EJP output - content
Content: The authors accepted manuscript PDF and any subsidiary items, e.g. figures, figure suppls, source code, source data, suppl files, videos (note; not tables or boxes, these are contained in the PDF) and XML manifest file
AWS bucket: elife-ejp-poa-delivery
File name structure: eg 1894_1_supp_mat_highwire_zip_54781_n54xj4
Note on file naming: File name is unique to each export. First portion is unique to article, but end (eg n54xj4) is unique to export, so Bot will know if an article is exported twice based on end 'hash'
Delivery details: 10 min past the hour, not every hour (depends on whether content exported)

Bot
Current action: Polls the bucket (CRON pull mechanism) and pulls the PDF and all subsidiary items (ignores XML)
Proposed action: Event-based alerts (push mechanism)

Bot
Current action: Uses the ID from the zip file, which does not correspond to anything else and is not used further
Proposed action: Tahi - uses the ms id in the zip file name

Bot
Current action: Polls the bucket 1/day (CRON pull mechanism)
Proposed action: Event-based alerts (push mechanism)
Additional issues: CSV is not great - incoding issues

EJP output - CSV metadata
Content: SQL queries of database result in a number of CSV files, which are used to build up the PoA metadata XML (9/10 CSV files used to build PoA XML)
AWS bucket: elife-ejp-ftp
Delivery details: Every two hours, 10 min past the hour, key times for eLife staff: 9.10am, 11.10am, 1.10pm, 3.10pm, 5.10pm GMT (note British Summertime, timings change for 2 weeks)
Proposed changes: Rename bucket so vendor agnostic eLife Bot/PoA workflow set up for additional outputs - currently only 11.10am.

Bot - EC2 instance
Current action: Creates transient activity folder once a day (at 10.30am)
Unzips the contents
Decapitates the PDF and rename
Generates XML file
Renames the files
Zips up the DS file
Delivers to: elife-poa-packaging
Proposed action: Tahi - uses the ms id in the zip file name
Not require DS zipped folder post HW?

Note re CSV output
Additional uses:
- Author email alerts - takes all author email addresses from the CSV as they are not all in the XML metadata for a published article
- Editor profiles - this will be the source for these details

Manual production workflow - V2s
Content: XML of V1 is edited as appropriate, and publication date of V1 is added. Also ms ID is updated (V2). Decapitated PDF created manually. Both are delivered to the:
AWS bucket: elife-poa-packaging (directory: outbox)
Issue for automated workflow - DS files regenerated but not always require replacement. If reloaded to HW require manual deletion to remove duplicate

Bot
AWS bucket: elife-poa-packaging (directory: outbox)
Current action: Delivered to this bucket at 10.50am each day. If unmatched pair detected error log is emailed to production and files remains in bucket (PDF or XML)
If decap fails, XML is delivered but email to production and production add decapped PDF to this bucket manually
If a ds.zip file is empty or corrupt not sent to HW
DS zipped file delivered to this bucket (Note: a PoA delivery does not have to have this component)
Remains in bucket for 45 minute window so production can intervene

Bot
AWS bucket: elife-poa-packaging (directory: published)
Current action: All matched pairs available are pulled into a single zip file and delivered with a go.xml file to HWX 11.30am
Corresponding DS zip folders are also delivered but to a different folder on HWX and no go XML manifest required

Note on XML produced
Additional info:
- The eLife published XML contains a string: the GitHub commit of the most recent version of the software
- Non-HW workflow, can add all the available metadata from the EJP database to PoA xml
- Non-HW workflow, can add date to XML

PoA output to HW delivery workflow

PoA

Bot
AWS bucket: elife-poa-packaging
Current action: Once files are delivered to ‘published’ subdirectory the following downstream delivery actions kick in

VoR

Production vendor
AWS bucket: elife-articles + elife-articles-HW + elife-tnq-crossref-delivery
Current action: Production vendor delivers to:
elife-articles-HW - repository of files delivered to HW (they deliver this to HW for live workflow simultaneously)
elife-articles - repository of files delivered to PMC (they deliver to PMC for live workflow simultaneously) + jpg figures that are used for Lens (eg: elife_2012_00003.jpg)
Content from this bucket is used by eLife Bot
elife-tnq-crossref-delivery - repository for XML delivered to CrossRef (they deliver to CrossRef for live workflow simultaneously)
Proposed changes: elife-articles-HW will become redundant and eLife Bot will use deliveries to elife-articles to create all downstream deliveries. Vendor will only make one delivery

Bot - Lens and initial unpacking
AWS bucket: elife-articles
Current action: Unzip all the files for Lens and copy to CDN for Lens display

Bot - PubMed workflow (PoA and VoR)
AWS bucket: elife-poa-packaging (subdirectory pubmed; subdirectory outbox)
Current action: PubMed XML is generated per article and delivered to subdirectory pubmed
Every hour files are polled to see whether they can be published (determined by whether the url is valid for that article and version)
Yes? Batch file is generated for PubMed delivery and pushed to subdirectory outbox
No? File remains in outbox for next hours polling
If a V2 or VoR that has been PoA’d, replaces tag used.
Pub date: today’s date is used for PoA V1; if VoR or PoA V2 etc, xml of file is used as source of published date

Bot - CrossRef workflow (currently PoA only)
AWS bucket: elife-poa-packaging (subdirectory crossref; subdirectory outbox)
Current action: CrossRef XML is generated for PoA per article and delivered to subdirectory crossref
Batch file is generated for PoA CrossRef delivery and pushed to subdirectory outbox
Pub date: today’s date is used for PoA V1; if VoR or PoA V2 etc, xml of file is used as source of published date. If PoA V1 reappears, is redelivered with new pub date (today’s date)
Proposed changes: Batch file is generated for all CrossRef delivery and pushed to subdirectory outbox. VoR CrossRef submission is built from production vendor XML delivery to elife-articles
Change from once a day delivery to when new XML file appears in elite-articles (VoR) or elife-poa-packaging (PoA)

Downstream deliveries workflow

PoA

Bot
AWS bucket: elife-poa-packaging
Current action: Once files are delivered to 'published' subdirectory the following downstream delivery actions kick in

VoR

Production vendor
AWS bucket: elife-articles + elife-articles-HW + elife-tnq-crossref-delivery
Current action: Production vendor delivers to:
elife-articles-HW - repository of files delivered to HW (they deliver this to HW for live workflow simultaneously)
elife-articles - repository of files delivered to PMC (they deliver to PMC for live workflow simultaneously) + jpg figures that are used for Lens (eg: elife_2012_00003.jpg)
Content from this bucket is used by eLife Bot
elife-tnq-crossref-delivery - repository for XML delivered to CrossRef (they deliver to CrossRef for live workflow simultaneously)
Proposed changes: elife-articles-HW will become redundant and eLife Bot will use deliveries to elife-articles to create all downstream deliveries. Vendor will only make one delivery

Bot - Publication Router workflow (VoR only)
AWS bucket: elife-poa-packaging (subdirectory pub_router; subdirectory outbox)
Current action: Once a day, new content arrived in elife-articles it is FTP'd to Publication Router site - all content delivered to the elite-articles bucket is delivered to Publication Router

Bot - Cengage workflow (VoR only)
AWS bucket: elife-poa-packaging (subdirectory cengage; subdirectory outbox)
Current action: Once a day, new content arrived in elife-articles is FTP'd to Cengage site - only article and figures PDF and XML

Bot - Publication author emails workflow (PoA and VoR)
AWS bucket: elife-poa-packaging (subdirectory publication-email; subdirectory outbox)
Current action: Once a day
Checks to prevent sending email twice.
There is a 'black list' of "don't send":
corrections
feature content
List of criteria for email wording and content:
- PoA
- VoR been PoA'd
- VoR not been PoA'd
- VoR with Insight pub at same time
- VoR published previously, new Insight added today

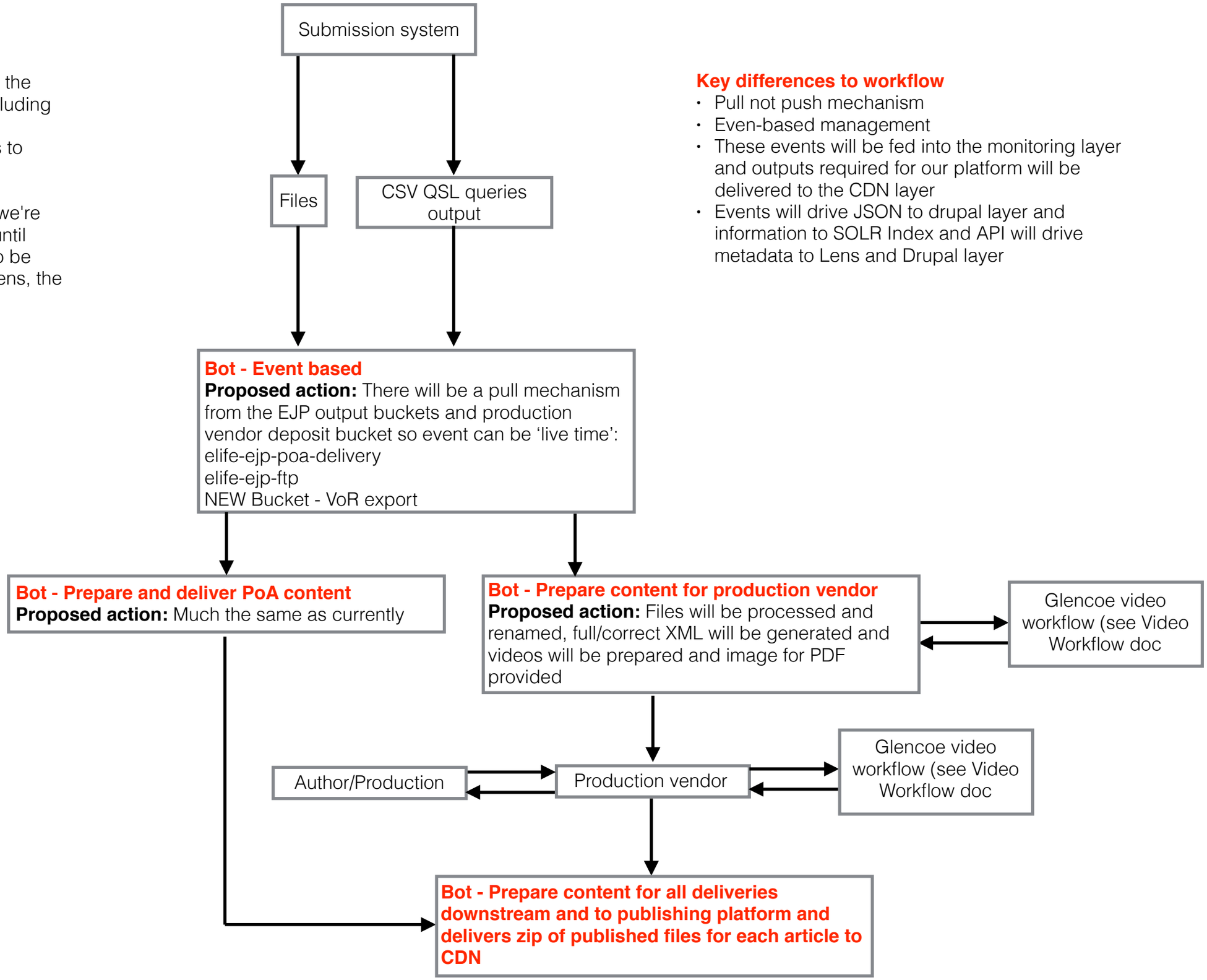
Workflow for new downstream deliveries is extendable
Proposed changes: PMC and CrossRef VoR deliveries via Bot
Future potential additions: Scopus, Paperity, LOCKS, CLOCKS, Chorus, Web of Science, etc

Note on additional changes
- eLife Bot to manage Glencoe workflow - see Video Workflow doc
- eLife API to produce publication date info for eLife Bot to add to PDF and XML (PoA and VoR) - needs more thought, complex workflow
- eLife Bot to implement CrossMark workflow - PDF stamper and XML metadata

Downstream deliveries workflow continued

Notes/changes:

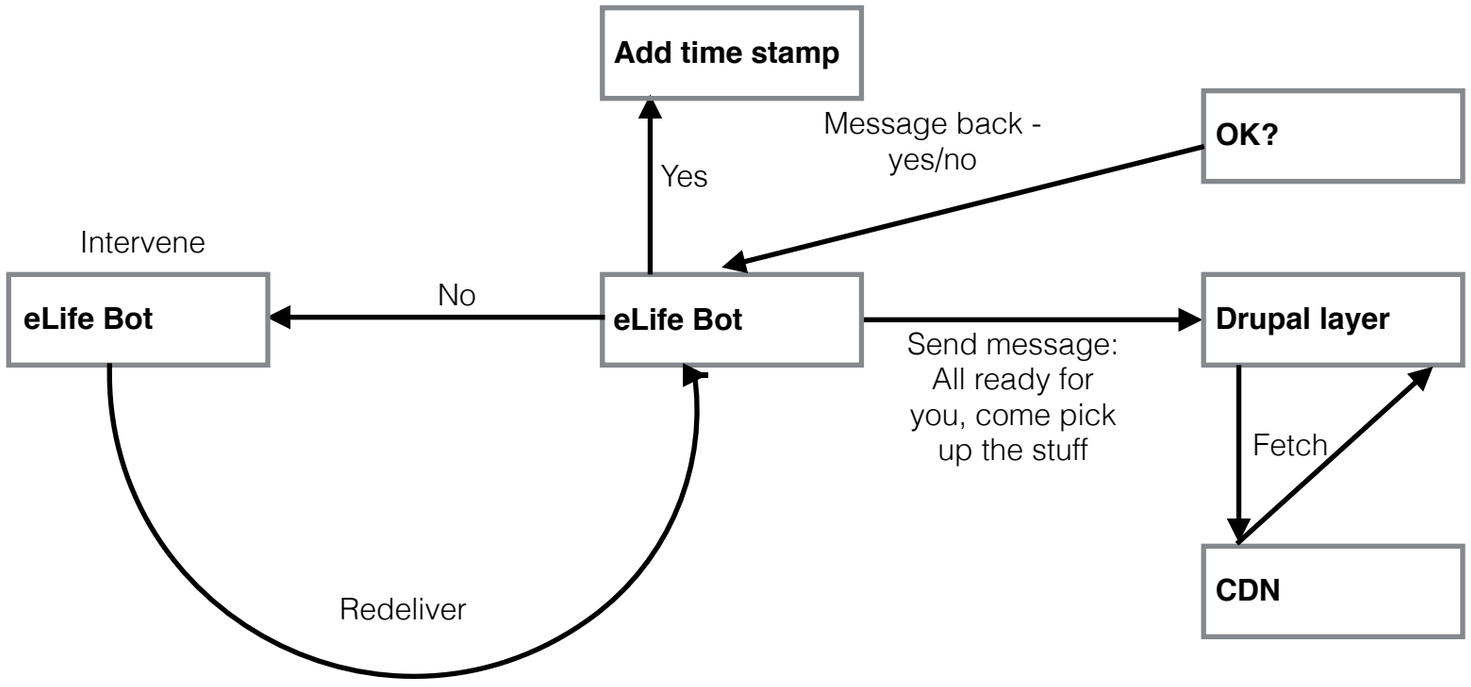
- At each stage a message is delivered to the dashboard so all events are tracked, including actions at production vendor end (also production vendor delivers new versions to AWS as each change is made during production process)
- The "Bot - event based" is ideal, except we're still stuck with a 2 hour timed schedule until there's an EJP system that allows data to be accessed in real-time. If that ever happens, the elife-ejp-ftp bucket will be redundant



Key differences to workflow

- Pull not push mechanism
- Even-based management
- These events will be fed into the monitoring layer and outputs required for our platform will be delivered to the CDN layer
- Events will drive JSON to drupal layer and information to SOLR Index and API will drive metadata to Lens and Drupal layer

Interactions with new publishing system



Undecided workflow - but a potential proposal