Payloads for RSS

Thu, Jan 11, 2001; by Dave Winer.

The click-wait system

When I started talking with Adam Curry late last year, he wanted me to think about high quality video on the Internet, and I totally didn't want to hear about it. Like a lot of people, I had tried it, and found it unsatisfying and frankly, exhausting.

I thought that video on the Internet was a loser for three reasons, that build on each other:

1. When I click on a link to view some video, I have to wait.

2. The wait is longer than the video. (In other words I have to wait two minutes for ten seconds of video.)

3. The quality is horrible.

All three effects are bad, but the first is the worst. The Internet lifestyle is frenetic. There's no time to wait. The remaining two negatives only make video less attractive, but the first is the killer.

But Adam persisted and showed me that if I was willing to change my point of view, it could work, without any waiting and with very high quality.

No click-wait

What if, in the middle of the night, while I'm not using my computer, it downloads huge video and audio stuff to my local hard drive. Then when I arrive in the morning there are fresh bits, news clips, a song of the day, whatever, provided by all kinds of content providers, from big TV networks like CNN and MSNBC, to a Dutch school where kids are taking a film class using inexpensive video recorders and iMacs.

Let's see what happens with 1, 2 and 3 in this scenario.

1. When I click on a link to view some video, it starts playing immediately, because it is already on my local disk.

2. The wait is zero.

3. The quality is limited by the size of my local disk, not by the capacity of my connection.

A different user interface

What's different about this system is that you *subscribe* to channels instead of clicking-and-waiting. I feel that nothing is lost with this approach, because video on the Internet never worked for me, and probably for many others. However, streaming news items through RSS works quite well, and it's a simple matter to teach

RSS about multimedia payloads.

Of course there's a change in user interface. You never see a video or music document until it's fully downloaded. The computer does the waiting, not you. In the system that Adam envisions, a video DJ, someone like Adam at MTV in the 80s and 90s, someone whose judgment you trust, or whose tastes you like, is pushing high fidelity bits onto your hard drive, using no more hardware and network connections than you already have.

There is no central authority, no spectrum to allocate, it's open to amateurs, like the Internet itself.

A new element

So that's the philosophy, now let's put some meat on the bones.

RSS already does most of what we want. With the addition of the <enclosure> sub-element of <item> any RSS element can describe a video or audio file (actually any type of file).

<enclosure> has three attributes: url says where the file is located, length says how big it is, and type says what its type is. This way a workstation or aggregator can know in advance, without having to do any communication, what it's going to get, and apply scheduling and filtering rules.

An example

Here's an example of a RSS file that links to several Grateful Dead songs. (Broken link removed).

Each <item> contains an <enclosure> as explained above.

<enclosure url="http://www.scripting.com/mp3s/touchOfGrey.mp3" length="5588242" type="audio/mpeg"/>

I've configured my system to download enclosures between 2AM and 4AM, a time when I don't use my computer.

When I arrive at work, I check the incoming Log and see there's a new song.

I click. It plays.

No wait.

Software

My company, UserLand, has a product in development called My.UserLand On The Desktop that supports both sides of this format. You can use it to create channels that have <item>s with <enclosure>s, and it manages downloads for these channels. So it's both an authoring and viewing tool.

It's a bootstrap, and it's open, meaning that anyone can develop software that works on either or both sides of this connection.

Adam's company is doing a graphic environment for music and video on PCs in Java. Others are welcome to join up and do software for all kinds of computers, content and people.

"This is just email"
First, everything on the Internet is just like something else. Or if it's any good it's just like *everything else.*

If you try to push video and audio through email you'll find the client user interface as daunting as the wait-click Web experience. How do I keep enclosures from downloading while I'm accessing email from a phone connection in a hotel? Sure, you could add a complex dialog to an already overly-complex email client user interface, and the result would be that no one could set it up properly and no new video would actually get to the desktop. (You'd hate the feature until you figured out how to turn it off.)

Homage

I'd like to pay special homage to the Grateful Dead, who have generously allowed their creations to be used to bootstrap new technology in non-commercial ways. To get this process going, we need a content base to get started with. So many technologists, me included, love the music of the Dead. I think they may have left a legacy for technologists that's as important as the legacy they left in music.

See also

1/5/01: SOAP meets RSS describes a publish-and-subscribe notification system based on XML-RPC and SOAP and of course RSS.