

# Choosing between XSLT 2.0 and XQuery 1.0

Michael H. Kay

### The Simple Answer

- XQuery is good at query
  - Designed for XML database query
- XSLT is good at transformation
  - Designed for document rendering



### What makes it complicated?

- Application architecture choices
- Different kinds of data
- Query and transformation are fuzzy categories
- Skills issues
- Product maturity and risk issues



# Technical Components of an XML-centric Application

- Database Technology
  - Long term information storage
  - Query access
- Application Development Tools
  - For coding the business logic
- User Interface Technology
  - Displaying information
  - Entering information
- Middleware
  - For binding the application components together



### **Application Development**

- First decision point:
  - -3GL *vs* 4GL
  - Procedural vs Declarative
  - Java/C# vs XSLT/XQuery
- Benefits of higher-level languages:
  - Productivity (faster to develop)
  - Flexibility (faster to change)
  - Reliability (fewer opportunities for bugs)



### Middleware: binding the components

- XML processing is well-suited to a pipeline architecture
- Each step in the pipeline is a transformation from XML to XML
- Optional validation between stages
- Benefits:
  - Reduced complexity
  - Component reuse
  - Flexibility of deployment
  - Ability to mix technologies



### Query vs Transformation

- Query: "find me..."
  - How many?
  - Select ... Where
  - Documents with ...
- Transformation: "change..."
  - XML into HTML
  - zzML 1.0 into zzML 1.1
  - Sort, group, up-convert
  - Copy everything except ....



## Joris Graaumans Study: Usability of XML Query Languages

- Measured performance and satisfaction on a number of tasks performed with both languages
- Clear win for XQuery
- But note:
  - The tasks were all queries
  - The programs were all very small
  - The users were novices



### What happens when you scale up?

- Transformations in a pipeline often change small parts of the document.
  - XSLT tackles this well using template rules
- One application has to handle a variety of inputs and outputs.
  - XSLT has run-time polymorphism
  - XSLT has build-time flexibility through xsl:import



### Scaling up (continued...)

- Big applications need to be selfmanaging
  - XSLT is XML (XQuery isn't)
- Optimization
  - XQuery relies heavily on the system to optimize queries
  - XSLT gives more control to the developer

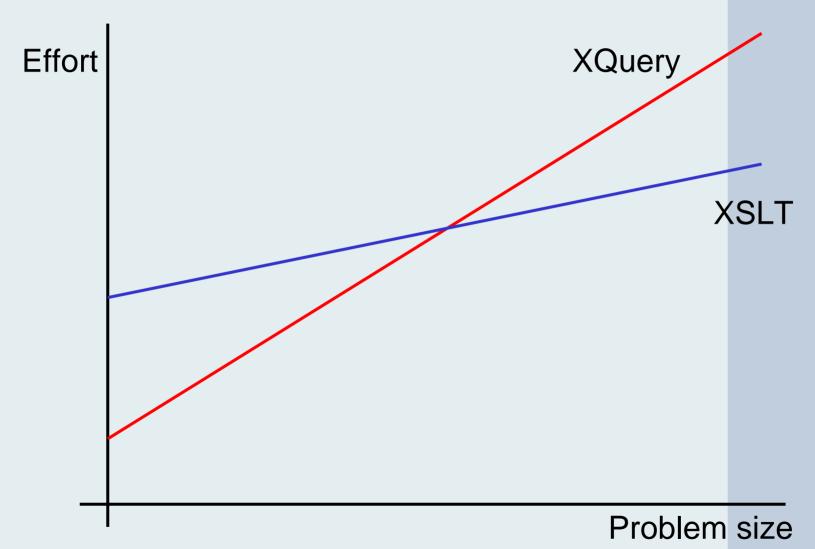


## Extra functionality in XSLT (XSLT 2.0 vs XQuery 1.0)

- Template rules
- Import precedence
- Grouping capabilities
- Formatting numbers and dates
- Namespace manipulation
- More powerful regex facilities
- Tunnel parameters
- Validation by schema type



### **Usability Comparison**



SAXONICA SAXUERY PROCESSING

#### Mix and match

- Pipeline architecture lets you mix components in different languages
- Data model is the same
- 80% of the concepts are shared
- So any professional XML developer should have both tools in the kitbag



#### **Conclusions**

- XQuery is great for query, XSLT is better for transformation
- XQuery is easier for small jobs,
  XSLT for large jobs
- They mix-and-match well in a pipeline architecture
- Developers should have both in their kitbag

