



adaptTo()

EUROPE'S LEADING AEM DEVELOPER CONFERENCE

27th – 29th SEPTEMBER 2021

Designing a cluster-aware application

Jörg Hoh, Adobe



What is this talk about?

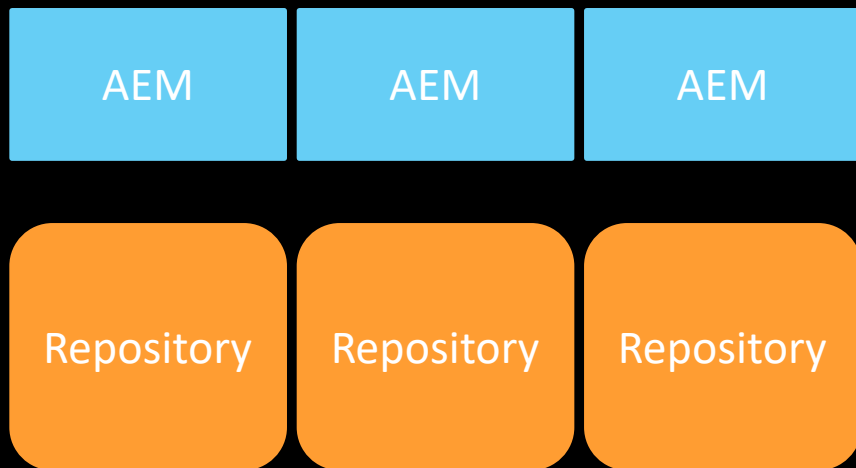
- Details of AEM clustering
- Its direct impact on common APIs and application patterns
- How to reflect this in your application



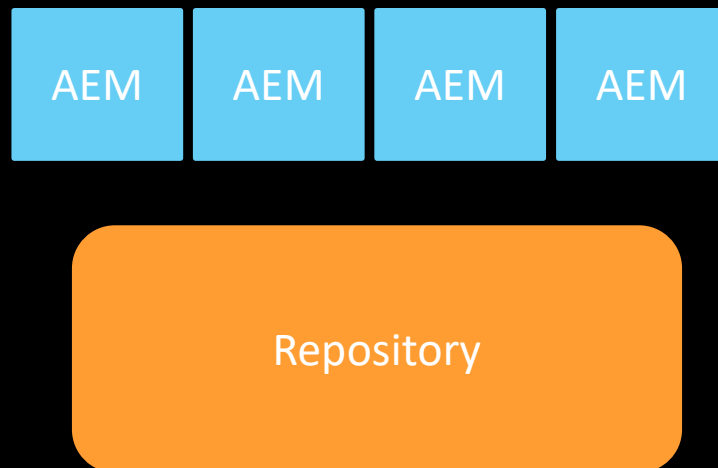
- Jörg Hoh, @joerghoh
- 10+ years experience with AEM/CQ5
- SRE @ Adobe

Farm vs Cluster (logical view)

Farm



Cluster

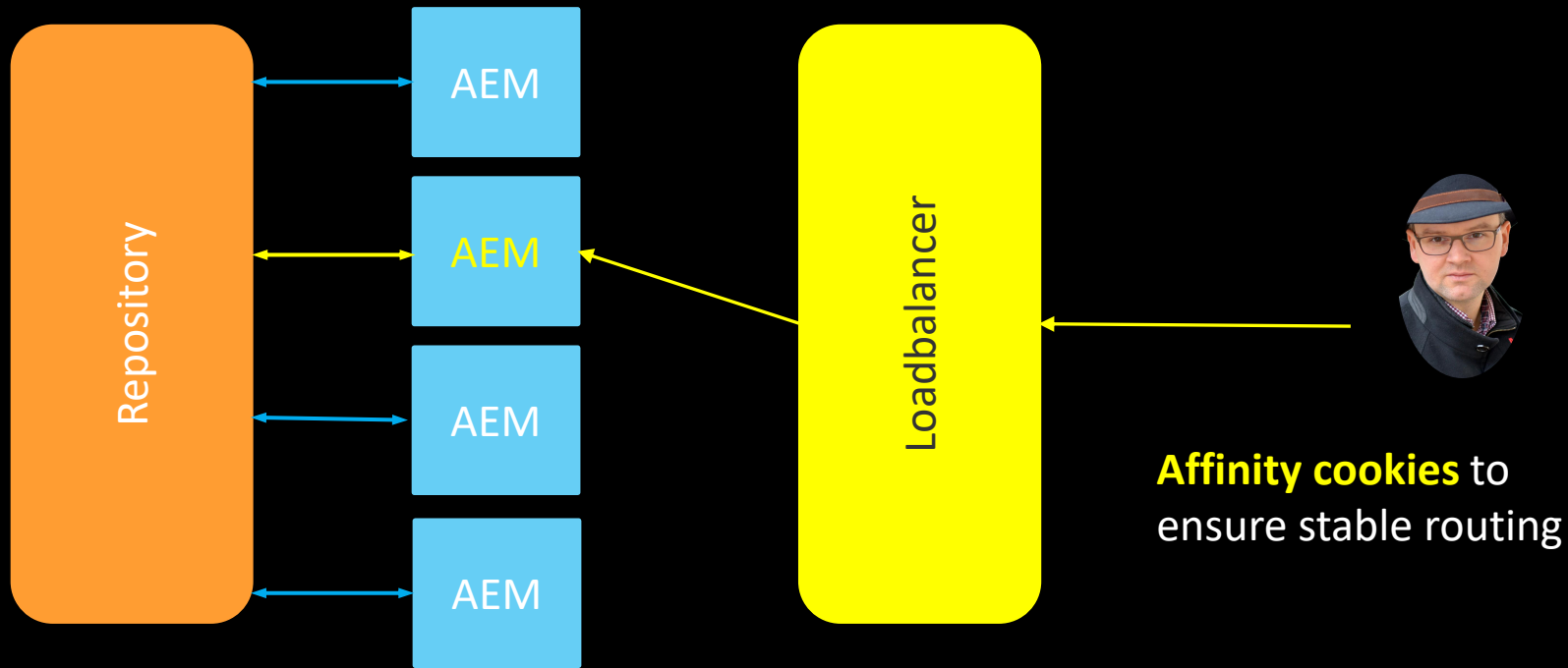


Specifics of a shared repository

- Multiple AEM instances read and write to the same repository.
- Changes made in a single cluster node can trigger changes in other cluster nodes.
- Eventually consistent

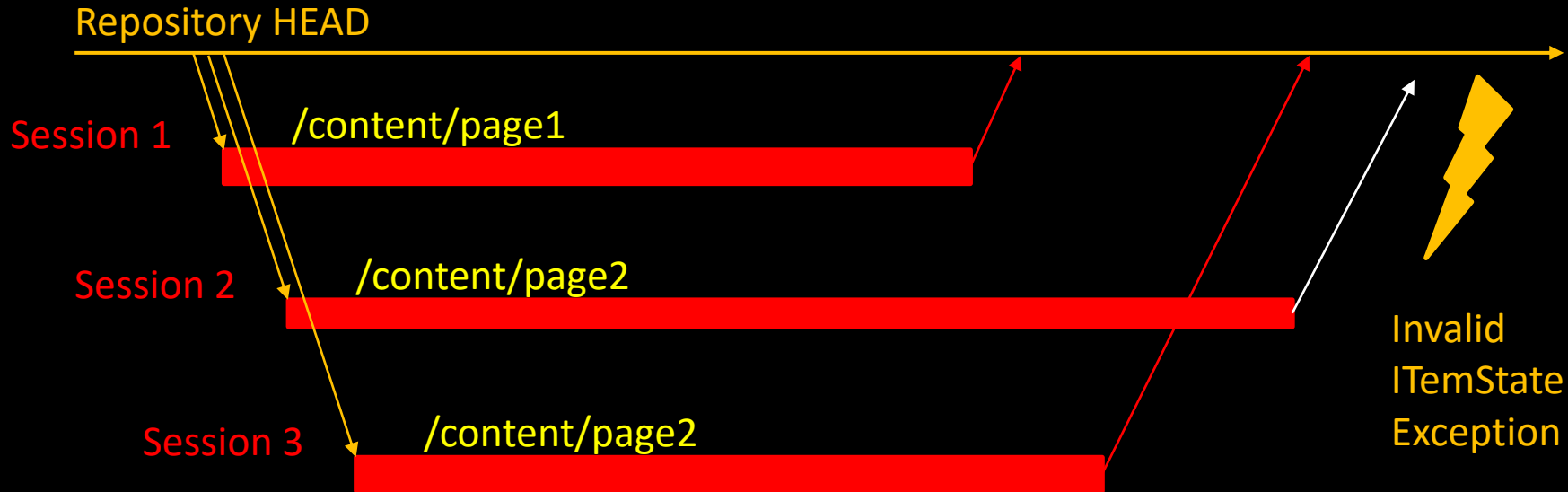
Eventual consistency

Sync delay up to 2 sec

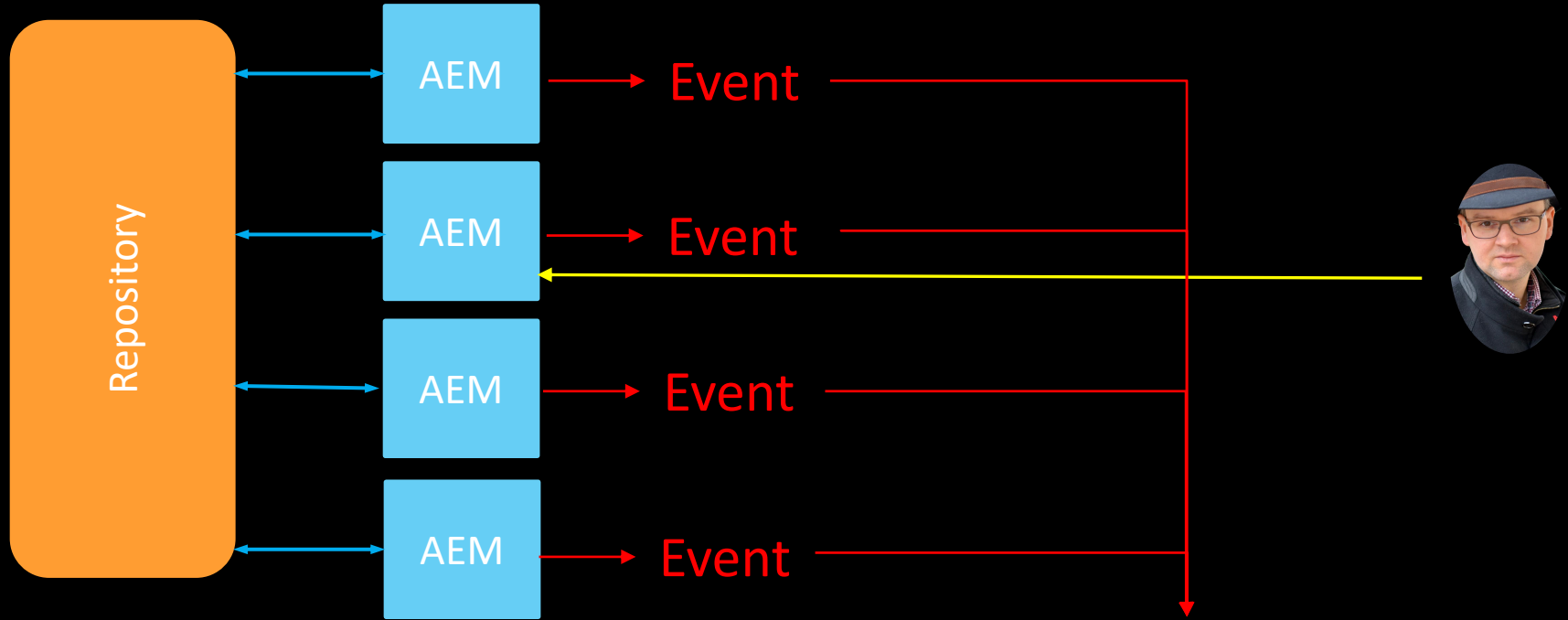


- Same rules** as in non-clustered AEM instances
- MVCC pattern: Concurrent updates get not visible during the runtime of a session unless you invoke `refresh()`
 - Expect `InvalidItemStateExceptions` when modifying nodes concurrently

Same rules as in non-clustered AEM instances

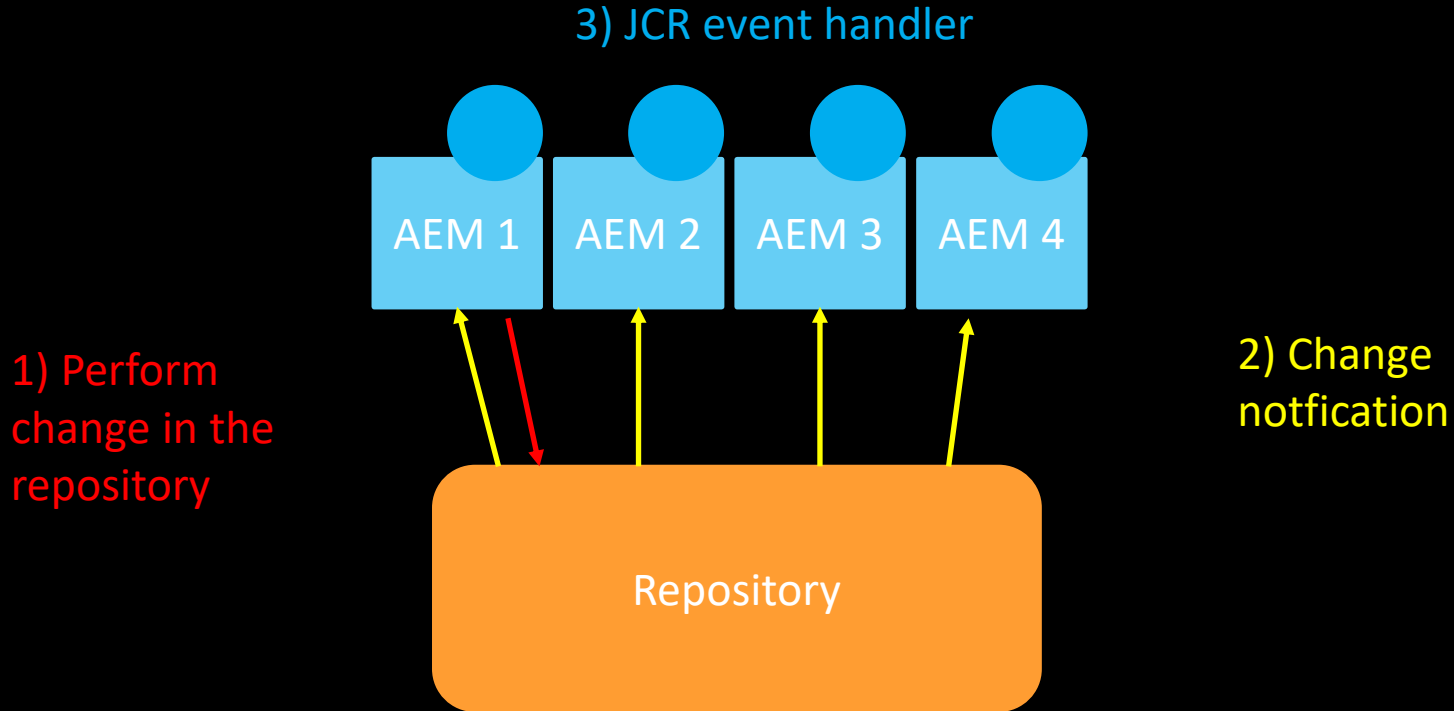


One event -- multiple event handlers



- By default you see all changes in the repository, including changes made on other cluster instances.
- You might handle the same event n times.
- Avoid any assumption that a local change has triggered this event.

JCR Observation Listener



JCR Observation – receive only local events

```
JackrabbitEventFilter ef = new JackrabbitEventFilter()  
    .setAbsPath("/content/mysite")  
    .setNodeTypes(new String[]{"cq:Page"})  
    .setEventTypes(Event.NODE_ADDED)  
    .setIsDeep(true)  
    .setNoExternal(true);  
  
JackrabbitObservationManager om =  
    (JackrabbitObservationManager)  
    session.getWorkspace().getObservationManager();  
om.addEventListener(this, ef);
```

Sling ResourceChange Event Listener

- **Abstracted JCR Observation**
- **Just local events: implement the**
`ResourceChangeListener` **interface**
- **All events: implement the**
`ExternalResourceChangeListener`
interface

- Normally used just locally.
- Distributed events possible, but rarely used.
- Mark events as distributable by adding the property `event.distribute` to the event properties.

- Nothing has changed.
- Exactly once guarantee

AEM workflows

- Luckily, AEM takes care of that.
- Workflows can be invoked on any node, but are executed only on the cluster leader.

- Each clusternode has its own scheduler.
- Support to run only once in a cluster (property “scheduler.runOn=LEADER”)
 - These jobs will only start on the cluster leader.

Usecase: cache

- In memory caches must always reflect the current state of content in the repository.
- It must not be maintained by the code modifying this content, but only by JCR Observation / ResourceChangeListener

Usecase: Execute a task exactly once

- **Easiest:** When there is a triggering action, let this action create a Sling Job or workflow.
- **Scheduled job** (“scheduler.runOn=LEADER”)
- **Trigger it externally via a request**

1 change -> multiple events

- Check all your event handlers!
- Do you have code, which needs to run exactly once?

Eventual consistency

- Respect the affinity cookie!

Thank you

@joerghoh

<https://cqdump.joerghoh.de>