Quina tecnologia usar en la implementació del servidor personal?

Rubén Barrio, Sergi Sánchez i Jordi Torres

Càtedra Telefónica-UPC

http://www.upc.es/web/CatedraTelefonicaUPC/

Working Report WR-2003-01, juliol 2003

Resum. Aquest Working Report conté una descripció de la tecnologia escollida per realitzar un conjunt aplicacions per oferir continguts a traves d'Internet. En ell es descriuen els principals elements que conformen una de les tecnologies amb més projecció de futur: els Web Services. En el document es pot trobar tan una petita descripció del concepte "Web Service" com una detallada explicació dels principals components d'aquesta tecnologia i de les plataformes que la ofereixen. Els "Web Services" han estat escollits com a tecnologia referent tant per la seva versatilitat com per la seva prometedora rellevància en les comunicacions en un futur pròxim.

1. Introducció

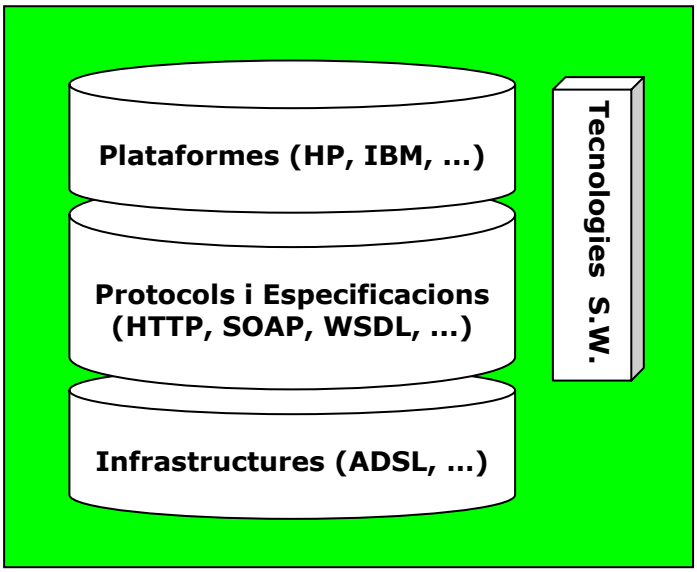
L'etiqueta de "web service" (servei web) és increïblement genèrica. Com qualsevol altra tendència tecnològica prometedora i definida genèricament, els components que la descriuen estaran subjectes a la relació entre l'especulació i el fet de "no perdre el tren" per part del mercat durant els mesos que han d'arribar. (...) El marketing de la indústria de les noves tecnologies ha dirigit moltes vegades el cicle de vida dels elements de la pròpia indústria, però rarament s'ha vist emergir un concepte tan ràpidament com ho ha fet el concepte dels "web services". [SlRo01]

L'etiqueta de "serveis web", parlant en línies generals, té dos nivells de significat, un específic i un conceptual.

- Específicament, els serveis web són una pila d'estàndards emergents (més o menys recents) que descriuen una arquitectura d'aplicació basada en components, orientada als serveis.

- Conceptualment, els serveis web representen un model en el qual tasques concretes dins dels processos de *e-business* són distribuïdes àmpliament a través d'una xarxa com Internet.

Partint d'aquestes dues definicions, al llarg dels diferents capítols d'aquest apartat s'exposaran les idees que hi ha darrera de les dues. Així, per la vessant específica, es parlarà de les tecnologies, de les plataformes i dels protocols i especificacions els quals hi ha darrera dels serveis web. I també es parlarà d'infrastructures que permeten l'ús de tot l'esmentat anteriorment per part de petits consumidors d'aquestes tecnologies. I per la vessant conceptual, es parlarà del mercat que està emergint entorn els serveis web, i més de les seves perspectives de futur.



Els serveis web són aplicacions modulars autodescriptives que es poden publicar, ubicar i invocar des de qualsevol punt de la web o des de l'interior d'una xarxa local basada en estàndards oberts d'Internet. Els serveis web combinen les millors prestacions de la programació amb components i la programació web, i presenten forma de mòduls que poden tornar-se a utilitzar sense preocupar-se per la implementació o el llenguatge, sistema operatiu o model de components utilitzat en la seva generació. Això implica que ja no és necessari saber el que hi ha instal·lat o com funciona per a utilitzar la seva funcionalitat. L'accés als serveis web es realitzen a través de protocols d'Internet

omnipresents com HTTP o SMTP basats en XML. Els desenvolupadors dels serveis web XML poden implementar-los en qualsevol llenguatge o model de components sota qualsevol sistema operatiu.

La creació d'aplicacions amb serveis web de nivell superior permet canviar d'aplicacions estretament lligades a aplicacions menys lligades. En general, els components o aplicacions estretament lligades són aquelles que necessiten vincular-se a la fase de disseny, mentre que els components que no estiguin fortament lligats poden vincular-se a la fase d'execució. Els components i aplicacions que no estan estretament lligats solen ser més escalables, manejables, ampliables i menys susceptibles a errors causats per modificacions que les implementacions dels més estretament lligats.

No obstant, els components que no estan estretament lligats presenten algunes desavantatges per al programador de l'aplicació, ja que presenten un major número d'errors. Al mateix temps, les eines i la infrastructura necessària per a la implementació d'aquests components han resultat ser un repte en el passat degut a la falta d'estàndards necessaris. El propòsit fonamental de l'àrea dels serveis web consisteix en apropar als usuaris al procés de creació d'aplicacions que puguin lligar-se i descobrir-se de forma dinàmica, i al mateix temps, beneficiar-se dels avantatges que ofereix la millorada arquitectura d'aquesta aplicació.

El concepte de serveis web va començar a prendre una forma definitiva amb la introducció de SOAP com a protocol de missatgeria entre ordinadors. SOAP és un protocol de cable senzill basat en XML que va aparèixer el 1998. Es va dissenyar per a la connexió entre ordinadors independentment dels seus sistemes operatius, llenguatges de programació o models d'objectes utilitzats (i inclòs la mancança total de model d'objecte). A pesar de que el seu nom pugui semblar que requereix l'ús de determinants objectes, SOAP especifica el format del missatge que accedeix i invoca els objectes, en lloc d'especificar els objectes en si.

En maig de l'any 2000, el W3C (*World Wide Web Consortium*) va reconèixer la proposta de SOAP presentada de forma conjunta per Hewlett-Packard Co., IMB Corp., IONA Technologies PLC, Lotus Development Corp., Microsoft Corp., entre d'altres. El fet de que un grup de diferents empreses recolzés la presentació de la proposta de SOAP suposava un magnífic signe a la futura acceptació de la indústria i la implantació d'un protocol interoperable de base estàndard obert. A l'actualitat, SOAP se segueix desenvolupant sota el *XML Protocol Working Group* a W3C.

Com s'ha mencionat anteriorment, els serveis web es basen en XML i d'altres estàndards oberts d'Internet com HTTP i SMTP.

1.1 Servidors Personals

Actualment, les connexions de banda ampla per accedir a Internet estan tenint una gran acollida tant per part d'usuaris domèstics com per part de les empreses. Aquest tipus de connexió no només permeten un accés a major velocitat sinó que suposen un gran canvi ja que per un preu fix al més permet accedir a la Xarxa 24 hores al dia. Aquest canvi de proposta provoca que molts usuaris mantinguin els seus ordinadors connectats tot el dia tant per poder consumir continguts de gran volum com per poder oferir-los.

Molts d'aquests ordinadors connectats havien estat fins ara únicament clients o consumidors, però aquest canvi genera que poc a poc alguns d'aquest usuaris es converteixin alhora tant en clients com en servidors o proveïdors. Inicialment aquest canvi es produeix lentament ja que els requisits per convertir un ordinador en servidor inclouen un cert grau de coneixements tecnològics que no tothom posseeix.

S'entén per tant com a servidor personal un ordinador que pugui estar connectat a la xarxa constantment i que alhora ofereixi continguts tant públics com privats. Aquests servidors personals poden constar de moltes i molt diverses aplicacions que ofereixin cadascun dels serveis desitjats, i cada eina amb una tecnologia diferent.

Creiem que es un gran aposta de futur voler aplicar la tecnologia "Web Services" a aquests servidor personals ja que dissenyant un portal estàndard creat amb aplicacions d'aquest tipus podem oferir un gran ventall de possibilitats als usuaris que vulguin ser servidors però que no tinguin mitjans per aconseguir-ho. D'aquesta manera es podrà crear un "Espai Personal" o un lloc des d'on oferir continguts de manera ràpida i senzilla. I el producte resultant no serà només els continguts embolicats en una Web, sinó un conjunt d'aplicacions que suportaran tots aquest continguts, amb capacitat d'interactuar amb altres usuaris i accessibles des de qualsevol indret, tot amb un cost baix i una instal·lació trivial.

1.2 El mercat emergent dels Serveis Web

Els serveis web són un model emergent per a desenvolupar i distribuir/publicar aplicacions software. Aquesta aproximació por millorar la flexibilitat i l'abast de les infrastructures existents de les TI. Això també crea moltes oportunitats per a implementar nous models de negoci basats en l'oferta i el consum de software. L'etiqueta de servei web és genèrica, però els venedors de software i les empreses clients estan configurant un relatiu i consistent significat del concepte. Els serveis web tenen dos nivells de significat, un conceptual (i orientat als negocis) i un (tècnicament) específic. Conceptualment, els serveis web representen un model en el qual petites peces de funcionalitat d'una aplicació estan disponibles com a serveis per a ser consumits i combinats amb d'altres aplicacions sobre una xarxa. Específicament, els serveis web son una pila d'estàndards emergents que defineix protocols i crear un marc feblement acoblat per a la programació de la comunicació entre sistemes diferents i heterogenis. [SRSA02]

Per a entendre els serveis web dins d'un marc real i de negoci, es pot veure el següent diagrama, on a les capes més altes apareixen menys aspectes tècnics, i més de gestió i de negoci.

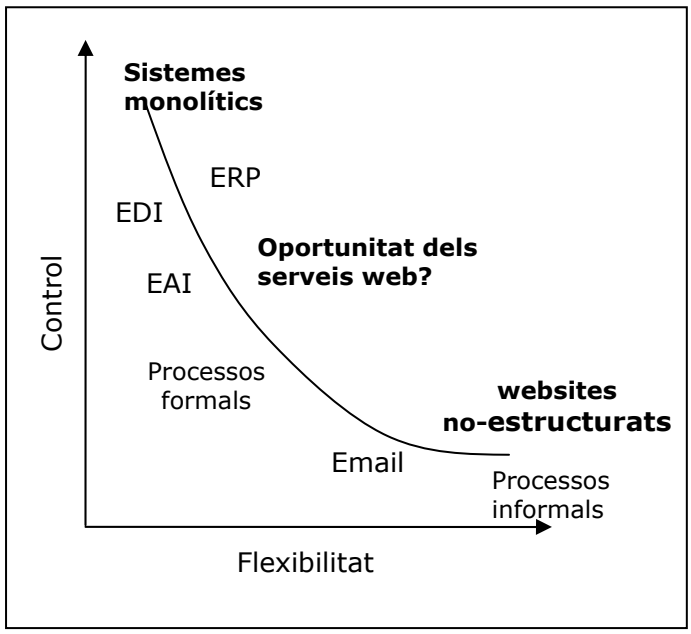
1.2.1 Una promesa de beneficis de negoci

En l'entorn econòmic incert d'avui dia, l'eficiència operacional és més important que mai. Els equips de gestió estan enfrontats a la temible tasca d'incrementar els valors dels negocis amb recursos reduïts. Inversions estratègiques en tecnologies de la informació són l'element clau d'aquest objectiu, però massa sovint, costosos desplegaments tecnològics ofereixen retorns limitats amb un risc substancial.

Els serveis web prometen canviar aquesta dinàmica. Permetent als negocis d'enfocar-se a les seves fites estratègiques, els serveis web influencien les inversions en TI d'una organització per tal de reduir riscos i lliurar resultats que es puguin predir de projectes de TI, per a millorar la visibilitat en operacions centrades en les dades i adoptar innovacions en els models de negoci.

Incrementar la flexibilitat: Els serveis web permeten a les companyies de prendre decisions estratègiques de negoci en raó al mèrit del negoci, més que en limitacions tecnològiques de les infrastructures existents. L'aproximació del venedor neutral representada pels estàndards oberts dels serveis web significa que les companyies poden triar les millors arquitectures tecnològiques i incrementar el valor de l'elecció del proveïdor. A més, les transaccions basades en serveis web redueixen costos significativament i permeten a les companyies d'establir o terminar associacions de negoci molt més àgilment i eficient.

A continuació es mostra un diagrama amb el balanç dels serveis web entre el control i la flexibilitat.



ERP: *Enterprise Resource Planning*
EDI: *Electronic Data Interchange*
EAI: *Enterprise Application Integration*

Reduir riscos: Tradicionalment, les TI han tingut un cost significatiu central per a molts negocis, i només algunes companyies s'han recolzat a la tecnologia com a avantatge competitiu. Els serveis web desplacen el model de cost de TI estratègiques des de despeses i desplegaments arriscats cap a un més controlable, inversions incrementals. Aquest desplaçament redueix riscos i allibera el capital disponible per a d'altres inversions estratègiques.

Augmentar la visibilitat: Possibilitant connexions fluides i en temps real entre sistemes operatius, els serveis web incrementen significativament la visió dels gestors sobre els processos de negoci centrats en les dades. També, els serveis web poden reduir dramàticament el cost operatiu per unitat de les companyies. No només les dades internes seran més accessibles, si no que els socis de negoci que comparteixin processos de negoci basats en serveis web també podran obtenir un grau més alt de coordinació i de compartició d'informació. Finalment, el procés de desenvolupament de software en ell mateix serà més transparent, permetent a les empreses de predir amb més cura quan seran operacionals els sistemes clau.

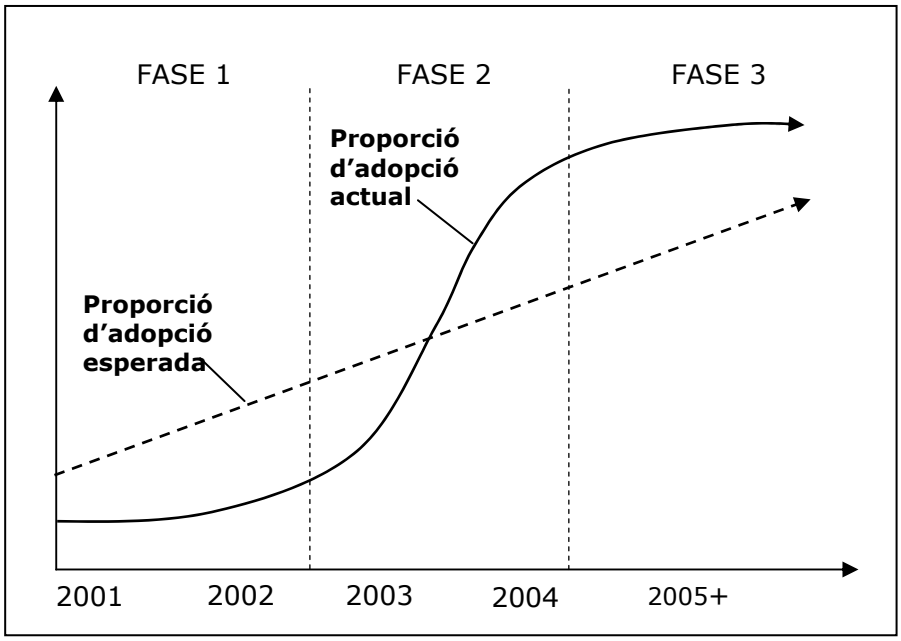
Oportunitats més grans d'innovació: Els directors de negocis lliures de models tecnològics inflexibles poden centrar-se en els processos de negoci, i no en les implementacions tecnològiques. Aquest canvi pot resultar en millores incrementals de les operacions de negoci, i les organitzacions poden experimentar amb una varietat de models de negoci basats en serveis que no haguessin estat viables sense una infrastructura oberta i cooperativa.

1.2.2 L'evolució del software empresarial

El software empresarial d'avui dia està caracteritzat per dues tendències significatives. Ambdues remarquen de prop associacions entre venedors i estratègies competitives, versions de productes i missatges de marketing. Primer està la pujada dels servidors d'aplicacions i la següent relaxació del mercat compartit entre les arquitectures de software centrades en Java i en tecnologies Microsoft. I segon, hi ha un reconeixement creixent que la majoria dels segments del software existents avui en dia estan convergint cap a un mateix punt: la integració i coordinació de sistemes tecnològics diferents en valuosos processos de negoci.

1.2.3 Preparats per a créixer

Mentre que el creixement i les esperances es perceben com a lineals, l'adopció actual es mourà per davant fins a que s'assoleixi un punt d'inflexió, a partir del qual creixerà exponencialment. L'adopció dels serveis web seguirà una progressió directa des de les característiques dels models dels primers en adoptar-los fins al moment en que es mostrin àmpliament acceptats per a usos majoritaris.
El següent diagrama mostra el creixement real del mercat dels serveis web davant de l'esperat:



Fase 1: Adopció de les bases

La primera etapa, ja pràcticament assolida del tot, emfatitza els projectes interns d'integració que estan dirigits pel cost eficient i l'habilitat d'utilitzar bens existents. Les companyies van començar a agafar els avantatges de que les tecnologies dels serveis web trenquessin les aplicacions monolítiques en components més petits. Aquells que adopten els serveis web els fan servir per a publicar projectes addicionalment a projectes interns, molts dels quals eren considerats prèviament massa costosos. Les connexions en aquest període es produeixen quan els coneguts extrems finals estan clars i els serveis són simples. La recerca en aquest sentit confirma que els serveis web estan essent recolzats per qui usualment acostuma a adoptar les noves tecnologies: firmes de serveis financers, fabricants d'alta tecnologia i companyies de telecomunicacions, i també per d'altres indústries una mica menys obvies com la de l'automòbil i les assegurances.

Serveis web públics ja estan disponibles a Internet: aquests serveis gratuïts tenen funcions simples com a calculadores, i no inclouen components de seguretat.

Fase 2: Publicació sistemàtica

Les companyies ja han començat a dirigir-se cap a estratègies sistemàtiques de desenvolupament, publicació i ús de serveis web. Aquest període de creixement estarà caracteritzat per una empenta cap a un increment del control i xarxes orientades a les transaccions. Encara que existeixin algunes oportunitats d'establir nous ingressos, les corporacions tractaran de racionalitzar els processos existents. Els estàndards i components dels serveis web referents a la indústria arribaran a ser dominants.

Fase 3: Nous models de negoci

Els més optimistes preveuen nous models de negoci possibilitats pels serveis web i definits per relacions de negoci veritablement dinàmiques creades i dissoltes en temps real. Les empreses utilitzaran els serveis web per a crear noves formes d'oferir aplicacions. Les primeres empreses en adoptar-los podran ser les més receptives a l'hora d'experimentar amb aquesta visió, més que els sectors més tradicionals.

1.2.4 Adaptant-se a un nou entorn

L'aparició dels serveis web tindrà un impacte significatiu no només en les empreses i companyies de software, també en els proveïdors de serveis de TI. Aquestes companyies necessitaran deixar de banda els seus models tradicionals d'integrar diferents fonts de dades i aplicacions, i dirigir-se cap a serveis basats més en la consistència dels processos i plataformes de més alt nivell orientades als negocis. Les companyies que facin aquest canvi seran les millors posicionades per a proveir els serveis més crítics a les empreses que estan adoptant els serveis web.

1.2.5 Factors crítics per a l'èxit

Inclús els partidaris més aferrissats admeten que els serveis web no resoldran tots els problemes. Tampoc esperen que el software actual sigui descartat cegament a favor del nou enfocament. De fet, alguns factors poden jugar un paper important per a afavorir (o obstaculitzar) el nivell d'èxit dels serveis web.

- Conformitat amb els estàndards emergents: sense una interoperabilitat assegurada amb socis o clients, molts dels beneficis econòmics i tecnològics que ofereixen els serveis web es perdran. Tot i que proveir-se tant de venedors com de clients ha impulsat els serveis web endavant, quan els estàndards actuals, una mica, estan encara emergint. Mentre que no és necessària una acceptació total de tots els estàndards, l'adopció dels bàsics ja ben definits si que és crítica. Les capes més altes que defineixen aspectes estratègics dels processos de negoci han de ser desenvolupades per a ser utilitzades en la mateixa escala que les altres. Per exemple, la proliferació d'entorns gràfics de creació, ús i connexió de components ajudarà a afavorir un major nombre de serveis web, i ajudarà a promoure l'adopció dels serveis web com una API estàndard.

- Optimització de les dades: XML és un format de dades llarg, i la riquesa de la seva semàntica és transforma en una gran sobrecàrrega a l'hora de la transmissió. Alguns estimadors mesuren el tamany d'un document XML típic com a 10 vegades de la seva transmissió EDI (*Electronic Data Interchange*) equivalent. Els analitzadors XML d'avui dia, tot i ser correctes, encara han d'incorporar

optimitzacions significatives abans de que la sobrecàrrega es converteixi en un defecte degut a la latència de processat. Algunes activitats encara requereixen d'un processament considerablement ràpid, com els motors financers, i no seran probablement candidats a aquesta nova aproximació de computació distribuïda.

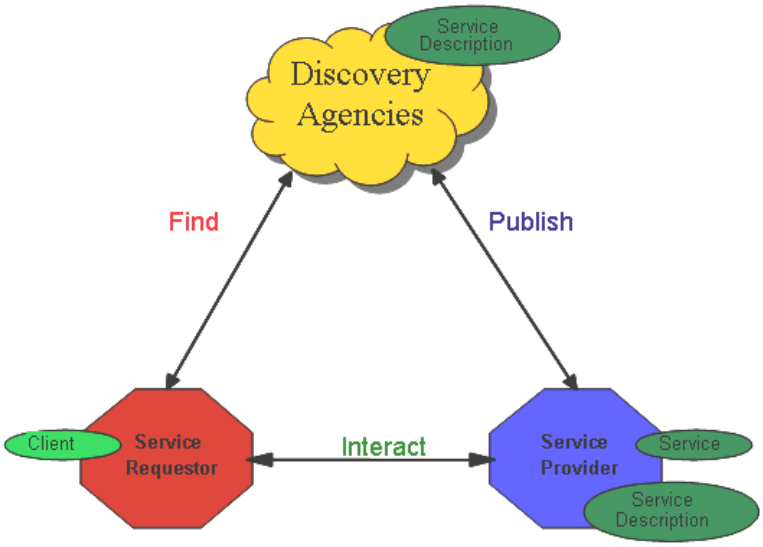
- Resolució dels temes de seguretat i autenticació: Encara no hi ha un estàndard de seguretat integrat als serveis web; les companyies actualment deleguen aquesta responsabilitat als *firewalls* i sistemes de seguretat de xarxa i d'aplicacions. Per tant, els socis han de crear un diàleg entre els socis connectats per a determinar el tipus i nivell de seguretat, la qual cosa limita aleshores l'escalabilitat del servei. Això no crearà una barrera funcional per als serveis web anònims o als estrictament interns, però les implementacions de processos de negocis estratègics a gran escala dins del context dels serveis web, no pot produir-se completament fins a que s'adopti un estàndard.

- Experimentació amb els models de negoci basats en serveis web: Els nous models de negoci estaran dirigits per la col·laboració entre empreses. Les companyies han de promoure l'experimentació per tal de trobar nous socis comercials, i per a reunir tots els processos de negoci de temps real. A més, quasi tots els serveis web actuals són gratuïts. Els models adequats de preus encara s'han de confirmar. No obstant, inclús si els serveis web no fan res per a fer més fàcil la seva integració amb aplicacions tradicionals, hauran fet una contribució significativa a les TI.

- Una volum crític de publicacions: Els beneficis dels serveis web són produïts en part per l'eficiència de la reutilització de components software. Sense un volum crític de serveis disponibles per al consum, les empreses necessitaran continuar recreant aplicacions de zero, i el model deixarà de ser tan efectiu com podria ser. Veient una oportunitat per a ajudar a solucionar aquest problema, des de principis del 2002, molts dels principals venedors van començar a oferir eines per a reduir esforços en aquest sentit. Aquestes eines fàcils d'utilitzar han d'actuar com a catalitzadors del mercat i incrementar el nombre de publicacions i ofertes de serveis web.

1.3 L'arquitectura dels serveis web

Una arquitectura orientada als serveis tindria la següent estructura en relació als conceptes anteriors: [CFNO02]

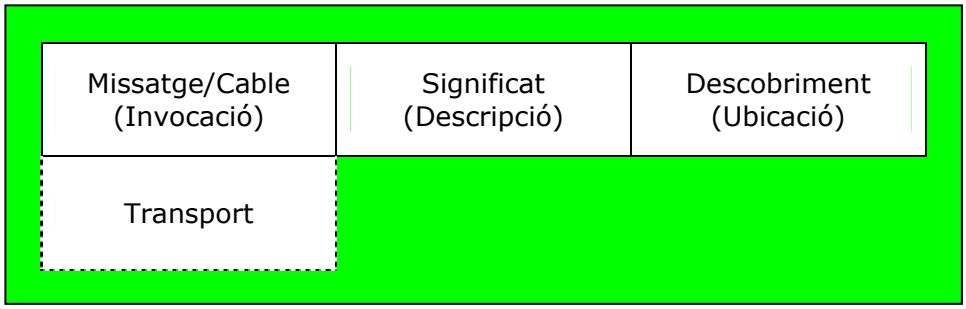


On els agents que intervenen són:

- **Sol·licitant del servei** (*Service Requestor*): sol·licita l'execució d'un servei web.
- **Proveïdor del servei** (*Service Provider*): processa una petició a un servei web.
- **Agències de descobriment** (*Discovery Agencies*): agències a través de les quals les descripcions dels serveis web són publicades i fetes accessibles.

Els principals blocs de construcció dels serveis web presenten tres aspectes que van en paral·lel als conceptes d'**ubicació**, **descripció** i **invocació**.

Al següent diagrama s'il·lustra la manera com encaixen entre si els tres conceptes. Cadascú d'aquests blocs consta d'una sèrie de capes. A continuació es parlarà dels blocs com a entitats.



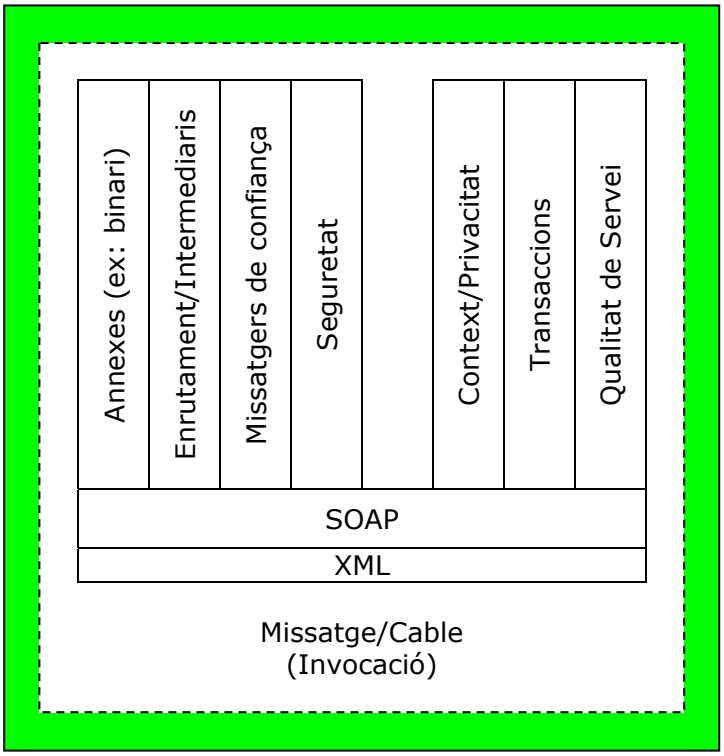
Per a que l'aplicació de l'usuari pugui utilitzar un servei web XML primer es necessari ubicar-lo o descobrir-lo. A continuació, l'usuari ha de familiaritzar-se amb les possibilitats que ofereix el servei web XML mitjançant la seva descripció. Una altra forma de pensar els blocs de construcció es com la representació del significat de servei web XML o, de

manera alternativa, com la metadada o la informació descriptiva de servei web XML. Finalment, l'usuari ha de poder cridar al servei web XML proporcionant-li els elements d'entrada necessaris i rebent la sortida apropiada (es a dir, cridar-lo). Aquest bloc conté el protocol SOAP i les seves diferents ampliacions. El bloc de construcció del missatge se situa sobre la capa de transport que, al seu torn, consta de protocols de transport: protocols d'Internet oberts i de base estàndard, com HTTP i SMTP.

1.3.1 Invocació

El bloc de crida dels serveis web se centra en el missatge o nivell de cable de l'arquitectura. En aquest nivell, **SOAP** és el component clau. SOAP és un protocol de format de missatge extensible que es vincula a diversos protocols estàndards de transport d'Internet, com HTTP i SMTP.

A continuació es mostra la pila d'invocació dels serveis web basada en XML i SOAP.

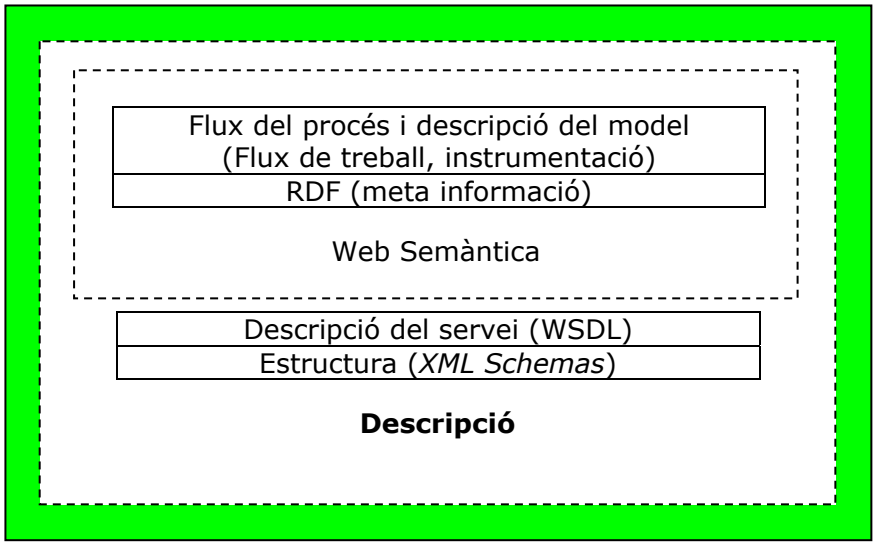


Aquest bloc es detallat a la secció de Protocols i Especificacions, quan es parla de SOAP.

1.3.2 Descripció

La descripció d'un servei web XML, és a dir, la descripció dels missatges que poden acceptar i generar (bàsicament, part del contracte de serveis web XML) es realitza principalment a través de WSDL, ja que compta amb d'altres especificacions XML com part de la seva base, com els esquemes XML, i els espais de noms XML.

L'arquitectura de la pila de descripció de serveis web es mostra a continuació.



La pila de descripció comença amb la descripció de l'estructura utilitzant **esquemes XML** (*XML Schemas*). Els esquemes XML proporcionen una forma d'especificar tipus de dades començant pels senzills, com *integer* i *string*, i arribant a dades de tipus més complexos com les matrius o tipus abstractes de dades definits pels usuaris.

El següent nivell de la pila de descripció és la descripció del servei, normalment realitzada mitjançant l'ús de **WSDL** (*Web Services Definition Language*). WSDL és un format XML que descriu serveis de xarxa considerant-los com un conjunt de punts finals que operen sobre missatges que continguin informació de tipus document o de tipus procediment. Inicialment, les operacions i els procediments es descriuen de manera abstracta i s'uneixen a un protocol de xarxa específic (o protocol de transport) i a un format de missatge. WSDL és extensible i permet la descripció de serveis de xarxa, independentment de la xarxa i protocol de missatgeria d'aquests. L'especificació actual només descriu enllaços a SOAP, HTTP, GET/POST i MIME i no inclou un marc complet per a la descripció de la composició i l'organització dels serveis de xarxa, que forma part de les capes de nivell superior de la pila de descripció.

WSDL es va formar per Ariba, IBM i Microsoft per a representar el concepte actual de la definició de serveis web. Al març del 2001, WSDL va rebre la categoria de '*Note*' al W3C com a WSDL 1.1. Actualment, des del juliol del 2001, s'està treballant en el '*Working Draft*' del que serà el WSDL 1.2. Les implementacions més actuals utilitzen WSDL per a la definició dels serveis proporcionats per SOAP.

Les capes de la "web semàntica" encara es troben a una fase primera de desenvolupament. L'objectiu d'aquestes capes és el de descriure, de forma comprensible per al software, les diverses característiques d'un servei web. Això inclou característiques com la fiabilitat i les aptituds, així com la seqüència dels missatges i una descripció de l'emissor i de l'hora en la que es realitza. El conjunt exacte de característiques descriptives continua encara en desenvolupament. El marc de recursos de descripció RDF

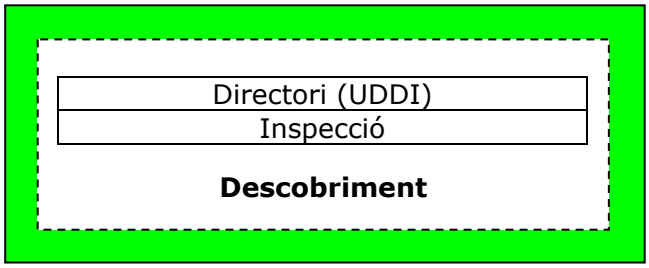
(*Resource Description Framework*) està considerat per diferents experts del camp com la base d'aquest esforç, ja que pot utilitzar-se per a la definició ontològica o per a donar significat als termes. Tanmateix, el curs del procés i les tècniques de pautes de descripció es poden utilitzar per a la descripció dels processos dins d'una empresa o d'una aplicació. Alguns llenguatges del procés de descripció poden arribar a desenvolupar-se per a satisfer aquest sector: les tecnologies del curs del procés permetrien al desenvolupador d'un codi connectar la seva aplicació a un servei web XML, utilitzar l'arxiu de descripció del curs del procés i determinar la seqüència de missatges al lloc del servei web.

D'aquest bloc se'n parla més a la secció de Protocols i Especificacions en parlar de WSDL.

1.3.3 Ubicació

Una vegada determinada la forma d'invocar un servei web XML i de que es pugui descriure, es necessita una manera per a permetre que d'altres el puguin ubicar. Semblant als motors de cerca, l'especificació **UDDI** (*Universal Description Discovery Integration*) descriu com un proveïdor pot publicar l'existència del seu servei web XML en un directori. Un usuari d'un servei web XML pot buscar al registre UDDI per a informar-se sobre el servei web que necessita, per categories: UDDI ofereix la possibilitat d'autodescripció de les empreses a una o més taxonomies, com el Sistema de Classificació de la Indústria Nord-americana NAICS (*North American Industry Classification System*) i la Classificació Universal Estàndard de Productes i Serveis UNSPSC (*UNiversal Standard Products and Service Classification*). Les taxonomies se semblen als vocabularis, en els quals s'assigna un significat a cada terme; no obstant, aquests termes es poden categoritzar en estructures jeràrquiques.

De manera que, una vegada determinat el servei web XML necessari, el següent pas consisteix en inspeccionar els seus detalls, la seva descripció de servei i el contracte de funcionament del curs del procés.



I finalment, d'aquest bloc també se'n parla més a la secció Protocols i Especificacions a l'hora de parlar de UDDI.

2. Plataformes

A la secció de Tecnologies s'han repassat els tres blocs de construcció principals dels serveis web: el missatge (o cable), la descripció i la ubicació. Aquests blocs estan relacionats i estan fets per a treballar junts dins d'un **Marc de serveis web**. En la següent secció, s'estudia el marc de serveis web en conjunt i la seva arquitectura en capes i mòduls, la manera com encaixen i la posició en la que es troben les diferents tecnologies.

Els blocs de construcció dins de les tecnologies dels serveis web que existeixen en l'actualitat són XML i els esquemes XML. Els components que es troben en un procés de desenvolupament continu són SOAP, UDDI i WSDL. SOAP constitueix els fonaments del marc del servei web. El marc permet la definició, l'ús, la manipulació i el desenvolupament de serveis descentralitzats de manera automatitzada, sense un control central. Aquest marc és la base d'aplicacions orientades als serveis i, amb el temps, d'Internet orientat als serveis. El marc admet una arquitectura en capes i escalable que satisfaci les necessitats de les implantacions senzilles i complexes.

El marc dels serveis web no restringeix excessivament les tecnologies que poden utilitzar-se per a la implementació dels serveis web, ni limita als desenvolupadors als productes d'un determinant proveïdor. Això significa que el marc dels serveis web es pot implementar (i no necessàriament d'una vegada) amb diverses tecnologies, com Java al sistema operatiu Linux, o Visual Basic i Visual C++ als sistemes operatius de Windows.

Des del punt de vista d'alguns autors de la plataforma de serveis web al W3C, un marc comú ajuda a assegurar que les implantacions siguin menys complexes, disminueix la possibilitat de que diferents empreses creuin marcs i augmenta la probabilitat d'assolir la interoperabilitat.

Un altre punt molt important es que el desenvolupament i l'adopció de components individuals del marc poden succeir de forma paral·lela i asíncrona, permetent la creació d'implementacions sense necessitat de tenir primer que estudiar tots els components del marc. Tanmateix, és possible afegir components nous i mòduls al llarg del temps, així com una característica crítica que permeti que el sistema es desenvolupi sense haver de canviar-lo per complet. Els components es desenvolupen de forma escalada amb increment, en direcció ascendent. Això dóna peu a que cada component funcioni de manera independent per si mateix. Al mateix temps, cada component serveis com a fonament per a la implementació de components en capes superiors.

El marc dels serveis web permet la utilització de subgrups de components per a assegurar uns nivells de servei específics requerits en certes circumstàncies. Per exemple, si una empresa vol desenvolupar una senzilla comunicació d'aplicació a aplicació per a la seva xarxa interna, pot utilitzar SOAP sense necessitat de complicar-se amb la majoria dels components de la plataforma de serveis web. Això és un avantatge per a la simplificació d'una implementació i la seva adaptació a usuaris amb diverses necessitats i situacions, des de les empreses més petites fins algunes de les grans multinacionals. La creació d'estàndards monolítics que tracten dominis d'ús complets porta a la creació d'estàndards diferents. Al crear una col·lecció d'estàndards de blocs de construcció que es puguin combinar en estàndards de domini si és necessari, se simplifica la implementació, la interoperabilitat i la verificació de conformitat . Com a conseqüència, això fa que el desenvolupament dels estàndards i la infrastructura dels serveis web sigui pràctica i rentable.

S'ha mencionat que el marc dels serveis web no especifica la implementació dels diversos mòduls especificats. Dit d'una altra manera, el marc dels serveis web permet que s'implementin en qualsevol llenguatge de programació i que utilitzin qualsevol sistema operatiu, base de dades o tecnologia *middleware*. Aquesta interoperabilitat es basa en formats de dades i protocols estàndards, no en les Interfícies de Programació d'Aplicacions o API. Això proporciona una flexibilitat i autonomia als proveïdors de software que permet la implementació del marc dels serveis web de la manera que coneguin més bé.

A continuació, es repassen breument les diverses iniciatives d'implementació que s'estan duent a terme pels proveïdors, és a dir: els diversos conjunts d'eines i marcs que proporcionen. [CCCD02]

2.1 Eines i marcs dels serveis web dels proveïdors

Encara que SOAP i alguns dels blocs de construcció relacionats amb els serveis web són relativament nous, existeixen implementacions en el mercat. La majoria dels principals proveïdors tenen la seva pròpia implementació de SOAP dins del seu entorn o llenguatge de programació. Aquestes implementacions es troben principalment com conjunts d'eines.

Dins de la comunitat Java, Apache SOAP (originàriament donat per IBM i la seva divisió de alphaWorks) i el conjunt d'eines dels serveis web d'IBM són les implementacions clau. Microsoft també té un conjunt d'eines de SOAP; no obstant, la implementació dels serveis web per part de Microsoft s'està duent a terme dins del seu marc .NET.

Tanmateix, altres grans proveïdors ofereixen conjunts d'eines i marcs de serveis web, com per exemple Sun Microsystems amb el seu *Web Services Development Pack*, i HP amb el seu *Core Services Framework*. Existeixen altres molts proveïdors que ofereixen diversos conjunts d'eines de serveis web i implementacions SOAP per a diferents plataformes de llenguatge i sistemes operatius (com conjunt d'eines per a PERL, Python, C++, PHP, etc.).

Mentre que les eines dels serveis web se centren en proporcionar la capacitat bàsica per al desenvolupament i l'ús de serveis web, alguns proveïdors compten amb ofertes més àmplies i de més gran abast. Aquestes ofertes es coneixen com a marcs o, a vegades, com a plataformes. Apart dels conjunts d'eines de serveis web bàsics, els marcs inclouen especificacions i guies d'arquitectura, així com eines i funcionalitats addicionals per a l'ús de serveis web. A continuació s'estudien alguns dels marcs i plataformes més coneguts que ofereixen alguns dels grans proveïdors.

2.2 HP i e-Speak (evolució a HP Web Services)

Hewlett Packard ha estat pionera a l'àrea dels serveis web, inclòs abans de que el terme fos popular. La seva primera oferta de producte va ser *e-Speak*, una plataforma oberta per al desenvolupament i l'ús de serveis electrònics. Un serveis electrònic es correspon estretament amb un serveis web, tal i com es coneix. Un servei electrònic es aquell que es troba disponible a Internet i desenvolupa una tasca o transacció en concret. Es pot ubicar dinàmicament, invocar i crear mitjançant altres serveis electrònics. Els inicis de *e-Speak* a HP van ser en forma de projecte d'investigació l'any 1995, i el 1998 es va iniciar un projecte per a transformar-lo en un producte. El primer producte *e-Speak* es va llançar al mercat el maig del 1999.

No obstant, des de llavors, HP s'ha desviat cap a estàndards de base XML per a serveis web com SOAP i UDDI, que compten amb una més gran acceptació per part de la indústria; el seu producte més recent en aquest sentit, el *HP Web Services Platform* s'ha construït en funció d'aquests estàndards. El paper que té HP als estàndards de serveis web continua augmentant i des de principis del 2002 HP és un node d'operació per al Registre Empresarial UDDI.

Java ha desenvolupat *e-Speak*, pel que els seus clients necessiten la seva API Java (coneguda com a Interfície E-Speak de Java, J-ESI) per a la creació de serveis electrònics i clients. Tanmateix, compta amb mecanismes de base XML par a la interacció amb serveis *e-Speak*.

e-Speak compta amb un model de seguretat fort, basat en estàndards d'infrastructura senzilla de clau pública SPKI. No obstant, independentment dels motius, no ha rebut la mateixa resposta que les noves tecnologies proposades per Microsoft i IBM, entre d'altres.

e-Speak defineix tres components tècnics que són: el motor *e-Speak Engine* (també conegut com el *e-Speak Core*), l'especificació de marc del servei *Service Framework Specification* (SFS) i el *e-Service Village*. El motor *e-Speak* consisteix en el propi motor i en l'API de Java per a la seva interacció. SFS defineix un protocol per a l'intercanvi de documents XML entre serveis. *e-Speak Village* és un directori per als serveis web.

2.3 El comerç electrònic dinàmic d'IBM

L'oferta de productes del servei web d'IBM es recull sota el concepte d'un "comerç electrònic dinàmic". Aquest concepte fa referència a l'ús de tecnologies d'Internet per a aconseguir que un negoci sigui més flexible i capaç d'adaptar-se als canvis. El concepte de comerç electrònic dinàmic reconeix que, sovint, l'evolució per a suplir necessitats no anticipades al moment del seu disseny domina els costos de la generació d'infrastructures i processos de gestió i no la seva creació inicial en si. Aquesta evolució pot dictar-se per les necessitats canviants dels usuaris o socis empresarials i, per regla general, comporta la integració d'un sistema a un nou entorn o la reorganització dels components de la lògica empresarial en combinacions novedoses.

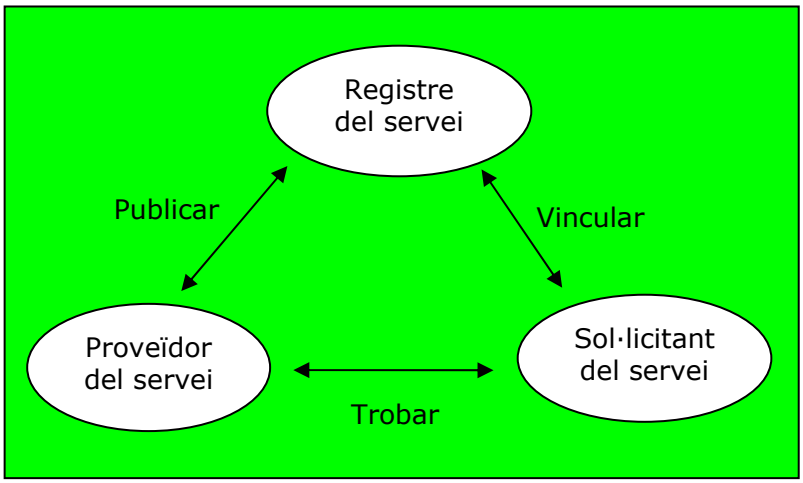
Tradicionalment, assolir aquest tipus de flexibilitat a través de dissenys de sistema de components intel·ligents suposava l'ús d'un marc sofisticat per a proporcionar la necessària gestió de components que permet una reorganització dels components menys costosa. El problema és que aquest tipus de sistemes, per exemple CORBA, solen imposar gran quantitat d'estructura relativament alta sobre els mateixos components i es termina substituint un tipus d'inflexibilitat (el disseny de sistema monolític) per un altre (d'estreta relació amb el marc). Un dels resultats predibles d'aquesta situació és que la interoperabilitat de components creats amb marcs de comerciants diferents pot resultar difícil d'aconseguir.

La tècnica del comerç electrònic dinàmic utilitza un emparellament menys restrictiu, basat en estàndards d'Internet com XML, SOAP, WSDL i UDDI, per a aconseguir el disseny del sistema desitjat sense la dominant influència d'un marc complex. Un sistema creat amb aquests estàndards es pot dissenyar com un conjunt de components, per exemple serveis web, que poden modificar-se i reorganitzar-se per a reflectir estratègies empresarials canviants i utilitzen interfícies estàndards que permeten relacionar-se amb altres sistemes amb el mínim de costos en integració.

2.3.1 Productes

El propòsit d'IBM és servir com un proveïdor d'eines que puguin utilitzar-se per a la construcció del sistema mencionat anteriorment. No obstant, el primer pas clau és l'adopció generalitzada dels estàndards sobre els que es vagin a crear serveis web. En aquest sentit, IBM s'ha mantingut actiu a la comunitat d'estàndards. IBM va donar el seu conjunt d'eines SOAP a Apache i s'ha mantingut actiu als esforços per l'estandardització de WSDL i UDDI. Es més, d'acord amb fonts d'IBM, la infrastructura dels serveis web d'IBM es caracteritza per ser més obertes que altres ofertes de la competència. Comparant-la amb Microsoft, que ha mostrar gran interès en la generació dels seus models de serveis web de XML amb el seu propi sistema operatiu de marca registrada (és a dir, Windows), IBM ha inclòs compatibilitat amb múltiples sistemes operatius mitjançant l'ús de Java com el seu principal llenguatge de programació per a implementar serveis web. La versió de Microsoft és que ells han inclòs un entorn multi-llenguatge (per mitjà de .NET), pel que mostren ser oberts en la seva tria de llenguatges, cosa que els comerciants que s'estandarditzen en Java no fan.

Els productes de serveis web d'IBM es basen en el següent model:



En aquest model, el Proveïdor del servei implanta la lògica empresarial que s'enviarà i anuncia la seva existència publicant informació amb el Registre del servei.

El Sol·licitant del servei desitjaria utilitzar els serveis del Proveïdor del servei, pel que consulta el Registre del servei per a trobar un proveïdor adequat.

Una vegada equipat amb la informació del Registre del servei, el sol·licitant del servei pot trobar al proveïdor del servei i aprendre sobre la seva interfície. Aquesta informació és suficient per a que el sol·licitant contacti amb el Proveïdor del servei i així pugui sol·licitar els serveis que ofereix.

Aquest model és una excel·lent representació d'un servei web, i explota la característica més valuosa d'aquests: els components estan relacionats de forma poc rígida, simplificant la inevitable adaptació del sistema als nous requeriments, cosa que resulta tan comuna al món empresarial d'avui en dia.

Al utilitzar les eines d'IBM, el desenvolupador del sistema usaria tecnologies estandarditzades. Per exemple, la comunicació entre les diferents parts d'aquest model tindria lloc mitjançant missatges SOAP, el format dels quals segueix el protocol XML i es transmeten per HTTP. El Proveïdor del serveis usaria l'estàndard WSDL (en format XML) per a descriure el servei. Les operacions de cerca i publicació es produirien al usar

l'estàndard UDDI, que utilitza missatges SOAP, per a la comunicació amb el Registre del servei.

IBM proporciona eines per a la creació del sistema d'acord amb aquest model. En lloc d'ubicar-se a un sol producte, l'estratègia d'IBM és la d'incloure aquestes eines a un repertori de productes, incloent DB2 (gestió de BBDD), Lotus Notes (*groupware*), Tivoli (gestió de BBDD), i de forma més notable, WebSphere (infrastructura d'Internet). Per exemple, DB2 inclou una varietat d'eines per a la conversió contínua dels continguts de la base de dades entre XML, i el servidor Domino dins de Lotus pot exportar serveis com a serveis web.

No obstant, WebSphere és el producte més important connectat als serveis web d'IBM. WebSphere és en si un conjunt d'eines, totes elles relacionades pel fet de considerar-les, encara que de manera distant, útils en la creació d'una aplicació de base web. Entre els seus components hi ha un servidor web, un servidor directori, eines d'autorització de pàgines web, serveis de seguretat, gestió de sessió i una àmplia varietat d'eines per a la creació d'un lloc de comerç electrònic. El component més important de WebSphere en relació amb els serveis web és el servidor d'aplicacions WebSphere (*WebSphere Application Server*). Aquest servidor d'aplicacions inclou eines com un marc J2EE per al desenvolupament i la implementació de *Enterprise Java Beans*, controladors JDBC per a la connexió a la base de dades, un contenidor *servlet*, un kit de desenvolupament amb Java, serveis de transacció i un grup d'eines enfocades especialment al desenvolupament d'una aplicació basada en el model de serveis web descrit amb anterioritat.

Aquestes eines específiques per als serveis web inclouen llibreries Java per a la producció i l'anàlisi de XML, la transmissió de missatges SOAP, la creació d'arxius WSDL i per a establir comunicació amb un registre del servidor mitjançant UDDI.

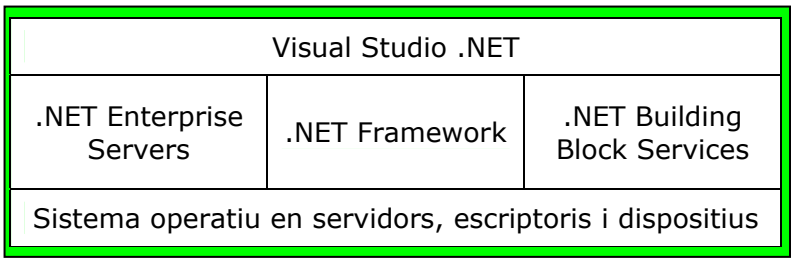
El conjunt d'eines de SOAP que inclou IBM al servidor d'aplicacions WebSphere és el mateix que ofereix Apache de forma gratuïta. No obstant, com succeeix en el cas de JDK que s'envia amb WebSphere, la versió se selecciona amb cura per a assegurar la compatibilitat amb la resta de WebSphere. Per aquesta raó, resulta més convenient la utilització del JDK i SOAP que s'inclouen a WebSphere quan es vagin a fer servir altres eines de WebSphere.

WebSphere inclou una eina anomenada *wsdlgen* que s'utilitza per a generar arxiu WSDL per a descriure un servei. La informació d'un arxiu WSDL inclou descripcions operatives de les funcions que un servei és capaç de realitzar, els protocols que es poden utilitzar per a comunicar-se amb ell i els extrems a Internet on es pot contactar. Un arxiu WSDL és llegible, però al tractar-se de XML, perd aquesta condició. Per tant, l'eina *wsdlgen* és útil ja que actua com un assistent que facilita la creació de l'arxiu WSDL.

Tanmateix, a WebSphere s'inclou la llibreria Java *uddi4j* que s'utilitza per a la comunicació amb un Registre de servei UDDI. UDDI és un sistema estàndard per a la publicació i la ubicació de serveis, semblant a un directori telefònic. Al igual que un directori, UDDI es divideix en pàgines blanques, que contenen informació sobre com contactar amb organitzacions; pàgines grogues, que categoritzen els serveis, i pàgines verds, amb informació sobre com realitzar negocis amb Proveïdors de serveis. Tot i que es pot utilitzar un registre UDDI privat, existeix un registre UDDI, únic i global, que pot utilitzar-se contactant amb qualsevol dels "lloc d'operació" de UDDI. Aquests llocs són serveis web, i utilitzen SOAP (i per tant XML) per a comunicar-se. No obstant, UDDI és un protocol sofisticat, pel que IBM ha implementat *uddi4j*, que facilita la tasca d'escriure una aplicació i es comunica amb un lloc d'operació per a publicar i buscar serveis. Tot i que no es requereix per a UDDI, WSDL és un complement natural d'aquest al actuar com el protocol amb el qual es pot descriure un servei i *uddi4j* inclou eines que ajuden en l'ús de WSDL per a la descripció del servei que vagi a registrar-se.

2.4 La plataforma .NET i .NET Framework de Microsoft

L'estratègia de desenvolupament i implementació de serveis web de Microsoft, especialment a llarg termini, és el marc .NET Framework. La iniciativa .NET es va anunciar al juny del 2000. Aquesta se centra al voltant de la plataforma .NET, que consisteix en bloc de construcció amb .NET Framework com a base. A continuació es fa un breu repàs de la plataforma .NET i els seus blocs de construcció.



El nivell de sistemes operatius està format pels sistemes operatius de Microsoft per a servidors com el Windows 2000 Server, sistemes operatius per a ordinadors com Windows XP, Windows 2000 Professional o Windows Me i sistemes operatius per a dispositius com Windows CE.

Al següent nivell es troben els servidors .Net Enterprise Servers com BizTalk 2000 Server (EAI o *Enterprise Application Integration* i B2B Server), Commerce Server 2000 (servidor de solucions de comerç electrònic), SQL Server 2000 (servidor de bases de dades), Exchange Server 2000 (servidor de correu electrònic i col·laboració), Application Server 2000 (gestió de llocs web d'alta disponibilitat i granges de servidors), Content Management Server 20001 (gestió de contingut web), Host Integration Server 2000 (connexió a sistemes de llegat), Internet Security and Acceleration Server 2000 (*firewall* i servidor *proxy*), Mobile Information Server 2001 (entrada de connexió a aparells mòbils) o SharePoint Portal Server 2001 (gestió de documents, cerca d'empreses, i motor de portal).

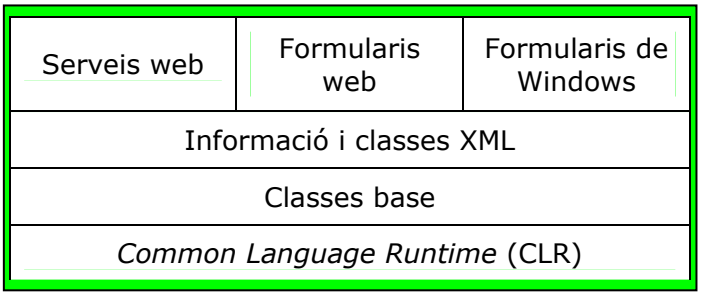
.Net Building Block Services són col·leccions de serveis web que Microsoft espera posar a la venda. Aquests poden utilitzar-se per desenvolupadors com a blocs de construcció a les aplicacions que generin, evitant d'aquesta manera dedicar massa temps a diversos aspectes de "fontaneria" de les seves aplicacions, com per exemple com succeeix amb l'autenticació. Un d'aquests serveis es troba disponible a la seva edició anterior: el servei Passport. Aquest permet un únic accés a través de llocs web. Microsoft ha anunciat els serveis anomenats "HailStorm", que se centraran en les experiències dels usuaris.

El nivell superior de la plataforma .Net és Visual Studio .NET. És una eina de desenvolupament d'aplicacions dissenyada per al desenvolupament ràpid de serveis web i altres aplicacions.

2.4.1 .NET Framework

L'última part de la plataforma .NET està formada per el marc .NET Framework. .NET Framework és la nova plataforma de desenvolupament que ofereix Microsoft i que conté una nova interfície de programes als serveis web i les API, integrant tecnologies com serveis de components COM+, el marc de desenvolupament web ASP (*Active Server Pages*), fiabilitat, seguretat, implementació senzilla i altres protocols dels serveis web

com SOAP, WSDL i UDDI. L'arquitectura de .NET Framework es mostra al següent diagrama:



En direcció ascendent, es comença amb el llenguatge comú en temps d'execució, *Common Language Runtime* (CLR). CLR opera per sobre del sistema operatiu. CLE és el centre neuràlgic de .NET Framework; és la infrastructura en temps d'execució que unifica les implementacions del sistema operatiu. En contret, CLR fa tasques com l'activació d'objectes, recollida d'escombraries, manejament de memòria, execució del codi, realització de comprovacions de seguretat, etc.

CLR és similar a la *Java Virtual Machine* (JVM). La diferència principal amb JVM és que CLR permet utilitzar tots els llenguatges que puguin representar-se amb *Common Intermediate Language* (CIL) o el llenguatge intermedi de Microsoft. Mentre que JVM només és compatible amb Java. Per altra banda, CLR s'ha pensant per a operar només amb sistemes operatius de Windows. Mentre que JVM es troba present per a la majoria de plataformes dels sistemes operatius.

Microsoft va presentar CIL a l'Associació Europea de Fabricants d'Ordinadors ECMA, organització estandarditzada que gestiona C#: el nou llenguatge que Microsoft ha introduït a la seva plataforma .NET. Tanmateix, ECMA també gestiona EMCAScript, un llenguatge de seqüència de comandes que es troba als principals navegadors com Internet Explorer (conegut com a JScript en aquest navegador) i Netscape Navigator (on es conegut com JavaScript). A part de C#, altres llenguatges de .NET són Visual Basic, C++ (si s'utilitza un codi gestionat), i JScript. També hi ha llenguatges de tercers que poden operar a .NET com COBOL, Pascal, Eiffel, Phyton i SmallTalk entre d'altres.

El següent nivell superior al diagrama és la capa de classes base. Aquestes són similars als grups de classes a ATL (*Active Template Library*) i MFC (*Microsoft Foundation Classes*) o algunes de les classes base de Java, i inclouen funcionalitat d'entrada/sortida, gestió de la seguretat, gestió de *threads*, comunicacions de xarxa i processament de grups, entre d'altres.

Les dades i les classes XML amplien les classes base proporcionant un conjunt de classes per al manejament de dades i manipulació XML. Per exemple, això inclou la gestió de dades emmagatzemades a les bases de dades mitjançant ADO.NET i SQL. La compatibilitat relacionada amb dades XML es basa en l'accés i la manipulació de dades XML, cerques de XML amb la incorporació de XPath i transformacions de XML amb XSLT.

La capa superior dels blocs de construcció per a .Net Framework consisteix en serveis web, formularis web i formularis de Windows. Els primer són les interfícies del programa i proporcionen classes per a la creació, implementació i ús dels serveis web mitjançant la utilització de protocols estàndards, com SOAP, WSDL i UDDI.

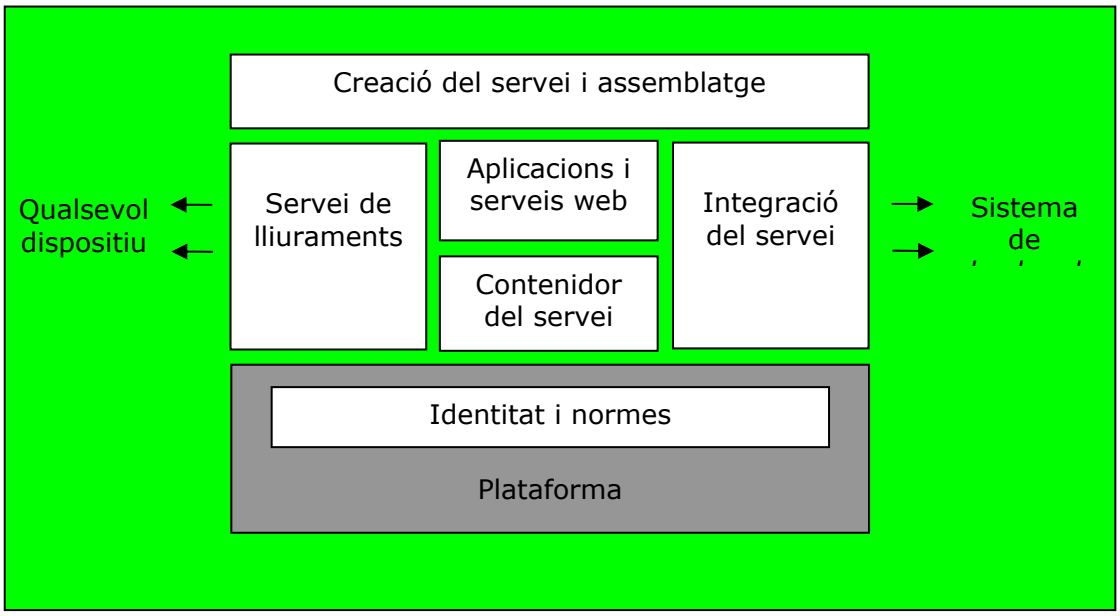
Els formularis web i de Windows proporcionen les operacions de la interfície d'usuari. Els formularis web proporcionen classes per al manejament del disseny i el desenvolupament de formularis web (de base HTML que operen dins d'un navegador). Els formularis de

Windows proporcionen classes per a la creació d'aplicacions natives de Windows *Graphical User Interface* (GUI).

2.5 Sun Microsystems i Sun ONE

L'oferta de Sun als serveis web inclou Sun ONE (*Open Network Environment*), una plataforma creada en torn la gamma del servidor Sun-Netscape Alliance iPlanet i la nova API XML per a serveis web coneguda com JAX Pack.

La plataforma Sun ONE està formada pels següents components:



La capa de creació i assemblatge del servei: conté eines que ajuden en el desenvolupament de serveis i aplicacions. Inclou les eines de Forte Developer de Sun.

La capa de servei de lliuraments: inclou tecnologies que ubiquen, connecten, agreguen, presenten, comuniquen, personalitzen, notifiquen i reparteixen continguts. El servidor iPlanet Portal Server s'ha creat per a aquesta capa.

La capa d'aplicacions i serveis web: dissenyada per a la transformació d'aplicacions de productivitat de transaccions de negocis tradicionals en serveis web que col·laboren. Els productes iPlanet Commerce (iPlanet BuyerXpert, BillerXpert, SellerXpert, MarketMaker) s'ocupen d'aquesta capa.

La capa del contenidor del servei: inclou productes que faciliten la implementació de serveis i els posa a disposició de l'usuari. Aquesta capa inclou l'aplicació iPlanet i els servidors web.

La capa d'integració del servei: ajuda a connectar sistemes i aplicacions actuals. Els productes que formen part d'aquesta capa inclouen el servidor d'integració iPlanet i el iPlanet ECXpert. El servidor d'integració permet l'accés a les dades i la lògica d'emmagatzematge, com SAP o inclús aplicacions personalitzades.

La capa d'identitat i política: s'ocupa dels perfils de gestió d'usuaris, usuaris, context, funcions i seguretat. Els productes de gestió de l'usuari iPlanet: Servidor del directori iPlanet, Meta-Directori, Enrutador d'accés al directori, Administrador, Servidor *proxy* i el Sistema de certificat de gestió iPlanet s'ocupen d'aquesta capa.

La capa de la plataforma: la plataforma del sistema operatiu recomanada per a aquests serveis. Sun recomana la plataforma de sistema operatiu Sun i Sun Cluster 3.0, encara que les eines i els productes mencionats a les capes anteriors funcionen a la majoria de plataformes de sistema operatiu UNIX i Windows.

Els components en aquesta pila es basen en les tecnologies estàndard XML com SOAP, SOAP amb annexes, WSDL, UDDI, etc. i en l'API de Java. Sun també ha definit un conjunt d'API Java per a XML, ambdues API orientades als documents, com JAXP, JAXB, així com les orientades als procediments com JAXM, JAXR i JAX-RPC. Tanmateix, Sun i altres empreses es troben ficades a la definició d'un conjunt d'API estàndard de Java per a XML.

2.6 L'alternativa open-source i la Fundació Apache

Vistes anteriorment les propostes de diferents companyies importants dins del món del software (especialment software propietari), a continuació es parlarà d'una alternativa *open-source* important. No només referint-se als serveis web, sinó a tot un conjunt de camps i aplicacions amb els que aquesta organització (*The Apache Software Foundation*) està relacionada.

2.6.1 La Comunitat Apache

La Fundació de Software Apache és una corporació no-lucrativa, formada a Delaware (USA) al juny de 1999. La Fundació és l'evolució del Grup Apache (*The Apache Group*), un grup de persones format inicialment al 1995 per a desenvolupar el servidor Apache (*Apache HTTP Server*).

La Fundació es va constituir principalment per a:

a. Proporcionar una fundació per a projectes de desenvolupament de software oberts i col·laboratius, facilitant les infrastructures necessàries de hardware, de comunicacions i de negoci.

b. Crear una entitat legal independent a la qual companyies i individus poden donar recursos i estar segurs que aquests recursos seran utilitzats per al benefici públic.

c. Proporcionar una manera a les persones voluntàries de protegir-se dels assumptes legals referents als projectes de la Fundació.

d. Protegir la marca 'Apache', aplicada als seus productes de software, de ser utilitzada per d'altres organitzacions.

La Fundació de Software Apache és una comunitat de desenvolupadors fortament descentralitzada. Està organitzada en grans projectes 'principals' que es divideixen en subprojectes. Alguns exemple d'aquest projectes poden ser:

- **HTTP Server**: conegut comunament com Apache httpd

- **Cocoon**: marc de publicació XML

- **Jakarta**: projectes Java a la part del servidor.

- **Web Services**: projectes relacionats amb el món dels serveis web.
- **XML**: solucions XML enfocades a la web.
- etc.

Es pot trobar més informació sobre la Fundació a la seva web: http://www.apache.org.

En el nostre cas ens centrarem en dues peces de software que es troben dins dels projectes Jakarta i Web Services, aquests són el servidor Tomcat i Axis. A continuació es parla una mica més d'ells.

2.6.2 Apache Tomcat

Tomcat és un contenidor de *servlets* que s'utilitza a la implementació de referència oficial de les tecnologies *Java Servlet* i *Java Server Pages*. Les especificacions d'aquestes tecnologies ha estat desenvolupada per Sun com a un procés de la comunitat Java.

El projecte Tomcat està dividit en tres versions per a donar suport a les diferents especificacions de Servlet/JSP. Així tenim la següent taula:

Especificació Servlet/JSP	Versió de Tomcat
2.4/2.0	5.0.2 Alpha
2.3/1.2	4.1.24
2.2/1.1	3.3.1a

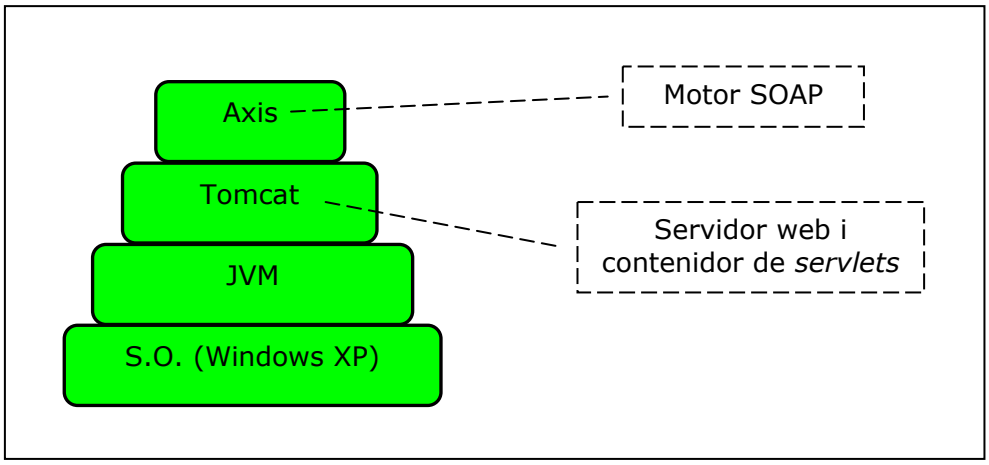
La versió que farem servir nosaltres serà la 4.1.24, ja que es tracta d'una versió completament compatible amb l'altra eina d'Apache que es farà servir (Axis).

La sèrie Tomcat 4.x implementa un nou contenidor de *servlets* (anomenat Catalina) que es basa en una nova arquitectura. La versió 4.1.24 estable és l'última publicació. Tomcat 4.1 és una revisió de Tomcat 4.0.x, i conté millores significatives, incloent:

- Característiques d'administració basades en JMX.
- Aplicació d'administració web basada en JSP i Struts.
- El nou connector Coyote (que suporta HTTP/1.1, AJP 1.3 i JNI).
- El compilador Jasper de pàgines JSP ha estat rescrit.
- Millores de rendiment i d'ús eficient de la memòria.
- etc.

Un contenidor de *servlets* és una part del servidor web o del servidor d'aplicacions que s'encarrega de realitzar tota la feina de les connexions, la descodificació MIME i gestionar la vida dels *servlets*. Pot ser-ho el propi servidor, com és el cas de Tomcat, o estar connectat com a extensió, com és el cas de Jserv, o el propi Tomcat quan treballa conjuntament amb el servidor web Apache.

L'arquitectura que farem servir en el nostre cas serà la següent:



Per a més informació es pot veure la pàgina del projecte Tomcat dins del projecte Jakarta: http://jakarta.apache.org/tomcat/.

2.6.3 Apache Axis

Apache Axis és un servidor i un client SOAP *open-source*, és a dir, és essencialment un motor SOAP, un marc per a construir processadors SOAP com clients, servidors, *gateways*, etc. La versió actual d'Axis està desenvolupada en Java, però s'està desenvolupant una implementació en C++. Axis implementa l'API de JAX-RPC, una de les formes estàndard de programar serveis en Java.

Però Axis no és només un motor SOAP, també inclou:

- Un servidor senzill que pot funcionar sol.
- Un servidor que es pot connectar a un contenidor de *servlets* com Tomcat.
- Suport extens per al *Web Service Description Language* (WSDL).
- Eines per a generar classes Java a partir de WSDL.
- Una utilitat per a monitoritzar paquets TCP/IP.

Axis és la tercera generació del Apache SOAP (que va començar a IBM amb "SOAP4J"). A finals del 2000, els participants d'Apache SOAP v2 van començar a discutir com fer un motor molt més flexible, configurable i capaç de manegar tant SOAP com les especificacions que arribaven del W3C sobre protocols XML.

Després d'un breu període es va veure que es requeria una reconstrucció des de la base. Diversos dels participants de la segona versió d'Apache SOAP van proposar dissenys similars, tots en torn de 'cadenes' configurables de '*handlers*' de missatges els quals implementen petites parts de funcionalitat d'una manera molt flexible i modular.

Després de mesos de contínues discussions i esforç dirigits en aquesta direcció, Axis ofereix actualment les següents funcionalitats:

- **Velocitat**: Axis utilitza l'analitzador SAX (basat en events) per a assolir una velocitat significativament més alta que les versions anteriors d'Apache SOAP.

- **Flexibilitat**: L'arquitectura d'Axis dóna al desenvolupador llibertat completa per a inserir extensions al motor i adaptar-lo millor al processat de capçaleres, millorar el sistema de gestió, o retocar qualsevol altre part.

- **Estabilitat**: Axis defineix un conjunt d'interfícies publicades que canvien relativament poc comparat amb la resta d'Axis.

- **Desenvolupament orientat a components**: Es poden fàcilment definir xarxes reutilitzables de '*handlers*' per a implementar patrons comuns de processament fets servir a les aplicacions, o per a distribuir-los.

- **Marc de transport**: Fa una abstracció neta i simple a l'hora de dissenyar el transport (i.e. intercanvi de missatges SOAP sobre diferents protocols com HTTP, SMTP, *middleware* orientat a missatges, etc.), i el nucli del motor és completament independent del transport.

- **Suport WSDL**: Axis suporta la versió 1.1 del *Web Service Description Language*, el qual permet fàcilment construir representants (*stubs*) per a accedir a serveis remots, i també exportar automàticament descripcions llegibles per la màquina dels serveis implantats en Axis.

Axis ve de "*Apache eXtensible Interaction System*", una forma imaginativa d'indicar que es tracta d'un motor SOAP molt configurable.

Arquitectura

Axis està format per un conjunt de subsistemes treballant alhora. Es tracta bàsicament de processar missatges. Quan la lògica central d'Axis s'està executant, sèries de '*handlers*' són executats en un ordre determinat. Aquest ordre depèn de dos factors: la configuració de la implantació (*deployment*) i si el motor és un client o un servidor. L'objecte que es passa a cada '*handler*' és un *MessageContext*, una estructura que conté diverses parts importants:

1) El missatge petició

2) El missatge resposta

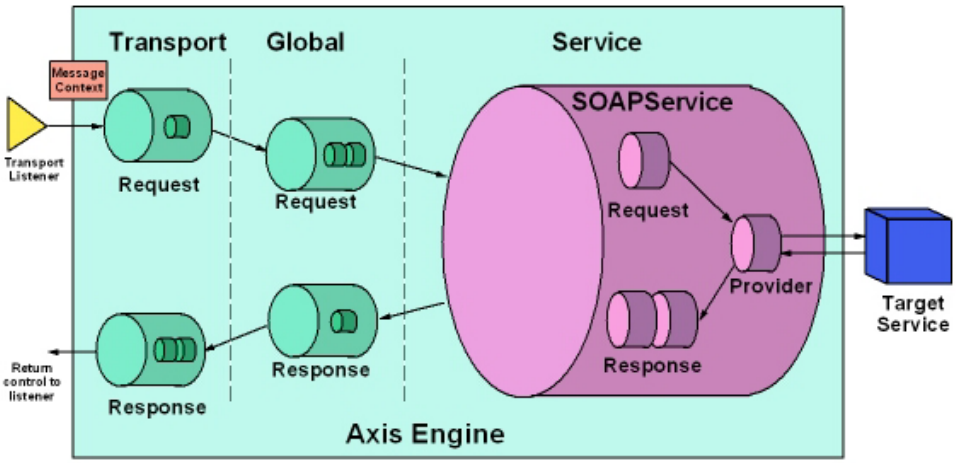
3) Un conjunt de propietats

Les formes bàsiques com és invocat Axis són les següents:

1. Com a **servidor**, un "oïdor del transport" (*Transport Listener*) crea un *MessageContext* i invoca el marc de processament d'Axis.

2. Com a **client**, el codi d'aplicació (normalment afegit mitjançant el model d'Axis per a la programació de clients) generarà un *MessageContext* i invocarà el marc de processat d'Axis.

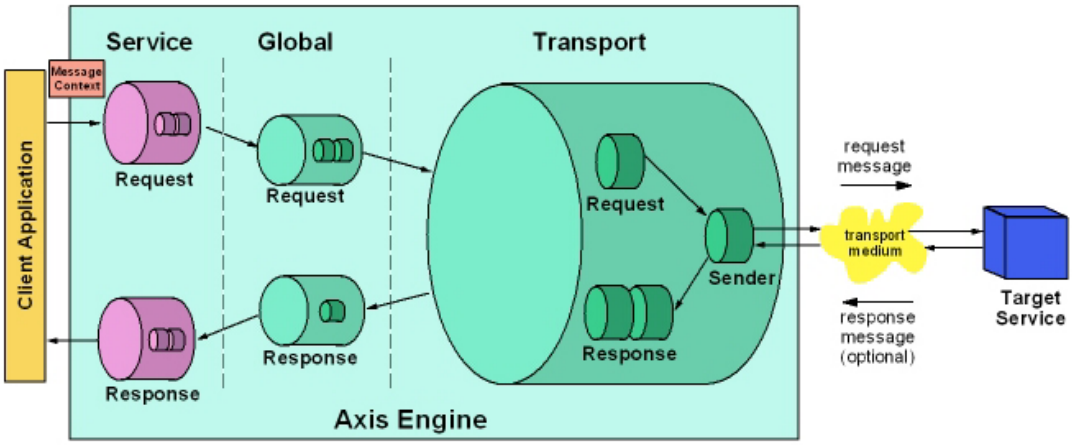
El camí dels missatges des del punt de vista del servidor es mostren al diagrama següent. Els cilindres petits representen els '*handlers*', i els gran representen 'cadenes' (col·leccions ordenades de *handlers*).



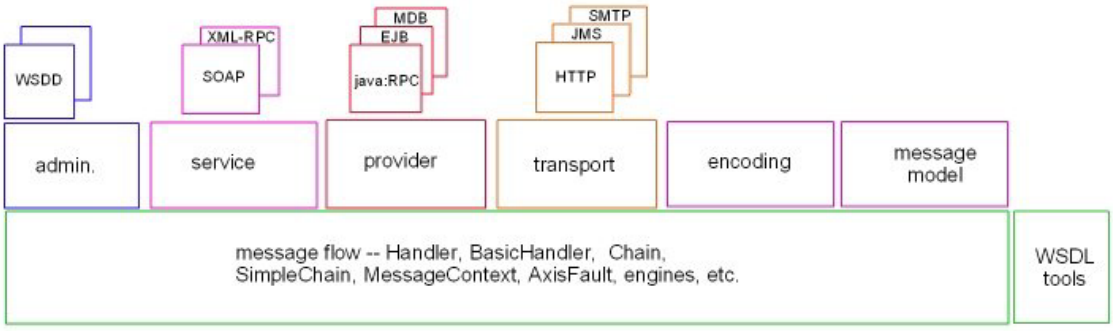
El missatge arriba (d'alguna manera específica d'un protocol) al *Transport Listener*. En aquest cas ens podem imaginar que es tracta d'un *servlet* de HTTP. La seva feina es empaquetar les dades específiques del protocol dins d'un objecte *Message* (org.apache.axis.Message), i ficar el missatge dins d'un *MessageContext*, juntament amb diferents propietats. Una vegada el *MessageContext* està llest, el *Transport Listener* el deixa en mans del motor d'Axis.

El primer que fa el motor d'Axis és buscar el transport pel nom. El transport és un objecte que conté una cadena petició, una cadena resposta, o les dues. Després del *handler* de petició de transport, el motor localitza una cadena de petició global, i invoca els *handlers* corresponents. Durant aquest processos, un *handler* s'encarrega d'assignar-li valor al camp *serviceHandler* del *MessageContext*. Aquest camp determina quin *handler* s'invocarà per a executar la funcionalitat específica del servei. Els serveis en Axis són normalment instàncies de la classe anomenada "SOAPService" (org.apache.axis.handlers.soap.SOAPService), les quals poden contenir cadenes petició o resposta, i que han de contenir un proveïdor, que es tracta d'un *handler* responsable d'implementar el retorn i la lògica del servei.

I des del punt de vista del client, el camí es similar a l'anterior menys en l'ordre de les fases. A continuació es mostra la figura que ho representa:



Axis comprèn diferents subsistemes treballant junts amb l'objectiu de separar clarament responsabilitats i fer Axis modular. El següent esquema mostra les capes dels subsistemes:

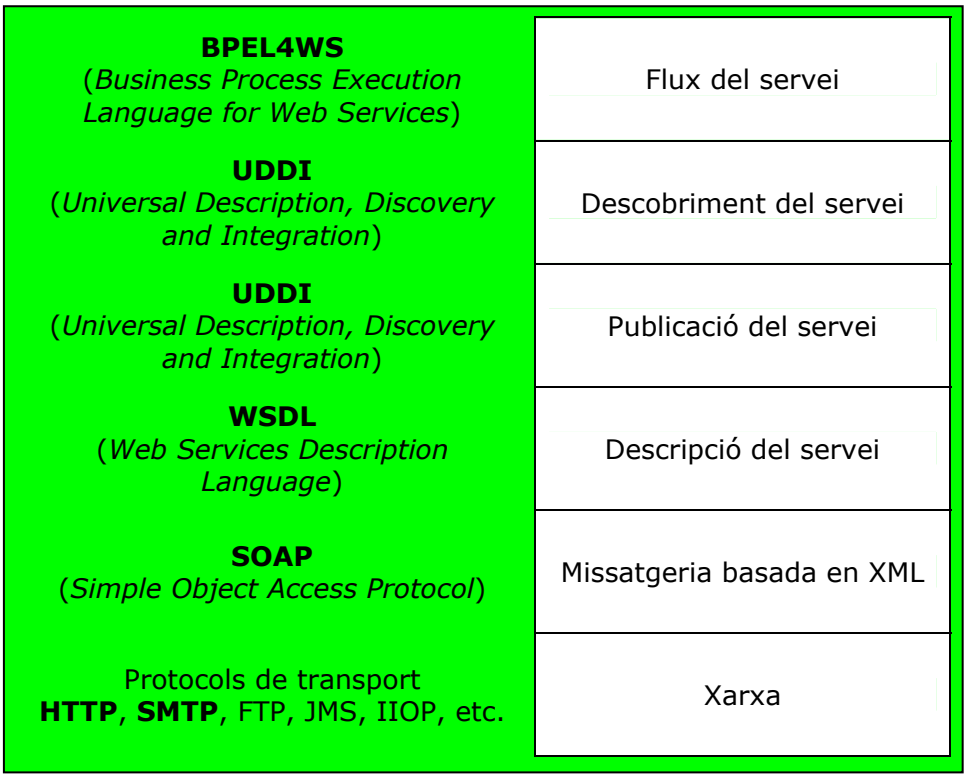


Les capes inferiors són independents de les superiors. Les caixes "apilades" representen alternatives mútuament independents, encara que no mútuament excloents. Per exemple els transports HTTP, SMTP i JMS són independents un de l'altre però es poden usar alhora.

Per a més informació es pot veure la pàgina del projecte Axis: http://ws.apache.org/axis.

3. Protocols i Especificacions

A continuació es mostra la pila dels serveis web i el lloc on apareixen els diferents protocols i especificacions que s'estudiaran.



3.1 SOAP

Com ja s'ha parlat anteriorment, al centre dels serveis web es troba el model SOAP (*Simple Object Access Protocol*), el protocol que fa possible la transmissió de missatges en forma de documents XML i invoca les prestacions als serveis web. L'estàndard de SOAP és la clau per als serveis web.

3.1.1 Fonaments de SOAP

SOAP és una especificació per a utilitzar documents XML com a missatges. L'especificació SOAP està formada per:

- Una sintaxi per a la definició de missatges com a documents XML, referits com a missatges SOAP.
- Un model per a l'intercanvi de missatges SOAP.
- Un conjunt de regles per a la representació de dades dins dels missatges SOAP, conegut com la codificació SOAP.
- Unes pautes que s'han de seguir al transport de missatges SOAP per HTTP.

- Una conversió per a realitzar crides a procediments remots (RPC) mitjançant l'ús de missatges SOAP.

El tema dels serveis web pot arribar a ser confús degut a tota l'activitat i sigles que hi ha darrera. La llista de protocols i tecnologies no para de créixer. De totes les sigles de serveis web, probablement SOAP és la més important, ja que s'està convertint en el protocol estàndard per a l'accés a serveis web, i aquest accés és clau. Per a que els serveis web funcionin com a tecnologia és necessari tenir tècniques ben definides per a la ubicació d'un servei (UDDI) i determinar les seves capacitats (WSDL). No obstant, per a aconseguir l'èxit de qualsevol servei web individual, aquestes tecnologies són opcionals: la documentació escrita o inclòs informal, poden definir la ubicació d'un servei i els seus mètodes. No obstant, sense un protocol amb el que accedir a aquests mètodes, el servei resulta inservible. Avui en dia, SOAP és la millor opció per a aquest protocol.

Tot i que SOAP representa una gran opció com a protocol de missatgeria dels serveis web, no és la única. Els serveis web poden funcionar senzillament amb HTTP GET o únicament mostrar les seves funcionalitats mitjançant XML-RPC. Això no converteix a aquests components en serveis web de menor categoria que un component que treballi amb SOAP. No obstant, generalment, SOAP és el protocol de missatgeria triat per als serveis web. Existeix una acceptació de SOAP a nivell generalitzat per part dels proveïdors i desenvolupadors independents i les eines i implementacions que funcionen amb SOAP milloren constantment.

3.1.2 El model d'intercanvi de missatges SOAP

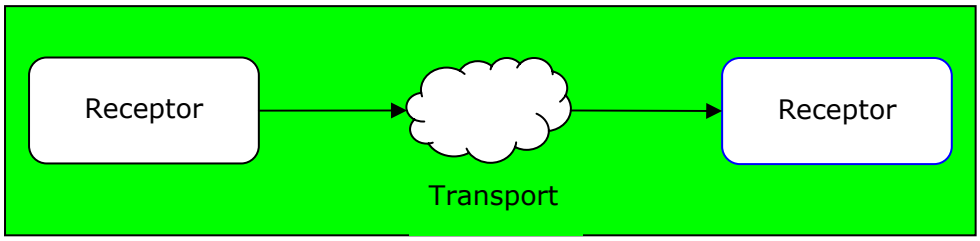
L'especificació SOAP defineix un model per a l'intercanvi de missatges que depèn de tres conceptes bàsics: els missatges són documents XML, viatgen d'un emissor a un receptor i els receptors poden encadenar-se entre si. Es possible la creació de sistemes sofisticats que depenguin de SOAP amb aquests tres conceptes.

3.1.2.1 Documents XML com a missatges

El concepte més bàsic del model SOAP és l'ús de documents XML com a missatges. Els missatges SOAP són XML. Això aporta diversos avantatges que no ofereixen altres protocols de missatgeria. Els missatges XML poden redactar-se i llegir-se amb un editor de text, el que simplificaria el procés de depuració molt més que utilitzant un complex protocol binari. Degut a la gran acceptació generalitzada de XML, existeixen eines que ajuden a utilitzar XML a la majoria de plataformes.

3.1.2.2 Emissors i receptors

Un intercanvi de missatges SOAP implica la participació de dues parts: un emissor i un receptor. El missatges va del emissor al receptor. Aquesta operació és el bloc de construcció bàsic de l'intercanvi de missatges SOAP, la unitat de treball més petita. L'esquema inferior il·lustra aquesta senzilla operació:



No obstant, en molts casos aquest tipus d'operació no és suficient. Un requisit més comú és el que els missatges s'intercanviïn en parell de petició/resposta, que és el mètode que utilitza SOAP amb el transport HTTP i/o la convenció RPC. No obstant, la necessitat del model dificultaria el disseny d'intercanvi de missatges de direcció única. En començar per l'operació més bàsica, un intercanvi de missatges de direcció única de l'emissor al receptor, poden compondre's intercanvis més complicats sense impedir que es produeixin els intercanvis més senzills. Això ofereix l'habilitat per a construir cadenes de missatges.

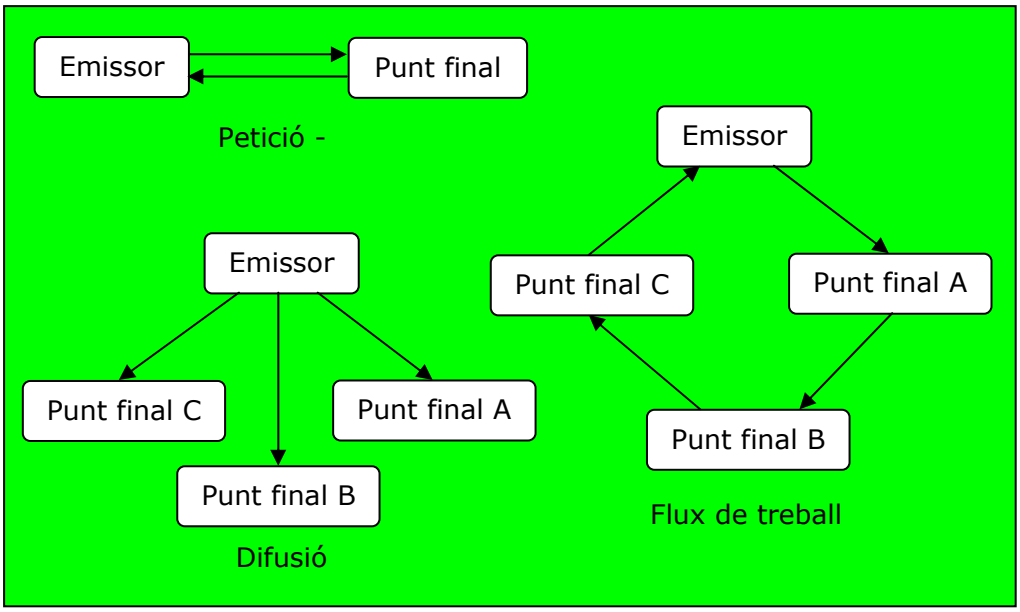
3.1.2.3 Cadenes de missatges

Els missatges SOAP no necessiten seguir un model tradicional de client-servidor. Els missatges poden intercanviar-se d'aquesta forma, com en el cas de HTTP o processar-se mitjançant una cadena d'entitats lògiques. Aquest concepte d'entitat lògica que desenvolupa part del processament d'un missatge SOAP es coneix com a **punt extrem**. Els punts extrems són receptors de missatges SOAP i la seva responsabilitat és la d'examinar un missatge i eliminar la part dirigida a aquest punt extrem per al seu processament.

Arribats a aquest punt convé dir que, tot i la 'O' a la paraula SOAP, no existeix res orientat a l'objecte en aquest model. Els punts extrems, al igual que els clients, poden escriure's en qualsevol llenguatge i no s'assumeix que existeixin "objectes" en cap dels extrems del cable.

Degut a que el model permet la combinació de missatges de direcció única a operacions més complexes, els punts extrems actuen com receptor i emissor. Aquesta capacitat permet la creació d'una cadena de processament, amb l'enrutament de missatges al llarg d'aquesta cadena i algun processament potencial a cada pas. Els punts extrems que actuen com a emissors i receptors transmetent els missatges que reben a un altre punt extrem, reben el nom d'**intermediaris**. Els intermediaris i el concepte de cadena del missatge ofereix als desenvolupadors l'oportunitat de crear sistemes més sofisticats, encadenats els punts extrems entre si.

Alguns dels exemples clàssics de cadenes de missatges poden ser els següents:



3.1.2.4 Comportament dels punts extrems

Plantejar-se SOAP en terme de punts extrems ajuda a comprendre la flexibilitat de la missatgeria SOAP. Independentment de la ruta que agafi un missatge o del número de punts extrems que el processin, tots ells han de processat els missatges d'una determinada manera. A continuació es desenvolupen els tres passos que un punt extrems ha de seguir per a ajustar-se a l'especificació:

- Anàlisi del missatge SOAP per a determinar si conté alguna informació destinada a aquest punt extrem.

- La determinació de les parts destinades al punt extrem específic que siguin obligatòries, en el cas de que hi hagi alguna. Si el punt extrem pot manegar aquestes parts obligatòries, es procedeix amb el processament del missatge. El missatge es rebutja en cas contrari.

- Si el punt extrem és un intermediari, es necessària l'extracció de totes les parts identificades al primer pas abans d'enviar el missatge al següent punt extrem.

En ajustar-se a aquests tres requisits, els punts extrems poden encadenar-se i formar sistemes complexes.

3.1.2.5 Disseny modular

SOAP és obert i extensible. Això significa que, per exemple, les situacions descrites a continuació són acceptables i estan admeses per l'especificació SOAP:

- Una aplicació d'escriptori composa un missatge SOAP que requereix les accions cotitzades a borsa, i ho envia com a contingut d'un HTTP POST. Un servidor web rep el POST, processa el missatge, i torna un missatge SOAP a la resposta de HTTP.

- Un procés d'un servidor redacta un missatge SOAP que descriu un event del sistema i publica el missatge mitjançant canals amb nom a d'altres servidors a la xarxa.

- Etc.

El motiu de que SOAP es pugui utilitzar en situacions tan diverses amb el mateix model, és el seu disseny modular. A la totalitat de la seva especificació, es guarden diversos llocs que mantenen oberts per a la futura extensió del protocol. SOAP està dissenyat per a ser extensible a tots els sectors que s'enumeren a continuació:

- **Sintaxi del missatge**: el format del missatge SOAP disposa d'un sector independent ideat per a futures extensions (aquest sector és l'element *Header*).

- **Dades**: La càrrega útil pot contenir qualsevol tipus de dades. SOAP proporciona un mètode per a la codificació de dades, però les aplicacions poden definir les seves pròpies regles.

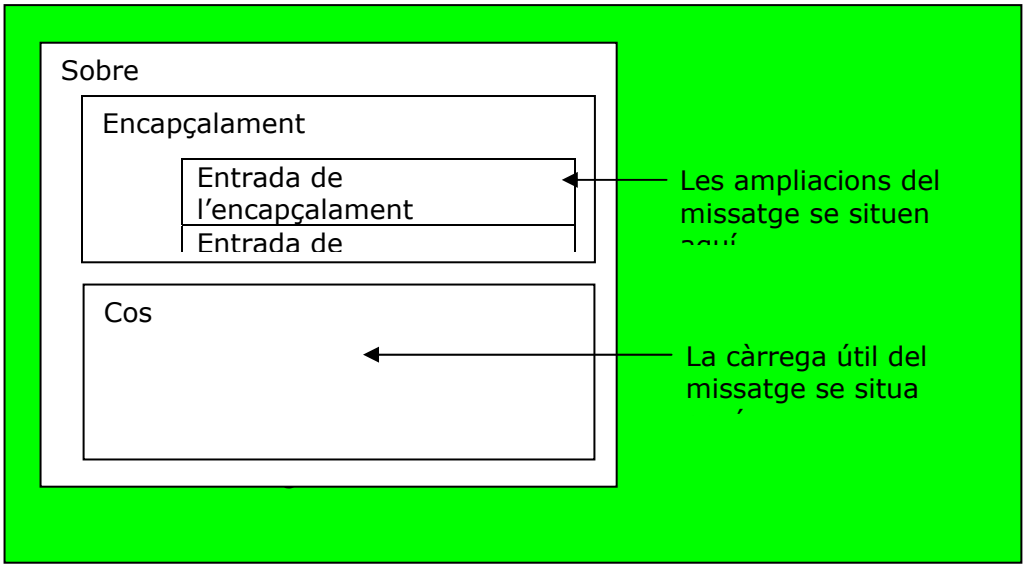
- **Transport**: SOAP no obliga la manera com els missatges es transporten durant l'intercanvi. SOAP defineix la manera com els missatges s'intercanvien per HTTP, però qualsevol protocol de comunicació o mètode pot substituir a HTTP.

- **Propòsit**: SOAP no defineix el que es vol incloure al missatge. Tot i que això pugui semblar com que estan comptant les dades dues vegades, existeix una diferència entre les dades i el propòsit.

L'extensibilitat és important, però sense algunes implementacions concretes d'aquests conceptes, SOAP seria simplement un munt de conceptes interessants. Afortunadament, els autors de SOAP proporcionen una descripció d'una implementació per a dades, transport i propòsit. Per a les dades, l'especificació proporciona les regles de codificació SOAP. Per al transport, es defineix un enllaç per a HTTP. Y finalment, per al propòsit, l'especificació defineix una convenció per a l'ús de missatges SOAP per a RPC.
És important recordar aquests quatre conceptes ja que el fet que estiguin separats del protocol i per tant siguin extensibles, és un dels avantatges principals de SOAP.

3.1.3 Missatges SOAP

Un cop estudiat SOAP des d'un nivell superior, a continuació s'estudien els detalls més importants de SOAP: l'estructura d'un missatge. Primer i abans de tot, SOAP utilitza la sintaxi XML per als missatges. L'estructura d'un missatge SOAP es mostra a continuació:



El diagrama mostra la manera com un missatge SOAP pot desglossar-se en components que més endavant es desenvoluparan en detall. Un missatge SOAP conté una càrrega útil, la informació de l'aplicació específica. Aquest és un exemple d'un missatge SOAP com a document XML real:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
       soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <soap:Header>
    <h:from xmlns:h="http://www.wrox.com/Header">SoapGuy@wrox.com
    </h:from>
  </soap:Header>
  <soap:Body>
    <w:GetSecretIdentity xmlns:"http://www.wrox.com/heroes/">
      <w:codename>XSLT-Man</w:codename>
    </w:GetSecretIdentity>
  </soap:Body>
</soap:Envelope>
```

Abans d'endinsar-nos en els continguts del missatge SOAP, repassarem el XML del missatge. Com pot observar-se, el missatge SOAP depèn en gran mesura dels espais de noms XML. Tots els elements d'aquest document es prefixen amb un espai de nom i existeix una bona raó del motiu pel qual l'especificació SOAP utilitza espais de nom de manera tan extensiva. Per a que un missatge SOAP porti alguna càrrega útil XML arbitrària, tots els elements del missatge han de trobar-se a l'abast per a poder evitar conflictes als noms dels elements.

Els espais de noms a la Recomanació de XML poden trobar-se a:
http://www.w3.org/TR/REC-xml-names/
El prefix 'soap' dels espais de noms s'utilitza a la majoria d'elements del missatge anterior. En aquest exemple el prefix s'associa a l'espai de nom URI http://schemas.xmlsoap.org/envelope/ i identifica els elements que formen part d'un missatge SOAP estàndard. Igualment que tots els prefixes de espais de nom, l'elecció de 'soap' és irrellevant. El prefix de l'espai de noms podria haver estat un completament diferent. Tanmateix, aquest prefix pot eliminar-se per complet si aquest espai de nom és el predeterminat per al document. L'espai de nom predeterminat s'assigna utilitzant l'atribut 'xmlns'.

Tots els elements del missatge associats amb l'espai de nom 'soap' són elements estàndard d'un missatge SOAP, al igual que els atributs. Qualsevol altre element està relacionat a les extensions del missatge o a la càrrega útil del missatge. Existeixen tres elements SOAP estàndard que apareixen a l'exemple: 'Envelope', 'Body' i 'Header'. Tanmateix, hi ha un altre element estàndard que no apareix al missatge que és l'element 'Fault'.

A continuació es detallen més aquests elements.

3.1.3.1 Envelope (Sobre)

Tal com indica el seu nom, l'element 'Envelope' actua com un recipient per als altres elements del missatge SOAP. En ser l'element superior, 'Envelope' és el missatge.

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
       soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <soap:Header>
    <h:from xmlns:h="http://www.wrox.com/Header">SoapGuy@wrox.com
    </h:from>
  </soap:Header>
```

```
  <soap:Body>
    <w:GetSecretIdentity xmlns:"http://www.wrox.com/heroes/">
      <w:codename>XSLT-Man</w:codename>
    </w:GetSecretIdentity>
  </soap:Body>
</soap:Envelope>
```

Els missatges SOAP indiquen la seva versió per l'espai de nom de l'element 'Envelope'. L'única versió reconeguda fins al moment per SOAP 1.1 és el URI http://schemas.xmlsoap.org/soap/envelope/. Els missatges que no utilitzen aquest espai de nom no són vàlids i els punts extrems que els reben missatges amb un altre espai de nom han de retornar un error.

L'ús de l'espai de nom 'Envelope' per a indicar les versions dels missatges és un bon exemple del punt fins al que l'especificació SOAP depèn dels espais de nom XML. Sense els espais de nom XML, seria molt difícil definir un format obert XML per a missatges sense un conflicte de noms amb càrrega útil XML del missatge.

3.1.3.1.1 Atribut encodingstyle

L'especificació defineix un atribut anomenat 'encodingstyle' que s'utilitza per a descriure la manera com les dades es representaran al missatge. La codificació és el mètode que s'utilitza per a la representació de les dades. L'atribut 'encodingstyle' pot aparèixer a qualsevol element del missatge però en el cas de la codificació de SOAP, amb freqüència es troba a l'element 'Envelope'.

3.1.3.2 Body (Cos)

L'element 'Body' d'un missatge SOAP és la ubicació per a les dades d'una aplicació específica. Aquest conté la càrrega útil del missatge, duent les dades que representen l'objectiu del missatge. Pot tractar-se d'una crida a procediment remot, una comanda de compra, un full d'estil o qualsevol XML que necessiti el seu intercanvi mitjançant un missatge.

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
       soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <soap:Header>
    <h:from xmlns:h="http://www.wrox.com/Header">SoapGuy@wrox.com
    </h:from>
  </soap:Header>
  <soap:Body>
    <w:GetSecretIdentity xmlns:"http://www.wrox.com/heroes/">
      <w:codename>XSLT-Man</w:codename>
    </w:GetSecretIdentity>
  </soap:Body>
</soap:Envelope>
```

L'element 'Body' ha de dependre directament de l'element 'Envelope'. En el cas que no existeixi un element 'Header', l'element 'Body' es considera el primer secundari; si un element 'Header' apareix al missatge, aquest va seguit immediatament de l'element 'Body' i es codifica d'acord amb la convenció i codificació triada.

3.1.3.3 Header (Capçalera)

L'estructura del missatge SOAP dels elements 'Envelope' y 'Body' és oberta, i s'assigna de manera correcta a les diferents situacions de la missatgeria. L'element 'Body' encapsula la càrrega útil del missatge, però en ocasions, les dades de la càrrega útil no són suficients: per exemple, un missatge pot ser part d'un conjunt de missatges que s'han de processar com a una única transacció lògica, o el missatge ha d'executar-se a un objecte persistent ubicat al servidor. Temes com les transaccions i les referències d'objectes resulten vitals per al missatge, però estan separades de la càrrega útil.

És impossible poder predir tots els tipus d'extensions que es necessitaran a un missatge SOAP. Per aquesta raó, els autors van crear l'element 'Header'. El seu objectiu es l'encapsulació de les extensions del format del missatge sense tenir que connectar-les a la càrrega útil o modificar l'estructura fonamental de SOAP. Això permet que les extensions com transaccions, encriptació, referències d'objecte, facturació i d'altres es puguin afegir al llarg del temps sense trencar l'especificació.

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
       soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <soap:Header>
    <h:from xmlns:h="http://www.wrox.com/Header">SoapGuy@wrox.com
    </h:from>
  </soap:Header>
  <soap:Body>
    <w:GetSecretIdentity xmlns:"http://www.wrox.com/heroes/">
      <w:codename>XSLT-Man</w:codename>
    </w:GetSecretIdentity>
  </soap:Body>
</soap:Envelope>
```

Al igual que l'element 'Body', l'element 'Header' ha de seguir immediatament l'element 'Envelope'. Tot i que és un element opcional, en cas de que aparegui, sempre ha de situar-se immediatament darrera. L'element 'Header' conté un o més elements secundaris coneguts com entrades. Les entrades del 'Header' s'utilitzen per a afegir qualsevol tipus d'extensió a un missatge que, de manera predeterminada, un punt extrem ignora si no ho comprèn. Això permet el desenvolupament de les extensions sense trencar els punts extrems ja existents. No obstant, les extensions que tinguin requisits addicionals utilitzen els atributs *mustUnderstand* i *actor*.

3.1.3.3.1 Atribut actor

El model SOAP permet encadenar punts extrems, pel que és necessari identificar les parts del missatge que van unides a un punt extrem específic de la cadena. Això resulta necessari en el cas de la càrrega útil, ja que el final del punt extrem de la cadena és l'objectiu d'aquesta. No obstant, en el cas dels elements 'Header', aquesta identificació es torna important.

L'atribut *actor* s'utilitza per a unir un element 'Header' amb un punt extrem en particular. El valor de l'atribut *actor* és un URI que identifica al punt extrem específic de l'element 'Header'. SOAP afegeix un valor especial a dos valors de l'atribut *actor* que s'encarreguen de temes relacionats amb l'encadenat del missatge. Si el valor és http://schemas.xmlsoap.org/soap/actor/next, l'entrada es dirigeix al primer punt extrem que trobi. L'omissió de l'atribut *actor* indica que l'entrada va dirigida al punt extrem final, el mateix que s'ocupa de processar la càrrega útil.

El tema de l'atribut actor cobra importància quan es tracta d'intermediaris i els elements 'Header' i la relació entre ells. Després de molt debat, el consens és que els intermediaris amb un comportament adequat saben reconèixer si són els últims punt extrems de la cadena i, en cas que no ho siguin, no han de modificar els elements 'Header' que no comptin amb un atribut *actor*. Tanmateix, els intermediaris han d'eliminar els elements 'Header' que processin.

3.1.3.3.2 Atribut mustUnderstand

A l'exemple de l'element 'Header' en el que es representa una transacció, els desenvolupadors no podrien acceptar el punt extrem i ignorar l'extensió. Si un missatge forma part d0una transacció ha de ser part d'una transacció, pel que els punts extrems que no puguin utilitzar-se a transaccions no han d'intentar processar un missatge. És aquí on l'atribut *mustUnderstand* resulta d'utilitat.

L'atribut *mustUnderstand* s'utilitza a qualsevol part d'un missatge SOAP, encara que amb freqüència es troba a l'element 'Header'. El valor de l'atribut pot variar entre '1', que indica que l'element és obligatori i '0', indicant que és opcional. L'absència de l'atribut equival al valor '0'.

3.1.3.4 Fault (Fallada)

Fins al moment, tot el que s'ha vist del format del missatge SOAP ha estat dirigit a la manera de crear missatges clars i sense cap fallada en el procés d'enviament o recepció. Clarament, aquesta no és una perspectiva real del comportament d'una aplicació. De la mateixa manera que els missatges SOAP tenen una ubicació i un format específic per a la seva versió, estil de codificació, càrrega útil i extensions, també compten amb una ubicació i un format per als errors. L'element que representa un error a un missatge SOAP és l'element 'Fault', que pot comparar-se a les excepcions del serveis Web o considerar-se una forma estàndard de tornar a l'emissor d'un missatge un informe sobre un comportament inesperat.

Per regla general, les fallades s'associen amb un missatge de resposta. Tot i que l'especificació no exclou als elements 'Fault' de les peticions, les implementacions del servidor no solen respondre de forma positiva a aquestes peticions.

En el cas de que aparegui un element 'Fault', aquest ha de fer-ho a la càrrega útil del missatge SOAP, el que significa que ha d'aparèixer com un element fill de 'Body'.

3.1.3.4.1 Element faultcode

L'element *faultcode* conté un valor que li comunica un estat d'error a l'aplicació. Això significa que aquest valor és per a l'ús d'una màquina i no per a la visualització per part d'usuaris potencials. El valor de l'element *faultcode* ha de ser el d'un nom qualificat, com si es tractés d'un element dins del mateix missatge.

Els valors estàndard de l'element *faultcode* estan definits a l'especificació de SOAP 1.1 i són:

- *VersionMismatch*: aquest valor indica que l'espai de nom de l'element 'Envelope' de SOAP no és http://schemas.xmlsoap.org/soap/envelope/. Actualment és la única versió acceptada d'un missatge SOAP i indica que el missatge s'ajusta a l'especificació 1.1.

- *MustUnderstand*: aquest valor es torna en un element *faultcode* quan un punt extrem es troba amb una entrada de l'element 'Header' (que contingui l'atribut *mustUnderstand* amb un valor '1') que no reconeix.

- *Client*: Aquest valor s'utilitza en l'element *faultcode* quan existeix un problema al missatge rebut. Això inclou des de la falta d'un element fins a un espai de nom incorrecte al contingut, encara que aquest valor de l'element *faultcode* afirma que el responsable de l'error és el missatge rebut.

- *Server*: A diferència del codi d'error *Client*, *Server* indica que ha sorgit un problema durant el processament que no està directament relacionat amb el contingut del missatge. Un exemple d'aquest tipus succeeix quan la base de dades que utilitza el punt extrem per a retornar informació no es troba operativa.

3.1.3.4.2 Element faultstring

Si l'element *faultcode* conté informació de la fallada dirigida a l'equip, el valor de l'element *faultstring* està dirigit a l'usuari. L'element *faultstring* conté un valor que descriu breument la fallada que ha tingut lloc per a que tingui sentit en el cas de que aparegués a l'usuari en un diàleg d'error. Això no exclou la seva naturalesa tècnica.

3.1.3.4.3 Element faultactor

Per regla general, és tan important saber el lloc en el que es produeix un error com el tipus d'error que és. Això demostra ser especialment cert en sistemes que inclouen intermediaris SOAP. Si un missatge ha de passar per una dotzena de punt extrems abans d'arribar al seu destí final per al processament de la càrrega útil, el desenvolupador ha de conèixer el lloc a la cadena d'enrutament en el qual te lloc l'error. L'element *faultactor* conté un URI que identifica el punt extrem en el s'ha originat la fallada.

L'element *faultactor* només és obligatori per als intermediaris. En el cas de que una fallada tingui lloc a un intermediari, aquesta ha de tenir un element *faultactor*. Si la fallada té lloc al destí final, al punt extrem no se li exigeix omplir aquest valor (tot i que pot triar fer-ho i resultaria apropiat per a desenvolupadors que utilitzessin els punts extrems). Això significa que pot assumir-se que una fallada sense un element *faultactor* s'ha produït al punt extrem final.

Com a exemple de missatge d'error amb aquests tres elements podríem tenir el següent:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
      soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <soap:Body>
    <soap:Fault>
      <faultcode>soap:Client.MissingParameter</faultcode>
      <faultstring>A parameter was missing</faultstring>
      <faultactor>http://www.wrox.com/heroes/endpnt.asp</faultactor>
    </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

3.1.4 Repàs al funcionament del punt extrem

Els tres passos al processament del missatge SOAP són els següents:

1. S'examina el missatge SOAP per a comprovar si conté informació dirigida al punt extrem concret. Tanmateix, s'estudia l'encapçalament per a comprovar si existeixen entrades dirigides a aquest punt extrem, tant per posició (el següent o l'últim punt extrem) com per URI. Si es tracta de l'últim punt extrem, també s'analitza el contingut.

2. S'estudien les entrades de l'encapçalament dirigit al punt extrem en qüestió. En cas de que existeixi cap marcada amb *mustUnderstand* = '1' que el punt extrem no ha reconegut, ha de tornar-se un codi d'error *MustUnderstand*.

3. Si el punt extrem és un intermediari, és necessari eliminar les entrades de l'encapçalament que s'han processat abans d'enviar-se el missatge al següent punt extrem. Això no inclou el contingut, ja que l'últim punt extrem és l'únic capaç de processar-lo.

Aquest passos, al igual que d'altres requeriments SOAP, són transparents per a la majoria de desenvolupadors. Les diverses eines e implementacions SOAP tindran en compte aquests requeriments a l'hora de generar els punts extrems. No obstant, part de l'atractiu de SOAP resideix en els escassos requeriments per als punts extrems. Si un punt extrem compleix els tres passos anteriors es considera vàlid.

3.1.5 Estil de codificació

La possibilitat de triar un conjunt de regles per a la representació de dades a un missatge és de gran importància a la naturalesa oberta de SOAP. Clarament, la definició d'un conjunt de regles no resulta avantatjosa si no es poden determinar les regles de codificació que s'han utilitzat a la codificació d'un missatge SOAP concret. L'atribut *encodingStyle* que defineix l'especificació SOAP s'utilitza per a identificar les regles de codificació usades a un missatge específic.

L'atribut *encodingStyle* es troba normalment a l'element 'Envelope'. No obstant, és possible utilitzar aquest atribut en qualsevol element d'un missatge SOAP. Tot i que l'especificació SOAP defineix un conjunt de regles que s'assignen correctament als blocs de programació, **no existeix la codificació predeterminada**. Això significa que si l'atribut *encodingStyle* no apareix al missatge, el receptor no pot assumir la manera com es representaran les dades dins del missatge. Del mateix mode, la longitud zero de *encodingStyle*="" equival a un atribut *encodingStyle* absent. En qualsevol cas, la falta de *encodingStyle* significa que la implementació ha de desxifrar la manera de descentralitzar les dades sense cap tipus d'ajuda.

3.1.5.1 Codificació SOAP

L'especificació SOAP defineix un únic conjunt de regles de codificació conegudes com a **codificació SOAP**. La codificació SOAP es basa en esquemes XML i, com a tal, segueix un estil molt semblant al de molts tipus estàndard i blocs de construcció que serien familiars als desenvolupadors. El valor de l'atribut *encodingStyle* per a la codificació SOAP és http://schemas.xmlsoap.org/soap/encoding/ , que apunta a l'esquema XML que defineix les regles de codificació.

3.1.5.1.1 Tipus de dades primitius

A la codificació SOAP, els tipus de dades primitius es representen en el contingut com a elements individuals. La codificació SIAO exposa tots els tipus primitius que es construeixen dins de l'especificació d'esquemes XML. Si s'utilitza un tipus primitiu a la codificació SOAP, aquest ha de procedir de l'esquema XML o derivar d'un tipus que ho faci. L'espai de nom que s'associa amb els tipus de dades dels esquemes XML és http://www.w3.org/1999/XMLSchema i proporciona els tipus comuns previstos per diversos programadors, com són *string*, *integer*, *float*, *date*, entre d'altres. Si s'assumeix que el prefix *xsd* està associat al URI dels esquemes XML i el prefix *soapENC* al URI, tots dos valors de càrrega útil funcionaran amb cadenes. Això fa referència als esquemes XML:

```
<codename xsi:type="xsd:string">Hulk</codename>
```

D'altra banda, això fa referència a la codificació SOAP:

```
<codename xsi:type="soapENC:string">Hulk</codename>
```

Per a consultar els esquemes XML on s'inclouen els tipus disponibles als esquemes XML es pot anar a http://www.w3.org/TR/xmlschema-0/ .

3.1.5.1.2 Atribut xsi:type

El propòsit de SOAP és el de facilitar la comunicació entre una àmplia varietat de llenguatges i no tots els llenguatges es creen del mateix mode. A molts llenguatges de comandes, el tipus no és un concepte concret. Per a ajudar a anivellar el camp de joc, SOAP agafa l'atribut *xsi:type* dels esquemes XML.

L'atribut xsi:type és una manera que tenen els elements de la càrrega útil d'indicar el seu tipus. Aquest atribut està associat als esquemes XML i el prefix xsi al URI http://www.w3.org/1999/XMLSchema-instance. Aquest pot aparèixer a qualsevol element de la càrrega útil.

3.1.5.1.3 Enumeracions

La codificació SOAP permet la definició de tipus enumerats. Un cop més, aquesta codificació s'agafa dels esquemes XML, que també utilitzen el concepte d'enumeració. Una enumeració és un conjunt de valors amb nom basats en un tipus bàsic. Per exemple, pot definir-se una enumeració que representi llocs geogràfics ("nord", "sud", etc.). Per a definir una enumeració es necessari utilitzar esquemes XML.

A continuació es mostra un exemple:

```
<simpleType name="Region" base="xsd:string">
  <enumeration value="North"/>
  <enumeration value="South"/>
  <enumeration value="East"/>
  <enumeration value="West"/>
</simpleType>
```

3.1.5.1.4 Dades binàries

SOAP i els esquemes XML proporcionen un tipus per a la representació de dades binàries, com a part dels tipus de dades primitius que utilitzen. Una tècnica per a la utilització de dades binàries és mitjançant l'ús del tipus *base64*. D'aquesta manera poden representar-se dades binàries, com un arxiu d'una imatge, com a matriu de bytes al missatge. El

tipus base64 transforma les dades binàries en text utilitzant l'algorisme de codificació-base64 dels esquemes XML. No existeix cap relació entre SOAP i la codificació-base64, pel que si s'utilitza, l'aplicació (o implementació de SOAP a la plataforma) ha de poder comprendre i utilitzar aquesta codificació.

3.1.5.1.5 Estructures

Les estructures o registres, són de fàcil representació en XML ja que compten amb membres amb noms exclusius.

Un exemple d'estructura en llenguatge de programació traduïda a missatge SOAP podria ser el següent:

```
Struct SuperHero {
  string sCodeName;
  string sFirstName;
  string sLastName;
  int nAge;
}

SuperHero hero = {"Hulk", "Bruce", "Banner", 32};
```

I dins de SOAP l'aspecte seria aquest:

```
<hero xsi:type="x:SuperHero">
  <sCodeName xsi:type="xsd:string">Hulk</sCodeName>
  <sFirstName xsi:type="xsd:string">Bruce</sFirstName>
  <sLastName xsi:type="xsd:string">Banner</sLastName>
  <nAge xsi:type="xsd:integer">32</nAge>
</hero>
```

3.1.5.1.6 Arrays

Els arrays també són tipus de dades compostos i es representen de forma molt semblant a les estructures. La diferència entre els arrays i les estructures resideix a la forma com es fan referència els seus membres. Les estructures estan formades per dades que poden identificar-se per nom i els membres dels arrays s'identifiquen per la seva posició. Els noms dels elements d'arrays són insignificants pel que no es poden utilitzar per a buscar un valor.

A la codificació SOAP, els arrays es consideren un tipus especial. Aquest tipus s'indica pel seu atribut *xsi:type*, que és *SOAP-ENC:Array*. Al igual que amb tota la codificació SOAP, l'espai de nom que té associat és http://schemas.xmlsoap.org/soap/encoding. Els elements amb aquest xsi:type estan declarats com arrays de codificació SOAP. El tipus de membre de l'array es declara utilitzant un altre atribut, *SOAP-ENC:arrayType*. Aquest atribut indica el tipus i el tamany de l'array. Els arrays a la codificació SOAP poden ser confosos; a continuació es mostra un exemple d'un array senzill format per cinc enters:

```
<number xsi:type="SOAP-ENC:Array" SOAP-ENC:arrayType="xsd:integer[5]">
  <item>10</item>
  <item>20</item>
  <item>30</item>
  <item>40</item>
  <item>50</item>
</numbers>
```

3.1.6 Separació del missatge i el transport

Una de les decisions de disseny millor preses pels autors de SOAP va ser la de separar la definició del missatge del transport del propi missatge. Pot semblar absurd, però no existeix res a l'especificació que requereixi que els ordinadors formin part d'un missatge SOAP, pel que, tenint això en compte, a continuació s'enumeren els possibles transports per als missatges SOAP (alguns més apropiats que d'altres).

- HTTP
- SMTP
- MQSeries
- Sockets "crus"
- Arxius
- ...

La majoria de desenvolupadors se centren en HTTP i en el transport estàndard per als seus missatges SOAP. A mesura que la compatibilitat de SOAP continuï creixent, sorgiran definicions i implementacions d'enllaços de transport SOAP per a qualsevol nombre de protocols.

3.1.6.1 Exemple amb HTTP

Quan diversos desenvolupadors pensen en SOAP, pensen en XML per HTTP. HTTP és un transport excel·lent per a SOAP degut a la seva acceptació generalitzada. HTTP és el protocol omnipresent de la Web i un constant recordatori de que els estàndards poden funcionar. La combinació de HTTP, el protocol de transport estàndard per a la Web i SOAP, el candidat avantatjat del format de missatges estàndard, proporciona una poderosa eina. HTTP resulta un transport tan magnífic per a SOAP que els autors es van assegurar que les regles per a la utilització de HTTP com a transport fossin part de l'especificació SOAP.

Existeixen únicament un parell de regles bàsiques per a la utilització de HTTP com a transport de SOAP. El mecanisme per a enviar un missatge SOAP per HTTP és el mètode estàndard de HTTP POST. Un HTTP POST envia un bloc de dades a un URI particular al servidor Web. En el cas dels missatges SOAP aquest bloc de dades és el mateix missatge SOAP. Degut a que el missatge SOAP és XML, l'encapçalament de tipus contingut del HTTP POST ha de ser *text/xml*. Si sorgeix una contestació al missatge, aquesta es retorna a la resposta HTTP.

El transport d'un missatge SOAP per HTTP tindria un aspecte semblant a aquest:

```
POST /endpoint.asp HTTP/1.1
Content-Type: text/xml
Content-Length: ###
SOAPAction: "urn:wroxheroes"

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
      soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <soap:Header>
    <h:from xmlns:h="http://www.wrox.com/Header">SoapGuy@wrox.com
    </h:from>
  </soap:Header>
  <soap:Body>
    <w:GetSecretIdentity xmlns:"http://www.wrox.com/heroes/">
      <codename>XSLT-Man</codename>
    </w:GetSecretIdentity>
```

```
  </soap:Body>
</soap:Envelope>
```

3.1.6.1.1 Encapçalament SOAPAction

L'especificació SOAP defineix un únic encapçalament HTTP addicional, *SOAPAction*, que s'utilitza quan els missatges SOAP es transporten per HTTP. Aquest encapçalament proporciona una pista als servidors de que un HTTP POST conté un missatge SOAP, i que el valor de l'encapçalament és un URI que indica el propòsit del missatge SOAP. Això permet que, en funció de la presència de l'encapçalament de *SOAPAction*, els *firewalls* i altres servidors puguin realitzar un processament condicional.

3.1.6.1.2 Codi de l'estat

SOAP imposa un requisit al transport HTTP quan s'utilitza per a l'intercanvi de missatges SOAP. Si el missatge de resposta conté un error, el codi de l'estat de la resposta HTTP ha de ser 500, la qual cosa indica error intern del servidor.

El següent exemple mostra la resposta a la crida de l'exemple anterior.

```
HTTP/1.1 200 OK
Content-Type: text/xml
Content-Length: ###

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
       soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
                xmlns:xsd="http://www.w3.org/1999/XMLSchema">
  <soap:Body>
    <w:GetSecretIdentityResponse mlns:"http://www.wrox.com/heroes/">
      <return xsi:type="xsd:string">Michael Kay</return>
    </w:GetSecretIdentityResponse>
  </soap:Body>
</soap:Envelope>
```

La resposta anterior té èxit pel que la primera línia de la resposta de l'exemple conté el codi d'estat 200. A continuació es torna a fer la mateixa crida però enviant una petició diferent que conté problemes.

```
POST /endpoint.asp HTTP/1.1
Content-Type: text/xml
Content-Length: ###
SOAPAction: "urn:wroxheroes"

<Envelope>
  <Body>
    <w:GetSecretIdentity xmlns:"http://www.wrox.com/heroes/">
      <codename>XSLT-Man</codename>
    </w:GetSecretIdentity>
  </Body>
</Envelope>
```

En aquest cas, l'espai de nom SOAP no hi és, pel que el punt extrem ha de retornar un error, el que significa que ha d'utilitzar un codi d'estat 500 a la resposta. Aquest

missatge de resposta es mostra a continuació, i conté un error *VersionMismatch* així com el codi d'estat apropiat.

```
HTTP/1.1 500 Internal Server Error
Content-Type: text/xml
Content-Length: ###

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
       soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
                xmlns:xsd="http://www.w3.org/1999/XMLSchema">
  <soap:Body>
    <soap:Fault>
      <faultcode>soap:VersionMismatch</faultcode>
      <faultstring>Incorrect SOAP namespace.</faultstring>
      <faultactor>http://www.wrox.com/endpoint.asp</faultactor>
      <detail>
        <w:errorinfo xmlns:w="http://www.wrox.com">
          <desc>The SOAP namespace was blank.</desc>
        </w:errorinfo>
      </detail>
    </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

Si es pensa que el transport és "com" s'envia un missatge, llavors la funció del missatge és el "per què". HTTP és el "com" triat de moment per SOAP, i a continuació es parlarà del "per què", la RPC.

3.1.7 SOAP i RPC

Com s'ha mostrat anteriorment, SOAP no té necessàriament que utilitzar-se per a crides a procediments remots. En realitat, RPC és el motiu de l'entusiasme per SOAP de la majoria de desenvolupadors i amb raó. La idea de que les complexitats de CORBA i DCOM poden oblidar-se mitjançant una petita mesura de XML és un concepte interessant i atractiu. L'especificació SOAP descriu de forma clara la forma com les crides a procediments remots han de representar-se als missatges SOAP.

No obstant, dir que SOAP substitueix CORBA o DCOM és una simplificació excessiva, ja que a SOAP li manquen la majoria de les característiques que esperen els desenvolupadors d'un protocol d'objectes distribuïts robust, com el *garbage collector* o l'agrupació d'objectes. Tot i la 'O' a SOAP, i tal com l'especificació es presenta a l'actualitat, no existeix cap "objecte" en el sentit de DCOM o CORBA. L'especificació SOAP exposa de manera més clara que aquest no va ser un objectiu de disseny dels autors. Per altra banda, es obvi que SOAP es va dissenyar amb RPC en ment.

3.1.7.1 Convenció de SOAP RPC

Ja s'ha dit de la naturalesa oberta de l'especificació SOAP en relació al transport de missatges. SOAP defineix una sintaxi de missatge i un model d'intercanvi, però no defineix el "com" de l'intercanvi de missatges. La **convenció** és un conjunt de regles que s'apliquen a un ús en concret dels missatges SOAP. L'especificació defineix regles per a una sola convenció: les crides a procediments remots o RPC. La convenció RPC pot definir-se com un mètode de codificació de les crides a procediments remots i les respostes com a missatges SOAP.

Al igual que HTTP, SOAP RPC utilitza un model de petició-resposta per a l'intercanvi de missatges. Realitzar una crida a procediment remot amb SOAP únicament suposa la creació d'un missatge SOAP. EL missatge de petició SOAP que s'envia al punt extrem representa la crida i el missatge de resposta SOAP representa el resultat d'aquesta crida. A continuació es repassen les regles per a la creació de mètodes de crides i respostes a missatges SOAP.

3.1.7.2 La crida

Realitzar una crida a un procediment remot amb SOAP suposa la creació d'un missatge SOAP. El missatge que s'envia al punt extrem representa la crida. La càrrega útil d'aquest missatge de petició conté una estructura i és la crida del mètode codificat. Els elements secundaris de l'estructura són els paràmetres d'entrada del mètode. El resultat final és una representació XML d'una crida que té l'aspecte esperat.

En codificar les crides RPC a un missatge SOAP, el nom de l'element ha de ser igual al nom del mètode, el mateix que els paràmetres. Considerem:

```
// Return the current stock price, given the company symbol
double GetStockQuote ( [in] string sSymbol );
```

Si el mètode d'espai de nom és http://www.wroxstox.com/, llavors la crida al mètode codificat que realitza una petició seria com aquesta:

```
<q:GetStockQuote xmlns:q=" http://www.wroxstox.com/">
  <q:sSymbol xsi:type="xsd:string">OU812</q:sSymbol>
</q:GetStockQuote>
```

El nom del mètode coincideix amb l'element, igual que el paràmetre. Mentre que el nom del paràmetre coincideix amb els elements secundaris, només els paràmetres d'entrada apareixen a la crida codificada. A continuació es mostra un exemple de com tres mètodes diferents representarien una mateixa crida:

```
// Reverse the string, s, and return the new string.
string ReverseString ( [in] string s );

// Reverse the string, s, and return the new string.
void ReverseString ( [in] string s, [out] string sRev );

// Reverse the string, s, passed in by reference.
void ReverseString ( [in, out] string s );
```

Els tres mètodes es representarien per la mateixa crida serialitzada. I un exemple podria ser el següent:

```
<x:ReverseString xmlns:x="http://www.wrox.com/">
  <s csi:type="xsd:string">ROHT</s>
</x:ReverseString>
```

A continuació, s'analitzarà la forma de codificar la resposta d'una crida a procediment remot, així com l'aspecte que presenten les respostes de les diferents formes del mètode d'exemple *ReverseString()*.

3.1.7.3 La resposta

Com s'ha mencionat anteriorment, la convenció RPC utilitza un mètode de petició-resposta. Al igual que la crida es representa al missatge de petició SOAP, el resultat de la crida es retorna al missatge de resposta SOAP. Tanmateix, la càrrega útil a la resposta conté una estructura i els elements secundaris són els paràmetres de sortida i/o el valor retornat pel mètode.

El nom de l'estructura del mètode de resposta pot ser qualsevol cosa, però una convenció és l'encarregada d'adjuntar la paraula "Response" al nombre de l'estructura del mètode de la crida. Per exemple, *ReverseString* resultaria *ReverseStringResponse*. Al igual que a la crida, els noms dels elements dels paràmetres són important i han de ser iguals que els paràmetres. Si el mètode retorna un valor, el nom és irrellevant, però ha de ser el primer element secundari de l'estructura del mètode.

Quina és la diferència entre un paràmetre en particular i un valor de resposta? En cert sentit, no hi ha cap diferència. Ambdós s'encarreguen de retornar un únic valor com a part del mètode de resposta. La veritable resposta està al nom del paràmetre. Si aquest és igual al paràmetre del mètode, llavors es tracta d'un paràmetre de sortida. Els valors de resposta no poden identificar-se pel nom, només per posició, però el nom no ha d'estar en conflicte amb els paràmetres del mètode.

Una vegada conegut l'aspecte que ha de tenir una resposta codificada, a continuació se segueix amb l'exemple de *ReverseString*. Com s'ha dit anteriorment, els tres mètodes de signatures per a *ReverseString* tenen com a resultat la mateixa crida codificada. No obstant, en que es diferencien les respostes codificades? La primera versió inverteix la cadena i retorna el resultat com el valor de resposta del mètode:

```
<x:ReverseStringResponse xmlns:x="http://www.wrox.com/">
  <x:ret xsi:type="xsd:string">THOR</x:ret>
</x:ReverseStringResponse>
```

La segona versió no conté un valor de resposta, en canvi utilitza un paràmetre de sortida anomenat *sRev*:

```
<x:ReverseStringResponse xmlns:x="http://www.wrox.com/">
  <x:sRev xsi:type="xsd:string">THOR</x:sRev>
</x:ReverseStringResponse>
```

La versió final inverteix la cadena després de passar-la per referència, pel que el paràmetre *s* és tant un paràmetre d'entrada com de sortida.

```
<x:ReverseStringResponse xmlns:x="http://www.wrox.com/">
  <x:s xsi:type="xsd:string">THOR</x:s>
</x:ReverseStringResponse>
```

Com pot comprovar-se, la diferència entre els valors retornats en paràmetres i el valor del mètode es troben al nom. En el primer cas, a l'element se li diu *ret* i, en el segon, *s*. Als paràmetres s'hi accedeix mitjançant el nom o la posició com a elements de l'estructura RPC, però al valor de resposta només es pot accedir per la seva posició.

Què succeeix quan no hi ha paràmetres de sortida o valors de resposta? En aquest cas, es continua tenint una contestació que representa al mètode de resposta, però sense dades:

```
<m:GetNothing xmlns:m="http://tempuri.org/"/>
```

Un cop estudiat el funcionament de les crides a procediment remot, seguidament es mostra un exemple més complex que utilitza missatges complets i un transport.

3.1.7.4 RPC i HTTP

Fins al moment s'ha tractat el tema de RPC de manera aïllada, però per a realitzar una crida a procediment remot, és necessari trobar una manera per a traslladar el missatge a un lloc "remot". És aquí on SOAP realment destaca: en combinar RPC i HTTP per a realitzar crides als serveis web.

Per exemple, en aquest cas es necessita cridar a un procediment remot d'un servidor web per a validar una combinació de ciutat, estat i codi postal. El servei web hipotètic es troba a www.livezipcodes.com. No es coneix el mètode d'implantació, només la forma d'accedir-hi. El mètode pot cridar-se a la URL http://www.livezipcodes.com/methods/, i el *SOAPAction* per al mètode és *urn:livezipcodes*. La signatura del mètode es mostra a continuació:

string GetZipCode (string city, string state);

A la creació del missatge de petició no és necessària cap extensió, pel que l'element *Header* pot excloure's. La càrrega útil és una estructura que representa el mètode de crida. Els paràmetres del mètode es passen com a elements fill. A continuació es mostra la petició HTTP, incloent el missatge de petició SOAP enviat a www.livezipcodes.com.

```
POST /call.asp HTTP/1.1
Content-Type: text/xml
Content-Length: ###
SOAPAction: "urn:livezipcodes"

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
       soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
                xmlns:xsd="http://www.w3.org/1999/XMLSchema">
  <soap:Body>
    <m:GetZipCode xmlns:m="http://www.livezipcodes.com/methods/">
      <city xsi:type="xsd:string">Modest Town</city>
      <state xsi:type="xsd:string">Virginia</state>
    <m:GetZipCode>
  </soap:Body>
</soap:Envelope>
```

Com veritablement existeix un lloc anomenat Modest Town, Virginia, la resposta del punt extrem tindria aquest aspecte:

```
HTTP/1.1 200 OK
Content-Type: text/xml
Content-Length: ###

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
       soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
                xmlns:xsd="http://www.w3.org/1999/XMLSchema">
  <soap:Body>
    <m:GetZipCodeResponse
      xmlns:m="http://www.livezipcodes.com/methods/">
      <zip xsi:type="xsd:string">23412</zip>
    </m:GetZipCodeResponse>
  </soap:Body>
</soap:Envelope>
```

Si aquest mateix mètode s'executés, però el punt extrem no pogués accedir a la seva base de dades d'informació geogràfica, la resposta seria semblant a aquesta:

```
HTTP/1.1 500 Internal Server Error
Content-Type: text/xml
Content-Length: ###

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
       soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
                xmlns:xsd="http://www.w3.org/1999/XMLSchema">
  <soap:Body>
    <soap:Fault>
      <faultcode>soap:Server.DatabaseDown</faultcode>
      <faultstring>The database is unavailable.</faultstring>
      <faultactor>http://www.livezipcodes.com/call.asp</faultactor>
    </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

3.2 WSDL

Tot i que SOAP proporciona una manera estàndard de transportar missatges per a l'ús de serveis web, no proporciona cap manera per a descriure el format que aquests missatges haurien de tenir en un determinat servei web. Al començament de l'especificació SOAP, quan els desenvolupadors van començar a implementar serveis web de base SOAP reals, es van adonar que l'únic mode de que algú conegués la forma d'invocar un serveis web era publicant una mostra de document SOAP que il·lustrés una crida al servidor web en qüestió. El problema amb aquesta tècnica és que no descrivia la manera de cridar al servei web.

WSDL (*Web Services Definition Language*) va sorgir dels esforços de Microsoft i IBM per a pal·liar aquest problema. WSDL busca la definició i descripció de serveis web, al igual que els arxiu d'encapçalament (o llibreries de tipus) defineixen i descriuen les llibreries binàries tradicionals.

3.2.1 Repàs a la història de WSDL

Els dos primers esforços per a la construcció d'una eina per a treballar amb SOAP venen de dos companyies: IBM i Microsoft. IBM desenvolupant eines SOAP per a Java, i Microsoft amb l'eina per a treballar amb SOAP de Visual Studio.

La solució de IBM va ser NASSL, el llenguatge *Network Accessibility Service Specification Language*, que es va llençar juntament amb la primera versió del conjunt d'eines SOAP per a Java. NASSL utilitza XML a la descripció d'interfícies per a serveis web i la manera com s'enllacen a adreces web específiques (URL). Tanmateix utilitza el llenguatge d'esquema W3CXML (XSD) per a descriure els tipus de dades que es passen cap a i des de els serveis web.

Al mateix temps (primavera del 2000) Microsoft va llençar SCL, el llenguatge *Service Contract Language*, també juntament amb el seu SOAP Toolkit per a Visual Studio. Al igual que NASSL, SCL fa ús de XML per a la descripció dels serveis web, però utilitza el format *XML Data Reduced* (XDR) de Microsoft per a definir els tipus de dades. Al poc temps, Microsoft va comercialitzar una versió actualitzada anomenada SDL, *Services*

Description Language, que es va llençar amb la primera pre-estrena pública de Visual Studio .NET.

No obstant, al poc temps d'aquests llançaments, la comunitat de desenvolupadors va detectar un greu problema. L'objectiu principal de SOAP i els serveis web era el de permetre la comunicació entre plataformes i les dues implementacions principals de SOAP eren incompatibles.

La solució a aquest problema va ser W SDL. IBM i Microsoft, juntament amb Ariba, van desenvolupar la primera versió de WSDL, que va unir totes les idees precedents dels seus respectius treballs a la definició dels serveis web. WSDL se centra en estàndards oberts, pel que utilitza XSD (que actualment és una recomanació de W3C completa a http://www.w3.org/XML/Schema) per a descriure els seus tipus i altres estàndards com SOAP i MIME per a la descripció d'enllaços específics. WSDL 1.1 es va presentar com Apunt al W3C al març del 2001. Actualment, la darrera versió de l'especificació es pot trobar a http://www.w3.org/TR/wsdl.

3.2.2 La sintaxi WSDL

WSDL proporciona una gramàtica per a la descripció de serveis com un conjunt de punts finals que intercanvien missatges. Un dels conceptes principals darrera d'aquestes descripcions és la idea de que hi ha una diferència entre la definició abstracta d'un missatge i la manera concreta com als missatges se'ls hi assigna un transport específic i/o protocol de codificació de dades.

Un document WSDL està compost per definicions. Aquestes defineixen un servei com un conjunt d'una o més xarxes de punts extrems o ports. Cada port està associat a un enllaç específic que s'encarrega de definir la manera com un conjunt abstracte d'operacions i missatges està enllaçat a un port d'acord un protocol específic. Un enllaç assigna un protocol específic a un tipus de port. El tipus de port està compost per una o més operacions, que representen un conjunt abstracte de possibilitats que el servei pot "realitzar". Cada operació està composta per un conjunt de missatges abstractes que representen les dades comunicades durant l'operació. Cada missatge conté una o més parts de dades, que es defineixen com a tipus.

Cada document WSDL està format pels següents elements:

- Tipus
- Missatges
- Operacions
- Tipus de ports
- Enllaços
- Ports
- Serveis

Cada definició abstracta d'un tipus, missatge o operació pot estar enllaçada a més d'un port en concret, de manera que el mateix missatge pot enviar-se a diferents ports utilitzant una codificació diferent.

WSDL està dissenyat per a ser extensible en gran mesura i, en teoria, qualsevol protocol pot definir-se i utilitzar-se per a l'enllaç de missatges. No obstant, només existeixen tres

enllaços als formats/transports de missatges que es defineixen com a part de l'especificació de WSDL 1.1:

- SOAP 1.1
- HTTP GET/POST
- MIME

Tanmateix, WSDL no està enllaçat a un llenguatge esquema específic per a la definició de tipus; pot utilitzar-se qualsevol forma per a descriure tipus, encara que per defecte és XSD.

3.2.3 Un exemple complet de WSDL

L'exemple que es presenta a continuació servirà per a explicar més detalladament les diferents parts d'un document WSDL. Com a exemple, defineix un servei web que conté una operació anomenada *GetCurrentTemperature*. *GetCurrentTemperature* demana un codi de ciutat de tres lletres i informa sobre la temperatura actual d'aquesta ciutat. Degut a que el servei està compost per tipus, missatges, etc., els quals s'afegeixen a la definició que els precedeix, el document comença com a tipus i termina com a servei al final del document. Això fa que resulti més senzill observar la jerarquia de la composició a WSDL. Cada part component es defineix per a després afegir-se al conjunt, concloent a la definició del servei final. A l'exemple s'utilitza SOAP 1.1 com a estil de codificació i l'enllaç més comú que és HTTP.

Com a document XML que és, el document WSDL comença amb:

```
<?xml version="1.0"?>
```

3.2.3.1 Espais de nom

Primer l'element *definitions* del nivell superior:

```
<definitions name="CurrentTemp"
             targetNamespace="http://example.com/currenttemp.wsdl"
             xmlns:tns="http://example.com/currenttemp.wsdl"
             xmlns:xsd1="http://example.com/currenttemp.wsdl"
             xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
             xmlns="http://schemas.xmlsoap.org/wsdl/">
```

La següent taula mostra un conjunt d'espais de noms utilitzats a WSDL i els prefixes utilitzats normalment. Altres espais de noms poden utilitzar-se per a enllaços addicionals, com SMTP, però depenen de l'autor del document i no es defineixen a l'especificació.

Prefix	Espai de nom URI	Definició
Wsdl	http://schemas.xmlsoap.org/wsdl/	Espai de nom WSDL per al marc WSDL.
Soap	http://schemas.xmlsoap.org/wsdl/soap/	Espai de nom WSDL per a enllaços WSDL SOAP.
http	http://schemas.xmlsoap.org/wsdl/http/	Espai de nom WSDL per a HTTP GET/POST.

mime	http://schemas.xmlsoap.org/wsdl/mime/	Espai de nom WSDL per a enllaços WSDL MIME
Soapenc	http://schemas.xmlsoap.org/soap/encoding/	Codificació d'espais de nom dins de la definició SOAP 1.1.
Soapenv	http://schemas.xmlsoap.org/soap/envelope/	Sobre d'espais de nom dins de la definició SOAP 1.1.
Xsi	http://www.w3.org/2001/XMLSchema-instance	Instància de l'espai de nom dins de la definició XSI.
Xsd	http://www.w3.org/2001/XMLSchema	Instància de l'espai de nom dins de la definició XSD.
Tns	El document d'espai de nom actual	El prefix tns (*this namespace*) s'utilitza com a convenció per a referir-se al document actual.

3.2.3.2 documentation

Existeix un element específic anomenat *wsdl:documentation* que conté documentació de lectura comprensible per a l'usuari i que pot incloure's dins de qualsevol element del llenguatge WSDL. El contingut de l'element *documentation* pot "barrejar-se" (text i elements XML) i s'utilitza per a comentaris dins d'un document WSDL, utilitzant-se en substitució a l'etiqueta de comentari XML (<!- ->), ja que, per exemple, els analitzadors sintàctics específic de l'aplicació poden voler utilitzar els elements *documentation* per a generar especificacions comprensibles per a l'usuari per als documents WSDL.

```
<documentation>
  the documentation element can have mixed <content/>
</documentation>
```

3.2.3.3 types

A continuació, els tipus de dades individuals que composen el missatge:

```
<types>
  <schema targetNamespace="http://example.com/currenttemp.wsdl"
          xmlns="http://www.w3.org/200/10/XMLSchema">
    <element name="CurrentTemperatureRequest">
      <complexType>
        <all>
          <element name="cityCode" type="string"/>
        </all>
      </complexType>
    </element>
    <element name="CurrentTemperature">
      <complexType>
        <all>
          <element name="temp" type="float"/>
        </all>
      </complexType>
    </element>
  </schema>
```

```
</types>
```

L'element *types* s'utilitza per a descriure els tipus o estructures de les dades contingudes als missatges. En teoria, qualsevol llenguatge d'esquema pot utilitzar-se dins de l'element *types* per a descriure el format d'un missatge, encara que d'acord amb l'especificació W3C, "WSDL opta per l'ús de XSD com el sistema de tipus canònic i el considera com el sistema de tipus intrínsec", és a dir, s'assumeix que les definicions de tipus utilitzen XSD, excepte els casos en els que no s'especifiqui el contrari.

Les estructures de dades descrites per l'element *types* es caracteritzen per ser tipus abstractes; són representacions del format conceptual de dades d'un missatge, no el format de cable. El format de cable o implantació es determina pel protocol que s'utilitza per a l'enllaç.

3.2.3.4 Missatges

Seguidament, aquelles dades que estan associades amb els elements *message*:

```
<message name="GetCurrentTemperatureInput">
  <part name="body" element="xsd1:CurrentTemperatureRequest"/>
</message>
<message name="GetCurrentTemperatureOutput">
  <part name="body" element="xsd1:CurrentTemperature"/>
</message>
```

Una vegada que es defineix un conjunt de tipus, aquest poden composar-se en missatges. Cada definició de missatge (és important que els missatges són abstractes, no missatges de format de cable concrets) es composa d'una o més parts lògiques, associades a tipus, com defineix el sistema de tipus en ús. Dit d'una altra manera, un element *message* conté elements *part* i cada *part* fa referència a un tipus definit dins de l'element *types*.

L'objectiu dels elements *part* és el de servir com a separació lògica a les dades dins d'un sol missatge.

3.2.3.5 Tipus de ports

Els elements *message* s'agrupen en operacions d'elements i els elements *operation* s'agrupen per a formar un element *portType*.

```
<portType name="CurrentTemperaturePortType">
  <operation name="GetCurrentTemperature">
    <input message="tns:GetCurrentTemperatureInput"/>
    <output message="tns:GetCurrentTemperatureOutput"/>
  </operation>
</portType>
```

Un tipus de port (*portType*) és un conjunt de missatges agrupats en operacions. Les operacions representen una sola unitat de treball per al servei que descriu. Un tipus de port pot estar compost per més d'una operació.

Cada operació pot estar composta per missatges *input*, *output* i *fault*. El fet que aquests tres missatge estiguin o no presents depèn del tipus d'operació que utilitzi. Resulta útil pensar en una operació com a un mètode. Els mètodes tenen [in] (paràmetres d'entrada), [out] (paràmetres de sortida) i errors potencials (excepcions). Les operacions

estan dissenyades de la mateixa manera i utilitzen *input* com a dades que s'introdueixen a l'operació, *output* com a dades que es passen des de l'operació, i l'opció *fault* com a dades que es passen per a indicar una excepció.

3.2.3.6 Enllaços

Un element *binding* assigna els elements *operation* a un protocol específic mitjançant un element *portType* (remès per l'atribut *type*).

```
<binding name="CurrentTemperatureSoapBinding"
         type="tns:CurrentTemperaturePortType">
  <soap:binding style="document"
                transport="http://schemas.xmlsoap.org/soap/http/"/>
  <operation name="GetCurrentTemperature">
    <soap:operation
      soapAction="http://example.com/GetCurrentTemperature"/>
    <input>
      <soap:body use="literal"/>
    </input>
    <output>
      <soap:body use="literal"/>
    </output>
  </operation>
</binding>
```

Un enllaç defineix la manera com una operació en concret s'enllaça a un protocol específic. Cada operació s'assigna a un protocol en particular utilitzant una sintaxi específica d'un protocol.

La sintaxi d'un enllaç és semblant a la d'un tipus de port, però cada operació s'assigna a una operació específica d'un protocol.

Degut a que una operació pot enllaçar-se a més d'un protocol i que cada un dels múltiples ports de WSDL poden utilitzar cada enllaç, cada enllaç ha d'especificar un sol protocol i no pot especificar una adreça.

3.2.3.7 Ports i serveis

Finalment, el servei està compost per elements *port*, que associen una adreça o punt extrem amb un enllaç específic.

```
<service name="CurrentTemperatureService">
  <documentation>My first service</documentation>
  <portName="CurrentTemperaturePort"
    binding="tns:CurrentTemperatureSoapBinding">
    <soap:address location="http://example.com/currenttemp/"/>
  </port>
</service>

</definitions>
```

Un port associa un enllaç a una adreça d'un protocol específic. Això representa els punts extrems d'una xarxa per la que es comunica un servei. Un servei representa una col·lecció de ports com a una unitat lògica.

Cada port especifica un enllaç, que al seu torn es refereix a un element *binding* definit amb anterioritat i a una única adreça. La manera com es determina l'adreça és específica d'un protocol.

Un sol servei pot contenir diversos ports que utilitzin el mateix *portType*, però poden tenir diferents enllaços i/o adreces. En aquest cas, aquests ports són alternatius, de manera que l'usuari del servei pot triar el protocol que desitja utilitzar per a comunicar-se amb el servei (possiblement mitjançant programes en llegir els espais de nom als enllaços) o l'adreça més propera. Per exemple, un sol servei pot contenir tres ports que utilitzin el mateix tipus de port (per exemple *GetCurrentTemperaturePortType*) però estar enllaçat a SOAP per HTTP, SOAP per SMTP i HTTP GET/POST respectivament. Això oferiria a l'usuari del servei una sèrie d'opcions de protocols per a l'ús del servei.

3.2.4 Transmissions primitives

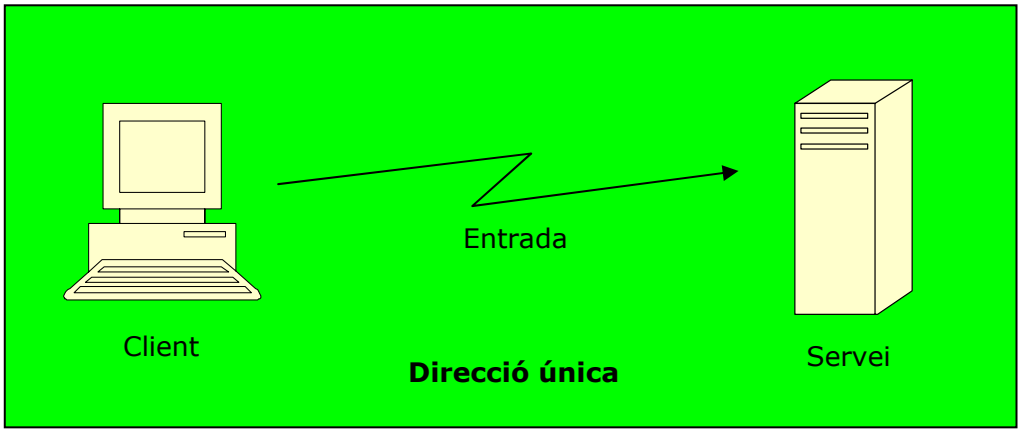
WSDL defineix quatre tipus bàsics d'operacions anomenades transmissions primitives. Aquests "estils" d'operacions representen les pautes d'ús més comuns per als serveis web. Degut a que cada operació definida per WSDL pot tenir una entrada i/o una sortida, les quatre transmissions primitives representen les possibles combinacions d'entrades i sortides. Les primitives són:

- De direcció única
- Petició/Resposta
- Sol·licitud/Resposta
- Notificació

Tot i que els quatre tipus d'operació estan descrits a l'especificació WSDL 1.1, aquesta només inclou els enllaços per a operacions de direcció única i de petició/resposta.

3.2.4.1 Operació de direcció única

Una operació de direcció única és aquella en la que el punt extrem del servei rep un missatge, però no envia una resposta. Un exemple d'això és una operació que representa la presentació d'una comanda a un sistema de compra al qual no s'espera una contestació immediata una vegada la comanda es enviada. En un context d'estil RPC, una operació de direcció única representa una subrutina a la qual no es dóna un valor de resposta.

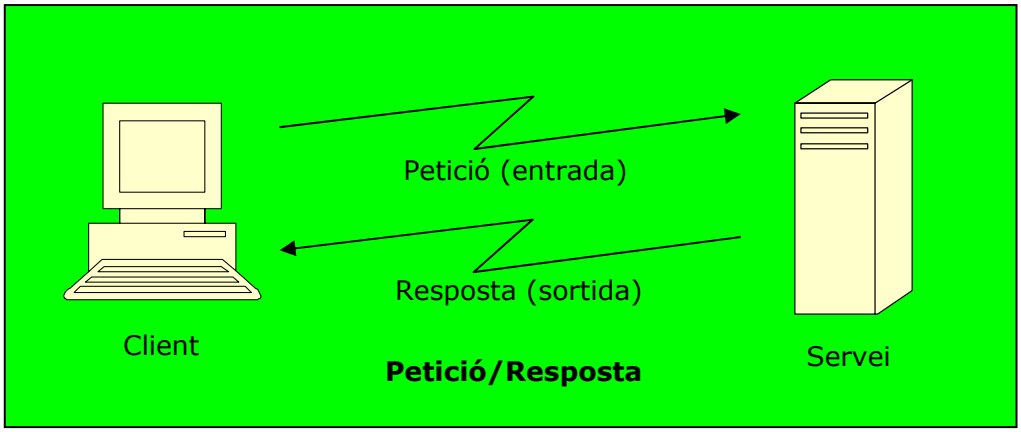


Una operació de direcció única defineix un missatge d'entrada, però no un missatge de sortida ni una fallada. Això significa que no hi ha cap manera de comprovar si hi ha hagut un error en un missatge de direcció única o a les dades retornades.

Una possible manera de resoldre una situació de dades d'error absents és mitjançant l'establiment de dos missatges de direcció única per a reemplaçar un parell de petició/resposta.

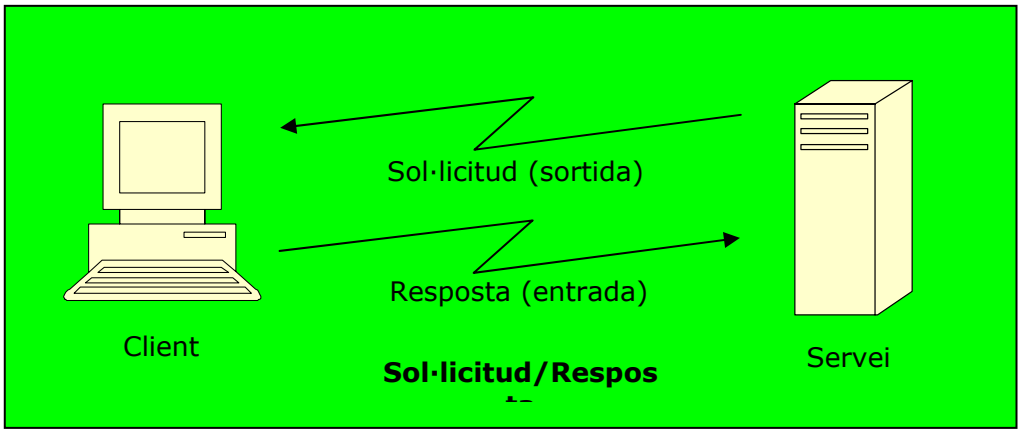
3.2.4.2 Operació de petició/resposta

Una operació de petició/resposta és aquella en la que el punt extrem rep un missatge i retorna un missatge de contestació. En un context d'estil RPC aquesta és equivalent a una crida a una funció, la qual rep una llista de paràmetres i retorna un valor. Degut a que l'operació de petició/resposta ha de respondre amb un valor, pot tanmateix retornar una fallada si existeix un error.



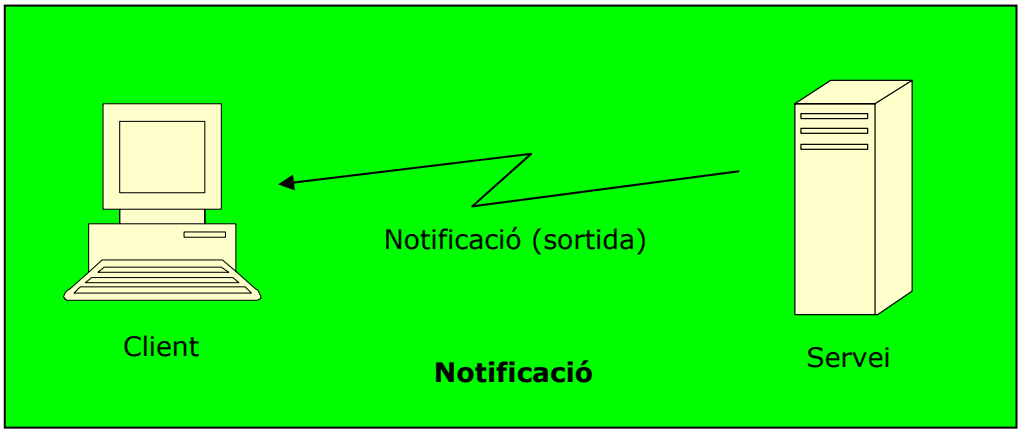
3.2.4.3 Operació de sol·licitud/resposta

Una operació de sol·licitud/resposta és aquella en la que el punt extrem del servei envia un missatge i rep una contestació en resposta. Aquesta operació és la contraria a la de petició/resposta, donat que el servei inicia l'operació (sol·licitant al client), enlloc de respondre a una petició.



3.2.4.4 Operació de notificació

Una operació de notificació és aquella en la que el punt extrem envia un missatge i no rep cap resposta. Un exemple d'això és el d'un model al qual s'informa sobre events i al qual un punt extrem informa del seu estat de forma periòdica. En aquest cas no es requereix cap resposta ja que les dades de l'estat s'emmagatzemen i no reben acció immediata.



3.3 UDDI

Un servei web és una aplicació o component independent que té les característiques següents, es pot descriure en un llenguatge de descripció de serveis i aquesta descripció es pot publicar. Els programes clients poden trobar aquestes descripcions, unir-se al servei descrit i finalment invocar els serveis.

UDDI (*Universal Description, Discovery and Integration*), Descripció Universal, Descobriment i Integració, proporciona la peça de registre a la pila dels serveis web. Això activa les tres funcions bàsiques dels serveis web, que són "publicar, trobar i unir".

- La funció **publicar** té a veure amb el mode com el proveïdor d'un servei web es registra a ell mateix i als seus serveis.

- La funció **trobar** tracta el mode com l'aplicació client troba la descripció d'un servei web o proveïdor de serveis.

- Finalment, **unir**, tracta el mode com una aplicació client es connecta i interactua amb el servei web després de trobar-lo.

UDDI defineix els criteris per als registres basats en la web. Les companyies poden registrar la informació en aquests registres sobre elles mateixes, sobre els serveis que ofereixen i la informació tècnica sobre com es pot accedir a aquests serveis. També poden buscar el registre d'altres companyies o proveïdors de servei (és a dir: buscar per la informació de servei a més de la informació de la companyia). Aquest registres poden ser d'un ordinador privat o el registre global *UDDI Business Registry*.

3.3.1 Fonaments per a UDDI

Tres empreses, IBM, Microsoft i Ariba van començar amb la iniciativa UDDI. El seu objectiu era definir criteris per a permetre que les empreses es descobrissin les unes a les altres, que interactuessin i compartissin informació al registre global.

Es pretén basar UDDI en estàndards i que s'utilitzi entre plataformes. Es construeix sobre estàndards existents d'Internet de XML, HTTP i DNS (*Domain Name System*). Des d'aleshores, altres companyies s'ha compromès a recolzar aquest criteri. La web UDDI.org ofereix un llistat d'aquestes empreses.

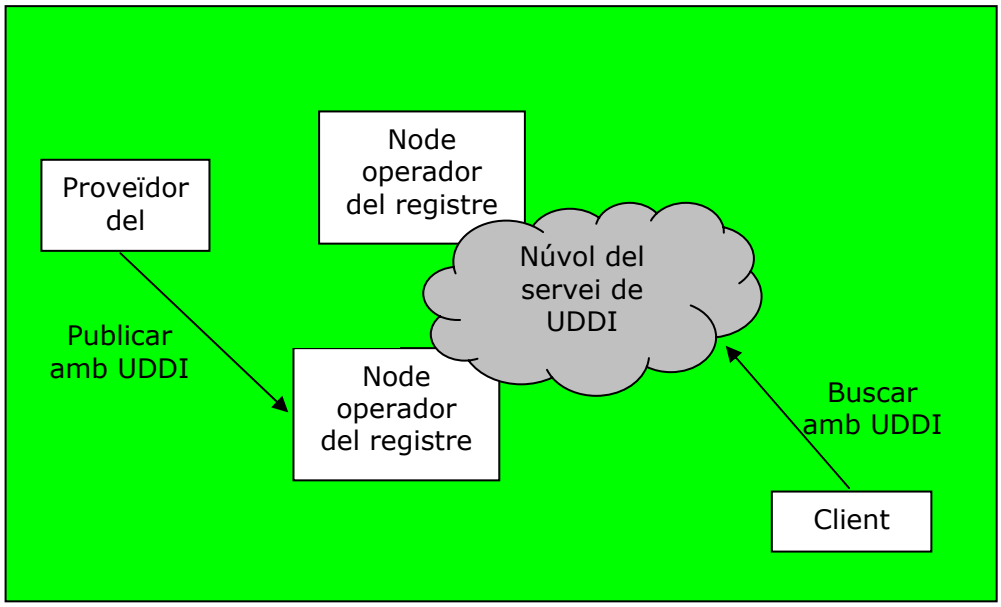
UDDI es va dissenyar per a solucionar els problemes a les interaccions empresa-empresa (B2B), proporciona un mecanisme per a realitzar una "cerca intel·ligent" de serveis web i permetre una agregació més senzilla dels serveis.

UDDI utilitza SOAP com a mitjà de transport.

3.3.2 Conceptes de UDDI

La versió 2 va aparèixer al juny del 2001, però la majoria de les implementacions encara són compatibles amb la versió anterior. El registre global de les empreses, *UDDI Global Business Registry* està en funcionament i en un principi només el van "acollir" les tres companyies fundadores. Des d'aleshores, Ariba s'ha quedat fora i Hewlett Packard va signar per a ser un operador UDDI. Les companyies que distribueixen el registre global de UDDI se'n diuen **operadors** UDDI. Aquests operadors necessiten seguir un procés definit per a manegar la informació del directori, la replicació, etc. Això s'especifica més formalment a la versió 2 de UDDI, i es pot trobar més informació a http://www.uddi.org/.

Aquests operadors de UDDI són lliures i les empreses poden publicar la seva informació i els detalls sobre els seus serveis. Una empresa no necessita registrar-se amb cada servei per separat, pot registrar-se com una de les companyies operadores, també anomenades **nodes operadors**. El registre funciona amb el principi "un cop registrat, publicat a tot arreu", és a dir, qualsevol empresa que es registra a un dels registres, tindrà la seva informació replicada a la resta de registres en un temps determinat. Aquest és el concepte del **núvol de servei** de UDDI que es mostra al següent diagrama que és, en essència, un registre replicat. Un client que busqui una empresa o un servei, ho pot fer a qualsevol dels operadors de registre, aquest han d'aconseguir la mateixa informació. Pot haver un retard en propagar la informació entre els nodes operadors, llavors, si un proveïdor de serveis actualitza una informació al node operador on estan registrats, ha de passar un petit lapse de temps abans de que els clients que busquin a d'altres nodes del registre puguin veure-ho.



Si una empresa necessita canviar cap informació que hagi publicat, tindrà que fer-ho amb el mateix operador al que es va registrar en un principi. Aquest procediment simplifica la seguretat al protocol de UDDI, ja que preveu els canvis conflictius als diferents nodes operadors. D'algun mode, el registre UDDI interpreta un paper semblant al del *Domain Name System* (DNS), manté una base de dades de serveis i de proveïdors de servei similar ala informació que manté DNS sobre noms de domini. No obstant, a diferència de DNS, en aquest cas la informació es replica a tots els nodes.

Resumint, el mode com funciona el *Business Registry* de UDDI és el següent:

1. Les entitats d'estàndards omplen el registre UDDI amb descripcions de diferents tipus de serveis.

2. Les empreses (proveïdores de serveis) omplen el registre amb les descripcions dels serveis que ofereixen.

3. El *Business Registry* de UDDI assigna uns identificadors únics a cada registre de serveis o empreses.

4. Tant els mercats virtuals, els motors de cerca i les aplicacions de negocis naveguen pel registre per a descobrir serveis.

5. Les empreses utilitzaran aquestes dades per a invocar els mètodes dels serveis remots i integrar-los a la web.

Els serveis d'una companyia es poden classificar utilitzant taxonomies estàndards industrials, del tipus codis NAICS (*North American Industrial Classification System*), UNSPSC (*UNiversal Standard Products ans Services Classification*) o SIC (*Standard Industrial Classification*). Aquestes taxonomies s'utilitzen per a classificar les categories de les empreses, indústries i productes, i defineixen els codis de categories per a les empreses i els criteris preexistents definits per la indústria. A més d'aquests, es poden utilitzar les taxonomies geogràfiques com ISO 3166 o GeoWeb.

3.4 Especificacions sobre workflow

3.4.1 Processos de negoci

Un procés de negoci pot ser definit com un conjunt de tasques interrelacionades associades a un activitat que engloba límits funcionals. Els processos de negoci tenen punts inicials i punts finals, i es poden anar repetint. Aquesta definició no revela el pensament, la claredat, el detall, i la inversió de temps requerits per a produir processos de negoci útils. Els processos de negoci útils generen i estalvien diners a les empreses.

Més important, el veritable valor de crear processos de negoci per a una empresa és el valor intel·lectual que aquests processos representen. Les eines que produeix un procés de negoci tenen un valor, però el coneixement de com fer aquestes eines també el té. Aquest coneixement pot ésser capturat, afegit i millorat a un procés de negoci. La llibertat del procés de creació d'eines és important perquè si es segueixen tots els passos s'asseguren eines de qualitat; fent més, o menys, o diferents passos resulten costos més elevats, qualitat més baixa, o totes dues coses. Valen la pena el temps i l'esforç invertits en definir i documentar processos de negoci.

3.4.2 Terminologia, definicions i orígens

Anem a definir alguns termes generals sobre *workflow* (basades en el model de referència de la WfMC, *Workflow Management Coalition*).

Workflow	De forma simple, és el camí de treball de principi a fi. Un *workflow* consisteix en la lògica de procés i les regles d'enrutament. La lògica de procés defineix la seqüència de tasques i les regles d'enrutament que s'han de seguir.
Definició d'un procés	Una definició gràfica, o mapa de procés, representa els elements de la lògica de procés d'un *workflow* i les seves relacions.
Instància d'un procés	Una instància, anomenada normalment 'feina' és una instància en execució de la definició d'un procés.
Sistema de gestió de *workflow*	Una aplicació software que emmagatzema definicions de processos i executa feines basades en aquestes definicions mitjançant el seu component motor de *workflow*. El motor de *workflow* és el mòdul d'execució.
Eina de definició d'un procés	Una eina software per a crear i modificar definicions de processos. Pot formar part d'un software de gestió de processos de negoci, una aplicació aïllada o un component d'un sistema de gestió de *workflow*.
Participant	Un dels següents tipus: un conjunt de recursos, un recurs específic, una unitat d'organització, un rol (funció d'una persona dins d'una organització), una persona, o un sistema (agent automàtic). Respostes a la pregunta "Qui?".
Activitat	Una tasca que forma un pas lògic a la definició d'un procés. Pot ser automàtica o manual. L'automatització fa referència a la capacitat de definir *scripts* o *triggers* durant l'operació d'un procés.
Propietari d'una	És el participant que pot declarar una activitat com a

activitat	completada, redireccionant el treball a la següent activitat del procés.
Propietari d'una feina	És el participant que té el control total d'execució de la instància d'un procés.
Element de treball	Representa el treball a ser realitzat per un participant a un activitat a una instància d'un procés.

La orientació del *workflow* dins del software prové de dos punts de vista diferents: processos de negoci basats en les persones, i processos d'automatització basats en regles; i els dos es complementen cada vegada més.

Els orígens del software de *workflow* basat en persones es troba a les eines de treball en grup i el "*groupware*". A les aplicacions i eines de treball en grup ('*suites*' com Lotus SmartSuite, Microsoft Office i StarOffice), la col·laboració de l'equip i el *workflow* implícit han estat sempre característiques importants. El "*groupware*" és software que expressament fa més senzill a la gent dins d'equips o de grups el col·laborar, i fa el seu *workflow* més senzill i eficient. El futur d'aquest software de *workflow* basat en la gent està en la millora de les capacitats dels serveis web.

Els orígens de les aplicacions d'automatització del *workflow* es pot trobar en les aplicacions amb motors de regles i la regla estàtica del "pas a pas", i l'automatització basada en regles dels processos de producció i fabricació. Aquest tipus de *workflow* s'està dirigint també per a suportar *workflow* basat en persones.

3.4.3 Workflow i EAI

El software de *workflow* ha de complir aquestes quatre funcions primàries com a part del *Enterprise Application Integration* (EAI): actuar com a component a les aplicacions verticals, treballar bé amb el software d'integració d'aplicacions, ser l'adhesiu per a aplicacions col·laboratives, i adequar-se a les arquitectures de serveis web.

Com a component a les aplicacions verticals per a empreses i indústries com la banca o les assegurances, una aplicació de *workflow* ha d'incrementar la interacció entre el desenvolupament i la línia de negoci mitjançant eines gràfiques intuïtives compartides entre les organitzacions. Per a indústries com la fabricació, les aplicacions de *workflow* han de millorar la flexibilitat de la producció i permetre sistemes de producció amb balanceig de càrrega. L'aplicació de *workflow* ha de tenir la capacitat d'actualitzar *workflows* fàcilment quan els processos i les organitzacions canvien. Per últim, però no menys important, una aplicació de *workflow* ha de permetre a l'empresa "adherir-se" a les regulacions governamentals i organitzatives estandarditzant i monitoritzant els processos de negoci.

Per a **treballar correctament amb el software d'integració d'aplicacions** i les seves APIs, una aplicació de *workflow* ha de proveir un suport flexible a Java que permeti a aquestes aplicacions ésser integrades amb aplicacions web i d'altres aplicacions de les TI, i ha de oferir suport també a la integració amb aplicacions de negoci existents. Per exemple, l'aplicació de *workflow* ha de ser capaç de suportar un *workflow* que es troba dins un sistema extern de *workflow*.

Les aplicacions de *workflow* han de ser l'adhesiu per a les aplicacions col·laboratives des dels dos punts de vista mencionats anteriorment. Les aplicacions col·laboratives sovint fan referència a aplicacions que estan fetes a partir d'altres aplicacions o serveis web requerits per a realitzar tasques i dirigir tota la interacció i el

flux de dades. Les aplicacions col·laboratives poden també significar l'automatització i la dinamització dels processos de negoci basats en les persones, de manera que les persones involucrades al *workflow* siguin més productives individualment, i més significativament, com equip.

Les aplicacions de *workflow* han d'adequar-se a les arquitectures dels serveis web. Els *workflows* es poden fer disponibles en forma de serveis web. Per exemple, un servei web de *workflow* pot ésser invocat per un venedor de fora enviant una petició per a rebre un pressupost. Per sobre de la creació de la petició, una aplicació de *workflow* podria enrutar aquesta petició a la persona apropiada, després automàticament crear i afegir *links* a contractes previs amb qui realitza la petició emmagatzemats a alguna base de dades, i generant pressupostos recomanats basats en aquests contractes anteriors i les condicions actuals de mercat. Quan la proposta és rebuda per la persona que ha d'omplir la petició, tot el necessari està disponible per a prendre una decisió informada sobre el preu per al pressupost del venedor. Un *workflow* pot també controlar el flux d'un conjunt de serveis web que formen una aplicació.

3.4.4 Especificacions

A les aplicacions col·laboratives fetes amb serveis web, on els processos de negoci són realment conjunts de tasques on participen serveis web, el control del *workflow* és molt important i la interacció entre *workflows* diferents és inevitable. Abans de que un *workflow* pugui cridar un altre i puguin interoperar, es necessita un estàndard que descrigui processos públics, la composició, *workflows* privats, i altres elements comuns sobre *workflow*. Aquests estàndards d'interoperabilitat estan encara per decidir, encara que hi ha alguns estàndards proposats. A continuació es mostra una taula amb diverses especificacions i definicions de *workflow* que han aparegut en els darrers temps.

Wf-XML	Wf-XML i el Model de Referència de *Workflow* del *Workflow Management Coalition* (WfMC): Wf-XML és una codificació basada en XML de missatges per a la interoperabilitat de *workflow*. El Model de Referència de *Workflow* és una descripció de l'arquitectura subjacent al sistema de *workflow*. Wf-XML no està vinculat a SOAP ni WSDL de moment.
WSFL	*IBM Web Services Flow Language*: especifica dos tipus de composicions amb serveis web, 1) un procés de negoci executable conegut com *flowModel*, i 2) una col·laboració de negoci coneguda com *globalModel*. Compatible amb SOAP, UDDI i WSDL.
XLANG	XLANG de Microsoft: llenguatge de modelat de negocis per a BizTalk, el qual és un component de .NET que permet EAI. *BizTalk Orchestration* és el motor de *workflow* i *BizTalk Orchestration Designer* és l'eina visual de modelatge de processos de negoci basada en XLANG.
BPEL4WS	*Business Process Execution Language for Web Services* és la fusió cooperativa de WSFL i XLANG per a l'orquestració dels serveis web, el *workflow*, i la composició.
ebXML BPSS	El *eBusiness Transition Working Group* porta endavant la definició de la conversa i orquestració del *workflow* a la capa *Business Process Specification Schema* (BPSS) d'ebXML, la qual defineix molts protocols i capes per a *e-business* basades en XML.
WSCI	*Web Services Choreography Interface* del consorci Sun, BEA, Intalio i

	SAP. És un llenguatge de descripció d'interfícies basat en XML que descriu el flux de missatges intercanviat per un servei web participant d'interaccions amb altres serveis.
WSCL	*Web Services Conversation Language* del W3C: una proposta de HP al W3C. Permet definir les interfícies abstractes de serveis web (això és, converses a nivell de negoci o processos públics suportats per un servei web), els documents XML que s'intercanviaran, i la seqüència d'aquests documents.
PIPs	*Partner Interface Process* de RosettaNet: defineix processos de negoci entre socis comercials mitjançant diàlegs especialitzats basats en XML de sistema a sistema. Molts PIPs han estats definits per a diversos escenaris.
JDF	*Job Definition Format* del CIP4 és un recent estàndard de *workflow* per a la indústria d'arts gràfiques dissenyat per a simplificar l'intercanvi d'informació entre diferents aplicacions i sistemes.

3.4.5 BPEL4WS

L'especificació *Business Process Execution Language for Web Services* (BPEL4WS) està ben posicionada per arribar a ser l'estàndard per a la composició en els serveis web. Permet crear processos complexes creant i escrivint juntes activitats diferents que poden, per exemple, invocar a serveis web, manipular dades, llençar excepcions o acabar processos. Aquestes activitats es poden encabir dins d'activitats estructurades que defineixen com s'han d'executar, ja sigui en seqüència, en paral·lel o depenent de determinades condicions. [WeCu02]

Avui dia els serveis web es poden comunicar amb d'altres serveis web, es poden anunciar a ells mateixos, i ser descoberts i invocats utilitzant especificacions esteses àmpliament a la indústria. No obstant, fins fa un cert temps, per a enllaçar junts aquests serveis dins de processos de negoci o dins d'una composició, l'usuari havia de triar entre una sèrie d'especificacions incompatibles entre si, com WSFL d'IBM o XLANG de Microsoft. BPEL4WS representa la unió de WSFL i XLANG, i amb una mica de sort es convertirà en la base de l'estàndard per a la composició de serveis web. BPEL4WS combina el millor dels dos, WSFL (suport per a processos orientats a grafs) i XLANG (construccions estructurades de processos), en un paquet cohesionat que suporta la implementació de qualsevol procés de negoci d'una forma natural. A més de ser un llenguatge d'implementació, BPEL4WS també es pot utilitzar per a descriure les interfícies dels processos de negoci.

3.4.5.1 Conceptes

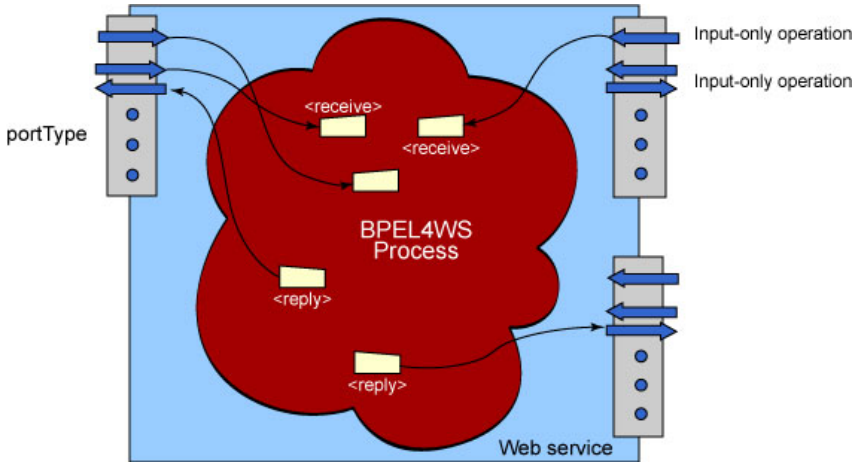
BPEL4WS pot ser utilitzat en dos escenaris diferents:

- Per a implementar processos de negoci executables
- Per a descriure processos abstractes no executables

Però habitualment serà el primer escenari el més freqüent ja que és lògic que les descripcions de processos abstractes s'acabin implementant d'alguna manera.

Com a llenguatge d'implementació de processos executables, el paper de BPEL4WS és definir un nou servei web a partir de la composició d'un conjunt de serveis existents. Així,

BPEL4WS és un llenguatge per a implementar aquesta composició. La interfície del servei compost es descriu com una col·lecció de *portTypes* de WSDL. La composició (anomenada "procés") indica com s'ajusta la interfície del servei a l'execució de la composició. A la següent figura es pot veure de forma més gràfica:



4. Conclusions

En aquest últim punt explicarem quines han estat les decisions preses respecte a quines especificacions utilitzar i quines eines.

S'ha escollit la tecnologia "Web Services" principalment per ser una de les tecnologies més prometedores dels pròxims anys. Cert es que es una tecnologia emergent, i per tant poc consolidada, però l'esforç que estan realitzant moltes empreses indica que no estem fent un treball sense recompensa. Es cert que les eines que es realitzaran en el projecte es podrien realitzar també amb un llenguatge de scripting de servidor com php, jsp, o asp però d'aquesta manera estaríem tancant la porta a factors de qualitats tal com la flexibilitat, la reusabilitat i la actualitat. El fet de fer servir Web Services permetrà deixar la porta oberta a qualsevol possible futura aportació en qualsevol projecte realitzat, i no només ho permetrà sinó ho facilitarà ja que aquesta tecnologia esta totalment orientada a un disseny modular independent i de molt fàcil adaptabilitat. Aquesta tecnologia permet també donar una gran transparència a les operacions a realitzar ja que les aplicacions es creen en una arquitectura similar a la de tres capes. D'aquesta manera en cada aplicació hi ha tres rols principals el client, el que ofereix la interfície (servidor Web) i el que ofereix el Servei. Es realment rellevant que cap rol ha de conèixer com estan realitzats els altres, ni on estan ubicats, però que de manera automàtica pot reconèixer el necessari per realitzar les operacions. Aquesta versatilitat es la principal característica que aporta la creació de serveis Web.

Pel que fa a la plataforma, nosaltres escollim Apache+Tomcat+Axis. Aquesta comunitat open-source no només ofereix una plataforma gratuïta, sinó que està considerada com una de les mes segures i fiables del mercat. A part d'aquests dos motius ens trobem en front d'una de les plataformes mes portables existents, ja que al estar completament realitzades en Java i no disposar d'interfície gràfica permeten portar tot el sistema a la major part dels sistemes operatius i plataformes actuals. I si a tot això li sumem que disposa d'una de les mes complertes documentacions i d'una gran comunitat d'usuaris i desenvolupadors amb un gran esperit col·laboratiu, llavors ens trobem amb la eina perfecta per realitzar un projecte on un dels principals objectius es aprendre.

Finalment voldria comentar que el gran marc que engloba tota l'àrea dels Web Services no pot ser tractada amb un sol projecte y que per tant la major part del treball es centrarà només en al part més tecnologia d'aquesta tecnologia ja que la part teòrica ha d'estar destinada també a altres temes que no tenen a veure amb aquesta tecnologia. Per tant els protocols de Ubicació i Cerca de serveis Web (UDDI) i els de lògica de negoci (BPEL4WS) no seran tractats durant el projecte.

5. Referències

[SRSA02] Brent Sleeper, Bill Robins, Ben Smith and John B. Anderson, "The Emerging Web Services Market", A.T. Kearney & The Stencil Group, Mar 2002, ref. http://www.stencilgroup.com/ideas_scope_200203atkws.html

[SlRo01] Brent Sleeper and Bill Robins, "Defining Web Services", The Stencil Group, Jun 2001, ref. http://www.stencilgroup.com/ideas_scope_200106wsdefined.html

[CCCD02] P. Cauldwell, R. Chawla, V. Chopra, G. Damschen, C. Dix, T. Hong, F. Norton, U. Ogbuji, G. Olander, M. A. Richman, K. Sanders y Z. Zaev, "Servicios Web XML", Anaya Multimedia / WROX, 2002

[CFNO02] Michael Champion, Chris Ferris, Eric Newcomer and David Orchad, "Web Services Architecture, W3C Working Draft", The World Wide Web Consortium (W3C), Nov 2002, ref. http://www.w3.org/TR/ws-arch/

[WeCu02] Sanjiva Weerawarana and Francisco Curbera, "Business Processes with BPEL4WS: Understanding BPEL4WS, Part I", IBM developerWorks, Aug 2002, ref. http://www-106.ibm.com/developerworks/library/ws-bpelcol1/