Adobe, 345 Park Avenue, San Jose, California 95110, USA.

02/22/2021

Contents

Adobe Experience Manager Architecture

Introduction

Adobe Experience Manager (AEM) is a modern, application for experience management that accelerates the delivery of omnichannel personalization throughout the customer journey. It optimizes marketer and developer workflows for the entire content lifecycle informed by data insights.
In order to use AEM capabilities effectively, you should first understand the architecture of AEM.

Objectives

After completing this course, you will be able to:

- Describe the applications in Adobe Experience Manager
- Describe the architecture of AEM
- Understand Cloud Manager for AEM
- Explain how AEM fits into the Adobe Experience Cloud

AEM Applications

AEM provides an easy-to-use solution to create, manage, publish, and update complex digital forms while integrating with back-end processes, business rules, and data.

Sites

AEM Sites provides the digital foundation you need to swiftly create, manage, and deliver personalized, engaging content to every customer who visits your site. An intuitive drag-and-drop interface, out-of-the-box components, and an easy-to-use template editor help marketers deliver content quickly with minimal effort.

Forms

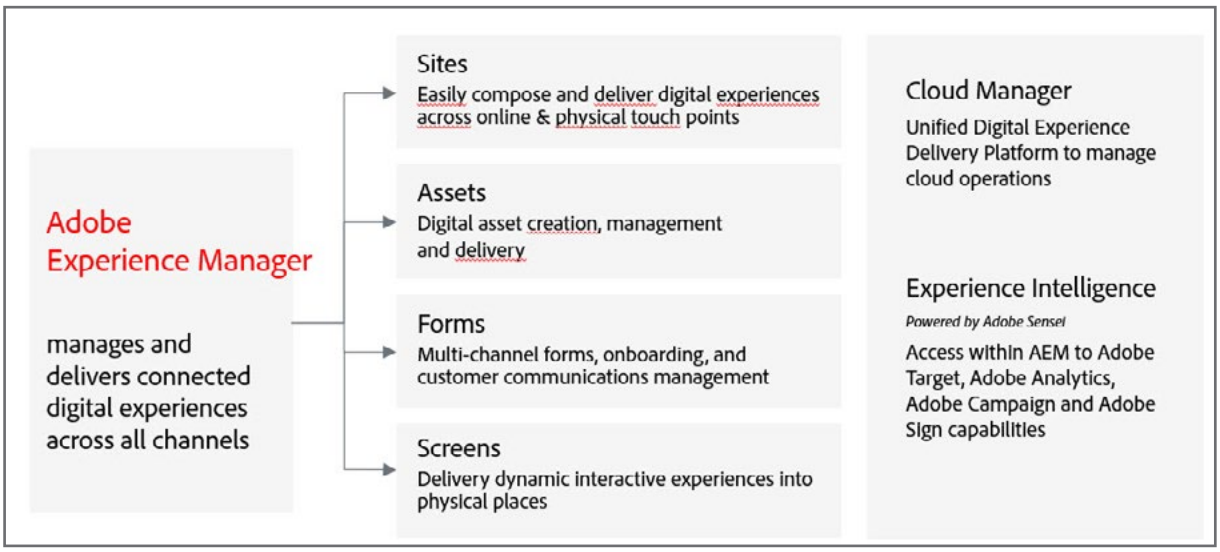
AEM Forms combine form authoring, management, and publishing along with correspondence management capabilities, document security, and integrated analytics to create engaging end-to-end experiences. Designed to work across web and mobile channels, AEM Forms can be efficiently integrated into your business processes, reducing paper processes and errors while improving efficiency.

Assets

AEM Assets is an application on the AEM Platform that allows our customers to manage their digital assets (images, videos, documents and audio clips) in a web-based repository. AEM Assets includes Metadata-support, Renditions, the Digital Asset Management Finder and the AEM Assets Administration UI.

Cloud Manager

Cloud Manager enables organizations to self-manage Experience Manager in the cloud. It includes a continuous integration and continuous delivery (CI/CD) framework that lets IT teams and implementation partners expedite the delivery of customizations or updates without compromising performance or security.



AEM Architecture

AEM as a Cloud Service introduces the next generation of the AEM product line. It builds on top of two decades of constant investments, preserving and extending all AEM use cases and functionalities, while turning the product into a cloud native solution.

AEM is a web-based client-server system, made up of several infrastructure-level and application-level functions. These functions are used to build relevant applications.

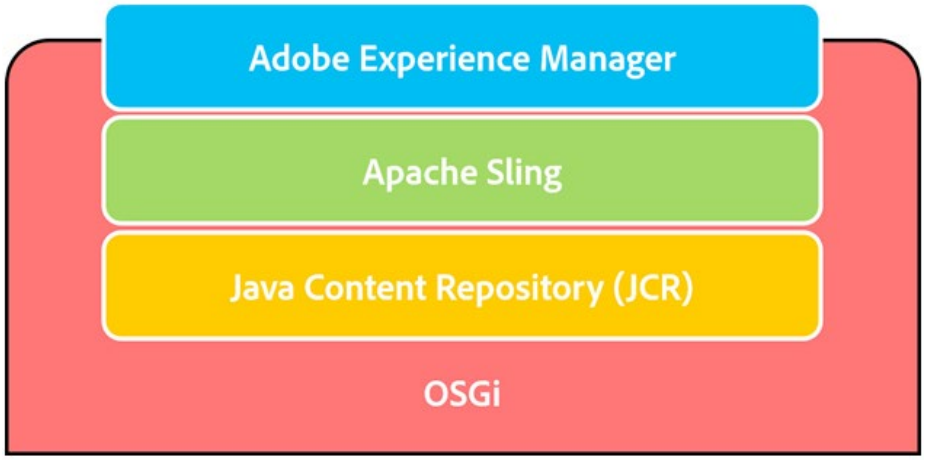
To learn more about AEM Architecture, go to:

https://docs.adobe.com/content/help/en/experience-manager-cloud-service/core-concepts/architecture.html.

Basics of AEM Architecture Stack

AEM architecture stack is based on technologies such as OSGi, Java Content Repository (JCR), and Apache Sling.

The following diagram depicts a high-level view of the AEM architecture stack:



Granite Platform

Granite is a general-purpose platform for building robust scalable applications. It is Adobe's open web stack and forms the technical foundation on which AEM is built. Granite supports an open architecture, which is based on both open standards (JCR and OSGi) and open source projects (Apache Sling and Apache Jackrabbit).

Note: Granite is an open development project within Adobe but not an open source project.

Technically, at the core, Granite provides:

- **An application launcher** for a standalone Java or Web application archive for deployment in the existing servlet containers or application servers.
- An **OSGi Framework** into which all applications are deployed.
- **OSGi Compendium Services** to support building applications, such as Log Service, Http Service, Event Admin Service, Configuration Admin Service, Declarative Services, and Metatype Service.
- A comprehensive **Logging Framework** providing various logging APIs, such as SLF4J, Log4F, Apache Commons Logging, and OSGi Log Service.
- A repository based on **Apache Jackrabbit Oak** and **JSR-283**.
- The **Apache Sling Web Framework**.

Granite UI

The consoles in the main navigation, tools, and editors of AEM are built using the Granite UI. The Granite UI provides a foundational UI framework for:

- UI widgets
- Extensible and plug-in-based admin UI

It also adheres to the following requirements:

- Mobile first (designing an online experience for mobile before designing it for the desktop)
- Extensible
- Easy to override

Note: The Granite UI is based on Coral 3, which is Adobe's universal UI across all products.

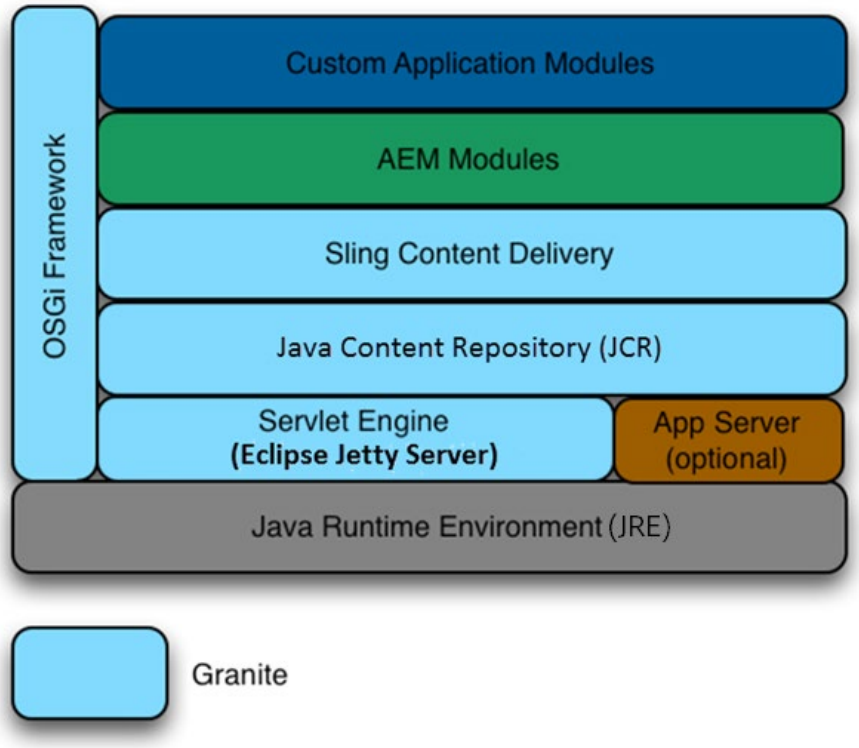
OSGi Framework

OSGi enables a collaborative and modular environment, where each application may be built and implemented as a small bundle. Each bundle is a collection of tightly coupled, dynamically loadable classes, JAR files, and configuration files that explicitly declare their external dependencies.

Note: OSGi used to stand for Open Service Gateway Initiative, but that name has been discontinued, and it is now officially no longer an abbreviation. It is just known in the industry as OSGi.

All content is stored in the content repository, which means backup is done at the repository level. OSGi runtime hosts Java applications that can access the repository by using the JCR API. As part of the application runtime, you get Apache Sling, a RESTful web application framework that exposes the full repository content using HTTP and other protocols.

The following diagram depicts the AEM platform foundation of Granite and the OSGi framework:



Apache Felix

Apache Felix is an open source implementation of OSGi for the AEM framework. It provides a dynamic runtime environment, where the code and content bundles can be loaded, unloaded, and reconfigured at runtime.

Note: You can learn more about OSGi, by visiting the link in the **References** section at the end of this module.

JCR

The JCR, specifically Java Specification Request-283 (JSR- 283), is a database that supports structured and unstructured content, versioning, and observation. In other words, it is a database that resembles a file system.

Note: The Adobe implementation of JSR-283 was known as the Content Repository eXtreme (CRX). Hence, you may see CRX in some tools and interfaces in AEM. However, the CRX as a feature is being phased out. In its place, AEM uses the Granite platform and Apache Jackrabbit Oak.

All data pertaining to AEM, such as HTML, HTML Template Language (HTL), CSS, JavaScript/Java, images, and videos are stored in the JCR object database. JCR is built with Apache Jackrabbit Oak, an open-source project.

The advantages of using JCR are:

- It provides a generic application data store for structured and unstructured content. While file systems provide excellent storage for unstructured and hierarchical content, the databases provide storage for structured data. This way, JCR provides the best of both the data storage architectures.
- It supports namespaces. Namespaces prevent naming collisions among items and node types that come from different sources and application domains. JCR namespaces are defined with a prefix, delimited by a single colon (:). For example, jcr:title. This means that this title property is defined in the jcr namespace.
- It provides one interface to interact with text and binary data. This helps with easy access and management of data in comparison to storing it in multiple places.

Note: You can learn more about JCR, by visiting the link in the **References** section at the end of this module.

Apache Sling

Apache Sling is a web application framework for content-centric applications and uses a JCR, such as Apache Jackrabbit Oak, to store and manage content.

The key features of Apache Sling include:

- It is Apache open source.
- It is based on REST principles and helps build applications as a series of OSGi bundles.
- It is resource-oriented (every resource has a URI) and maps to JCR nodes.

A request URL is first resolved to a resource, and then based on the resource, Apache Sling selects the Servlet or script to handle that request. Servlets and scripts are handled as resources and are accessible by a resource path. This means every script, Servlet, filter, and error handler is available from the Resource Resolver just like normal content—providing data to be rendered on request.

Note: You can learn more about Apache Sling, by visiting the link in the **References** section at the end of this module.

Infrastructure of AEM

When it comes to underlying infrastructure for AEM, there are three different solutions available, Cloud Service, Managed Services, and On-premise.

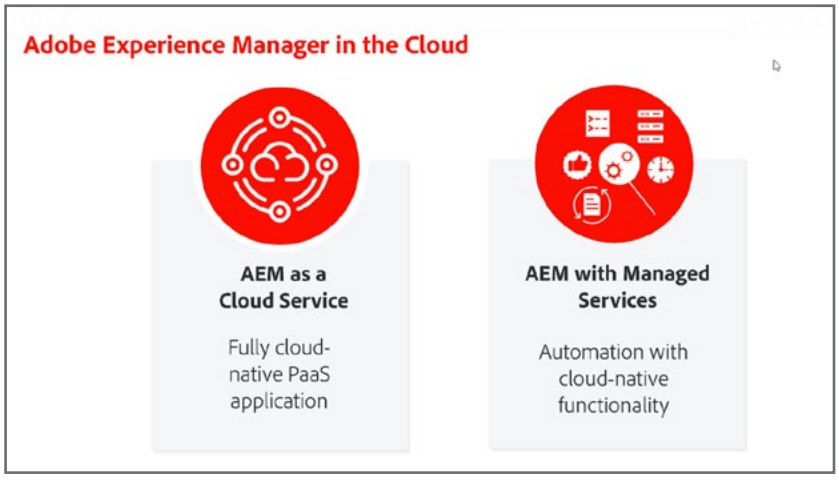
AEM as a Cloud Service

AEM as a Cloud Service is the cloud-native way of leveraging the AEM applications.
It enables you to provide your customers with personalized, content-led experiences, by combining the power of the AEM Content Management System with AEM Digital Asset Management. The solution has been entirely designed for the cloud and is scalable, secure, always available and up-to-date.

AEM as a Cloud Service has resulted in changes to the architecture. AEM as a Cloud Service now has a dynamic architecture with a variable number of AEM images.

The architecture of AEM as a Cloud Service:

- Is scaled based on the actual traffic and actual activity.
- Has individual instances that only run when needed.
- Uses modular applications.
- Has an author cluster as default; this avoids downtime for maintenance tasks.



Managed Services

AEM Managed Services is a complete solution for Digital Experience management. It provides benefits of experience delivery solution in the cloud while retaining all the control, security and customization benefits of an on-premise deployment. AEM Managed Services enables customers to launch faster by deploying on the cloud and also by leaning on the best practices and support from Adobe. Organizations and business users can engage customers in minimal time, drive market share, and focus on creating innovative marketing campaigns while reducing the burden on IT.

AEM On-Premise

AEM deployed and managed in your corporate environment. You can install AEM on servers in your Corporate environment. Typical installation instances include: Development, Testing and Publishing environments.

References

You can use the following links for more information on:

- Architecture:
  - Dispatcher in the Cloud: https://docs.adobe.com/content/help/en/experience-manager-cloud-service/implementing/dispatcher/overview.html
  - Manage Environments - Cloud Service: https://docs.adobe.com/content/help/en/experience-manager-cloud-service/implementing/using-cloud-manager/manage-environments.html
  - AEM as a Cloud Service Architecture: https://docs.adobe.com/content/help/en/experience-manager-cloud-service/core-concepts/architecture.html
  - OSGi: http://www.osgi.org
  - JCR 2.0 Specification: http://jackrabbit.apache.org/jcr/jcr-api.html
  - JSR-283: https://jcp.org/en/jsr/detail?id=283
  - Apache Sling: https://sling.apache.org/
  - Apache Felix: https://felix.apache.org/