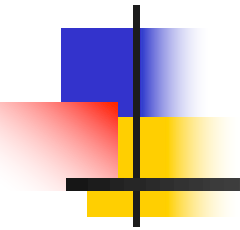
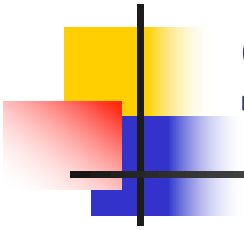


Web Services



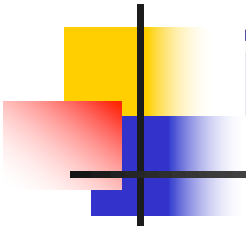
Šta su Web Servisi?

- Navikli smo da web sajtovima pristupamo direktno iz browsera
  - Browser pronalazi dokument na osnovu URL
  - Browser i server razmenjuju zahteve i odgovore, odgovor servera je najčešće HTML. Sve se transportuje preko HTTP
- Web Service uopštava ovaj model tako da računari (softverske komponente) mogu direktno da komuniciraju (putem interneta)

Istorijat

- Web servisi su razvijeni na osnovu prethodnih tehnologija koje su služile istoj nameni- RPC, ORPC (DCOM, CORBA i JAVA RMI).
- Web Servisi su somišljeni da reše neke osnovne probleme:
  - Interoperabilnost
  - Prolazak kroz Firewall
  - Kompleksnost rešenja

Prethodni problem - interoperabilnost



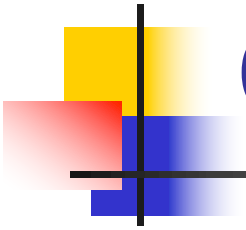
- Raniji distribuirani sistemi su imali probleme sa interoperabilnošću jer je svaki proizvođač implementirao sopstvene formate za razmenu poruka između distribuiranih objekata.
- DCOM je na primer bio vezan za Windows OS.
- RMI je bio striktno vezan sa Javom.

Prethodni problem - firewall-i

- Saradnja između korporacija je bila problem jer su sistemi poput CORBA i DCOM koristili nestandardne portove.
- Web Servisi koriste HTTP kao transportni protokol, a većina firewalla je konfigurisana da propušta konekcije na port 80 (HTTP).

Prethodni problem - kompleksnost

- Web Servisi su u suštini developer-friendly.
- Većina prethodnih tehnologija kao što su RMI, DCOM, i CORBA zahtevali su mnogo više učenja i navikavanja na koncepte.
- Često je bilo potrebno naučiti dodatne jezike ili tehnologije.



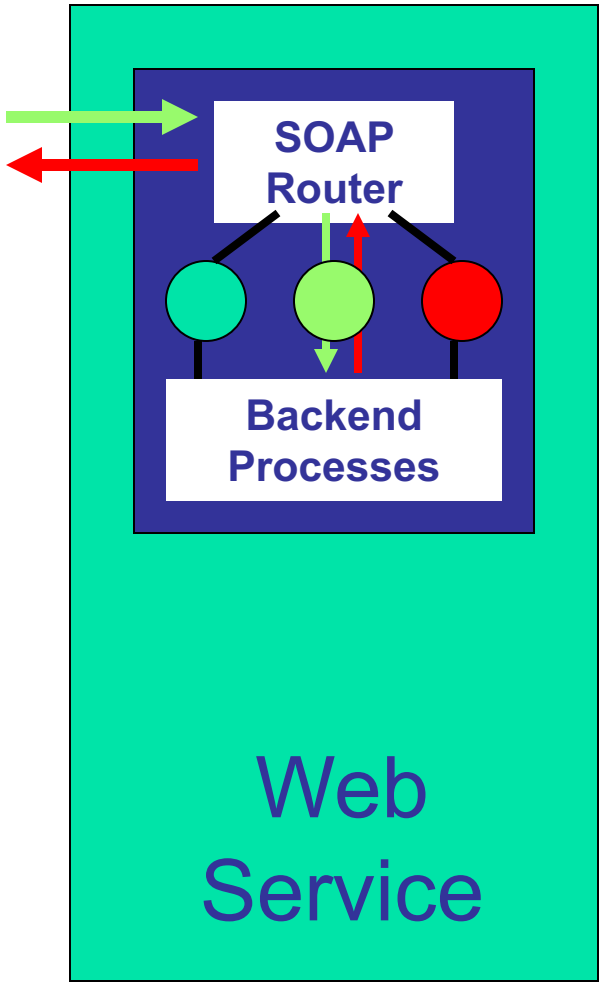
CORBA pristup

- CORBA je imao
  - Način da se eksportuju “client stubs”
    - “client stub” je mogao da sadrži i određenu logiku za odlučivanje koju je server isporučivao npr. “na koji datacentar se povezati”
    - Na ovaj način se serveru daje određeni vid daljinske kontrole
  - Factory servisi: kreiranje određenih vrsta objekata na zahtev
    - “pronalaženje” se moglo realizovati i kao aktivnost “kreiranja servisa”

CORBA je objektno orijentisana

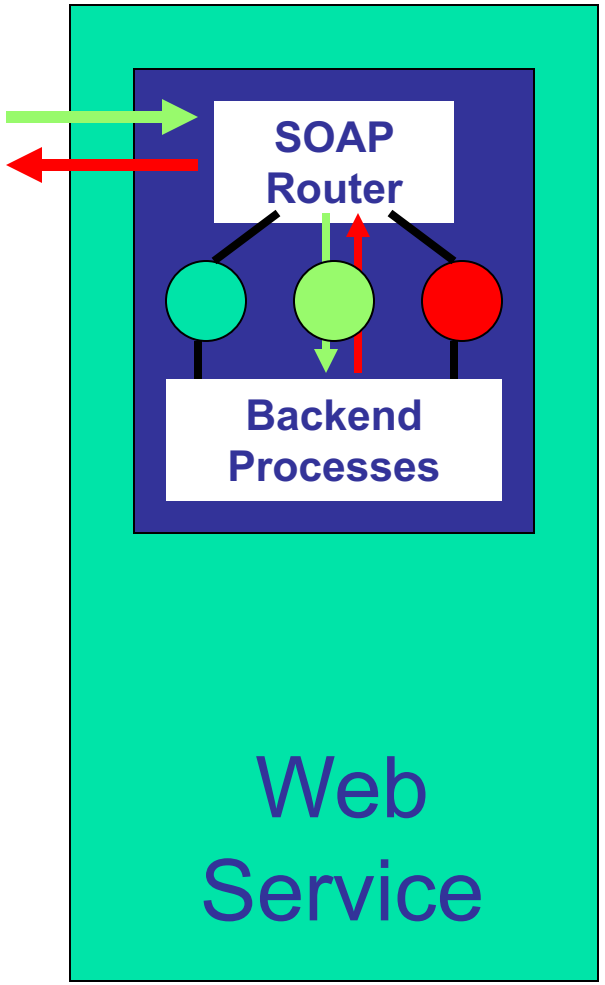
- CORBA je u potpunosti objekt-centrična
  - Objekti mogu biti
  - … pasivni (podaci)
  - … aktivni (programski)
  - … perzistentni(podaci koji se snimaju)
  - … volatile (stanje definisano samo tokom izvršavanja)
- U CORBA-i aplikacija koja upravlja objektom je nerazdvojiva od samog objekta
  - “client stub” je praktično deo aplikacije

Šta su onda Web Servisi?



Šta su Web Servisi?

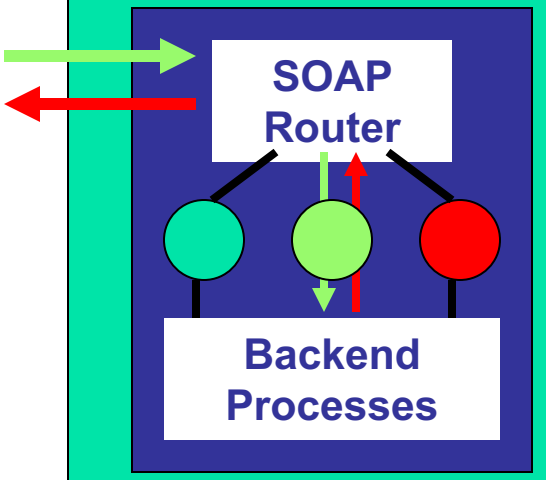
- "Web servisi su softverske komponente opisane pomoću WSDL, a kojima je moguće pristupiti preko standaradnih mrežnih protokola (SOAP over HTTP)"



Šta su Web Servisi?

- "Web servisi su softverske komponente opisane pomoću WSDL, a kojima je moguće pristupiti preko standaradnih mrežnih protokola (SOAP over HTTP)"

SOAP je primarni standard. Obezbeđuje pravila za kodiranje zahteva i njegovih argumenata – format pakovanja poruke.

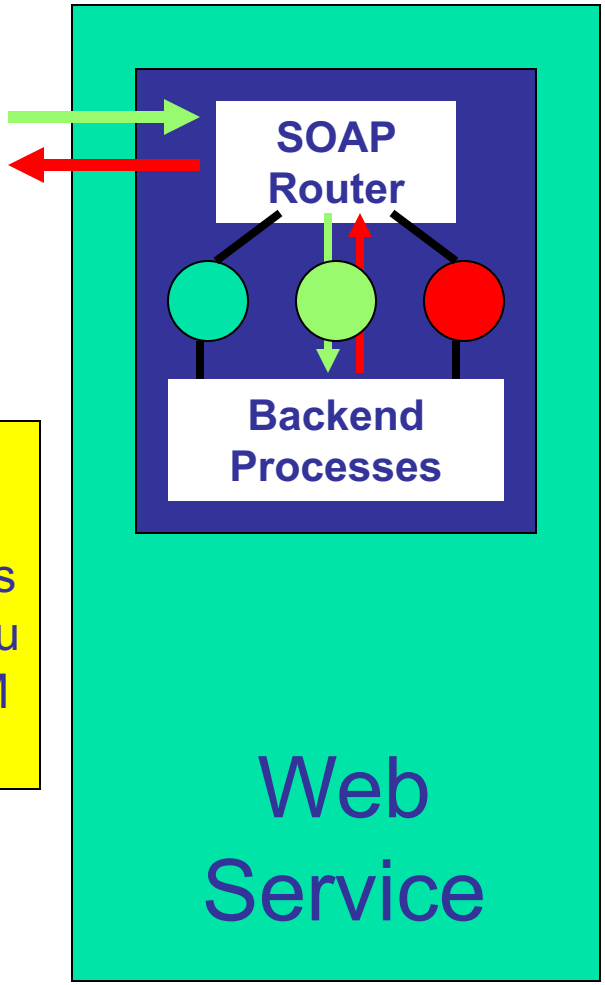


Web
Service

Šta su Web Servisi?

- "Web servisi su softverske komponente opisane pomoću WSDL, a kojima je moguće pristupiti preko standaradnih mrežnih protokola (SOAP over HTTP)."

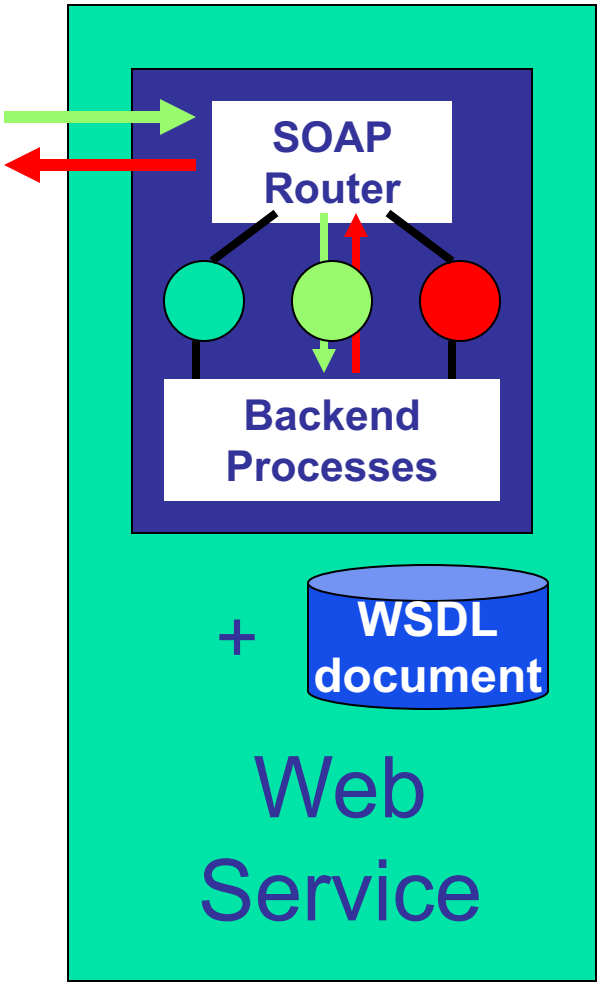
Arhitektura WS ne pravi pretpostavku da mora da se radi pristup preko HTTP-a na TCP-u. Na primer .NET koristi Web Services "interno" ćak i u okviru jedne mašine. Ali se u tom slučaju komunikacija odvija preko COM ili WCF



Šta su Web Servisi?

- "Web servisi su softverske komponente opisane pomoću WSDL, a kojima je moguće pristupiti preko standaradnih mrežnih protokola (SOAP over HTTP)."

WSDL dokumente razvojni alati koriste kao uputstvo za formiranje objekata, generisanje koda…

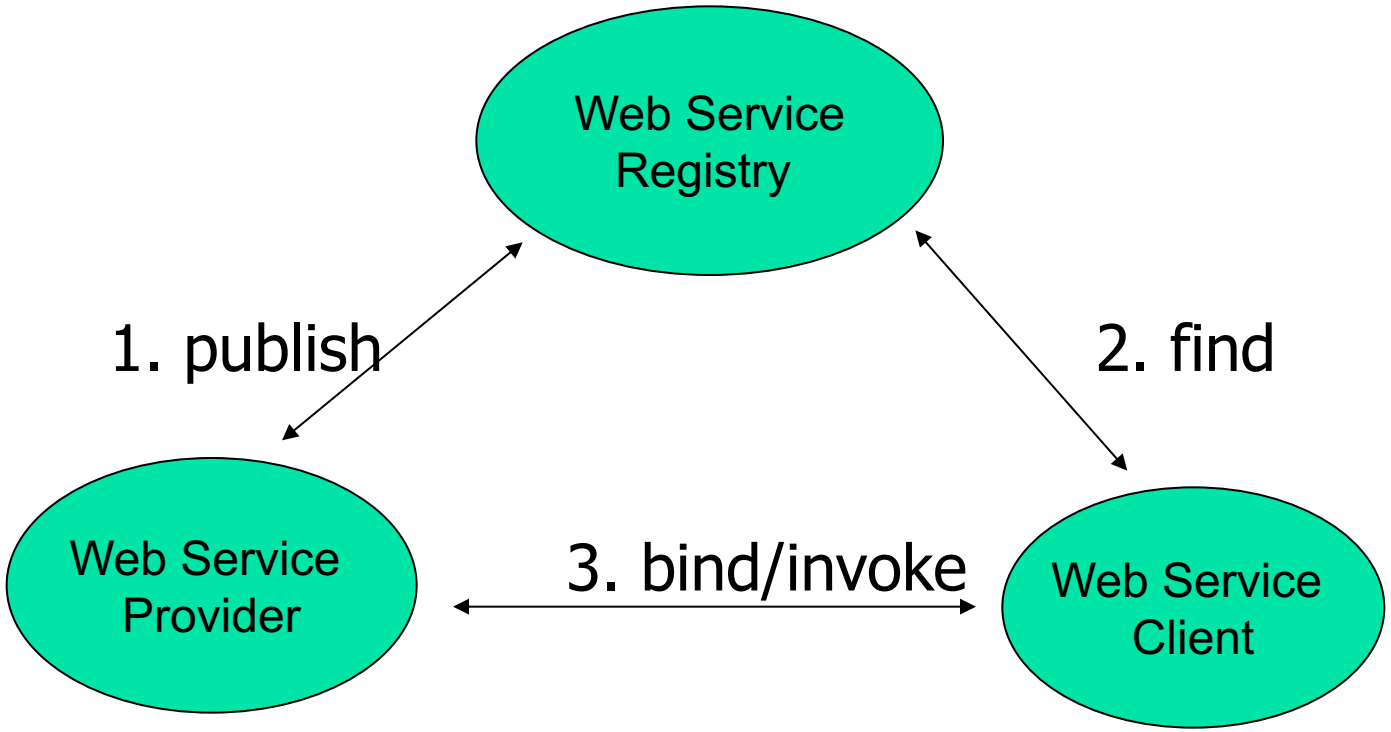


Komponente koje čine Web servise

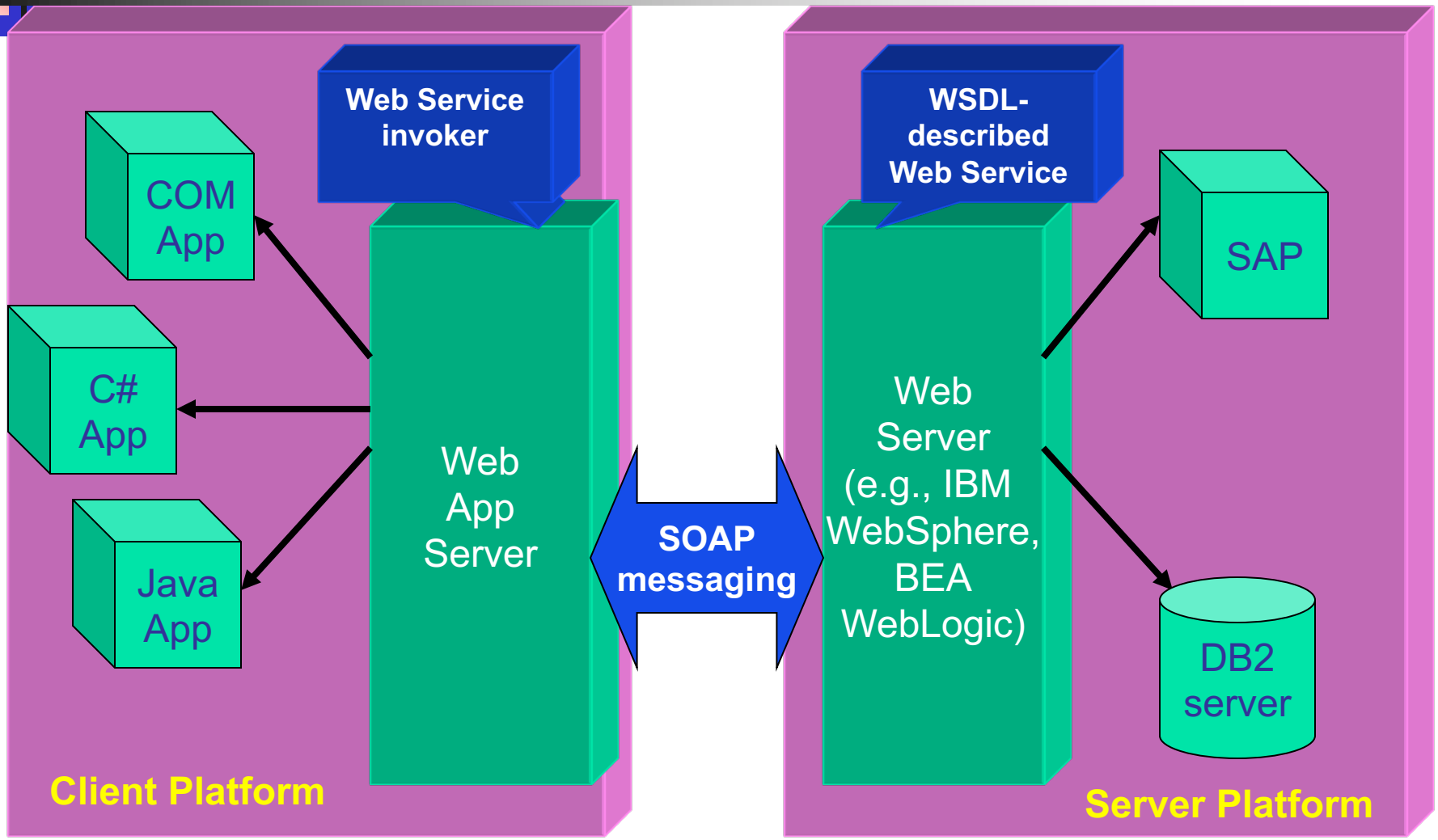
- **XML** – Koristi se kao jedinstven način reprezentacije i razmene podataka.
- **SOAP** – (nekada Simple Object Access Protocol) – standardni način komunikacije.
- **UDDI** – Universal Description, Discovery and Integration specification - mehanizam za registraciju i lociranje servisa u katalogu.
- **WSDL** – Web Services Description Language – standardni meta jezik za opis ponuđenih servisa.

Model web servisa

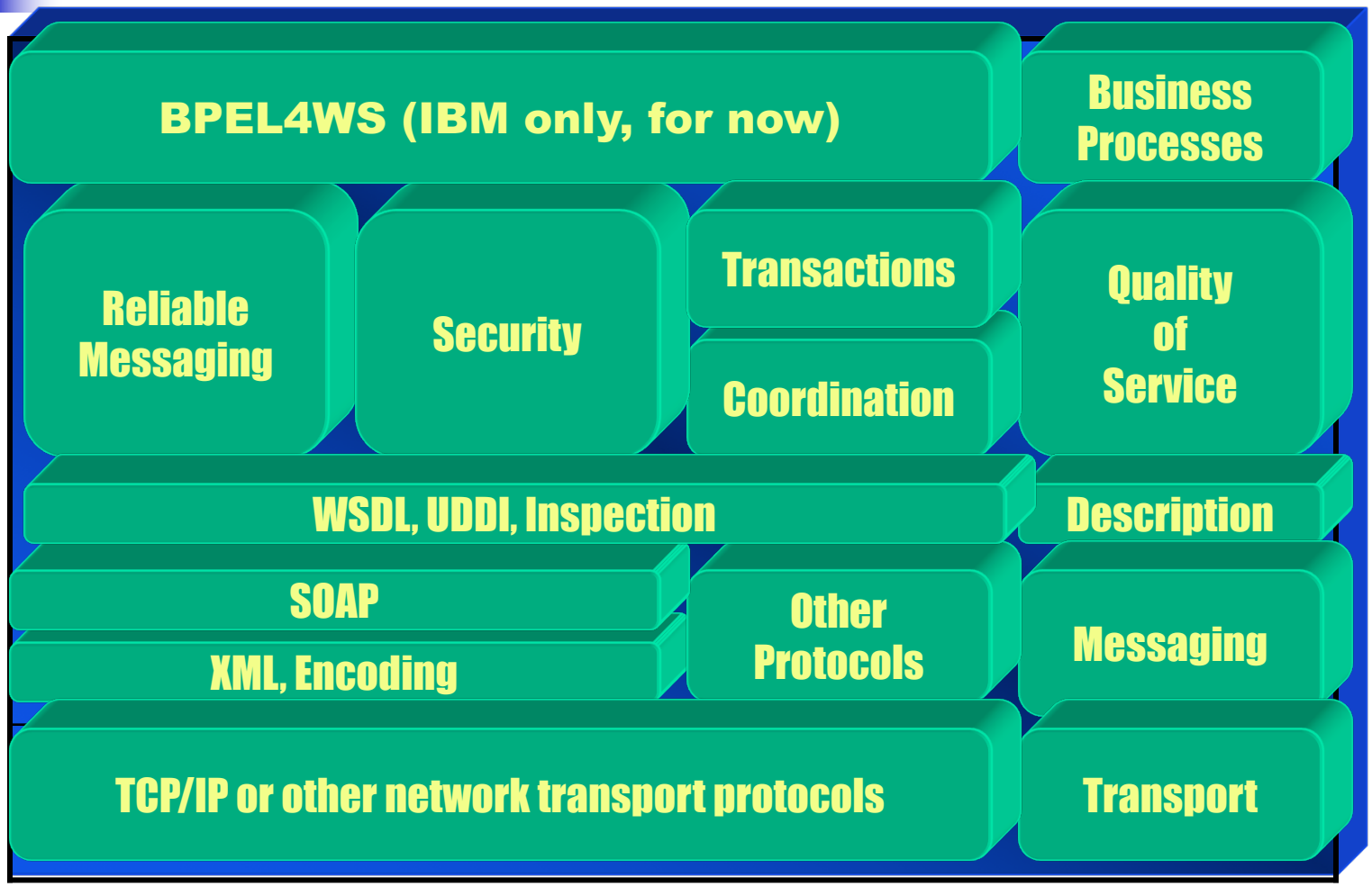
publish, *find*, i *bind* paradigma.



Web Servisi su često front-end vaše aplikacije



“Stack” Web servisa



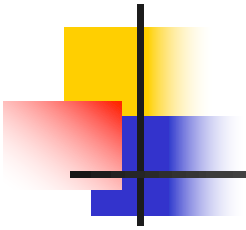
Web Servisi su dokument-centrični

- Sva komunikacija se obavlja slanjem dokumenata na server i primanjem dokumenta-odgovora sa servera
- Većina garancija na sadržaj i podatke je vezana za te dokumente a ne za sam servis
  - WS_RELIABILITY se ne odnosi na pouzdanost samog servisa, već definiše pravila kako se zahtevi za pouzdanost iskazuju i kako se povezuju sa dokumentom.
  - Nasuprot tome - CORBA fault-tolerance standard je specificirao kako obezbediti da CORBA servis bude visoko dostupan.

Prednosti Web servisa?

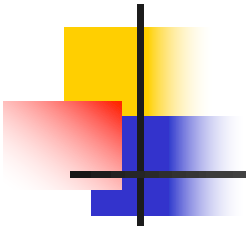
- Obezbeđuju interoperabilnost između heterogenih sistema koji se izvršavaju na heterogenim platformama.
  - "agnostik po pitanju proizvođača, platforme i programskog jezika"
- Web servisi koriste otvorene standarde i protokole. Kad god je moguće protokoli i formati podataka su tekst-bazirani
  - Relativno lako razumljivi.
- Time što se prenos podataka vrši "na leđima" HTTP web servisi mogu raditi kroz većinu uobičajenih firewall-a, a da pri tome nije neophodno njihovo posebno podešavanje

Kako rade web servisi



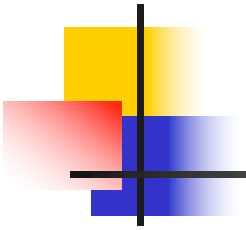
- Klijent prvo *pronalazi* servis.
- Tipično klijent potom izvršava povezivanje na servis (binding).
  - Tako što se uspostavi TCP veza ka adresi pronađenoj u opsiu servisa.
  - Ovo povezivanje na servis nije uvek neophodno.

Kako rade web servisi



- Klijent potom kreira SOAP zahtev: (*Marshaling*)
  - Popunjava informacije o tome koji servis je potrebno izvršiti, kao i podatke (argumente) koji su servisu neophodni za izvršavanja. Zapakovan zahtev šalje serveru.
  - XML je standardni način "pakovanja" podatka (ali uvodi jako veliki "overhead")
- SOAP ruteri usmeravaju zahtev na odgovarajući server (ukoliko ih ima više od jednog)

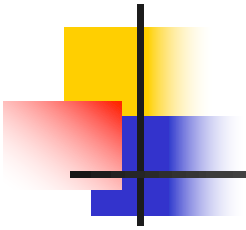
Kako rade web servisi



- Server raspakuje zahtev (*Unmarshaling*) obrađuje ga i proizvodi rezultat.
- Rezultat se šalje nazad.

Problemi sa *Marshalling-om*

- Podaci koji se razmenjuju moraju biti predstavljeni u platformski nezavisnom formatu.
  - "Endian"ness se razlikuje na različitim platformama.
  - Pitanje preciznosti (poravnanja) podataka (16/32/64 bits)
  - Različite reprezentacije *floating pointa*.
  - Pokazivači
    - Podrška za legacy sisteme



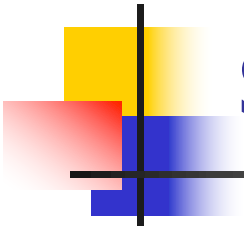
Pronalaženje servisa

- Problem pronalaženja “pravog” servisa
  - Jedan način pomoću URL
  - Kod Web servisa se koristi i URN: Uniform Resource Name
- Uopšteniji pristup je da se koristi posrednik: servis za pronalaženje servisa (katalog)

Primer repozitorijuma (kataloga) servisa

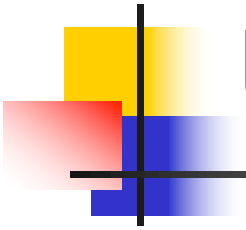
Name	Type	Publisher	Toolkit	Language	OS
Web Services Performance and Load Tester	Application	LisaWu		N/A	Cross-Platform
Temperature Service Client	Application	vinuk	Glue	Java	Cross-Platform
Weather Buddy	Application	rdmgh724890	MS .NET	C#	Windows
DreamFactory Client	Application	billappleton	DreamFactory	Javascript	Cross-Platform
Temperature Perl Client	Example Source	gfinke13		Perl	Cross-Platform
Apache SOAP sample source	Example Source	xmethods.net	Apache SOAP	Java	Cross-Platform
ASS 4	Example Source	TVG	SOAPLite	N/A	Cross-Platform
PocketSOAP demo	Example Source	simonfell	PocketSOAP	C++	Windows
easysoap temperature	Example Source	a00	EasySoap++	C++	Windows
Weather Service Client with MS- Visual Basic	Example Source	oglimmer	MS SOAP	Visual Basic	Windows
TemperatureClient	Example Source	jgalyan	MS .NET	C#	Windows

Repozitorijum (katalog) servisa



- Praktično baza koja izlistava spisak servera koji nude servise
- Opian je korišćenjem UDDI jezika (XML notacija)
  - Samim tim moguće ga je pretraživati na iste načine kao i bilo koji XML
- Proširiv standard
  - Definiše određene zahtevane informacije (dostupne interfejse, tipove argumenata…)
  - Ali sami servisi mogu obezbediti dodatne informacije.

Uloga kataloga i opisa

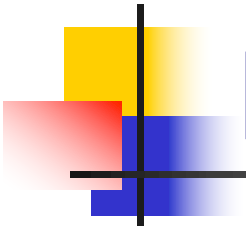


- UDDI se koristi da opiše informacije koje predstavljaju jedan zapis u katalogu
- WSDL dokumentuje interfejse i tipove podataka koje servis koristi

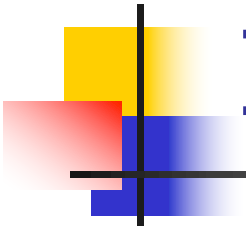
Pronalaženje i konvencije imenovanja

- The topic raises some tough questions
  - Dosta okruženja, poput veliki data-centara, često imaju neku standardnu strukturu (sadržaja i servisa). Da li se pronalaženje može automatizovati?
  - Kako debugovati aplikaciju ako se ponekad poveže sa pogrešnim servisom?
  - Delegiranje i migracija mogu biti problematični
  - Da li sistem može i treba da automatski pokrene servis na zahtev?

Zašto je otkrivanje servisa problematično

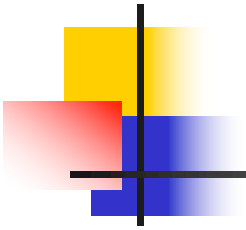


- Client ima svoje viđenje stvari
  - "Želim na mapi prikazane dužine svih linija prevoza i to odmah"
- Service ima svoje
  - Amazon.com npr. želi da svi zahtevi iz mesta Ithaca budu usmereni na njegov NJ-3 datacenter, i ukoliko je moguće na istu instancu unutar klastera
- DNS ima svoje
  - Dosta sistema se "poigrava" sa mapiranjem imena na IP adrese
- Internet ima svoje (kako obaviti rutiranje)



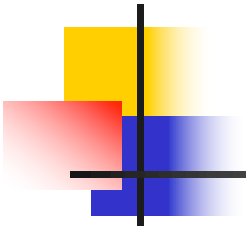
I šta je stvarno problem?

- Web Servis ne opisuje neki standardizovani način kako obaviti ove korake pretpostavlja da ih je na neki način moguće obaviti
- UDDI i WSDL su samo deo celokupne slike!



Network address translation…

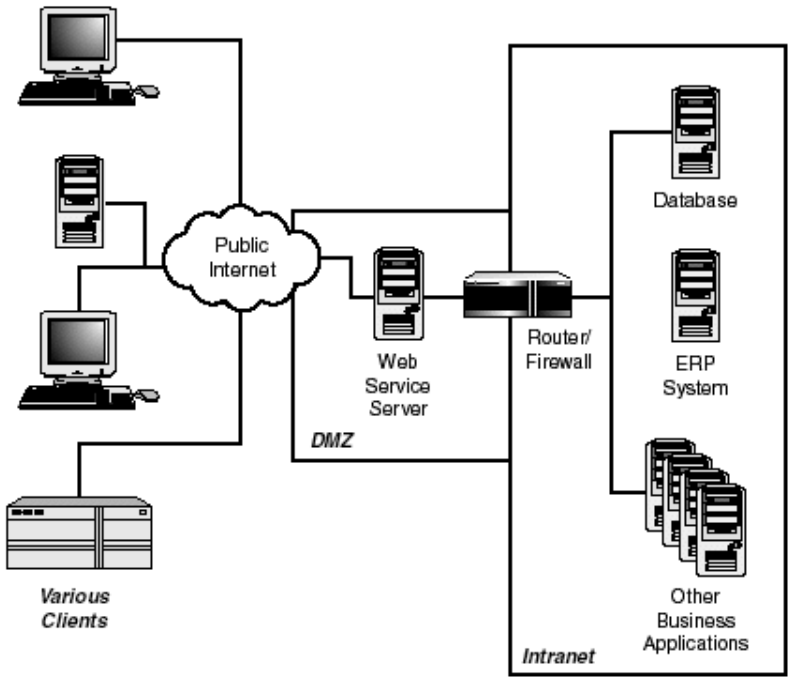
- Ovo je dodatni problem: interne adrese servera na mreži često i nisu dostupne spolja!
  - Dodaje se ponekad i kao mera zaštite.
  - Ali ako je RPC ili WS server iza NAT onda je to problem!
    - Neophodno je konfigurisati NAT da propušta određeni saobraćaj.
    - U opštem slučaju saobraćaj ka WS pošto se koristi HTTP se propušta.



Firewalls

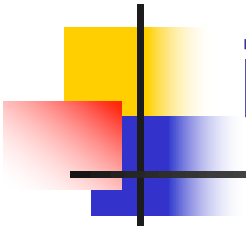
- Dozvoljavaju ili ne dozvoljavaju prolazak saobraćaja zavisno od izvora, destinacije, protokola….
  - Često se konfigurišu da samo dozvoljavaju ODLAZNE konekcije
- Mogu biti stateful: pamti aktivne tokove i odbacuje neočekivane pakete (NAT)
  - Opet moraju se konfigurisati da bi određeni saobraćaj prošao kroz njih (generalni problem RPC)
  - (WS)HTTP se ne suočava sa ovim problemom.

Demilitarized Zone (DMZ)

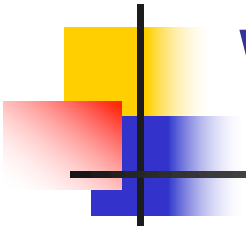


- DMZ: često se formira da se u nju smeste javno dostupni servisi webpages, ftp, dns.
- Dobro mesto i za Web Service!
- DMZ se smešta van privatne mreže.
- Ako je napadnut DMZ, šteta je limitirana na tu zonu.

Da li Web servisi “pomažu” pri imenovanju i otkrivanju servisa?



- Web Servisi nam govore kako klijent može:
  - … pronaći server i…
  - … povezati se na njega…
  - … poslati zahtevi koji ima smisla…
  - … i kako razumeti odgovor
- Dakle pomažu



Web servisi neće …

- Omogućiti da data center (aplikacija) ima uticaj na odluke klijenta
- Rešiti probleme u skalabilnim klaster okruženjima
  - Kako uraditi balansiranje opterećenja?
  - Šta se dešava ako server koji izvršava servise padne?
  - Kako upravljati dinamičkim konceptima? Definisanje najboljeg servera za opsluživanje određenog klijenta može biti funkcija opterećenja, afiniteta, nedavnih taskova…
- Rešavanje ovakvih stvari nije u domenu WS