H2O Clouds: Issues, Challenges and Solutions

Shahram Ghandeharizadeh
Computer Science Department
University of Southern California
Los Angeles, CA 90089, USA

Abstract

Home-to-Home Online (H2O) devices use their wireless communication to complement existing wired infrastructure such as Internet and provide data services to individual households. Those within the same radio range form an ad-hoc network of peers that collaborate to increase availability of data. This data might range from educational and entertainment content to personal libraries uploaded by each household. This paper describes the H2O framework and challenges posed by its design. It provides a summary of our in-progress work to date and outlines future research directions.

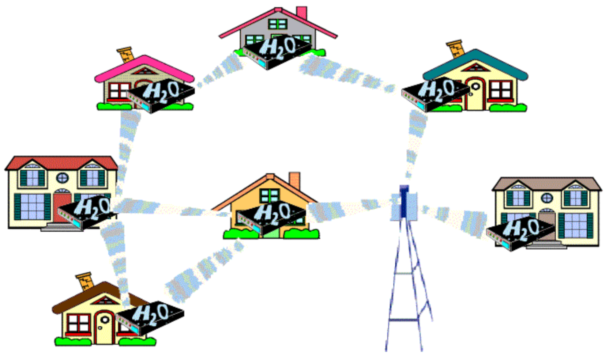

Figure 1: Home-to-home on-line devices streaming continuous media.

1 Introduction

Advances in computer processing, storage performance and high speed wireless communications have made it feasible to consider peer-to-peer network of economical devices that provide access to a large volume of data. Intel, for example, offers a small device that consists of a 500 MHz processor and a wireless component that operates in the 5 GHz spectrum, offering transmission rates in the order of tens of Megabits per second, Mbps. The cost of this device is approximately $85. Similar to desk top personal computers, one may extend this device with mass storage. An application of these devices is to stream continuous media, e.g., audio clips, movies, news clips, etc., for home entertainment systems. Home-to-Home Online (H2O) devices may collaborate to deliver multimedia content to an actively displaying H2O device. One example deployment of H2O might be that of Figure 1 with a cellular base station serving as its interface to a wired infrastructure such as Internet. A household may store its personal video library on a H2O cloud. This would make the library widely available to enable a user to retrieve their content anywhere, e.g., at a friend's home. (This flexibility is a building component of complex systems such as Memex [2] and MyLifeBits [4].) The system might encrypt the content to either protect it from un-authorized access, i.e., authentication, or implement a business model for generating revenues.

A H2O cloud complements the existing wired solutions based on xDSL technology or cable networks [6, 16]. It continues to rely on wired centralized servers, e.g., Mitra [12], and cache servers [13, 15] as the main repository for continuous media. Each H2O device pre-stages data in order to bring it closer to a consuming client, minimizing the number of accesses to the remote servers. In addition to displaying a clip, a H2O device (say H2O$_j$) might collaborate with a displaying H2O device (say H2O$_d$) by either producing data for display by H2O$_d$, routing data from another H2O device to either the same H2O$_d$ or a different one, or both[1]. Ideally, a displaying H2O device should find its referenced clip either locally or on its neighboring H2O devices. This reduces the amount of delay encountered by a user to display a clip. (This delay is termed startup latency.) In addition, it frees both network and remote server bandwidth to service other H2O contents, i.e., maximizes throughput.

Remote centralized servers maintain a copy of all published data and provide administrative services for H2O clouds, e.g., billing. Moreover, in the presence of H2O node removals, these servers minimize the possibility of permanent data loss. H2O devices continue to service requests in the absence of a centralized server as long as a replica of the referenced data item is available in the H2O cloud.

Display of continuous media in a H2O cloud raises interesting research challenges. These challenges are outlined in Section 2. Section 3 describes alternative techniques to place data across devices in order to display continuous media. Brief conclusions and future research directions are provided in Section 4.

[1]One scenario that requires H2O$_j$ to play this dual role with H2O$_d$ is when the blocks of the referenced clip are dispersed across the network.

2 Challenges

This section describes the following challenges that must be addressed by a H2O system design: privacy of users, authorized access to content, effective user interfaces, hiccup-free displays, admission control, and dynamic data delivery scheduling techniques. This order is the outline of this section.

Privacy of users is an important challenge faced by a commercial deployment of H2O devices. This challenge is two folds: 1) preventing improper profiling and their use, and 2) preventing un-authorized display of private content. Consider each in turn. With the first, a H2O cloud may gather profiles of each household and their viewing pattern. As described in Sections 1 and 3, these profiles are useful for controlling placement of data in anticipation of their future use. At the same time, profiles are useful for target advertising. Not all target advertising is bad. For example, a household may elect to participate in target advertising in return for a service, e.g., video e-mail. Another example is a household that desires a profile describing what percentage of their time is spent viewing adult content and during what time of day. They may desire this profile in order to discontinue programming not suitable for children. While this appeals as a service, when gathered without the household's knowledge, this profiling might be perceived as a violation of privacy.

With the second, each household may use the H2O infrastructure to store and retrieve a variety of content, ranging from permanent to time-shifted. Permanent content pertains to personal libraries. This content might be diverse ranging from videotaping of a wedding to a family reunion. Time shifted content is continuous recording of a camcorder attached to a H2O device. This recording might be used for surveillance of an area, e.g., monitoring an infant's room or the house when away for travel, etc. A time-shifted session might be configured to maintain recording for the past δ time units, e.g., δ=30 minutes. Moreover, a user may want to display this content remotely. Viewing of both content types must be limited to those authorized members of the household. The deployed policies and mechanisms to address this issue must be flexible enough to empower a household member to (1) view content using a display (located anywhere) with ability to connect to a H2O cloud, and (2) grant (and revoke) permission to others who may want to display a clip.

A H2O cloud may consists of a variety of content ranging from entertainment to educational. Moreover, households may publish information for viewing by every one. This might be similar to web pages maintained by businesses and individuals on Internet. We speculate the key difference to be wide spread use of audio and video clips. Searching for relevant content is a challenge in this domain. As an example, consider a user searching for a music clip without remembering its key properties, e.g., its title, performer's name, etc. The user recalls portions of the clip's lyrics and tone. Techniques that enable this user to search and locate the relevant clip are paramount to H2O's effective deployment. Recent research has tried to address this challenge by using humming of a music clip to facilitate its retrieval [18]. Content-based retrieval has also been an intense area of research for still images and video clips [3]. Adaptation and wide-scale deployment of these techniques remains a future research direction.

Effective user interfaces that enable a household to utilize H2O devices to their full potential is another challenge. These interfaces must be simple and intuitive. A keyboard to enable users to input their user-id and password is most likely the wrong interface to either facilitate privacy of content or identify the household to bill for a viewing. Novel interfaces are required to enable an individual to select and preview content, publish new content for either private use or sharing with other users, etc.

The network bandwidth and storage capacity of each H2O device must be managed intelligently in order to support display of continuous media, audio and video clips. Continuous media consist of a sequence of quanta, either audio samples or video frames, that convey meaning when presented at a pre-specified rate [5, 10]. Once the display is initiated, if the data is delivered below this rate then a display might suffer from frequent disruptions and delays, termed hiccups. In addition to a hiccup-free display, a display must be initiated in a timely manner. With a two hour movie, a user might tolerate a delay in the order of minutes watching advertisements. However, the same does not hold true for a 4 minute audio-clip. A technique that downloads an entire clip prior to initiating its display might result in unacceptable delays. For example, to download a 4 minute audio clip with a 126 Kbps bandwidth requirement using a 4 Mbps connection would require almost 30 seconds. To download a 2 hour movie with a 4 Mbps bandwidth requirement using a 10 Mbps connection would require almost 48 minutes. One way to minimize these delays is to overlap the display of a clip with its remote retrieval. This can be conceptualized as a displaying H2O device (H2O$_d$) that consumes data at the display rate of the clip while the H2O cloud is producing data for this device at either the same or a different rate. Pipelining of data must be done in a manner to prevent the possibility of H2O$_d$ starving for data. With multiple simultaneous requests, each request might be activated in three possible ways. general approaches. The first, termed batching, delays requests until they can be merged with other requests for the same clip. The second, termed buffer sharing, processes a request on-demand by initiating a stream as soon as possible and detect opportunities to merge multiple streams into one. The third is a hybrid of these two approaches.

Consider the second approach to processing requests. When the available bandwidth between a producing H2O (H2O$_p$) and a displaying H2O (H2O$_d$) is less than the bandwidth required to display a clip X, H2O$_d$ may prefetch enough data to prevent data starvation. Assuming S_C denotes X's size, the amount of prefetch data is [7]: $S_P = S_C - \lfloor \frac{B_{Link}}{B_{Display}} \times S_C \rfloor$, where B_{Link} is the link with lowest bandwidth that participates in the path between H2O$_p$ and H2O$_d$, and $B_{Display}$ is the bandwidth required to display the clip. The time required to stage S_P bytes of data at H2O$_d$ dictates the incurred startup latency. For example, if a 2 hour movie requires 4 Mbps for its continuous display ($B_{Display}$=4 Mbps) and B_{Link} is 3 Mbps, then H2O$_d$ must prefetch 900 MBytes of data. The startup latency is dictated by the amount of time required to stage this much data on H2O$_d$. If 3 Mbps is dedicated for this stagging, the startup latency observed by H2O$_d$ would be 40 minutes. Either all 900 MBytes or a fraction of it might be pre-staged on H2O$_d$ to reduce this startup latency.

A H2O cloud requires a decentralized admission control mechanism that admits requests whose bandwidth requirements can be satisfied. This component may control choice of titles available to a H2O device (say H2O$_1$) based on how much network bandwidth is available between H2O$_1$ and other H2O devices containing content. When these paths offer abundant bandwidth, H2O$_1$ may offer a household a wide selection of titles. As this bandwidth becomes scarce, the choice of titles might reduce. When the network bandwidth to H2O$_1$ is completely exhausted, it offers the household members only those titles resident on its local storage. This prevents a scenario where a user repeatedly selects a title only to be rejected by the admission control component due to insufficient network bandwidth. Techniques that enable H2O$_1$ to predict availability of bandwidth are useful because H2O$_1$ might make certain titles available based on such predictions and the expected startup latency tolerated by the display of these clips.

There is a tradeoff between storage capacity of each H2O device and wireless network bandwidth available between different H2O devices. If the storage capacity of each H2O device was unlimited then the entire repository would fit on each H2O device, minimizing demand for network bandwidth. Moreover, new content would be multi-cast for permanent storage at each H2O device. Similarly, if the wireless network of H2O devices offered infinite bandwidth then each H2O device would store only the first few blocks of a clip on a permanent basis and retrieve the rest of a clip remotely. (By pre-staging the first few blocks, the startup latency of each clip is minimized [7, 9]; see the following paragraph for details.) Unfortunately, each H2O device is configured with both a finite amount of storage and a wireless network card that offers a fixed maximum bandwidth. Thus, a H2O cloud must intelligently trade storage of each H2O device for its network bandwidth with the objective to maximize the number of hiccup free displays, while minimizing the startup latency incurred by each display. This tradeoff is the focus of Section 3.

The characteristics of both continuous media and H2O clouds offer opportunities to address the identified challenges. First, continuous media clips are typically accessed sequentially, e.g., a video clip is typically displayed from its beginning. As described in Section 3, this is an opportunity because it argues that the first few blocks of a clip are needed more urgently than its last few blocks. Thus, in order to minimize startup latency, one may replicate a clip's first few blocks more frequently [9]. Second, a H2O cloud consists of devices that are stationary most of the time. This is an opportunity because it enables a H2O device to construct a topological map of network connectivity for its neighbors. This map is useful to identify potential bottleneck links in a cloud, enabling both the replication and admission control components of each H2O device to render intelligent decisions to minimize bottlenecks. Depending on the available bandwidth, a H2O device may stage a larger fraction of popular clips in order to enhance its chances of displaying a clip.

One approach to save network bandwidth is to merge multiple streams referencing the same clip into one. To illustrate, consider two different H2O devices (H2O$_1$ at time T_1, and H2O$_2$ at time T_2) that initiate the display of the same clip (say clip X) at different times. Moreover, assume H2O$_2$ routes the blocks of X from a producing H2O (H2O$_p$) to H2O$_1$. In this case, H2O$_2$ may start to store the blocks of X currently being forwarded to H2O$_1$ for its own future use. Simultaneously, it may request the initial blocks of X from either H2O$_p$ or another candidate H2O device. This is feasible as long as H2O$_2$ has sufficient network bandwidth to receive two streams. An alternative scenario is when H2O$_2$ is neighbor to another H2O device (say H2O$_3$) that is currently routing clip X to H2O$_1$. Techniques that enable H2O$_2$ to detect H2O$_3$'s activities and request H2O$_3$ to save its stream of data for future transmission to H2O$_2$ might save network bandwidth. This might be detected in either an eager or a lazy manner. With eager, H2O$_2$ and H2O$_3$ might detect one another prior to initiating a stream to H2O$_2$. With lazy, H2O$_2$ and H2O$_3$ detect one another over time, terminating redundant data streams. Detection of candidate merges might be performed by H2O$_2$, H2O$_p$, or intermediaries (including H2O$_3$). The H2O cloud must determine when such a saving is beneficial. Moreover, if H2O$_3$'s storage capacity has been exhausted then it must employ techniques to decide whether to either accept or reject H2O$_2$'s request. If it accepts H2O$_2$'s request then it must replace existing content. Once again, techniques are needed to decide which clips should be swapped out of H2O$_3$'s storage. Given a few popular titles referenced most frequently by many clients at the same time, there are other opportunities to design smart techniques that support many simultaneous displays while freeing bandwidth for use by other H2O devices.

3 Data Placement and Replication

Continuous media is almost always read-only data. This enables the H2O devices to replicate a clip and place its copies across different devices in order to maximize the number of simultaneous displaying H2O clients in the presence of limited network bandwidth. It is not necessary for the system to replicate all titles because the distribution of access to the clips is most likely skewed where a few popular titles are referenced most frequently. By replicating these clips most often, the network bandwidth is freed to service those requests that reference less popular clips. A clip might be replicated at the granularity of either a clip or a block. In the following, we provide an overview of these two alternatives. Next, we make a case for a hybrid approach that employs both techniques by analyzing the available network bandwidths and storage capacities.

With clip-level replication, different copies of a clip are assigned to different H2O devices. This raises two key questions: First, how to compute the number of replicas for each clip? Second, which H2O devices should contain a clip replica? In the following, we discuss the first question. (The second is discussed in the following paragraphs.) With the first, the number of replicas can be computed using the resources consumed by the clip and its frequency of access [8]. Consumed resources might be system's storage occupied by the clip, i.e., clip size, network bandwidth required to display the clip, clip display time that quantifies how long its display consumes network resources, etc. In [8], we analyzed these alternatives. One technique is shown to outperform others by maximizing number of simultaneous displays supported by the system (with a string topology). This technique computes the number of replicas for a clip proportional to (1) its bandwidth relative to the bandwidth of all clips, and (2) the square-root of the clip access frequency.

With block-level replication, a clip is partitioned into fix-sized blocks. The system may replicate the first few blocks of a clip more frequently based on the intuition that they are needed more urgently to minimize startup latency. In [9], we examine this technique while assuming a fixed delay to transfer a block from one H2O device to another. We derived analytical models to approximate the amount of required storage for three different H2O topologies: string, grid and graph. In addition, we described a decentralized algorithm, termed Timer, to determine both the number of replicas for each block and their placement across the nodes. A key characteristic of Timer is its uniform utilization of storage provided by H2O devices (assuming all devices offer the same amount of storage). Timer is shown to offer more than 95% savings in required storage when compared with a full replication of the repository on each node. A naive application of Timer to clip replication is to assume each clip is one block in size. This is naive because Timer replicates the first block of a clip on each node, resulting in full replication of the entire repository on each node.

There exists a spectrum of techniques with clip and block replication constituting its two ends. The suitability of a technique is dependent on the requirements of the target application and the characteristics of a H2O cloud. We speculate the feasibility of a flexible technique with adjustable parameters to emulate the different algorithms in this spectrum. This technique is desirable for two reasons. First, it would replicate clips of different applications in a different manner based on each application's requirements. Second, it can be invoked in the presence of changes to both the H2O cloud (e.g., node removals and insertions), and the database (e.g., insertion of new media types with different storage and bandwidth demands). The details of this technique might be as follows. It conceptualizes a H2O cloud as a graph and identifies those critical wireless links that may result in formation of queues. These links are termed candidate bottleneck links. They partition the graph into several sub-graphs. Across sub-graphs, this new technique would replicate a larger fraction of each clip (a higher number of blocks), degenerating into clip replication when bandwidth is severely limited. In each subgraph, the fraction of each clip replicated on each node is a function of (1) the characteristics of H2O devices in the subgraph, e.g., average link bandwidth, storage capacity, etc., (2) database characteristics, e.g., size of repository, average bandwidth required to display popular clips, etc., and (3) an application's requirements, e.g., tolerable startup latency, desired cost per stream, etc.

4 Summary

With H2O clouds, the connectivity of the wireless H2O devices are dictated by its physical surrounding and the geographical distance between devices. We have conducted extensive experiments using 802.11a cards offering advertised bandwidth of 54 Mbps and observed bandwidths ranging from 2 to 30 Mbps. When many such devices are in the same radio range and attempt to transmit data simultaneously, we observe a linear decrease in bandwidth allocated to each connection. This allocation is typically unfair with each connection observing some loss. The network behavior becomes more complex with geographically dispersed H2O devices [1, 17]. It is perhaps impossible to enumerate all possible interactions between H2O peers, motivating adaptable frameworks that monitor their environment and adjust to its characteristics. One such framework was outlined in Section 3 with focus on placement of continuous media. Another framework more suitable for exchange of XML data is NAM [11] which trades CPU processing for network bandwidth.

Content provides recognize the importance of H2O clouds as the wave of future. For example, [14] investigates the recent decline of the music market and quantifies its cost structure. It explores alternative business models for delivery of digital music with several applying to H2O clouds. Such studies underline the timeliness of H2O clouds and the importance of technical community to develop practical solutions that address its challenges.

References

[1] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz. A Comparison of Mechanisms for Improving TCP Performance over Wireless Links. *IEEE/ACM Transactions on Networking*, 5(6):756–769, 1997.

[2] V. Bush. As We May Think. *The Atlantic Monthly*, 176(1):101–108, July 1945.

[3] B. Furht. *Handbook of Multimedia Computing*. CRC Press and IEEE Press, ISBN 084931824, 1999.

[4] J. Gemmell, B. Gordon, R. Lueder, S. Drucker, and C. Wong. MyLifeBits: Fullfilling the Memex Vision. In *ACM Multimedia*, December 2002.

[5] J. Gemmell, H. M. Vin, D. D. Kandlur, P. V. Rangan, and L. A. Rowe. Multimedia Storage Servers: A Tutorial. *IEEE Computer*, 28(5):40–49, 1995.

[6] S. Ghandeharizadeh. Cable Networks and Distributed Video Repositories. In *Proceedings of the Cable 1997*, March 1997.

[7] S. Ghandeharizadeh, A. Dashti, and C. Shahabi. Pipelining Mechanism to Minimize the Latency Time in Hierarchical Multimedia Storage Managers. *Computer Communications*, 18(3):38–45, March 1995.

[8] S. Ghandeharizadeh and T. Helmi. An Evaluation of Alternative Continuous Media Replication Techniques in Wireless Peer-to-Peer Networks. In *Third International ACM Workshop on Data Engineering for Wireless and Mobile Access (MobiDE, in conjunction with MobiCom'03)*, September 2003.

[9] S. Ghandeharizadeh, B. Krishnamachari, and S. Song. Placement of Continuous Media in Wireless Peer-to-Peer Networks. *IEEE Transactions on Multimedia*, April 2004.

[10] S. Ghandeharizadeh and R. Muntz. Design and Implementation of Scalable Continuous Media Servers. *Parallel Computing*, 24(1):91–122, May 1998.

[11] S. Ghandeharizadeh, C. Papadopoulos, P. Pol, and R. Zhou. NAM: A Network Adaptable Middleware to Enhance Response Time of Web Services. In *Eleventh IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, October 2003.

[12] S. Ghandeharizadeh, R. Zimmermann, W. Shi, R. Rejaie, D. Ierardi, and T. Li. Mitra: A Scalable Continuous Media Server. *Multimedia Tools and Applications Journal*, 5(1):79–108, July 1997.

[13] M. R. Korupolu, C. G. Plaxton, and R. Rajaraman. Placement Algorithms for Hierarchical Cooperative Caching. In *SODA: ACM-SIAM Symposium on Discrete Algorithms (A Conference on Theoretical and Experimental Analysis of Discrete Algorithms)*, 1999.

[14] P. Premkumar. Alternative Distribution Strategies for Digital Music. *Communications of the ACM*, September 2003.

[15] L. Qiu, V. N. Padmanabhan, and G. M. Voelker. On the Placement of Web Server Replicas. In *IEEE Infocom*, 2001.

[16] C. Shahabi and F. Banaei-Kashani. Decentralized Resource Management for a Distributed Continuous Media Server. *IEEE Transactions on Parallel and Distributed Systems*, 13(6):455–465, June 2002.

[17] G. Xylomenos and G. C. Polyzos. TCP and UDP Performance over a Wireless LAN. In *INFOCOM (2)*, pages 439–446, 1999.

[18] Y. Zhu and D. Shasha. Warping Indexes with Envelope Transforms for Query by Humming. In *ACM SIGMOD*, June 2003.