RADU POPESCU

A HIGH-LEVEL PUBLICATION INTERFACE FOR CERNVM-FS

MOTIVATION

- The CernVM-FS server tools offer low-level primitives, do not cover publication workflow
- Using multiple publishers (release managers) comes with added complexity
- Any higher-level tooling is project (user) specific, leading to duplication
- There is interest in a repository API (get status of specific publication, check replication status etc.)

OBJECTIVES

- Put together a set of generic tools to manage publishing to CernVM-FS, offering a higher-level abstraction based on publication jobs
- Try to encapsulate the publishing system (user shouldn't need to know what is inside)
- Minimise amount of extra infrastructure required (ideally no new VMs)
- Should be easy to integrate with existing setups

Draw heavily from LHCb's nightly build publishing system

ANATOMY OF A CVMFS PUBLICATION

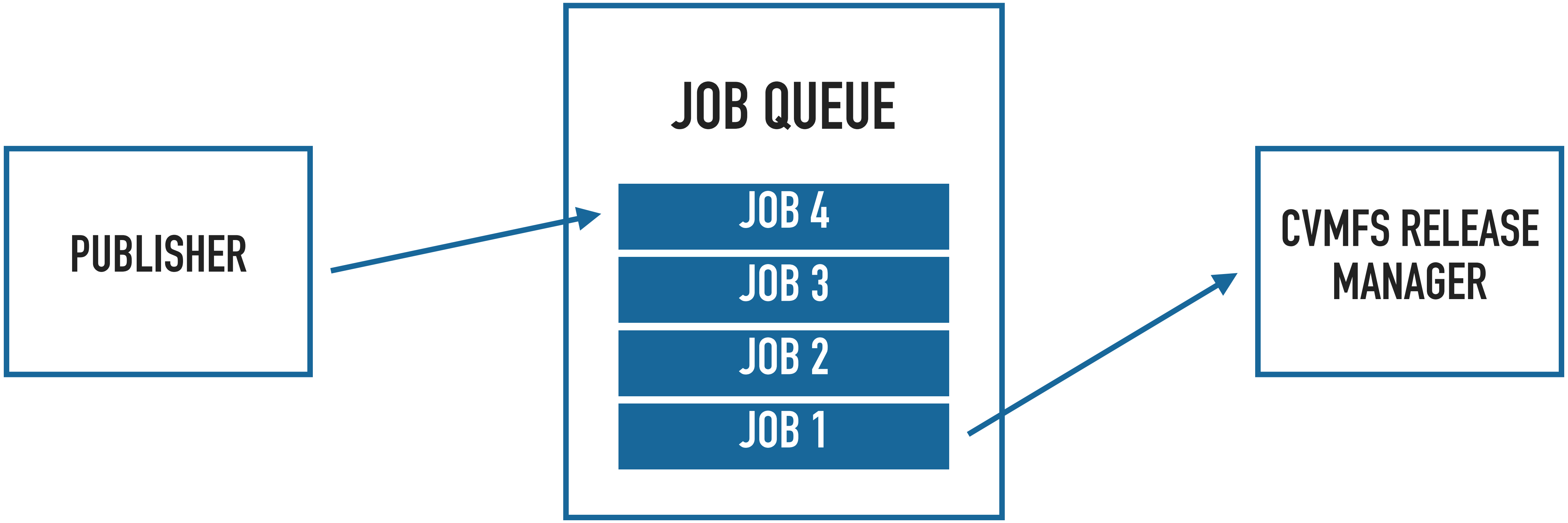
Login to release manager	`# ssh lxcvmfs93.cern.ch`
Open a transaction	`# cvmfs_server transaction sft.cern.ch/some/path`
Add, remove, or modify files	`# cp /some/files /cvmfs/sft.cern.ch/some/path`
Run some touch-up scripts	`# /cvmfs/sft.cern.ch/post-install.sh`
On success, commit the transaction	`# cvmfs_server publish sft.cern.ch`
On error, abort the transaction	`# cvmfs_server abort sft.cern.ch`

JOB DESCRIPTION

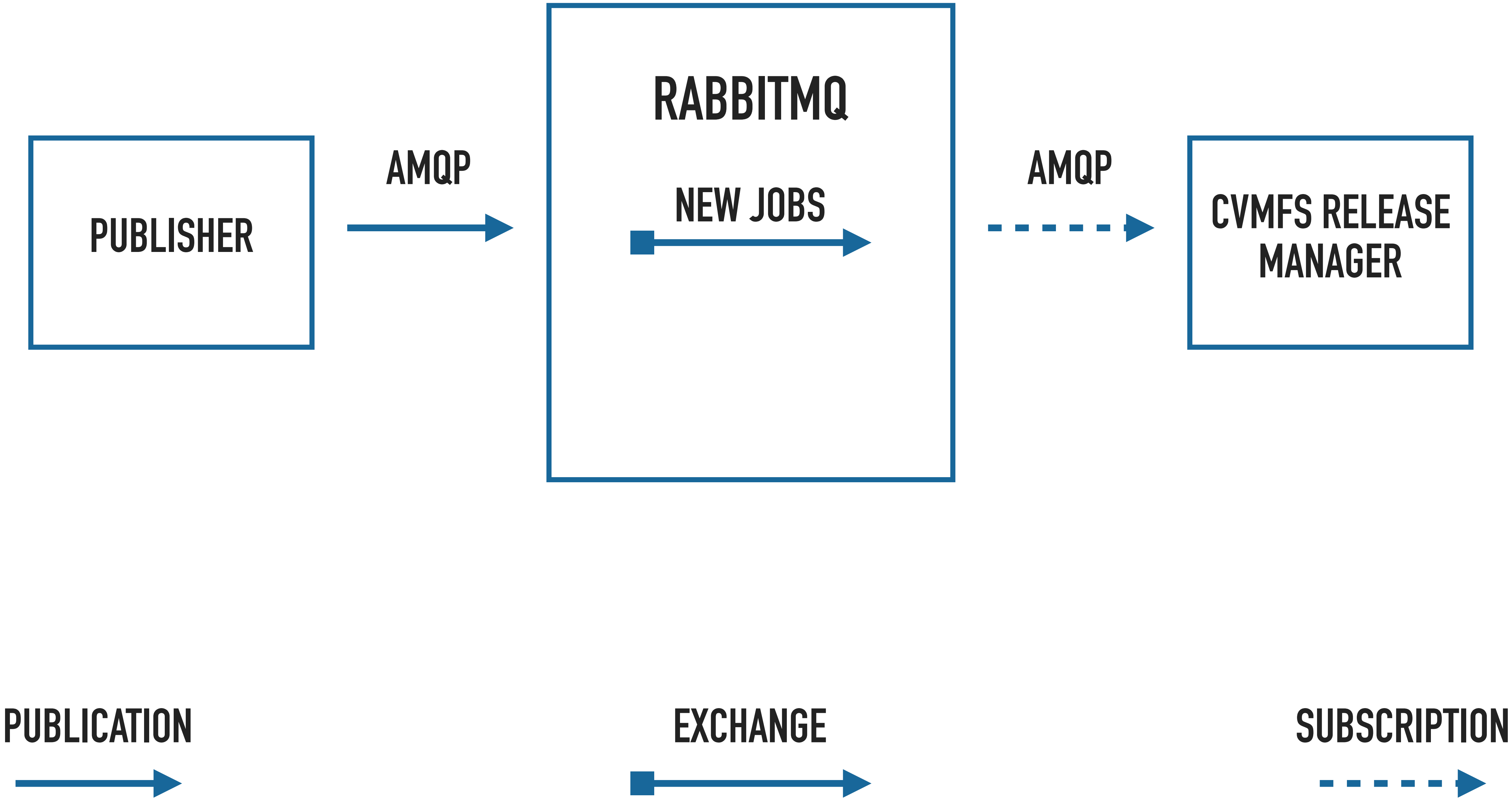
At minimum, we need:

- Repository
- Subpath
- Payload (URL)
- Script

CONCEPTUALLY…



IMPLEMENTATION



PUBLISHER

AMQP

RABBITMQ

NEW JOBS

AMQP

CVMFS RELEASE MANAGER

PUBLICATION

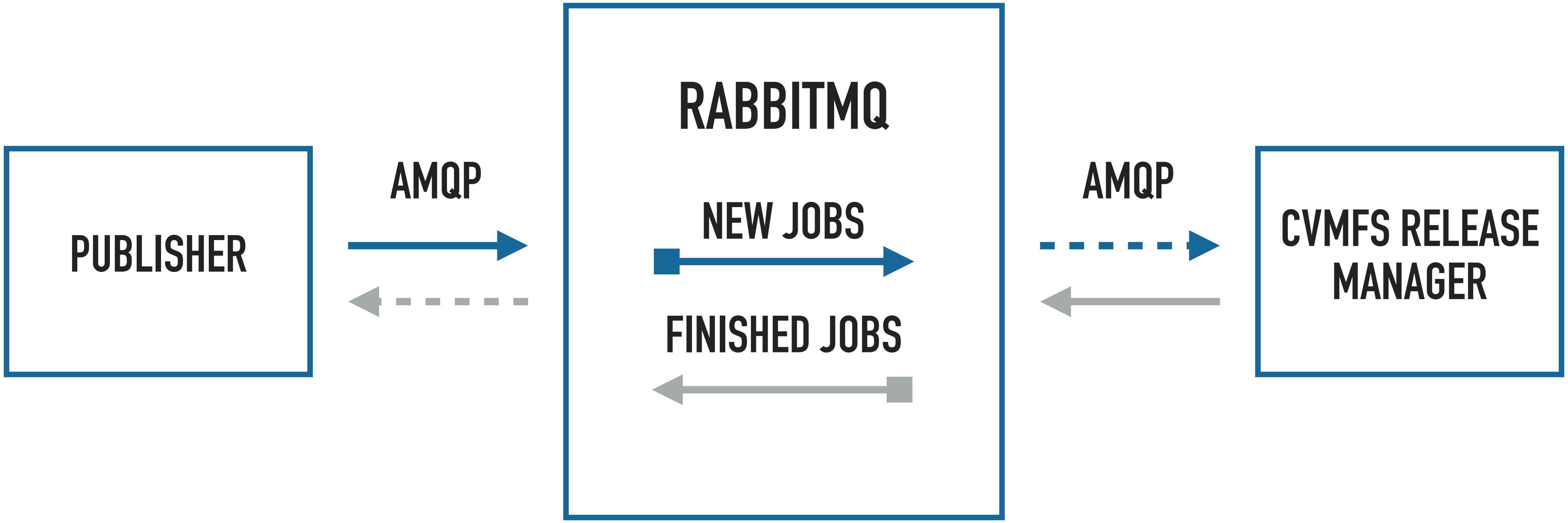
EXCHANGE

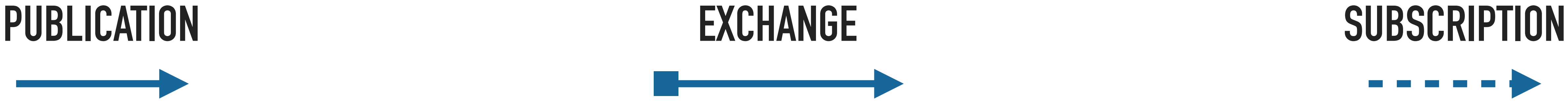
SUBSCRIPTION

ADD UUID TO JOB DESCRIPTION

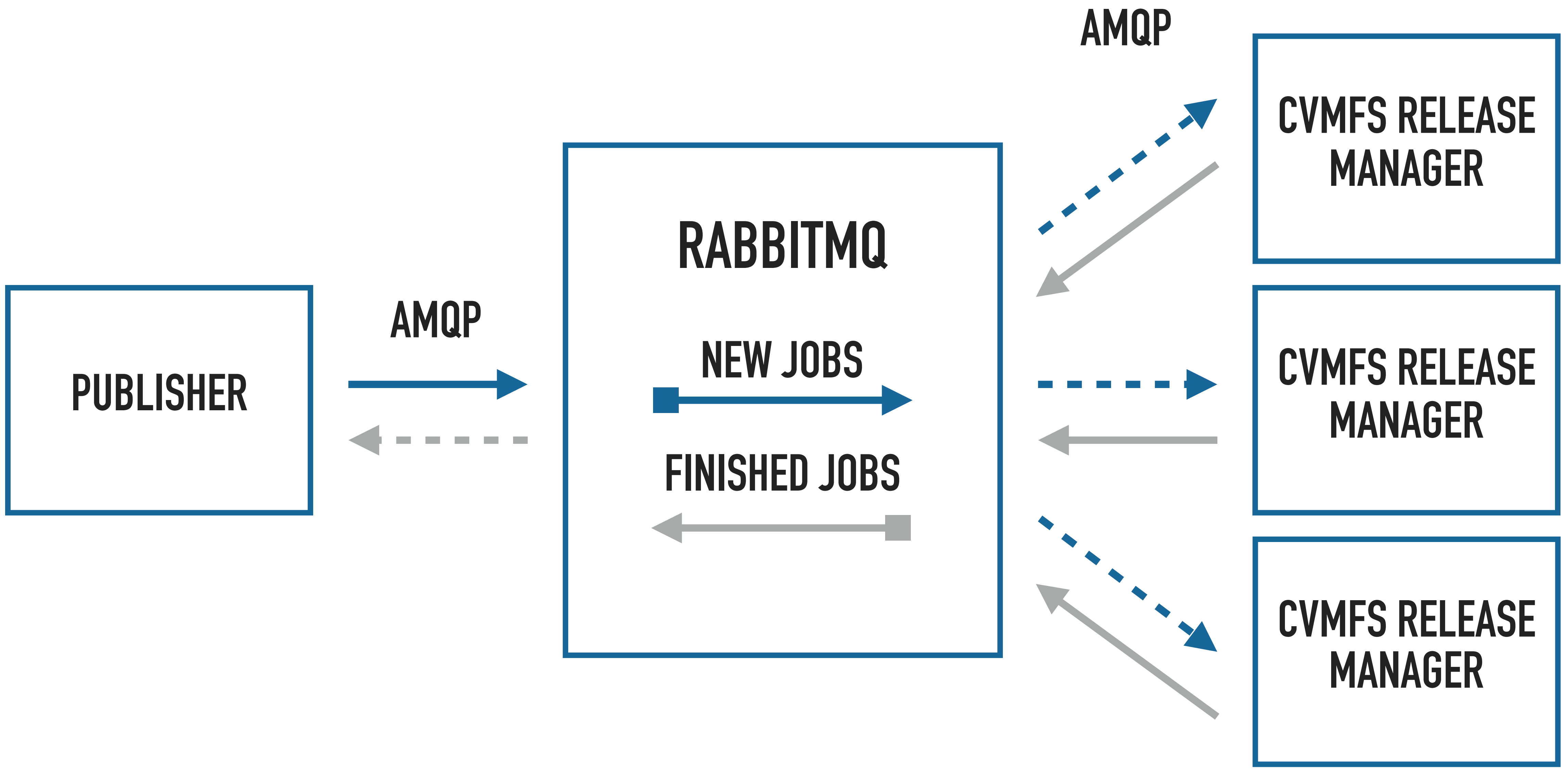
- Repository
- Subpath
- Payload (URL)
- Script
- **UUID** (ex: e7b67a20-f61d-11e8-a00e-7200045cde30)

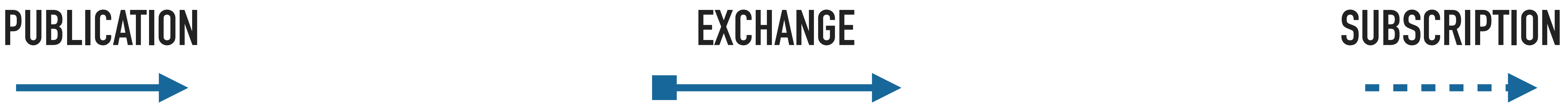
WAITING FOR PUBLISHING TO FINISH?





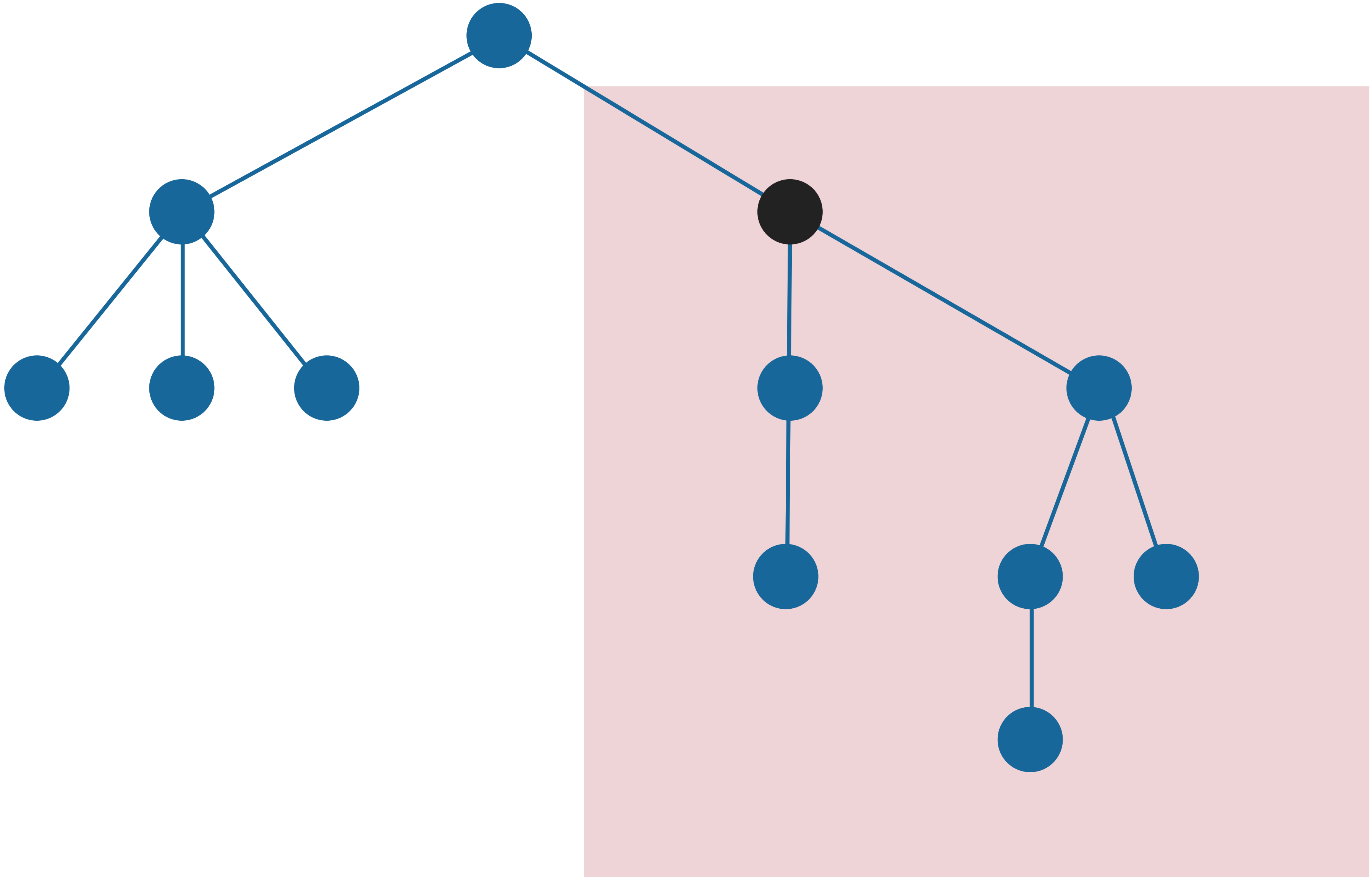
MULTIPLE RELEASE MANAGERS





DEPTH OF THE LEASE PATH

- Taking a lease too high in the repository is bad for parallelism



DEPTH OF THE LEASE PATH (EXAMPLE)

A typical publication in sft.cern.ch:

- Publish a number of packages under:
  `lcg/releases/<RELEASE_NUM>/<PACKAGE_NAME>/<PACKAGE_VER>`
- Update view at:
  `lcg/views/<RELEASE_NUM>/<PLATFORM>/…`
- Entire **`lcg`** subpath is locked
- Split into multiple transactions, publish packages in parallel

DEPTH OF THE LEASE PATH (EXAMPLE)

A typical publication in sft.cern.ch:

- Publish a number of packages under:
  `lcg/releases/<RELEASE_NUM>/<PACKAGE_NAME>/<PACKAGE_VER>`
- Update view at:
  `lcg/views/<RELEASE_NUM>/<PLATFORM>/…`
- Entire **`lcg`** subpath is locked
- Split into multiple transactions, publish packages in parallel
- **Problem**: updating the view depends on packages being published

ADD DEPENDENCIES TO JOB DESCRIPTION

- Repository
- Subpath
- Payload (URL)
- Script
- UUID
- **Dependencies** (list of UUID)

WAITING FOR DEPENDENCIES

- Subscribe to messages from the “Finished Jobs” exchange
- Before starting a job, wait for all its dependencies to finish

WAITING FOR DEPENDENCIES

- Subscribe to messages from the “Finished Jobs” exchange
- Before starting a job, wait for all its dependencies to finish

Problem: If we subscribe too late, some messages will not ever be received

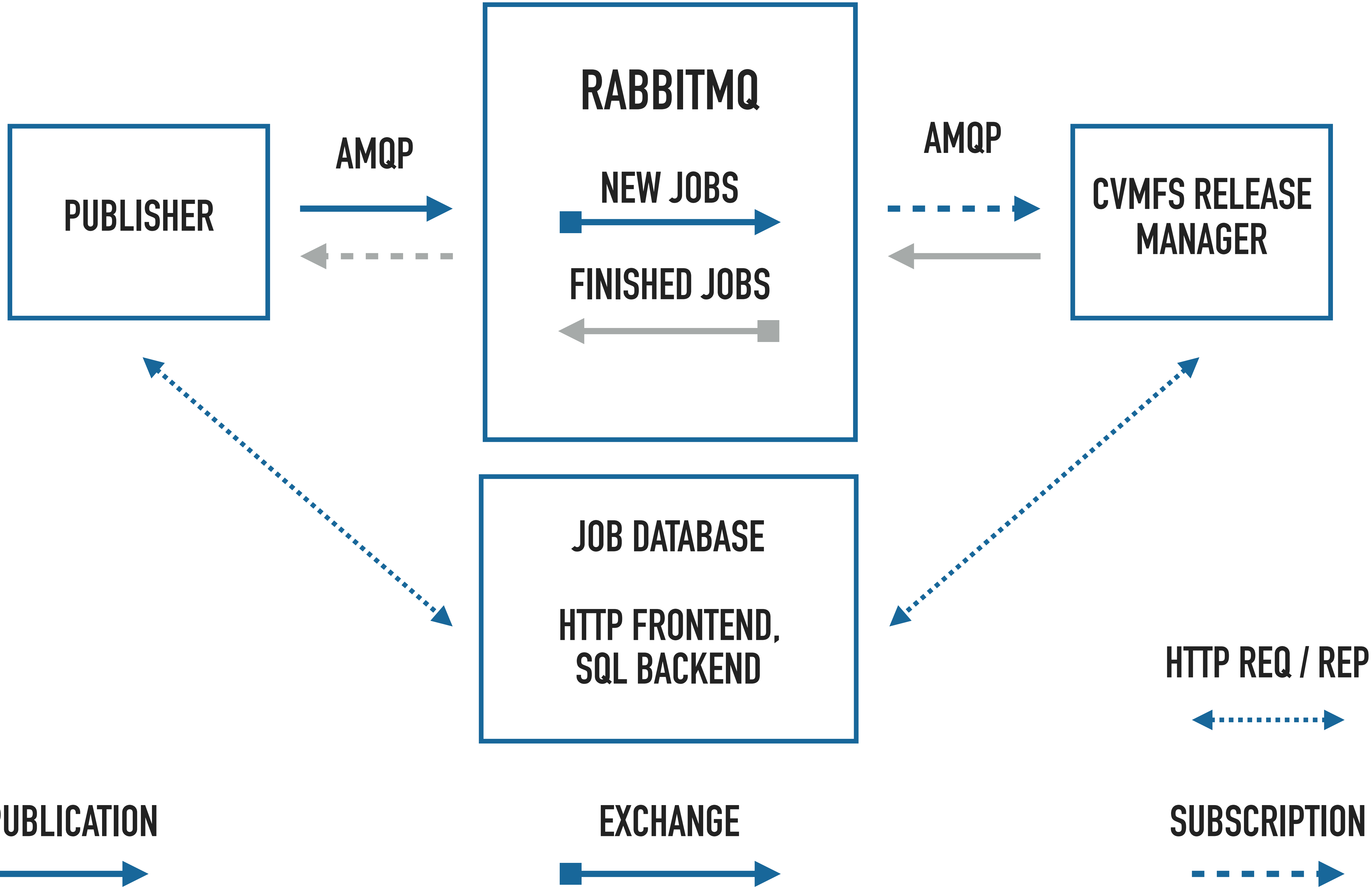
WAITING FOR DEPENDENCIES

- Subscribe to messages from the “Finished Jobs” exchange
- Before starting a job, wait for all its dependencies to finish

Problem: If we subscribe too late, some messages will not ever be received

Solution: Add a persistent database for finished jobs (successful and failed). Use both notifications and database queries to resolve status of dependencies

JOB DATABASE



PUBLISHER

AMQP

RABBITMQ

NEW JOBS

FINISHED JOBS

AMQP

CVMFS RELEASE MANAGER

JOB DATABASE

HTTP FRONTEND, SQL BACKEND

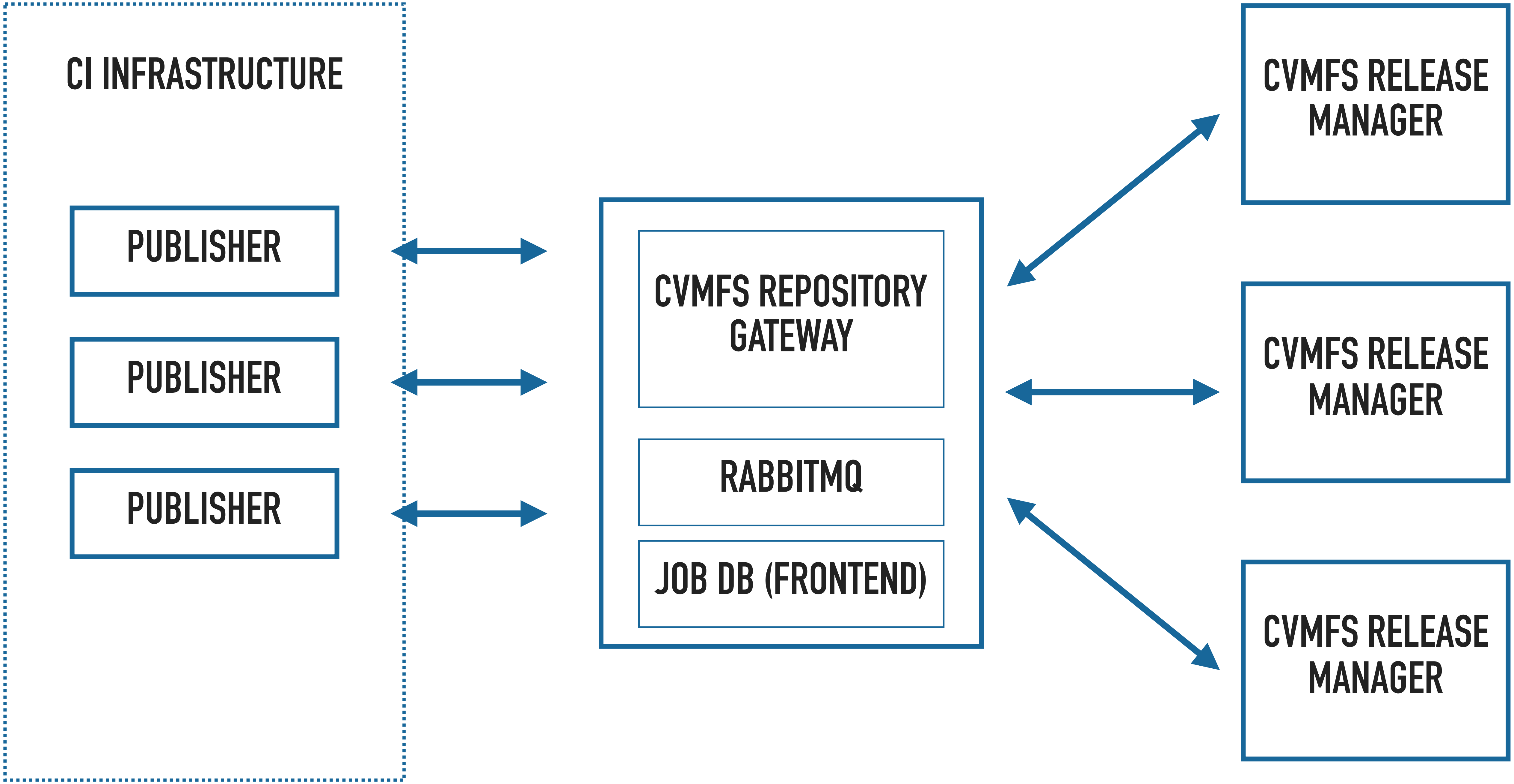
HTTP REQ / REP

PUBLICATION

EXCHANGE

SUBSCRIPTION

DEPLOYMENT EXAMPLE



CI INFRASTRUCTURE

PUBLISHER

PUBLISHER

PUBLISHER

CVMFS REPOSITORY GATEWAY

RABBITMQ

JOB DB (FRONTEND)

CVMFS RELEASE MANAGER

CVMFS RELEASE MANAGER

CVMFS RELEASE MANAGER

CVMFS PUBLISHER TOOLS

WIP @ https://github.com/cvmfs/cvmfs-publisher-tools

- Command-line tool for submitting and consuming jobs, and running the job DB front-end:
  ```
  cvmfs_job submit|consume|db
  ```
- SystemD service files for consumer and job DB
- Provisioning scripts for configuring RabbitMQ, DB
- Single RPM which depends on compatible CVMFS (at least 2.5.2)

DEMO

CLOSING REMARKS

- Very promising!
- Could serve as the base for a unified CernVM-FS repository API
- CernVM-FS Portals could use this system as a backend
- Set up parallel publishing into sft-nightlies.cern.ch and sw.hsf.org