# Analisa Unjuk Kerja Format Serialisasi Data JSON dan XML

Muhammad Ghazali[1]

Teknik Informatika, Fakultas Teknik, Universitas Widyatama
Cikutra No. 204 A Bandung, Jawa Barat, Indonesia 40124
muhammad.ghazali@widyatama.ac.id

**Ringkasan** JSON digunakan untuk merepresentasikan data berbasis teks dalam format yang dapat dikonsumsi dengan mudah oleh aplikasi lain. Penulis memilih format serialisasi data JSON karena JSON lebih mudah ditulis dan dibaca oleh mesin (komputer) dan manusia. Selain itu JSON lebih mudah untuk diproses karena memiliki struktur yang lebih sederhana dibandingkan XML[7][4].
Penulis membuktikan bahwa data yang diserialisasikan dalam format JSON memiliki ukuran yang lebih kecil dibandingkan XML. Disini penulis hanya membandingkan ukuran data yang dihasilkan setelah dilakukan serialisasi. Hal tersebut dibuktikan dengan membangun purwa-rupa Web API untuk menerapkan JSON pada *resource* yang akan dikonsumsi oleh aplikasi lain. Dengan membangun purwa-rupa Web API ini dibuktikan juga bahwa API yang menserialisasikan data dalam format JSON mampu melayani permintaan *web resource* lebih banyak dibandingkan dengan API yang menserialisasikan data dalam format XML.
Kata kunci: Web API, JSON, Format Serialisasi Data

## 1 Pendahuluan

Dalam penelitian ini penulis membahas mengenai topik menarik tentang analisa unjuk kerja format serialisasi data JSON[1] dan XML[2]. Hal yang dipelajari adalah untuk membuktikan bahwa data yang diserialisasi dalam format JSON memiliki ukuran data yang lebih kecil jika dibandingkan dengan data yang diserialisasi dalam format XML. Untuk membuktikan hal tersebut penulis mengambil studi kasus implementasi Web API untuk CampusLife *mobile information directory application.*

[1] Lihat bagian Landasan teori: JSON
[2] Lihat bagian Landasan teori: XML

Web API[3] dibangun dengan tujuan untuk membuka akses secara tidak langsung ke *data store*[4] yang tersimpan di salah satu layanan *Database as a Service*[5] yang digunakan oleh LLM di AppFog[6]. Seluruh data-data *event* yang tersimpan di *data store* akan diolah oleh Web API menjadi data dengan format yang dapat dikonsumsi dengan mudah oleh aplikasi *mobile* CampusLife. Proses pengolahan tersebut dinamakan serialisasi data[7].

Namun, hal yang menjadi fokus di penelitian ini bukanlah tentang implementasi Web API tersebut, melainkan tentang analisa unjuk kerja dari penerapan format serialisasi data ketika data dikirimkan dari Web API ke *client*. Web API hanya akan dibangun sebagai pendukung untuk membantu penulis dalam menganalisa ukuran data yang diformat JSON dan XML ketika dikirimkan ke *client*. Mulai dari bab ini dengan selanjutnya data-data yang yang dikirimkan dari Web API ke *client* akan sering disebut *resource*.

Adapun masalah yang diteliti dalam penelitian ini dapat dirumuskan sebagai berikut:

1. Seberapa kecil ukuran *resource* yang diformat dalam JSON jika dibandingkan dengan ukuran *resource* yang diformat dalam XML?
2. Bagaimana implikasi terhadap jumlah permintaan *resource* yang berhasil dikembalikan oleh Web API jika *resource* diformat dalam JSON dan XML?

## 2 Metodologi

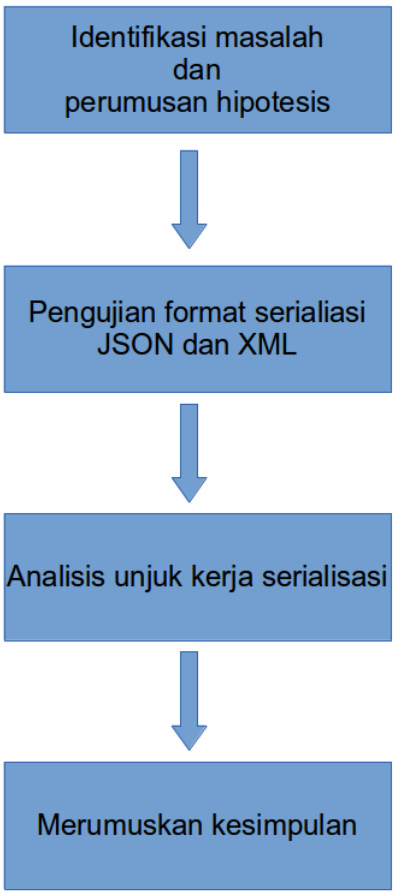
Adapun tahapan penelitian yang dilakukan oleh penulis adalah sebagai berikut:

1. **Identifikasi masalah dan perumusan hipotesis**. Pada tahap ini penulis merumuskan masalah yang diteliti.
2. **Pengujian format serialiasi JSON dan XML**. Pada tahap ini penulis melakukan pengujian serialisasi dengan membangun purwa-rupa Web API untuk mendukung proses pengujian serialiasi data dalam format JSON dan XML.
3. **Analisis unjuk kerja serialisasi**. Pada tahap ini penulis menganalisis unjuk kerja dari format serialisasi JSON dan XML dengan melakukan beberapa skenario pengujian.

[3] Lihat bagian Landasan teori: Web API
[4] http://en.wikipedia.org/wiki/Data_store
[5] http://en.wikipedia.org/wiki/Cloud_database
[6] http://www.appfog.com/
[7] Lihat bagian Landasan teori: Serialiasi Data

4. **Merumuskan kesimpulan**. Pada tahap ini penulis membuat kesimpulan hasil analisis dari unjuk kerja format JSON dan XML.

Visualisasi dari tahapan yang dijelaskan di bagian sebelumnya dapat dilihat pada gambar 1.




**Gambar 1.** Visualisasi Metodologi

# 3 Analisis

## 3.1 Hasil Pembandingan Antara JSON dan XML

Penulis melakukan pembandingan antara JSON dan XML dalam hal ukuran data yang dihasilkan dan jumlah respon permintaan yang berhasil dikembalikan ketika Web API diakses oleh *client*. Pembandingan akan dilakukan dengan menggunakan data yang sama dan jumlah detik yang digunakan saat melakukan pembandingan.

Untuk melakukan hal ini penulis menggunakan Apache Benchmark[8]. Kakas tersebut akan bertugas mengirimkan permintaan Web API *end point* daftar *event*. Berikut adalah skenario pembandingan yang penulis lakukan:

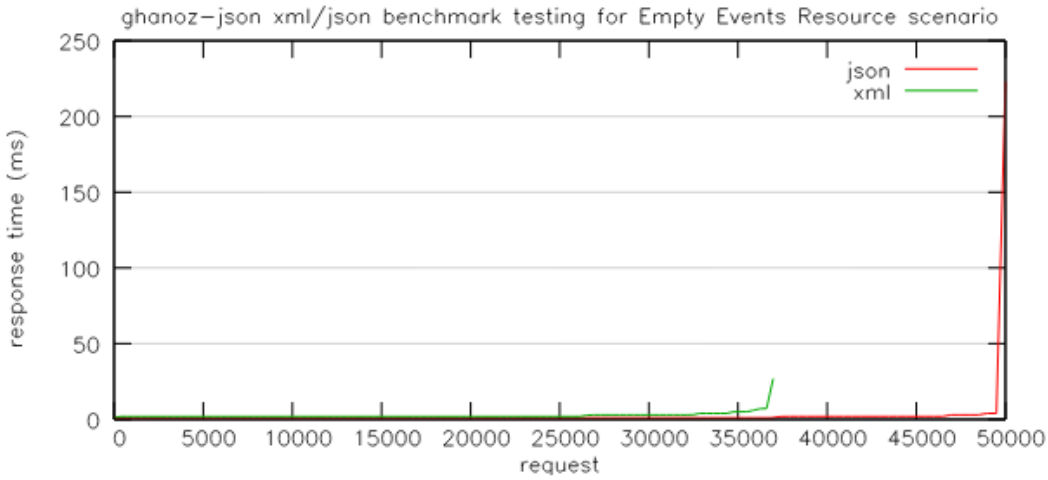
1. *Empty Resource*. Melakukan pembandingan jumlah permintaan yang berhasil direspon dengan respon yang dikembalikan berisi *resource* dengan setiap *field* dari *resource* diberi *resource* kosong dengan setiap *field* dari *resource* diberi *string* kosong
2. *Normal Resource*. Melakukan pembandingan jumlah permintaan yang berhasil direspon dengan respon yang dikembalikan berisi *resource* dengan setiap *field* dari *resource* diberi data yang "normal"
3. *Just Once*. Melakukan pembandingan ukuran data terhadap hasil respon yang dikembalikan berisi *resource* dengan setiap *field* dari *resource* diberi data yang "normal"

Hasil detail pembandingan dari masing-masing skenario tersebut dapat dilihat di bagian selanjutnya.

**Hasil Pembandingan Skenario Empty Resource** *Empty Resource*. Melakukan pembandingan jumlah permintaan yang berhasil direspon dengan respon yang dikembalikan berisi "*resource* kosong" dengan setiap *field* dari *resource* diberi *string* kosong. Representasi dalam bentuk grafik[9] dari hasil pembandingan tersebut dapat dilihat pada gambar 2.

Bisa dilihat melalui grafik di atas, bahwa dalam 73,853 detik, Web API mampu mengembalikan respon dari 50000 permintaan dalam format JSON. Sedangkan dalam 90 detik, Web API hanya mampu mengembalikan respon dari 36973 permintaan dalam format XML.

[8] http://httpd.apache.org/docs/2.4/programs/ab.html
[9] Grafik dibuat dengan menggunakan kakas gnuplot http://www.gnuplot.info/




**Gambar 2.** Perbandingan Jumlah Permintaan yang berhasil direspon untuk skenario Empty Resource

### 3.2 Hasil Pembandingan Skenario Normal Resource

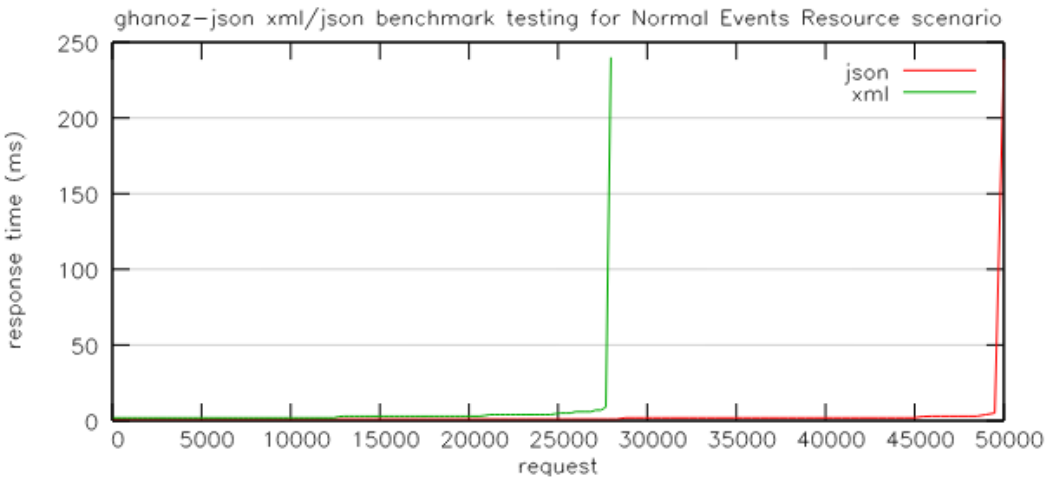
*Normal Resource*. Melakukan pembandingan jumlah permintaan yang berhasil direspon dengan respon yang dikembalikan berisi *resource* dengan setiap *field* dari *resource* diberi data yang "normal". Representasi dalam bentuk grafik dari hasil pembandingan tersebut dapat dilihat pada gambar 3.

Bisa dilihat melalui grafik di atas, bahwa dalam 87.414 detik, Web API mampu mengembalikan respon dari 50000 permintaan dalam format JSON. Sedangkan dalam 90 detik, Web API hanya mampu mengembalikan respon dari 27966 permintaan dalam format XML.

### 3.3 Hasil Pembandingan Skenario Just Once

*Just Once*. Melakukan pembandingan ukuran data terhadap hasil respon yang dikembalikan berisi *resource* dengan setiap *field* dari *resource* diberi data yang "normal". Berikut detail hasil pembandingannya:

1. Ukuran data *resource* dalam format JSON dengan besar 4377 *bytes* berhasil direspon oleh Web API dalam waktu 0,0004 detik
2. Ukuran data *resource* dalam format XML dengan besar 6635 *bytes* berhasil direspon oleh Web API dalam waktu 0,0005 detik




**Gambar 3.** Perbandingan Jumlah Permintaan yang berhasil direspon untuk skenario Normal Resource

# 4 Penutup

## 4.1 Kesimpulan

Selama menjalani masa penelitian penulis mendapatkan beberapa kesimpulan dari hasil penerapan JSON sebagai format serialisasi data:

1. Berdasarkan hasil pengujian yang sudah dilakukan, terbukti bahwa data yang diserialisasi dalam format JSON memiliki ukuran data yang lebih kecil jika dibandingkan dengan data yang diserialisasi dalam format XML.
2. Implikasi dari penerapan JSON pada *resource* adalah jumlah permintaan yang berhasil dikembalikan jauh lebih banyak dibandingkan ketika Web API mengembalikan *resource* dalam format XML.

## 4.2 Saran

Untuk mengembangkan hasil penelitian ini, penulis menyarankan beberapa topik pengembangan berikut:

1. Membandingkan JSON secara eksplisit dengan beberapa format serialisasi yang lain dalam hal keunggulan dan kelemahan
2. Melakukan pengujian Web API dengan mengaksesnya melalui aplikasi
3. Meneliti isu-isu keamanan ketika menerapkan JSON

4. Menganalisa trend penggunaan format serilisasi data tertentu *mobile*

Demikian saran-saran yang dapat penulis sampaikan, penulis berharap laporan ini dapat bermanfaat bagi semua pihak yang membaca.

## Pustaka


[1] Rasmusson, Jonathan (2010) The Agile Samurai: The Pragmatic Bookshelf.

[2] Deepak, G., and Dr. Pradeep B S. *Challenging Issues and Limitations of Mobile Computing*. International Journal of Computer Technology and Applications 3.1 (2012): Academic Journals Database. Web. 8 Jan. 2013.

[3] Audie Sumaray dan S. Kami Makki. *A comparison of data serialization formats for optimal efficiency on a mobile platform*. 6th International Conference on Ubiquitous Information Management and Communication (2012): Artikel No. 48. ACM Digital Library. Web. 24 Jan 2013

[4] Marinescu, Floyd dan Tilkov, Stefan. "Debate: JSON vs. XML as a data interchange format." *Infoq*. Web. 20 Januari 2013. http://www.infoq.com/news/2006/12/json-vs-xml-debate

[5] Marinescu, Floyd dan Tilkov, Stefan. "How REST replaced SOAP on the Web: What it means to you." *Infoq*. Web. 11 Januari 2013. http://www.infoq.com/articles/rest-soap

[6] Rodriguez, Alex. "RESTful Web services: The basics." *IBM*. Web. 11 Januari 2013. http://www.ibm.com/developerworks/webservices/library/ws-restful/

[7] "JSON: The Fat-Free Alternative to XML." *JSON Official Website*. Web. 20 Januari 2013. http://www.json.org/xml.html

[8] "Introducing JSON" *JSON Official Website*. Web. 20 Januari 2013. http://www.json.org/

[9] Ramirez, Ariel Ortiz. "Three-Tier Architecture." *Linux Journal*. Web. 18 Maret 2013. http://www.linuxjournal.com/article/3508

[10] Reuven, Lerner. "APIs." *Linux Journal*. Web. 18 Maret 2013. http://www.linuxjournal.com/content/apis

[11] Irani, Romin. "JSON Continues its Winning Streak Over XML." *ProgrammableWeb Blog*. Web. 8 April 2013. http://blog.programmableweb.com/2010/12/03/json-continues-its-winning-streak-over-xml/

[12] Perkins, Luc. "Why JSON will continue to push XML out of the picture." *ProgrammableWeb Blog*. Web. 8 April 2013. http://blog.appfog.com/why-json-will-continue-to-push-xml-out-of-the-picture/

[13] Crockford, Douglas. "The application/json Media Type for JavaScript Object Notation (JSON)." *RFC Index*. Web. 5 April 2013. http://tools.ietf.org/html/rfc4627

[14] "Manifesto Pengembangan Perangkat Lunak Agile" *Agile Manifesto*. Web. 8 April 2013. http://agilemanifesto.org/iso/id/

[15] "Introducing BDD" *Dan North & Associates*. Web. 7 Mei 2013. http://dannorth.net/introducing-bdd/

[16] "What's in a Story?" *Dan North & Associates*. Web. 7 Mei 2013. http://dannorth.net/whats-in-a-story/

[17] "Just Barely Good Enough Models and Documents: An Agile Best Practice." *Agile Modelling*. Web. 10 Mei 2013. http://www.agilemodeling.com/essays/barelyGoodEnough.html

[18] "Multitier architecture." *Wikipedia*. Web. 15 Maret 2013. http://en.wikipedia.org/wiki/Multitier_architecture

[19] "Application Programming Interface." *Wikipedia*. Web. 22 Maret 2013. http://en.wikipedia.org/wiki/Application_programming_interface

[20] "Web API." *Wikipedia*. Web. 20 Januari 2013. http://en.wikipedia.org/wiki/Web_API

[21] "XML." *Wikipedia*. Web. 7 April 2013. http://en.wikipedia.org/wiki/XML

[22] "Acceptance Testing." *Wikipedia*. Web. 7 Mei 2013. http://en.wikipedia.org/wiki/Acceptance_testing

[23] "Serialization." *Wikipedia*. Web. 24 Januari 2012. http://en.wikipedia.org/wiki/Serialization

[24] "Agile software development." *Wikipedia*. Web. 24 Januari 2013. http://en.wikipedia.org/wiki/Agile_software_development