# The Case for **JSON** Data Exchange

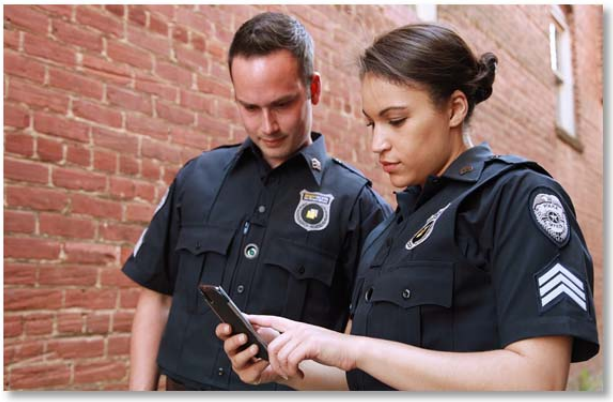
With studies showing performance up to *thousands* of times faster with JSON, major enterprises like Twitter, Google and Facebook already favor the "fat-free" alternative to XML (and for use *with* standards based on XML, like NIEM). Here's why you should too.

Law enforcement personnel need good information to make smart decisions. Whether it's an officer making a seemingly routine traffic stop, or a dispatcher potentially sending a colleague into harm's way, they all must be able to find and communicate information about the situation quickly, reliably and accurately.

Most current information sharing protocols in the industry have been shaped by the FBI's National Crime Information Center's text-based data, a format originally inspired by *ticker tape*. To be fair, that approach has served the industry well for many years. Unfortunately, it is failing to keep abreast of modern developments, particularly the proliferation of smart, diverse devices all connected to the Internet – first laptops in cruisers, and now smart phones, tablets and other handheld devices.

This growing "Internet of Things" has real-world implications. For example, a single dispatcher would historically serve tens of officers simultaneously, taking inquiries from each, and running them simultaneously on a workstation. Now, officers have devices on their person and in their cruiser that can directly access that information, and dispatchers are increasingly focused on the higher-priority tasks that only a human can perform, e.g. coordinating emergency response situations.

The most popular data format in law enforcement is NIEM (National Information Exchange Model), which traces its lineage back through GJXDM (Global Justice XML Data Model) back to the XML (eXtensible Markup Language), one of the oldest and most common standardized data formats.[i] Its "markup" provides a way to annotate documents and data in a highly structured way. It is also extremely verbose and, thus, inefficient. Today, we have a better way to communicate information.



## JSON (JavaScript Object Notation)

JSON – the data format of choice of many major web application APIs today, including Twitter[ii], Google Maps[iii], and Facebook[iv] – represents more than 60% of new APIs. (XML, by contrast, has dwindled as a data format, used for less than 20% of new APIs.[v])

That's because JSON preserves all of the strengths of XML (and implementations based on XML, like NIEM) while increasing efficiencies – often dramatically.

When the law enforcement industry originally adapted XML into GJXDM and then NIEM, they were unconcerned with characteristics like size and efficiency: the cost of resources like memory and bandwidth were plummeting, while new use-cases like mobile apps and cloud-based services had not yet begun their rise. Today, however, we must be able to exchange information between all manner of devices from a rapidly growing set of sources.

In the end, law enforcement relies on efficient service that enhances communications. That's what JSON delivers, and why JSON should be the data format of choice for organizations aiming to deliver better service.

# JSON: The "Fat-Free" Alternative To XML

It's not the purpose of this paper to delve into the technical intricacies of XML or JSON. Suffice it to say that JSON directly maps data to the underlying data structures used by software languages and is a light-weight or "fat-free" alternative to XML-based formats.[vi] That is, JSON can be easily read by most programming languages without the need for extra manual coding required for processing XML data.

JSON retains all of the same relevant advantages of NIEM (and other XML-based formats) over legacy text as a data format while simultaneously offering much more performant parsing and validation.

XML and JSON differ in how they communicate pieces of data. For example, CHIEF-compliant criminal history data is also based on the NIEM standard (which is itself built on XML). If you wanted to transmit a subject's name between two applications, the XML format requires multiple start and end tags to "wrap" the data:

```
<nc:Person
xmlns:nc="http://release.niem.gov/niem/niem-core/3.0/">

 <nc:PersonName>
   <nc:PersonGivenName>LESTER</nc:PersonGivenName>
    <nc:PersonSurName>TESTER</nc:PersonSurName>
 </nc:PersonName>

</nc:Person>
```

GCN, a technology news and education site for Public Sector IT, notes:

> "The duplication of [labels] is a source of concern for many programmers who hate to waste any bandwidth or processing power."[vii]

And although the NIEM standard is based on XML, it does *not* require the use of XML; it can be completely implemented using JSON instead of XML for data


exchange, without changing application logic. At its core, NIEM is a naming and structuring convention, and JSON simply serves as a more efficient method of packaging data while still using the NIEM standard.

In response to these developments, the NIEM Technical Architecture Committee (NTAC) is anticipated to release an executive summary presenting a roadmap for supporting a NEIM-conformant JSON instance in 2016.

In NIEM-compliant JSON, the same data could be formatted much more simply and concisely:

```
"Person": {
 "PersonName": {
  "PersonGivenName": "LESTER",
  "PersonSurName": "TESTER"   } }
```

JSON cuts the amount of code that must be transmitted by over half. Using XML for data exchange, by contrast, consumes more bandwidth, memory, and disk for audit, archival, and DR. XML slows down processing and eats valuable resources that JSON does not – a concern that scales rapidly with the growing number of connected devices in Law Enforcement's burgeoning "Internet of Things."

According to *Programmable Web*:

> "Take a situation in which we need to retrieve user information from a database. An [SOAP/XML] call is relatively long – over 10 lines. The return is just as dense, including superfluous data clouding the requested information in the middle of the response. Taking the same scenario, JSON … could call and retrieve the same information in 4 lines."[viii]

| Figure 1. Resource Consumption XML vs. JSON | | |
|---|---|---|
| | **XML (KB)** | **JSON (KB)** |
| **CJIS Data** | 161 | 161 |
| **+ Overhead** | 1,523 | 411 |
| **= Total Size** | 1,684 | 572 |

We see the same principles at play in our own work with criminal justice data. In fact, in many cases the difference is even more substantial, with NIEM-compliant JSON allowing us to use just a *third* of the resources (see Figure 1).

Other studies have also demonstrated that JSON blows XML away in performance, as we'll detail next.

# Performance Tests: Measuring The Difference

## Oracle A-Team

Oracle created an app that retrieves a data set from a server. Theirs was nested, structured data of departments and employees, but it could have easily been an officer requesting a background on a suspect. In their test, XML's payload was nearly 300% the size of JSON: 77.3 KB versus 26.2 KB; A finding that substantiates our own experience.

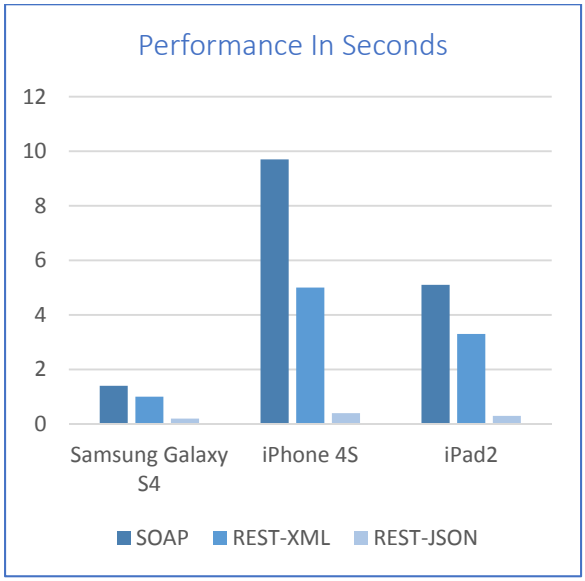



*Figure 1. Source: Oracle*

As a result, when Oracle tested the performance of several mobile applications, they found JSON (used with the REST web service protocol) worked up to 30 times faster than SOAP-XML. [ix] REST used with JSON even worked in a fraction of the time as REST and XML. (We explore REST and SOAP more in-depth in our companion white paper).

That's evidence not just of XML's unnecessary weight but also of its impedance to rapid information exchange in the face of modern technology.

Oracle noted: "JSON is favored due to its more compact representation, its easier to read and is the native data format in JavaScript. Interestingly, newer APIs which only support JSON are on the rise and 45% of APIs now support JSON with many new APIs offering JSON as the only data format."[x]

## University of Montana - Bozeman Study

Researchers at the University of Montana also investigated the time required to retrieve data sets. They created two scenarios: the first was the transmission of 1 million objects, which took 75.77 minutes for XML … and just 78.26 *seconds* for JSON.[xi] That's nearly 6,000% faster.

| Figure 2. UM-Bozeman Study: JSON vs. XML Timing, Scenario 1 | | |
|---|---|---|
| | **JSON** | **XML** |
| **No. of Objects** | 1 million | 1 million |
| **Total Time (sec)** | 78.26 | 4546.70 |
| **Avg. Time (ms)** | 0.08 | 4.55 |

Their conclusion: "The average values of measurements from scenario 1 indicate that sending data in JSON encoding is in general faster than using XML encoding. The average time and total time

measures provide an indication that JSON's speed outperforms XML's speed."

Scenario 2 looked at five different samples (of 20k, 40k, 60k, 80k, and 100k objects). They found the average transmission time per object in Scenario 2 was magnitudes longer for XML (3.1 milliseconds) versus JSON (0.08ms). Resource consumption was closer than in Scenario 1, but the average system CPU utilization was still triple for XML than JSON.

Figure 3. UM-Bozeman Study:
JSON vs. XML Timing, Scenario 2

| Utilization: | JSON | XML |
|---|---|---|
| **No. of Objects** | 100,000 | 100,000 |
| **Total Time (sec)** | 7.5 | 310 |
| **Avg. Time (ms)** | 0.08 | 3.1 |

## JSON's Advantages Go Further Than Just Performance

To be fair, XML shines in certain applications. Considering it grew out of the HTML standard, it's especially effective as a document markup language. In fact, XML-based formats are the default for many leading software tools, including Microsoft Office development toolkits, like the .NET Framework.

But the story changes when we look at its use for application to application information sharing, especially in law enforcement scenarios.

And it's not just because JSON is faster and lighter for users. JSON is also easier and cheaper for IT staff and programmers to develop and maintain.

The lack of verbosity in JSON eases human consumption over XML: that makes it easier and faster to write in the first place, and then debug and troubleshoot later. That ease-of-use can translate into faster development times, less time spent on maintenance, and lower costs overall.

In fact, one of JSON's most compelling attributes is a syntax that enables direct mapping to programming language objects, as opposed to XML and XML-based formats that require manually coding to translate between its own data structures and the application's


or database's. For developers, that means JSON is often far easier to use. In addition to reduced development time and expense, it leaves a clean code base that can be easily maintained over time, leading to lower long-term maintenance costs.

Solutions that better address bandwidth consumption and resource (memory, CPU, disk) usage are simply superior in today's fast-evolving environment and proliferating devices.

Or as Oracle, in evaluating its performance findings, concludes: "The only real solution is to embrace industry best practices and move to REST-JSON services."

We agree.

CPI The OpenFox Company

*Proven Progress. Proven Protection.*

**Computer Projects of Illinois, Inc. (CPI)**, with its headquarters in Bolingbrook, Illinois, is a privately held corporation and is the acknowledged leader in information-sharing software systems for the law enforcement and criminal justice community.

CPI's sole focus has been, and will continue to be, this sector. CPI expends all of our energies on the development, installation and maintenance of our software products. CPI systems are state-of-the-art and cost-effective; ensuring that our customers get the most for their investment.

Computer Projects of Illinois, Inc.
475 Quadrangle Drive, Suite A
Bolingbrook, IL 60440

Tel: (630) 754-8820
Fax: (630) 754-8835

www.openfox.com

The "OpenFox" Company

# References

[i] **Wikipedia**. "XML." Retrieved November 2015 from https://en.wikipedia.org/wiki/XML#History.
[ii] **Twitter**. "REST APIs." Retrieved November 2015 from https://dev.twitter.com/rest/public.
[iii] **Google**. "The Google Maps Geocoding API." Retrieved November 2015 from https://developers.google.com/maps/documentation/geocoding/intro#JSON.
[iv] **Facebook**. "JSON with Unity." Retrieved November 2015 from https://developers.facebook.com/docs/unity/reference/current/Json.
[v] **DuVander**, A. (2013, Dec 26). "JSON's eight year convergence with XML." *Programmable Web.* Retrieved November 2015 from http://www.programmableweb.com/news/jsons-eight-year-convergence-xml/2013/12/26.
[vi] **JSON.org**. "JSON: The Fat-Free Alternative to XML." Retrieved November 2015 from http://www.json.org/xml.html.
[vii] **Daconta**, M.C. (2014, Apr 16). "JSON vs. XML and the impact of design decisions." *GCN.* Retrieved November 2015 from https://gcn.com/blogs/reality-check/2014/04/json-xml-design-tools.aspx.
[viii] **Doerrfeld**, B. (2015, Jan 2). "Why JSON triumphed over SOAP." *Programmable Web.* Retrieved November 2015 from http://www.programmableweb.com/news/why-json-triumphed-over-soap/elsewhere-web/2015/01/02.
[ix] **Davelaar**, S. (2015, Feb 27). Performance study – REST vs. SOAP for mobile applications. *Oracle.* Retrieved August 2015 from http://www.ateam-oracle.com/performance-study-rest-vs-soap-for-mobile-applications/.
[x] **Mason**, R. (2011, Oct 20). "How REST replaced SOAP on the Web: What it means to you." *InfoQ.* Retrieved November 2015 from http://www.infoq.com/articles/rest-soap.
[xi] **Nurseitov**, N., **Paulson**, M., **Reynolds**, R., & **Izurieta**, C. (2009). "Comparison of JSON and XML Data Interchange Formats: A Case Study." *Montana State University – Bozeman. Retrieved November 2015 from* http://www.cs.montana.edu/izurieta/pubs/caine2009.pdf.