Q & A

Benchmarking XML Databases: New TPoX Benchmark Results Available

Abstract

TPoX stands for "Transaction Processing over XML" and is an XML database benchmark that Intel and IBM developed several years ago and then released as open source.

A couple of months ago, the project published some new results. (http://nativexmldatabase.com/2011/03/04/new-tpox-benchmark-results-available/)

To learn more about this I interviewed the main leaders of the TPoX project, Dr. Matthias Nicola, Senior engineer for DB2 at IBM and Dr. Agustin Gonzalez, Senior Staff Software Engineer at Intel® Corporation.

Q1. What is exactly TPoX?

Matthias: TPoX is an XML database benchmark that focuses on XML transaction processing. TPoX simulates a simple financial application that issues XQuery or SQL/XML transactions to stress XML storage, XML indexing, XML Schema support, XML updates, logging, concurrency and other components of an XML database system. The TPoX package comes with an XML data generator, an extensible Workload Driver, three XML Schemas that define the XML structures, and a set of predefined transactions. TPoX is free, open source, and available at http://tpox.sourceforge.net/.

Although TPoX comes with a predefined workload, it's very easy to change this workload to adjust the benchmark to whatever your goals might be. The TPoX workload driver is very flexible, it can even run plain old SQL against a relational database and simulate hundreds concurrent database users. So, when you ask "What is TPoX", the complete answer is that it is an XML database benchmark, but also a very flexible and extensible framework for database performance testing in general.

Q2. When did you start with this project? What was the original motivation for TPoX? What is the motivation now?

Matthias: We started with this project in the 2003/2004 timeframe. We were working on the native XML support in DB2 that was released in DB2 version 9.1 in 2006. We needed an XML workload -a benchmark- that was representative of an important class of real-world XML applications and that would stress all critical parts of a database system. We needed a tool to put a heavy load on the new XML database functionality that we were developing. The research community including XMark, MBench, XMach-1, XBench, X007 had proposed some XML benchmarks. They were useful in their respective scope, such as evaluating XQuery processors, but we felt that none of them truly evaluated a database system in its *entirety;* did not represent all relevant characteristics of real-world XML applications. For example, many only defined a read-only and single-user workload on a single XML document. However, real applications typically have many concurrent users, a mix of read and write operations, and millions or even billions of XML documents. That's what we wanted to capture in the TPoX benchmark.

Agustin: And the motivation today is the same as when TPoX became freely available as open source: database and hardware vendors, database researchers, and even database practitioners in the IT departments of large corporations need a tool evaluate system performance, compare products, or compare different design and configuration options.

At Intel, the main motivation behind TPoX is to benchmark and improve our platforms for the increasingly relevant intersection of XML and databases. So far, the joint results with IBM have exceeded our expectations.

Q3. TPoX is an application-level benchmark. What does it mean? Why did you choose to develop an application-level benchmark?

Matthias: We typically distinguish between micro-benchmarks and application-level benchmarks, both of which are very useful but have different goals. A micro-benchmark typically defines a range of tests such that each test exercises a narrow and well-defined piece of functionality. For example, if your focus is an XQuery processor you can define tests to evaluate XPath with parent steps, other tests to evaluate XPath with descendant-or-self axis, other tests to evaluate XQuery "let" clauses, and so on. This is very useful for micro-optimization of important features and functions. In contrast, an application-level benchmark tries to evaluate the end-to-end performance of a realistic application scenario and to exercise the performance of a complete system as a whole, instead of just parts of it.

Agustin: As an application-level benchmark, TPoX has proven much more useful and believable than "synthetic" micro-benchmarks. As a result, TPoX can even be used to predict how similar real-world applications will perform, or where they will encounter a bottleneck. You cannot make such predications with a micro-benchmark. Another important feature is that TPoX is very scalable – you can run TPoX on a laptop but also scale it up and run on large enterprise-grade servers, such as multi-processor Intel Xeon platforms.

Q4. How do you exactly evaluate the performance of XML databases?

Agustin: Well, one way is to use TPoX on a given platform and then compare to existing results on different combinations of hardware and software. I know that this is a simplistic answer but we really learn a lot from this approach. Keeping a precise history of the test configurations and the results obtained is always critical.

Matthias: This is actually a very broad question! We use a wide range of approaches. We use micro-benchmarks, we use application-level benchmarks such as TPoX, we use real-world workloads that we get from some of our DB2* customers, and we continuously develop new performance tests. When we use TPoX, we often choose a certain database and hardware configuration that we want to test and then we gradually "turn up the heat". For example, we perform repeated TPoX benchmark runs and increase the number of concurrent users until we hit a bottleneck, either in the hardware or the software. Then we analyze the bottleneck, try to fix it, and repeat the process. The goal is to always push the available hardware and software to the limit, in order to continuously improve both.

Q5. What is the difference of TPoX with respect to classical database benchmarks such as TPC-C and TPC-H?

Matthias: One of the obvious differences is that TPC-C and TPC-H focus on very traditional and mature relational database scenarios. In contrast, TPoX aims at the comparatively young field of XML in databases. Another difference is that the TPC benchmarks have been standardized and "approved" by the TPC committee, while TPoX was developed by Intel and IBM, and extended by various students and Universities as an open source project.

Agustin: But, TPoX also has some important commonalities with the TPC benchmarks. TPC-C, TPC-H, and TPoX are all application-level benchmarks. Also, TPC-C, TPC-H, and TPoX have each chosen to focus on a specific *type* of database workload. This is important because no benchmark can (or should try to) exercise all possible types of workloads. TPC-C is a relational transaction processing benchmark, TPC-H is a relational decision support benchmark, and TPoX is an XML transaction processing benchmark. Some people have called TPoX the "XML-equivalent of TPC-C". Another similarity between TPC-C, TPC-E, and TPoX is that all three are throughput oriented "steady state benchmarks", which makes it straightforward to communicate results and perform comparisons.

Q6. Do you evaluate both XML-enabled and Native XML databases? Which XML Databases did you evaluate?

Matthias: TPoX can be used to evaluate pretty much any database that offers XML support. The TPoX workload driver is architected such that only a thin layer (a single Java* class) deals with the specific interaction to the database system under test. I personally have used TPoX only on DB2. I know that other companies as well as students at various Universities have also run TPoX against other well-known database systems.

Q7. How did you define the TPoX Application Scenario? How did you ensure that the TPoX Application Scenario you defined is representative of a broader class of applications?

Matthias: Over the years we have been working with a broad range of companies that have XML applications and require XML database support. Many of them are in the financial sector. We have worked closely with them to understand their XML processing needs. We have examined their XML documents, their XML Schemas, their XML operations, their data volumes, their transaction rates, and so on. All of that experience has flown into the design of TPoX. One very basic but very critical observation is that there are practically no real-world XML applications that use only a single large XML document. Instead, the majority of XML applications use very large numbers of small documents.

Agustin: TPoX is also very realistic because it uses a real-world XML Schema called FIXML, which standardizes trade-related messages in the financial industry. It is a very complex schema that defines thousands of optional elements and attributes and allows for immense document variability. It is extremely hard to map the FIXML schema to a traditional normalized relational schema. In the past, many XML processing systems were not able to handle the FIXML schema. But, since type of XML is used in real-world applications, it is a great fit for a benchmark.

Q8. How did you define the workload?

Matthias: Again, by experience with real XML transaction processing applications.

Q9. In your documentation you write that TPoX uses a "stateless" workload? What does it mean in practice? Why did you make this choice?

Matthias: It means that every transaction is submitted to the database independently from any previous transactions. As a result, the TPoX workload driver doesn't need to remember anything about previous transactions. This makes it easier to design and implement a benchmark that scales to billions of XML documents and hundreds of millions transactions in a single benchmark run.

Q10. Why not define a workload also for complex analytical queries?

Matthias: We did! And we ran it on a 10TB XML data warehouse with more than 5.5 billion XML documents. That was a very exciting project and you can find more details on my blog: http://nativexmldatabase.com/2010/07/14/a-10tb-xml-data-warehouse-benchmark/.

Although the initial wave of XML database adoption was more focused on transactional and operations systems, companies soon realized that they were accumulating very large volumes of XML documents that contained a goldmine of information. Hence, the need for XML warehousing and complex analytical XML queries was pressing. We wanted to show that DB2's shared-nothing architecture scales horizontally for XML warehousing just as it does for traditional relational warehousing workloads.

Agustin: Admittedly, we have not yet formally included this workload of complex XML queries into the TPoX benchmark. Just like TPC-C and TPC-H are separate for transaction processing vs. decision support, we would also need to define two flavors of TPoX, even if the underlying XML data remains the same. A TPoX workload with complex queries is definitely very meaningful and desirable.

Q11. What are the main new results you obtained so far? What are the main values of the results obtained so far?

Agustin: We have produced many results using TPoX over the years, with ever larger numbers of transactions per second and continuous scalability of the benchmark on increasingly larger platforms. A key value is to provide strong data points that demonstrate and quantify how XML database processing can be done with very high performance. In particular, the first public 1TB XML benchmark that we did a few years ago has helped establish the notion that efficient XML transaction processing is a reality today. Such results give the hardware and the software a lot of credibility in the industry. And of course we learn a lot with every benchmark, which allows us to continuously improve our products.

Q12. You write in your Blog (http://nativexmldatabase.com/2011/05/09/news-flash-intel-publishes-tpox-benchmark-results-on-new-10-core-westmere-ex-cpus/) "For 5 years now Intel has a strong history of testing and showcasing many of their latest processors with the

Transaction Processing over XML (TPoX) benchmark. Why has Intel been using the TPoX benchmark? What results did they obtain?

Matthias: I'll let Agustin answer this one.

Agustin: Intel uses the TPoX benchmark because it helps us demonstrate the power of Intel platforms and generate insights on how to improve them. TPoX also enables us to work with IBM to improve the performance of DB2 on Intel platforms, which is good for both IBM and Intel. This collaboration of Intel and IBM around TPoX is an example of an extensive effort at Intel to make sure that enterprise software has excellent performance on Intel. You can see our most important results on the TPoX web page: http://tpox.sourceforge.net/tpoxresults.htm.

Q13. Can you use TPoX to evaluate other kinds of databases (e.g. Relational, NoSQL, Object Oriented, Cloud stores)? How does TPoX compare with the Yahoo! YCSB benchmark for Cloud Serving Systems?

Matthias: Yes, the TPoX workload driver can be used to run traditional SQL workloads against relational databases. Assuming you have a populated relational database, you can define a SQL workload and use the TPoX driver to parameterize, execute, and measure it. TPoX and YCSB have been designed for different systems under test. However, parts of the TPoX framework can be reused to quickly develop other types of benchmarks, especially since TPoX offers various extension points.

Agustin: Some open source relational databases have started to offer at least partial support for the SQL/XML functions and the XML data type. Given the level of parameterization and the extensible nature of the TPoX workload driver it would be very easy to develop custom workloads for the emerging support of the XML data type on open source databases. At the same time, the powerful XML document generator included in the kit can be used to generate the required data. Using TPoX to test the performance of XML in open source databases is an intriguing possibility.

Q14. Is it possible to extend TPoX? If yes, how?

Matthias: Yes, TPoX can be extended in several ways. First, you can change the TPoX workload in any way you want. You can modify, add, or remove transactions from the workload, you can change their relative weight, and you can change the random value distributions that are used for the transaction parameters. We have used the TPoX workload driver to run many different XML workloads, also on other XML data than just the TPoX documents. We have also used the workload driver for relational SQL performance tests, just because it's so easy to setup concurrent workloads.

Second, the database specific interface of the TPoX workload driver is encapsulated in a single Java class, so it is relatively easy to port the driver to another database system. And

third, the new version TPoX 2.1 allows transactions to be coded not only in SQL, SQL/XML, and XQuery, but also in Java. TPoX 2.1 supports "Java-Plugin transactions" that allow you to implement whatever activities you want to run and measure in a concurrent manner. For example, you can run transactions that call a web service, send or receive data from a message queue, access a content management system, or perform any other operations – only limited by what you can code in Java!

Agustin: At Intel we have been using TPoX internally for various other projects. Since the TPoX workload driver is open source, it is straightforward to modify it to support other type of workloads, not necessarily steady state, which makes it amenable to testing other aspects of computer systems such as power management, storage, and so on.

Q15. What are the current limitations of TPoX?

Matthias: Out of the box, the TPoX workload driver only works with databases that offer a JDBC interface. If a particular database system has specific requirements for its API or query syntax, then some modifications may be necessary. Some database system might require their own JDBC driver to be compiled into the workload driver.

Q16. Who else is using TPoX?

Matthias: You can see some examples of other TPoX usage on the TPoX web site at http://tpox.sourceforge.net/tpoxresults.htm#Other_TPoX_Usage. We know that other database vendors are using TPoX internally, even if haven't decided to publish results yet. I also know a company in the data security space that uses TPoX to evaluate the performance of different data encryption algorithms. And TPoX also continues to be used at various universities in Europe, US, and Asia for a variety of research and student projects. For example, the University of Kaiserslautern in Germany has used TPoX to evaluate the benefit of solid-state disks for XML databases. Other universities have used TPoX to evaluate and compare the performance of several XML-only databases.

Q17. TPoX is an open source project. How can the community contribute?

Matthias: A good starting point is to *use* TPoX. From there, contributing to the TPoX project is easy. For example, you can report problems and bugs on http://sourceforge.net/projects/tpox/, or you can submit new feature requests. Or even better, you can implement bug fixes and enhancements yourself and submit them to the SVN code repository on sourceforge.net. If you design other workloads for the TPoX data set, you can upload new workloads to the TPoX project site and have your results posted on the TPoX web site.

Agustin: As is customary for an open source project on Sourceforge, anybody can download all TPoX files and source code freely. If you want to upload any changed or new files or

modify the TPoX web page, you only need to become a member of the TPoX Sourceforge project, which is quick and easy at http://sourceforge.net/project/memberlist.php?group_id=185925. Everybody is welcome, without exceptions or exclusions.

Notices

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS OTHERWISE AGREED IN WRITING BY INTEL, THE INTEL PRODUCTS ARE NOT DESIGNED NOR INTENDED FOR ANY APPLICATION IN WHICH THE FAILURE OF THE INTEL PRODUCT COULD CREATE A SITUATION WHERE PERSONAL INJURY OR DEATH MAY OCCUR.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or go to: http://www.intel.com/design/literature.htm