adaptTo()

APACHE SLING & FRIENDS TECH MEETUP
BERLIN, 22-24 SEPTEMBER 2014

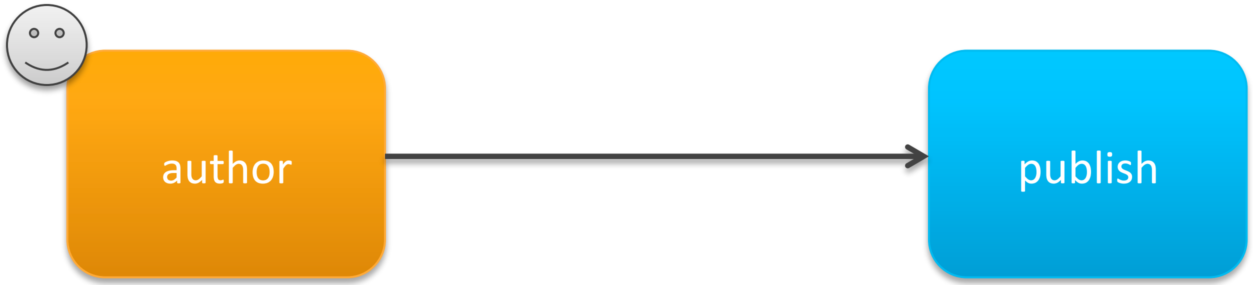
Data replication in Sling

Tommaso Teofili, Adobe Systems

Use cases

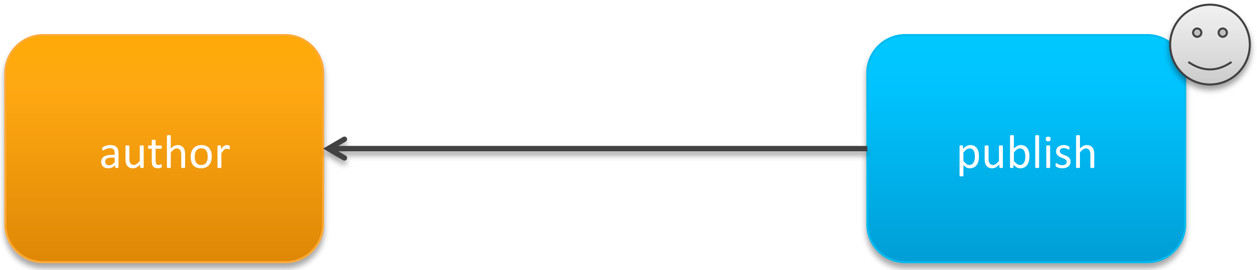
Replication in AEM

- Moving authored content to publish servers



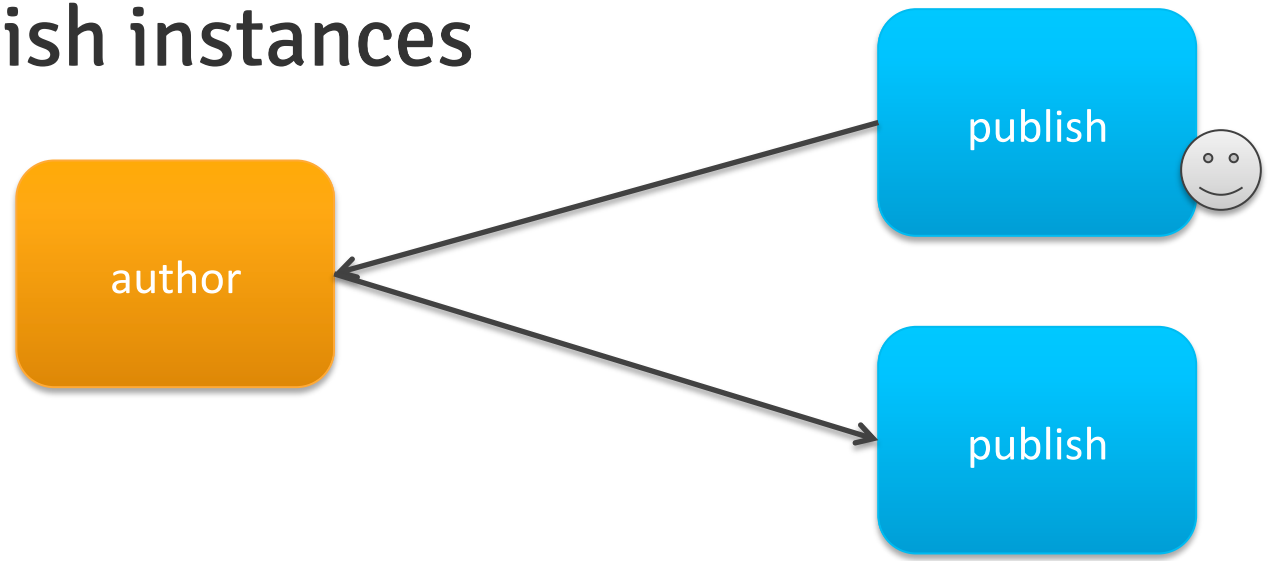
Replication in AEM

- Moving user generated content back to author for moderation



Replication in AEM

- Moving user generated content to other publish instances



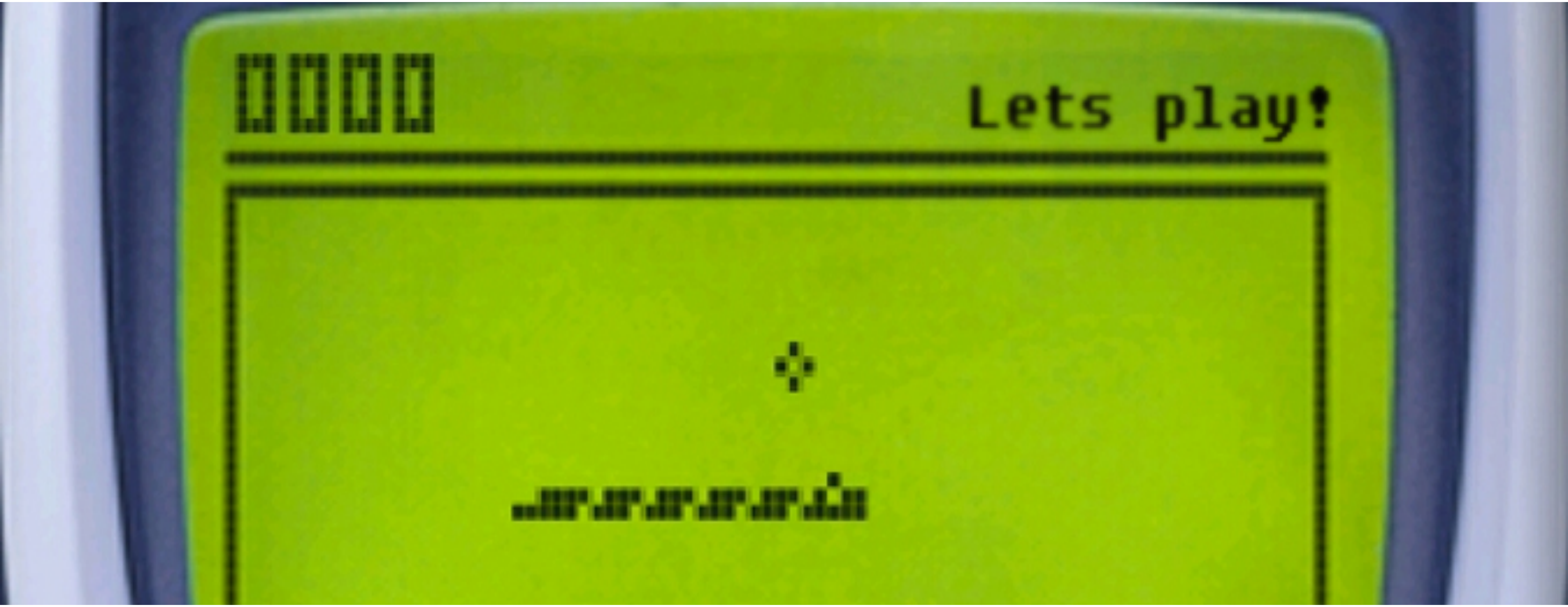
Replication module should ...

- transfer resources between Sling instances
  - push from server A to server B
  - pull from server C to server D

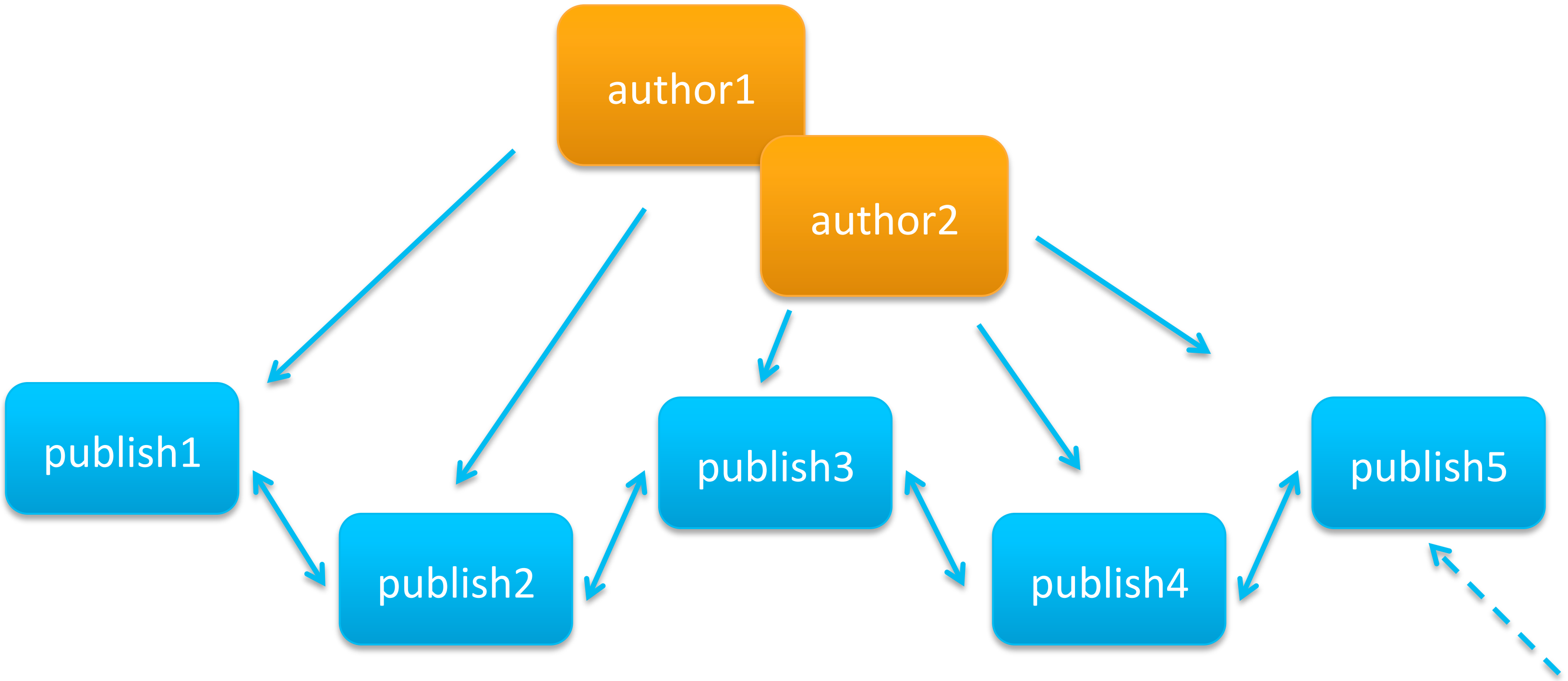
Example: replication of /foo/bar

- HTTP request for pushing /foo/bar
  - Resources get
    - packaged
    - sent
    - received
    - persisted

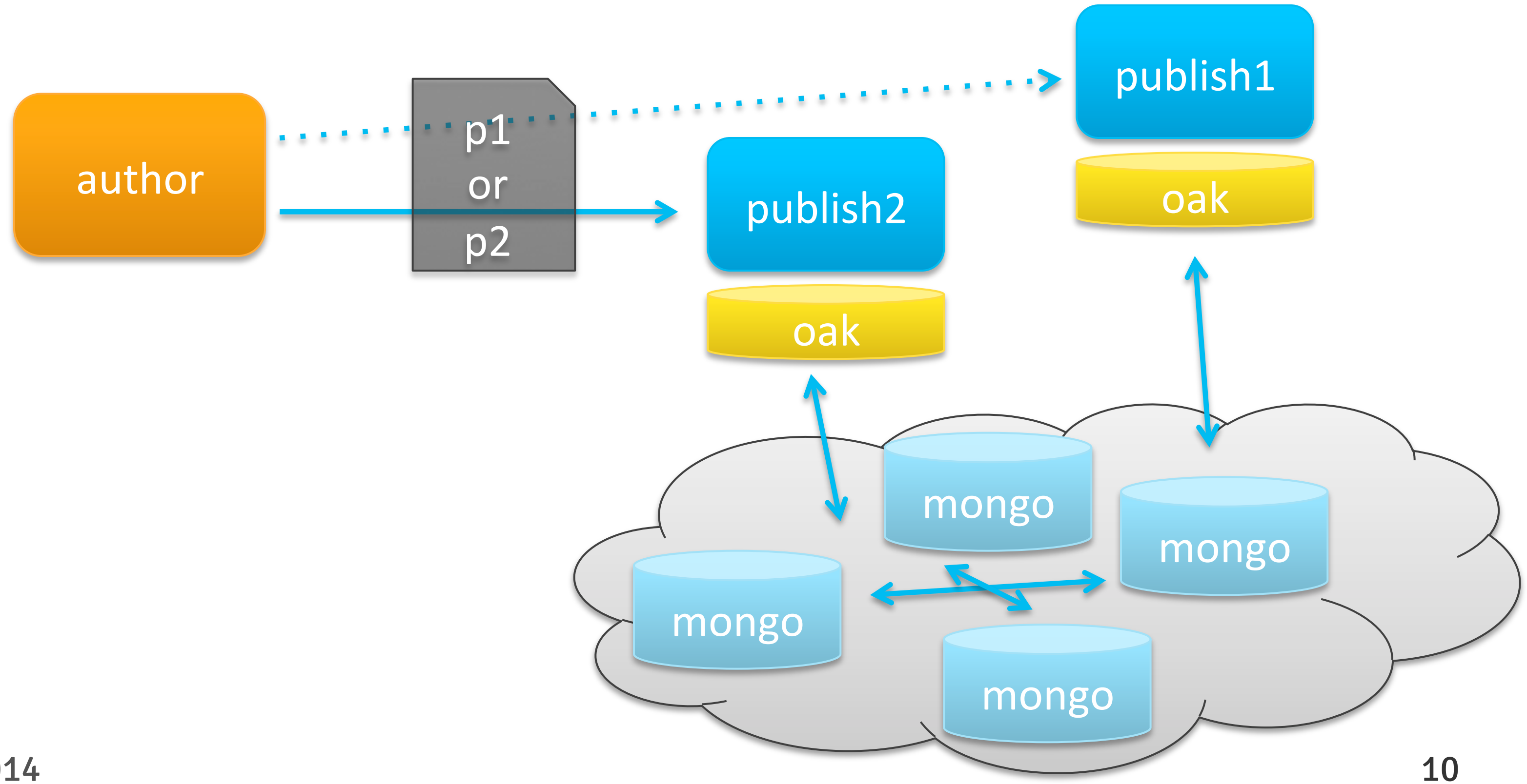
we’re not in 1999 ... anymore



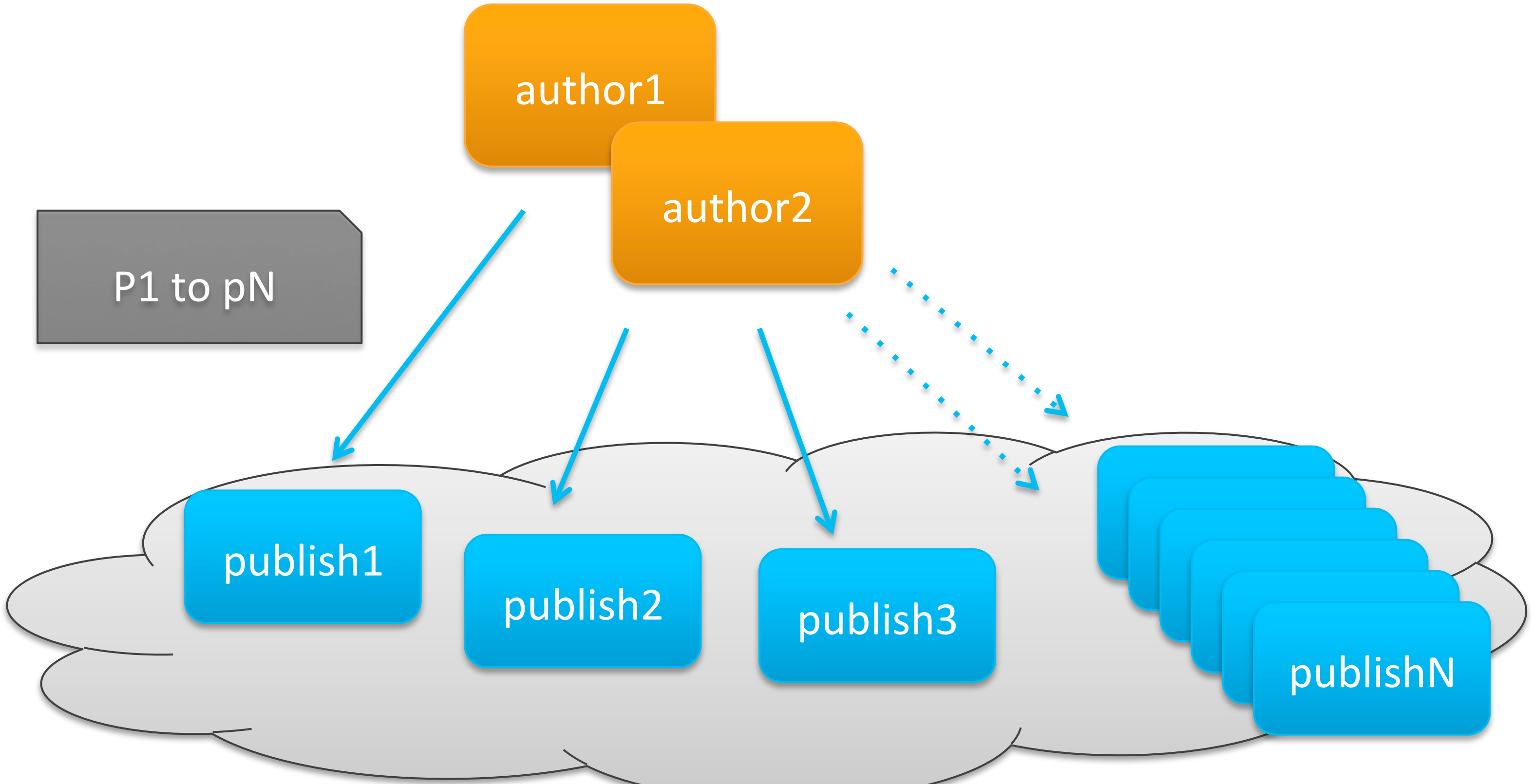
NRT sync on N servers



Oak instances served by same MongoDB



Cloud based infrastructures



Overview

Sling Replication

- Contributed to Sling in November 2013
- First release ?
- Main goals
  - Simple
  - Resilient
  - Fast

Replication agents

- Execute replication requests by:
  - exporting replication packages from a (remote) Sling instance
  - importing replication packages into a (remote) Sling instance

Replication agents

- A “push” agent has
  - A “local” exporter
    - Creating a package locally for the resources to be replicated (e.g. from the underlying JCR repo)
  - A “remote” importer
    - Importing the exported package remotely by sending it to a designated endpoint to persist it

Replication agents

- A “pull” agent has
  - A “remote” exporter
    - Pulling a package from a remote Sling instance designated endpoint
  - A “local” importer
    - Importing the package locally by persisting it into the Sling instance

Replication agents

- A "coordinating" agent has
  - A "remote" exporter
    - Pulling a package from a remote endpoint
  - A "remote" importer
    - Importing the package remotely into a Sling instance

Replication agents

- A “queuing” agent has
  - A “local” exporter
    - Creating a package locally for the resources to be replicated (e.g. from the underlying JCR repo)
  - No importer

Replication package serialization

- Payload to be sent / received
  - Package builders for (de)serialization
    - Jackrabbit FileVault based package builder

Replication queues

- Multiple queue providers
  - Sling Jobs based (sling.event bundle)
  - In memory
- Multiple queue distribution strategies
  - Single
  - Error aware
  - Priority

adaptTo()

Replication rules

- Rules can be defined within agents
  - To trigger replications upon resource changes
  - To schedule periodic replications
  - To chain replicate
  - …

Accessing replication resources

- Agents as OSGi services
- Can be defined via ConfigurationAdmin
- Resource providers control
    - Service access
    - CRUD operations on configs

Agent configuration example (1/3)

- Config for a ‘push agent’

```
{
    "jcr:primaryType" : "sling:OsgiConfig",
    "name" : "publish",
    "type" : "simple",
     "packageExporter": [
        "type=local”,
        "packageBuilder/type=vlt",
        ...
     ],
...
```

Agent configuration example (2/3)

```
...
   "packageImporter" : [
       "type=remote",
       "endpoints[0]=http://.../replication/services/importers/default”,
       "authenticationFactory/type=service",
       "authenticationFactory/name=user",
       ...
       "packageBuilder/type=vlt",
       ...
   ],
 ...
```

Agent configuration example (3/3)

```
  ...
    "queueProvider" : [
        "type=service",
        "name=sjh"
    ],
    "queueDistributionStrategy" : [
        "type=service",
        "name=error"
    ]
}
```

Anatomy of a forward replication request

Anatomy of a fwd. replication (1/4)

- HTTP POST on /libs/sling/replication/service/agents/publish with form parameters
  - Action = ADD
  - Path = /content/replication
- Agent with name ‘publish’ gets picked up by the agent resource provider

Anatomy of a fwd. replication (2/4)

- Agent ‘publish’ calls its Exporter (local) to create the payload
  - the ‘local’ exporter calls its Package builder
  - the ‘vlt’ package builder creates a FileVault package for resources under /foo/bar

Anatomy of a fwd. replication (3/4)

- Agent ‘publish’ dispatches the package to its queue provider and distribution algorithm
  - The queue provider is asked to provide queues depending on the distribution strategy
  - The request gets queued

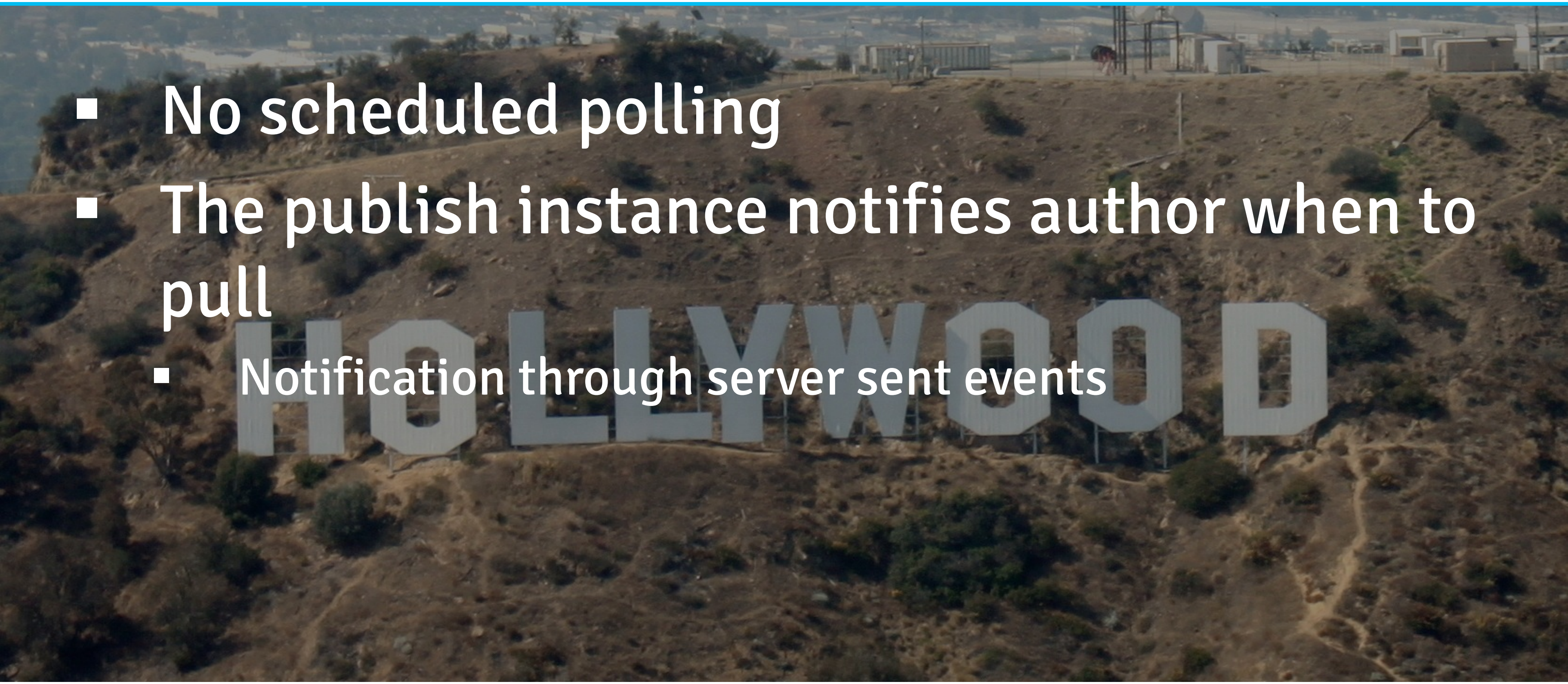
Anatomy of a fwd. replication (4/4)

- When queue entry gets processed the (remote) importer is called
  - Package sent over the wire to an endpoint bound to a local importer on the receiving server
  - The local importer on the receiving side deserializes and persists the replication package via its package builder

Reverse replication

- Pull agent on author
  - Periodically polling
- Queuing agent on publish

Event based reverse replication



- No scheduled polling
- The publish instance notifies author when to pull
  - Notification through server sent events

Looking forward

- Get information on other instances
  - e.g. replicate to all instances with run mode ‘xyz’
- NRT publish sync
  - Coordinate agents++
- Performance

adaptTo()

Thanks!

- Questions?