

Pasado, Presente y Futuro de los Sistemas de Información Distribuidos

Luis Iribarne

Departamento de Lenguajes y Computación, Universidad de Almería
Ctra. Sacramento s/n, 04120, Almería, 2001

Una de las disciplinas de la ingeniería del software que más impulso está teniendo en los últimos tiempos es aquella que describe, desarrolla y utiliza técnicas software para la construcción de sistemas abiertos y distribuidos. Un ejemplo de esto son los sistemas de información distribuidos.

La disciplina de los sistemas de información distribuidos, como tal, ha empezado a ser reconocida ampliamente desde hace relativamente poco tiempo. Es un término que surge en la década de los 90, cuando los ingenieros, encargados de desarrollar y de mantener los grandes sistemas de información de la empresa, ven la necesidad de escalar y ampliar sus sistemas para dar cobertura ya no solo al personal interno de una sección, de un departamento o de una organización, si no también para dar servicio a otros miembros de la organización ubicados en diferentes localizaciones geográficas, y como no, a otros miembros externos a la organización.

Los sistemas de información distribuidos —en ocasiones también denominados sistemas informáticos distribuidos— surgen justo en la época donde comienza el “boom de Internet”, y con ello, un cambio radical en las metodologías de desarrollo de los sistemas de información. En pocos años se pasa de una mentalidad centralizada, donde prevalecía la confidencialidad y sistemas de información basados en Intranet, a una mentalidad totalmente opuesta, descentralizada y basada en Internet. Evidentemente todo esto se ve influenciado por la caída progresiva de los equipos hardware y materiales de comunicación, lo cual, como hemos dicho, permitiría que en pocos años surgiera una multitud nueva de tecnología alrededor de una única idea: mantener sistemas de información descentralizados, distribuidos y abiertos, generalmente —si no en su totalidad, sí una parte— funcionando sobre Web.

Como vemos, el término Sistema de Información Distribuido surge como confluencia interesada de dos disciplinas elementales. Por un lado la disciplina de los sistemas de información, como idea original de un sistema centralizado en sí, y por otro lado la disciplina de los sistemas distribuidos, como idea original de un sistema descentralizado.

La disciplina de los *sistemas de información* históricamente ha estado relacionada con la disciplina que analiza, diseña, desarrolla, implanta y mantiene el sistema informático de una empresa ([28], [51], [Pre01], [101]). Esto está relacionado con todos los procesos de ingeniería de software que los profesionales del software utilizan para el desarrollo de sistemas informáticos.

La disciplina de los *sistemas distribuidos* históricamente ha estado relacionada con el paradigma de la programación distribuida, como algoritmos distribuidos, modelos para la implementación abstracta de memoria compartida distribuida, sistemas de archivos y sistemas de gestión de bases de datos distribuidos, comunicación y paso de mensajes entre procesos concurrentes, sincronización, seguridad, y tolerancia a fallos, entre otros factores ([7], [CDK01], [28], [63], [TV01]).

Como hemos dicho, el explosivo crecimiento de Internet y la enorme variedad de información disponible por la red ha dado lugar a una nueva realidad, la convergencia de estas dos disciplinas. La rápida evolución de los sistemas de computadoras y de las tecnologías de última generación tiene que estar en constante sintonía con las demandas reales de los profesionales de desarrollo de software, organizaciones y empresas.

El proceso de desarrollo de sistemas informáticos de empresa ha ido cambiando gradualmente en pocos años para pasar de un modelo centralizado y rígido hacia un modelo descentralizado, abierto y distribuido. El sistema informático o sistema de información

La disciplina SID
empezó su recorrido
en los años 90

Sistemas centralizados
frente a sistemas
descentralizados

SID = SI + SD

SI

SD

de una empresa —a nivel de recursos software, hardware y humanos— solía estar localizado en un mismo espacio geográfico, en un departamento o una sección de la empresa. Desde aquí, el equipo humano de profesionales, que tradicionalmente estaba compuesto por las categorías de analistas y programadores de sistemas, elaboraba las aplicaciones del sistema de información haciendo uso de conocimientos y prácticas tradicionales del proceso de ingeniería del software.

Mediados de los años 80: los PCs, origen de la descentralización

A mediados de los años 80 empiezan a converger diversos factores en el mundo de la informática que serían el detonante de un cambio en el proceso de ingeniería de los sistemas de información. Por un lado comienza la explosión de los PCs, que irrumpe con fuerza dentro de la empresa, básicamente en los centros de cálculo. Aunque la mayor parte de la lógica de negocio¹ aún residía en grandes estaciones de trabajo o en mainframes, la masiva presencia de equipos de bajo coste (PCs —comparados con los grandes sistemas) permitiría a los ingenieros desarrollar grandes aplicaciones desglosadas en módulos software que podían estar ubicados en distintos ordenadores, dando lugar ahora a nuevo enfoque en el desarrollo de sistemas de información.

Las grandes computadoras y SO tipo Unix, fomentan la investigación paralela y concurrente

Inicialmente, estos bloques software funcionaban como elementos de cómputo independientes dentro del sistema, pero pronto, los ingenieros vieron la necesidad de disponer nuevas técnicas para la comunicación y transferencia de datos entre estos elementos de cómputo. Precisamente por esta fecha, ajeno a estas necesidades, empezaban a consolidarse fuertes líneas de investigación en computación paralela y programación concurrente ([1], [20], [23], [69]), motivado en un principio por la masiva presencia de sistemas operativos tipo Unix en sistemas multiprocesador.

RPC

Estas líneas de investigación en programación paralela y concurrente, junto con las necesidades de comunicación entre procesos en ambientes de cómputo independientes, dieron lugar a primeros esfuerzos en la elaboración de nueva tecnología para la programación distribuida de aplicaciones ([28], [82]). Precisamente, uno de los primeros resultados fue el desarrollo de la técnica RPC (*Remote Procedure Call*), origen de gran parte de la tecnología middleware actual. Esta técnica permite que los desarrolladores de software puedan diseñar sus aplicaciones mediante módulos comunicantes, como si fuesen un conjunto de procesos cooperativos independientes.

Problema 1: falta de un modelo distribuido

Esta nueva técnica empezó a utilizarse de forma masiva en la empresa para el desarrollo de grandes sistemas de información. Pero esto provocó principalmente dos problemas. Por un lado, se empezó a echar en falta un modelo distribuido estándar que sirviera de guía para los ingenieros en la elaboración de sus aplicaciones distribuidas. Debido a la rápida utilización de la técnica RPC, se empezó a dar forma a todo un entorno de computación distribuida sin la elaboración de un marco teórico que lo sustentase.

Solución: DCE

Esto dio lugar a la aparición del primer modelo de distribución en 1994, conocido con el nombre de DCE (*Distributed Computation Environment*, [78]). Este modelo fue desarrollado por OSF (*Open Systems Foundation*), una organización formada por IBM, DEC y Hewlett-Packard. El modelo establecía las pautas y normas que los ingenieros de sistemas de información debían seguir para desarrollar sus sistemas. Entre otras características [78], el modelo DCE destacó por ser un *modelo cliente/servidor* basado en el *lenguaje C* y que inicialmente funcionaba para *plataformas Unix*². Posteriormente el modelo se extendió para soportar diversos sistemas operativos, como VMS, Windows y OS/2, entre otros.

Problema 2: Legacy Systems

Por otro lado, esta nueva mentalidad de construir aplicaciones divididas en partes comunicantes y residentes en distintos ambientes de cómputo fue un gran paso en el campo programación distribuida. Evidentemente, las antiguas aplicaciones del sistema no dejaron de funcionar, pero los ingenieros sí vieron pronto la necesidad de integrar las partes existentes del sistema con las nuevas diseñadas.

Esto dio paso a la aparición de nuevos conceptos, como *legacy systems* o sistemas

¹Software crítico, de cálculo o muy ligado a los gestores de bases de datos

²La terna *cliente/servidor + Unix + C* se puso muy de moda en esas fechas. La existencia de un modelo distribuido ya reconocido hizo que la demanda de profesionales con conocimientos en estas áreas se incrementase enormemente.

heredados, que hace referencia a la integración de partes software existentes con las del sistema actual. Otro concepto es el de *wrapper*, que son porciones de códigos especialmente diseñados para encapsular y dar funcionalidad a otras partes del sistema ya existentes; o el concepto *glue*, que son porciones de código cuyo efecto es similar al de un “pegamento” y que sirve para unir distintas partes envueltas y funcionando con wrappers.

Pero el concepto más importante que ha cambiado y sigue cambiando los procesos de ingeniería y reingeniería, es el concepto de *componente*. Inicialmente este concepto surge ante la necesidad de reutilizar partes o módulos software existentes que podían ser utilizadas para la generación de nuevas extensiones de las aplicaciones, o incluso para la generación de completas aplicaciones. Pero esto suponía un gran esfuerzo, pues había que localizar estas partes reutilizables y almacenarlas en repositorios o librerías de código especiales que más tarde podían ser consultadas en fase de diseño.

Este es uno de los puntos clave más importantes dentro de los Sistemas de Información Distribuidos (SID), pues empiezan a diferenciarse dos estilos de desarrollo de software —utilizados después en los procesos de ingeniería para la construcción de SID—. Por un lado está el desarrollo de software basado en reutilización, donde las aplicaciones se construyen a partir de otras partes software ya existentes y accesibles en repositorios conocidos. Por otro lado está el desarrollo de software para reutilización, donde se ponen en práctica procesos de ingeniería para la elaboración de eficientes partes software que luego pueden ser utilizadas para la construcción de aplicaciones (en el otro estilo de desarrollo de software). A estas partes software se las conoce como componentes software, y han dado lugar a los paradigmas de programación de componentes *top-down* (para reutilizar) y *bottom-up* (basado en reutilización).

Pero el uso generalizado de los componentes en procesos de ingeniería de software realmente empieza a tomar presencia y sentido con la aparición de nuevos modelos de distribución, como CORBA, DCOM o EJB, modelos que actualmente se están utilizando para el desarrollo de aplicaciones distribuidas. Su predecesor, el modelo DCE, empieza a ser visto por los ingenieros de sistemas como un modelo difícil y costoso de llevar a la práctica.

Por este motivo, la *Object Management Organization* (OMG) empezó a desarrollar un modelo para la distribución y localización dinámica de objetos en tiempo de ejecución, el modelo CORBA (*Common Object Request Broker Architecture*). Por otro lado, Sun Microsystems (tecnología Unix) y Microsoft (tecnología Windows) elaboran sendos modelos, conocidos como EJB (*Enterprise Java Beans*) y DCOM (*Distributed Component Object Model*), respectivamente.

Sin embargo, la presencia de distintos modelos de objetos distribuidos dentro de la empresa, cada vez más influenciada por intereses de la industria —intereses de soluciones *Sun* frente a intereses de soluciones *Microsoft*—, y la fuerte evolución de nuevas tecnologías (XML, SOAP, Servlets, seguridad, entre otros), está haciendo que los ingenieros de sistemas tengan que hacer grandes procesos de ingeniería de requisitos para seleccionar aquellas tecnologías adecuadas para el desarrollo de sus sistemas. Incluso, en la mayoría de los casos, los ingenieros se ven obligados a utilizar e incorporar múltiples métodos y técnicas para dar soporte a distintos clientes —software y humanos— del sistema de información.

Por tanto, los grandes sistemas de información de hoy día están basados en modelos cliente/servidor con arquitecturas multicapa y que hacen uso de una gran variedad de tecnologías. La tendencia actual, es que los sistemas de información de la empresa estén distribuidos y localizados en distintos lugares geográficos, comunicándose sus partes con modelos distribuidos CORBA, EJB y/o DCOM, haciendo uso de normas y técnicas de seguridad importantes, utilizando nuevas técnicas como XML para la representación intermedia de información entre componentes software, o SOAP para la localización y activación automática de servicios web, entre otras muchas nuevas tecnologías.

Esto último ha sido motivo para la consolidación del concepto *abierto*, que se utiliza cuando se habla de sistemas de información distribuidos y abiertos. El concepto

Solución: reingeniería de componentes software

Top-down vs. Bottom-up

Modelos de objetos distribuidos

La revolución industrial

Sistemas abiertos

abierto significa que el sistema de información distribuido debe ser heterogéneo y estar preparado en todo momento para sufrir cualquier modificación (proceso de mantenimiento y actualización) sin que esto altere el funcionamiento normal de ninguna parte del sistema. Pero esto está dando paso a la necesidad real de estándares y sobre todo de un metamodelo de computación distribuida global que abarque cualquier modelo distribuido.

∞ REFLEXIÓN...

La tendencia en los procesos de ingeniería del software para el desarrollo de sistemas de información distribuidos, es elaborar sistemas de información cooperativos y colaborativos, compuesto por subsistemas, componentes y objetos especializados y coordinados para ofrecer servicios. En este sentido, están empezando a distinguirse distintas subdisciplinas de la ingeniería conocidas como “ingenierías basadas” o “ingenierías orientadas”, como por ejemplo:

- *Ingeniería del software basada en componentes,*
- *Ingeniería del software basada en aspectos,*
- *Ingeniería del conocimiento,*
- *Ingeniería de requisitos,* entre otros.

Esto está obligando a la necesidad de disponer nuevas categorías de profesionales especializados en estas ingenierías de la informática. Actualmente, los profesionales adquieren estos conocimientos desde cursos de formación internos que se imparten en la propia empresa o desde masters impartidos por organizaciones que se dedican exclusivamente al desarrollo y formación en nuevas tecnologías. No es extraño, por tanto, que en pocos años podamos ofertar en la Universidad distintas ingenierías de informática, ante la fuerte demanda para los próximos años de estos profesionales desde las empresas³.

Investigación relacionada

En la línea del último comentario del apartado anterior, una de las investigaciones que, a nuestro parecer, más impacto está teniendo y tendrá en próximos años en el área de los sistemas de información distribuidos, es aquella que trabaja en la elaboración de un modelo distribuido para entornos web. En España hay varios grupos de investigación interesados en este campo de la informática. Por ejemplo, actualmente hay un proyecto de investigación que trabaja en el desarrollo de una metodología que sustente la programación distribuida bajo web, o dicho de otra forma, en la elaboración de procesos de ingeniería basados en Internet. En este proyecto trabajan grupos de investigación nacionales de las Universidades de Málaga, Valencia y Sevilla⁴, junto con grupos de investigación de países latinoamericanos, como Argentina, Brasil, Chile y Colombia.

Ingeniería del software
basada en Internet

Agentes, sistemas
multiagentes, y
computación móvil

Otro campo de investigación de interés para los sistemas de información distribuidos son los agentes, sistemas multiagente y código móvil. Resultados de esta investigación serán claves para el desarrollo de futuros sistemas de información “inteligentes”. En este caso, la computación móvil y los agentes aportan la tecnología necesaria para desarrollar sistemas de información distribuidos divididos en ambientes de cómputo con estado interno y con capacidad para tomar decisiones sin la intervención del factor humano. Estos trabajos ya fueron iniciados en 1999 por Luca Cardelli, de Microsoft ([17], [18]). En la página 7 de este informe ofrecemos referencia a la literatura en el área de la computación móvil y agentes.

Ingeniería del software
basada en
componentes
distribuidos

Otro de los campos de investigación importantes en los sistemas de información es el que cubre la ingeniería del software basada en componentes distribuidos, conocido más

³En Estados Unidos, por ejemplo, está reconocido el título de Ingeniero en Internet

⁴La Universidad de Almería colabora indirectamente en este proyecto.

por el término en inglés DCBSE (*Distributed Component-Based Software Engineering*). Este campo se preocupa, por ejemplo, de los procesos de formalización y especificación de componentes software, arquitecturas software distribuidas, interoperabilidad entre componentes software, composición y servicios web. Recientemente, en esta subdisciplina de la ingeniería del software se están desarrollando esfuerzos de investigación en el campo de los componentes comerciales COTS (*Commercial off-the-shelf*), concretamente en el área de la ingeniería de requisitos.

Seguidamente en las tablas 1 y 2 mostramos algunos de los grupos de investigación nacionales e internacionales que actualmente están trabajando en algunas de las áreas de los sistemas de información distribuidos. En la página 7 de este informe ofrecemos una lista con algunos de estos términos y/o áreas más usuales.

Grupo	Dirección
Grupo de Ingeniería del Conocimiento, Datos y del Software. Línea de investigación de Ingeniería del Software basada en Componentes Distribuidos COTS. Universidad de Almería.	http://www.cotstrader.com
Grupo de Ingeniería del Software (GISUM). Universidad de Málaga.	http://www.lcc.uma.es/gisum/
Grupo de Ingeniería del Software e Inteligencia Artificial. Universidad Complutense de Madrid.	http://bogart.sip.ucm.es/
Grupo de Ingeniería del Software. Línea de investigación de Ingeniería del Software basada en Componentes. Universidad de Murcia.	http://www.um.es/giisw/isbc/
Grupo de Ingeniería del Software y del Conocimiento. Universidad de Las Palmas de Gran Canaria.	http://rigel.dis.ulpgc.es/gisc.htm
Grupo de Ingeniería del Software. Universidad de Sevilla.	http://www.lsi.us.es/is/
Grupo de Modelización Conceptual Orientada al Objeto y Bases de Datos. Universidad de Valencia.	http://www.dsic.upv.es/users/oom/
Grupo de Reutilización y Orientación al Objeto. Universidad de Valladolid.	http://jupiter.dcs.fi.uva.es/

Cuadro 1: Grupos de investigación nacionales con algunas líneas de trabajo en Sistemas de Información Distribuidos (por orden alfabético)

Grupo	Dirección
CORBA and Distributed Systems Research Group	http://nenya.ms.mff.cuni.cz/thegroup/
Distributed Software Engineering	http://www.dse.doc.ic.ac.uk/research/
CREWS: Cooperative Requirements Engineering With Scenarios	http://sunsite.informatik.rwth-aachen.de/CREWS/
Formal Modeling of Information Systems	http://paradis.ift.ulaval.ca/projects/forspec/forspec-ang.html
ESI: European Software Institute	http://www.esi.es
OMG: Object Management Group	http://www.omg.org
RENOIR: Requirements Engineering Network	http://www.cs.ucl.ac.uk/research/renoir/
RESG: Requirements Engineering Specialist Group	http://www.cs.york.ac.uk/bcs/resg/
SEI: Software Engineering Institute	http://www.sei.cmu.edu/
Sun Microsystems	http://www.sun.com
W3C: World Wide Web Consortium	http://www.w3.org

Cuadro 2: Grupos y organizaciones internacionales conocidos por algunas de sus líneas de trabajo de investigación en Sistemas de Información Distribuidos

Recursos tecnológicos

En este apartado recogemos algunos de los recursos tecnológicos más utilizados en estos últimos años, algunos de ellos incluso, desde hace pocos meses. En este apartado

no pretendemos exponer toda la tecnología existente, como hemos dicho, sólo ofrecemos un listado con el vocabulario tecnológico que, a nuestro parecer, es frecuentemente usado cuando se trabaja con sistemas de información distribuidos.

MICROSOFT Recursos Microsoft.

- DCOM
- IIS (*Internet Information Server*). Es un servidor web con normas de seguridad.
- ASP (*Active Server Page*). Es tecnología para la programación estilo CGI.

OMG Recursos OMG (*Object Management Group*).

a) CORBA (*Common Object Request Broker Architecture*). Es el modelo de objetos distribuidos de OMG. Actualmente existen en el mercado una gran variedad de implementaciones del modelo CORBA, pero destacamos tres:

- a) Orbix y Orbacus de IONA.
- b) ObjectBroker de Bea Systems.
- c) VisiBroker de Visigenic/Borland.

De estas casas comerciales, IONA es la que mantiene implementaciones más completas del modelo de componentes CORBA. Actualmente la casa comercial IONA cubre casi el 66% de la cuota de mercado en ventas de ORB, y con más de 65 millones de euros en ventas sólo en el año 1998⁵.

SUN Recursos Sun Microsystems.

- EJB (*Enterprise Java Beans*). Es el modelo de componentes del lado servidor de Sun Microsystems.
- Servlets y JSP (*Java Server Page*). Tecnología para programación estilo CGI.
- JAXP (*Java API for XML Processing*). Soporte para la programación Java-XML del lado servidor.

W3C Recursos W3C (*World Wide Web Consortium*)

- XML (*eXtensible Markup Language*) y XMLSchemas. XML es tecnología para la modelización y representación intermedia de datos. XMLSchemas, o simplemente esquemas, es un metalenguaje que permite dar contenido semántico a las plantillas XML. Hace unos meses a esta tecnología se la conocía como DTD (*Data Type Definition*).
- XQuery. Es una especificación de un lenguaje de consulta sobre plantillas XML propuesta en 2001. Actualmente existen muy pocas implementaciones de la especificación, pero la más conocida es XQEngine, disponible en W3C.
- SOAP (*Simple Object Access Protocol*), UDDI (*Universal Description, Discovery, and Integration*) y WSDL (*Web Services Description Language*). Es tecnología de última generación para servicios web. Los servicios web son objetos o componentes que pueden ser activados, registrados y accedidos desde Internet en tiempo de ejecución por otros objetos o componentes.

⁵No hemos encontrado datos más recientes.

Términos más usados en la disciplina

A continuación ofrecemos un listado de los términos que, a nuestro parecer, están más relacionadas con la disciplina de los *Sistemas de Información Distribuidos*. Junto a estos términos ofrecemos aquellas referencias bibliográficas más relevantes. Este apartado también puede servir de ayuda al lector para facilitar la lectura bibliográfica que se ofrece al final del presente documento⁶.

Agentes : [12] [35] [38] [43] [45] [53] [59] [60] [75] [103]

Concurrencia : [1] [69]

COM/DCOM : [4] [14] [26] [81] [52] [62] [80] [85]

Componentes en general : [6] [5] [15] [?] [29] [55] [61] [65] [68] [67] [*Szy98*] [89] [90]

Coordinación : [21] [27] [30] [36] [41] [79]

CORBA : [9] [*BVD01*] [22] [26] [40] [81] [86] [95] [ZL01]

COTS : [19] [47] [48] [49] [50]

Ingeniería del Software basada en Componentes : [8] [16] [54] [58] [99]

Interoperabilidad : [64] [93] [92]

Instituciones : [32] [44] [76] [84] [97]

Métodos formales en componentes : [87] [73] [3]

Movilidad : [11] [17] [18] [37] [42] [70] [46] [72]

Objetos, objetos distribuidos : [39] [94] [56] [66] [77] [88] [83] [31] [100]

Paralelismo : [2] [20] [23] [34]

Sistemas abiertos : [89] [90]

Sistemas de información : [102] [101]

⁶No hemos catalogado todas las referencias bibliográficas, aunque prácticamente están todas ellas. En *cursiva* hemos marcado aquellas referencias a la bibliografía básica o complementaria.

Sistemas distribuidos : [7] [10] [24] [25] [28] [78] [51] [57] [63] [74] [82] [96]

SOAP : [*Cho01*] [98]

XML : [13] [33] [*Gla01*] [*MTU00*]

Miscelánea en Internet

La finalidad de este apartado es, simplemente, mostrar al lector un repertorio de utilidades, servicios y sitios webs que, a nuestro parecer, están estrechamente relacionados con la disciplina de los Sistemas de Información Distribuidos. Evidentemente este repertorio de utilidades es incompleto y puede quedar obsoleto en cuestión de pocos meses, o años. No obstante, el lector podrá encontrar esta lista disponible en el sitio web <http://www.ual.es/personal/liribarn/LSID>, donde la lista es actualizada y mantenida con regularidad.

Componentes Software	
MTS vs EJB	http://members.tripod.com/gsraj/misc/ejbmts/ejbmtscomp.html
DCOM,CORBA,Java-RMI - A Comparison	http://www.execpc.com/gopalan/misc/compare.html
ARPA Project Directory	http://www.cis.ohio-state.edu/lair/Projects/ARPA/arpa.html
Component Based Development Forum	http://www.butlerforums.com/cbdforum/news/frame.htm
Component Development Strategies	http://www.cutter.com/cds/index.html
Component Technology	http://www.odateam.com/cop/
Component Strategies Online	http://www.componentmag.com/html/from_pages/feature.shtml
Component Design Research Study	http://www.sba.uwm.edu/components/
Component-Based Development	http://www.users.globalnet.co.uk/rxv/CBDmain/index.htm
Component Oriented Middleware	http://www.objectwatch.com/magart8.htm
Component Software Resources	http://www.cs.virginia.edu/mvm3k/resources.html
ComponentWare Vision and Product Roadmap	http://www.i-kinetics.com/wp/cwvision/CWvision.htm
Open Directory: Component Frameworks	http://dmoz.org/Computers/Programming/Component_Frameworks/
Why Component-Oriented Programming?	http://www.odateam.com/cop/copwhy.html
CORBA, DCOM y EJB	
Microsoft COM Technologies	http://www.microsoft.com/com/default.asp
COM and CORBA Side by Side	http://www.distobj.com/com_corba.html
Object Management Group	http://www.omg.org
The Free CORBA page	http://adams.patriot.net/tvalesky/freecorba.html
Real-time CORBA with TAO (The ACE ORB)	http://siesta.cs.wustl.edu/schmidt/TAO.html
Distributed Object Computing and CORBA	http://siesta.cs.wustl.edu/schmidt/corba.html
Distributed Objects	http://www.DistributedObjects.com/
IONA Technologies: OrbixWEB	http://www.iona.com/products/internet/orbixweb/
Enterprise JavaBeans(TM) Technology	http://java.sun.com/products/ejb/index.html
JavaBeans(TM): Development Tools	http://java.sun.com/beans/tools.html
View Source: Advanced JavaBeans	http://developer.netscape.com/viewsource/
Computación móvil y Agentes	
JAE project	http://www-i4.informatik.rwth-aachen.de/jae/
IBM Aglets Software Development Kit	http://www.trl.ibm.co.jp/aglets/
UMBC Agent Web	http://www.cs.umbc.edu/agents/
Mobile Agents and the Future of the Internet	http://www.cs.dartmouth.edu/dfk/papers/kotz:future2/
Applications of a Web Query Language	http://atlanta.cs.nchu.edu.tw/www/PAPER267.html
Agents, Actors, Roles, and Semantic Interfaces	http://www.alumni.caltech.edu/croft/research/agent/role/
Ambient Calculus – (Luca Cardelli)	http://www.luca.demon.co.uk/Ambit/Ambit.html
Security in Mobile Agent Systems	http://inf.informatik.uni-stuttgart.de/ipvr/vs/projekte/mole/security.html
XML	
MSDN Online: XML Developer Center	http://msdn.microsoft.com/xml/default.asp
Arbortext: Enterprise XML Software	http://www.arbortext.com/index.html
XML WORLD	http://www.interdoc.ca/conference/xml/index.htm
XML.com - XML Object Persistence	http://www.xml.com/
World Wide Web Consortium, W3C	http://www.w3.org
Website Abstraction XML Resources	http://wsabstract.com/webresource/xml.htm
Sistemas distribuidos	
Middleware Espectra	http://www.middlewarespectra.com/
Design for Open Systems in Java	http://gee.cs.oswego.edu/dl/coord/index.html#infospheres
Bringing Distributed Objects to the WWW	http://www.interlog.com/resnick/javacorb.html
Tecnologías Abiertas	http://www.angelfire.com/al/acs/
Distributed Computing Environment (DEC)	http://www.sei.cmu.edu/activities/str/descriptions/dce_body.html
Visual Modeling, UML, Rational Rose	http://www.researchitect.com/mag/index.shtml
UML Resource Center	http://www.rational.com/uml/index.jtimpl
Software Development Online	http://www.sdmagazine.com/uml/focus.htm

Cuadro 3: Utilidades disponibles por Internet relacionadas con los SID

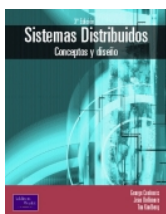
Libros de texto básicos

- [IA02] Luis Iribarne y Rosa Ayala. *Sistemas de Información Distribuidos*. ISBN: 84-8240-522-5. Servicio de Publicaciones de la Universidad de Almería. Febrero, 2002.



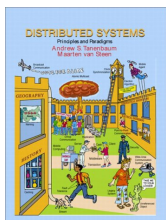
En él intentamos recoger los fundamentos teóricos y tecnológicos más esenciales para el desarrollo de sistemas de información distribuidos. En el libro hacemos una breve introducción al modelo cliente/servidor, presentamos los principios fundamentales de la programación distribuida, describimos las técnicas actuales para el desarrollo de SID basados en componentes y analizamos el concepto y uso de los agentes software. También describimos tecnología que actualmente usa la industria para la construcción de sistemas distribuidos, como CORBA, DCOM, XML, ebXML, SOAP o Servlets, entre otros. En este libro también ofrecemos una sección dedicada al desarrollo de ejercicios prácticos. Además, el libro dispone de soporte en línea desde el **Laboratorio de Sistemas de Información Distribuidos** del *Departamento de Lenguajes y Computación* de la *Universidad de Almería*, <http://www.ual.es/personal/liribarn/LSID>.

- [CDK01] George Coulouris, Jean Dollimore and Tim Kindberg. *Sistemas Distribuidos. Conceptos y Diseño (3ra edición)*. ISBN: 84-7829-049-4. Addison Wesley, 2001.



Desde su primera edición en 1990, este libro se está convirtiendo, a nuestro parecer, en todo un clásico dentro del área de los sistemas distribuidos. Su segunda edición en 1994, y su más reciente actualización en el año 2001, ha permitido actualizar sus contenidos acorde a la demanda real. El libro recoge información sobre principios y práctica que subyace en el diseño de SID, tanto basados en Intranets como en cualquier otro tipo de red. El libro también ofrece contenidos sobre algoritmos distribuidos, coordinación entre procesos, middleware (básicamente CORBA), memoria compartida distribuida y transacciones distribuidas. Aunque trata con tecnología de objetos distribuidos mediante CORBA y Java, y hace referencia a sistemas distribuidos para web, el libro carece de contenido teórico y práctico consistente para el desarrollo SID bajo Internet, como XML, ebXML, SOAP, servlets, servicios webs y otros. En definitiva, el libro no recoge información suficiente para el desarrollo de sistemas del lado servidor, que básicamente es en lo que sustenta los principios de un SID.

- [TV01] Andrew S. Tanenbaum and Maarten Van Steen. *Distributed Systems. Principles and Paradigms*. ISBN: 0130888931. Prentice-Hall, 2001.

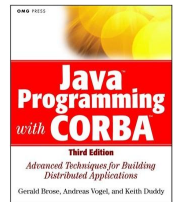


En este libro los autores dividen los conocimientos sobre SID, en lo que ellos denominan las siete claves del conocimiento acerca de los SID. El contenido del libro gira alrededor de estas siete claves: comunicación, procesos, "naming", sincronización, consistencia, tolerancia a fallos y seguridad. El libro desarrolla contenidos sobre sistemas basados en coordinación y objetos distribuidos, y recoge tecnología como CORBA, DCOM y Jini. Sin embargo, a nuestro parecer, el libro no abarca conocimientos suficientes para el desarrollo de sistemas distribuidos basados en Internet. No trata aspectos como XML, ebXML o SOAP, ni tampoco aspectos de sistemas de agentes. No obstante, el libro se adecua bien como libro de texto básico para los Fundamentos de la Programación Distribuida y Técnicas de desarrollo de SID.

Libros de texto complementarios

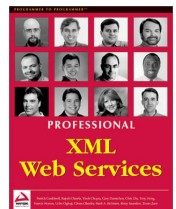
- [BVD01] Gerald Brose, Andreas Vogel and Keith Duddy. *Java Programming with CORBA. Advanced Techniques for Building Applications (Third Edition)*. ISBN 0-471-37681-7. John Wiley and Sons, Inc., 2001.

Esta es la tercera edición de uno de los libros más completos en programación CORBA, desarrollado por los propios miembros del OMG, creador del estándar CORBA. Aunque el libro tiene un nivel bastante alto, sí es aconsejable por sus ejemplos y su visión general para la construcción de aplicaciones distribuidas. Este libro es adecuado como complemento de CORBA.



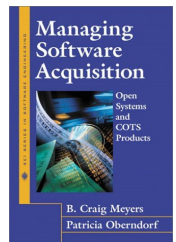
- [Cho01] Vivek Chopra et al. *Professional XML Web Service*. ISBN: 1861005091. Wrox Press Inc. Sep., 2001.

Este libro cubre extensamente conocimientos acerca de la reciente tecnología de los *servicios web*, a nuestro parecer, futuro inmediato de la computación distribuida por web. Entre la tecnología que cubre el libro, destacamos SOAP, UDDI y WDSL. El principal inconveniente de este libro es que está escrito por doce profesionales de la industria en informática distribuida, y en algunos casos, durante su lectura, se pierde conexión entre los capítulos del libro. No obstante, por la novedad del tema y por su escasa literatura, hemos considerado que este libro debería estar en la bibliografía complementaria SID. El libro es adecuado para Técnicas para el desarrollo de SID.



- [MO01] B. Craig Meyers y Patricia Oberndorf. *Managing Software Acquisition*. ISBN: 0-201-70454-4. SEI series in Software Engineering, 2001.

Este libro quizá sea algo avanzado, pero lo hemos seleccionado porque ofrece un completo capítulo sobre los estándares y certificación, utilizados cada vez más en la construcción de grandes sistemas distribuidos abiertos. El libro es adecuado para Normas y recomendaciones de estandarización y El modelo cliente/servidor.



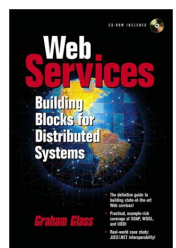
- [Flo00] Michael Floyd. *Creación de sitios Web con XML*. ISBN: 84-8322-259-0. Prentice Hall, 2000.

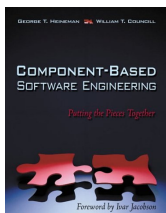
En este libro introduce nociones básicas de XML para el desarrollo de aplicaciones Web. El libro también cubre aspectos de diseño XML basado en técnicas propias del lado servidor (ASP, CGI, Servlets) y del lado cliente (estilos, scripts, applets). El libro está dirigido a profesionales de la informática, administradores de sitios web o desarrolladores en general que necesiten construir un sitio Web con XML. La modelización de datos en XML como base para el desarrollo de un SID está contemplado como una parte imprescindible y es adecuado como Técnicas de desarrollo de SID.



- [Gla01] Graham Glass. *Web Services: Building Blocks for Distributed Systems*. ISBN: 0130662569. Noviembre, 2001.

Este libro es bastante interesante para la construcción de SID que necesiten hacer uso de servicios web. El libro abarca áreas como SOAP, CORBA, COM y UDDI. A nivel práctico, también incluye un pequeño proyecto construido mediante Java, EJB y Microsoft.NET para demostrar la naturaleza interoperable de los servicios Web. El libro es muy adecuado como complemento de las Técnicas para el desarrollo de SID.





- [HC01] George T. Heineman and William T. Councill. *Component-Based Software Engineering. Putting the Pieces Together*. ISBN 0-201-70485-4. Addison-Wesley, 2001.

Esta es una de las más recientes obras dentro de la tecnología de componentes distribuidos. Cubre apartados como CORBA, EJB, COM+ y XML, y describe estrategias y metodologías para el desarrollo de SID a gran escala que hacen uso de componentes software reutilizables y componentes comerciales (COTS). Este libro es muy adecuado en Ingeniería del software basada en componentes distribuidos y en Técnicas para el desarrollo de SID.



- [MTU00] Hiroshi Maruyama, Kent Tamura and Naohiko Uramoto. *Creación de sitios Web con XML y Java*. ISBN: 84-8322-242-6. Prentice-Hall, 2000.

Este libro parte de conocimientos básicos en XML para la construcción de sitios Web. En este libro se tratan aspectos de análisis, construcción y generación de documentos XML, manipulación de estructuras DOM, interconexión de bases de datos mediante XML (JDBC y SQLX), intercambio seguro de mensajes en Internet (SSL y firmas digitales) y desarrollo de aplicaciones haciendo uso de JavaBeans y XML. Como en el caso anterior, este libro lo consideramos adecuado como texto de complemento para las Técnicas de desarrollo de SID.



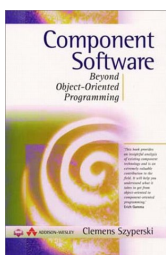
- [OHE00] Robert Orfali, Don Harkey y Juri Edwards. *Cliente/Servidor. Guía de supervivencia (2da edición)*. ISBN: 9701017609. MacGraw-Hill, 2000.

Esta es la segunda edición de uno de los libros más galardonados en los últimos años en el área de la informática distribuida. Los autores combinan explicaciones técnicas detalladas en tono humorístico utilizando caricaturas, recuadros y citas. Recorre aspectos relacionados con entornos cliente/servidor, sistemas operativos, comunicaciones, arquitecturas de aplicación que incorporan bases de datos, procesamiento de transacción y objetos distribuidos. El libro se adecua bien para el modelo Cliente/Servidor.



- [Pre01] Roger S. Pressman. *Ingeniería del software. Un enfoque práctico. (5ta edición)*. McGraw-Hill, 2001.

La quinta edición del libro de Pressman se ha convertido en uno de los libros de referencia más importantes en procesos de ingeniería de software. A nuestro parecer, este libro es de obligada presencia en la bibliografía SID por su relación con los procesos de construcción de sistemas distribuidos en capítulos de reutilización de software y cliente/servidor. Además, trata estos procesos de construcción para el caso de sistemas basados en web. El libro es adecuado como complemento en El modelo Cliente/Servidor y Técnicas de desarrollo de SID.

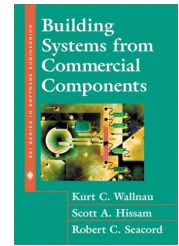


- [Szy98] Clemens Szyperski. *Component Software. Beyond Object-Oriented Programming*. ISBN: 0-201-17888-5. Addison-Wesley, 1998.

Junto con [HEI01], es uno de los libros que debe estar presente en toda bibliografía reciente de componentes distribuidos. El libro analiza con buenos ejemplos el concepto de componente software y sus inicios dentro de la ingeniería de requisitos. El autor justifica extensamente la conveniencia de considerar como disciplina la ingeniería del software basada en componentes distribuidos. A nuestro parecer, esta neodisciplina será —relativamente a corto plazo— una de las bases más importantes en el desarrollo de futuros SID. Este libro es muy adecuado en Ingeniería del software basada en componentes distribuidos y en Técnicas para el desarrollo de SID.

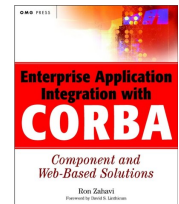
- [WHS02] Kurt C. Wallnau, Scott A. Hissam and Robert C. Seacord. *Building Systems from Commercial Components*. Software Engineering Institute (SEI), Carnegie Mellon University. ISBN: 0-201-70064-6. Addison-Wesley. Enero, 2002.

Este libro, aunque es bastante avanzado, nos ha parecido interesante incluirlo en la bibliografía complementaria porque cubre técnicas y metodologías que están siendo usadas recientemente para la construcción de sistemas distribuidos a gran escala. El libro describe todo un entorno metodológico, principalmente en fases de ingeniería de requisitos, para construcción de grandes sistemas a partir de bloques software o componentes software comerciales. Este libro es adecuado en Ingeniería del software basada en componentes distribuidos y en Técnicas para el desarrollo de SID.



- [ZL01] Ron Zahavi and David S. Linthicum. *Enterprise Application Integration with CORBA Component and Web-Based Solutions*. OMG Press, 2001.

Este libro ofrece una buena visión general acerca de los métodos y técnicas para la integración de aplicaciones del lado servidor. El libro justifica el papel importante del middleware CORBA como integrador entre partes heterogéneas de módulos software, y ofrece excelentes ejemplos de aplicaciones distribuidas extraídos del mundo real. Como sucede con el libro anterior, es adecuado como complemento de CORBA para Técnicas de desarrollo de SID.



Bibliografía

- [1] G. A. Agha, S. Frølund, W. Y. Kim, R. Panwar, A. Patterson, and D. C. Sturman. Abstraction and Modularity Mechanisms for Concurrent Computing. In *TR*. Agha, 1993.
- [2] G. A. Agha, W. Y. Kim, and R. Panwar. Actor Languages for Specification of Parallel Computations. *DIMACS*, 1994.
- [3] P. Alencar and D. Cowan. The Role of Formal Methods in Component-Based Software Engineering. *International Workshop on Component-Based Software Engineering*, pages 187–192, 1999.
- [4] F. Alvarez-García and D. Alvarez-Gutierrez. Component Object Model (COM). *Handbook of Object Technology*. Saba Zamir (ed.). CRC Press, 1998.
- [5] Andersen-Consulting. Building Enterprise Solutions With Distributed Components. Technical report, Andersen Consulting, 1997.
- [6] Andersen-Consulting. Understanding Components. Technical report, Andersen Consulting, 1998.
- [7] H. Attiya and J. Welch. *Distributed Computing: Fundamentals, Simulations and Advanced Topics*. McGraw-Hill, 1998.
- [8] F. Bachman, L. Bass, C. Buhman, S. Comella-Dorda, F. Long, J. Robert, R. Seacord, and K. Wallnau. Technical Concepts of Component-Based Software Engineering. Technical report, SEI technical report CMU/SEI-2000-TR-00, 2000. <http://www.sei.cmu.edu>.
- [9] S. Baker. *CORBA Distributed Objects*. Addison-Wesley Longman, 1997.
- [10] I. Ben-Shaul and G. Kaiser. Coordinating Distributed Components over the Internet. *IEEE Internet Computing*, 98(2):83–86, March-April 1998.
- [11] K. Bharat and L. Cardelli. Migratory Applications. *Mobile Object Systems-Towards the Programmable Internet*. J. Vitek and C. Tschudin, Editors. *Lecture Notes in Computer Science*. Springer, 1222:131–148, 1997.
- [12] L. Bic, M. Fukuda, and M. Dillencourt. Distributed Computing Using Autonomous Agents. 39(8):55–61, 1996.
- [13] J. Bosak. XML, Java, and the future of the Web. Technical report, Sun Microsystems, 1997.
- [14] D. Box. *Essential COM*. Addison-Wesley, 1998.
- [15] P. Brereton. Evolution of Component Based Systems. *International Workshop on Component-Based Software Engineering*, pages 123–126, 1999.
- [16] A. W. Brown and K. C. Wallnau. The Current State of CBSE. *IEEE Software*, 15(5):37–46, Sep./Oct. 1999.

- [17] L. Cardelli. Mobile Computation. *Mobile Object Systems - Towards the Programmable Internet*. J. Vitek and C. Tschudin, Editors. *Lecture Notes in Computer Science*, 1222:3–6, 1997.
- [18] L. Cardelli. Mobile Ambients. *Foundations of Software Science and Computational Structures*, Maurice Nivat, Editor. *Lecture Notes in Computer Science*, 1378:140–155, 1998.
- [19] D. Carney. Requirements and COTS-Based Systems: A Theory Question Indeed. Technical report, SEI Interactive. Software Engineering Institute (SEI), 1999.
- [20] N. Carriero and D. Gelernter. *How to Write Parallel Programs: A First Course*. MIT Press, Cambridge, MA, 1990.
- [21] N. Carriero and D. Gelernter. Coordination Languages and their Significance. *Commun. ACM*, 35(2):97–107, Feb. 1992.
- [22] CCM. *The CORBA Component Model*. Object Management Group, June 1999. <http://www.omg.org>.
- [23] K. Chandy and J. Misra. *Parallel Program Design: A Foundation*. Addison Wesley, Reading, MA, 1988.
- [24] K. M. Chandy et al. A Framework for Structured Distributed Object Computing. *Commun. ACM*, 1997.
- [25] M. Chandy and A. Rifkin. *Systematic Composition of Objects in Distributed Internet Applications: Processes and Sessions*. Proceedings of the 30th HICSS, 1997.
- [26] P. E. Chung, Y. Huang, and S. Yajnik. DCOM and CORBA Side by Side, Step by Step, and Layer by Layer. Technical report, Bell Laboratories, Lucent Technologies, 1998.
- [27] P. Ciancarini et al. Coordinating Multiagent Applications on the WWW: A Reference Architecture. *IEEE Trans. Softw. Eng.*, 24(5):362–375, May 1998.
- [28] J. R. Corbin. *The Art of Distributed Applications*. SV, 1991.
- [29] B. Cottman. ComponentWare: Component Software for the Enterprise. *Engineering Institute. SEI Interactive*, 1998. <http://www.i-kinetics.com/wp/cwvision/CWvsion.htre>.
- [30] E. Denti et al. An Extensible Framework for the Development of Coordinated Applications. In *Proc. of COORDINATION'96*, number 1061 in LNCS, pages 305–319. Springer-Verlag, 1996.
- [31] D. F. D'Souza and A. C. Wills. *Objects, Components, and Frameworks with UML: The Catalysis Approach*. ISBN: 0-201-31012-0. Addison Wesley, 1999.
- [32] ESI. European Software Institute, 2000. <http://www.esi.es/>.
- [33] M. Floyd. *Creación de sitios Web con XML*. ISBN: 84-8322-259-0, 2000.
- [34] I. Foster. *Designing and Building Parallel Programs*. Addison-Wesley, 1995.
- [35] S. Franklin and A. Graesser. Is it an Agent, or Just a Program?: A Taxonomy for Autonomous Agents. In *Proc. of the 3rd International Workshop on Agent Theories, Architectures and Languages*, LNCS. Springer-Verlag, 1996.
- [36] S. Frølund. *Coordinating Distributed Objects: An Actor-Based Approach to Synchronization*. MIT Press, 1996.

- [37] A. Fugetta, G. Picco, and G. Vigna. Understanding Code Mobility. *IEEE Transactions on Software Engineering*. Vol. 24, Num. 5, pp., 342-361, 1998.
- [38] C. Guifoyle. Vendors of Intelligent Agent Technologies: A Market Overview. *Agent Technology. Foundations, Applications and Markets*. Springer-Verlag. Edit. N. Jennings and M.J. Wooldridge, 1998.
- [39] B. Henderson-Sellers, R. Pradhan, C. Szyperski, A. Taivalsaari, and A. C. Wills. Are Components Objects? In *OOPSLA'99 Panel Discussions*, Nov. 1999.
- [40] M. Henning. Binding, Migration, and Scalability in CORBA. Technical report, CRC for Distributed Systems Technology. University of Queensland, 1999. <http://zeus.cs.wayne.edu/corba/archive/index.htm>.
- [41] A. Holzbacher. A Software Environment for Concurrent Coordinated Programming. In *Proc. of COORDINATION'96*, number 1061 in LNCS, pages 249-266. Springer-Verlag, 1996.
- [42] E. Horowitz. Migrating Software to the World Wide Web. pages 18-21, May-June 1998.
- [43] C. Iglesias, M. Garijo, and J. González. Metodologías orientadas a agentes. *Revista Iberoamericana de Inteligencia Artificial*, 6, Otoño 1998.
- [44] IONA. OrbixWeb 3.1 The Internet ORB. Technical report, Iona Technologies, 1999. <http://www.iona.com/>.
- [45] L. Iribarne. Una metodología para diseñar sistemas de resolución distribuida de problemas (SRDP) basados en sistema multiagente (SMA). Technical report, Department of Lenguajes y Computación. University of Almería, Feb 1999.
- [46] L. Iribarne. Una visión global de la computación móvil. Ambients Calculus. Technical report, R-Ambients-99. Department of Lenguajes y Computación. University of Almería, Sep 1999.
- [47] L. Iribarne. Plantillas para la especificación de componentes COTS. Technical report, TR-Templates-00. Department of Lenguajes y Computación. University of Almería, Feb 2000.
- [48] L. Iribarne and A. Vallecillo. Searching and Matching Software Components with Multiple Interfaces. In *CBD Workshop at TOOLS Europe'2000. France*, Jun. 2000. <http://apolo.lcc.uma.es/~av>.
- [49] L. Iribarne and A. Vallecillo. Sobre la búsqueda y emparejamiento de componentes COTS con múltiples interfaces. In *JISBD'2000. Jornadas de Ingeniería del Software y Bases de Datos*, Nov. 2000. <http://www.ual.es/~liribarn>.
- [50] L. Iribarne and A. Vallecillo. Una metodología para el desarrollo de software basado en COTS. Technical report, 1er taller de Ingeniería del Software basada en Componentes Distribuidos IScDIS00. Universidad de Extremadura, 2000.
- [51] N. Islam. Distributed Objects. Methodologies for Customizing Systems Software. *IEEE Computer Society Press*, 1996.
- [52] P. Jason. *COM and CORBA Side by Side*. ISBN: 0-201-37945-7. Addison Wesley, 1999.
- [53] N. R. Jennings and M. Wooldrige. Applications of Intelligent Agents. Technical report, Agent Technology: Foundations, Applications, and Markets, 1998. <ftp://ftp.elec.qmw.ac.uk/pub/isag/distributed-ai/publications/agt-technology.pdf>.

- [54] K. Kang. Issues in Component-Based Software Engineering. *International Workshop on Component-Based Software Engineering*, pages 207–212, 1999.
- [55] D. Kiely. Are Components the Future of Software? *Computer*, 32(2):10–11, Feb. 1998.
- [56] D. Konstantas. Interoperation of Object Oriented Applications. In O. Nierstrasz and D. Tschritzis, editors, *Object-Oriented Software Composition*, pages 69–95. Prentice-Hall, 1995.
- [57] D. Krieger and R. M. Adler. The Emergence of Distributed Component Platforms. 41(3):43–53, Mar. 1998.
- [58] D. Kunda and L. Brooks. Human, Social and Organisational Influences on Component-Based Software Engineering. *International Workshop on Component-Based Software Engineering*, 1999.
- [59] D. B. Lange and M. Oshima. *Programming Mobile Agents in Java –with the Java Aglet API*. IBM Research, 1997.
- [60] S. C. Laufmann. Agent Software for Near-Term Success in Distributed Applications. *Foundations, Applications and Markets*. Springer-Verlag. Edit. N. Jennings and M.J. Wooldridge, 1998.
- [61] G. T. Leavens and M. Sitaraman. *Foundations of Component-Based Systems*. ISBN: 0-521-77164-1. Cambridge University Press, 2000.
- [62] D. Leijen. Functional Components. Using COM components in Haskell. Master’s thesis, University of Utrecht, 1998.
- [63] N. Lynch. *Distributed Algorithms*. Morgan Kaufmann, 1996.
- [64] F. Manola. Interoperability Issues in Large-Scale Distributed Object Systems. *ACM Comput. Surv.*, 27(2):268–270, June 1995.
- [65] P. Maurer. Components: What If They Gave a Revolution and Nobody Came? *IEEE*, Vol. 33, Num. 6, pages 28–34, Jun., 2000.
- [66] C. McFall. An Object Infrastructure for Internet middleware. IBM on Component Broker. *IEEE Internet Computing*, 98(2):46–51, March-April 1998.
- [67] B. Meyer. The Significance of Components. *Beyond Objects column, Software Development*, 7(11), nov 1999.
- [68] B. Meyer, C. Mingins, and H. Schmidt. Trusted Components for the Software Industry. *Computer*, 31(4), may 1998. <http://www.trusted-components.org/>.
- [69] R. Milner. *Communication and Concurrency*. Prentice Hall, 1989.
- [70] J. Milojicic, F. Douglass, and R. Wheeler. *Mobility, Processes, Computers and Agents*. Addison-Wesley, 1999.
- [71] R. Monson-Haefel. *Enterprise JavaBeans*. ISBN: 1-56592-605-6. O’Reilly, 1999.
- [72] P. Morreale. Agents on the Move. *IEEE Spectrum*, 1998.
- [73] E. Najm and J.-B. Stefani, editors. *Formal Methods for Open Object-based Distributed Systems*, volume 1. Chapman & Hall, Paris, 1996.
- [74] E. Najm and J.-B. Stefani. Computational Models for Open Distributed Systems. In H. Bowman and J. Derrick, editors, *Proc. of FMOODS’97*, Canterbury, 1997. Chapman & Hall.

- [75] H.Ñwana. Software Agents: An Overview. *Knowledge Engineering Rev.*, 11(3):205–244, sep 1996.
- [76] OMG. The Object Management Group, OMG. <http://www.omg.org>.
- [77] R. Orfali and D. Harkey. *The Essential Distributed Objects Survival Guide*. John-Wiley and Sons, 1996.
- [78] OSF. *OSF DCE Application Development Guide*. Cambridge, MA, 1994. <http://www.opengroup.org>.
- [79] G. Papadopoulos and F. Arbab. Coordination Models and Languages. *Advances in Computers*, 48, 1998.
- [80] J. Pritchard. *COM and CORBA. Architectures, Strategies and Implementations*. Addison-Wesley Longman, 1999.
- [81] G. S. Raj. A Detailed Comparison of CORBA, DCOM and Java/RMI. Technical report, <http://www.execpc.com/gopalan>, 1999.
- [82] M. Raynal. *Distributed Algorithms and Protocols*. John Wiley & Sons, 1988.
- [83] D. C. Schmidt. Lessons Learned Building Reusable Object-Oriented Frameworks for Distributed Software. *Commun. ACM*, 40(10):85–87, oct 1997.
- [84] SEI. Software Engineering Institute. COTS-Based Systems (CBS) Initiative. <http://www.sei.cmu.edu/cbs/index.html>.
- [85] R. Sessions. *COM and DCOM. Microsoft's Vision for Distributed Objects*. John Wiley and Sons, Inc., 1998.
- [86] J. Siegel. *CORBA 3. Fundamentals and Programming*. John Wiley & Sons. OMG Press, 2000.
- [87] S. Smith and C. Talcott, editors. *Formal Methods for Open Object-based Distributed Systems*, volume 4. Stanford, CA, Sept. 2000.
- [88] C. Thompson, T. Bannon, G. Hansen, F. Manola, M. Palmer, P. Pazandak, and V. Vasudevan. Scaling Object Service Architectures to the Internet. Technical report, OBJS (Object Services and Consulting) document., 1998. <http://www.objs.com/OSA/Final-Report.html>.
- [89] J. M. Troya and A. Vallecillo. Integrating Components into Open Systems. In *Actas de JIS'97*, pages 245–258, San Sebastián, sep 1997.
- [90] J. M. Troya and A. Vallecillo. Integración de componentes heterogéneos en sistemas abiertos y distribuidos. In *Actas de SEID'99*, pages 49–58, Santiago de Compostela, feb 1999. Tórculo Ediciones.
- [91] Valesky. *Enterprise JavaBeans*. ISBN: 0-201-60446-9. Addison Wesley, 1999.
- [92] A. Vallecillo, J. Hernández, and J. M. Troya. Object Interoperability. In *Object-Oriented Technology: ECOOP'99 Workshop Reader*, number 1743 in LNCS, pages 1–21. Springer-Verlag, 1999.
- [93] A. Vallecillo, J. Hernández, and J. M. Troya. New Issues in Object Interoperability. In *Object-Oriented Technology: ECOOP'2000 Workshop Reader*, number 1964 in LNCS. Springer-Verlag, 2000.
- [94] W. van den Heuvel, M. Papazoglou, and M. Jeusfeld. Configuring Business Objects from Legacy Systems. In *Proc. of the 11th Conference on Advanced Information Systems Engineering (CAiSE'99)*, Heidelberg, Germany, June 1999. MIT Press.

-
- [95] S. Vinoski. New Features for CORBA 3.0. *Communications of the ACM*, 41(10):44–52, October 1998.
- [96] J. Vitek. New Paradigms for Distributed Programming. Technical report, European Research Seminar in Advanced Distributed Systems, ERSADS'97, Zinal, Switzerland, 1997. <http://www.cs.purdue.edu/homes/jv/publist.html>.
- [97] W3C. The World Wide Web Consortium. <http://www.w3.org>.
- [98] W3C. *Simple Object Access Protocol (SOAP) 1.1*. World Wide Web Consortium, May 2000. <http://www.w3.org/TR/SOAP>.
- [99] K. Wallnau, A. Earl, E. Morris, E. Litvak, and P. Feiler. Engineering Component-Based Systems with Distributed Object Technology. Technical report, Software Engineering Institute (SEI), Carnegie Mellon University, 1997.
- [100] N. Weiderman, S. Tilley, L. Northrop, K. Wallnau, and D. Smith. Implications of Distributed Object Technology for Reengineering. *Software Engineering Institute (SEI), Carnegie Mellon University*, 1997. <http://www.sei.cmu.edu/>.
- [101] J. Whitten, L. Bentley, and V. Barlow. *Análisis y Diseño de Sistemas de información*. Irwin, 1996.
- [102] G. Wiederhold. Mediation in Information Systems. *ACM Comput. Surv.*, 27(2):265–267, jun 1995.
- [103] C. Zhang and L. Dickson. *Multi-Agent Systems Methodologies and Applications*. Second Australian Workshop on Distributed Artificial Intelligence, Cairns, QLD, 1997.