US009271015B2

(12) **United States Patent**
Bloch et al.

(10) **Patent No.: US 9,271,015 B2**
(45) **Date of Patent: *Feb. 23, 2016**

(54) **SYSTEMS AND METHODS FOR LOADING MORE THAN ONE VIDEO CONTENT AT A TIME**

(71) Applicant: **JBF Interlude 2009 LTD-Israel**, Tel Aviv-Yafo (IL)

(72) Inventors: **Jonathan Bloch**, Brooklyn, NY (US); **Barak Feldman**, Tenafly, NJ (US); **Tal Zubalsky**, Tel Aviv-Yafo (IL); **Kfir Y. Rotbard**, Ramat HaSharon (IL)

(73) Assignee: **JBF Interlude 2009 LTD** (IL)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 84 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **14/069,694**

(22) Filed: **Nov. 1, 2013**

(65) **Prior Publication Data**

US 2014/0178051 A1 Jun. 26, 2014

Related U.S. Application Data

(63) Continuation of application No. 13/437,164, filed on Apr. 2, 2012, now Pat. No. 8,600,220.

(51) **Int. Cl.**

H04N 5/76	(2006.01)
H04N 5/78	(2006.01)
H04N 21/2343	(2011.01)
H04N 21/433	(2011.01)

(Continued)

(52) **U.S. Cl.**

CPC ***H04N 21/23439*** (2013.01); ***H04N 21/433*** (2013.01); ***H04N 21/47202*** (2013.01); ***H04N 21/8541*** (2013.01)

(58) **Field of Classification Search**

CPC H04N 21/23439; H04N 21/433; H04N 21/47202; H04N 21/8541

USPC 386/296, 293, 248, 290, 291, 323, 324, 386/341

See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,568,602	A	10/1996	Callahan et al.
5,607,356	A	3/1997	Schwartz
5,636,036	A	6/1997	Ashbey

(Continued)

FOREIGN PATENT DOCUMENTS

DE	10053720	A1	4/2002
EP	1033157	A2	9/2000

(Continued)

OTHER PUBLICATIONS

An ffmpeg and SDL Tutorial, "Tutorial 05: Synching Video," Retrieved from internet on Mar. 15, 2013: <http://dranqer.com/ffmpeg/tutorial05.html>, (4 pages).

(Continued)

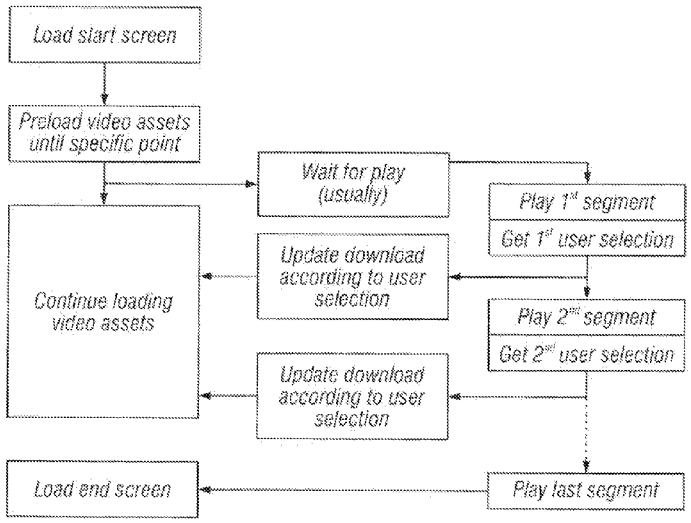
Primary Examiner — Robert Chevalier

(74) *Attorney, Agent, or Firm* — Goodwin Procter LLP

(57) **ABSTRACT**

A system for loading videos includes an interactive video player is with a loader. A product configuration file in operation configures files for a user in creation of a custom video. External assets are configured for a design of an interactive layer of the video. The interactive video player in operation creates in real-time a custom video that includes a plurality of video segments.

19 Claims, 6 Drawing Sheets



(51) **Int. Cl.**
H04N 21/472 (2011.01)
H04N 21/8541 (2011.01)

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,734,862	A	3/1998	Kulas
6,122,668	A	9/2000	Teng et al.
6,128,712	A	10/2000	Hunt et al.
6,191,780	B1	2/2001	Martin et al.
6,222,925	B1	4/2001	Shiels et al.
6,298,482	B1	10/2001	Seidman et al.
7,379,653	B2	5/2008	Yap et al.
7,444,069	B1	10/2008	Bernsley
7,627,605	B1	12/2009	Lamere et al.
7,917,505	B2	3/2011	van Gent et al.
8,065,710	B2	11/2011	Malik
8,190,001	B2	5/2012	Bernsley
8,650,489	B1	2/2014	Baum et al.
9,021,537	B2	4/2015	Funge et al.
2002/0091455	A1	7/2002	Williams
2002/0120456	A1	8/2002	Berg et al.
2002/0177914	A1	11/2002	Chase
2003/0159566	A1	8/2003	Sater et al.
2003/0183064	A1	10/2003	Eugene et al.
2003/0184598	A1	10/2003	Graham
2003/0221541	A1	12/2003	Platt
2004/0138948	A1	7/2004	Loomis
2005/0055377	A1	3/2005	Dorey et al.
2005/0091597	A1	4/2005	Ackley
2005/0102707	A1	5/2005	Schnitman
2006/0028951	A1	2/2006	Tozun et al.
2006/0064733	A1	3/2006	Norton et al.
2006/0150072	A1	7/2006	Salvucci
2006/0155400	A1	7/2006	Loomis
2006/0200842	A1	9/2006	Chapman et al.
2006/0224260	A1	10/2006	Hicken et al.
2007/0118801	A1	5/2007	Harshbarger et al.
2007/0157261	A1	7/2007	Steelberg et al.
2007/0162395	A1	7/2007	Ben-Yaacov et al.
2007/0239754	A1	10/2007	Schnitman
2008/0021874	A1	1/2008	Dahl et al.
2008/0022320	A1	1/2008	Ver Steeg
2008/0031595	A1	2/2008	Cho
2008/0086754	A1	4/2008	Chen et al.
2008/0091721	A1	4/2008	Harboe et al.
2008/0092159	A1	4/2008	Dmitriev et al.
2008/0148152	A1	6/2008	Blinnikka et al.
2008/0276157	A1	11/2008	Kustka et al.
2008/0300967	A1	12/2008	Buckley et al.
2008/0301750	A1	12/2008	Silfvast et al.
2008/0314232	A1	12/2008	Hansson et al.
2009/0022015	A1	1/2009	Harrison
2009/0024923	A1	1/2009	Hartwig et al.
2009/0055880	A1	2/2009	Batteram et al.
2009/0063681	A1	3/2009	Ramakrishnan et al.
2009/0116817	A1	5/2009	Kim et al.
2009/0199697	A1	8/2009	Lehtiniemi et al.
2009/0228572	A1	9/2009	Wall et al.
2009/0320075	A1	12/2009	Marko
2010/0077290	A1	3/2010	Pueyo
2010/0153512	A1	6/2010	Balassanian et al.
2010/0161792	A1	6/2010	Palm et al.
2010/0167816	A1	7/2010	Perlman et al.
2010/0186579	A1	7/2010	Schnitman
2010/0262336	A1	10/2010	Rivas et al.
2010/0268361	A1	10/2010	Mantel et al.
2010/0278509	A1	11/2010	Nagano et al.
2010/0287033	A1	11/2010	Mathur
2010/0287475	A1	11/2010	van Zwol et al.
2010/0293455	A1	11/2010	Bloch
2010/0332404	A1	12/2010	Valin
2011/0007797	A1	1/2011	Palmer et al.
2011/0010742	A1	1/2011	White
2011/0026898	A1	2/2011	Lussier et al.
2011/0096225	A1	4/2011	Candelore
2011/0126106	A1	5/2011	Ben Shaul et al.
2011/0131493	A1	6/2011	Dahl
2011/0138331	A1	6/2011	Pugsley et al.
2011/0191684	A1	8/2011	Greenberg
2011/0197131	A1	8/2011	Duffin et al.
2011/0200116	A1	8/2011	Bloch et al.
2011/0246885	A1	10/2011	Pantos et al.
2011/0252320	A1	10/2011	Arrasvuori et al.
2011/0264755	A1	10/2011	Salvatore De Villiers
2012/0005287	A1	1/2012	Gadel et al.
2012/0094768	A1	4/2012	McCaddon et al.
2012/0110620	A1	5/2012	Kilar et al.
2012/0198412	A1	8/2012	Creighton et al.
2012/0308206	A1	12/2012	Kulas
2013/0046847	A1	2/2013	Zavesky et al.
2013/0054728	A1	2/2013	Amir et al.
2013/0055321	A1*	2/2013	Cline et al. 725/77
2014/0040280	A1	2/2014	Slaney et al.
2014/0129618	A1	5/2014	Panje et al.

FOREIGN PATENT DOCUMENTS

EP	2104105	A1	9/2009
GB	2359916	A	9/2001
GB	2428329	A	1/2007
JP	2008005288	A	1/2008
WO	WO-00/59224	A1	10/2000
WO	WO-2007/062223	A2	5/2007
WO	WO-2007/138546	A2	12/2007
WO	WO-2008/001350	A2	1/2008
WO	WO-2008/052009	A2	5/2008
WO	WO-2008/057444	A2	5/2008
WO	WO-2009/137919	A1	11/2009

OTHER PUBLICATIONS

Archos Gen 5 English User Manual Version 3.0, Jul. 26, 2007, pp. 1-81.

Gregor Miller et al. "MiniDiver: A Novel Mobile Media Playback Interface for Rich Video Content on an iPhoneTM", Entertainment Computing a ICEC 2009, Sep. 3, 2009, pp. 98-109.

International Search Report for International Patent Application PCT/IL2010/000362 dated Aug. 25, 2010 (2 pages).

International Search Report for International Patent Application PCT/IL2012/000080 dated Aug. 9, 2012 (4 pages).

International Search Report for International Patent Application PCT/IL2012/000081 dated Jun. 28, 2012 (4 pages).

International Search Report and Written Opinion for International Application PCT/IB2013/001000 mailed Jul. 31, 2013, 12 pages.

Labs.byHook: "Ogg Vorbis Encoder for Flash: Alchemy Series Part 1," [Online] Internet Article, Retrieved on Jun. 14, 2012 from the Internet: URL:http://labs.byhook.com/2011/02/15/ogg-vorbis-encoder-for-flash-alchemy-series-part-1/, 2011, (pp. 1-8).

Sodagar, I., (2011) "The MPEG-DASH Standard for Multimedia Streaming Over the Internet", *IEEE Multimedia*, IEEE Service Center, New York, NY, US, vol. 18, No. 4, pp. 62-67.

Supplemental European Search Report for EP10774637.2 (PCT/IL2010/000362) mailed Jun. 20, 2012 (6 pages).

Supplemental European Search Report for EP13184145 dated Jan. 30, 2014 (6 pages).

Yang, H, et al., "Time Stamp Synchronization in Video Systems," Teletronics Technology Corporation, <http://www.ttcdas.com/products/daus encoders/pdf/ tech papers/tp 2010 time stamp video system.pdf>, Abstract, (8 pages).

Barlett, Mitch, "iTunes 11: How to Queue Next Song," Technipages, Oct. 6, 2008, pp. 1-8, retrieved on Dec. 26, 2013 from the internet http://www.technipages.com/itunes-queue-next-song.html.

* cited by examiner

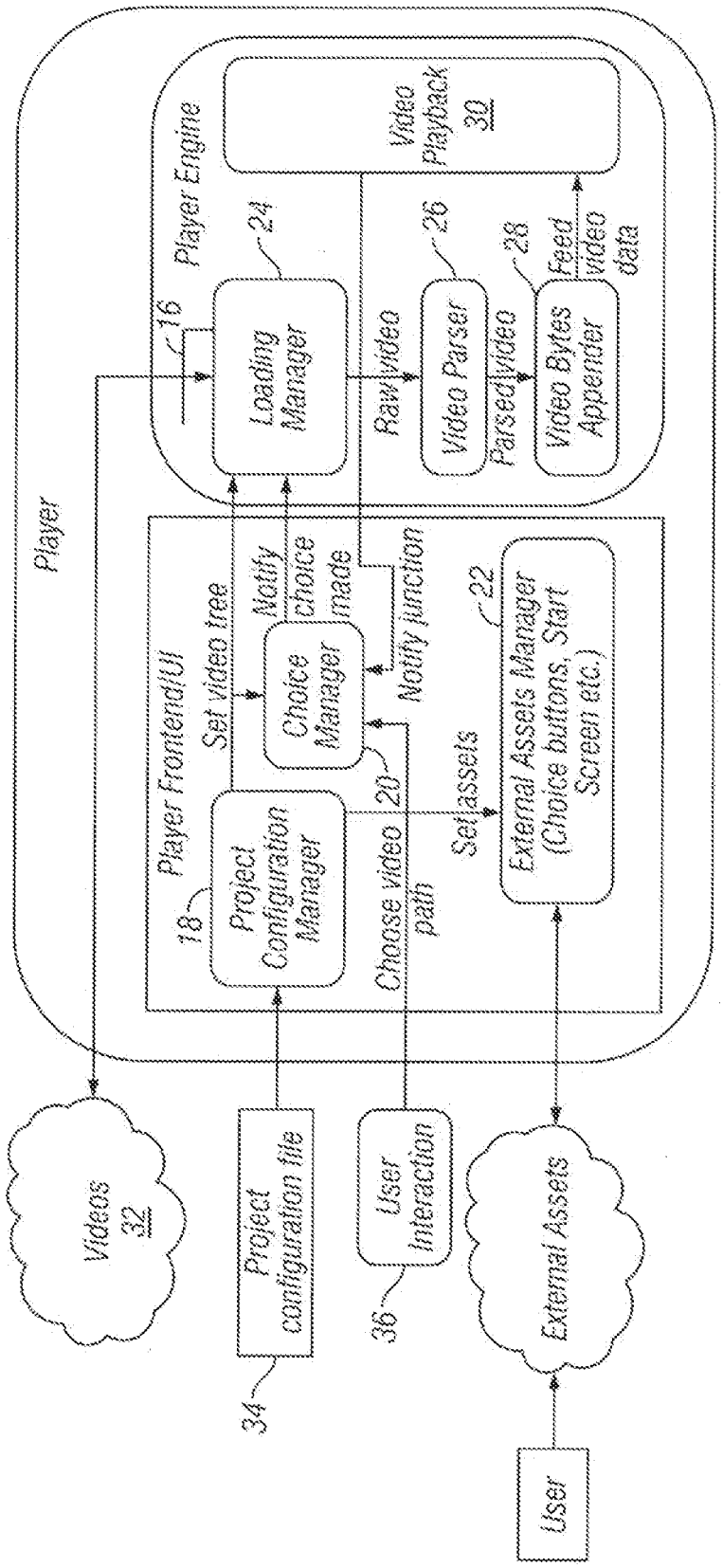

10
Player
Player Engine
Video Playback 30
24
Loading Manager
16
Raw video
26
Video Parser
Parsed video
28
Video Bytes Appender
Feed video data
Player Frontend/UI
Set video tree
Notify choice made
Choice Manager
Notify junction
22
External Assets Manager (Choice buttons, Start Screen etc.)
20
Set assets
18
Project Configuration Manager
Choose video path
Videos 32
Project configuration file
34
User Interaction
36
External Assets
User

FIG. 1

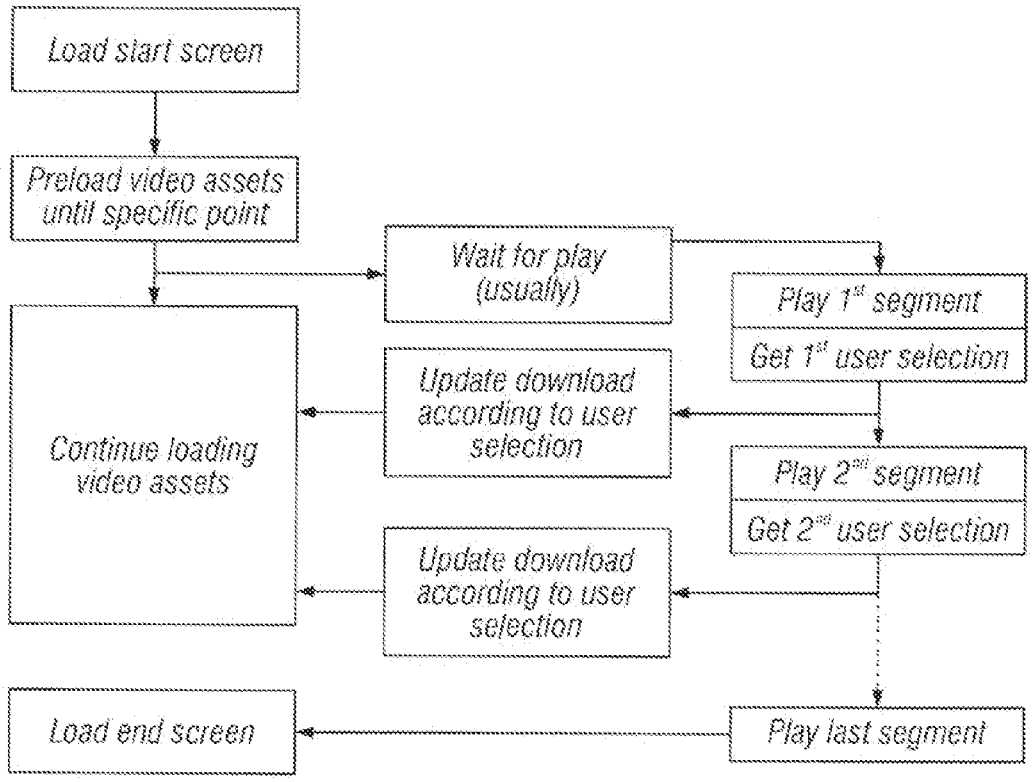

Load start screen
Preload video assets until specific point
Wait for play (usually)
Play 1st segment
Get 1st user selection
Update download according to user selection
Continue loading video assets
Play 2nd segment
Get 2nd user selection
Update download according to user selection
Load end screen
Play last segment

FIG. 2

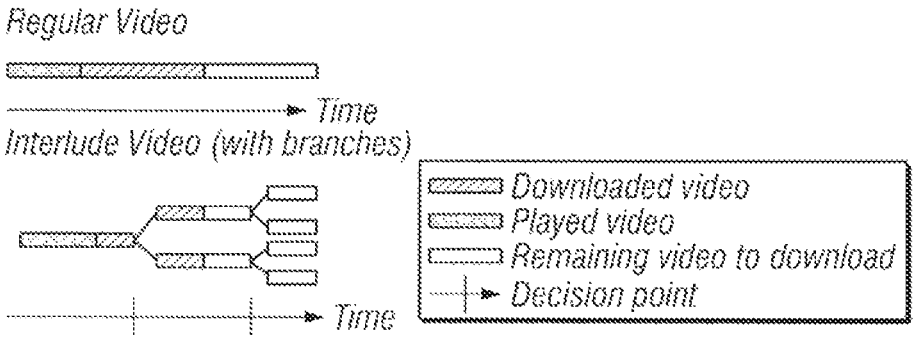

Regular Video
Time
Interlude Video (with branches)
Time
Downloaded video
Played video
Remaining video to download
Decision point

FIG. 3

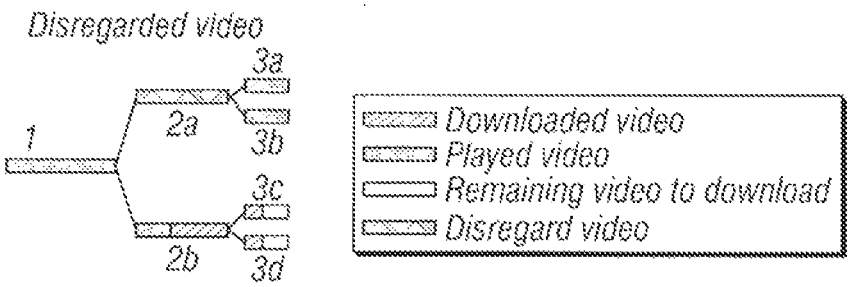

Disregarded video
1
2a
2b
3a
3b
3c
3d
Downloaded video
Played video
Remaining video to download
Disregard video

FIG. 4A

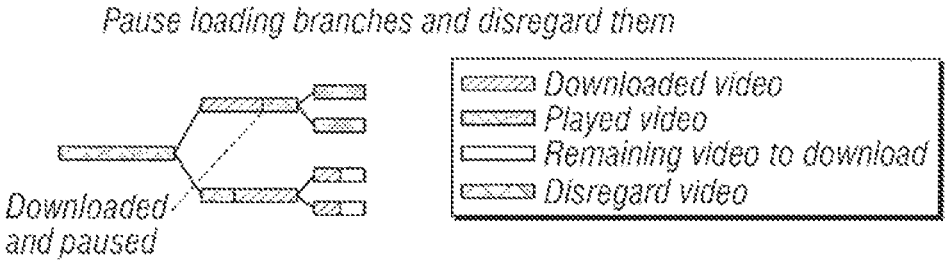


FIG. 4B

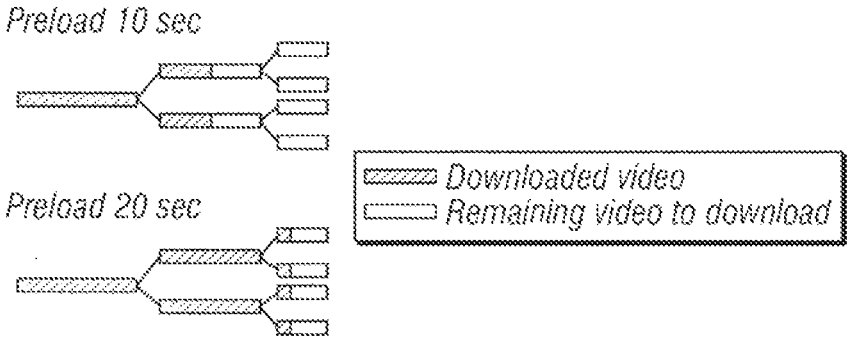


FIG. 5

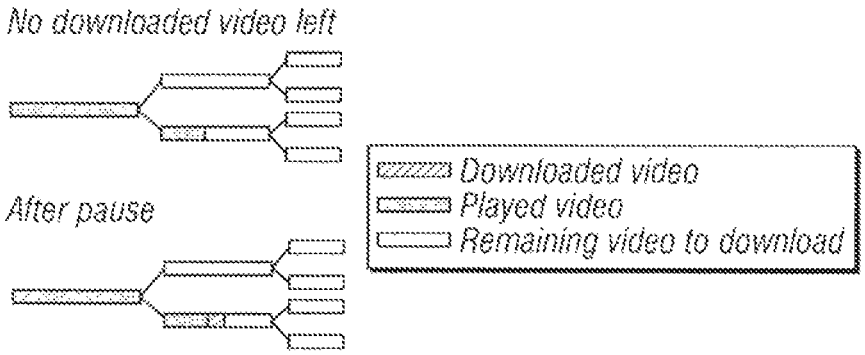


FIG. 6

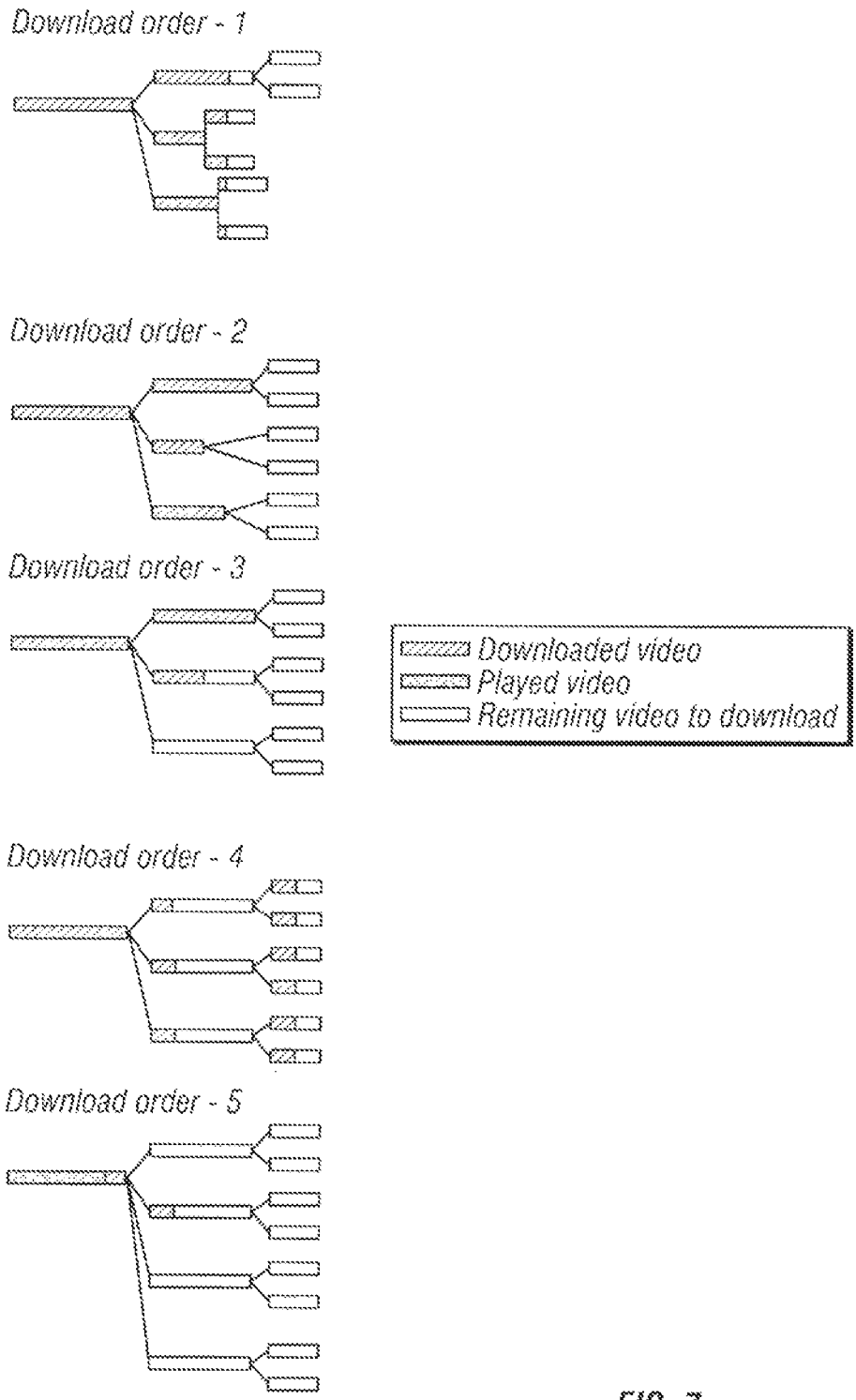

Download order - 1
Download order - 2
Download order - 3
Downloaded video
Played video
Remaining video to download
Download order - 4
Download order - 5

FIG. 7

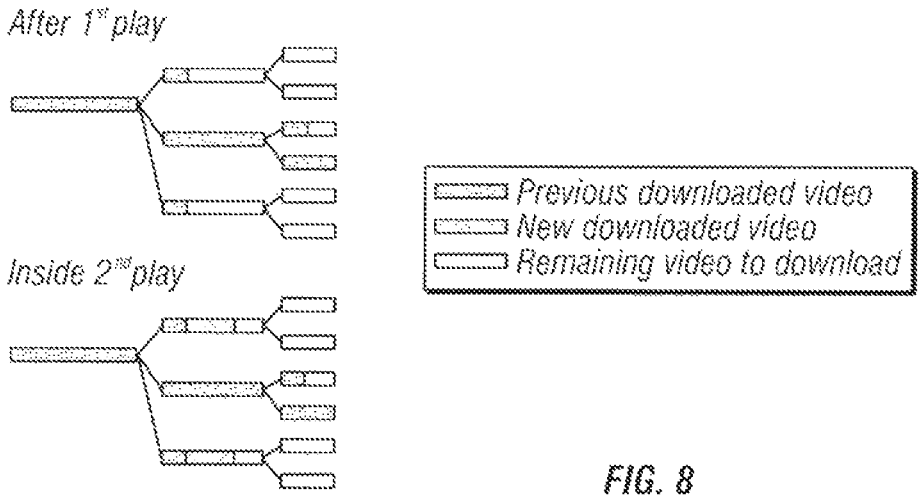


FIG. 8

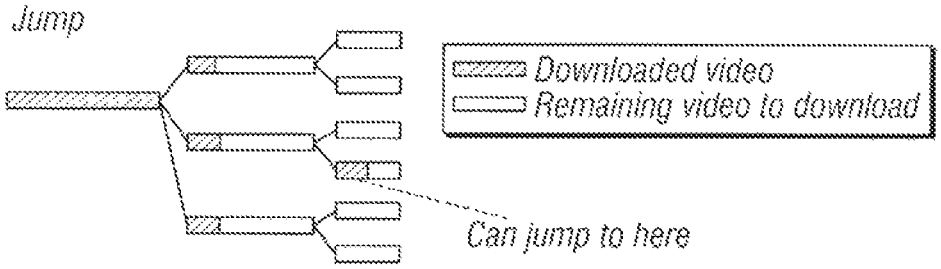


FIG. 9

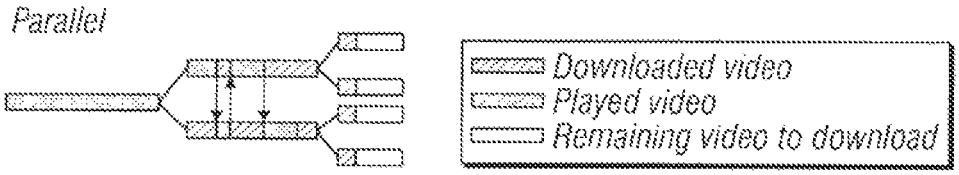


FIG. 10

SYSTEMS AND METHODS FOR LOADING MORE THAN ONE VIDEO CONTENT AT A TIME

CROSS-REFERENCE TO RELATED APPLICATION

This application is a continuation of U.S. patent application Ser. No. 13/437,164, filed on Apr. 2, 2012, and entitled "Systems and Methods for Loading More Than One Video Content at a Time," which is hereby incorporated by reference in its entirety.

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention is directed to systems and methods for loading video content, and more particularly to systems and methods that load more than one video at the same time.

2. Description of the Related Art

All web servers are capable of progressive download. This is merely the method of delivering a video file via HTTP to a browser on the client side. The process is similar to downloading a file from any website but the difference is that media players can begin to play the video while it is downloading rather than having to wait until the entire video has been downloaded.

When a video is being delivered via HTTP progressive download, typically one sees the buffer bar grow as the video downloads. One is not able to watch the video if the scrubber button is moved past the amount that has downloaded already. This makes it impossible to jump to the end of the video. If the site has a slow web server or limited bandwidth, or the end user is on a slow Internet connection, then the end user will notice buffering. Buffering occurs when the download can't stay ahead of the video playback. In this instance, the video will stop and only after the player will download an additional portion of the video will the playing resume. If the user pauses the video and allows the downloading of a large portion of the video, the user will likely watch the video uninterrupted.

In almost all progressive downloading, there will be a certain amount of preloading, which means that a certain amount of video data is loaded, before the start of playback. Some players will begin to play a video as soon as a small amount of data is received and some will wait until the entire file is downloaded before it plays (in order to prevent buffering).

Current methods load the entire video and just one video is provided. Only after the video is finished playing can another one be loaded.

There is a need to provide improved systems and methods for loading videos. There is a further need for loading more than one video at a time.

SUMMARY OF THE INVENTION

An object of the present invention is to provide systems and methods for improving the loading of videos.

Another object of the present invention is to provide systems and methods for loading more than one video at a time.

Yet another object of the present invention is to provide systems and methods where the several videos are loaded at the same time and played in seamless manner both for audio and video.

Yet another object of the present invention is to provide systems and methods where videos are loaded at the same time in a seamless manner and concatenated on client side.

A further object of the present invention is to provide systems and methods for loading videos where a plurality of videos is loaded, each having a plurality of segments that are standalone videos.

Still another object of the present invention is to provide systems and methods for loading videos that allow a user to create customized videos.

Another object of the present invention is to provide systems and methods that connect the loaded videos seamlessly on client side.

Another object of the present invention is to provide systems and methods for loading videos where a user can create a new customized video using selected segments of an original video.

Still another object of the present invention is to provide systems and methods to create customized videos from a video that is playing.

These and other objects of the present invention are achieved in a system for loading videos. An interactive video player is provided with a loader. A product configuration file in operation configures files for a user in creation of a custom video. External assets are configured for a design of an interactive layer of the video. The interactive video player in operation creates in real-time a custom video that includes a plurality of video segments.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** illustrates one embodiment of an overall system architecture of the present invention.

FIG. **2** is a flow chart illustrating one embodiment of downloading video segments and creating a custom video.

FIG. **3** is a diagram illustrating the differences between a regular video and a video of the present invention.

FIGS. **4**(*a*) and **4**(*b*) illustrate an embodiment of a sequence of downloading videos in an embodiment of the present invention.

FIG. **5** illustrates an embodiment of the present invention with two possible values for preloading time.

FIG. **6** illustrates an embodiment of the present invention where a loader downloads additional video segments to play and once a threshold is reached the video continues playing.

FIG. **7** illustrates an embodiment of the present invention illustrating that the loader can have priorities.

FIG. **8** illustrates an embodiment of the present invention with socket communication.

FIG. **9** illustrates an embodiment of the present invention with a jump feature.

FIG. **10** illustrates an embodiment of the present invention showing that the user interacts with videos in real time.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Definitions

Player

Player Engine

Loading Manager—Is in charge of downloading web-based videos according to a certain loading logic. Gets a tree as input from the Project Configuration Manager, and is notified when user/system choices are made by the Choice Manager (for instance, in order to stop loading unreachable videos once a selection has been made).

Video Parser—Analyzes and modifies the raw video content, for instance, in order to concatenate two separate videos into a single timeline.

Also in charge of inserting cue points into the video stream (for instance, junction event).

Video Bytes Appender—Takes different parsed video streams and appends their bytes into a single stream of bytes which is then fed into the video playback.

Video Playback—Plays back the video for the user to watch. Also notifies of events using cue points that were embedded within the video stream by the Video Parser.

Player Frontend/UI

Project Configuration Manager—Manages all project specific configurations. Gets as input an external configuration files. The Project Configuration Manager notifies the Loading Manager of the tree structure and web-based video files, sets the Choice Manager's choices and the External Assets Manager's configuration.

Choice Manager—Accepts junction notifications from the Video Playback component, thus showing a user selection GUI (selection buttons). Listens for User Interaction and notifies the Loading Manager that a selection has been made.

External Assets Manager—Is in charge of the loading and displaying of all project specific external assets (for instance, start screen, play button, transport bar, choice buttons, end screen and more).

External Resources (Web Based)

Videos—Web-hosted video files

Project Configuration File—A file that outlines the entire project's data, including the tree structure, video files, external assets and any other project specific data.

External Assets—External web-based files that make up the project's start screen, end screen, choice buttons, transport bar and other visual objects.

Other

User Interaction—Any input by user (such as mouse click, keyboard input, gesture etc.).

The present invention provides systems and their methods to load several videos at the same time. With the present invention, the system does not know which video will be played and relies on user selection. The connection between the videos is sufficiently seamless so there is no time to load a video and then wait for the next one to be loaded and there are no jumps or cuts on connection points between videos.

With the system of the present invention, a plurality of videos are loaded, each having a plurality of segments that are a standalone video. In one embodiment, a user downloads various segments and these are then selected as segments to be loaded. A custom download is created.

In one embodiment, the system **10** includes a pre-loader **12** which loads some video before the user can start playing. This can be done on every player, which as a non-limiting example can be you-tube, Netflix, vimeo, and the like. A production company **14** creates the segments of the video.

A video created by the system **10** is an interactive video that offers the user the opportunity to make a choice, as the video is playing (i.e., without pausing/stopping the video), that affects the course of the video in real-time. In one embodiment, the video is non-linear with many segments connected by branches. A tree like structure is created with several video segments that can appear at the same time point based on how a user chooses to engage with the video. The video player has to be smart to control the way the segments are downloaded in that the video player considers all possible valid continuations of the video, and makes sure the video will be played until the end on all possible scenarios. The player can include other interactive elements including but not limited to, links, pop-ups text, pictures, animation, other videos and the like.

Referring to FIG. **1**, the system **10** includes a player **16** with a project configuration manager **18**, choice manager **20**, external assets manager **22**, a loading manager **24**, a video parser **26**, a video bytes appender **28** and a video playback **30**. The player **16** communicates with videos **32**, a product configuration file **34**, user interaction and external assets **36**. The external assets are design assets used for the design of an interactive layer.

The user can create the video through the use of a user device coupled to the system, including but not limited to, a computer, cell phone, such as Apple's iPhone, other portable electronic devices, such as Apple's iPod Touches, Apple's iPads, and mobile devices based on Google's Android operating system, and any other portable electronic device that includes software, firmware, hardware, or a combination thereof that is capable of at least receiving the signal, decoding if needed, exchanging information with a transaction server to verify the buyer and/or seller's account information, conducting the transaction, and generating a receipt. Typical components of the user device may include but are not limited to persistent memories like flash ROM, random access memory like SRAM, a camera, a battery, LCD driver, a display, a cellular antenna, a speaker, a Bluetooth circuit, and WIFI circuitry, where the persistent memory may contain programs, applications, and/or an operating system.

Referring to the flow chart of FIG. **2**, a load start screen is used to preload video assets until a specific point. After pressing play the system plays a first segment, collect first user selection, then plays the second segment, collect the second user selection and plays the last segment. Following the first segment, an updated download is provided according to the user selection. Following the second segment, a download update is provided according to the user's selection, and so on.

The system **10** downloads a video and allows for the creation of multiple segments of video that can be linked together via branches. In certain embodiments, there can be virtually no limit to the number of segments and branches.

As a non-limiting example, a first segment can be the first ten seconds of a video. Two additional segments can be created for the next ten to twenty seconds, each coupled seamlessly to the first segment via an associated branch.

The downloading continues and the user selects the video it wishes to see. The system **10** then continues to download. In FIG. **3**, the four rectangles on the right are different remaining segments.

With the present invention, several segments of videos are downloaded at the same time because the system does not know which will be played.

The diagram in FIG. **3** below illustrates the differences between a regular video and one created by the present invention. In FIG. **3**, the regular video is represented as a played video, downloaded video and remaining video to be downloaded. With the video of the present invention, a downloaded video from a played video is the first segment. A decision point exists after the first segment where it branches into two second segments with remaining video to download. The remaining video to download branches from the respective two second segments.

When loading segments for the video, the system loads all possible segments that can be chosen by the user. These segments are either loaded, or their loading is initiated, prior to the connection point in order to ensure that the next segment is ready with all of its possible variations.

In one embodiment, the loader **12** loads all the possible paths and disregards paths that the user cannot reach once a particular video segment is selected and viewed.

As a non-limiting example, illustrated in FIGS. **4**(*a*) and **4**(*b*), a sequence of loading is illustrated. Segment **1** is loaded. Segments **2***a*, **2***b* are then loaded. The user selects option **2***b*. Segment **2***a* is then disregarded with its entire branch and segment **2***b* is loaded. Segments **3***c* and **3***d* are also loaded.

If the system **10** already started loading segment **2***a*, and the user then selects option **2***b*, the system **10** stops loading **2***a*, as illustrated in FIG. **4**(*b*).

In one embodiment, video segments are pre-loaded in order to give the loader a head start and ensure a seamless video viewing experience. The pre-loader **12** knows how to dynamically calculate the internet speed for an individual user and how much to preload at the start before the video begins to play in order for the video to be played without the video stopping and resuming only after it loads more content, e.g., buffering. The system **10** reduces the chance for buffering so the video won't stop. The preload can be configured to any amount of time needed by the video creator. As a non-limiting example, with a very fast internet connection, the system and method of the present invention can preload a very small amount of video before playing and knows that the system manages loading all the rest of the videos before playing them on every possible branch.

When the system **10** knows that an internet connection is slow, more video time is loaded before playing otherwise the player tries to play video that was not downloaded yet (buffering).

FIG. **5** illustrates an embodiment with two possible values for preloading time.

The loader **12** optimizes the net bandwidth utilization to download additional video content according to an individual user's bandwidth constraints in order to continue downloading video. As a non-limiting example, this can occur after the preload and before a user presses play, or when the video is paused. Optimization is achieved by continuing loading during this "dead time" before the user pressed play so more video time is loaded and the chance for buffering is reduced.

After the video starts to play, the loader **12** continues downloading all of the video segments it can until it has nothing left to download. The loader **12** has a map of all existing videos, knows the length of each segment and the possible options to continue. The loader **12** downloads all possible videos until reaching the last segment on every possible branch.

The downloading speed can be affected by many factors including but not limited to, computer status, service provider, video hosting, programs running in the background, net congestion and the like. If, however, during the video, the player played all of the segments of the video that were successfully downloaded, and the loader still didn't manage to load more content to play or if the amount of downloaded video that remains is less than a minimum threshold, the system **10** will enter a buffering state. The video playback pauses, the loader **12** downloads additional video segments to play, and after a certain threshold is reached, the video continues playing, as illustrated in FIG. **6**.

At the end of the played video, the loader **12** can download more video segments that were effectively discarded during the last play. These video segments can be used for the next play. This provides for optimization. When the system **10** loads more segments in current playback it has less to load on the next play and this significantly reduces the next loading time.

Referring to FIG. **7**, in one embodiment of downloading order, the loader **12** can choose how to download the video segments and in what order. In one embodiment, the same amount of play time is loaded for each segment. This can be the default setting. In another embodiment, all videos are loaded until the end of certain segments and can be used when segments have different lengths for each variation. In another embodiment, segments are loaded by order of importance or popularity. This can be, by way of a non-limiting example, knowing which branch is more likely to be played. In another embodiment, only the start of each segment or selected segments are loaded. In another embodiment, loading is done only after a specific segment is selected, such as, by the non-limiting example, if there is a very large amount of variation for the same segment.

The loader **12** has priorities for every segment and decides what to load according to the priorities and the user bandwidth. The first priority is what is currently playing. The second priority is the next segment and prioritization between all of the subsequent segments according to popularity or importance. The third priority is the segments beyond the next segment, e.g, subsequent segments, or segments that can be jumped to.

The loader **12** can choose not to download all the segments but only a subset of them. Each segment has some kind of "Is_loadable" variable that indicates if the segment should be loaded or not.

This can occur in several scenarios. In one, if options are effectively chosen in advance, as a non-limiting example if a user enters its gender in the beginning, and the impact of that option only appears later during the video play. Another scenario is if the system **10** makes a choice according to information that was not directly inputted by the user, such as IP address, social network ID, additional identifiers, and the like. Additionally, the system **10** can load only the most likely segments to be played and loads them unless the user chooses differently In terms of which segments to load.

In another embodiment, the loader **12** can have other logic that controls the process differently, where logic is another part of the player that can control the loader **12** by defining the "Is_loadable" variable for each segment. The system **10** can allow the user to play the video with certain segments on the first viewing and different segments on the second viewing such that the loader **12** acts accordingly.

In another embodiment, another scenario is after a user interacts with a video and makes certain choices associated with the video. In this scenario the user can share his/her passive version of the video with others (i.e., a third party would not be able to make choices and would only watch the video as the choices were selected by the initial user). In this "passive mode," the video plays only the selected sequence of video segments and the loader **12** only downloads those segments. However, when the loader **12** completes the required downloads, it can start downloading the other video segments in the background, while the video is playing, for potential active mode by the new user.

The system **10** can also provide support for download pause and resume feature. This feature set includes a way to pause a download, save the partial data that was already downloaded, and, at a later time, continue downloading from the same point on all branches.

The system **10** allows for download pause and resume. In one embodiment, this is achieved by adding a "Range" field to an HTTP request. However, Flash does not allow this field to be added to an HTTP request due to security restrictions. The HTTP "Range" field, as defined by RFC 2616, section 14.35, allows the retrieval of a sub-range of bytes of the requested file, thus allowing a download of partial data, or resuming of a partially downloaded file from the place we left off.

In order to get around this security restriction, the system **10** implements a more low-level form of communication instead of using the standard feature set that Flash provides.

Because the standard Adobe Flash API for HTTP requests does not allow addition of the "Range" header to an HTTP request, the system **10** avoids using the standard API and implements its own HTTP client.

Referring to FIG. **8**, in one embodiment, socket communication is used to implement HTTP communication and partial downloads are manually saved to an array of bytes, also known as ByteArray. With this method, the system **10** pauses a download, saves partial data, and later sends an HTTP request with a "Range" field, so that the download restarts at the correct offset. The new incoming data is appended in the ByteArray. This is useful to system **10** player's loading logic because one can now pause video stream downloads which are not currently needed, but may be required later in playback or replay, without re-downloading the entire file. This increases efficiency and avoids redundant network traffic and processing. In another embodiment, the system re-downloads the entire dropped file from the beginning and dumps the partial data that was downloaded. However, this can be wasteful.

In various embodiments, the system **10** provides a jump feature, illustrated in FIG. **9**. In some videos the user can jump to a different part of the tree. As a non-limiting example, if the character dies or the user wishes to skip a part, then the loader **12** downloads the destination part in order to prevent buffering on such a jump. Because every segment has its own priority, the loader **12** can have a high priority for a distant segment. The user then starts loading before loading closer segments. In order to ensure that if the user will decide to jump to this segment it will already be loaded.

Referring now to FIG. **10**, in some videos the user can move back and forth between two or more branches. This means that the loader **12** does not pause the loading of the segments that were not selected since they can still be used. Back and forth movement is achieved in the same manner. The options always appear on the screen and the two videos are played simultaneously, and the loader **12** needs to download both parallel segments.

In various embodiments, the user interacts with videos of the present invention in real-time while the video is playing. This user can interact with the video with any known variety of mechanisms, including but not limited to, mouse clicks, mouse movement, eye movement, keyboard and the like.

In various embodiments, the segments, even if offered at a single decision point, can have different lengths. As a non-limiting example, a user can choose between segment A**1** or A**2** where A**1** and A**2** have different lengths. A decision point is not limited to a binary option. The number of decision point options can be larger and a segment can continue to any number of presented options.

With the present invention, the interactive nature of a video impacts the video's content and the user can control the video's path. With the present invention, every option changes something on the video content. With the present invention, moving from one video segment to the next can take some time and can also be immediate upon clicking the desired option, with virtually no delay from the time a decision is made to the playing of the associated video segment. In most instances, the options are selected by the user. However, in some circumstances the system **10** can select the options for the user. As non-limiting examples this can be by IP, physical location, weather, user id and the like. The video can also include personal data inside the video.

The player can be used on any device that displays video and enables user engagement including but not limited to, personal computers, tablets, cellular phones, music players and the like.

The player can includes an interactive layer on top of the video. This layer presents output to the user, such as option buttons, a clock and the like, and collects the input from the user. Interactive elements on the interactive layer can be different for each video segment to provide that when a new segment is loaded and played the player can add the appropriate interactive elements to it.

Expected variations or differences in the results are contemplated in accordance with the objects and practices of the present invention. It is intended, therefore, that the invention be defined by the scope of the claims which follow and that such claims be interpreted as broadly as is reasonable.

The invention claimed is:

1. A system for presenting media content, the media content comprising a plurality of video segments, the system comprising:

a video loading manager for determining a first subset of the video segments to download;

a video playback engine for presenting a first one of the video segments; and

a choice manager for receiving a selection of a second video segment for playback,

wherein the video loading manager is further for determining a second subset of the video segments to download based at least in part on the selection of the second video segment, and

wherein the plurality of video segments is organized according to a tree structure comprising decision points at which one of the video segments may be selected for downloading and viewing such that only certain of the video segments remain potentially viewable.

2. The system of claim **1**, wherein the video loading manager is further for downloading at least a portion of each of the first subset of video segments.

3. The system of claim **2**, wherein the video loading manager is further for stopping the download of at least one video segment in the first subset based at least in part on the selection of the second video segment.

4. The system of claim **2**, wherein the video loading manager is further for downloading at least a portion of each of the second subset of video segments.

5. The system of claim **1**, wherein the determination of the first subset of video segments is based at least in part on a download priority.

6. The system of claim **5**, wherein the download priority is based at least in part on a popularity of the first subset of video segments among previous viewers thereof.

7. The system of claim **5**, wherein the download priority is based on one or more user characteristics.

8. The system of claim **7**, wherein the user characteristics comprise one or more of a gender, an age, a location, and a prior viewing history.

9. The system of claim **5**, wherein the video loading manager is further for initiating download of the first subset of video segments according to the download priority and prior to completion of playback of the first video segment.

10. The system of claim **1**, wherein the determination of the second subset of video segments is further based at least in part on a download priority.

11. The system of claim **10**, wherein the download priority is based at least in part on a location of the second video segment in the tree structure and which subsequent segments are connected to the second video segment within the tree structure.

12. A method of presenting media content, the media content comprising a plurality of video segments, the method comprising:

determining a first subset of the video segments to download;

presenting a first one of the video segments;

receiving a selection of a second video segment for playback; and

determining a second subset of the video segments to download based at least in part on the selection of the second video segment,

wherein the plurality of video segments is organized according to a tree structure comprising decision points at which one of the video segments may be selected for downloading and viewing such that only certain of the video segments remain potentially viewable.

13. The method of claim **12**, further comprising downloading at least a portion of each of the first subset of video segments.

14. The method of claim **13**, further comprising stopping the download of at least one video segment in the first subset based at least in part on the selection of the second video segment.

15. The method of claim **12**, wherein the determination of the first subset of video segments is based at least in part on a download priority.

16. The method of claim **15**, wherein the download priority is based at least in part on a popularity of the first subset of video segments among previous viewers thereof.

17. The method of claim **15**, wherein the download priority is based on one or more user characteristics.

18. The method of claim **12**, wherein the determination of the second subset of video segments is further based at least in part on a download priority.

19. The method of claim **18**, wherein the download priority is based at least in part on a location of the second video segment in the tree structure and which subsequent segments are connected to the second video segment within the tree structure.

* * * * *