Revista Iberoamericana de Automática e Informática Industrial
CEA comité español de automática
RIAI
ISSN: 1697-7912. Vol. 5, Núm. 2, Abril 2008, pp. 107-117
http://riai.isa.upv.es

DESARROLLO ORIENTADO A COMPONENTES DE UNIDADES DE CONTROL DE ROBOTS: APLICACIÓN A ROBOTS DE LIMPIEZA DE CASCOS DE BUQUES

Andrés Iborra, Juan A Pastor, Bárbara Álvarez, Francisco Ortiz, Carlos Fernández.

División de Sistemas e Ingeniería Electrónica (DSIE)
Universidad Politécnica de Cartagena
e-mail: andres.iborra@upct.es

Resumen: Una operación crítica en el mantenimiento de buques es el granallado del casco antes de su repintado. Desde hace más de una década han ido apareciendo en el mercado soluciones que pretenden resolver este problema mediante la robotización de dicha operación. Aún así, esta operación continua realizándose habitualmente en los astilleros de un modo manual. Este artículo presenta una de familia de robots que supone una solución óptima al problema planteado y que satisface razonablemente todos los requisitos de operatividad, eficiencia, seguridad y respeto al medio ambiente necesarios para trabajar hoy en día en un astillero. Para el diseño de los sistemas de control se ha seguido un enfoque basado en componentes con el objeto de incrementar la reutilización del código y disminuir los tiempos de desarrollo. El trabajo aquí presentado ha sido desarrollado en el marco del proyecto "EFTCoR", financiado por el VPM de la UE.

Palabras Clave: Robótica de servicios, teleoperación de robots, arquitecturas software.

1. INTRODUCCIÓN

Al igual que muchas plantas industriales, los buques son sometidos a paradas técnicas de mantenimiento, que se programan periódicamente cada 4-5 años. Una de las operaciones de mayor importancia consiste en la retirada del casco del buque de las adherencias marinas, así como en la preparación de dicha superficie para su repintado. Esta operación se realiza para conservar la integridad del casco garantizando las condiciones idóneas de navegación. Mantener la superficie del casco en buenas condiciones hidrodinámicas permite minimizar el consumo de combustible y por tanto reducir la contaminación atmosférica. La tecnología actual (Wayle, 1998) (Smith,1999), para la limpieza de cascos de buques más ampliamente utilizada, y que es preferida por la mayor parte de los armadores, consiste en el granallado o "chorreado" a cielo abierto (ver Fig.1). Con la utilización de dicha tecnología se consigue un acabado superficial (SA 2 ½ y ausencia de oxidación del casco) que no se alcanza con el empleo del agua a muy alta presión (Ultra High Pressure Water Jetting), lo cual redunda en una mejor adherencia de la pintura y por tanto en periodos más prolongados en los que no es necesario repintar el buque.

Sin embargo, la tecnología de granallado es poco respetuosa con el medio ambiente al realizarse habitualmente a cielo abierto y generar una gran cantidad de residuos en forma de polvo que se esparce fácilmente por la atmósfera y las proximidades del astillero, incluido el mar. El polvo que se genera consiste en una mezcla de pinturas ricas en metales pesados y biocidas, así como en fragmentos de granalla (piritas, arenas de sílice, etc.). Por todo ello, el granallado a cielo abierto se está prohibiendo progresivamente en los países europeos más sensibles a los problemas medio ambientales, con una tendencia clara a ser prohibida definitivamente en el resto de Europa, y forzando por tanto a que el granallado sea sustituido irremediablemente por el agua a muy alta presión (UHP) a pesar de que no se obtienen los mismos resultados en cuanto a calidad de la superficie. De ahí que los propietarios de los barcos estén acudiendo a astilleros en países en los que el granallado a cielo abierto está todavía permitido (países del Este, Corea, China, etc) con las consecuentes pérdidas económicas para Europa.

La robotización de estas tareas empleando granallas reutilizables, trabajando en ciclo cerrado y confinando la zona del casco que se está limpiando

es un tema mecánico cuya solución no es nada trivial. El trabajo en entornos con gran número de obstáculos (grúas, raíles, andamios, casetas, equipos de mantenimiento, cables, hélices, etc) y con superficies de casco de muy diferentes formas y tamaños dificultan el diseño de dispositivos mecánicos robotizados de propósito general.

Un problema mucho más simple es la limpieza de grandes superficies verticales, existiendo desde hace tiempo en el mercado soluciones robotizadas para limpieza de este tipo de superficies tanto con agua (Goldie, 1999b) como con granalla (Goldie, 1999a) y en las cuales se obtiene un rendimiento muy alto, si bien a un coste nada despreciable.

Este artículo presenta una familia de robots especializados, de bajo coste utilizados para limpieza con granalla y capaces de obtener una alta calidad en la preparación de la superficie (SA 2 ½) junto a una drástica reducción de los residuos generados. Los distintos sistemas robóticos que se van a presentar han sido desarrollados en el marco del proyecto europeo EFTCoR (EFTCoR, 2002), y son el resultado de aunar los esfuerzos de astilleros, fabricantes y centros de investigación.

En la sección 2, se muestran los criterios de diseño impuestos por los astilleros. La sección 3 presenta desde un punto de vista funcional la familia de robots y los beneficios que se han obtenido mediante el empleo de los mismos. La sección 4 describe globalmente el sistema EFTCoR y en el cual se integran los robots vistos en la sección anterior. En la sección 5 se muestra la arquitectura del sistema de control, la cual se descompone jerárquicamente en tres niveles. En esa misma sección se muestra un marco arquitectónico para el desarrollo de unidades de control basado en componentes (ACROSET), así como, algunos ejemplos de instanciación realizados para el EFTCoR. Por último, en la sección 6 se resumen las conclusiones.

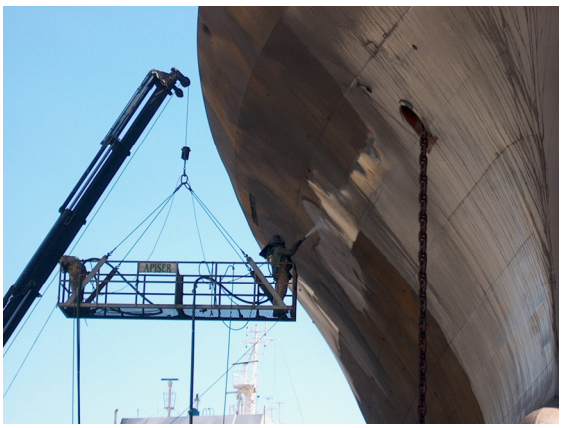


Figura 1. Operación de limpieza manual.

2. CRITERIOS GENERALES PARA EL DISEÑO DE LOS DISPOSITIVOS ROBOTIZADOS.

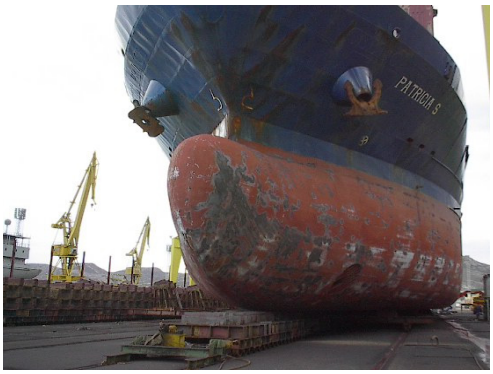
Los criterios de diseño mecánico que hay que tener en cuenta a la hora de robotizar este tipo de tareas deben considerar los requisitos funcionales y no funcionales impuestos por el astillero. Algunos de estos fueron los siguientes:

- **La técnica de base para la limpieza debe ser el granallado**, con el objeto de obtener el mejor acabado superficial, y evitar los problemas de oxidaciones que conlleva el chorreado con agua a alta presión.
- **La cantidad de polvo emitido a la atmósfera debe ser lo más reducida posible**, lo cual obliga a confinar la zona de granallado, y utilizar métodos de aspiración que arrastren tanto la granalla utilizada como los residuos que se generan.
- **La cantidad de residuos generados debe ser la menor posible**, con el objeto de solventar los problemas de recogida, transporte y almacenaje de los mismos. Este requisito obliga a utilizar granallas que puedan ser reutilizadas un número determinado de ciclos y a incorporar elementos de recogida de la granalla, separación de los residuos, almacenaje temporal y recirculación.
- **El material de granallado reciclable deberá tener las propiedades mecánicas necesarias para que se pueda obtener una calidad superficial** al menos tan buena como la que se obtiene con granallas desechables. Dichas propiedades deben degradarse lo menos posible en relación con el número de ciclos de trabajo y a un coste razonable.
- **Las dimensiones y formas de los buques son muy diferentes** debido a sus características hidrodinámicas, pudiendo existir sobre la superficie del barco obstáculos de muy diferente naturaleza (ojos de buey, remaches, deformaciones debidas a colisiones, chapas de refuerzo, alerones, etc.).
- **Las condiciones de trabajo son muy diferentes** según la zona del buque donde se esté trabajando (fondos, finos o superficies verticales), así como del tipo de astillero (dique seco o elevadores del tipo Synchrolift)(ver Fig.2).
- **Desde el punto de vista operacional existen dos modos de trabajo** ("full blasting" y "spot blasting"). El "full blasting" consiste en el chorreado completo del casco del buque, mientras que el "spot blasting" consiste en chorrear numerosos puntos aislados en los que se ha observado corrosión. El "full blasting" requiere dispositivos robotizados capaces de posicionar grandes cabezales de limpieza que se muevan a lo largo de todo el casco del buque

con objeto de obtener elevados rendimientos. Por el contrario, el "spot blasting" requiere de dispositivos robotizados que posicionen cabezales pequeños rápidamente y con mucha precisión.

- **Los sistemas robotizados deben poder ser utilizados para ejecutar otras operaciones** de mantenimiento (baldeado y pintado).

a) Fondos. Syncrolift.



b) Finos. Syncrolift.

c) Superficies verticals. Dique seco.

Figura 2. Posibles entornos de trabajo en un buque.

La importancia de cada uno de estos requisitos es relativa y depende de la cultura y prioridades de negocio del astillero en cuestión. Por lo tanto, una solución orientada al cliente, tal y como se presenta aquí deberá poder ser adaptable a las necesidades del mismo, y condiciona los criterios de diseño de los distintos sistemas robotizados. La tabla 1 resume los principales requisitos impuestos por dos importantes astilleros europeos. Cómo puede observarse, algunos de ellos no son homogéneos.

Tabla 1. Requisitos impuestos por dos astilleros

Requisitos	Astillero 1	Astillero 2
Costes	*No más que los costes actuales incluyendo los sueldos*	*Igual o mejores que los costes operativos obtenidos con las técnicas convencionales. Los costes de los abrasivos deberían reducirse.*
Prestaciones	*5 m2/man-hour. Eficiencia de la boquilla 10 m2/hora.*	*No las cuantifica numéricamente*
Medio ambientales	*Reducción de las emisiones de polvo (al menos 70%)*	*La cantidad de abrasivo utilizado debería reducirse drásticamente*
Área de trabajo	*Synchrolift. Separación entre buques muy estrecha..La eliminación de obstáculos en el área de trabajo supone un problema organizacional.*	*Dique seco. Área de trabajo muy extensa, pero espacio disponible limitado.*
Capacidad de adaptación a los diferentes modos de trabajo("full blasting" y "spot blasting")	*El "Spot" supone el 80% de los trabajos.*	*El "Spot" supone el 35% del trabajo y el 48% de todo el negocio de chorreado.*
Calidad de la superficie obtenida	*SA 2 1/2 (ISO 8501-1)*	*SA 2 1/2 (ISO 8501-1)*
Capacidad de adaptación a otros trabajos de mantenimiento	*Baldeado, pintado.*	*Baldeado, pintado.*
Capacidad de adaptación a diferentes tipos de barcos y de formas de los cascos	*Buques de hasta: 125 m de longitud 25 m profundidad 23 m amplitud Gran variabilidad en cuanto a formas de barcos*	*Buques cisterna de hasta 300m. Gran cantidad de superficies verticales.*
Facilidad de operación	*Debe poder manejarse por personal poco cualificado*	*Debe poder manejarse por personal poco cualificado*
Posibilidad de automatización	*Si*	*Si*
Otras	*Posibilidad de acceso on line por el departamento de calidad*	*Fácil de transportar y de montar*

3. FAMILIA DE ROBOTS EFTCoR

Dentro del proyecto EFTCoR, se optó por resolver el problema planteado mediante el diseño de una familia de robots de servicios especializados y de bajo coste que pudieran trabajar en diferentes tareas, en vez de diseñar un robot de propósito general. La diferente naturaleza y requisitos de los diferentes modos de trabajo ("full blasting" y "spot blasting") dieron lugar a diferentes sistemas robotizados dependiendo de la zona del casco del buque donde fueran a trabajar (superficies verticales, finos y fondos).

Todos los sistemas constan de un sistema primario de posicionamiento de tres grados de libertad, un elemento secundario opcional (montado sobre el primario) y un cabezal de limpieza que puede ser

una turbina (Fig. 3) o una boquilla de granallado con campana de confinamiento (Fig. 4).

Como sistemas primarios se han diseñado y construido los siguientes dispositivos:

- **Torres verticales**. Consisten en 2 torres verticales con tres grados de libertad, guiadas sobre carriles y con una capacidad de carga de 500 Kg. La primera de las torres (Fig. 2c) se ha instalado sobre un dique seco, tiene una altura de 30 metros (eje Z), se puede desplazar unos 300 metros en el eje X sobre carriles en el suelo y unos dos metros en el eje Y. La segunda de las torres (Fig. 3) se ha instalado sobre un Syncrolift, tiene una altura de 12 metros (eje Z), se puede desplazar unos 100 metros en el eje X y unos 2 metros en el eje Y. Además cuenta con dos grados de libertad adicionales para orientar las herramientas según la forma del casco del buque. Ambas se emplean para aquellas operaciones en las que se requiere cubrir zonas de trabajo muy amplias.
- **Robot escalador**. Consiste en un vehículo (Fig. 5) que se agarra magnéticamente sobre el casco del barco, desplazándose a una velocidad de 0,5 m/seg sin granallar y a 0,2 m/seg granallando. Tiene una capacidad de carga de 10 Kg. Es utilizado principalmente para poder acceder a aquellas zonas que el resto de sistemas no pueden alcanzar debido a la naturaleza de los obstáculos existentes o a la falta de espacio.

También se ha empleado como elemento primario una mesa elevadora motorizada para la limpieza de fondos, la cual es un elemento comercial.

Como elemento secundario se ha desarrollado una mesa XYZ (Fig. 6 y 7) para trabajos de Spot Blasting, cuyo cabezal se mueve a una velocidad de 1 m/seg. para posicionar el cabezal y 0,2 m/seg durante la operación de chorreado.

Todos los dispositivos del EFTCoR han sido probados en los astilleros de Navantia en el Ferrol y Cartagena bajo condiciones reales de funcionamiento (tres meses en Cartagena utilizando maquetas y tres semanas en el Ferrol con barcos reales). La Tabla 2 resume los desarrollos realizados.

La figura 3 muestra la segunda torre vertical usada como primario para posicionar la mesa XYZ o una turbina (tal y como se ve en la figura).

Tabla 2. Familia de robots EFTCoR.

OPERACIÓN DE LIMPIEZA	**Área del casco considerada**		
	Superficies verticales	**Finos**	**Fondos**
Full blasting *Grandes superficies*	*Sistema primario:* Torres verticales + *Cabezal:* Turbinas	*Sistema primario*: Torres verticales + *Cabezal:* Boquilla	*Sistema primario:* Mesa elevadora + *Cabezal:* Turbina
		Sistema primario: Vehículo escalador + *Cabezal:* Boquilla	*Sistema primario:* Vehículo escalador + *Cabezal:* Boquilla
Spot blasting: *Pequeñas y múltiples superficies dispersas por toda la obra viva del casco.*	*Sistema primario:* Torres verticales + *Sistema secundario:* Mesa XYZ + *Cabezal:* Boquilla	*Sistema primario:* Torres verticales + *Sistema secundario:* Mesa XYZ + *Cabezal:* Boquilla	*Sistema primario:* Mesa elevadora + *Sistema secundario:* Mesa XYZ + *Cabezal:* Boquilla
	Sistema primario: Vehículo escalador + *Cabezal:* Boquilla	*Sistema primario:* Vehículo escalador + *Cabezal:* Boquilla	*Sistema primario:* Vehículo escalador + *Cabezal:* Boquilla

Figura 3. Torres verticales y maqueta de pruebas.

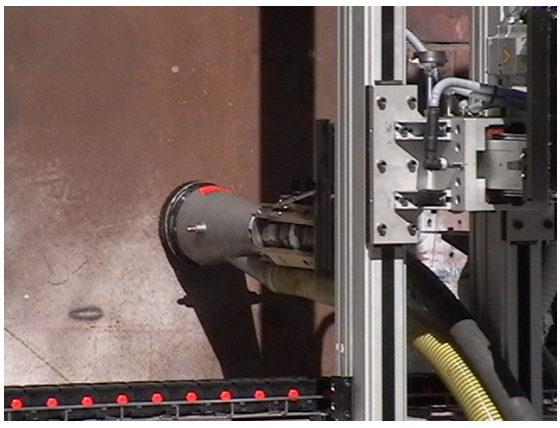


Figura 4. Cabezal de limpieza

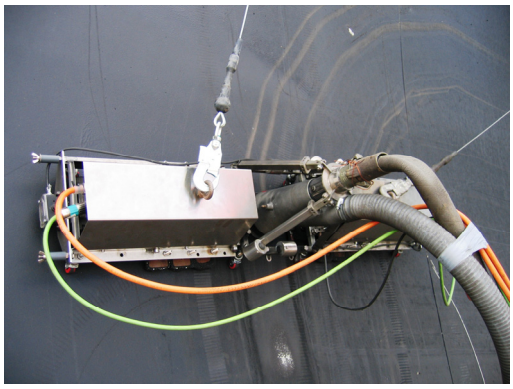


Figura 5. Vehículo de limpieza Lázaro

Figura 6. Calibración mesa XYZ en Navantia.

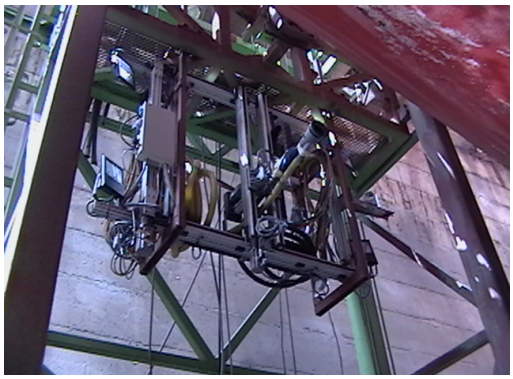


Figura 7. Mesa XYZ (sistema de posicionamiento secundario) montada sobre torres verticales (sistema de posicionamiento primario).

3.1 Evaluación de las prestaciones obtenidas en astilleros.

A la hora de obtener unos resultados que nos permitan evaluar las prestaciones de los diferentes sistemas robotizados desarrollados, tendremos que tener en cuenta que éstos no sólo van a depender del tipo de barco con el que se hagan las pruebas, sino también del tipo de instalación en el que se vayan a realizar. La Tabla 3 muestra las características más importantes de los barcos que pueden operar en astilleros donde fueron evaluados nuestros robots. Como se puede ver, la muestra es lo suficientemente representativa como para considerar que los resultados obtenidos puedan ser generalizables a cualquier otro tipo de astillero.

Tabla 3. Características de los astilleros seleccionados para realizar las pruebas.

ASTILLERO	**DWT [Ton]**	**Manga [m]**	**Calado * [m]**	**Eslora [m]**	**Altura [m]**
NAVANTIA-Cartagena	Hasta 5500	23	9	125	25
NAVANTIA-Ferrol	5. 000-34.0000	15-70	4-25	70-360	NA

*(desde la quilla a la línea de flotación)

La tabla 4 resume los resultados obtenidos con los robots de la familia EFTCoR en los dos astilleros de referencia, y comparando los mismos con los parámetros obtenidos utilizando el procedimiento manual habitual. Los parámetros que se han registrado tienen que ver con la eficiencia horaria y total (incluidos los tiempos muertos), así como los costes. Como se puede ver, existen importantes diferencias de un astillero a otro debido a que los entornos de trabajo así como las técnicas empleadas son muy distintas (dique seco en Ferrol, Synchrolift en Cartagena). En el peor de los casos la eficiencia se ha mantenido y en algunos casos se ha incrementado. El impacto en la eficiencia total (m^2/día) es aún mayor porque no es necesario realizar paradas periódicas, como en las operaciones manuales, donde sí que lo es debido a las duras condiciones de trabajo.

Tabla 4. Resultados obtenidos.

PÁRAMETRO EVALUADO		**ASTILLERO**			
		NAVANTIA Cartagena		**NAVANTIA Ferrol**	
		Full	**Spot**	**Full**	**Spot**
Eficiencia horaria	Manual	25 m^2/hora	17,5 m^2/hora	180 m^2/hora	NA
	EFTCoR	**30 m^2/hora**	**22,3 m^2/hora**	**180 m^2/hora**	**35 m^2/hora**
Eficiencia total	Manual	400 m2/día	290 m2/día	1500 m^2/día	NA
	EFTCoR	**540 m^2/día**	**325 m^2/día**	**1500 m^2/día**	**620 m^2/día**
Costes	Manual	8.1 €/m^2	10.7 €/m^2	NA	NA
	EFTCoR	**7 €/m^2**	**9 €/m^2**	**7 €/m^2**	**9 €/m^2**

Aún en los casos en los que la eficiencia total es similar a la operación manual, la ventaja de tener un sistema que opera en ciclo cerrado, separando por un lado los residuos y reutilizando la granalla

por otro, supone un claro avance sobre las técnicas anteriores.

Los costes mostrados en la Tabla IV incluyen los costes de la granalla. Aun utilizando un abrasivo más caro (TIGRIT®) los costes se reducen debido a que puede llegar a ser reutilizado unas 200 veces gracias al sistema de recirculación. Los costes de personal también se reducen.

4. DESCRIPCIÓN GENERAL DEL SISTEMA EFTCoR.

El sistema EFTCoR (ver Fig. 8) consta de los siguientes elementos:

- **La familia de robots**, vista en el apartado anterior.

- **Una unidad de control**, por cada robot, adaptada a la funcionalidad y misiones que debe llevar a cabo. Así, la unidad de control del vehículo escalador es un PC industrial empotrado con RT LINUX, mientras que las unidades de control de las torres se basan en un control más convencional basado en autómatas. Cada unidad de control dispone de sus propias interfaces hombre-máquina, algunas muy sofisticadas, otras más sencillas. Según el modo de funcionamiento, las unidades de control pueden recibir órdenes de sus interfaces locales, de la plataforma de teleoperación o de otros sistemas externos, como el de visión.

- **Una plataforma de teleoperación** que incluye un sistema CAD con los datos del buque que se encuentra bajo operación y sus parámetros. La plataforma es capaz de controlar y coordinar hasta diez robots con el objeto de optimizar la calidad final obtenida y los tiempos de operación.

- **Sistemas de visión** que inspeccionan las superficies del casco, determinan las zonas a chorrear y proporcionan las trayectorias que deben seguir cada uno de los robots. También comprueban la calidad final obtenida en el chorreado. Según el robot considerado, la función de cada sistema de visión es diferente. A modo de ejemplo, en el caso de la grúa articulada, sirve para alinear la herramienta según el contorno del casco, mientras que en el caso de spot blasting con la mesa XYZ el sistema de visión determina la matriz de puntos a chorrear. El sistema de visión está descrito en (Navarro, 2006).

- **Un sistema de reciclado de la granalla y separación de los residuos** para eliminar y empaquetar los productos resultantes del chorreado.

- **Una plataforma de monitorización** que facilita servicios tales como planificación, work-flow y simulación de las operaciones de chorreado, manejo de la bases de datos del sistema, control de operadores, etc.

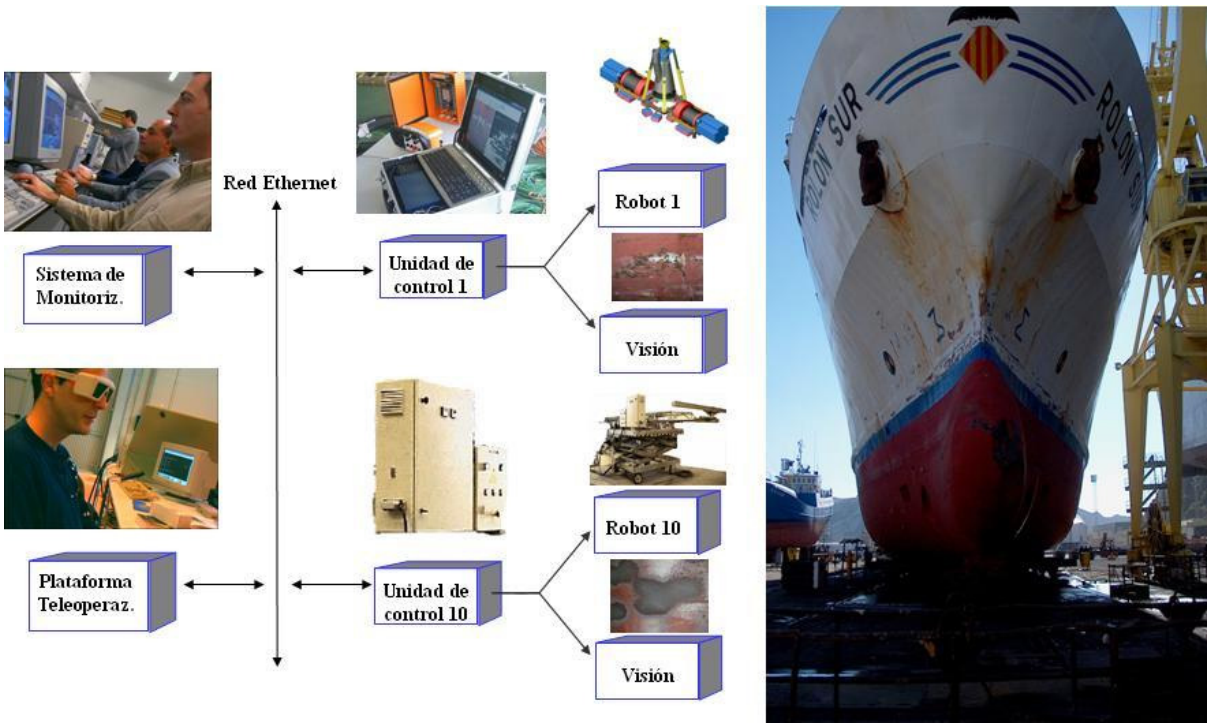


Figura 8. Esquema global del sistema EFTCoR.

5. ARQUITECTURA DEL SISTEMA DE CONTROL.

Uno de los factores claves en el éxito del sistema EFTCoR es la arquitectura de control. Dicha arquitectura se estructura jerárquicamente en tres niveles.

El nivel superior se corresponde con el sistema de monitorización. Este nivel se ocupa de la gestión global de las tareas de mantenimiento. Es un sistema de información que reparte el trabajo y monitoriza su grado de cumplimiento. Es ante todo una herramienta de work-flow para los gestores, que genera órdenes de trabajo para los niveles inferiores.

El nivel intermedio se corresponde con la plataforma de teleoperación. Esta plataforma es la adaptación y extensión de una ya previamente utilizada para teleoperar robots de servicios en centrales nucleares (Iborra, *et al.*, 2003). Su desarrollo se basa en el empleo de una arquitectura de referencia que fue diseñada utilizando ingeniería de dominio (Alvarez, *et al*., 2001). No obstante, dicha arquitectura tuvo que ser adaptada de acuerdo con los nuevos requisitos funcionales y no funcionales que brevemente se resumen a continuación:

- La plataforma de teleoperación original sólo consideraba un robot, mientras que la actual considera hasta un máximo de diez robots trabajando simultáneamente. La plataforma de teleoperación debe repartir la carga de trabajo entre los robots de acuerdo con sus capacidades y situación en la zona de trabajo y monitorizar en línea la ejecución de los trabajos. Este nuevo requisito supuso la extensión de los módulos funcionales de la arquitectura original, sin implicar grandes cambios estructurales.
- En los sistemas originales, los robots eran totalmente teleoperados, mientras que algunos de los robots actuales tienen un grado de decisión y autonomía bastante elevado, siendo mucho más significativo el componente reactivo en su comportamiento. Estos nuevos requisitos implican cambios estructurales profundos en la arquitectura de control. Para evitar estos cambios se ha permitido que puedan introducirse comandos de muy alto nivel que son resueltos en las unidades de control locales de cada robot.

El nivel inferior está constituido por cada una de las unidades de control locales de los robots de servicio. Cada unidad de control tiene su propia arquitectura adaptada a su tipo de comportamiento, desde la teleoperación pura hasta grados de autonomía muy altos. Este aspecto representaba un nuevo reto tecnológico, en el sentido de ser capaces de reutilizar bloques funcionales completos en robots con arquitecturas de control muy diferentes. Esto nos llevó a desarrollar un marco arquitectónico para desarrollar unidades de control (ACROSET) (Álvarez, *et al.*, 2006), en el cual ha sido fundamental utilizar conceptos avanzados de la ingeniería del software, en especial el paradigma de desarrollo basado en componentes (Brown, *et al*, 1998) que se describe en el siguiente apartado.

El esfuerzo de ingeniería ha ido encaminado ante todo a integrar soluciones y herramientas software ya existentes para proporcionar un todo robusto y eficiente. El control a nivel de servomecanismos de los robots mencionados es relativamente sencillo, estando la complicación fundamentalmente en la integración de componentes software muy diversos (sistema de visión particular de la unidad de control considerada, relaciones con los niveles superiores, sincronismo con sistemas externos como el de reciclado, sistemas sensoriales, controles locales de los ejes y de las herramientas, …). Nuestra preocupación se centra no sólo en qué cosas se hacen, sino en cómo se hacen, de ahí nuestro interés en aplicar los paradigmas más avanzados de la ingeniería del software (arquitecturas software, programación orientada a componentes y *model driven engineering*).

5.1 Marco arquitectónico para las unidades de control (ACROSET).

Un aspecto importante en el diseño de robots es que muchos de los componentes son desarrollados por diferentes equipos de trabajo y deben ser integrados al construir el sistema completo. Con objeto de permitir esta integración es necesario que los componentes hayan sido específicamente diseñados para ser integrados, lo cual favorece el uso de enfoques de desarrollo basados en componentes (Component Based Development, CBD). CBD es un paradigma relativamente nuevo dentro de la ingeniería software que postula la creación de módulos, definidos únicamente en función de sus interfaces (servicios provistos y requeridos), que se ensamblan para construir aplicaciones completas como si fueran piezas de un mecano. Un ejemplo de este enfoque, en otro campo estrechamente relacionado con la robótica, es el de las redes de sensores y nesC. NesC es una extensión de C utilizada para programar redes de sensores que sigue fielmente el paradigma CBD.

El enfoque seguido en este proyecto es incluso más ambicioso, pues pretende combinar componentes pertenecientes a diferentes disciplinas y combinar componentes comerciales (software de control de servomecanismos y algoritmos de visión artificial) con otros desarrollados a medida (máquinas de estado, estrategias de representación de datos,

algoritmos muy a medida, etc.). Para que esto sea posible se encapsula la funcionalidad en módulos, que al estar definidos sólo en términos de sus interfaces, son fácilmente reutilizables de una a otra plataforma de ejecución.

Siguiendo esta filosofía, y como parte del trabajo de investigación del grupo DSIE y de las tesis doctorales realizadas en su seno, resulta el entorno de desarrollo basado en componentes ACROSET.

ACROSET se ha definido siguiendo la metodología ABD (Architecture Based Design Method) (Bachman, *et al.*, 2000) propuesta por el SEI/CMU (Software Engineering Institute of the Carnegie Mellon University) para diseñar arquitecturas software para un dominio de aplicación o para familias de productos.

Entre los objetivos que ACROSET pretende alcanzar pueden destacarse:

- La posibilidad de que diferentes arquitecturas de control puedan compartir los mismos componentes.
- Posibilidad de definir diferentes arquitecturas de control con los mismos componentes, enfocadas bien a sistemas puramente teleoperados o a sistemas reactivos con alto grado de autonomía.
- Una separación clara entre los componentes y la forma en que interaccionan, asumiendo que un mismo componente puede ser utilizado de diferentes formas y circunstancias en diferentes sistemas. Esto permite reutilizar un componente en sistemas muy distintos.
- Permitir la combinación de componentes implementados en hardware y en software y la sustitución de unos por otros según convenga.
- La integración de sistemas externos que aporten servicios (p.e.: sistemas de visión, redes de sensores, etc.) o inteligencia.

La descripción más abstracta posible de ACROSET se muestra en la figura 9. Según dicha figura para cualquier robot y arquitectura se consideran cuatro grandes subsistemas:

- **El susbsistema de control, coordinación y abstracción de dispositivos (CCAS)**, cuyas responsabilidades son la coordinación funcional y control de los dispositivos que componen el sistema, la abstracción o representación del hardware presente en el sistema, la utilización de estrategias de coordinación y control intercambiables y configurables, y ofrecer acceso individual a los distintos mecanismos y dispositivos.
- **El subsistema de inteligencia (IS),** tiene responsabilidad sobre: la supervisión y control de la ejecución de misiones pre-programadas, la realización de ciertos procesamientos autónomos sencillos y algunas tareas de planificación, y la generación de ciertos comportamientos reactivos que deban combinarse con las órdenes del operador (por ejemplo, evitación de obstáculos).

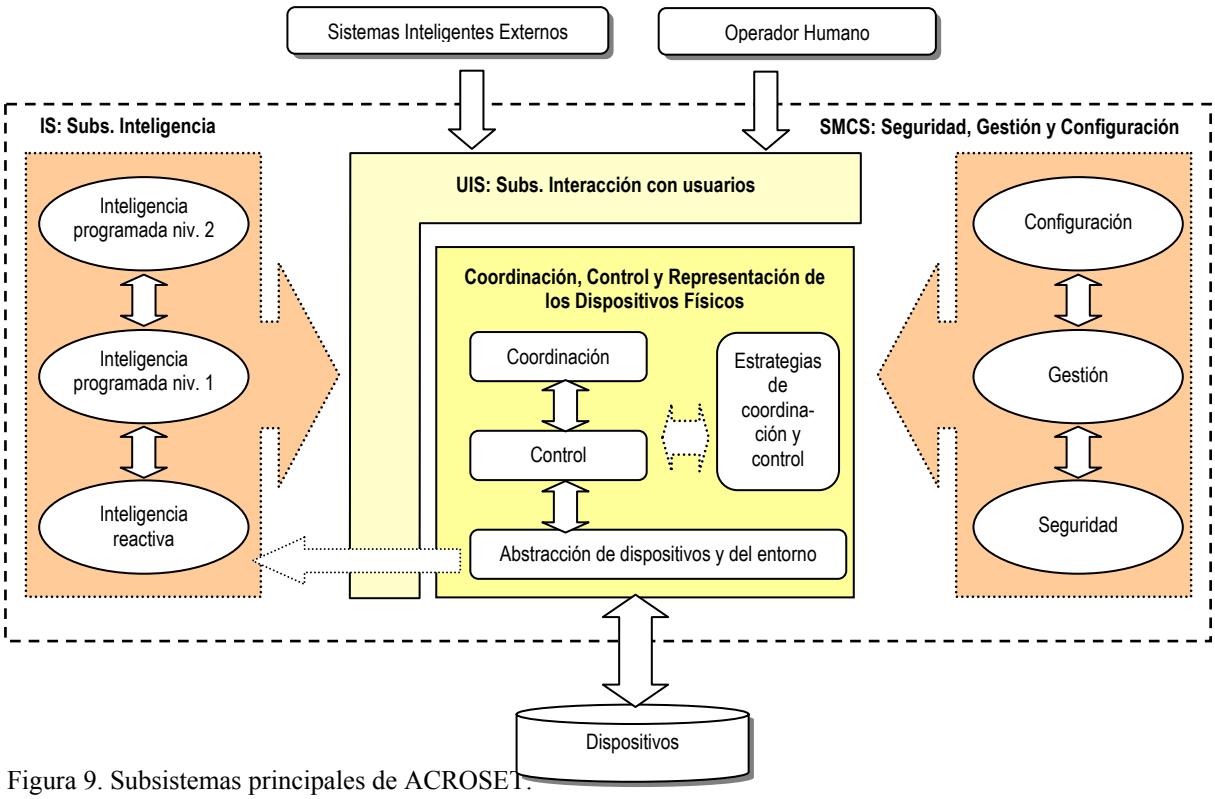


Figura 9. Subsistemas principales de ACROSET.

- **El subsistema de interacción con los usuarios (UIS)** proporciona al operador y a otros posibles usuarios del CCAS acceso a los servicios del sistema y mostrar el estado del mismo. Interpreta los comandos que llegan de los usuarios y los redirecciona al subsistema o componente correspondiente. Interpreta sí los comandos son viables según el estado del sistema. Gestiona el modo de control activo en el sistema, y se encarga del arbitraje de las órdenes que lleguen a la vez de distintos usuarios.
- **El sistema de seguridad, gestión y configuración (SMCS)** monitoriza el estado y correcto funcionamiento del resto de subsistemas. Realiza diagnósticos al arranque del sistema. Informa al usuario ante eventos que puedan producirse en el diagnóstico. Chequea periódicamente el funcionamiento de dispositivos y componentes del sistema. Implementa políticas de tolerancia a fallos. Monitoriza el sistema en su conjunto. Almacena el estado y diagrama de estado global del sistema y gestiona la ejecución del mismo, incluyendo la puesta en marcha y finalización de la aplicación. Arranca y para los sistemas según una secuencia adecuada. Ubica los diferentes componentes de la aplicación. Se encarga de la activación y desactivación de subsistemas. Ofrece una interfaz para la configuración de componentes, parámetros de funcionamiento y estrategias de control del sistema. Ofrece una interfaz para la instalación y desinstalación de componentes, y finalmente comprueba la consistencia de las configuraciones e interacciones entre subsistemas.

5.2 Instanciación de ACROSET para el EFTCoR.

A modo de ejemplo se van a mostrar dos arquitecturas definidas con ACROSET una correspondiente al controlador de la mesa XYZ y otra correspondiente al vehículo escalador. Ambas arquitecturas se corresponden con el subsistema de control, coordinación y abstracción de dispositivos descrito en el apartado anterior. El resto de subsistemas se definirían de una manera análoga. Obsérvese en las figuras 10 y 11 cómo se hacen explícitos tanto los componentes como sus interacciones y conexiones, siguiendo el enfoque CBD antes mencionado. Los componentes fundamentales que se observan en las figuras se corresponden fundamentalmente con controladores de dispositivos físicos (control de uno o varios ejes, de una herramienta, etc.). Cuando un controlador se refiere a un solo accionamiento recibe el nombre de SUC (Simple Unit Controller), cuando coordina varios controladores más simples recibe el nombre de MUC (Mechanism Unit Controller). El agregado de SUCs y MUCs según una cierta arquitectura es el controlador del robot (que en la terminología ACROSET recibe el nombre de RUC, Remote Unit Controller). A veces los controladores únicamente encapsulan el acceso a un dispositivo físico real, como es el caso del MUC de la mesa XYZ (Fig. 10) que se corresponde con la interfaz de acceso a un controlador comercial de SIEMENS para un autómata 315-2DP. En el caso del vehículo escalador el MUC está constituido por módulos software escritos en Ada95 y se ejecuta sobre un PC empotrado.

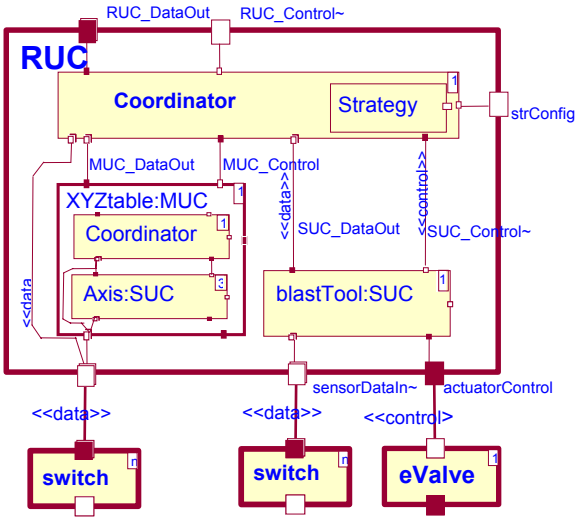


Figura10. Componentes CCAS de la unidad de control de la Mesa XYZ.

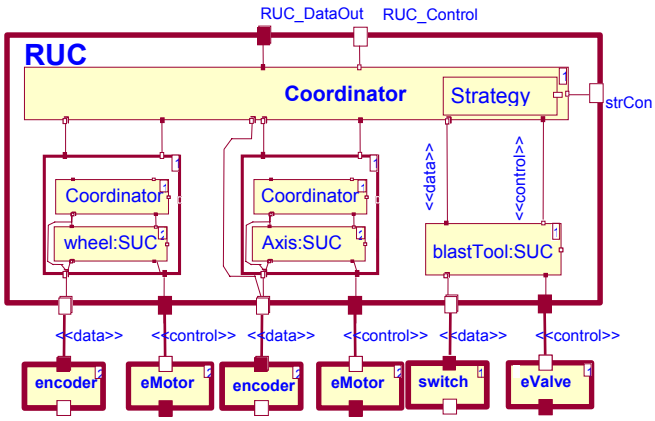


Figura 11. Componentes CCAS del robot escalador.

Cada componente define los servicios que ofrece y los que requiere a través de sus interfaces. Estas interfaces se corresponden con puertos, a través de los cuales se conectan los componentes usando conectores. Para que dos puertos se puedan conectar uno debe requerir los servicios que ofrezca el otro. Puesto que los servicios pueden requerirse de diferentes maneras, el conjunto puerto

conector puede encapsular un protocolo de comunicaciones.

Este enfoque orientado a componentes permite diseñar el software de control a partir de componentes software que se ensamblan como si fueran componentes hardware, tal y como se puede apreciar en las figuras 10 y 11. Este modelo abstracto que representa ACROSET se ha concretado sobre dos plataformas. Por un lado ACROSET se ha materializado sobre la infraestructura de desarrollo que proporciona SIEMENS usando un PLC 315-2DP y Profibus, siendo el entorno de desarrollo STEP7. Las torres y la mesa XYZ, se han implementado usando esta plataforma. La otra materialización de ACROSET ha sido sobre un PC empotrado, haciendo uso del lenguaje Ada 95.

Para que este enfoque sea factible es necesario traducir cada concepto de ACROSET a código compilable estableciendo correspondencias entre los SUCs, MUCs y RUCs y sus conexiones con elementos propios del lenguaje de programación que vaya a utilizarse (bloques de función, clases, llamadas a procedimiento, etc.). Como se acaba de mencionar, esto se ha realizado en el caso del EFTCOR considerando dos lenguajes de programación: STEP 7 de Siemens y Ada95. Sin entrar en muchos detalles, puede decirse que en el primer caso los componentes ACROSET se han traducido a bloques de función y las conexiones entre componentes se corresponden bien con llamadas a función, o bien con accesos a las tablas de entrada/salida del PLC. En el caso de Ada95 los componentes se corresponden con clases y las conexiones se llevan a cabo empleando patrones de diseño propios de la orientación a objetos, como el patrón OBSERVADOR (Gamma, 1995). Otros aspectos, como la concurrencia, se llevan a la práctica en STEP7 mediante un ejecutivo cíclico y en Ada95 mediante el empleo del patrón de diseño OBJETO ACTIVO (Schmidt, 2000). En el caso del EFTCOR las traducciones de componente a código se han realizado manualmente. La implementación de los componentes ha sido mucho más sencilla en Ada 95, debido al nivel de abstracción que proporciona el lenguaje. En el caso de STEP 7, la implementación ha sido mucho más laboriosa dada la falta de soporte del lenguaje para los conceptos anteriormente mencionados. En la actualidad se está trabajando para traducir de forma automática un diseño ACROSET a un código ejecutable en Ada95.

El hecho de utilizar ACROSET nos ha permitido:

- Diseñar los controladores y sus interacciones a un nivel de abstracción independiente de la plataforma. El diseño es así reutilizable entre diferentes plataformas y mucho más claro y fácil de entender, ya que los componentes y sus interacciones se muestran de forma explícita.
- Reutilizar los componentes abstractos que define ACROSET en sistemas que utilizan diferentes plataformas de ejecución.
- Reutilizar los mismos componentes concretos en sistemas con diferentes arquitecturas que comparten la misma plataforma de ejecución. Por ejemplo, se han utilizado componentes comunes en las dos torres y en la mesa XYZ, si bien con diferente parametrización. Esto ha sido posible simplemente instanciando el componente y conectándolo con sus vecinos.
- Facilitar la extensión de los sistemas con funcionalidad adicional, ya que se simplifica en gran medida la adición y sustitución de componentes.
- Reducir significativamente los tiempos de desarrollo.
- Abrir una puerta a la reutilización de diseños y componentes a mayor escala, incorporando los principios de la ingeniería dirigida por modelos a la robótica de servicios.

6. CONCLUSIONES

En este artículo se han descrito una serie de robots de servicios para limpieza de cascos de buques que trabajan cooperativamente para dar solución a un problema que preocupa a la industria europea de reparación de barcos.

Se ha presentado además un marco arquitectónico de desarrollo basado en componentes para el desarrollo de las unidades de control de este tipo robots (ACROSET) y su empleo en dos de los robots de servicio desarrollados. Se ha intentado mostrar como el uso de este enfoque ha permitido definir las arquitecturas de los sistemas de control, reutilizar componentes y disminuir por tanto los tiempos de desarrollo.

La mayor dificultad para seguir este enfoque ha sido la falta de soporte de los lenguajes y plataformas para seguir un estilo de programación basado en componentes, lo que ha requerido un gran esfuerzo para definir e implementar los mismos en las correspondientes plataformas.

AGRADECIMIENTOS

El DSIE agradece a la Unión Europea (V PM), al gobierno español (CICYT) y al gobierno regional de Murcia (Programa Séneca) el apoyo prestado a estos trabajos.

REFERENCIAS

Álvarez B, A Iborra, A Alonso,J.A. de la Puente (2001). Reference architecture for robot teleoperation: Development details and practical use Vol. 9 (4). Pp.395-402. *Control Engineering Practice*, Apr 2001.

Álvarez, B., F. Ortiz, J.A.Pastor, P. Sánchez, F. Losilla y N.Ortega (2006). Arquitectura para Control de Robots de Servicio Teleoperados. Pp. 79-89. *Revista Iberoamericana de Automática e Informática Industrial*. *RIAI*, Vol. 3, Num. 02, Abril 2006.

Bachman, F., L. Bass, G. Chastek, P. Donohoe y F. Peruzzi (2000). "The Architecture Based Design Method" *TR CMU/SEI-200-TR-001*.

Brown, A.W., Kurt C. Wallnau. The current state of CBSE. *IEEE Software*, Vol. 15, nº5, Sep/Oct 1998.

EFTCoR (2002) "Environmentally Friendly and Cost-Effective Technology for Coating Removal (EFTCOR)". *Fifth Framework Programme, European Community*, Subprogram Growth ref. GRD2-2001-50004. www.dsie.upct.es

Gamma, E., R. Helm, R. Johnson y J. Vlissides (1995). "Design patterns: elements of reusable object-oriented software". *Addison-Wesley*, New York.

Goldie, B (1999a) "A comparative look at dry blast units for vertical surfaces", *Protective Coating Europe*. pp. 26-28, July 1999.

Goldie, B (1999b) "Comparing robotic units made to clean vertical surfaces with UHP waterjetting", *Protective Coating Europe*. pp. 26-28, Sept 1999.

Iborra, A, J.A.Pastor, B.Álvarez, C.Fernández y J.M. Fernández-Meroño (2003). "Robots in Radioactive Environments"**.** Volumen10,Nº4 Pp.12-22. *IEEE Robotics & Automation Magazine*. Dec. 2003

Navarro, P., J. Suardiaz, P. Alcover, R. Borraz, A. Mateo y A. Iborra (2006) "Teleoperated Visual Inspection System for Hull Spot-Blasting". *32nd Annual Conference of the IEEE Industrial Electronics Society. IECON'2006.* pp. 3845-3850, Paris, Nov 2006.

Schmidt, D., M. Stall, H. Rohnert y F. Buschmann (2000). "Pattern oriented software architecture, Vol .II". *John Wiley and Sons*, New York.

Selic, B., G. Gullekson y P.T. Ward (1994). "Real-time Object-Oriented Modeling". *John Wiley and Sons*, New York.

Smith, A (1999) "Marine coating: the coming future" *Protective Coating Europe*. pp. 18-20, Feb 1999.

Wayle, A (1998). "Trends for European shipyards". *30th International Conference on Automated Applications*, pp. 126-130. Yohaio, Japan, 1998.