Cloud Computing: SciTube

A scientific Video-on-Demand-Solution powered by Microsoft Azure

Markus Herche[1], Benjamin Hildebrandt[2]

Abstract: The saying goes that a picture says more than a thousand words. This should apply to scientific papers as well, because why describe e.g. the deformation of a curve, when that can just be shown in a quick video. The following paper outlines a basic cloud-based video-on-demand-solution with a pay-per-view-model.

Note: This project made possible through the Microsoft Azure Academic Program.

1 Prerequisites and Scenario

Although it is possible to find scientific content and thorough explanations of complex materia on established platforms like YouTube, it can be hard to find among other non-science related stuff. Discussion of topics is also limited as the comment sections tend to either be full of unanswered questions or just typical internet madness. By creating a platform for professionals, that might hopefully change.

Monetization is also an important factor as established publishers of journals seem content to charge around 40 euros for an article, as in: a six-page-excerpt of a journal. This is a strong indication that enriched content can also be charged for.
On Project *SciTube*, Users can create and publish Videos and recorded talks and cut out a preview for all other users. This feature coupled with a rating-system should make it easier for interested viewers to decide if they should pay for the whole thing. Creators can also decide to make the video free for everyone, which introduces ads to at least try and cover server costs.
A richer discussion through comments could be made possible through a community-based moderation-system similar to platforms like StackExchange. Users can gain reputation for quality content and proven experts (say, Doctors and Professors) could contact the team and receive extra reputation upon proof of academic success.

[1] Hochschule Fulda, Dept. of Computer Science, Leipziger Straße 123, 36037 Fulda, Germany markus.j.herche@cs.hs-fulda.de
[2] Hochschule Fulda, Dept. of Computer Science, Leipziger Straße 123, 36037 Fulda, Germany benjamin.hildebrandt@cs.hs-fulda.de

2 Microsoft Azure Fundamentals

Microsoft offers both *Infrastructure* and *Platform* as a Service through Microsoft Azure for customers to use and implement Software as a Service. Management functionality itself is provided as SaaS as well (accessible through Webbrowser and REST-API). Applications consist of multiple „building blocks“ that can be managed individually, but organized together in *resource groups*. These are groups for sorting and encapsulating different services or parts of one service. Every other component have to be in such a *resource group*.

Managing and Configuring can be done via GUI in any Webbrowser through the Azure portal [3] or through *Azure CLI* (Command Line Interface) scripts (similar to Shell- or Batch-scripts). Those can be exported from the GUI for easy reuse, minimizing the risk of missing a checkbox or a specific option. To run such scripts locally, it’s necessary to install software [4], which introduces platform-dependency. Parameters for options can be enclosed in high commas on Linux-systems, but must be put in double quotes on Windows.

Azure CLI commands (for the most part) have the following pattern:
```
az [resource-type] [action] [--option 'parameters'] [further options]
```
Upon execution they either terminate with an error message or by printing a JSON received from Azure on success, offering details about the performed action.

Azure provides many service solutions where one is a Video-on-demand service. This implementation is shown in figure 1 and contains the following cloud components[5]:

- **BlobStorage:** Can store a large amount of unstructured data and can be accessed from anywhere by using http or https. Here the files will be stored.
- **Streaming Endpoint:** Deliveres content directly to a player application or to a content-delivery network (CDN).
- **Azure Encoder:** Creates jobs, which convert media files from one encoding to another.
- **Azure CDN:** Delivers content with a broad global reach.
- **Azure Media Player:** A player application which can be used to directly stream media files from *Streaming Endpoints*.
- **Multi-DRM content protection:** Multi-DRM or AES clear key encryption.

[3] portal.azure.com
[4] https://docs.microsoft.com/de-de/cli/azure/install-azure-cli?view=azure-cli-latest
[5] https://azure.microsoft.com/en-gb/solutions/architecture/digital-media-video/

3 Our Implementation

A modification of the in section 2 proposed architecture was implemented because of cost issues and is shown in figure 1. To get a fast overview of all uploaded videos and mask the underlaying Azure architecture of the service and its credentials, a Node.JS webserver along a sql database is used as interface between user and service. After the user specified a file, it is uploaded to the webserver where a asset and a corresponding job is created by calling the Azure API. After processing and uploading successfully all information are stored in the database and the file is deleted from the webserver. These information contain a Azure streaming URL. This specific URL is used to stream the video directly from Azure by inserting in a Azure Media Player.

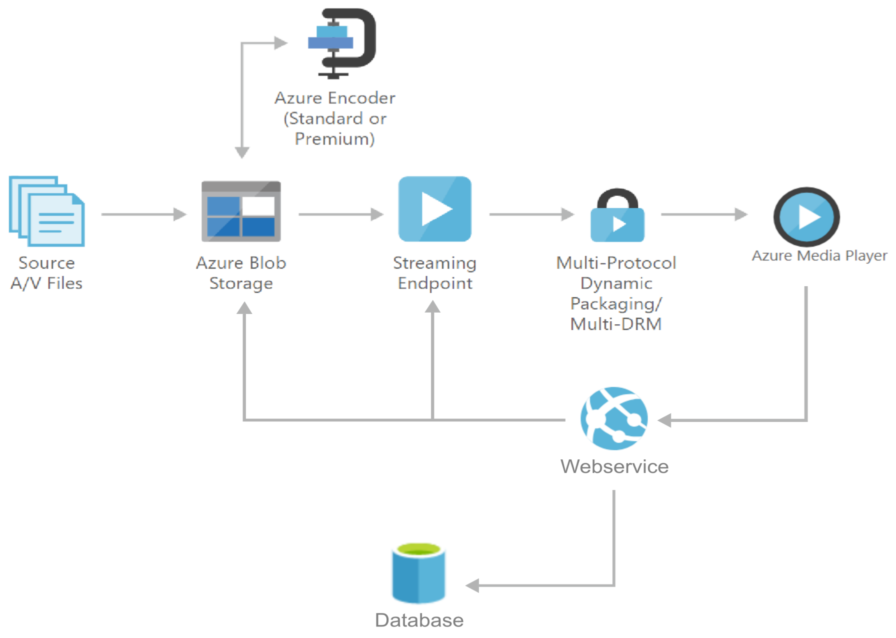

Fig. 1: Modificated implementation of Microsofts VoD architecture

Using Media Services results in a *two-tier* server-architecture, in which the NodeJS-powered AppService has the end-user's devices as clients. The webserver itself is a client to the Azure Media Services Backend, requesting Encoding- and Streaming-Tasks by submitting *Jobs*. This is also for security reasons, as Media Services uses a separate account-system with it's own credentials, generated by Microsoft. Those are confidential and must not be exposed to end-users. CORS (Cross Origin Resource Sharing) must be enabled for the Storage Account.

The cloud components used scale completly automatic when needed. The NodeJS AppService is configurated to scale along the workload of the CPU resulting in creating another instance of the service. This happens once the defined threshold is reached. Incoming requests will automaticly distributed by Azure. The database behaves in the same way, scaling up and down when needed depending on the used price tier.

4 Evaluation of cloud criteria

As defined by the US-based *National Institute of Standards and Technology* (NIST) a cloud service must fulfill certain criteria [**nistCloud**]. This section gives a brief overview and reviews the application's fulfillment of each.

- **On-Demand Self Service:** Users can create and scale resources through the aforementioned (chapter 2) user interfaces at any time and to any capacity they desire. Only starting up and allocating resources takes some time. The implementation uses the Azure API when execute *AzureMediaServices.azcli*.
- **Broad Network Access:** Provided, as Microsoft holds a global server network and content can be delivered as HTML to be viewed in browsers on supporting devices. The access can be expanded by using the in chapter 2 proposed CDN component.
- **Resource Pooling:** Resources can be organized in logical groups, but that's not the point. Resources of the provider are shared between multiple users, called 'tenants'.
- **Rapid Elasticity:** Illusion of infinite resources and scaling: provided by Azure as far as end users have a valid subscription and the cost plan includes scaling. Vertical scaling (scale-up) requires manual action, especially because the shift in price between tiers can be dramatic. Horizontal scaling (scale-out) also depends in some cases from price tiers. Scale-outs usually depend on parameter like CPU, RAM or storage usage (NodeJS webservice). Other service scale totally automatic, as the payment is per GB (BlobStorage).
- **Measured Services:** provided by azure: Insights and Billing, Alerts. Moreover, the implementation specific measurment will be the time, a user watches videos and storage is used by one user.

5 Financing

Using CDN is preferable, because "When Azure CDN is not enabled for Media Services, data transfer is charged at Data Transfer Pricing. When Azure CDN is enabled for Streaming Units, data transfer charges do not apply. Data transferred is instead just charged at standard CDN pricing." While at first the pricing for both is near identical, it's the range from the 50th on to the 150th TB in one month that already shows the disparity: $0.07 per GB through standard file transfer vs \$0.0037 via CDN. Bandwidth and CDN are also priced differently depending on region

According to an online filesize calculator [6] a 25 FPS FullHD-Video of 20 minutes length would estimate to around 1 GB in size (average between *YoutubeHD* and *NetflixHD* with a necessary bandwidth of 8Mbps (1MB/s) in case of *YoutubeHD* bitrate. Assume that after a good start there are now 100 videos as the one described before. A total of 500 visitors arrive each month and watch 2 videos each: $(500 * 2GB) = 1TB$ Streaming Throughput and $(500 * 2 * 20min)$ = 2000 minutes (33,33 hours) of streamed video. (1000GB * 0,08ct) = 80 dollar The necessary blob storage amounts to $100 * 1GB = 100GB$. Let's also assume we receive 10 new videos each month, which results in an encoding output of $(10 * 20mins)$ = 200 minutes (6.75 dollar).

This brings us to a rough estimate (appsrv+streaming + encoding + storage) of 130 dollar each month. Spreading the costs to the users means every user uses services worth (130 / 500) = 0,26ct. If we omit the costs for the AppService and only account for costs generated through user interaction we get (90 / 500) = 0,18ct per user. The AppService is a mostly fixed cost as it is billed per running time, but media services costs scale upwards with user interaction.

Costs: 80/20 Creator / SciTube Every encoding costs: $Length * BasePriceHD$ Every view costs: $VideoSize * (Streaming + CDN)$ Every video costs: $Size * StoragePrice$ Every month costs: $AppServiceBasePrice * Throughput$

6 Future Work

In the future, a final financing plan must be implemented. Some attempts were already discussed in the previous chapter. Instead of using one plan straight, after aquiring user (financing by advertising), a subscription model can be implemented. On the technical side, a CDN component should be implemented for a broad global reach and to minimize the costs as mentioned in chapter 5. Also scaling needs to be tested under real conditions to find out the indicating parameter. On one hand uploading the file from the user directly to the Azure Media Service would increase the performance of the application aswell. On the other hand it would decrease the quality of service.

[6] `https://toolstud.io/video/filesize.php`

Tables

Microsoft Azure Estimate 1/2			SciTub Basic
Service type	**Custom name**	**Description**	**Estimated Cost**
Storage	table storage 1gb	Table Storage, Standard, LRS Redundancy, 1 GB Capacity, 100 Storage transactions	$0,11
Azure Backup		Azure VMs Type, 1 Instance(s) x 0 GB, LRS Redundancy, Moderate Average Daily Churn, 30 Daily RPs, 0 Weekly RPs, 0 Monthly RPs, 0 Yearly RPs, After 1st year Duration, 0 Total Storage	$0,00
App Service		Basic Tier; 1 B1 (1 Core(s), 1.75 GB RAM, 10 GB Storage) x 730 Hours; Linux OS	$38,69
Content Delivery Network		Zone 1: 20 GB, Zone 2: 0 GB, Zone 3: 0 GB, Zone 4: 0 GB, Zone 5: 1 GB, DSA: 0 GB	$1,78
Load Balancer		Basic Load Balancer is free of charge	$0,00
Application Gateway		Basic tier, Medium Instance size: 0 Gateway hours instance(s) x 730 Hours, 0 TB Data processed unit(s), 5 GB Zone unit(s)	$0,00
IP Addresses		1 Dynamic IP Addresses, 5 Static IP Addresses, 0 Remaps	$0,00

Microsoft Azure Estimate 2/2			**SciTub Basic**	
Service type	**Custom name**	**Region**	**Description**	**Estimated Cost**
Storage	Temp_blob_for_encoding	East US	Block Blob Storage, General Purpose V1, LRS Redundancy, 10 GB Capacity, 100 Storage transactions	$0,28
Storage	blob_for_streaming	East US	Block Blob Storage, Blob Storage, LRS Redundancy, Hot Access Tier, 50 GB Capacity, 100,000 Write operations, 100,000 List and Create Container Operations, 100,000 Read operations, 100 Other operations. 10 GB Data Retrieval, 10 GB Data Write	$2,08
Support			Support	$0,00
			Licensing Program	Microsoft Online Services Program (MOSP)
			Monthly Total	**$42,93**
			Annual Total	**$515,16**

Images

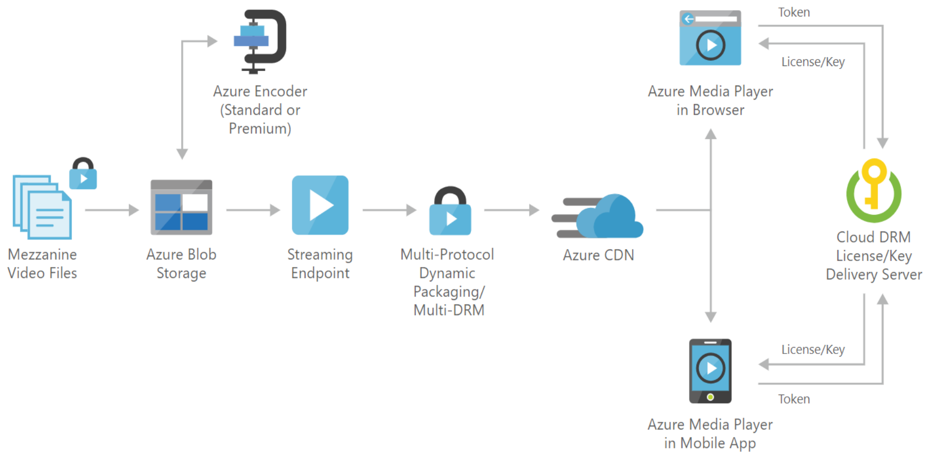

Fig. 2: Microsofts proposed VoD architecture