ALLAN THRAEN | 6 years ago | PDF |

Integrations Digizuite Optimizely (Episerver) CMS Addon Development

# EPISERVER CONTENT PROVIDER RESILIENCE STRATEGIES

**Content Providers for Episerver is a powerful tool with huge potential for integrations. But how should you handle fault resilience when dealing with a real time connection to an external system? As part of the integration to Digizuite DAM I have helped build, I have given this a great deal of thought.**

If you are a returning reader to this blog, you might notice that I have a certain passion for content providers in Episerver. The ability to connect any content source, edit it and use it to deliver a great web experience is incredibly cool. In many cases I even prefer it to the more classic approach of content replication - for a full discussion of the differences and benefits read more here:
Content Provider or Content Replication

With great power, comes great responsibility

But as always - "with great power comes great responsibility". And when you are a super-hero developer dealing with content providers you need to consider both site resilience and performance your responsibility.

When you have a typical Episerver site running in the cloud there are certain things you can take for granted: You'll always have access to your database, you'll always have access to your blobs and since the code is running on your site you'll have access to that. Since web site visitors are accessing your site you can even assert that your server is alive and it has a working internet connection! Sure, the server might crash, or there might be an earthquake at the hosting center, but then your hosting team probably have a disaster failover in place, in case your are depending on near 100% uptime.

But once you start with real-time integrations to third party systems - for example through a content provider, you need to start thinking about what happens if something goes wrong to the third party system. What if it slows down? What if it's server crashes? What if a config file is modified closing the connection? What if a router stops working? Either you need to make sure you run the connected application in exacly the same fail-over scenario - and ideally in the same hosting centers or you risk problems in the third party app will take down your entire site.

## Rule No. 1: Always keep the site running

So - to start with we decided on a few basic dogma's. The most important is that as far as possible, problems with a provider should never cause a public site to crash or even miss important elements. And only in very few cases should the user experience fall back to graceful degredation.

In the case of the Digizuite Integration, our interpretation of this is that whenever you use an asset, such as an image on a published content item, we make sure to store an emergency back-up of the asset and it's meta-data in the Episerver database and the blob storage - *but still handled by the same content provider*. That means that as a developer you won't have to do any special coding to handle an error scenario, instead the content provider will detect that it is in a fault-state, and fetch the fail-over asset if needed. For some asset's it's not always possible - like large streaming videos - but in those cases, it should degrade gracefully for the end-user.

## Rule No. 2: Keep edit-mode operational

Just as we'll want to keep the public site in mint condition, it's also important to ensure that editors can still work in edit-mode - at least with elements that does not involve the third party service. This means that it's important to ensure any UI components that might call the third party service won't crash or hang if it's unavailable - again, the content provider should just gracefully degrade it's service - but of course still make it obvious why it's not behaving as desired.

## Rule No. 3: Notify a super hero of the situation

Of course it's still essential to let the proper staff know that we are in an error state and that they should attend to it. For instance both through edit-mode notifications, but also more detailed information in the log file is important.

## Rule No. 4: Resurrect

Once the error-scenario has been fixed and connection is restored, it's of course important that the content provider can detect this, re-initialize what it needs to initialize and start to function again. You can't always count on an IIS reset on the Episerver site in these cases.

These are some of the initial thoughts that we have put into the first version of Digizuite DAM for Episerver - and I think it's considerations worth keeping in mind if you are building a content provider for an external service. In the past I've seen numerous problems with content providers - often because issues such as this hasn't been handled properly and eventually has taken a site down with a yellow page of death. Feel free to share your experiences in the comments below - how do you handle site resilience with external dependencies?

Integrations Digizuite Optimizely (Episerver) CMS Addon Development

**CodeArt ApS**
Teknikerbyen 5, 2830 Virum, Denmark
Email: info@codeart.dk
Phone: +45 26 13 66 96
CVR: 39680688