Games'08), 2008.

[18] J. Gregory, *Game Engine Architecture*. AK Peters, Ltd, 2009.

[19] T. Hampel, T. Bopp, and R. Hinn, "A Peer-to-Peer Architecture for Massive Multiplayer Online Games". In *Proceedings of 5th ACM SIGCOMM Workshop on Network and System Support for Games* (Net Games'06), 2006.

[20] Hattrick browser game, http://www.hattrick.org/

[21] Hattrick Manager Application www site, http:// www.hattrickmanager.org/
[22] Hattrick Wiki Datapage page, http://wiki.hattrick.org/wiki/Database
[23] HTML, http://www.w3.org/html/
[24] HTML5, http://www.w3.org/TR/html5/
[25] Hyperiums browser game, http://www.hyperiums.com/
[26] M. Häsel, "Rich Internet Architectures for Browser-Based Multiplayer Real-Time Games – Design and Implementation Issues of virtual-kicker.com". In *Proceedings of Network-Based Information Systems: First International Conference* (NBiS 2007), LNCS, vol. 4658, pp. 157–166, 2007.
[27] Java Applets, http://java.sun.com/applets/
[28] JavaScript Mozilla's Official documentation, https://developer.mozilla.org/en/JavaScript
[29] JavaServer Pages, http://java.sun.com/products/jsp/overview.html
[30] C. Klimmt, H. Schmid, and J. Orthmann, "Exploring the Enjoyment of Playing Browser Games". *CyberPsychology & Behavior*, XII (2), pp. 231-234, 2009.
[31] J. Kuuskeri, J. Lautamäki and T. Mikkonen, "Peer-to-Peer Collaboration in the Lively Kernel". In *Proceedings of the 2010 ACM Symposium on Applied Computing* (SAC 2010), pp. 812-817, 2010.
[32] J. Kuuskeri, and T. Mikkonen, "Partitioning Web Applications Between the Server and the Client". In *Proceedings of 2009 ACM Symposium on Applied Computing* (SAC 2009), pp. 647–652, 2009.
[33] N. Mattar and T. Pfeiffer, "Interactive 3D Graphs for Web-based Social Networking Platforms". *International Journal of Computer Information Systems and Industrial Applications*, III, pp. 427-434, 2011.
[34] Microsoft Active Server Pages (ASP), http://msdn.microsoft.com/en-us/library/aa286483.aspx
[35] Microsoft ASP.NET, http://www.asp.net/
[36] Microsoft Silverlight, http://www.microsoft.com/silverlight/
[37] Microsoft SQL Server, http://www.microsoft.com/sqlserver/en/us/default.aspx
[38] T. Mikkonen and A. Taivalsaari. "Reports of the Web's Death are Greatly Exaggerated", *Computer*, XLIV (5), pp. 30-36, 2011.
[39] Moonlight, http://www.mono-project.com/Moonlight
[40] MySQL, http://www.mysql.com/products/enterprise/database/
[41] Perl programming language, http://www.perl.org/
[42] PHP scripting language, http://www.php.net/
[43] Python programming language, http://www.python.org/
[44] RuneScape browser game, http://www.runescape.com/title.ws
[45] D. Schultheiss, N.D. Bowman, C. Schumann, "Community vs solo-playing in multiplayer internet games". In *Proceedings of The [Player] Conference*, pp. 452-471, 2008.
[46] A. Taivalsaari, T. Mikkonen, M. Anttonen, and A. Salminen, "The Death of Binary Software: End User Software Moves to the Web". In *Proceedings of The Ninth International Conference on Creating, Connecting and Collaborating through Computing* (C5'11), IEEE, 2011.
[47] The Nethernet browser game, http://www.pmog.com/
[48] Travian browser game, http://www.travian.com/
[49] S. Trent, M. Tatsubori, T. Suzumura, A. Tozawa and T. Onodera, "Performance Comparison of PHP and JSP as Server-Side Scripting Languages". In *Proceedings of Middleware 2008, ACM/IFIP/USENIX 9th International Middleware Conference*, pp. 164-182, 2008.
[50] Unity Web Player, http://unity3d.com/webplayer/
[51] Unity game engine, http://unity3d.com/
[52] Unusual Engine, http://www.unusualengine.com/
[53] J-M. Vanhatupa. "Browser Games: The New Frontier of Social Gaming". In *Proceedings of The Second conference on Wireless & Mobile Networks* (WiMo 2010), pp. 349-355, 2010.
[54] J-M. Vanhatupa. "Browser Games for Online Communities", *International Journal of Wireless & Mobile Networks*, II (3), pp. 39-47, 2010.
[55] J-M. Vanhatupa. "On the Development of Browser Games – Technologies of an Emerging Genre", In *Proceedings of 7th International Conference on Next Generation Web Services and Practices* (NWeSP), pp. 363-368, 2011.
[56] Vision game engine, http://www.trinigy.net/en/products/vision-engine
[57] WebGL, http://www.khronos.org/webgl/

Author Biographies

Juha-Matti Vanhatupa Juha-Matti Vanhatupa is a postgraduate student in Department of Software Systems at Tampere University of Technology. He received M.Sc. in software engineering at 2007. His research interests include browser games, tool support for game programming and content generation for computer games.