Decoupling content management using content repositories

Abhishek Tiwari

Citation: *A. Tiwari*, "Decoupling content management using content repositories", Abhishek Tiwari, 2013. doi:10.59350/5wwze-snc29

Published on: October 15, 2013

In last 5 years or so, content management systems have evolved. New breed of content management systems (CMS) are powered by content repositories. Behind the scenes content repositories use familiar persistence managers such as relational databases, file systems, in-memory etc. That said, content repositories are more than just plain persistence storage. Content repositories offer some interesting features such as,

1. Content hierarchy and sorting
2. Transactions (Multiple and parallel changes on same content)
3. Workspaces (Multiple sites and channels)
4. Versioning (Versioning content and changes)
5. Locking (Locked for editing)
6. Access control (Authorization, authentication etc)
7. API Access (REST/HTTP API access to content objects)
8. Query access (Content access using database or ORM like queries)
9. Events and Listeners (Content change and cascading)
10. Full-text search
11. Import and export
12. Workflows - Publication process
13. Channels (Multi-channel content, for different devices, online or offline)

Some of these features are file system-like such as hierarchy, access control, locking etc. Others are typical database features such query access, transactions etc. Features like versioning, full-text search, events are more content management centric.

Traditional content management systems are backed by relational databases and often considered as monolithic beasts. Moreover, various CMS features are tightly coupled with underlaying database technology. As you can see in following illustration, in a traditional CMS whole system appears to be a single monolithic block. They are not designed to add or remove a component or use best of breed components. With this kind of systems switching from one CMS to another can be a daunting task.

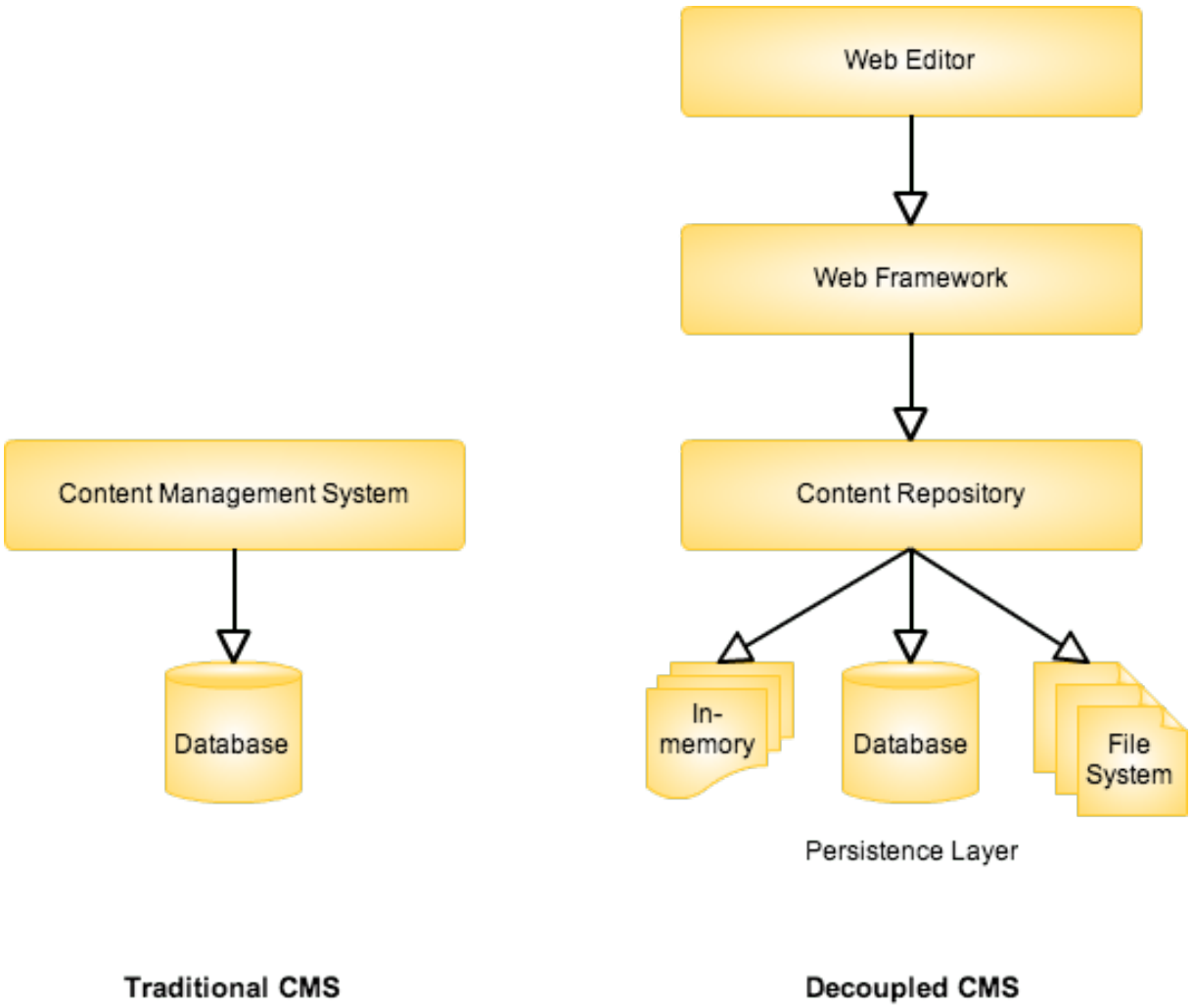

Figure 1: Decoupling content management systems using content repositories, a more simplified version

When comparing a traditional CMS to a content repository powered CMS - in terms of architecture later one is more flexible, modular and scalable because whole system is decoupled. This decoupling means more oppertunities. You can choose best of breed components for each layer. You can use web editor of your choice, web framework which you love and plug-in any persistence storage. With content repositories developers are talking with object APIs and not SQL which means there are no hard dependencies on persistence manager. You can also use multiple persistence manager in same application.

Multi-device, Multi-channel

Managing and delivering content in a multi-device, multi-channel environment has been one of the key drivers behind the growing adoption of content repositories. Using content repository a News agency can serve content both offline(print) and online(desktop, mobile and tablet) world.

Java world and JCR

With more and more content management systems adopting content repository, things are getting better. Specially in Java world, if you are using a Java Content Repository (JCR) API implementation you can easily switch from one CMS to another. For instance migrating from Magnolia to Adobe AEM or CQ5 will be always easier. Magnolia is using open source JCR implementation Apache Jackrabbit while CQ5 is using commercial JCR implementation Content Repository Extreme (CRX). With either JCR implementation you can use Apache Sling web framework which is designed to expose a JCR content repository through an HTTP-based REST API. If you want you can use Apache Jackrabbit or any other JCR-compliant source as content repository for CQ5.

JCR has become de-factor standard for content repositories. A lot of JCR ports have been developed for other programming languages including PHP, Python etc. Popular Symfony PHP framework is using PHPCR - a PHP adoption of JCR, for their CMF initiative. Midgard offers alternative to JackRabbit with support for C, Python and PHP.That said not all leading CMS platforms are built around the content repositories but many of them provide content repository like features. For instance, .net based Sitecore CMS provides a clean separation of content and presentation layer by providing content access APIs.

Personally I would like to see better JCR ports in Python and Ruby community.