computers are bad

You are receiving this facsimile because you signed up for fax delivery of this newsletter. To stop delivery, contact Computers Are Bad by email or fax.

https://computer.rip - me@computer.rip - fax: +1 (505) 926-5492

2021-11-06 smart audio for the smart home

Sonos offers a popular line of WiFi-based networked speakers that are essentially a consumer distributed audio system. It's a (relatively) affordable and easy to use spin on the network-based audio rendering systems already common in large building background music/PA and, increasingly, entertainment venue and theatrical sound reinforcement. The core of it is this: instead of distributing audio from one amplifier position to multiple speakers over expensive and lossy (in commercial contexts) or difficult to install (in consumer contexts) speaker-level audio cables, audio is distributed over an IP network to a small amplifier colocated with each speaker [1].

This isn't a new concept, although Sonos had to invest considerable effort in getting it to function reliably (without noticeable synchronization issues) over unreliable and highly variable consumer WiFi networks. From the perspective of commercial audio, it's just a more consumer-friendly version of Dante or Q-LAN. From the perspective of consumer audio, it's either revolutionary, or the sad specter of two decades of effort in network-enabled consumer AV systems. After all the work that was done, and all the technologies that could have succeeded, Sonos is what we ended up with: a feature-incomplete, walled-garden knockoff of DLNA.

Yes, I'm being unfair to Sonos. The biggest problem that Sonos figured out how to solve, precise and reliable synchronization of audio renderers without special network facilities like PTP, isn't one that DLNA attempted to address. And it remains a hard problem today; even my brand-new bluetooth earbuds regularly experience desynchronization problems when my phone's mediocre Bluetooth stack gets behind (maybe my ailing phone is more to blame than theoretical complexity).

But every time I really look at today's home media streaming and management landscape, which is largely dominated by Sonos, Apple's AirPlay, and Google's ChromeCast, it's very hard not to see it as a shadow of DLNA.

So what is DLNA, and why did it fail? In general, why is it that all of the home network AV efforts of the late '90s through the '00s amounted to nothing, and were replaced by thin, vendor-locked "cast" protocols?

The answer, of course, is Microsoft and Capitalism (rarely have there been two more ominous bedfellows). But let's get there the long way, starting with the evolving home media landscape of the late '90s.

The compact disc, or CD, came into widespread popularity in the late '80s and represented a huge change in music distribution. Besides the low cost, small size, and excellent fidelity of CDs, they were the first widespread consumer audio recording

format that was digital. For the first time it was, in principle, possible to create an exact digital copy of CD. Once written to another storage device, the CD could be handled as computer data [2].

“CD Ripping” actually did not become especially common until the early ’00s. In practice, “ripping” audio CDs from PCs is somewhat complex because of the surprisingly archaic architecture of PATA CD drives. Early computers, in the ’90s, often weren’t capable of fast enough I/O to read an audio CD at 1x speed (meaning as fast as the audio bitrate, allowing real-time decoding). Even for those that were, audio CDs amounted to hundreds of megabytes of data and hard drives that large were very costly. As a result, PC CD drives played audio CDs by behaving as plain old CD players and outputting analogue audio. Some of you may remember installing an IDE CD drive and having to connect the three-wire analogue audio output from the CD drive to the sound card. Some of you may even remember the less common CD drives with discrete playback control buttons on the front panel, allowing it to be used to play music without any media software to control it.

These CD drives, when playing audio, behaved a lot like smartphones making phone calls: the computer actually wasn’t “in the loop” at all, all it did was send the CD drive commands to play/pause/etc and the CD drive decoded the audio and converted it to analog internally, sending it straight to the amplifiers in the sound card. With this architecture, the way to “rip” a CD was actually to tell the CD drive to start playback and then use the soundcard to record its analogue output. This ran only at real-time speed, not all soundcards were capable without an external jumper from the line out to line in, and the double conversion reduced quality. It didn’t take off, although Gnome’s Sound Juicer continued to use this method until around 2005.

The much better method, of actually reading the CD as data and decoding the audio in software, took off in the ’00s, mostly in the form of WinAmp 5.0 and Windows Media Player in XP. It greatly accelerated a trend which already seemed clear to industry: consumers would purchase their music on physical media (downloading it was still infeasible on most consumer’s internet connections), but in the future they would immediately rip it, store it on a central device, and then play it back in digital form using a variety of devices.

Microsoft leaned hard into this vision of the future, not just a little because it strongly implied multiple licensed copies of Windows being involved in the modern home stereo. A significant development effort that entailed the addition of multiple new operating system features lead to Windows XP Media Center Edition, or MCE, released in 2002. MCE was a really impressive piece of software for the time. An MCE computer wasn’t just a computer, it was a “home media hub” capable of consuming media from multiple formats, storing it, and then streaming it over the network to multiple devices. MCE introduced consumers to significant expansions of the role of the “computer” in media. For example, MCE supported playback and recording of television from cable and multiple OEMs sold MCE desktops with preinstalled cable tuners. An XP MCE machine could compete with Tivo, but with distributed playback which Tivo would not introduce until later.

Microsoft leveraged their newfound place in the home theater, the Xbox, to create an ecosystem around the platform. First-gen Xbox consoles with a purchased upgrade and all Xbox 360 consoles could function as “media center extenders,” which entailed opening an RDP session to the MCE machine to present the MCE interface on the Xbox. Extensions to RDP implemented to support this feature, namely efficient streaming of HD video over RDP without re-encoding, went on to drive the “network projector” functionality between Windows Vista and various Ethernet-enabled projectors. This was

arguably the primary precursor to the modern ChromeCast as RDP introduced some (but not nearly all) of the offloading to the streaming target that ChromeCast relies on. In fact the Media Center Extender and Network Projectors are closely related both technically and in that they have both faded into obscurity. Modern “network projectors” typically rely on the simpler, AirPlay-like and Microsoft-backed but open Miracast protocol [3].

Because Media Center Extender relied on RDP, it essentially required that the Extender implement a good portion of the Windows graphics stack (remember that RDP is basically a standardization of Citrix’s early application streaming work, which made a lot of assumptions about a Windows client connecting to Citrix/Windows server environment and offloaded most of the drawing to the client). This was no problem for the Xbox, which ran Windows, but the improvements to RDP and open standards that make non-Windows RDP clients practical today were not yet complete in the ’00s and it was unreasonable to expect conventional consumer A/V manufacturers to implement the Extender functionality. Microsoft didn’t make anything near a full line of home media products, though, and so it needed some kind of Ecosystem.

Conveniently, Intel was invested in the exact same issue, having begun to introduce a set of Intel-branded media streaming solutions (including WiDi, a true unicorn to see in the wild, which allowed certain Intel WiFi adapters to stream video to a television well before this was a common consumer capability). Because media streaming required fast I/O and network adapters, both Intel offerings, it had real potential to expand consumer interest in Intel’s then commercial products. Intel kicked off, and recruited Microsoft and consumer A/V giant Sony into, a new organization called the Digital Living Network Alliance. That’s DLNA.

DLNA built on UPnP, the then-leading consumer network auto-discovery and autoconfiguration protocol, to define a set of network standards for the new integrated home media network. DLNA defined how a set of devices broadly categorized as Media Servers, Media Players, and Media Renderers would discover each other and exchange signaling in order to establish various types of real-time media streams and perform remote control. As a broad overview, a Media Server offered one or more types of stored (e.g. files on disk) or real-time (e.g. cable tuner) media to the network using established protocols like RTSP. A Media Player allowed a user to browse the contents of any available Media Servers and control playback. A Media Renderer received media from the Media Server and decoded it for playback.

The separation of Media Player and Media Renderer may seem a bit odd, and in practice they were usually both features of the same product, but enabled network-based remote control of the type we see today with Spotify Connect. That is, you could launch a Media Player on your computer (say Windows Media Player) and use it to play audio on the Media Renderer integrated into your stereo receiver. Other Media Players could discover the renderer and control its playback as well. This was all possible in Windows XP, although it was very seldom used because of the paucity of Media Renderers.

The problem is not necessarily that DLNA failed to spawn an ecosystem. While Microsoft added pretty complete DLNA support to Windows through MCE, Windows Media Player, and Explorer (where it became intertwined with the Homegroup peer-to-peer network sharing system), JRiver and TwonkyMedia were major third-party commercial DLNA Media Servers. On the open source side, Xbox Media Center (XBMC), now known as Kodi, had fairly strong DLNA support. Plex, still a fairly popular home media system today, originated as a port and enhancement of XBMC and went on to gain even more complete DLNA support. Even Windows Media Center itself was extensible by third-parties,

providing a public API to add applications and features to the MCE interface.

Years ago, around 2008, I successfully interoperated TwonkyMedia, Plex, Windows Media Player, and a Logitech hardware renderer all via DLNA. This has become more difficult over time rather than less, as the market for DLNA-capable hardware has thinned and DLNA software support has fallen out of maintenance.

One major challenge that DLNA faced was the low level of consumer readiness for fully digital home media. It's an obvious problem that consumers weren't yet used to the idea of "smart TVs" and other devices that would integrate DLNA clients. DLNA clients were somewhat common in devices like HD-DVD and Blu-Ray players (these already needed a pretty significant software stack so adding a DLNA client wasn't much extra effort), but it tended to be a second-class feature that was minimally marketed and usually also minimally implemented. I remember a particular LG Blu-Ray player that had a mostly feature-complete DLNA Media Player and Media Renderer but struggled to actually play anything because somewhere between its chipset and 802.11g, network streaming performance was too poor to keep up with 1080p content.

And 802.11g is part of the problem as well. Home networks were surprisingly bad in the early to mid '00s. I suppose it shouldn't be that surprising, because "home network" was largely a new idea in that time period that was displacing a single computer directly connected to a modem. Broadband was taking over in cities, but DOCSIS and DSL modems still provided a USB interface and it was not uncommon for people to use it. Almost no one ran ethernet because of the high cost of installing it concealed and the frustration and aesthetic impact of running it along floorboards [4]. 802.11g, the dominant WiFi standard, was nominally fast enough for all kinds of media streaming but in practice congestion and range issues made 802.11g performance pretty terrible most of the time. All of this is still mostly true today, but we've gone from nominal WiFi speeds of 54mbps to well over a gigabit, which allows for a solid 20mbps even after the *very* substantial performance reduction in most real environments.

The biggest problem, though, is the troublesome concept of a "server" in the home. Microsoft seemed to operate, at the time, on the assumption that a desktop computer would serve as the Media Server. In fact, MCE specifically introduced power management enhancements to Windows XP intended to allow the MCE machine to stay in standby mode but wake when needed for media recording or streaming (this never seems to have worked very well). Microsoft further reinforced this concept with their marketing strategy for MCE, which was only released to developers and OEMs and could not be purchased as a regular standalone license. If you were going to use MCE, it was going to be on a desktop computer sold to serve as a hybrid workstation and media server.

Unfortunately, around the same time period the household desktop started to become a less common fixture. Lower prices and better performance from laptops could free up a desk and add a lot of flexibility, and consumers were just getting less excited about buying a mid-tower. And besides, the hard drive capacities of the time made a normal desktop somewhat limited as a media server. Remember, this was back in that awkward period of physical media where it was common for desktops to have *two* optical drives because people wanted a DVD player and a CD burner, and no one had figured out how to economically fit both into one device yet. Hard drives were usually in the hundreds of GB only.

What was needed, it seemed, was some sort of home media server appliance that was compact, affordable, and user-friendly enough that people would consider it a typical

network appliance like a router/AP combo. That's sort of what Apple delivered in the AirPort Time Capsule (and then later kind of in the Apple TV, it got confusing), and pretty much what we would call a consumer NAS today (like a Drobo or QNAP or something). Back in the mid '00s neither of these options were available on the market, and most people weren't going to watch the episode of Screen Savers about connecting a JBOD to a desktop on the cheap. So, in 2007, Microsoft served up a real wonder: Windows Home Server.

Windows Home Server was a stripped down (although surprisingly little) version of Windows Server 2003 R2, intended to be easily set up and managed by a consumer using a desktop client called the Windows Home Server Console. The Console was a bit like the MMC in principle, but with fewer features and more wizards. Although I'm having a hard time confirming this, I believe it actually worked via RDP using the Virtual App technology, meaning that the actual Console ran on the server and only the GUI drawing occurred on the client. This probably eased implementation but had the result that you could only really manage Windows Server from Windows, which did not help at all against Apple's widely perceived media dominance.

Not a lot of Windows Home Server devices made it to the market. HP released several, one of which I used to own. The larger HP Home Servers were pretty similar to a modern consumer NAS, with multiple front-accessible drive bays. Home Server functioned (perhaps first and foremost) as a DLNA server, but also offered a number of other services like backup, brokering RDP connections from the internet for remote desktop, and a web-based file browser to get your files remotely--naturally tied into .NET Passport. Home Server was extensible and third-party software could be installed from a small app store and managed from the Console; both Plex and Twonky offered Home Server versions at the time. Several commercial antivirus packages were available as well, as this was a bit before Microsoft took antivirus and HIDS on as a first-party component, so Norton and McAffee could still make a lot of money off of getting you to pay for one more license. In fact the Home Server versions of these antivirus products were interesting because they often functioned more like a miniature enterprise antivirus/HIDS platform, centrally controlling and monitoring your host-based security products from the Home Server.

Windows Server also introduced a significant new feature called Drive Extender, which was a high-level file system feature that allowed the user to combine multiple drives into one logical drive and use multiple drives for redundancy, all in a way which was largely agnostic to the drives and interfaces in use. This was presumably a response to the high cost of the hardware RAID controllers that served the same purpose in most "real" Windows Server machines, but it compared favorably with the software-first approach to storage management that would shortly after spread through the POSIX world in the forms of ZFS and Btrfs. Ironically Drive Extender was a source of a lot of frustration and data loss as it turned out to buggy, which is perhaps why Microsoft ignominiously killed the feature along with Windows Server. Years later later it would reappear, seemingly in a much reworked form, under the name Storage Spaces.

And what would you guess happened with Windows Home Server? That's right, it failed to gain traction and slowly faded away. Windows Home Server 2011 did make it to the light of day but proved to be the last version of the concept.

Ultimately I think it was a victim of Microsoft's usual failures: it addressed a new use-case rather than an existing one, so it wasn't something that consumers had much interest in to begin with. Microsoft massively failed at creating a marketing campaign that would convince consumers otherwise. The cost of Home Server devices was pretty high (for much the same reasons that consumer NAS continue to be rather

expensive today), and at the end of the day it just kind of sucked. In a fashion very typical of Microsoft, it had a lot of interesting and innovative features but the user experience and general level of polish were surprisingly poor. The Console, I remember, was sometimes nearly impossible to get to connect without rebooting the server. The Drive Extender faults lead to a lot of instances of data loss which then lead to bad press. It formed a part of Microsoft's generally confusing and poorly designed home sharing experience, later Homegroups, and so got tangled up in all of the uncertainty and poor usability of those features (for example to easily get access to the server's storage via SMB). The whole thing was a failure to launch.

And that sort of sums up the fate of DLNA as well: despite the best efforts of its promoters, DLNA never really got to a position where it was attractive to consumers. It was mostly intended to solve a problem that consumers didn't yet have (access to their non-existent local media library and their non-existent computer cable tuner [5]), and time has shown would never really have. The ecosystem of DLNA products, outside of Windows, was never that large. Dedicated media renderers never gained consumer adoption, and devices that threw in DLNA as a value-add (like a lot of HD-DVD players) did a bad job and didn't promote it. DLNA was complex and a lot of implementations didn't work all that well on home networks, which was also true of related technologies (like SMB file sharing) that were important parts of the whole home network ecosystem.

Moreover, DLNA was wiped out by two trends: the cloud, and proprietary casting systems.

First, the cloud: the idea that consumers would have a large local library of music, TV shows, and movies, has never really materialized. The number of people who have a multi-terabyte local media collection is vanishingly small and they are basically all prolific pirates, which makes the broader media industry uninterested in keeping them happy. In fact much of the media industry worked to actively discourage this kind of use pattern, because it is difficult to monetize in a reliable way.

Instead, most consumers get their media from a cloud-based streaming service like Spotify, which has no requirement for any particular features on the local network. Since these types of services already need significant cloud support, it becomes easier to implement features like casting and network remote control within the product itself, without any need (or support) for open standards. No one needs DLNA because they use Spotify, and Spotify has worked commercial partnerships to get Spotify Cast support in their A/V devices.

Second, proprietary casting systems: one of the features but also, in hindsight, defects of DLNA was its underlying assumption of a peer-to-peer "matrix" system in which many devices interacted with many devices in a flexible way. In practice, 90% of consumer use cases can be solved by a much simpler system in which a media renderer operates under direct remote control (and perhaps receiving media directly from) of a computer or phone. Miracast, for example, operated in this fashion and while it never became that common it was much more practical to integrate Miracast support into a device like a TV than a DLNA renderer and player.

Moreover, casting features offer a compelling opportunity for vendor lock-in, since it is natural to integrate them with operating systems and specific applications. While Microsoft made some effort to promote the Miracast standard it was lackluster at best, so the whole space was dominated by Apple (which had a head start in the form of iTune's remote playback capabilities and massive leverage by integrating the feature into iOS) and Google (which mostly won by making the Chromecast extremely cheap,

although leveraging the YouTube app was also a major boon). Neither of these companies have much interest in facilitating a multi-vendor ecosystem, and in the case of consumer A/V devices where they have little strength they operated for closed partnerships with device manufacturers over open standards. Along with commercial incentives the scheme works: my TV supports the closed standards AirPlay and Chromecast, but ironically not the open standard Miracast. The only device I have with integrated Miracast support is a no-brand sub-$200 DLP projector, where it works very well.

DLNA formally dissolved in 2017, although the writing was on the wall as early as 2010 when Spotify began to transform the way music was consumed. Similar capabilities are now found in various vendor-proprietary systems, but few of these approach the original ambitions of DLNA. A huge industry preference for cloud-intermediated platforms and increasing consolidation of home media onto one of a small number of walled gardens makes any serious resurgence of a DLNA-like project unlikely.

And Sonos? I don't want to be too mean to Sonos, the technology is pretty cool. I just have a hard time dropping $500 on a speaker that handles the unsolved problems of 2008 but not the solved ones. These newer distributed media systems (also Yamaha Musicast, for example) are impressive but fail to provide the flexible, multi-source matrix architecture that DLNA had once put within reach.

At the end of the day, who killed DLNA? In some ways DLNA was ahead of its time, as it required better home networks than most people had and enabled a media consumption pattern that was the future, not the present. Of course it was ahead of a time that turned out not to exist, as the cloud streaming services took over. In this way, technological progress (or more cynically the twisted economics of the cloud) killed DLNA. Microsoft, and to a lesser extent Intel, killed DLNA through poor marketing, few partnerships, and repeatedly bungled products. Apple and Google killed DLNA by seeing a simpler and more commercially advantageous solution and making it widespread (and we can't blame them for this too much, as AirPlay and ChromeCast really are plainly easier to implement well than DLNA ever was).

And, you know, capitalism killed DLNA, because as an open-standards distributed system its profit-making potential was always limited. The consumer A/V industry that could have flocked to DLNA because it offered wide interoperability instead flocked to the proprietary standards because they offered money. Heavyweights Apple and Google were playing for keeps. Microsoft's unusually generous dedication to open systems ultimately left them holding the bag, and to this day Windows lacks a coherent media streaming ecosystem.

Also frankly all the DLNA products pretty much sucked but hey, I'm running on nostalgia.

[1] This approach can be used as an alternative to, or in combination with, high-voltage audio systems which we have discussed before.

[2] Due to the surprisingly analogue CD mastering process and the use of generous error correction and error tolerance in CD playback, it is surprisingly common for it to not actually be possible to create an exact copy. But the general point still stands.

[3] Miracast is kinda-sorta a subset of WiFi, being standardized by the WiFi alliance, and builds on previous HD-media-over-WiFi efforts like Intel WiDi. These were tremendously unsuccessful but are important precursors of ChromeCast and AirPlay. A

lot of devices, including all Windows machines, support Miracast but it's pretty rare to see it used in practice as TV manufacturers have not been enthusiastic about it (which is to say Microsoft has not incentivized integrated Miracast in the way that Apple has incentivized integrated AirPlay).

[4] This is still basically true today, although structured wiring has made some inroads in new, especially multi-family, construction. Of course I continue to meet people who live in a home or apartment with structured ethernet and CATV who don't even realize it or don't understand how to use it. Consumer interest in plugging cables into things remains low.

[5] In fact the whole DLNA story is tied up in the CableCard story, which I'll probably write about in the future. The short version is that the whole idea of using an arbitrary tuner for a cable subscription was unpopular with cable carriers and hard to implement. Cable carriers preferred to provide their own set-top boxes and DVRs, which consumers were mostly happy with. The concept of connecting your cable, or even TV antenna, to a computer just never really went anywhere.