SAXONICA.COM
XSLT AND XQUERY PROCESSING

XML Projection and Streaming
Compared and Contrasted

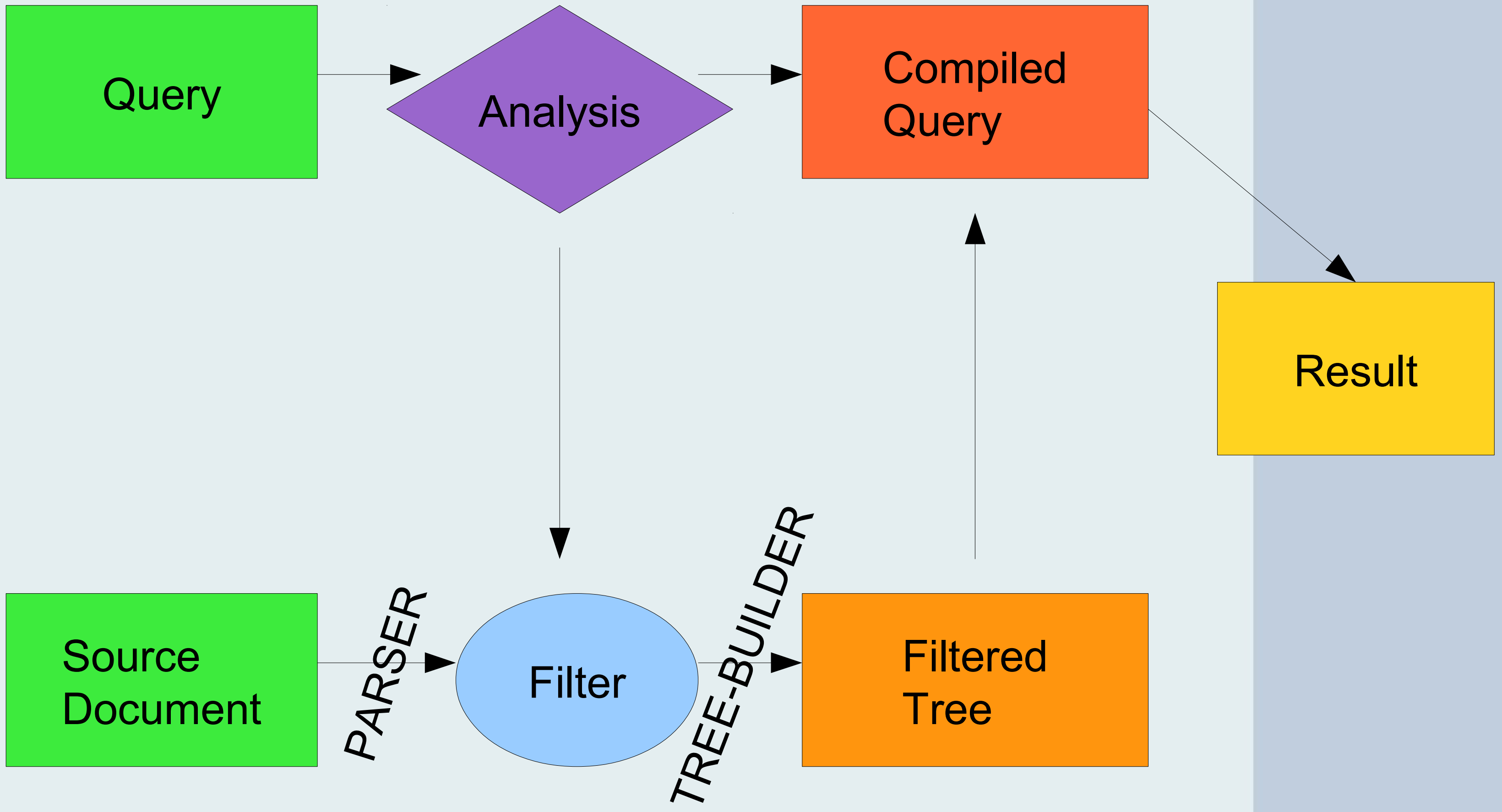
Michael Kay, Saxonica
XML Prague 2017

What is XML Projection?

Amélie Marian & Jérôme Siméon
VLDB 2003

- Take a query as input
- Work out which parts of the document are needed
- Create a filter
- Build the document in memory, filtering out everything the query doesn't need
- Run the query against the filtered document

Document Projection Flow



Example

```
<books>
 <book>
  <title>XSLT 2.0</title>
  <author>Michael Kay</author>
  <date>2008</date>
  <isbn>978-0470192740</isbn>
  <price curr="USD">41.68</price>
 </book>
 <book>
  <title>XQuery 3.0</title>
  <author>Priscilla Walmsley</author>
  <date>2015</date>
  <isbn>978-1491915103</isbn>
  <price curr="USD">47.99</price>
 </book>
</books>
```

```
for $b in //book[date gt 2000]
order by $b/price
return $b/title
```

```
<books>
 <book>
  <title>XSLT 2.0</title>
  <date>2008</date>
  <price>41.68</price>
 </book>
 <book>
  <title>XQuery 3.0</title>
  <date>2015</date>
  <price>47.99</price>
 </book>
</books>
```

Document Projection in Saxon

- Available since Saxon 9.0
- XQuery only, needs Saxon-EE
- Not widely used
- Somewhat rusty – not updated for new XQuery 3.0 features
- Implementation is completely separate from XSLT streaming

Benefits of Document Projection

- No change to query
- No hard restrictions on the query
  - benefits joins, sorting, grouping
- No extra costs when there are no benefits to be gained
- Potential for large memory savings
  - typically a linear reduction, say 95%

Limitations of Document Projection

- Only applies to "single-shot" queries
  - source document is built to run one query
- Diminishing returns as the query complexity increases
- Requires static reachability analysis:
  - no dynamic function calls, "eval", template rules etc
  - Hence XQuery-only

XMark Q1

```
for    $b in /site/people/person[@id="person0"]
return $b/name
```

- Query analysis: 430ms
- Tree building (110Mb): 1400ms
- Query execution: ~~0.12ms~~ *50ms!*

XMark Results for Document Projection

- 20 queries, all fairly simple
- 19 of the queries achieve 95% memory reduction
  – Consistent between Saxon and Marian/Siméon results
- No measurable cost for query analysis
- 25% improvement in tree building time
- 5-10% faster query execution

Streaming

XSLT 3.0 (CR 7 Feb 2017)
saxon:stream() in XQuery

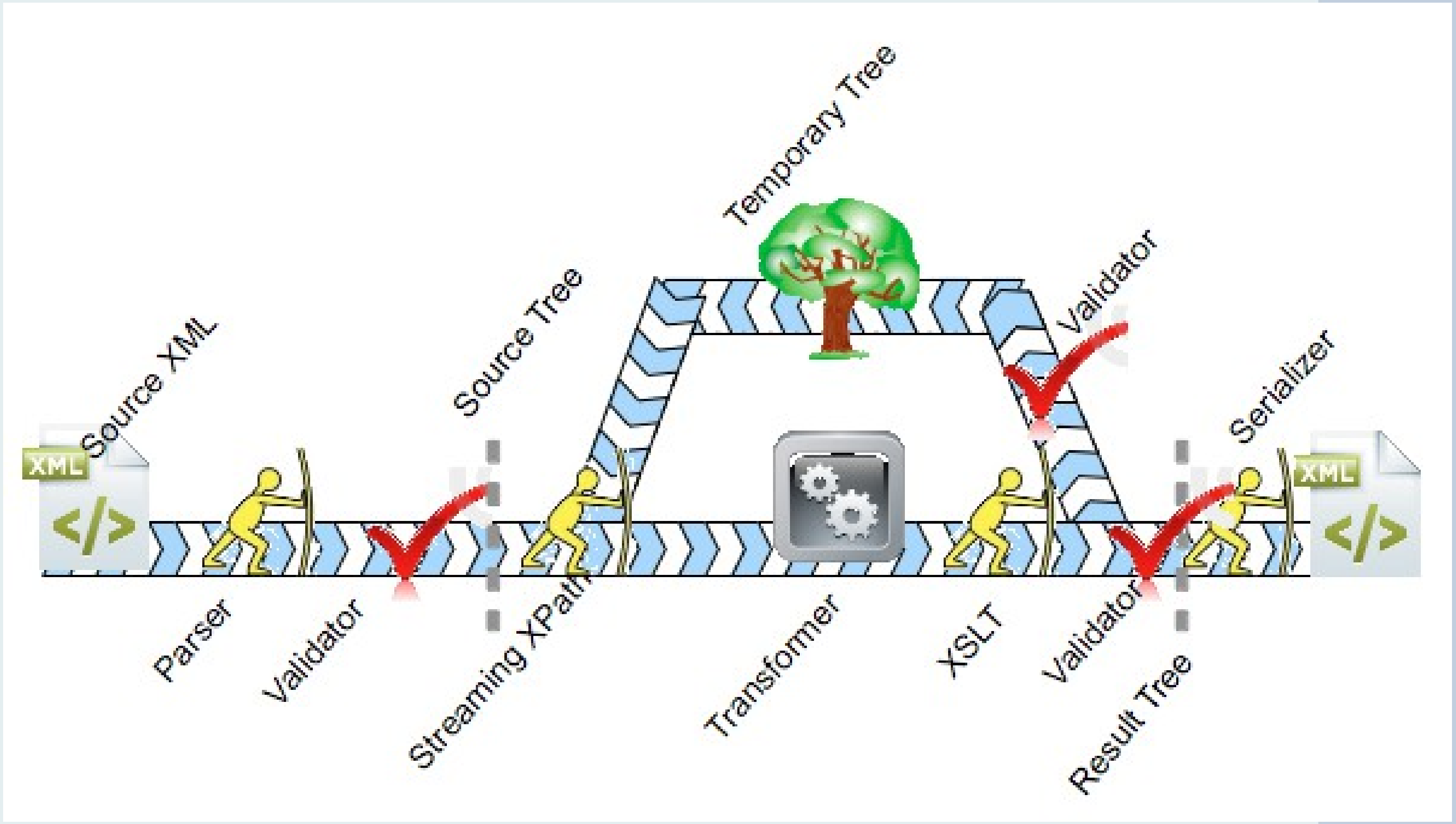
How does Streaming work?

- Code is analyzed for streamability
- If streamable:
  - no source tree is built
  - the XSLT/XQuery is executed directly on parsing events.
  - this is only possible if the data is processed in order of arrival
- If not streamable:
  - choice of failure / fallback processing

Streaming in Saxon

- Has been developing over a series of releases
- XSLT and also XQuery (via extensions)
- Push-mode implementation

(from XML Prague 2014)



SAXONICA.COM
XSLT AND XQUERY PROCESSING

Streaming Results for XMark

- Queries need to be rewritten to satisfy streamability rules
- After rewrite, 14 out of 20 are streamable
  - five use joins
  - one uses sorting
- Query execution time (for 110Mb data file): typically 900ms

Comparison

	PROJECTION	STREAMING
Depends on static "reachability" analysis	✔	✔
Document is read once per query execution	✔	✔
Constant memory for infinite documents	✘	✔
Allow non-linear queries (joins, sorting)	✔	✘
Early exit	✘	✔

Opportunities

- Reuse static analysis algorithms
- Automate "snapshot copy" streaming
- Projection in xsl:source-document
- Binding streamed nodes to variables
- Declarative annotation of functions