

Thursday, July 7, 2005

XMLRPC: Sebuah Perkenalan

Pengembangan perangkat lunak adalah sebuah proses yang bisa jadi mahal dan memakan banyak waktu. Banyak riset telah dilakukan untuk mengurangi biaya proses pengembangan sekaligus meningkatkan kualitas dari perangkat lunak yang dihasilkan. Hal ini tidak bisa dilepaskan pula dari kemungkinan pengembangan perangkat lunak tersebut dalam perluasan sistem informasi tiap saat dibutuhkan. Internet yang berkembang begitu pesat dalam satu dekade terakhir, tidak bisa lagi ditinggalkan sebagai faktor yang sangat perlu diperhitungkan dalam perencanaan pengembangan aplikasi dalam lingkungan komputasi terdistribusi. Kebutuhan ini diadopsi oleh semua organisasi untuk meluaskan kebutuhan internalnya dengan tambang informasi yang tersedia di internet, baik sebagai sarana publikasi maupun kolaborasi kerja antar lingkungan dan platform berskala world wide.

Kebutuhan akan pemanfaatan informasi melalui internet, membuat arus pertukaran informasi melalui internet meningkat pesat. Di atas tcp/ip, sudah berjalan begitu banyak aplikasi dan pertukaran informasi. Ini membuat kebutuhan akan ketersediaan bandwidth menjadi sangat tinggi. Perkembangan perangkat lunak pun tidak bisa dilepaskan dari kebutuhan akan hal tersebut. Dalam komputasi lingkungan terdistribusi berskala luas, yang dicari adalah pemanfaatan bandwidth seefisien mungkin, karena ketersediaan infrastruktur internet tidak berjalan sepesat kebutuhan akan ruang pertukaran informasi.

World Wide Web adalah salah satu ruang pilihan tersebut. Maka pengembangan perangkat lunak yang diarahkan pada pemanfaatan halaman web semakin tinggi. Halaman web, sudah mengalami perubahan drastis dari hanya media publikasi statis, menjadi aplikasi yang jauh lebih kompleks. Pengembangan perangkat lunak untuk kebutuhan ini difokuskan pada pengembangan standar metode pertukaran informasi dan format data, untuk memudahkan komunikasi antar beragam sistem yang berbeda lingkungan dan platform pengembangannya.

Webservice dan Sistem Terdistribusi

Webservice adalah proses dan set protokol untuk menemukan dan menghubungkan sebuah sistem dengan perangkat lunak yang diekspose sebagai services melalui web. Web Services dapat berupa apa saja, mulai dari movie review, informasi bursa, nama dan isi cd musik sampai layanan pemesanan hotel dan penerbangan. Infrastruktur teknis Webservice memastikan bahwa layanan-layanan dari vendor-vendor yang berbeda.

Web services mengambil bentuk dari visi object-oriented. Object-oriented merakit perangkat lunak dari komponen-komponen menjadi satu kesatuan aplikasi untuk output yang telah ditetapkan dari aplikasi tersebut. Web services menyatukan berbagai layanan dalam bangunan blok yang loosely coupled. Ada tiga aspek utama pada web services yaitu:

Service provider, yaitu penyedia antarmuka yang dapat membawakan seperangkat task tertentu.

Service requester, yaitu bagian yang mencari dan memanggil software service yang menyediakan layanan bisnis tertentu. Requester ini akan memanggil remote procedure call tertentu, mengirimkan data dan menerima hasilnya.

Broker, yaitu bagian yang mengatur dan mempublikasikan services. Semua permintaan layanan akan melalui broker yang terikat dengan sebuah penyedia layanan tertentu.

XML digunakan untuk mengkodekan seluruh komunikasi ke Webservice. Jadi request dikirimkan dalam bentuk XML message dan diterima dalam bentuk XML response. Oleh karena itu Web services tidak terikat pada satu sistem operasi atau bahasa pemrograman tertentu. Java di Windows dapat berbicara dengan Perl di linux yang terhubung dengan Java di Solaris. Protocol Stack adalah sekumpulan protokol yang digunakan untuk mendefinisikan, menemukan dan mengimplementasikan Web Services. Di dalamnya terdiri dari empat layer:

Services Transport: Lapis ini berfungsi untuk mengirimkan message antar aplikasi. Saat ini, yang digunakan adalah HTTP, SMTP, FTP dan protokol-protokol yang lebih baru lainnya seperti Blocks Extensible Exchange Protocol (BEEP).

XML Messaging: Lapis ini berfungsi untuk mengkodekan messages dalam format umum XML sehingga messages tersebut bisa dimengerti penerimanya. Dalam hal ini adalah XML-RPC dan SOAP.

Service Description: Lapis ini berfungsi untuk mendeskripsikan antarmuka publik ke Web services tertentu. Hal ini dilakukan melalui WSDL (Web Services Definition Language).

Service Discovery: Lapis ini berfungsi untuk memusatkan layanan ke bentuk yang tersusun dan menyediakan cara mudah untuk mempublikasikan dan fungsionalitas untuk menemukannya. Saat ini service ini dilakukan dengan UDDI (Universal Description, Discovery and Integration).

Fenomena yang paling utama dari ide tentang Web Services adalah : terdesentralisasi, loosely coupled dan synergistic

(membawa sinergi dari semua kelebihan web) dengan kebutuhan akan adanya sistem yang lebih reseptif dan adaptif terhadap perkembangan industri. Komunikasi antar sistem membutuhkan sebuah set standar yang berjalan di atas protokol yang dominan dipakai saat ini. Hal itu termasuk kebutuhan untuk menjalankan proses maupun prosedur pada sistem yang berbeda dalam lingkungan distribusi yang beragam dan kompleks. Di antara standar-standar komunikasi yang banyak dibicarakan adalah CORBA (Common Object Request Broker Architecture), DCOM (Distributed Common Object Model), XML-RPC (eXtensible Markup Language-Remote Procedure Call) dan SOAP (Simple Object Application Protocol). XML-Remote Procedure Call dan Simple Object Access Protocol (SOAP) adalah protokol-protokol berbasis XML. Keduanya adalah teknologi webservice yang dihasilkan dari perkembangan lanjutan dari XML. Jadi keduanya secara literal merupakan kombinasi dari Web dan HTTP yang membuka kemungkinan baru pertukaran data antar lingkungan yang terhubung dalam jaringan. SOAP adalah perspektif middleware pertukaran data terdistribusi dan interaksi yang loosely coupled (tak terikat erat). SOAP saat ini merupakan rekomendasi World Wide Web Consortium (W3C). XML-RPC adalah spesifikasi final yang lebih sederhana dan mudah diimplementasikan dibanding SOAP. Secara fundamental perubahan yang dibawa oleh kedua adalah kemampuan untuk memindahkan data kemana pun melalui web. Sebelumnya hal itu hanya mampu ditangani oleh Electronic Data Interchange (EDI). EDI mendefinisikan message dan protocol yang digunakan untuk pertukaran data melalui jaringan. Tapi hal ini mengunci jaringan-jaringan yang tergabung pada satu standar tertentu, dan membuat mahal implementasi pada jaringan baru yang akan digabungkan. Pendekatan berikutnya adalah membangun infrastruktur obyek terdistribusi (distributed object infrastructure) yang berjalan di atas internet. Ada beberapa standar yang kita kenal, yaitu: Common Object Request Broker Architecture (CORBA), Remote Method Invocation (RMI) dan Distributed Component Object Model (DCOM). Masing-masing memilih protokol sendiri yang berjalan di atas TCP/IP untuk menangani komunikasi antar obyek-obyeknya. Yang sangat populer dan sering dipertentangkan sebagai standar adalah DCOM dan CORBA. DCOM merupakan teknologi lanjutan dari Component Object Model (COM) dari Microsoft. COM menjadi dasar komunikasi antar proses obyek-obyek pada sistem operasi keluarga Windows. Sementara CORBA menjadi dasar penting komunikasi antar proses obyek-obyek pada sistem operasi jaringan non-Windows dan Java. CORBA menggunakan Internet Inter-ORB Protocol (IIOP), sementara DCOM menggunakan Object Remote Procedure Call (ORPC) dan RMI menggunakan Java Remote Method Protocol (JRMP). Pendekatan ini berjalan bagus pada lingkungan masing-masing, memecahkan masalah bahasa dan sebagian lingkungan pengembangan, tapi meninggalkan kesulitan pada komunikasi antar mereka. Artinya CORBA hanya mampu berbicara dengan sistem lain yang dibangun dengan standar CORBA, demikian juga berlaku pada DCOM dan RMI. Untuk saling berbicara, maka sistem yang heterogen harus ditambahkan layer ekstra pada arsitektur yang sudah begitu kompleks tersebut. Jika kita membangun sistem berbasis aplikasi yang secara alamiah menggunakan COM dan Active Server Pages (ASP), sistem tersebut hanya mampu berbicara dengan sistem lain yang menggunakan standar COM juga, dalam hal ini keluarga Windows. COM ini menjadi DCOM pada server-server windows yang saling berbicara mengkomunikasikan proses-proses yang berjalan dalam sistem mereka. Jikalau memungkinkan dibangun aplikasi menggunakan ASP di lingkungan UNIX/Linux maka obyek-obyek yang bekerja akan menggunakan standar CORBA yang tidak bisa berbicara dengan COM di Windows kecuali dibangun antarmuka sebagai lapisan ekstranya. Padahal pada teknologi ini sebenarnya selain semestinya mendukung interoperabilitas antar obyek-obyek yang sudah dibuat juga komunikasi antar obyek untuk kepentingan pertukaran data. XML-RPC dan SOAP dibangun dengan mengkombinasikan kemampuan data XML dan kemampuan transport HTTP, yang dengan demikian memecahkan masalah EDI dan sistem obyek terdistribusi seperti CORBA, COM dan RMI. XML-RPC dan SOAP dikerjakan dengan mengubah sekumpulan parameter (skalar, string, tanggal, array, record, data biner) ke transmisi XML. XML-RPC didefinisikan beroperasi di atas koneksi HTTP, sementara SOAP mendeskripsikan format amplop-nya untuk RPC request yang bisa dikirimkan melalui HTTP, SMTP, FTP atau beberapa protokol TCP/IP lainnya. XML-RPC melewati parameternya dengan posisi sementara SOAP melewati parameternya dengan nama. Keduanya mampu melewati data biner dengan menggunakan pengkodean Base-64. Yang jelas dalam hal ini XML-RPC lebih mudah diimplementasikan meskipun tidak memiliki fitur sebanyak yang dimiliki SOAP. Salah satu kebutuhan tambahan dalam implementasi di antara cara-cara di atas, adalah kemudahan dan pemanfaatan standar pertukaran data antar beragam sistem di atas protokol HTTP, yaitu XML. Format XML paling memungkinkan komunikasi antar proses pengolahan data pada sistem yang berbeda dengan format yang mampu menampung parameter sekaligus metode bagaimana parameter tersebut harus diproses. Procedure Call adalah pemanggilan procedure yang berada pada remote environment, untuk menjalankan proses dengan parameter yang dikirimkan pemanggil agar memberikan respon yang diinginkan atas sebuah rangkaian proses komputasi. XML adalah format yang paling mudah dimengerti oleh manusia dan komputer, dengan keleluasaannya mengatur struktur data dan bahasa yang pendefinisianannya lebih mudah. SOAP dan XML-RPC adalah dua metode yang memanfaatkan XML untuk format pengiriman parameter dan pemanggilan procedure dalam internal sebuah sistem maupun remote system. XMLRPC XML-RPC adalah pemanggilan prosedur jarak jauh melalui HTTP dengan menggunakan XML sebagai cara pengkodeannya. XML-RPC adalah salah satu metode komputasi terdistribusi, webservice yang paling sederhana, dan implementasinya sudah digunakan secara luas, dalam berbagai bahasa pemrograman dan platform. Message XML-RPC adalah sebuah HTTP-POST request. Isi (body) dari request tersebut berupa XML procedure yang dieksekusi di server dan nilai hasil eksekusi dikembalikan dalam bentuk XML juga. XML-RPC menyerahkan kerumitan yang harus dipikirkan bagaimana message dikirimkan antar server atau obyek-obyek yang berinteraksi dalam lingkungan terdistribusi, kepada HTTP. Fokus utamanya adalah apa yang akan disampaikan bukan bagaimana menyampaikan

pesan. Contoh request XML-RPC adalah sebagai berikut: POST /PC2 HTTP/1.0 User-Agent: Frontier/5.1.2 (WinNT) Host: betty.userland.com Content-Type: text/xml Content-length: 181 examples.getStateName41 XML-RPC awalnya dikerjakan oleh Dave Winner dari Userland (kemudian menjadi salah satu kontributor SOAP, sebagai teknologi lanjutan yang diperluas dari XML-RPC). Masalah klasik yang mesti dipecahkannya mula-mula adalah bagaimana membuat perangkat lunak yang bisa berjalan di lingkungan yang berbeda saling berbicara. Komunikasi lintas-platform ini didemonstrasikan Winner dengan meletakkan perintah-perintah remote procedure pada body HTTP POST. Selanjutnya hanya diperlukan kosakata XML yang mendefinisikan nama-nama potongan kode perintah jarak jauh dan parameter-parameter yang diperlukannya. Elemen XML pada XML-RPC dengan sederhana mendefinisikan kosakata untuk menyampaikan informasi prosedur-prosedur mana yang akan dieksekusi pada remote-server. Ketika server menerima message XML-RPC dari HTTP POST request, dokumen XML-nya akan digunakan untuk memicu sebuah remote-procedure dan hasilnya dikirimkan balik pada pengirimnya sebagai XML juga. Dengan semangat pakai-ulang (reusability), XML-RPC menggunakan tipe data XML Schema untuk parameter dan prosedur yang akan dipanggilnya. Berikut ini adalah daftar tipe dataskalar untuk parameter XML-RPC dan nilai-nilainya. Tipe Parameter Skalar untuk XML-RPC Tag Tipe Contoh atau four-byte signed integer-897 0 (false) atau 1 (true) 1 ASCII string blaah double precision signed floating point number-78.23 date/time 20032224T20:01:01 base64-encoded binary 7HYBsu76HT7HJD Dengan begitu dokumen XML yang tersusun baik bisa dipindahkan dengan mudah menggunakan internet. Syaratnya cuma server yang sudah menjalankan HTTP service dan memiliki kemampuan menandai dokumen XML yang datang sekaligus mengenali dan menguraikan elemen-elemen-nya agar dapat mengeksekusi prosedur apapun yang dispesifikasikan dalam elemen metode Call. XML-RPC mensyaratkan hal-hal sebagai berikut sebelum dapat digunakan:

Dokumen XML harus tersusun baik dan mengandung stuktur tunggal elemen method Call.

Elemen method Call harus mengandung sub-sub item methodName yang berisi string nama method mana yang bisa dipanggil.

Kalau parameter diperlukan maka elemen method Call harus mengandung parameter yang berisi individual elemen-elemen param, dan masing-masing mengandung nilai tunggal.

XML-RPC Response Sesuai aturannya, maka setelah diproses, dan mengeksekusi beberapa prosedur dan kode, maka dapat memberikan nilai balik berdasarkan hasil procedurnya atau elemennya tidak valid. Bagaimana proses itu dilaksanakan, selanjutnya tinggal melihat bagaimana hasil procedure tersebut dikirimkan balik atau elemen fault. Di atas HTTP maka semua dianggap data. Spesifikasi XML-RPC menyebutkan bahwa hasil remote procedure call adalah XML berstruktur tunggal yaitu methodResponse, yang berisi hasil eksekusi dalam parameter tunggal atau elemen fault yang berisi informasi apa saja yang menyebabkan kesalahan eksekusinya. Contoh response dari XML-RPC request: HTTP/1.1 200 OK Connection: Close Content-Length: 158 Content-Type: text/xml Date: Fri, 17 Jul 1998 19:45:23 GMT Server: Userland Frontier/5.1.2-WinNT California Kecuali ada level error yang lebih rendah lagi, maka response selalu 200 OK. Content-Type adalah text/xml dan Content-Length harus ada dan benar. Isi (body) dari response berupa XML berstruktur tunggal, sebuah , yang berisi tunggal, yang berisi tunggal, yang berisi value tunggal bertipe string. MethodResponse dapat juga berisi yang berisi yang -nya berisi dua elemen, berupa dan berupa . Method Response juga dapat berisi keduanya dan . Contoh method Reponse dengan : HTTP/1.1 200 OK Connectio: close Content-Length: 426 Content-Type: text/xml Date: Fri, 17 Jul 1998 19:46:00 GMT Server: Userland Frontier/5.1.2-WinNT faultCode 4 faultString Too many parameters. Strategi/Tujuan Firewall. Tujuan protokol ini meletakkan landasan yang kompatibel lintas lingkungan-lingkungan yang berbeda. Tak ada sesuatu terlalu rumit ditambahkan di atas antarmuka CGI. Firewall bisa mengawasi semua transaksi Content-Type yang berupa text/xml Discoverability. Yaitu berupa penyusunan format yang sangat sederhana yang memungkinkan penulis HTML biasa untuk melihat file yang berisi XML-RPC, mengerti apa yang sedang dilakukan dan dapat memodifikasinya dan membuatnya bekerja kembali pada penggunaan berikutnya. Mudah diimplementasikan. Membuat sebuah protokol yang mudah diimplementasikan dan bisa diadaptasikan di lingkungan lain atau sistem operasi lainnya. Penutup XML-RPC menyederhanakan banyak hal. Dibanding SOAP, XML-RPC mungkin terlalu sederhana. Tapi juga justru itulah salah satu keunggulan XML-RPC dibandingkan arsitektur lainnya. Bersama dengan SOAP, XML-RPC bersifat terbuka, bisa dan sudah diimplementasikan di banyak platform sistem operasi dan bahasa pemrograman. Saya kira penggunaan Webservice mengatasi masalah komunikasi antar platform, karena selain menggunakan standar terbuka juga tidak mengacu pada satu lingkungan tertentu yang bersifat proprietary. Itulah salah satu sebab implementasi XML-RPC sudah luas sekali, bukan hal aneh lagi bahkan sudah digunakan pada aplikasi-aplikasi cms atau blog yang populer seperti serendipity, drupal, postnuke atau mambo. sumber: www.xmlrpc.org, beberapa buku dan majalah

Posted by Meta Nurwidyanto in ICT at 07:29

makasih atas artikelnya, coz berguna banget buat saya yang bentar lagi mo TA. moga sukses dan artikelnya ditambah lagi, biar yang mo TA gak usah cari buku susah-susah and mahal lagi.

Keep Rockin
Anonymous on Mar 30 2008, 04:20

Blog Export: Meta Soliloquy Blog, <http://meta.wacana.net/>

Post Bagus mas :), saya lagi pengen buat aplikasi berbasis xml-rpc ini.
Anonymous on Dec 18 2008, 12:55

artikel yang bagus.. kalau ada posting donk contoh penerapannya.
Anonymous on Mar 24 2009, 15:46

nice article bang...
kalo boleh buat ulasan lagi dong tentang JSON RPC agar lebih maknyuuuussss

saya mau nanya donk...
bedanya SOA ama RPC apa????
dua duanya kan pake protokol tuuhhh
Anonymous on Oct 4 2009, 19:11

JSON RPC menarik, lebih enteng dan bisa di javascript-kan sepenuhnya karena formatnya memang JSON (Javascript Object Notation) yang dipakai buat RPC. Tapi belum bisa atau setidaknya belum saya pakai/coba adalah json rpc (yg pake json format) buat ngirim binary data (jadi base64 encoding, kalau di xmlrpc). XMLRPC dgn 6 tipe data saja, sudah saya pergunakan bertahun-tahun dengan cukup memuaskan hati saya sendiri.

SOA itu arsitektur, kalau SOAP itu protokol webservices. SOAP dan XMLRPC sama2 webservices. Bedanya, SOAP lebih complicated tapi lebih mendukung banyak protokol transport (tidak hanya http seperti xmlrpc). SOAP tipe datanya jauh lebih banyak, tapi datanya juga bengkak sekali. Payload-nya bisa naik lebih dari 30%. XMLRPC saja naik segitu, apalagi SOAP. Tapi jika pada implementasinya disertai proses kompresi seperti phpxmlrpc (misal), maka itu sangat menolong meringankan http payload.

Yang menurut saya layak dipertimbangkan buat di-hack (digali lebih dalam) selain Webservices XMLRPC adalah JSON-RPC dan REST.
Anonymous on Oct 5 2009, 15:17

waaahh terimakasih atas infonya bang...
saya mau bertanya lagi neehh,, seandainya saya ingin membandingkan suatu performansi web service XML-RPC dan JSON -RPC dimana yang menjadi acuan saya adalah message interchange(pertukaran data) dalam bentuk JSON dan XML, kira kira parameter uji ukur yang tepat disisi client dan servernya apa ya bagusnya????
sample client untuk uji suatu web service itu ada standarnya g siihh????

terimakasih sebelumnya,,
salam kenal bang,,
Anonymous on Oct 7 2009, 00:13

Kalau membandingkan, coba terapkan di platform yg sama. Server/Client Java semua baik json-rpc maupun xmlrpc, atau php semua atau delphi semua. Tapi itu tergantung parser atau library yg digunakan juga. Kalau pertukaran data saya rasa json-rpc masih kalah dari xmlrpc soal base64 encoding. xmlrpc sendiri kalah banyak support protokol dan tipe data dibanding soap.
Efisiensi message interface tergantung library atau pemrograman terapannya. Coba pakai asumsi, jadi memang dari awal sudah memihak salah satu protokol. hehehe. Kalau json-rpc rasanya paling efisien untuk webservices yang melayani halaman-halaman AJAX. json-rpc + ajax, paling bagus dibanding xmlrpc + ajax atau soap + ajax. Aku rasa aspek interoperability-nya yg perlu ditinjau: development platform (bahasa pengembangan), environment platform (OS, Virtualisasi), kemudahan implementasi, pustaka pendukung (java lib, php lib, delphi lib) dan kebutuhan (pada kasus tertentu).
Yang pernah saya uji baru php dgn delphi (client-server dibolak-balik), sama-sama cepat dan mudah, php command line /vbscript command line (windows scripting host) dgn php/delphi, terakhir groovy (java) dengan php. Well, now xmlrpc implementation has never been so easy. have fun!
Anonymous on Oct 7 2009, 09:47

waahhh terimakasih bang informasinya,,,
saya ingin menguji web servic dengan protokol xml-rpc dan json-rpc dimana kira kira aplikasi yang bagus untuk dibangun seperti apa ya bang apakah booklist(seperti AWS amazon web service) atau ada aplikasi lain yang bisa disarankan bang?? dimana bahasa yang dipake rencananya adalah PHP dalam hal ini saya hanya membandingkan pengaruh format JSON dan XML pada kasus RPC web service...

terima kasih atas informasinya,,, saya sangat tertarik sekali dengan teknologi web service ini,,,
Anonymous on Oct 7 2009, 12:55

Kalau soal format, json jelas lebih kecil payload-nya, tapi seperti saya jelaskan, kalau library-nya sudah mendukung kompresi, xmlrpc jadi setara dengan json. Masih tergantung libray yg dipakai apa dulu. Halaman AJAX saya mungkin pilih json.

Coba misalnya perpustakaan online. PHP is fine. Bisa diakses dari client sms, web, symbian, j2me (tapi berguna tidak ya, hehehe, since mini opera sudah bisa jalan web dan ajax juga) pakai xmlrpc webservice. JSON, seperti saya bilang mesti diakali buat ngirim data binary (misal image cover buku), misalnya dengan cara binary code to text scheme, Tapi prinsipnya sama. Kalau pakai php, kebetulan phpxmlrpc bisa digunakan juga library-nya sebagai json-rpc. good luck.
Anonymous on Oct 7 2009, 13:35

bang saya mau nanya dong,,,
untuk web service itu sendiri kan memakai tiga tahap
1. description
2. package

Blog Export: Meta Soliloquy Blog, <http://meta.wacana.net/>

3. discovery

naaahh untuk study kasus packaging menggunakan XML-RPC proses descriptionnya bisa ga menggunakan WSDL... terus untuk discoverynya menggunakan UDDI.. (karena setau saya yang bisa menggunakan WSDL dan UDDI itu cuman protocol web service jenis SOAP, tapi ga tau juga bener apa ga maklum bang saya masih newbie..) mohon bimbingannya....

Anonymous on Nov 6 2009, 18:52

ok, thank u atas infonya, lagi coba belajar XML RPC nih:)

Anonymous on May 26 2010, 17:21

makasih bang. sangat berguna sekali konsep nya

Anonymous on Apr 3 2011, 10:45

keren mas penjelasan nya. makasih banyak ya mas.

Anonymous on Dec 2 2011, 13:23

Thanks for publishing this information to the web.

Anonymous on Dec 26 2011, 15:23

Everyone loves what you guys are usually up too. This kind of clever work and coverage! Keep up the excellent works guys I've included you guys to our blogroll.

Anonymous on Feb 17 2012, 16:26