swissbib

Developing Infrastructure for Knowledge Work

CBS Partner Meeting 2019

Silvia Witzig – Metadata Specialist, Project swissbib
Günter Hipler – Systems Architect, Project swissbib

Universität Basel
Universitätsbibliothek

Agenda

- Update on CBS related work
- Use case series entries
- Infrastructure for this use case and beyond
- Using this infrastructure for knowledge work
- «Walking for CBS»

CBS @ swissbib in 2019

- Update to CBS 8.2.28
  and csfmr_manager in July
- Minor adaptions to MR clustering
- Improve FRBR clustering
- Work for SLSP

FRBR Clustering

- Goal: Display only one entry on result list of swissbib.ch for a cluster
- Clustering on expression level of FRBR
- Elements to check: title, authors, IDs, series title and number, coordinates, scale, language
- Testset of 1190 records with various materials, manually clustered
- Script to assess the quality of clustering

SLSP

- swissbib takes care of mass corrections and deduplication for SLSP
- Up to now: mass corrections
- Until spring 2020: deduplication
- End of 2020: Go Live

Use Case Series Entries - Reasons

- IDS uses only 490 (with control number in $w)
- RERO uses 490 and 800/830
- SLSP wants to use 490 and 800/810/811/830 to be fully compliant with MARC21
- swissbib has to generate these fields after merging of the records (only one per series record!)

Use Case Series Entries - Rules

- If the series record has a field 100/110/111: generate field 800/810/811
- Otherwise: generate field 830
- To generate the fields 8xx use content from the existing series entry and from the series record
- Keep one 8xx per series record

Use Case Series Entries – Realisation 1

CBS:

- write all fields 490, 800 and 830 with a $w in a temporary field 839
- Remove identical fields without $w during merging
- Export as MARC XML

Use Case Series Entries – Realisation 2

"Developing infrastructure for knowledge work"

If we look at the series entries use case (as one example):

What could be the role of swissbib in the context of knowledge work and what do we need to be able to fulfil this role?

Trends in the metadata world (OCLC)

1) Metadata for inventory management versus metadata for knowledge work, which needs more context. (content for SLSP, linked.swissbib)

2) Trends in the role of libraries: helping users to find the right content as open access leads to potential explosion of content availability versus libraries being actively engaged in knowledge work, supporting content creation and maybe even publishing in an open access world. (various discovery services, automatic content creation/enrichment)

3) Traditional understanding of library collections versus the full scholarly record, i.e. extending the content of interest beyond just research output to e.g. research data and non-peer reviewed publications. (swissbib isn't very traditional, we are more thinking into the direction of the second part)

4) Metadata in the duality of just identifying and disambiguating versus describing in detail (rich metadata). (only automatically done)

Additional trends in the metadata world of swissbib

data workflows have to be simple (decoupled Microservices)

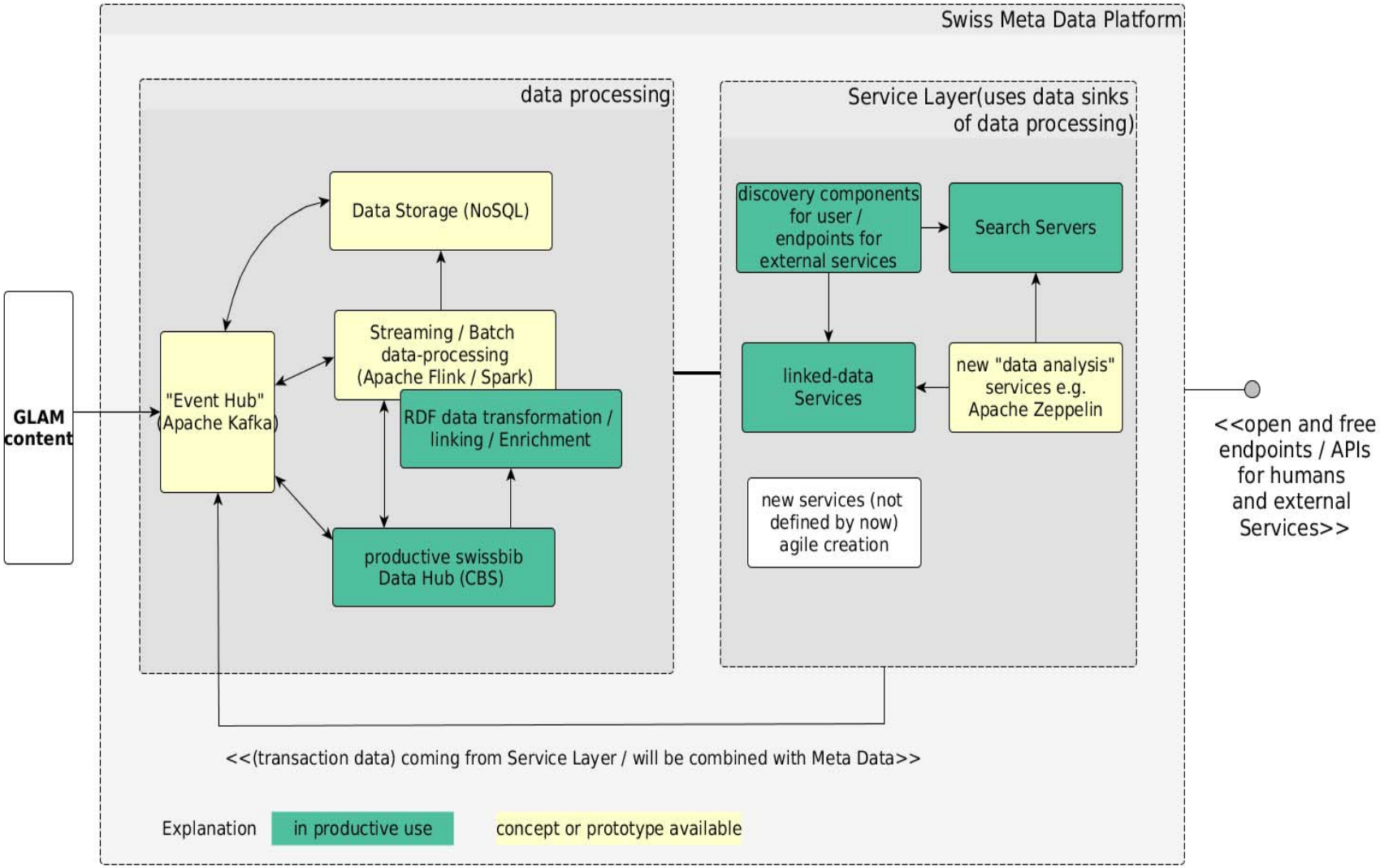
agile and fast development (modern devops, containerization, orchestration)

“DataWork” for “KnowledgeWork”: clustering, enrichment, transformation for very diverse services, all kind of data (structured / unstructured / sparse / rich / archives / educational platforms / audio-video material)

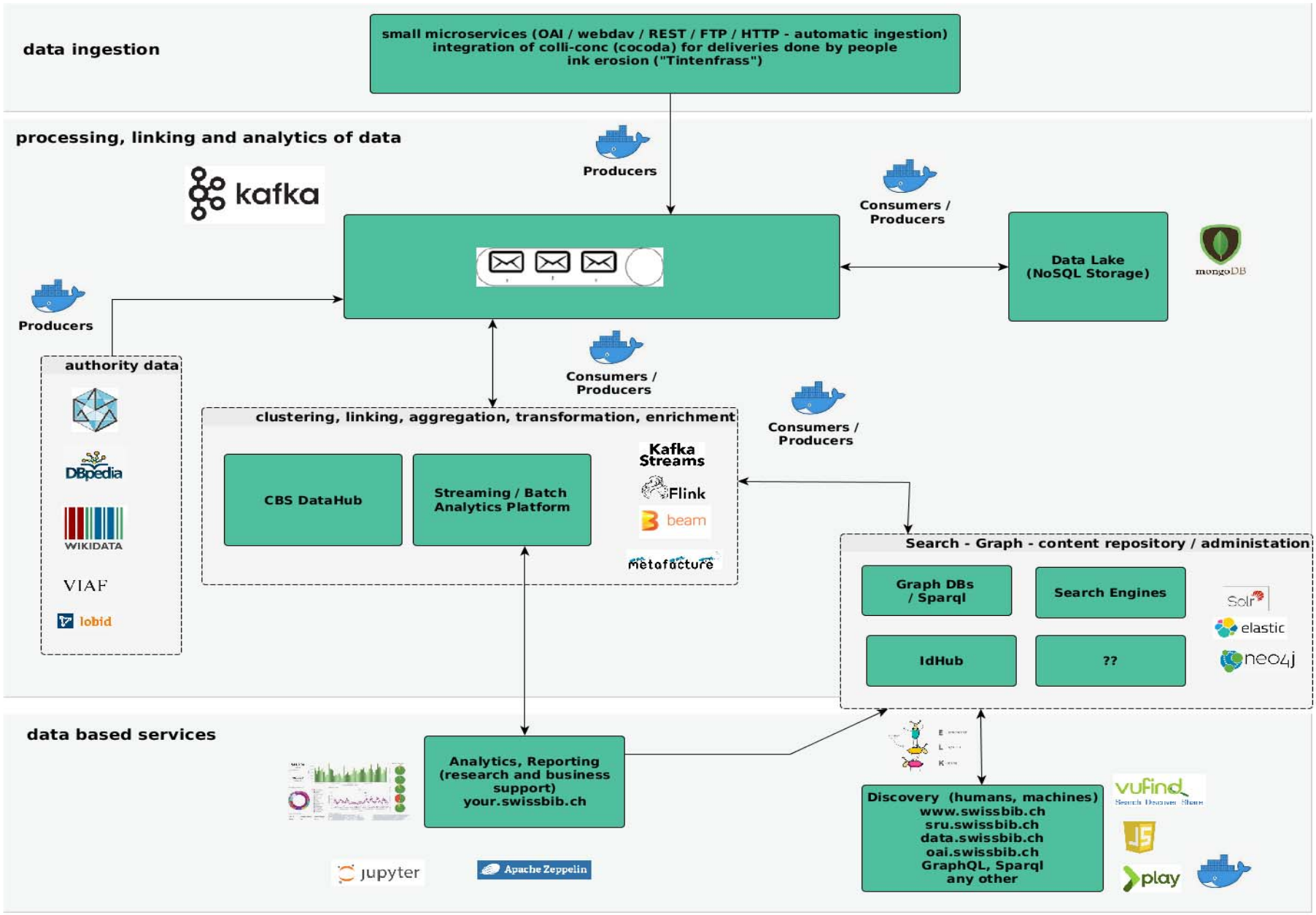
Enable (specialized) users to work with (meta-) data directly (without intermediate discovery software)

Combination of traditional and proven library software (CBS) with new software trends

architectural blueprint: «Swiss Meta Data Platform» (cbs meeting 2017)



Infrastructure for Knowledge Work (2019)



Implementation of the series entries use case with Apache Flink

What is Apache Flink? (a few words)

A distributed stream processor with expressive APIs to implement stateful stream processing applications (unification of batch and streaming is actually the main goal)

originally developed at Universities around Berlin (http://stratosphere.eu)

Further development under Apache by startup “data artisans”, January 2019 bought by Alibaba for 100 Million Euros (now https://www.ververica.com

Data pipes (Batch and Streaming), data analytics, event processing, libraries for Machine Learning, strong integration with Apache Beam

a lot of documentation and presentations are available
(e.g. https://www.flink-forward.org/), very helpful community

Serial entries: Implementation steps for Flink I (using the Flink API to build data pipelines)

```
/*
Step 1: Parse MARC-XML records (17 millions for SLSP)
 */
val env = ExecutionEnvironment.getExecutionEnvironment
env.getConfig.setGlobalJobParameters(pt)
val xmlRecords = env.readTextFile(inputFilePath.get)
  .filter(isRecord _)
  .map(parseRecord _)

/*
Step 2: Use MARC-XML records to build intermediary series and volume records; join records with matching local
system numbers
 */
val volumes = xmlRecords.filter(isVolumeRecord _)
  .flatMap(createStrippedVolumeRecords _)
val series = xmlRecords.flatMap(createSeriesRecords _)
val volumesSeriesJoin = volumes.join(series, JoinHint.REPARTITION_HASH_FIRST)
  .where( firstLeftField = "seriesId")
  .equalTo( firstRightField = "localRecId")
```

https://gitlab.com/swissbib/slsp/series-transformation/volumes-series-enrichment-flink

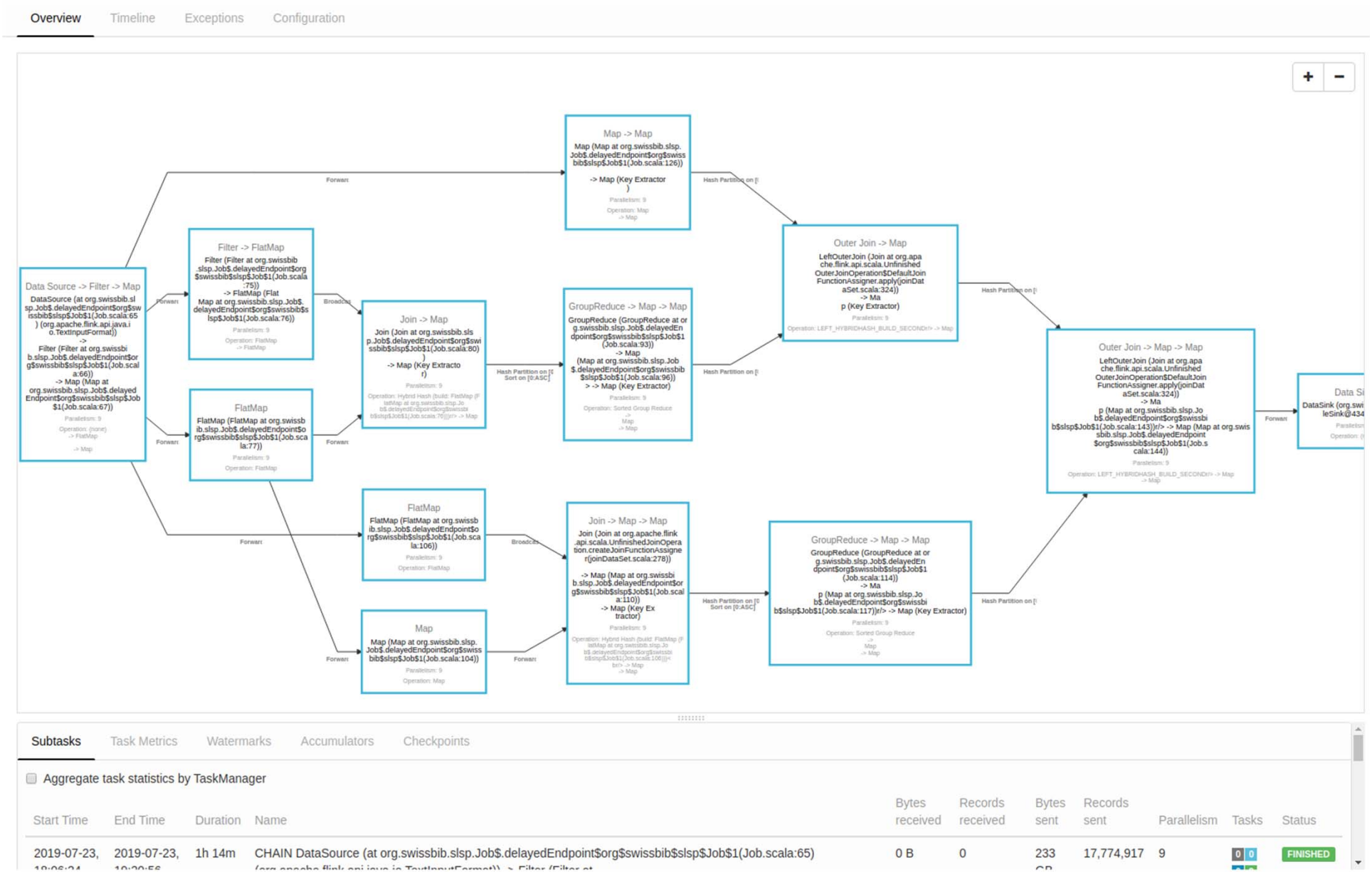
Serial entries: Implementation steps for Flink II (very concise with Scala)

```scala
/*
Step 3: Group by volume's recId, filter out duplicate content (records stemming from the same series) and
discard all unneeded XML content (ie. content not going to field 800, 810, 811 or 830)
*/

val fields8xx = volumesSeriesJoin
  .groupBy(_._1.recId)
  .combineGroup { (recs: Iterator[(VolumeRecord, SeriesRecord)], out: Collector[KeyedField]) =>
    combine8xxFields(recs, out)
  }

/*
Step 4: Clean new volumes stream of redundant 7xx fields
 */

val localKeyRecKeyMap = series.map(s => LocalKeyRecKeyEntry(s.localRecId, s.recId))
val fields7xxWithSeriesRecIds = xmlRecords.flatMap(extract7xxFields _)
  .join(localKeyRecKeyMap, JoinHint.BROADCAST_HASH_SECOND)
  .where( firstLeftField = "seriesId")
  .equalTo( firstRightField = "localRecId")
  .map(j => Field7xx(j._1.fieldTag, j._1.recId, j._2.recId, j._1.fieldContent))
val fields7xx = fields7xxWithSeriesRecIds
  .groupBy(_.recId)
  .combineGroup {
    (fields: Iterator[Field7xx], out: Collector[KeyedField]) => combine7xxFields(fields, out)
  }
```

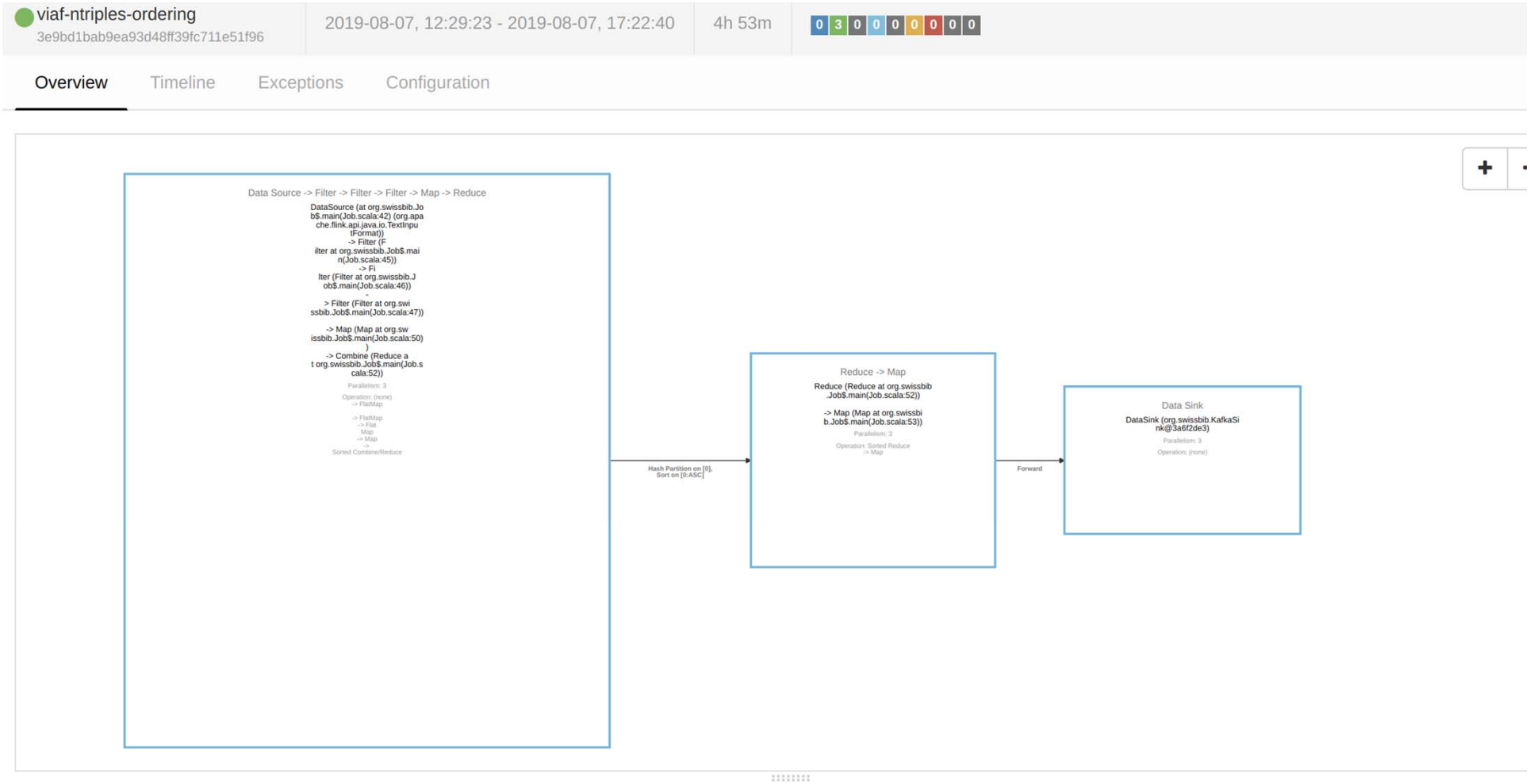
Serial entries: Implementation steps for Flink III (finally: data flow graph is sent to the cluster)

```
  /*
  Step 5: Join, possibly reduce and merge the 7xx and 8xx fields with original volume stream
   */

  val recordsWith8xxFields = xmlRecords
    .map(remove839Fields _)
    .leftOuterJoin(fields8xx, JoinHint.BROADCAST_HASH_SECOND)
    .where(getField001Content _)
    .equalTo(_.recId)
    .apply(applyMerge(_, _)(mergeField8xx))

  val finalRecords = recordsWith8xxFields
    .leftOuterJoin(fields7xx, JoinHint.BROADCAST_HASH_SECOND)
    .where(getField001Content _)
    .equalTo(_.recId)
    .apply(applyMerge(_, _)(mergeField7xx))

  /*
  Step 6: Write records to file(s)
   */
  finalRecords.map(serializeRecord _)
    .map(s => s.replaceAll( regex = "ind2=\"([^\"]+)\" ind1=\"([^\"]+)\" tag=\"([^\"]+)\"",
      replacement = "tag=\"$3\" ind1=\"$2\" ind2=\"$1\""))
    .output(fileSink)


  env.execute( jobName = "Volumes Series Enrichment")
}
```

Serial entries: Implementation steps for Flink IV (running graph on the Flink cluster)

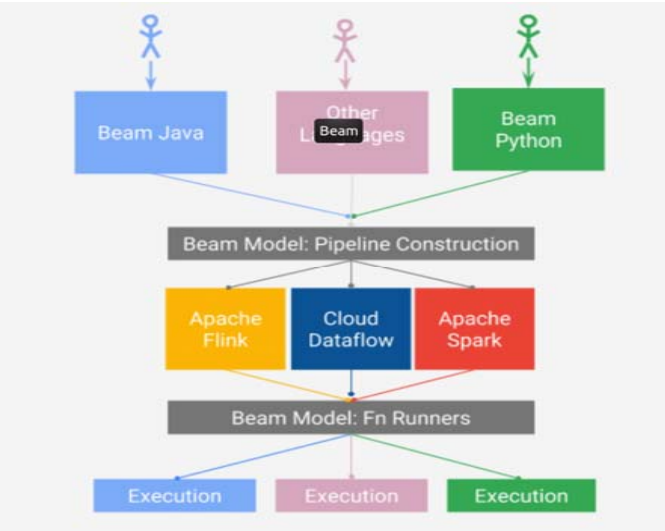


Another Flink data pipe example: load viaf nt triples into Kafka



https://gitlab.com/swissbib/linked/viaf-grouped-ntriples

Using data pipes and data analytics on the user level with Apache Beam (one goal for «knowledge work» for the swissbib project)



https://opensource.com/article/18/5/apache-beam

The vision of Apache Beam in a nutshell:
(by now not runnning – we are working on it)

- language portability
people (researchers, students, librarians) can build data tasks on a high level with various languages (preferably Python) using the beam model

- engine portability
these data tasks can be executed on different runnners (in our case Apache Flink)

possible example use case:
Natural language processing based on PubMed and Mesh

https://www.youtube.com/watch?v=7GwXJJApPtg&list=PL4dEBWmGSIU_jJ82n0WK46agJy4ThegIQ&index=31

Thanks for your time

for questions you can reach us at

Günter Hipler

Systems Architect

swissbib

Universitätsbibliothek Basel

guenter.hipler@unibas.ch

Silvia Witzig

Metadata Specialist

swissbib

Universitätbibliothek Basel

silvia.witzig@unibas.ch

More about swissbib

- GitHub
  https://github.com/swissbib
  https://gitlab.com/swissbib-unibas
  https://github.com/linked-swissbib
  https://gitlab.com/swissbib
- Blog / Twitter
  https://swissbib.blogspot.com
  https://twitter.com/swissbib
- People behind swissbib
  https://swissbib.blogspot.com/2019/01/die-personen-hinter-swissbib-les.html (only german and french)