Workpackage 3 Part 14: Establishing Repository and Storage facilities for media content produced by the six Colleges.

This section lies in three pieces - our experience in testing existing software to see if anything suited our needs, our trial period and experiences, and the migration to a longer-term “production” version of the proof of concept site.

Comparison of Software Packages

We compared several open-source media-focused CMSs and Apple’s Podcast Producer Solution to find a well-rounded and easy-to-use solution for hosting media.

The options we considered were Apple’s Podcast Producer, Drupal, Kaltura CE, MediaCore, MediaMosa, and OpenCast Matterhorn.

Podcast Producer 2 (Apple)

Apple’s Podcast Producer is the solution Apple launched to support institutions wishing to build themselves an iTunes U presence. It is intended to be a scalable way for institutions to send media to a cluster of Apple XServe machines to have it re-encoded and (to some extent) published to an iTunesU sites. Our own experiments with Podcast Producer 2 have led to us having immense difficulty getting Podcast Producer 2 configured correctly, as even for testing it requires a fully functional Kerberised directory service for authentication as well as a functioning XGrid cluster service running, and although it offered to autoconfigure these services on initial setup of PCP2, it did not ‘Just Work’, nor could we find enough information in the logs to divine why it did not seem to be processing any media sent to it. It seems that other institutions had also had little luck with PCP2, and rather stuck with the original PCP, which apparently was more flexible. This was an option unavailable to us and our test XServe, as Podcast Producer 1 was only bundled with the previous version of OS X Server to the one installed on our machine.

Despite our unsuccessful attempts to get PCP2 going, many things did become clear, namely that there wasn’t much scope for a cloud based PCP solution, given the requirement of physical Apple server hardware (the mainstay of which, the XServe, being discontinued in January 2011 without any viable alternate being suggested by Apple, and throwing the future of OS X Server and Podcast Producer into doubt), and that even if we were to continue down the podcast producer route, to provide access to members across the Bloomsbury colleges there would be an uphill struggle to unite all 6 authentication systems of the colleges into one Active Directory instance.

Drupal
Drupal is an Open Source content management system. Whilst it’s native functionality does not include media management or publishing, this can be added in with third party modules. There are examples within the consortium already of sites serving media using Drupal, such as SOAS’s OpenAir radio and the Endangered Languages Archive. However, from OpenAir’s experience, we know that building a complete media platform out of Drupal is a time-consuming and not a clear-cut process, furthermore there is no obvious or planned provision for the re-encoding of media with the plugins used in OpenAir’s site.

MediaMosa

MediaMosa is discussed alongside Drupal as it is a management framework based upon Drupal - it is an entire Web-based media management and distribution framework, which was developed for Dutch institutions to manage educational resources. We trialled an installation of MediaMosa, and whilst the backend seemed quite promising in terms of flexibility, we found ourselves coming stuck with a lot of the documentation only available in Dutch - and that we would have to craft our own frontend to the system with very little English documentation (that we could find).

Kaltura

Kaltura describes itself as an open source video platform, and is available as a hosted or self-hosted solution, with the hosted solution taking advantages of Amazon’s Web Services platform. We had a brief test of the self-hosted “community” edition, which was fairly similar to MediaMosa in that it allowed us to set up different repositories to support different sites, and could manage an encoding cluster based on similar tools, however again this would require crafting a frontend with support to cope with the various types of media thrown at the system.

Mediacore

Mediacore is an active Open Source project designed to create an all-in-one media management and publishing solution, and although in it’s early stages, the development roadmap and features it has already - namely the ability integration with cloud storage platforms and cloud-based media processing platforms are in tune with the general aims of the project. Our initial testing found Mediacore to be simple to use and administer once installed, and whilst there is no scope currently for integration with directory services, there are parties - including the developers of Mediacore themselves - willing to work on features like this for a price. This could also mean that there may eventually be support built in for platforms such as Kaltura.

Also, Mediacore is flexible enough to allow publishing of media hosted on other servers through it’s frontend, allowing basic integration with any other system, such as existing MP4/FLV streaming servers that colleges may have, or Youtube/Vimeo and other public video hosting sites, which provides an easy route to migrate in existing content from other colleges.

OpenCast Matterhorn
OpenCast Matterhorn is an education-focused Open Source solution aimed at managing large amounts of Media, namely as a lecture-capture focused system. A version 1.0 release was announced in August 2010, after a year of international collaborative development work on the project. If development continues at the pace it has done for the first year, Matterhorn could turn out to be an incredibly good value for money suite of tools in terms of media encoding and management, however as it currently stands it is not mature enough in terms of feature set (especially with regards to the incomplete authentication/media portal) to be the solution to fit our needs at the moment.

Another issue is that as it is entirely free software, there may be long-term licencing issues in terms of the final codecs that are used for publishing content encoded by the Matterhorn suite - this hasn’t been an issue at the front of many people’s minds due to the widespread use of free software packages such as ffmpeg and mencoder for converting content into MP3/h264/flash video, however this may be due to the fact that no legal action has been brought regarding the idea of patents on video compression yet - either way, the last thing we want is to be that test case.

**MediaCore Pilot**

We implemented a trial of Mediacore as a proof-of-concept site on shared space on a development server within SOAS (a 6 year old Dell Optiplex 1750), and has been undergoing a period of soak testing with the help of the media officers to see how the site copes under stress and time, as well as testing upgrades to the base Mediacore software.

This also allowed us to test various storage options and how the performance relates to the end user, and also evaluate the extent that we could integrate with cloud services.

In terms of cloud storage, we looked at Rackspace Cloud and Amazon S3 (Simple Storage Service). Google Storage was quickly discounted as it’s currently aimed at developers and only available to US developers. Amazons S3 also won out in a simple comparison of features against Rackspace Cloud due to Amazon’s Cloudfront service, which uses S3 backed storage but can provide content as a download or a stream, at Amazon’s endpoint closest to the user, meaning that the end user would ideally get the fastest possible download/stream relative to his location.

Support within Mediacore for Amazon S3/Cloudfront is currently not great - however version 0.9 due in early 2011 promises a modular storage engine system, with greater support for S3, and in the current version a workaround can theoretically be built with some server-side tweaking.

The other thing that attracted us to Mediacore was the included frontend - we haven’t customised the look and feel that much from the included templates simply because the default layout and design works fine out-of-the-box - we simply haven’t needed to as a priority.
The one aspect that soon became an issue was the lack of any encoding engine, as we soon came across a situation where users testing the site were uploading videos which were either in unsuitable video codecs, or encoded in correct codecs (FLV/MP4) at too high bitrates or without the flags that support the video streaming/pseudo-streaming. Eventual support for a cloud-based processors such as Pandastream, Bits on the Run, Brightcove and Encoding.com are touted, however our initial expectations for the rate of which content would initially come into the Bloomsbury Media Cloud would not justify the monthly cost of these services at the moment (starting $100/month + cost of storage on Amazon S3 in most cases), however knowing that the possibility for re-encoding incoming content in future should be available means that if/when the platform becomes popular enough that we are unable to cope with shepherding users into using suitable formats/encoding parameters as part of a quality control process, we can start looking at implementing one of these solutions.

**Migration to Production State**

By November after extensive testing we decided to move the installation over to a hosted instance at the University of London Computing Centre. Whilst we did consider running our instance of Mediacore on Amazon’s EC2 cloud computing platform, it had the drawback that we could not pay the entire bill upfront, rather we would have to pay as the system is used, which would leave us struggling to nail down exact costs as we would be paying based on how popular the service would be.

The Bloomsbury Media Cloud portal is now running Mediacore version 0.8.2 on a Virtualised server running Red Hat Enterprise Linux 6. ULCCs standard offering was RHEL5, however the version of Python (the programming language Mediacore runs on) provided by RHEL5 was not current enough (2.4 provided by RHEL5, 2.5 or greater needed, RHEL6 provides 2.6) to run Mediacore.

We also intended to run Mediacore using the Apache mod_fastcgi module rather than under mod_wsgi as we had on the development box, as this would have provided less hassle and would mean that we could upgrade easily without seeking support from ULCC, however we seemed to keep running into errors with file and system permissions with the fastcgi configuration method, so we reverted back to using the tried-and-tested mod_wsgi. The side was fully migrated by December 22rd and the previous SOAS hosted development site was taken down the next day due to decommissioning of the server.

In terms of storage, as the initial presence we have is under 50Gb, we are covered by the basic ULCC hosting package, and storage can be increased as and when it is needed by agreement with ULCC, although there is nothing preventing us from using Amazon’s S3 service instead to extend capacity if needs be, and facilitation of this (and other services) should be made simpler in later versions of Mediacore - failing which, similar results could theoretically be achieved with the s3fs FUSE module and apache rewrite/redirect rules, at the expense of more bandwidth usage on the ULCC hosted instance.
2. **WP3.15: Setting up the Bloomsbury iTunesU and YouTube EDU sites**

With regards to iTunes U, a contract signed in the name of RVC, as sites are only granted to degree-giving institutions and must be under their name name, and despite our unique situation Apple have been unwilling to bend this rule to suit the consortium approach, due to issues of who would be responsible for the content if legal complications were to arise. The only workable solution would be to have a site per institution – i.e. six separate sites, although this causes issues with some colleges possibly not having enough content, or ability to gain enough support for an initiative like this from within.

We have not been able to gain access to the new ‘public site manager’ for the RVC’s iTunesU site (due to the consortium issues), however I have tested our developmental infrastructure - namely feeds generated by our test MediaCore installation against the old site manager and there have been no issues.

The “Podcast” functionality of the MediaCore site would allow an institution or many to aggregate content from the media cloud onto iTunes U.

As for Youtube EDU, the application form states “To be considered for YouTube EDU your channel should already be established with a representative amount of educational videos. YouTube EDU is intended for educational materials as opposed to promotional.”

Of course our status as a consortium may again cause issues, but also the fact that we have little educational material gathered in video form, mainly ‘Academic Angles’ interviews with Heads of Colleges which could be seen as being more promotional – other more ‘educational’ content is mainly in audio form. YouTube seems to be seeking more in the way of recorded lectures.

3. **WP 3.16: Investigating the best possible authentication method for the private iTunesU site and WP 3.17 Investigating iTunesU compatibility with Blackboard**

Since I attended the iTunesU Tech Series Briefing on managing internal sites held by Apple on July 7th, we have been experimenting with how authentication/authorisation to private iTunesU sites works.

The crux of this is outside web applications (such as a Blackboard module) provides the authentication challenge to the user and redirects them to the Private iTunesU site - either to a holding page with access to the specific relevant content, or directly to the course they wanted to see. We spent time experimenting with the various permissions available within the iTunesU site – Administrator, Tutor, Student, Unauthenticated (Anonymous), as well as custom permissions, however manually managing these

The Blackboard module installed on staging server for the Bloomsbury Learning Enviroment and linked to the RVC’s private iTunes U page, can create iTunesU courses on an iTunesU private site for Blackboard Courses and authenticate and link students to them - which authorises users to view and edit content that they should
have access to based on Blackboard’s permissions, which will be the most comprehensive authorization details already in existence across the colleges. However, the module will only work with one iTunesU private site, and this brings us back to the issues with Apple over a shared iTunesU presence.

This leaves us with two options if we were to proceed with a private iTunesU site:

1) Use one institution’s private site as the iTunesU site for all VLE media content, however whether we’d be caught in the same issues that beset the public site remains to be seen, so this would require further discussion with Apple.

2) Tinker with the module to split users off to separate iTunesU sites based on Course. (this should be technically possible, however as my experience with Blackboard is limited I am not entire sure of the viability of this option.)

However, given that authentication and authorisation to a private iTunes U site would be best suited to be done through the Virtual Learning Environment anyway, there is no reason why it would not be more practical to provide content through the VLE in a more direct fashion - Kaltura provides a level of Blackboard integration, and Echo360, which is used already in some institutions within the consortium now (as of version 3.0) can re-encode media not generated as part of the lecture capture process and publish that to blackboard courses - routing this content through iTunes instead would probably make it harder for everyone to access, as not everyone has/can have iTunes installed on their machine.