Titulació:

Enginyeria en Automàtica i Electrónica Industrial

Alumne:
Marc Güell Brocal

Títol PFC:
Disseny d'una estructura de recol·lecció de dades d'un robot ABB140

Director del PFC:
Juan Carlos Hernández Palacín

Convocatòria de lliurament del PFC
Gener 2012

Contingut d'aquest volum: **-MEMORIA-**

INDEX GENERAL

Escola Tècnica Superior d'Enginyeries
Industrial i Aeronàutica de Terrassa
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Disseny d'una estructura de recol·lecció de dades d'un robot ABB140

VOLUM I – MEMÒRIA

Alumne: **Marc Güell Brocal**

Director de Projecte: **Juan Carlos Hernández Palacín**

Gener 2012

TAULA DE CONTINGUTS

TAULA DE FIGURES

TAULA DE TAULES

RESUM

Amb el present treball s'exposa el desenvolupament i implementació d'un sistema de monitorització de les dades manipulades per la controladora S4 del robot IRB140 del laboratori de robòtica i CIM de l'edifici TR11 de la ETSEIAT.

Les dades s'adquiriran des d'una aplicació que actuant com a client OPC accedirà a les dades de la controladora S4 la qual actuarà com a servidor OPC.

També es desenvoluparà i implementarà una base de dades amb les seves respectives taules on es desaran totes les dades monitoritzades segons el tipus de dades.

Al mateix temps, l'aplicació permetrà visualitzar la base de dades en qualsevol moment sempre i quan no s'estigui realitzant la monitorització.

RESUMEN

Con el presente trabajo se expone el desarrollo e implementación de un sistema de monitorización de las dadas manipuladas por la controladora S4 del robot IRB140 del laboratorio de robótica y CIM del edificio TR11 de la ETSEIAT.

Los datos se adquirirán desde una aplicación que actuando como cliente OPC accederá a los datos de la controladora S4 que realizará las funciones de servidor OPC.

También se desarrolla e implementa una base de datos con sus respectivas tablas donde se almacenará el todos los datos monitorizados dependiendo de su condición.

Al mismo tiempo, la aplicación permitirá visualizar esta base de datos en cualquier instante, siempre y cuando no se esté realizando la monitorización.

ABSTRACT

With the present work, we exposed the development and implementation of a monitoring system for data manipulated by the S4 controllers of the IRB140 robot located in the robotics and CIM laboratory of the TR11 building at ETSEIAT.

The data will be acquired from an application that acting as an OPC client which will have access to the S4 controller data which will perform as an OPC server.

It also develops and implements a database with their respective tables which will do the storage of all data depending on their type. Furthermore, the application makes possible display the database at any time while no monitoring is being performed.

AGRAÏMENTS

En el pla personal, no puc deixar d'agrair als meus pares, Jordi i Cristina, tot el que han fet per a que pogués arribar fins aquest punt. Als meus germans, Bernat, Jon i Raquel que, a distància, també han aportar el seu granet de sorra.

A una peça del engranatge que ha sabut encaixar d'una forma especial a la peça que sóc jo. A Nerea li agraeixo estar aquí per què gràcies a ella les llargues hores de treball es fan menys pesades sabent que al final del dia es troba ella.

A tots ells, quantes vegades els hi agraeixi tantes es quedaran curtes.

Servint de pont entre el pla personal i el professional, he de mencionar als meus companys de carrera.

Finalment, vull donar el meu més sincer agraïment a Don Juan Carlos Hernández Palacín, per haver-me dirigit aquest projecte de fi de carrera i estar sempre al costat quan l'he necessitat. Per totes les atencions, pel temps que m'ha dedicat i, sobretot, pel seu suport.

1. OBJECTE

El present projecte es redacta com a Projecte Final de Carrera a l'Escola Tècnica Superior d'Enginyeries Industrial i Aeronàutica de Terrassa (ETSEIAT) de la Universitat Politècnica de Catalunya (UPC) i té per objecte el disseny i implementació d'una aplicació capaç de mostrar i monitoritzar les senyals i/o variables que s'utilitzin en el robot ABB IRB140 del qual es disposa en el laboratori CIM de l'edifici TR11 de l'escola ETSEIAT mitjançant el servidor OPC del propi robot, així com incorporar la capacitat de monitoritzar al mateix temps dades disponibles d'altres servidors OPC.

2. ABAST

En aquest projecte es realitzaran totes les configuracions i eines necessàries per a fer la monitorització de les variables implementades en llenguatge RAPID per als programes que s'han d'executar dins del robot ABB IRB140, així com de les seves senyals digitals d'entrada i sortida.

Al mateix temps, donant sentit a l'amplia capacitat d'interacció amb diferents dispositius de la comunicació OPC també es dotarà a l'eina la capacitat de connectar-se amb altres servidors OPC.

Per a facilitar l'estudi posterior de les senyals llegides i per dotar al projecte d'un caràcter més professional també s'implementarà dins l'eina de monitorització l'enregistrament de les dades a una base de dades desenvolupada mitjançant el programari Microsoft SQL Server 2005.

3. JUSTIFICACIÓ

Des de ja fa temps que en el món de la industria, especialment en l'automatització, es treballa de forma conjunta amb eines, comunicacions i dispositius fabricats i implementats per diferents fabricants, on els problemes i dificultats en la seva posta a punt no sorgeixen de la preparació i funcionament individual del propi element sinó en la seva interacció amb la resta de components del sistema.

Per aquest motiu es tria l'estàndard OPC per a comunicar i utilitzar les diferents eines i aplicacions utilitzades en aquest projecte, estàndard que va néixer farà poc més d'una dècada i que té com a principal impulsor la companyia Microsoft, però que des dels seus inicis comptava amb un gran nombre d'empreses col·laboradores i que amb el pas del temps el gruix de membres a crescut enormement.

D'aquesta manera, multitud d'eines i programari desenvolupats per els diferents fabricants i participants d'aquest estàndard es poden entendre i funcionar correctament, que és el que es pretén amb aquest projecte: desenvolupar diferents eines i unificar-les de forma que treballin conjuntament per a que siguin de gran utilitat.

Així doncs, es pretén desenvolupar un eina que permeti utilitzar l'estàndard OPC de forma que l'aplicació que es porti a terme pugui en posteriors treballs fer la monitorització d'altres dispositius que disposin de servidor OPC així com possibilitar la lectura de les dades i senyals del robot ABB IRB140 des de qualsevol altre aplicació que actuï com a client OPC utilitzant el propi servidor OPC del robot.

4. ESPECIFICACIONS BÀSIQUES

A continuació es redacten les especificacions bàsiques que haurà de complir l'eina produïda:

- Configurar el servidor OPC de la controladora S4 per a que estiguin disponibles les targetes d'entrades i sortides digitals,
- Configurar el servidor OPC de la controladora S4 per a que estiguin disponibles les variables numèriques del programa en RAPID utilitzat,
- El client OPC serà capaç tant d'accedir al servidor OPC de la controladora S4 com de la resta de servidors disponibles ala xarxa,
- El client OPC ha de poder accedir amb un temps suficient com a mínim per a establir una connexió en temps real,
- De la mateixa forma que té permesa la lectura, el client OPC podrà escriure si fos necessari en qualsevol de les senyals monitoritzades,
- La creació de les taules i base de dades necessàries per enregistrar els valors que prenguin les senyals monitoritzades,
- El client OPC tindrà connexió amb la BBDD per a poder mostrar les taules amb els valors emmagatzemats sempre i quan l'aplicació tingui la comunicació client-servidor aturada.

5. EINES UTILITZADES

Amb el present capítol es pretén raonar i justificar l'elecció de les eines utilitzades per a portar a terme el projecte final de carrera.

El programari utilitzat es farà servir amb les següents finalitats:

- Programació de l'aplicació principal,
- Configuració del robot,
- Programació de l'aplicació de manipulació del robot, i
- Creació de les bases de dades.

Per a cada una d'aquestes finalitats es farà ús d'un programari determinat per desenvolupar cada part.

5.1. Programari de l'aplicació principal

Per a desenvolupar l'eina de visualització i enregistrament de les dades provinents dels diferents servidors OPC s'ha triat finalment utilitzar el programari Visual Basic 6 (VB6).

VB6 és un programa que permet desenvolupar aplicacions gràfiques, des de les més senzilles fins a les més complexes, a partir del llenguatge orientat a objectes i esdeveniments Visual Basic al mateix temps que permet incorporar totes les característiques i elements d'un programa de Windows.

Un objecte té característiques o propietats que defineixen el seu aspecte, per exemple color o mida, però també té mètodes i esdeveniments que són elements significatius de l'objecte.

Al contrari que en altres llenguatges, els errors en Visual Basic no es generen tan sovint isolen ser més fàcils de corregir ja que cada objecte treballa de forma independent, posseeix el seu propi codi font.

En una primera presa de decisions es va intentar migrar a la versió superior Visual Basic .NET (VB.NET), la qual permetia passar a la nova

forma d'integració amb els nous sistemes operatius i la resta de programes .NET gràcies a la utilització comuna de *Frameworks.NET*.

A aquesta nova manera d'integració que evita alguns dels problemes derivats de la utilització de llibreries específiques de la màquina utilitzada en el moment de desenvolupament se li suma una millora en la gestió d'errors en el llenguatge .NET.

Altres avantatges i millores adoptades en aquesta nova versió són la total capacitat d'orientació a objectes així com la incorporació d'herència, la millora del sistema d'accés a dades amb el nou ADO.NET o la programació de Web Forms específics per a aplicacions per a Internet.

No obstant, tot i aquest conjunt de millores i avantatges esmentats, l'eina triada per a fer el desenvolupament de l'aplicació de visualització i enregistrament de dades és la primera de les versions citades: Visual Basic 6.

Això és degut a que quan es va iniciar el projecte, les llicències disponibles al Laboratori de Robòtica i CIM només es disposava de les llicències per a la controladora S4 i que sols eren compatibles fins a la versió de VB6, i no pas per a VB.NET.

Per aquest motiu, es va triar el programari Visual Basic 6 com a eina de desenvolupament de la aplicació en comptes de la versió .NET.

5.2. Configuració del robot

Per a fer tota la configuració de la controladora S4 per a que sigui capaç de comunicar-se mitjançant comunicació OPC s'ha triat el paquet WebWare SDK.

Aquest paquet és un conjunt de potents eines que simplifica la comunicació entre les aplicacions de PC i els controladors de robots d'ABB i que es capaç d'interactuar amb llenguatge Visual Basic, Visual C++ o seqüències d'ordres (*scripts*) mitjançant controls ActiveX.

El Webware SDK es basa en el mòdul de comunicació Interlink d'ABB, un component fonamental per a la comunicació amb el controlador de robots d'ABB. L'Interlink s'instal·la conjuntament amb el software Webware SDK.

A continuació es mostren els diferents productes que es basen en Interlink, entre ells el WebWare SDK:

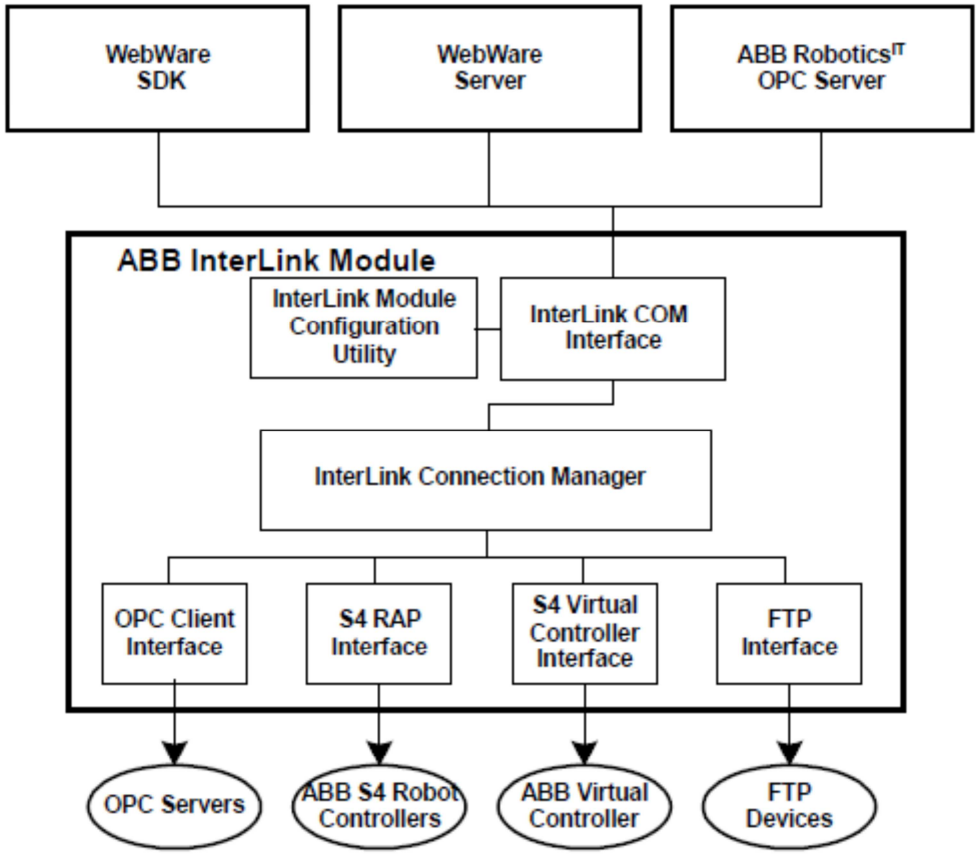


Figura 1. *Arquitectura dels productes basats en l'Interlink Module i la seva comunicació amb WebWareSDK i altres complements i controls.*

El mòdul d'ABB Interlink crea, manté i controla l'estat de l'enllaç de comunicació amb el robot. Aquest mòdul notifica a l'aplicació de qualsevol canvi en l'enllaç de comunicació a través d'esdeveniments i propietats.

Una de les prestacions que aporta aquest mòdul és la utilitat de configuració del mòdul (ABB InterLink Module Configuration Utility, ICU).

El mòdul també manté el nom de les senyals d'entrada/sortida, el nom de taules i el valor actual de totes les senyals, al mateix temps que genera esdeveniments quan qualsevol variable canvia el seu valor.

El Webware SDK s'utilitza mitjançant la incorporació dels controls ActiveX en una aplicació. El funcionament de cada control es configura

a través de les propietats del control un cop inserit a l'aplicació. Inclou quatre ActiveX específics per a robots: *Helper*, *Button*, *Pilot Light* i *Label*.

En el nostre cas s'utilitzarà únicament el control *Helper*, motor d'interfície principal del Webware SDK. Actua com un ajudant invisible (en temps d'execució) que proporciona mètodes, propietats i esdeveniments per exposar tota la interfície de comunicació S4.

5.2.1. Interlink module

El mòdul Interlink d'ABB conté tots els components clau necessaris per comunicar-se amb un o més controladors de robots i altres equips de producció, i està penat per facilitar les comunicacions bàsiques de suport necessàries per a tots els components i dispositius compatibles amb Webware.

Com es pot veure en la Figura 1. Arquitectura dels productes basats en l'Interlink Module i la seva comunicació amb WebWareSDK i altres complements i controls., el mòdul es compon de diferents unitats:

- InterLink Module Configuration Utility: aquesta aplicació s'utilitza per portar a terme la configuració del mòdul Interlink de cara a la comunicació del PC amb el robot, amb servidors FTP i OPC, i fer l'administració de llicències
- InterLink COM Interface: interfície COM privada i que es recomana no fer-ne ús en les aplicacions ja que pot ser que no sigui suportada. ABB no assegura la comptabilitat ni l'estabilitat.
- InterLink Connection Manager: s'utilitza per a connectar les aplicacions client a la interfície adequada en cada cas. Principalment s'encarrega de fer el servei i seguiment de l'estat de totes les aplicacions d'usuari, mantenir l'estat de les interfícies configurades, gestionar la comunicació amb tots els dispositius connectats i dirigir les sol·licituds de servei a la interfície apropiada.
- S4 RAP Interface: implementa la interfície de comunicacions amb la controladora S4 del robot utilitzant el protocol S4 RAP. El mòdul Interlink s'encarrega de crear una nova instància d'aquesta interfície per a cada controlador S4.

- S4 Virtual Controller Interface: implementa una interfície COM per al controlador virtual S4 mitjançant un subconjunt de comandaments proporcionada pel component de RAP S4.
- OPC Client Interface: implementa la interfície COM entre l'Interlink i un Servidor OPC. El dispositiu d'OPC pot ser connectat a un servidor OPC Data Access o un servidor OPC Alarmes i Esdeveniments.
- FTP Device Interface: implementa la interfície COM entre l'Interlink i un servidor FTP qualsevol sempre i quan el dispositiu sigui compatible amb FTP.
- WebWare SDK: és el paquet software anteriorment mencionat el qual ens proporciona els controls ActiveX necessaris per a comunicar les aplicacions desenvolupades amb la controladora del robot, a més d'incloure també el servidor OPC i la controladora virtual.
- WebWare Server: ofereix una web basada en el control de la producció i elaboració d'informes. Està implementat com un lloc web completament desenvolupat que presenta la informació recopilada a través de qualsevol mòdul Interlink adjunt. La informació es presenta en una sèrie de pàgines web dinàmiques que es poden veure utilitzant el navegador web Microsoft Internet Explorer.
- ABB Robotics OPC Server: proporciona la interfície de l'estàndard OPC per al mòdul Interlink, permetent l'accés de clients OPC tant d'accés a dades (DA) com d'alarmes i esdeveniments (AE) al servidor.

5.3. Programació de l'aplicació de manipulació del robot

Utilitzant una mecànica semblant al raonament seguit per a triar l'eina de desenvolupament de l'aplicació principal, el programari escollit per a fer aquesta aplicació de manipulació del robot és Visual Basic 6 per motius de compatibilitat entre aquest programari i la controladora S4.

Per tant, el motiu principal és que per aquesta aplicació es farà ús del control *Helper*, component que com s'ha esmenat anteriorment incorpora mètodes, propietats i esdeveniments que permeten la interacció amb el robot, i que aquest és només compatible amb la versió 6.0 de Visual Basic.

Si s'hagués disposat tant de les llicències oportunes com dels nous controls per a la nova controladora capaç d'interactuar mitjançant entorn .NET podríem haver utilitzat el programari Visual Basic.Net.

5.4. Creació de les bases de dades

Per a la creació de les bases de dades s'utilitza el programa Microsoft SQL Server 2005. Una de les raons principals és que aquesta plataforma complia amb la integració al llenguatge .NET amb el qual estaria basat el projecte si s'hagués acabat desenvolupant les aplicacions mitjançant Visual Basic .NET.

Una altra raó de pes és que el programari era el ja disponible al laboratori amb totes les llicències en correcte funcionament. A més, ja es tenia certa experiència en l'ús d'aquest programari i, per tant, és més còmode per a l'autor la seva utilització de cara al desenvolupament del conjunt del projecte.

També ha ajudat a la presa de decisió final per a l'ús d'aquest programari que a la instal·lació s'acompanyés del software *Microsoft SQL Server Management Studio Express* el qual permet fer la creació, edició i configuració tant de les bases de dades com de les seves taules de forma més fàcil i intuïtiva.

A més, *SQL Server 2005* està basat en l'estàndard SQL amb el qual, des de les aplicacions desenvolupades amb Vsiual Basic podrem crear noves taules, inserir informació o actualitzar a les ja disponibles o eliminar aquelles que ja no siguin necessàries, entre altres funcions compatibles.

6. INTRODUCCIÓ

A partir d'aquest punt i en els capítols següents es redacta els passos seguits per a la confecció del projecte.

El sistema de monitorització no serà altre que una aplicació elaborada amb el programari Visual Basic 6 la qual haurà d'actuar com a client OPC. Aquest client, es comunicarà amb el servidor OPC de la controladora S4 disponible al laboratori i que serà l'encarregada de comunicar-se i actuar sobre el robot IRB140.

Fent ús de l'alta capacitat de comunicació amb altres dispositius que ens ofereix el protocol OPC, també es disposarà i permetrà al mateix temps establir connexió amb qualsevol altre servidor OPC que es trobi a la xarxa i fer la monitorització de les seves dades.

Aprofitant que es desenvolupa una eina de monitorització, s'ha cregut adient implementar al mateix temps una base de dades on emmagatzemar la informació i les dades que es vagin llegint, de forma que siguin accessibles i permetin la seva visualització per a un estudi posterior, per citar un exemple del seu possible ús.

Finalment, necessitem manipular el robot d'alguna manera, ja sigui mitjançant el comandament de la controladora o bé utilitzant el llenguatge RAPID, llenguatge propi dels robots de l'empresa ABB.

Atenent a aquests primers punts a complimentar, sorgeixen una successió de punts a resoldre abans de començar a desenvolupar l'aplicació en si mateixa.

Primerament, hem de tenir clar quina serà l'estructura de comunicació i el procés a seguir, així com fixar els emplaçaments de les aplicacions i components participants en la comunicació. Un punt a tenir en compte, es que tant l'aplicació de monitorització com la controladora ja tenen el tipus de comunicació especificat: la comunicació client-servidor OPC, establerta com un objectiu a assolir segons l'enunciat del projecte.

Per a respondre tot aquest seguit de primeres qüestions es desenvolupa el 7 Comunicacions, on trobem tant l'arquitectura

implementada com la forma de comunicar-se les diferents parts del projecte i els mètodes utilitzats.

Un cop resolt el tema de les comunicacions, cal configurar el servidor OPC de la controladora S4 que, valgui la redundància, controlarà i manipularà el robot IRB140. Per establir la configuració correcte, farem ús del programari *WebWare SDK Controls Reference* i ens ajudarem en la major part del manual d'usuari que podem trobar en la seva ajuda.

Tots els passos necessaris i que s'han seguit per a configurar el servidor segons la conveniència del projecte està detallada en el 8 Configuració del servidor OPC. En passos resumits, primer s'haurà de crear un alies, nom, per al servidor i a continuació subscriure totes aquelles variables que vulguin ser utilitzades per a estar disponibles al servidor.

Després del servidor OPC, el següent pas lògic és el desenvolupament del client. Aquest, serà directament l'aplicació de monitorització. Com l'aplicació serà implementada necessitem conèixer com establir el tipus de comunicació OPC des de VB6 actuant com a client.

Els complements necessaris per al programari els podem trobar en el subapartat 7.2.2 Client OPC mentre que el procés d'establiment de comunicació amb el servidor el podem trobar en el subapartat següent, 7.2.3. Comunicació client-servidor.

El codi necessari per a comunicar-nos amb el servidor es troba explicat en al final del mateix subapartat, des de l'establiment de la connexió amb el servidor fins a l'intercanvi de dades.

A continuació, un cop tenim el sistema preparat per accedir i rebre les dades de forma correcte, cal implementar la monitorització de les dades tant en temps real com en un futur.

No obstant, per al cas d'anàlisis de les dades i poder visualitzar-les en un temps posterior a l'execució necessitem fer ús d'una base de dades que reculli els valors de les variables cada cert temps o quan aquests siguin modificats.

Per a aquest punt trobem dues parts a resoldre: una primera encarada a la construcció i creació de la nova base de dades on s'han d'emmagatzemar totes aquestes dades, i una segona que és la

implementació de la interacció amb l'aplicació de monitorització de dades.

Per al primer cas, on es fa la implementació de la BBDD mitjançant el programari *Microsoft SQL Server 2005 Express Edition* podem trobar els passos seguits per a la seva resolució en el capítol dedicat explícitament a la construcció de la base de dades, 9 Base de dades.

En el segon cas, el treball es bifurca novament en dues seccions. La primera va destinada a la recopilació de la informació i l'emmagatzematge a la BBDD. Aquest punt es resoldrà en el capítol dedicat a l'aplicació de monitorització.

L'altre punt a desenvolupar per a l'aplicació de monitorització respecte a la base de dades és la visualització de la informació emmagatzemada un cop s'hagi tancat la comunicació OPC entre aplicació i controladora. La solució adoptada està recopilada en el subapartat 7.2.5 Base de dades - Aplicació, on es tracta la forma d'accés a la base de dades i com manipular-la si és necessari, entre altres punts.

Finalment, un cop tinguem clar com comunicar servidor amb client, com monitoritzar les variables desitjades i com fer l'enregistrament de les dades i la seva posterior visualització, hem de fer la manipulació d'aquestes variables d'alguna manera.

Amb aquesta finalitat s'ha dissenyat paral·lelament una aplicació per a manipular el robot. Aqueta aplicació simula una estació de treball formada per un robot i tres màquines que executen diferents processos a un conjunt de peces i, un cop treballades les peces, es deixen sobre una cinta transportadora comuna per a totes elles.

L'aplicació incorpora un comptador de peces processades per a cada una de les diferents màquines, permet interactuar amb elles de forma que es demani l'atenció a màquina o es reiniciï el recompte de peces, i informa de l'estat de robot i màquines.

7. COMUNICACIONS

A continuació s'exposa el sistema de comunicació utilitzat per al desenvolupament del projecte. Primer de tot cal tenir clar quins seran els elements que formaran part del sistema. Atenent a les especificacions i funcionament que ha de complir el projecte, els elements que formaran part del sistema de comunicacions són:

- La controladora S4 del robot ABB IRB140,
- El robot ABB IRB140,
- Un PC on hi hauran les aplicacions de monitorització de dades i de manipulació del robot així com la Base de Dades.

7.1. Arquitectura

L'arquitectura de comunicacions està formada per un servidor OPC resident a la controladora del robot que es podrà connectar també a altres servidors OPC a més del servidor de la controladora, un client OPC que serà l'aplicació de monitorització de dades, el control *Helper* utilitzat en l'aplicació de manipulació del robot utilitzat per comunicar amb la controladora S4 i finalment el propi robot ABB IRB140.

Per al servidor OPC s'utilitzarà principalment el servidor que incorpora la controladora del robot i que s'haurà de configurar prèviament a la seva utilització. Com s'ha comentat anteriorment, aquest residirà dins de la pròpia controladora. Al mateix temps, per ser una aplicació amb possibilitats de connexió amb diferents servidors, també es farà ús dels diferents servidors OPC que es trobin connectats a la xarxa.

Per el que respecta al servidor de la controladora S4, principalment es comunicarà amb l'aplicació de monitorització de dades, la qual actuarà com a client OPC. De forma interna, aquest servidor té accés directe a totes les variables donades d'alta per a accedir-hi del programa RAPID que s'estigui executant.

Les dues aplicacions d'aquest projecte residiran en un mateix PC per motius de llicències amb els components ABB, però per el disseny del

sistema seria factible (i de fet aquest és el seu propòsit) tenir les aplicacions en dos ordinadors diferents.

La primera d'elles, com s'ha comentat, actuarà com a client OPC el qual estarà en permanent contacte amb els diferents servidors per a fer la monitorització de les dades i variables desitjades.

La segona aplicació amb la qual es farà la manipulació del robot utilitzarà el control *Helper*, control *ActiveX* proporcionat per el paquet informàtic *WebWare SDK* el qual ens proporciona un conjunt de mètodes, esdeveniments i propietats per comunicar-se amb la controladora del robot.

Controladora i robot es comuniquen sobre una línia sèrie a través d'Ethernet.

En ambdós casos la connexió entre PC i controladora es realitzarà mitjançant una xarxa LAN Ethernet amb la controladora del robot. La configuració d'adreces d'aquesta xarxa es realitza amb l'aplicació d'ABB *Interlink Module* en el cas de la controladora S4 i des del configurador de xarxa en els PC's.

En aquest cas, s'utilitzarà la configuració disponible i utilitzada en el laboratori per mantenir la utilització i compatibilitat amb els altres usuaris.

Al mateix temps, també participarà en les comunicacions una Base de Dades situada al mateix PC que les aplicacions, la qual es comunicarà amb el client OPC per fer el registre de les variables que s'estiguin monitoritzant. En un entorn real seria més recomanable implementar-la en un ordinador especialitzat i per separat.

A continuació es mostren les figures de l'arquitectura utilitzada en el laboratori i de l'arquitectura desitjada en un entorn industrial:

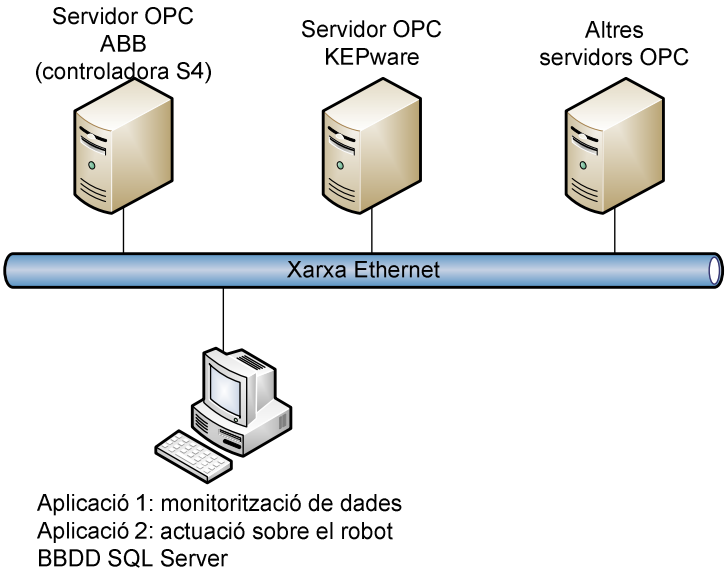


Figura 2. *Arquitectura implementada en el laboratori.*

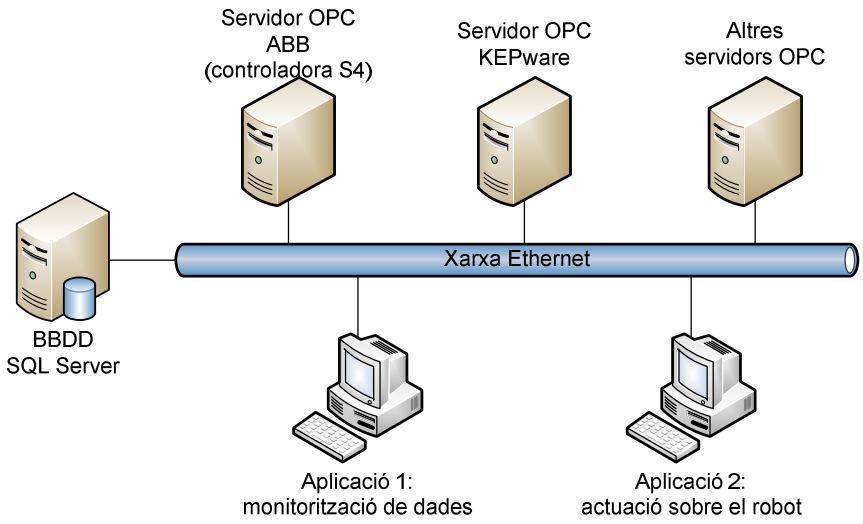


Figura 3. *Arquitectura desitjada per a un entorn real.*

7.2. Comunicació entre aplicacions

Per comunicar les diferents aplicacions i amb la controladora i altres servidors OPC s'utilitzarà el tipus de comunicació client-servidor OPC i el control ActiveX *Helper*.

Pel que fa a la comunicació client-servidor OPC, es diferencien els dos elements en que el servidor és on resideix tot el conjunt de dades amb

el seu valor mentre que el client sol·licitarà l'accés a aquestes dades, no necessàriament han de ser totes de les que disposi el servidor.

A més, el client només podrà modificar les dades si té els permisos necessaris i les dades dins del servidor siguin configurades com a lectura/escriptura. En cas contrari, només se'n podrà fer la lectura.

7.2.1. Servidor OPC

Per definició, un servidor OPC és una aplicació de programari que compleix amb una o més especificacions definides per la OPC Foundation. Un servidor OPC fa d'interfície comunicant d'una banda amb una o més fonts de dades utilitzant els seus protocol nadius (per exemple un PLC o, com en el cas que ens ocupa, la controladora d'un robot) i per l'altra banda amb Clients OPC.

En una arquitectura Client / Servidor OPC, el servidor OPC és l'esclau mentre que el Client OPC és el mestre. Les comunicacions entre el client OPC i el servidor OPC són bidireccionals, el que significa que els clients poden llegir i escriure en els dispositius a través del Servidor OPC.

Hi ha quatre tipus de servidors OPC definits per la OPC Foundation, i són els següents:

1. Servidor OPC DA – Basat en l'especificació OPC Data Access especialment dissenyat per a la transmissió de dades en temps real.
2. Servidor OPC HDA – Basat en l'especificació d'Accés a Dades Històriques que proveeix al Client OPC HDA de dades històriques.
3. Servidor OPC A & E Server – Basat en l'especificació d'Alarmes i Esdeveniments - transfereix Alarmes i Activitats des de el dispositiu cap al Client OPC A & E.
4. Servidor OPC UA – Basat en l'especificació d'Arquitectura Unificada - basat en el set més nou i avançat de la OPC Foundation, permet als Servidors OPC treballar amb qualsevol tipus de dades.

Pel que fa al servidor OPC, utilitzarem el servidor integrat dins la controladora S4 del robot ABB, el servidor ABBS4, prèvia configuració

des del programari *Interlink Module* (8 Configuració del servidor OPC). Per a aquest cas, farem ús principalment del servidor tipus DA, ja que serà on es troben les dades que es monitoritzaran.

Com tots els servidors OPC, aquest s'estructura en 3 nivells: Canal, Dispositiu i Ítem.

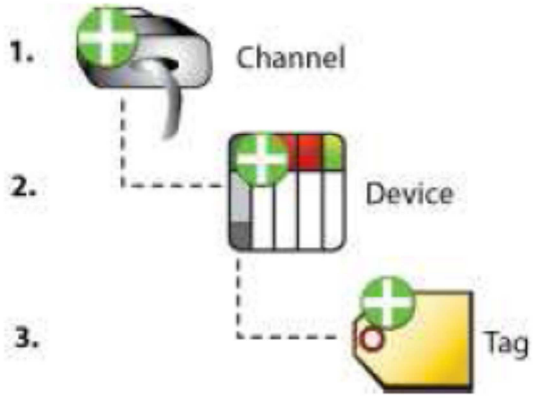


Figura 4. *Estructura en nivells de la informació en un servidor OPC.*

Mentre que el Servidor conté la informació i unifica les dades dins d'un Grup, aquest últim dota d'un mecanisme que conté de forma lògica els ítems i els classifica en públic o local. L'Ítem és un valor, una variable, que pot variar o mantenir-se en el temps. És una adreça específica de les dades i no la font de dades.

7.2.2. Client OPC

El client OPC s'ha desenvolupat mitjançant el programa informàtic Visual Basic 6. Per a fer possible aquest tipus de comunicació cal afegir la referència al projecte de la DLL (llibreria dinàmica) "*OPC DA Automation Wrapper 2.02*". Aquesta llibreria implementa tots els objectes OLE (Object Linking and Embedding) necessaris per el client.

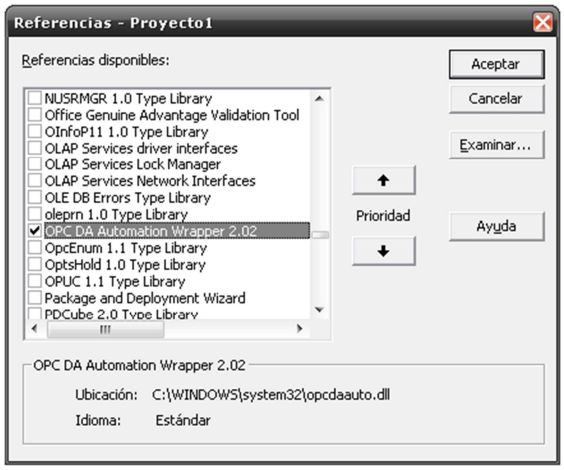


Figura 5. *Afegint referència OPC DA Automation Wrapper 2.02.*

Els objectes utilitzats en aquest projecte proporcionats per la DLL són els següents:

- OPCServer: s'encarrega de crear la connexió necessària al servidor.
- OPCGroups: s'utilitza com a col·lecció per emmagatzemar els grups d'un servidor i agafar la interfície de grups del servidor.
- OPCGroup: agrupa un conjunt d'ítems de forma que aquests comparteixin moment d'activació/desactivació, període de refresc i deadband.
- OPCItems: té un funcionament similar a la col·lecció OPCGroups, en aquest cas s'utilitza per copiar la interfície dels ítems d'un grup.
- OPCItem: és l'objecte amb estructura d'ítem OPC.

7.2.3. Comunicació client-servidor

Per a realitzar la connexió del client OPC amb un servidor i fer ús de tots o part dels ítems que aquest conté cal seguir una determinada seqüència. Abans de connectar amb el servidor, primer s'han de declarar els objectes dins l'aplicació. A continuació ja es pot connectar amb el servidor i iniciar l'obtenció de la interfície de grups.

Depenent de si el grup ja està creat en el servidor o es tracta d'un grup nou, es seleccionarà el grup determinat o se n'afegirà un de nou. Seguidament s'obté la interfície d'ítems del grup per poder ser finalment afegits a l'aplicació i utilitzar-los correctament. A partir

d'aquest moment els ítems es refrescaran depenent del temps assignat durant la creació del grup al qual pertanyen.

El diagrama de flux següent mostra els passos a seguir de forma gràfica:

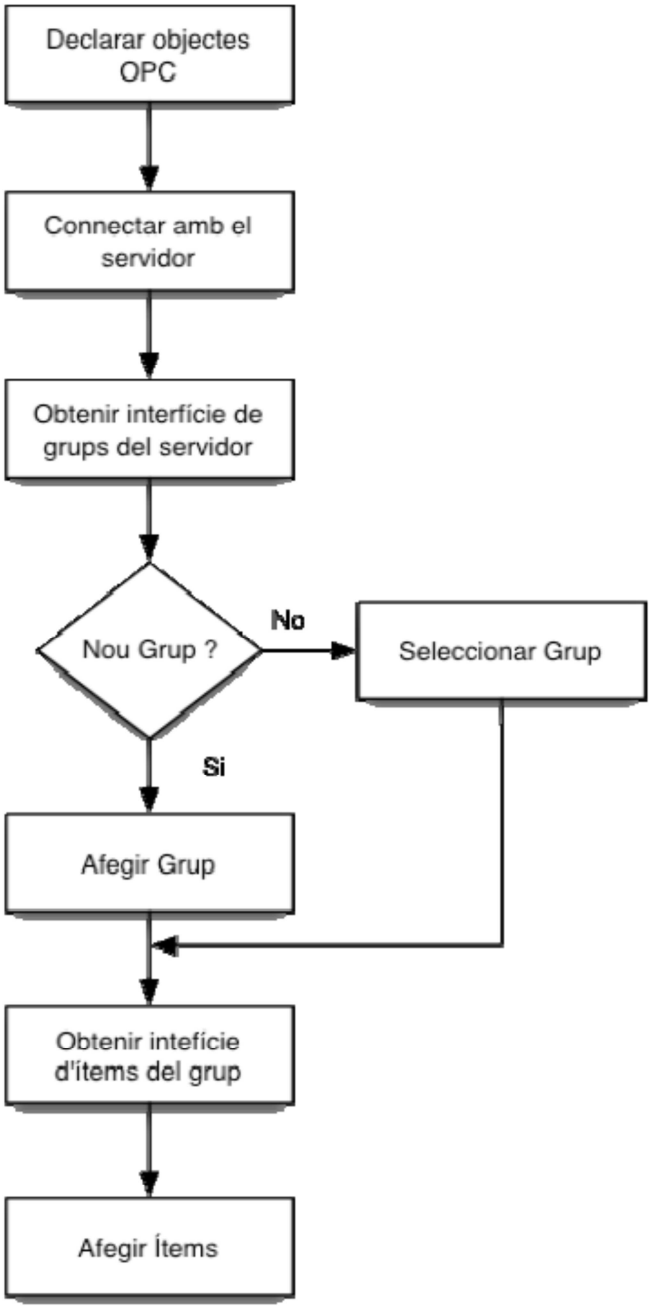


Figura 6. *Connexió amb servidor OPC.*

Per a que la connexió sigui exitosa, caldrà fer un últim pas, indicar i referenciar els ítems utilitzats entre client i servidor. De forma

independent, s'indexen tots els ítems utilitzats tant en el client (ClientHandles) com tots els ítems continguts dins el servidor (ServerHandles), de forma que no sigui necessari que l'ítem que utilitzi el client tingui el mateix valor d'índex que el adjudicat en el servidor.

Per tant, es fa necessària una forma d'assignació de correspondència. Aquesta assignació es fa quan un client accedeix per primera vegada a un servidor i incorpora els ítems. El client envia els ClientHandles corresponents als ítems als quals vol accedir i el servidor li respon amb els valors dels ServerHandles dels ítems corresponents.

Ara només cal que el si el client vol accedir al servidor li enviï el ServerHandle corresponent al ítem desitjat o, en cas que el servidor actualitzi un valor del client, el servidor enviï el ClientHandles.

Figura 7. *Relació entre ClientHandles i ServerHandles.*

Seguint la figura anterior, en la situació en que el client vol manipular un ítem del servidor, el client envia el ServerHandle=10 de l'ítem segons el servidor. En el cas contrari, on el servidor actualitza el valor en el client, el servidor envia el ClientHandle=1 per actuar sobre l'ítem correcte en el client.

Donat que el sistema ha de permetre connectar-se a diferents servidors, i les dades no estan definides a priori, de manera que s'han de crear diferents grups i afegir els ítems de manera dinàmica s'han creat una sèrie d'objectes que donen més flexibilitat al client OPC.

Les classes utilitzades són les següents:

- *OPCServerClass*: és l'objecte principal ja que permet la creació de l'objecte *OPCServer* i la col·lecció *OPCGroups* amb els quals

podrem fer la connexió amb el servidor OPC i fer la correcta gestió dels grups que afegim a aquest servidor respectivament.

Aquesta classe incorpora totes les funcions necessàries per a realitzar les funcions més comunes per a un servidor OPC, on prenen principal importància les següents funcions donada la seva operativitat:

- *ConnectOPCServer*: afegeix el servidor i prepara la interfície per afegir els grups. Necessita de tres paràmetres: el nom del servidor, la clau o nom amb que es referencia el servidor que s'està connectant i l'índex del servidor (versió numèrica de la clau). Quan un objecte OPCServerClass ha estat creat i connectat, els objectes OPCGroupClass i OPCItemClass ja poden crear les seves pròpies claus.
- *DisconnectOPCServer*: elimina la connexió del servidor seleccionat. No necessita cap argument.
- *GetOPCServerKey*: permet recuperar la clau o nom amb que s'ha fet referència al servidor seleccionat dins la col·lecció de servidors.
- *AddOPCGroup*: permet afegir un grup al servidor actual. Necessita de 5 paràmetres: el nom del grup, el temps de refresc, el *deadband* i el valor per activar o mantenir desactivat el grup (variable booleana). L'últim paràmetre serà retornat per aquesta funció indicant la clau o nom de referència del grup. Aquesta funció té la particularitat de crear una instància de l'objecte *OPCGroupClass*.
- *RemoveOPCGroup:* elimina un grup del servidor i la col·lecció.
- *GetOPCServerGroupCollection*: permet obtenir la referència a la col·lecció *OPCServerGroups,* la qual conté la llista de grups com a objectes del tipus *OPCGroupClass*.

La resta de funcions que incorpora aquesta classe es poden utilitzar, d'una banda, per obtenir informació del servidor com ara l'estat, el venedor i/o el creador del servidor, les versions o quan s'ha actualitzat per darrera vegada entre altres.

D'altra banda, les funcions afecten a configuració o propietats dels grups com ara establir o agafar els valors de DeadBand, UpdateRate o activar/desactivar els grups que pertanyen a aquest servidor.

- *OPCGroupClass*: permet la creació de l'objecte *OPCGroup* i la col·lecció *OPCItems* amb els quals podrem afegir i fer la correcta gestió dels ítems que afegim al grup seleccionat. La instància d'aquest objecte es realitza quan es crida a la funció *AddOPCGroup*. Destaquen d'aquesta classe les funcions següents:
  - *SetOPCGroup*: es crida quan afegim un grup amb la funció *AddOPCGroup* de la classe *OPCServerClass* i s'utilitza per iniciar l'objecte OPCGroupClass per al grup que s'estigui creant. Necessita quatre paràmetres: l'objecte de tipus *OPCGroup* sobre el que actuar, el nom del grup, la clau o nom de referència del grup i el valor numèric del nom de referència o clau.
  - *GetGroupKey*: retorna la referència del grup dins la col·lecció *OPCServerGroups.*
  - *AddOPCItem*: permet afegir ítems al grup seleccionat i crea i estableix la interfície per a manipular els ítems a través de l'objecte *OPCItemClass*. Necessita de quatre paràmetres: nom de l'ítem, tipus de dada de l'ítem, l'estat actiu/inactiu de l'ítem i la seva clau o nom de referència.
  - *RemoveOPCItem*: permet eliminar l'ítem seleccionat del grup al qual pertany. Únicament necessita com argument la clau o nom de referència de l'ítem que es vol eliminar. Retorna un valor booleà per indicar si s'ha pogut realitzar correctament la funció o no.
  - *ValidateOPCItem*: permet comprovar si l'ítem està disponible i/o la direcció es correcte així com el tipus de dada i el seu estat de forma que la connexió amb aquest ítem sigui satisfactòria.
  - *AsyncWriteOPCItem*: permet l'escriptura sobre un ítem. Necessita dos arguments: un el objecte tipus

OPCItemClass qua actua com objecte amb totes les propietats d'un ítem OPC i el valor que volem escriure.

- *GetOPCGroupItemsCollection*:
- *OPCGroupObj_DataChange:* atén a l'esdeveniment *DataChange* el qual s'activa quan un valor ha canviat i ha passat el període de refresc del grup. És aquí quan s'actualitzen els valors dels ítems.

La resta de funcions que incorpora aquesta classe es poden utilitzar per a modificar o agafar el nom del grup, la clau o nom de referència del grup, i el valor numèric del nom de referència o clau.

També conté, com passava en el cas anterior, les funcions per configurar o agafar els valors actuals del DeadBand i UpdateRate, establir o modificar el tipus de dada dels ítems, o activar/desactivar els ítems que pertanyen a aquest grup.

- *OPCItemClass*: aquest objecte permet la creació dels objectes tipus *OPCItem* i conté totes les funcionalitats utilitzades per a interactuar amb els ítems OPC. La instància d'aquest objecte es realitza quan es crida a la funció *AddOPCItem*. Destaquen d'aquesta classe les funcions següents:
  - *SetOPCItem*: es crida quan afegim un ítem amb la funció *AddOPCItem* de la classe *OPCGroupClass* i s'utilitza per iniciar l'objecte OPCGroupClass per al grup que s'estigui creant. Necessita quatre paràmetres: l'objecte de tipus *OPCItem* sobre el que actuar, l'identificador de l'ítem, el número d'ítem corresponent al total d'ítems afegits fins ara contant l'actual i el tipus de dada que és l'ítem.
  - *UpdateOPCItemData:* es crida des de la funció *OPCGroupObj_DataChange* i retorna el valor actual de l'ítem, la seva qualitat i el temps en el qual s'ha produït el canvi de valor. No necessita cap paràmetre sinó que retorna els tres valors esmenats.
  - *GetItemID:* retorna el nom amb el que s'ha afegit l'ítem.
  - *GetItemValue:* retorna el valor de l'ítem en el moment en que es llença la funció.

- *GetItemQuality:* retorna el valor de la qualitat de l'ítem en el moment en que es llença la funció.
- *GetItemServerHandle:* torna el valor del punter que el servidor utilitza per a referenciar l'ítem indicat, el qual no ha de ser forçosament el mateix que el valor del client.

La resta de funcions que incorpora aquesta classe es poden utilitzar per a recuperar el valor temporal del moment en que s'ha actualitzat un ítem, el tipus de dada que és un ítem o configurar o agafar el valor per a activar/desactivar l'ítem, entre altres funcions.

El codi necessari per establir la connexió entre client i servidor, crear un grup, afegir els ítems i fer la lectura dels seus valors es pot trobar desglossat en els següent punts, cada un amb la seva finestra de codi:

- Connexió amb el servidor

Per a connectar-nos amb el servidor es fa ús de la funció *AddSelectedOPCServerMain* a la qual cal passar-li el nom del servidor indicant el nom del servidor:

Codi 1. *Crear i afegir un servidor OPC.*

```
Sub AddSelectedOPCServerMain(OPCServerName As String)

      Dim OPCServer As OPCServerClass
      Set OPCServer = New OPCServerClass
      Dim Result As Boolean

      ' Crea una identificació única per al nou servidor
      Dim SrvName As String
      SrvName = "Server" + Str(Module1.ServerIndex)

      ' Prova la comunicació amb el nou servidor
      Result = OPCServer.ConnectOPCServer(OPCServerName, SrvName,
      Module1.ServerIndex)

      ' Prepara l'índex per al proper servidor que s'afegeixi
      Module1.ServerIndex = Module1.ServerIndex + 1

      ' Si la connexió és exitosa, afegeix el servidor a la classe
      ' de servidors OPC utilitzats
      If (Result = True) Then
            With OPCServers
                  .Add OPCServer, SrvName
            End With
            ' Afegeix l'objecte Node.
            Dim nodX As Node ' Declara la variable Node.
            ' Afegeix el servidor com a branca principal
            Set nodX = fPrincipal.tvTreeView.Nodes.Add(, ,
            SrvName, OPCServerName)
            nodX.EnsureVisible
            ' Selecciona el nou servidor creat
```

```
            Set Module1.SelectedOPCServer = OPCServer
            ' Neteja qualsevol grup seleccionat
            Set Module1.SelectedOPCGroup = Nothing
            Set Module1.SelectedOPCItem = Nothing
            ' Neteja la llista d'ítems
            lvListView.ListItems.Clear
      End If
End Sub
```

- Creació d'un grup

Per crear un grup, utilitzarem la funció *AddOPCGroupMain* la qual permet afegir el grup creat al servidor prèviament seleccionat indicant el nom del grup, el temps de refresc, el *deadband* i l'estat actiu o inactiu del grup:

Codi 2. *Crear i afegir un grup.*

```
Sub AddOPCGroupMain(ByVal GroupName As String, ByVal UpdateRate As
   Long, ByVal DeadBand As Single, ByVal ActiveState As Boolean)

      Dim GroupKey As String
      If Module1.SelectedOPCServer.AddOPCGroup(GroupName,
      UpdateRate, DeadBand, ActiveState, GroupKey) = True Then

            ' Afegeix l'objecte Node.
            Dim nodX As Node ' Declara la variable Node.
            ' Afegir el nou grup com una arrel en l'arbre
            Set nodX = fPrincipal.tvTreeView.Nodes.Add
             (Module1.SelectedOPCServer.GetOPCServerKey, tvwChild,
             GroupKey, GroupName)
            nodX.EnsureVisible
      End If
End Sub
```

- Afegir els ítems

Per afegir els ítems utilitzarem la funció *AddOPCItemMain* indicant:

Codi 3. *Afegir un ítem.*

```
Function AddOPCItemMain(ByVal ItemID As String, ByVal
      DataTypeSelected As Integer, ByVal ActiveState As Integer)

      Dim ItemKey As String
      ' Ens assegurem que el grup sigui vàlid
      If Not Module1.SelectedOPCGroup Is Nothing Then
         ' Intenta afegir un nou ítem
         If Module1.SelectedOPCGroup.AddOPCItem(ItemID,
              DataTypeSelected, ActiveState, ItemKey) = False Then
           AddOPCItemMain = False
           GoTo ErrorOnAdd
         End If
      End If

      Dim itmX As ListItem
      ' Afegeix el nou ítem a la llista
      Set itmX = lvListView.ListItems.Add(, ItemKey, ItemID)
      itmX.SubItems(1) = "" 'Iniciem ítem sense valor
      itmX.SubItems(2) = "Bad" ' Iniciem qualitat com a dolenta
```

```
      ' La funció ha funcionat
      AddOPCItemMain = True
ErrorOnAdd:
End Function
```

- Lectura de valors

Els ítems ja estan agregats al grup i aquest es va actualitzant amb el període de refresc indicat en el camp *UpdateRate*. No obstant, encara no s'obtenen els valors si aquest varien. Per a tal fi es fa ús de l'esdeveniment *Group_DataChange* que incorpora l'objecte *OPCGroup*, el qual s'activa quan un dels valors del grup canvia de valor:

Codi 4. *Lectura d'ítems mitjançant l'esdeveniment DataChange.*

```
Dim OPCGroupItemsCls As Collection
Dim OPCGroupToUpdate As OPCGroupClass
Dim OPCItemData As OPCItemClass
Dim OPCServerGroupsCls As Collection

' Agafar la col·lecció OPCItemClass del grup OPC seleccionat
Set OPCServerGroupsCls =
            Module1.SelectedOPCServer.GetOPCServerGroupCollection
Set OPCGroupToUpdate =
           OPCServerGroupsCls.Item(tvTreeView.SelectedItem.Key)
Set OPCGroupItemsCls = OPCGroupToUpdate.GetOPCGroupItemsCollection
' Si no hi han Ítems afegits saltar actualització
If OPCGroupItemsCls.Count = 0 Then
      GoTo SkipDisplayUpdate
End If
' Actualitzem només ítems visibles
Dim itmX As ListItem

' Agafem el primer ítem visible a la llista
Set itmX = lvListView.GetFirstVisible
If Not itmX Is Nothing Then
      Dim NumLinesDisplayed As Integer
      ' Calculem les línies que es veuran
      NumLinesDisplayed = (lvListView.Height / 214)

      Dim i As Integer
      Dim a As Integer
      Dim GroupItemIndex As Integer
      Dim OPCItemToUpdate As OPCItemClass
      Dim Quality As Long
      Dim ItemValue As Variant
      a = itmX.Index

      ' Canviar la llista d'ítems visibles si l'usuari mou la
      ' finestra
      If LastTopItem <> a Then
         If LastTopItem <> -1 Then
            Set lvListView.SelectedItem = itmX
         End If
      LastTopItem = a
      End If

      For i = 1 To NumLinesDisplayed
         'Agafem la clau que enllaça els ítems amb grup i servidor
         GroupItemIndex = Val(Mid(itmX.Key, InStr(itmX.Key, " ")))
```

```
            ' Un cop tenim la clau de l'ítem podem identifcar-lo en
            ' la classe OPCGroupItemCls
            Set OPCItemToUpdate =
                              OPCGroupItemsCls.Item(Str(GroupItemIndex))
            ' Mirem el valor de l'ítem en l'estat actual
            ItemValue = OPCItemToUpdate.GetItemValue(OPCItemLocal)
            itmX.SubItems(1) = ItemValue

            ' Agafem la qualitat en la comunicació amb l'ítem
            Quality = OPCItemToUpdate.GetItemQuality(OPCItemLocal)
            If Quality And &HC0 Then
               itmX.SubItems(2) = "Bona"
            Else
               itmX.SubItems(2) = "Dolenta"
            End If

            ' Repetir la iteració però amb el següent ítem visible de
            ' la llista
            a = a + 1
            Set itmX = lvListView.ListItems.Item(a)
        Next i
End If
```

7.2.4. Helper

El control *Helper* és un control ActiveX disponible per a Visual Basic una vagada s'ha instal·lat el programari WebWare SDK. Per a la seva utilització, i possibilitar així la comunicació amb la controladora S4, cal afegir el complement des de VB al projecte de l'aplicació.

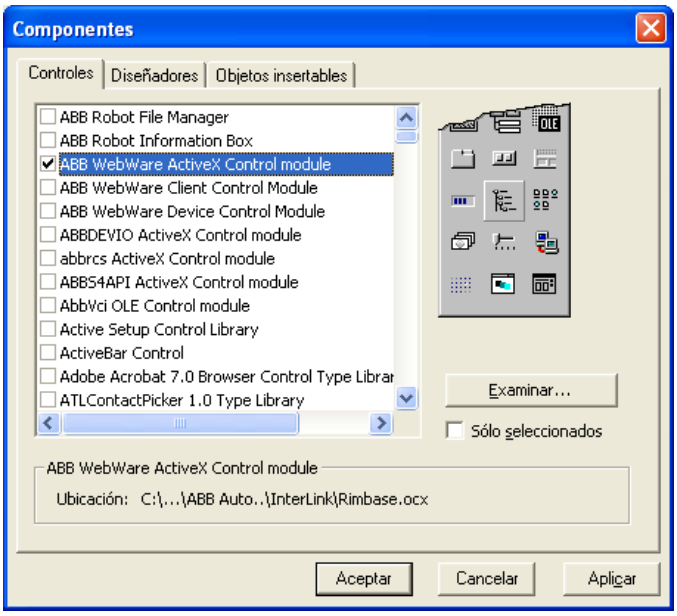


Figura 8. *Afegint el complement Helper.*

Un cop inserit, cal afegir-lo al formulari i modificar el nom del control per el nom alies introduït durant la configuració del InterLink Module. Per la configuració que disposa i que se li ha donat al laboratori durant

el seu procés d'instal·lació i posta a punt, l'alies del robot serà ASTERIX.

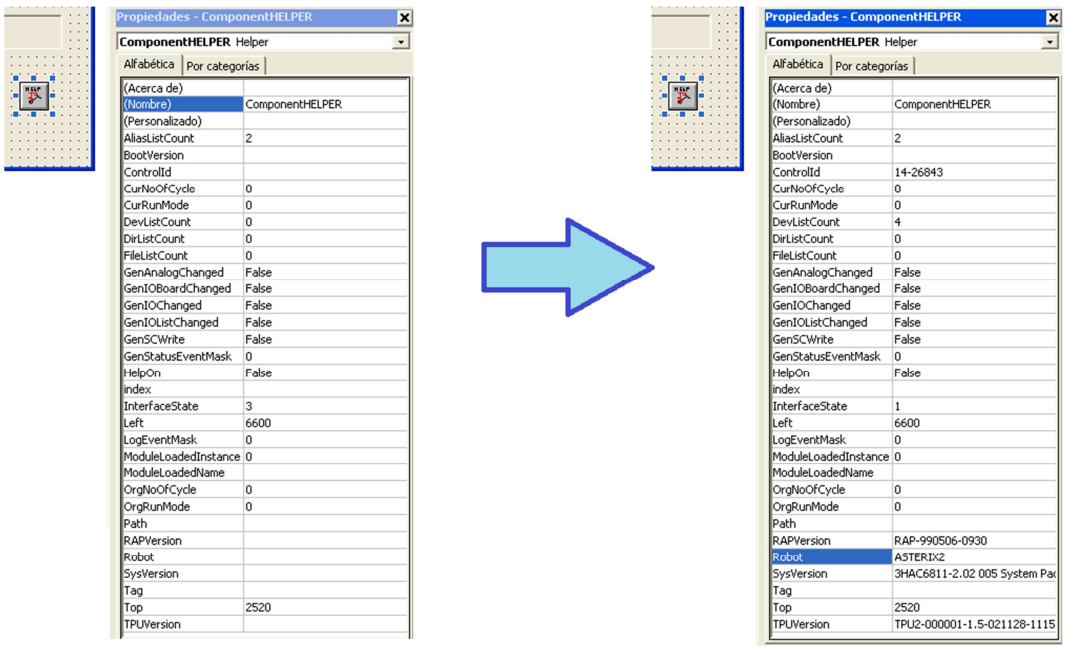


Figura 9. *Complement Helper configurat.*

Un cop introduït l'alies com a nom del controlador, aquest es configura tot sol i de forma automàtica amb tots els paràmetres de forma correcte per a la realització de la comunicació.

A partir d'aquí s'ha d'implementar el codi necessari per a cada una de les diferents possibilitats que ens ofereix el complement: escriptura i lectura de variables RAPID, recol·lecció d'esdeveniments, escriptura i lectura de les entrades digitals i analògiques, entre altres.

Per a més informació, es recomana al lector la lectura ABB Automation. 2003. *Helper Control and HelperScript Object* en "*WebWare SDK Controls Reference: User's Guide*". United States of America.

7.2.5. Base de dades - Aplicació

La base de dades i l'aplicació, com per motius de disseny s'ha implantat en el mateix PC, compartiran el servidor local que crea el programa *Microsoft SQL Server 2005* al instal·lar-se.

No obstant, aquest fet no afecta gaire a la seva utilització, doncs serà des de l'aplicació de monitorització de dades que es realitzarà la connexió amb la base de dades.

Com passava amb el client OPC, per utilitzar les funcions de base de dades caldrà inserir referències al projecte de VB6. En aquest cas, afegirem la referència *Microsoft ActiveX Data Objets 2.8 Library* (Figura 10. Agregant referència '*Microsoft ActiveX Data Objets 2.8 Library*'), la qual ens afegirà els objectes necessaris per connectar amb la BBDD, afegir noves dades, modificar les ja existents o eliminar-les entre altres.

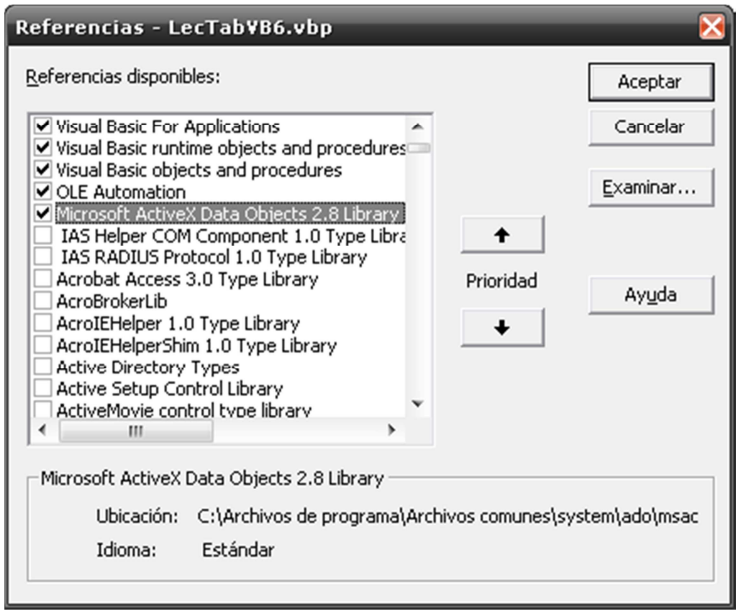


Figura 10. *Agregant referència 'Microsoft ActiveX Data Objets 2.8 Library'*

A mode de visualització, en el formulari pertinent a la visualització de les dades enregistrades a la taula on s'emmagatzemen, s'ha inserit també un component, *Microsoft Hierarchical FlexGrid Control 6.0 (OLEDB)*, que agrega una graella on es veuran les dades de la taula de forma ordenada.

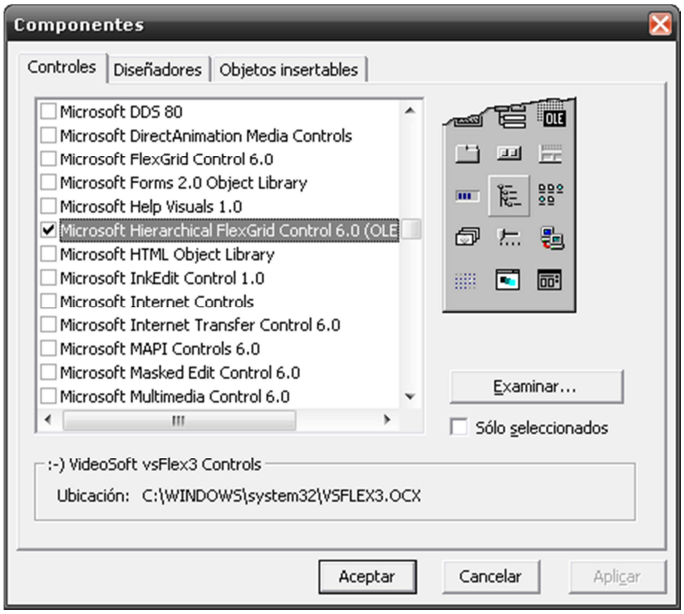


Figura 11. *Afegint el component Microsoft Hierarchical FlexGrid Control 6.0 (OLEDB)*

El codi necessari per establir la connexió entre l'aplicació i la Base de Dades i realitzar les operacions de visualització de les taules, enregistrament de les dades i modificació d'aquestes es pot trobar desglossat en els següent punts, cada un amb la seva finestra de codi:

- Emmagatzematge de les dades

Per desar les dades, primer necessitem crear l'objecte de connexió. A continuació escrivim la sentència que es desitja utilitzar dins d'una variable *String*, en aquest cas amb la sentència INSERT INTO (ja que volem escriure les dades dins de la BBDD en la primera fila lliure que trobem): INSERT INTO "nom_taula" (columna_1,columna_2,...) VALUES (valor_1,valor_2,...).

A la sentència INSERT INTO l'acompanya el nom de la taula, a continuació i entre parèntesis a les columnes que s'escriurà (no cal que siguin totes si la taula permet valors nuls) separades per comes, i mitjançant la paraula VALUES i també entre parèntesis els valors de cada una de les columnes, també separant els valors mitjançant comes, i amb el valor entre cometes a excepció de si la variable és de tipus text, cas en el que afegirem l'apòstrof (o cometes simple) a principi i final del valor.

Després cal obrir la connexió indicant el proveïdor, el nom de la base de dades, el servidor on es troba la base de dades, i finalment la seguretat i la forma de registrar-se.

Un cop oberta la connexió, creem l'objecte *recordset* per a continuació obrir-lo indicant la cadena on hi ha tota la instrucció (variable *String* creada anteriorment), l'enllaç de connexió utilitzada, com obrir la taula (estàtica o dinàmica depenent del procés a efectuar), i finalment el tipus de gestió a executar.

Codi 5. *Emmagatzematge de les dades a una taula d'una Base de Dades.*

```
' Creem una connexió per a la BBDD
Dim cn As ADODB.Connection
Set cn = New ADODB.Connection

' Creem i assignem la cadena de selecció que utilitzarem
Dim sSelect As String

' NomTaula ➔ Nom de la taula
' (NumLectura, ....) ➔ Nom de les columnes de la taula
' VALUES (" & NumLectura & ",..) ➔ Valor de les variables
sSelect = "INSERT INTO " & NomTaula & " (NumLectura, Dia_Hora,
      Peces_Escatat, Peces_Fresadora, Peces_Soldadura,
      Estat_Robot, Estat_Escatat, Estat_Fresat, Estat_Soldadura)
      VALUES (" & NumLectura & ",'" & Now & "',
      " & lblValorOPCItem(0) & "," &lblValorOPCItem(1) & ",
      " & lblValorOPCItem(2) & ",'" & lblValorOPCItem(3) & "',
      '" & lblValorOPCItem(4) & "','" & lblValorOPCItem(5) & "',
      '" & lblValorOPCItem(6) & "')"

' Prov= "registre necessari per indicar procés SQL"
' IC="Nom de la base de dades (on es troba la taula a la qual es
'     vol accedir)"
' DS="(servidor on es troba la BBDD [local --> server=propi PC])
'     \ instancia SQL [per defecte SQLEXPRESS]"
' Últim línia reservada a autoritzacions i logging, per defecte
' propi de Windows.
cn.Open "Provider=SQLNCLI; " & _
      "Initial Catalog=IRB140; " & _
      "Data Source=(local)\MARCSQL; " & _
      "integrated security=SSPI; persist security info=True;"

' Creem el recordset per accedir a les dades
Dim rs As ADODB.Recordset
Set rs = New ADODB.Recordset

' Obrim el recordset indicant cadena on hi ha tota la instrucció,
' l'enllaç de connexió, obrir com a dinàmica (ja que afegim dades)
rs.Open sSelect, cn, adOpenDynamic, adLockOptimistic

' Tanquem per alliberar recursos
cn.Close
Set cn = Nothing
```

- Lectura de la taula de la base de dades

El procés es força semblant a l'anterior. Primer creem l'objecte de connexió i seguidament la variable on desar la sentència a realitzar. En aquest cas farem ús de la instrucció SELECT*FROM, que permet recol·lectar tota la informació d'una taula d'una BBDD, instrucció a la qual només cal indicar la taula a la qual accedir.

A partir d'aquí, els passos son idèntics als utilitzats per emmagatzemar dades en una taula, caldrà obrir la connexió, crear l'objecte *recordset* i obrir-lo, per finalment tancar connexió i *recordset* per alliberar recursos.

Notis que en aquest cas, abans de tancar els recursos es desa el *recordset* a una graella anomenada *GraellaBBDD* i que és un objecte tipus MSHFlexGrid (Microsoft Hierachical Flex Grid), el qual s'omplirà de forma automàtica amb tota la taula indicada per a la lectura.

Codi 6. *Lectura de tota una taula d'una Base de Dades.*

```
' Creem una connexió per a la BBDD
Dim cn As ADODB.Connection
Set cn = New ADODB.Connection

' Creem i assignem la cadena de selecció que utilitzarem
Dim sSelect As String

'SELECT*FROM "taula a accedir de la BBDD indicada en el IC"
sSelect = "SELECT * FROM " & cboSelTaula.Text & ""

'Prov= "registre necessari per indicar proces SQL"
'IC="Nom de la base de dades"
'DS="(servidor on es troba la BBDD"
'últim línia autoritzacions i logging
cn.Open "Provider=SQLNCLI; " & _
      "Initial Catalog=IRB140; " & _
      "Data Source=(local)\MARCSQL; " & _
      "integrated security=SSPI; persist security info=True;"

' Creem el recordset per accedir a les dades
Dim rs As ADODB.Recordset
Set rs = New ADODB.Recordset

' Obrim el recordset indicant cadena on hi ha tota la instrucció i
' l'enllaç de connexió
rs.Open sSelect, cn, adOpenDynamic, adLockOptimistic

' Asignar el recordset al FlexGrid
Set GraellaBBDD.DataSource = rs

'Tanquem per alliberar recursos
rs.Close
cn.Close
Set rs = Nothing
Set cn = Nothing
```

- Escriptura de la taula de la base de dades

Els passos per a l'escriptura d'una dada en una posició específica de la taula són molt semblants als dos anteriors exemples. En aquest cas, la instrucció que indiquem en la sentència serà la instrucció UPDATE: UPDATE "nom_taula" SET "columna_a_canviar" = "nou_valor" WHERE "columna_especifica" = "valor determinat".

En aquesta instrucció cal indicar primer la taula on es realitzarà el canvi, a continuació i amb la paraula SET indiquem a quina columna volem realitzar el canvi i quin nou valor li volem posar, i amb WHERE indiquem la fila o files on realitzar la modificació si la nova columna especificada té el valor indicat.

Codi 7. *Escriptura d'un valor en una posició específica d'una taula d'una Base de Dades ja creada i amb dades.*

```
' Creem una connexió per a la BBDD
Dim cn As ADODB.Connection
Set cn = New ADODB.Connection

' Creem i assignem la cadena de selecció que utilitzarem
Dim sSelect As String

' UPDATE "nom_taula" ➔ Modificarem un valor d'aquesta la taula
' SET "columna" = "valor" ➔ Columna a modificar = Valor nou
' WHERE "columna principal" = "valor" ➔ Canviar dades on la
' columna principal = valor especificat
sSelect = "UPDATE " & cboSelTaula.Text & " SET " &
      Nom_columna_a_modificar & " = '" & Valor_nou & "' WHERE
      NumLectura = " & txtFila.Text & ""

'Prov= "registre necessari per indicar proces SQL"
'IC="Nom de la base de dades"
'DS="(servidor on es troba la BBDD"
'últim línia autoritzacions i logging
cn.Open "Provider=SQLNCLI; " & _
      "Initial Catalog=IRB140; " & _
      "Data Source=(local)\MARCSQL; " & _
      "integrated security=SSPI; persist security info=True;"

' Creem el recordset per accedir a les dades
Dim rs As ADODB.Recordset
Set rs = New ADODB.Recordset

' Executem sentència d'instrucció
cn.Execute sSelect

'Tanquem per alliberar recursos
cn.Close
Set cn = Nothing
```

En tots els casos, tant de creació de la taula com escriptura o lectura d'aquesta, es pot canviar la forma d'obrir la connexió si prèviament l'usuari ha creat una DSN (Data Source Name o nom d'origen de dades en català).

La DSN permet definir la base de dades que serà interrogada sense necessitat de passar per l'aplicació que hem utilitzat per construir-la, és a dir, amb simples trucades i ordres des d'un programa podrem obtenir les dades que busquem sense necessitat d'executar el gestor de la base de dades com Microsoft Access o el MySQL els quals, evidentment, no tindran per què trobar-se al servidor on treballem.

Amb la DSN creada, serà suficient indicar el nom de la DSN en el moment d'obrir la connexió de forma que no s'hagi d'indicar explícitament per codi els paràmetres del procés SQL, la base de dades, la taula a la qual s'està actuant i el *logging*, doncs la sentència DSN emmagatzema tota aquesta informació i que ha estat introduïda per l'usuari prèviament.

Els únics canvis que s'haurien de realitzar en el codi són els següents:

Codi 8. *Connexió amb una taula d'una BBDD mitjançant DSN.*

```
'Prov= "registre necessari per indicar proces SQL"
'IC="Nom de la base de dades"
'DS="(servidor on es troba la BBDD"
'últim línia autoritzacions i logging
cn.Open "Provider=SQLNCLI; " & _
      "Initial Catalog=IRB140; " & _
      "Data Source=(local)\MARCSQL; " & _
      "integrated security=SSPI; persist security info=True;"
```

```
'Connexió mitjançant DSN
cn.Open "DSN=Nom_Origen_Dades_MARC"
```

8. CONFIGURACIÓ DEL SERVIDOR OPC

Per a realitzar tota la configuració del servidor OPC del robot cal començar per la configuració pròpia de la controladora S4 del robot i s'utilitzarà el programa *Interlink Module Configuration Utility* (ICU) del qual ja se n'ha parlat en capítols anteriors.

El mòdul Interlink necessita la descripció de cada dispositiu que participi a la xarxa. Aquesta descripció o nom és diu Àlies, i per configurar com ha d'operar i funcionar aquest àlies es farà ús de l'eina ICU esmenada.

Així doncs, el primer pas és la configuració de l'àlies del robot. L'àlies és el nom amb el qual es fa referència a les dades d'una controladora de robot i que utilitzaran les aplicacions client que hi vulguin interactuar. Es necessari un nom per cada un dels diferents dispositius que tinguem a la xarxa.

La major part de la configuració de l'àlies està continguda en un arxiu anomenat perfil. Els perfils es creen i modifiquen a través de l'Editor de Perfil (*Profile Editor*) que és una part integral de la utilitat de configuració del mòdul Interlink.

Un cop ja disposem de l'àlies, caldrà fer la configuració del dispositiu de comunicació OPC que es desitgi introduir, en el nostre cas, el servidor OPC de la pròpia controladora.

8.1. Configuració de l'alies del robot

Només arrancar el programari, ens trobarem en la pàgina de configuració de l'alies del robot per defecte. Aquí ens trobem l'alies del qual disposem i en farem ús. En cas de ser el primer cop en iniciar-lo, la finestra restaria en blanc.

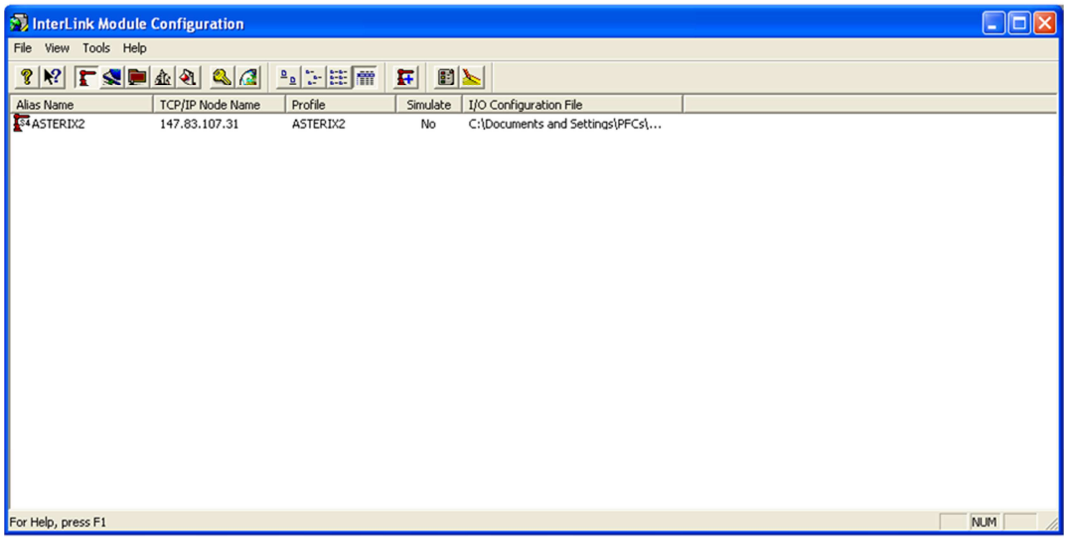


Figura 12. *Finestra principal Interlink Module Configuration.*

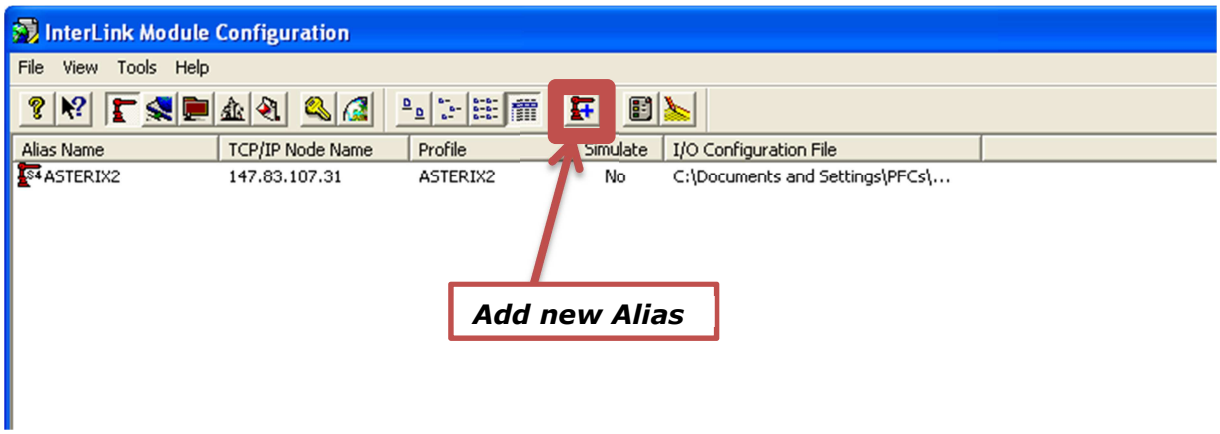


Figura 13. *Crear un nou Alias.*

Primer de tot crearíem un nou alies prement el botó *Add New Alias*: li donem un nom, introduïm el node TCP/IP que utilitzarà i li atorguem un perfil (Figura 14. Finestra Edit Alias.). El perfil conté els paràmetres de configuració de l'Interlink associats a un àlies.

Un cop introduïts aquests camps, ja tindrem l'alies creat per a la seva utilització. Un cop creat, sempre podrem canviar algun d'aquests paràmetres en cas que sigui necessari mitjançant l'opció *Edit Alias*.

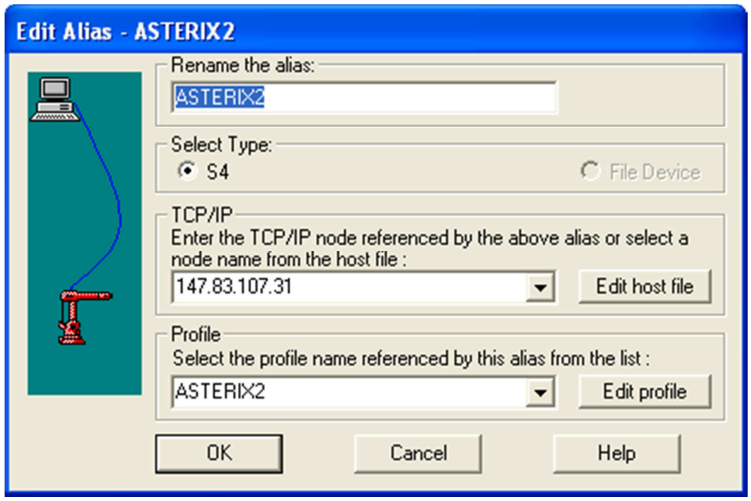


Figura 14. *Finestra Edit Alias.*

No obstant, per no afectar a altres usuaris es farà ús de l'alies del qual es disposa ja al laboratori. Per a més informació sobre la creació i edició de l'alies des de la ICU es recomana la lectura ABB Automation. 2003. *InterLink Module Configuration Utility* en "*WebWare SDK Controls Reference: User's Guide*". United States of America p. 4-12 – 4-19.

8.1.1. Subscripció de les variables

Abans de configurar el servidor, cal subscriure les variables dels programes realitzats en RAPID al àlies del dispositiu que es pretendrà connectar mitjançant comunicació OPC.

Per a fer possible la subscripció de les variables primer hem d'obrir la configuració del perfil: *Edit Profile*. Podem accedir-hi utilitzant el menú *Tools* i clicant a l'opció *Edit Profile* o bé mitjançant el botó d'accés directe a l'edició del perfil:

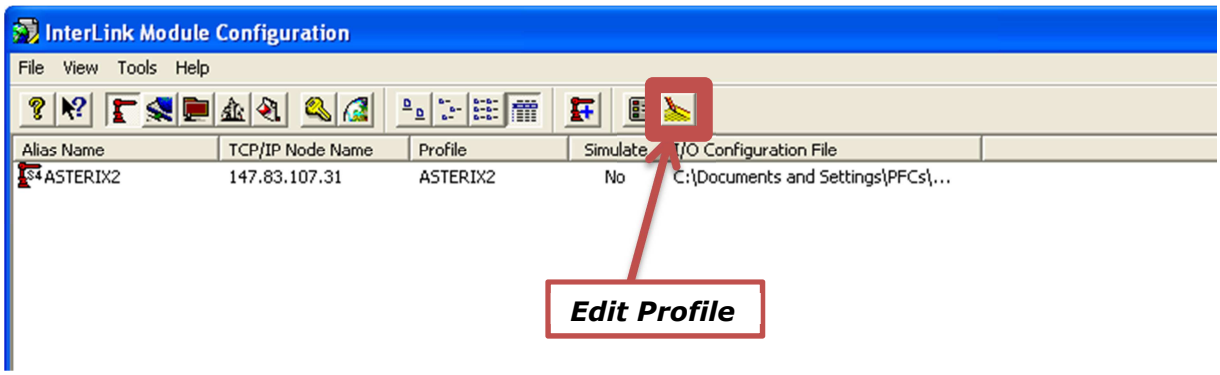


Figura 15. *Editar perfil.*

Un cop entrem a l'edició del perfil cal seleccionar el perfil que tenim associat al nostre alies. De la nova finestra emergent (Figura 16. Finestra de selecció de perfils.), veurem una llista on es mostren els diferents perfils disponibles.

Amb els botons situats a la dreta de la finestra, seguint l'ordre de dalt a baix, podrem seleccionar un perfil per a modificar-lo, tornar enrere, demanar ajuda al programa, afegir un nou perfil o eliminar-lo

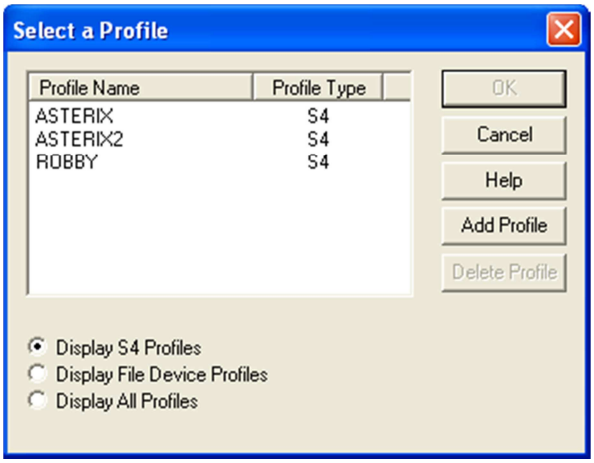


Figura 16. *Finestra de selecció de perfils.*

En el nostre cas seleccionarem el perfil anomenat ASTERIX2 el qual tenim associat al nostre dispositiu ASTERIX2. Tot i que en aquest cas perfil i dispositiu tenen el mateix nom, no és necessari que sigui així sempre, poden tenir noms diferents i el seu funcionament serà correcte igualment.

Un cop triat el perfil, es tancarà la finestra de selecció de perfil i s'obrirà la d'edició. En aquesta nova finestra trobarem 4 pestanyes diferents: *Interface*, *Variables Subscriptions, Backup File Set* i *FTP* amb les quals podrem respectivament configurar les propietats de la interfície (com per exemple activar la sincronització interna del robot o el temps d'enquesta), subscriure dades al àlies, seleccionar els ítems que volem fer-los un còpia de restauració i activar l'intercanvi de dades mitjançant FTP.

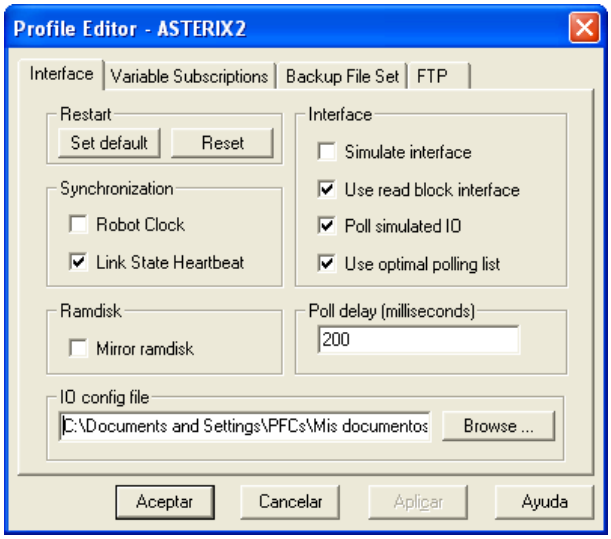


Figura 17. *Edició del perfil.*

Per al cas que ens ocupa, accedim directament a la pestanya de subscripció de variables: *Variables Subscriptions* (Figura 18. Edició del perfil: Subscripció de variables.).

Les variables que es subscriguin hauran de ser utilitzades en algun dels diferents programes RAPID introduïts a la controladora i hauran d'estar declarades com a variables persistents. Una variable persistent és aquella capaç de mantenir el valor de la variable encara que s'aturi l'aplicació i/o s'iniciï novament.

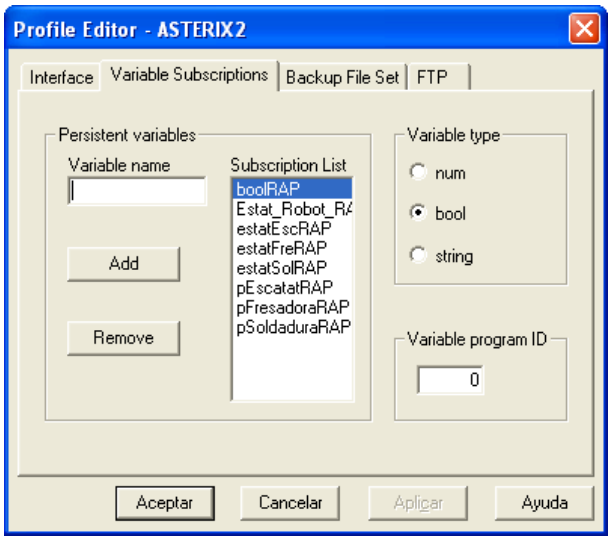


Figura 18. *Edició del perfil: Subscripció de variables.*

En el camp central, *Subscription List*, es mostra la llista de les variables que hi ha subscrites actualment al perfil. Per afegir noves

variables cal introduir el nom de la variable que es desitja subscriure en el camp *Variable Name*, i aquest ha d'ésser idèntic al nom de la variable dins del programa RAPID.

Un cop introduït el nom, cal indicar el tipus de variable que és entre les tres possibles opcions següents: *num,* variable del tipus numèric, *bool*, variable del tipus booleana, o *string*, variable de tipus text. Aquests tres tipus de variables són les úniques que el llenguatge RAPID permet identificar com variables persistents i ha de coincidir amb el de l'origen.

En el camp *Variable program ID* s'indica l'identificador del programa on està la variable. Per defecte i si no s'indica una altra cosa, aquest valor haurà de romandre sempre a 0, valor que permet accedir a la variable si està en qualsevol programa RAPID dels insertats a la controladora.

Un cop omplerts tots els camps, cal clicar al botó *Add* per acceptar la subscripció de la variable, i la qual passarà a mostrar-se i formar part de la llista de variables subscrites *Subscription List*. Per treure una variable, només caldrà seleccionar d'aquesta llista la variable que es desitja eliminar de la subscripció i prémer el botó *Remove*.

8.2. Configuració del servidor OPC

El procés es bastant semblant que per a la creació i configuració de l'alias. En aquest cas, caldrà anar primer a la pàgina *OPC Device Page* (Figura 19. Finestra *OPC Device Page*.). El primer pas serà afegir un servidor, o més ben dit, donar-li un nom al dispositiu OPC que es desitgi utilitzar, en aquest cas el servidor OPC de la controladora S4.

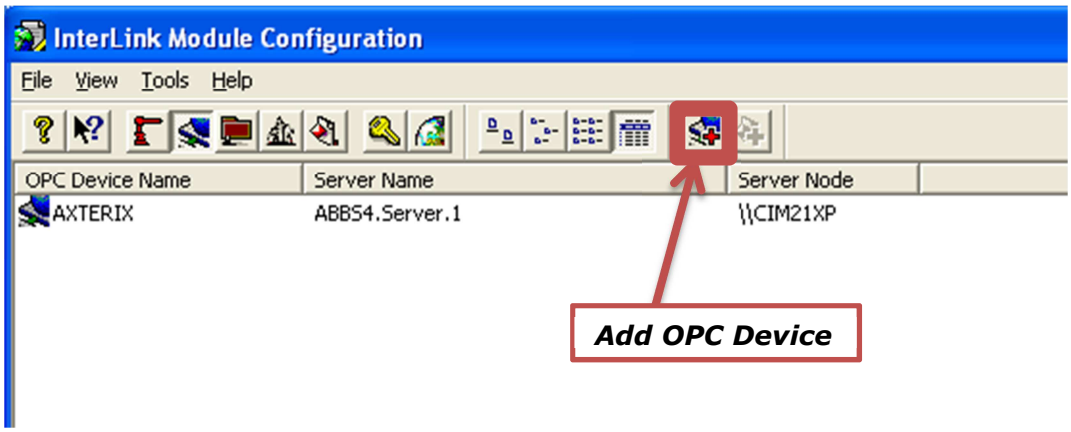


Figura 19. *Finestra OPC Device Page.*

Un cop premem el botó per afegir un nou dispositiu, se'ns obrirà una finestra des d'on podrem triar el dispositiu, ara el servidor ABBS4.Server.1, fent ús d'un desplegable.

Notis que els servidors estan dividits entre accés a dades, *Data Access*, representats amb un símbol blau connectat a una xarxa i en alarmes i esdeveniments, *Alarm and Events*, representats mitjançant una campana vermella.

Els primers els utilitzarem si desitgem fer la comunicació amb dades que s'hagin de mostrejar en temps real i de forma continua des de dispositius d'adquisició de dades. Per especificacions de la OPC Foundation s'haurà de comunicar el valor de la variable, la qualitat de la comunicació i el temps en que s'ha pres la mostra.

Les dades històriques no entren dins d'aquest grup, sino que tenen el seu propi tipus de servidor: *OPC Historical Data Acces*.

Els segons s'utilitzen per a comunicar dades de tipus alarma o esdeveniments puntuals i per tant, la comunicació no ha de ser contínua sinó esporàdica.

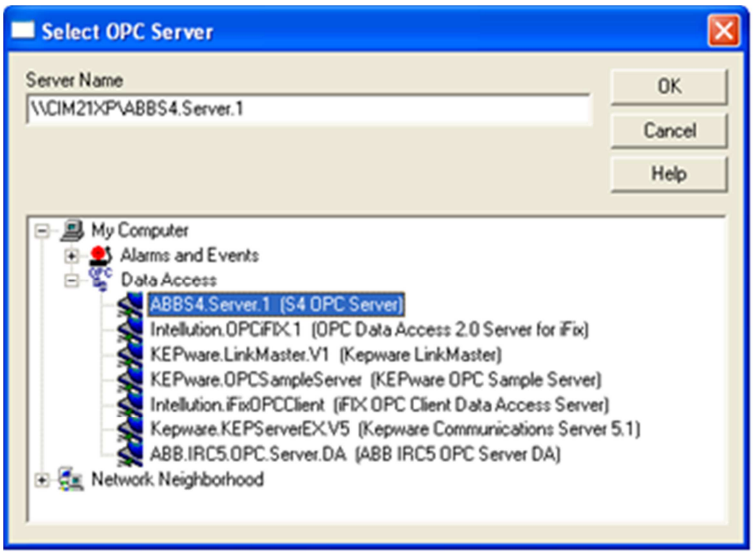


Figura 20. *Selecció del servidor OPC tipus Data Access.*

Al acceptar el tipus de servidor a afegir, per a nosaltres de tipus accés a dades, passarem a una nova finestra amb tres pestanyes: *Server*, pestanya inicial per defecte i on s'especifica la informació del servidor, *Tags*, des d'on es fa el control de les variables que utilitzarem, i *Main*

Page Layout, la qual conté la informació que apareixerà en la interfície d'usuari del servidor WebWare.

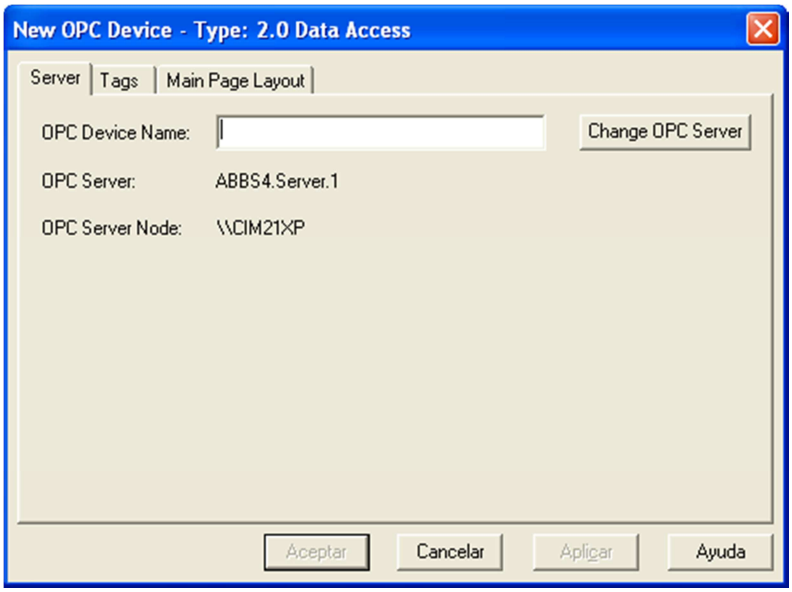


Figura 21. *Configurant el nou dispositiu OPC: pestanya Server.*

En aquesta primera pestanya, podrem indicar el nom que li volem donar al nostre servidor, nom al qual farem referència des de les aplicacions client, en el camp *OPC Device Name*. En els camps *OPC Server* i *OPC Server Node* es mostra la informació de a quin servidor s'està utilitzant amb aquest sobrenom actualment i el node del servidor respectivament.

També disposem d'un botó en aquesta pestanya, *Change OPC Server*, per si es desitja canviar de servidor. En aquest cas, se'ns obrirà novament la finestra mostrada en la Figura 20. Selecció del servidor OPC tipus Data Access.

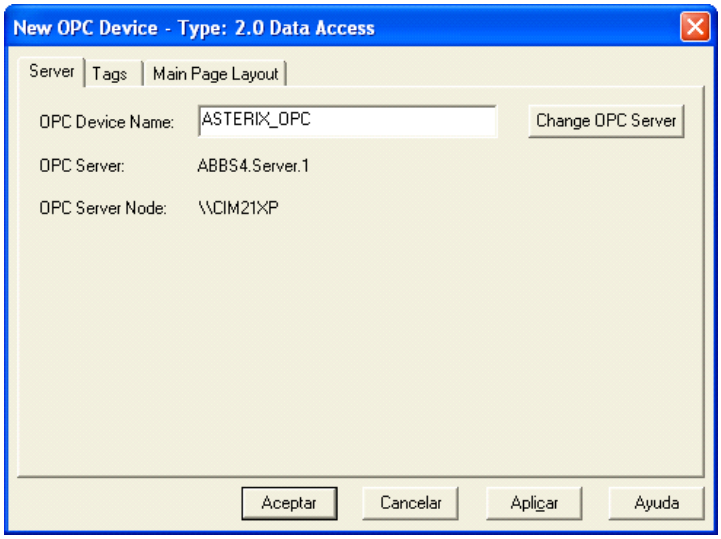


Figura 22. *Configurant el nou dispositiu OPC: pestanya Server (2).*

A la pestanya *Tags* (Figura 23. Configurant el nou dispositiu OPC: pestanya Tags.) trobem un llista amb les variables que actualment estan afegides al dispositiu i a les quals podrem accedir des d'un client.

Al costat de la llista trobem dos botons que ens permetran afegir noves variables o eliminar-ne aquelles que ja no siguin necessàries, *Add Tag* i *Remove Tag* respectivament.

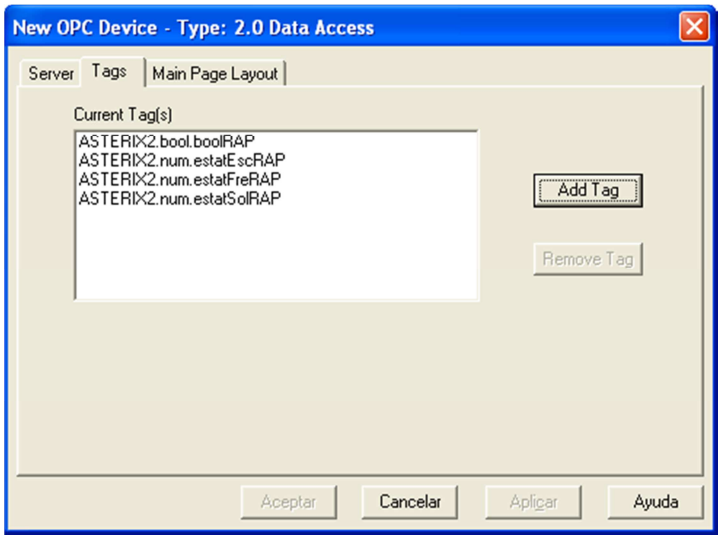


Figura 23. *Configurant el nou dispositiu OPC: pestanya Tags.*

Si seleccionem una variable que prèviament s'hagi afegit a la llista, en la part inferior de la pestanya trobem el camp *Selected Tag Data Type*

el qual ens permet modificar el tipus de dada de la variable en *Bool*, *Short*, *Long*, *Double*, *String* o bé mantenir el tipus per defecte.

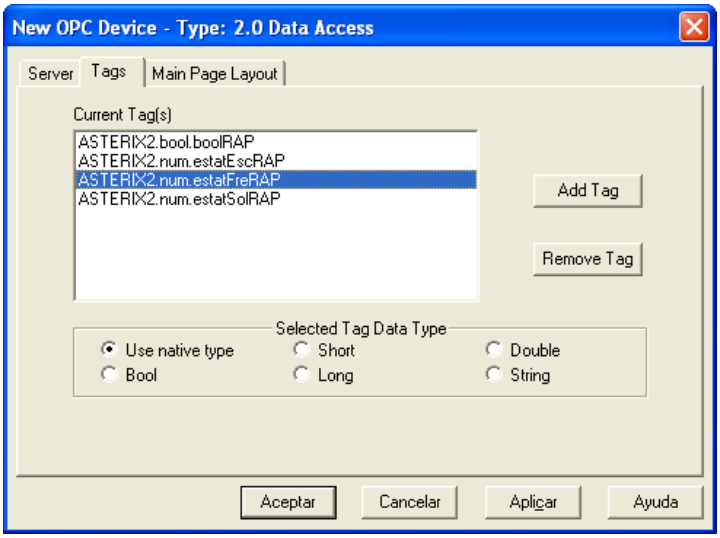


Figura 24. *Configurant el nou dispositiu OPC: pestanya Tags.*

En cas de voler afegir una nova variable, un cop premem el botó corresponent, se'ns obrirà una nova finestra (Figura 25. Afegint una nova variable.) des d'on podrem afegir les variables de dues maneres possibles. La primera opció es fer ús del camp *Item Name* on podem introduir el nom complet del ítem que es vol afegir si es coneix.

La segona, és utilitzar el panell *Browse Items* que ens permet buscar la variable desitjada dins dels diferents dispositius detectats. Al panell situat a la dreta del primer esmenat, trobarem la llista de les variables que conté el dispositiu seleccionat.

Si s'introdueix manualment la variable, només caldrà repetir el pas 4 tantes vegades com variables es desitgin afegir, mentre que si es fa ús del navegador els passos a seguir són:

1. Seleccionar el dispositiu del qual volem triar la variable des del panell de l'esquerra de la imatge,
2. Seleccionar la variable del panell dret que s'haurà omplert un cop realitzat el primer punt,
3. Seleccionar el tipus que de dada que volem que sigui la variable a afegir en el camp *Data Type*. Si es desconeix el tipus de dada o no ens interessa canviar el tipus de dada la podem deixar amb

el tipus nadiu de la variable activant l'opció *Use native type* i que ve marcada per defecte,

4. Afegim la variable mitjançant el botó *Add Item*. Si volem afegir més variables només cal repetir els passos 1 a 3.
5. Si ja tenim agregades totes les variables desitjades, per acceptar i tornar a la pantalla anterior caldrà prémer el botó *Done*.

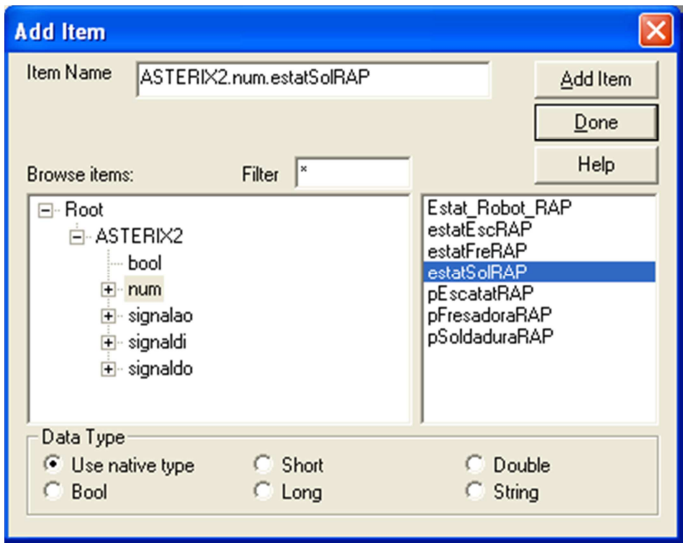


Figura 25. *Afegint una nova variable.*

En la última de les pestanyes del *OPC Device*, *Main Page Layout* (Figura 26. Configurant el nou dispositiu OPC: pestanya Main Page Layout.), podem especificar la informació que apareixerà en la interfície d'usuari del servidor WebWare i que podrà ser accessible a través del client OPC.

Totes les dades que es configurin seran utilitzades únicament com a informació del dispositiu.

En aquesta pestanya els camps estan dividits en dos blocs. En el primer, *General*, podrem introduir la informació de l'autor, la versió, la companyia i una petita descripció del servidor OPC que s'estigui configurant.

En el segon bloc, *User Defined*, trobem quatre quadres d'edició on podrem introduir el títol de la variable i l'estat de la variable. Si s'activa la casella *Tag*, ens permetrà associar el camp seleccionat a una variable ja existent.

En ambdós casos, la casella d'activació permet habilitar/inhabilitar que la informació estigui disponible o no per al client OPC de tipus Data Acces, de forma que aquesta informació s'actualitzi automàticament.

Aquesta informació està pensada principalment per ser mostrada a la interfície WebWare Server com a pàgina d'informació principal del dispositiu al qual s'estigui connectat, tot i que també pot ser utilitzada per mostrar-la en altres aplicacions OPC.

Les caselles d'activació *Tag* permeten associar cada un dels diferents camps d'informació a una variable específica. Si s'activa la casella se'ns obrirà una nova finestra semblant a la de la figura 25 per a seleccionar la variable a la qual referenciar aquesta informació.

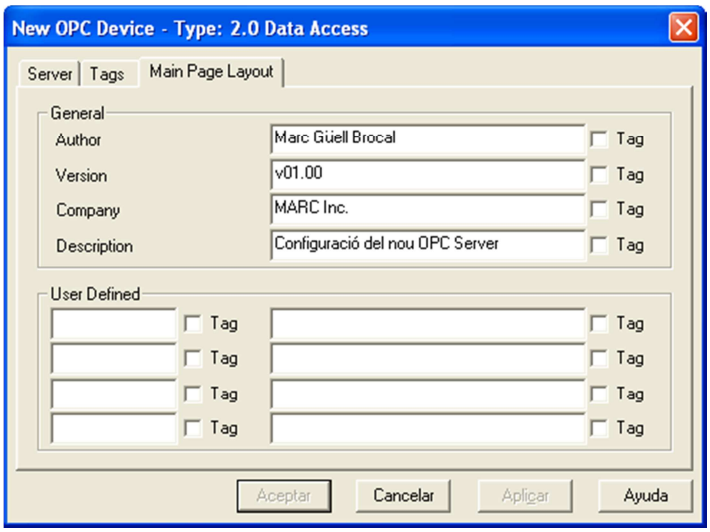


Figura 26. *Configurant el nou dispositiu OPC: pestanya Main Page Layout.*

En cas de voler afegir i configurar un tipus de servidor d'alarmes i esdeveniments (Figura 27. Selecció del servidor OPC tipus Alarms and Events.) la finestra compta amb algunes pestanyes més que per al cas del servidor de dades, al mateix temps que alguns camps de les pestanyes compartides seran diferents.

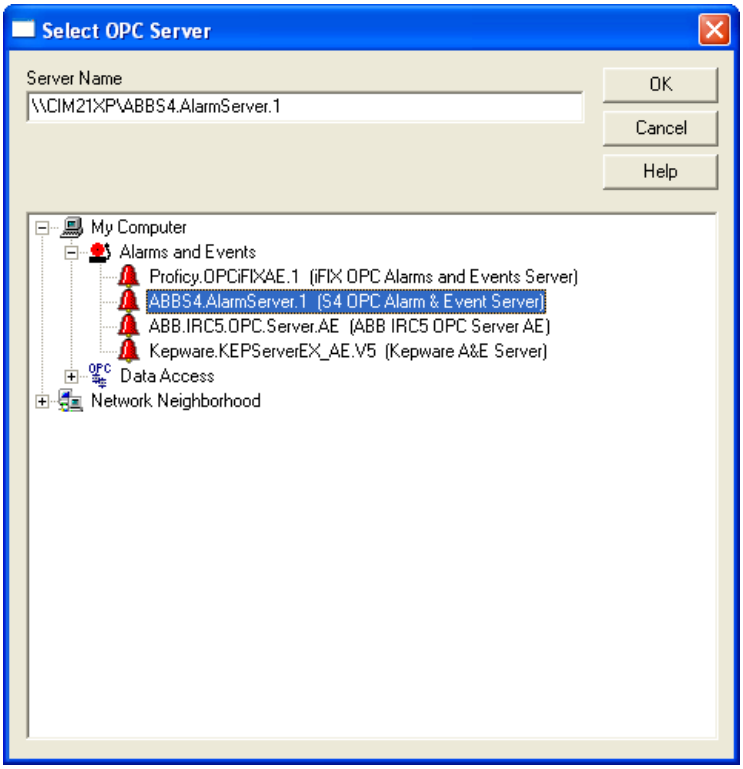


Figura 27. *Selecció del servidor OPC tipus Alarms and Events.*

Si ens movem per les diferents pestanyes podrem configurar els paràmetres del servidor (*Server*) els tipus d'esdeveniment i el seu rang de severitat (*Type and severity*), les categories que es mostraran (*Categories*), les àrees (*Areas*) i fonts a les quals es tindrà accés (*Sources*), i la pestanya *Main Page Layout* idèntica a la disponible per a la configuració d'un servidor de dades però sense permetre l'associació amb una variable.

Així doncs, en la primera de les pestanyes (*Server*) podem fer la configuració del servidor d'alarmes i esdeveniments que es desitgi afegir.

Molt semblant que per al cas d'accés a dades, els primers camps que trobem són el nom del servidor que li vulguem donar (*OPC Device Name*), el servidor OPC utilitzat, en aquest cas serà el *ABBS4.AlarmServer.1* per ser el servidor d'alarmes i esdeveniments de la controladora S4, i el node del servidor OPC.

Per ser un servidor també d'esdeveniments tenim la possibilitat d'habilitar la subscripció a als esdeveniments que es generin especificant el temps d'actualització per comprovar la generació

d'esdeveniments (camp *Update Rate*) i el número màxim d'esdeveniments permesos (*Max. # of Events*)

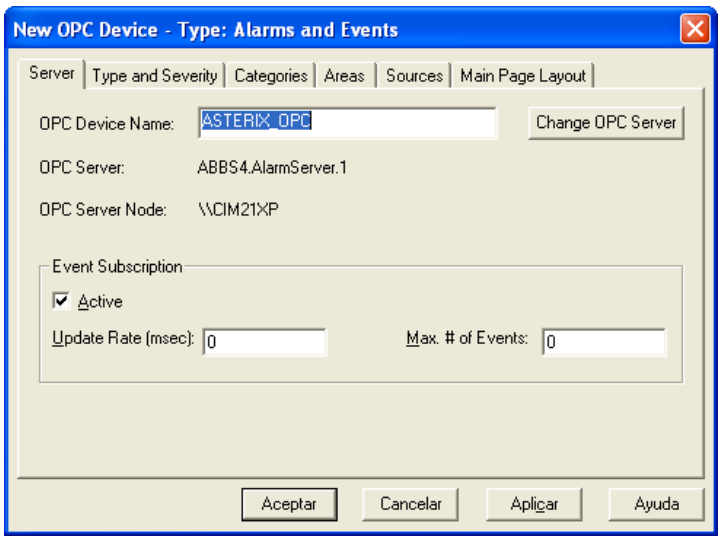


Figura 28. *Configuració dispositiu OPC d'alarmes i esdeveniments: pestanya Server.*

La pestanya *Type and Severity* permet especificar el tipus d'esdeveniments que es mostraran mitjançant les caselles d'activació del camp *Event Type* i els límits inferior i superior per generar l'esdeveniment mitjançant el camps de text *Low* i *High* respectivament continguts dins el camp *Severity Range*.

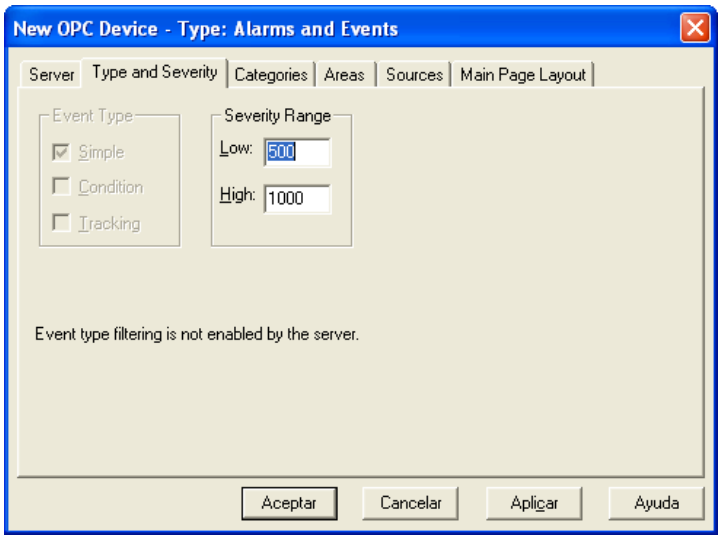


Figura 29. *Configuració dispositiu OPC d'alarmes i esdeveniments: pestanya Type and Severity.*

En la següent pestanya, *Categories*, podem especificar les categories de les alarmes i esdeveniments que es mostraran al servidor. En una

primera llista trobem totes les categories disponibles, llista *Available*, i en una segona les categories seleccionades per a ser utilitzades, llista *Select*.

Per afegir les categories de la llista de disponibles a la llista de seleccionades cal seleccionar la categoria en la primera de les llistes i afegir-la mitjançant el botó situat al centre de la finestra *Add*. Per treure una categoria de la llista de seleccionades caldrà primer seleccionar-la i després prémer el botó *Remove*.

Si es vol inhabilitar el filtre de categories, és a dir, tenir totes les categories disponibles independentment de quines han estat seleccionades i quines no mitjançant la casella *Disable Category Filter*.

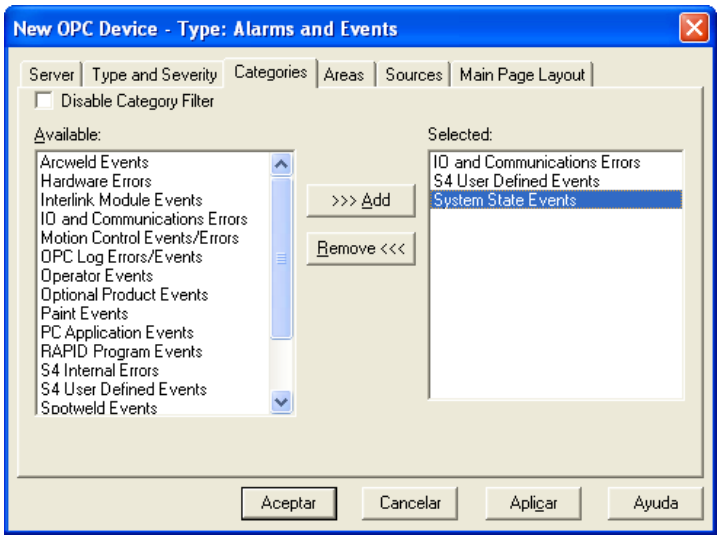


Figura 30. *Configuració dispositiu OPC d'alarmes i esdeveniments: pestanya Categories.*

En la pestanya *Areas* podem seleccionar les àrees per al dispositiu. Aquesta pestanya és molt semblant a l'anterior, on disposem de dues llistes, una amb totes les àrees disponibles i una altra amb les seleccionades amb els botons pertinents per afegir i treure àrees de les seleccionades per al dispositiu OPC.

A més, també es disposa del camp *Filter*, utilitzat per filtrar les àrees a mostrar en la llista de disponibles. Si es volen veure totes, no fer un filtre d'àrees, cal introduir un asterisc, el qual ve posat per defecte.

També conté el botó *Add Children*, botó amb el qual no només afegirem l'àrea seleccionada sinó també totes les àrees fill, àrees dependents de la àrea seleccionada.

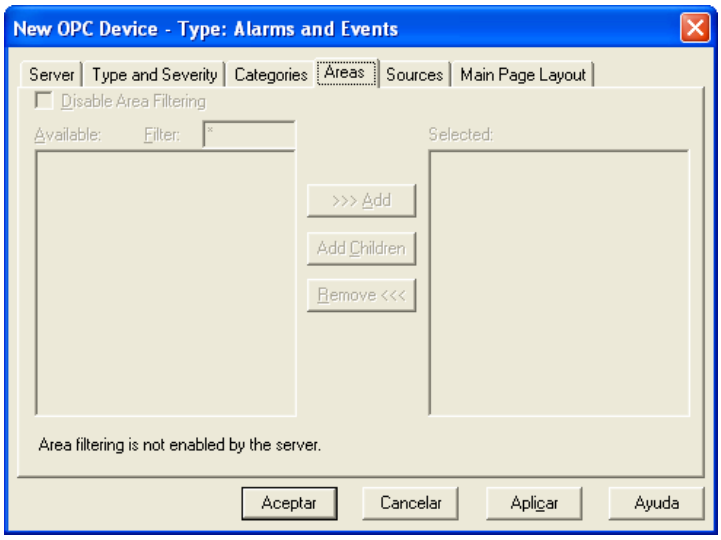


Figura 31. *Configuració dispositiu OPC d'alarmes i esdeveniments: pestanya Areas.*

La següent pestanya, *Sources*, permet seleccionar les fonts que s'incorporaran al dispositiu OPC que s'està configurant. Tant el seu aspecte com la forma d'ús son idèntics a la pestanya anterior d'*Areas*, però en comptes d'àrees en aquest cas es manipulen fonts.

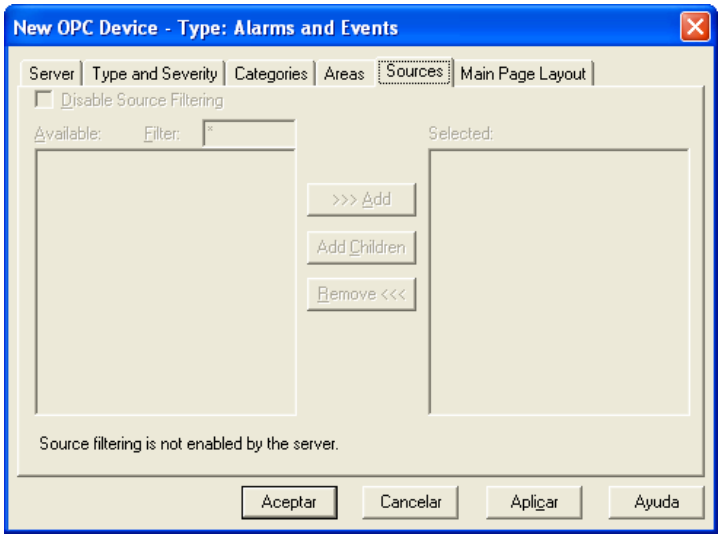


Figura 32. *Configuració dispositiu OPC d'alarmes i esdeveniments: pestanya Sources.*

L'última de les pestanyes, *Main Page Layout*, és molt semblant a la pestanya de configuració amb mateix nom disponible en el servidor *Data Acces*.

Aquí s'especifica la informació que apareixerà en la interfície d'usuari del servidor WebWare i que podrà ser accessible a través del client OPC sobre l'autor, versió, companyia i descripció del servidor OPC, a més de la possibilitat d'afegir quatre camps definits per l'usuari.

Totes les dades que es configurin seran utilitzades únicament com a informació del dispositiu. En aquest cas però, no es pot associar la informació a una variable.

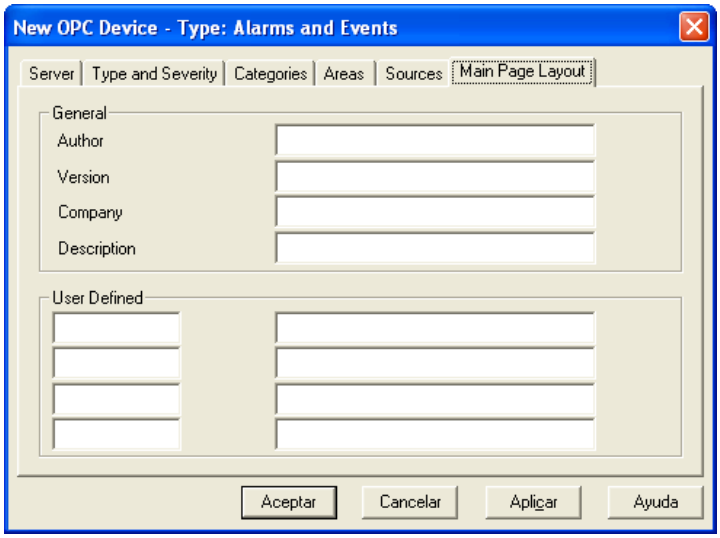


Figura 33. *Configuració dispositiu OPC d'alarmes i esdeveniments: pestanya Main Page Layout.*

9. BASE DE DADES

Com ja s'ha comentat, el projecte incorpora una base de dades on es pretén emmagatzemar la informació i dades de les variables monitoritzades. Aquesta base de dades (BBDD) està creada mitjançant el programari de Microsoft SQL Server 2005, sistema gestor de base de dades basat en models de taules relacionals.

Per al projecte només s'ha necessitat crear una única taula anomenada *IRB140*. Aquesta conté tota la informació pertanyent al robot i, des de la aplicació de monitorització, es fa la gestió de les dades, omplint-la a mesura que es fa la lectura de les variables.

Per a crear la BBDD iniciem el programari *Microsoft SQL Server Management Studio Express* i se'ns obrirà de forma automàtica una finestra per a connectar-nos al servidor de la xarxa. Agafem el que disposem per defecte i premem el botó *Conectar*.

Figura 34. *Connexió al servidor on trobarem la BBDD.*

Un cop connectats, estarem en la finestra principal del programa. Des d'aquí, pressionem amb a *Base de datos* amb el botó esquerra del ratolí i indiquem que volem un nova base de dades, *Nueva base de*

datos. En la nova finestra emergent, indicarem el nom de la BBDD, deixant la resta de paràmetres per defecte.

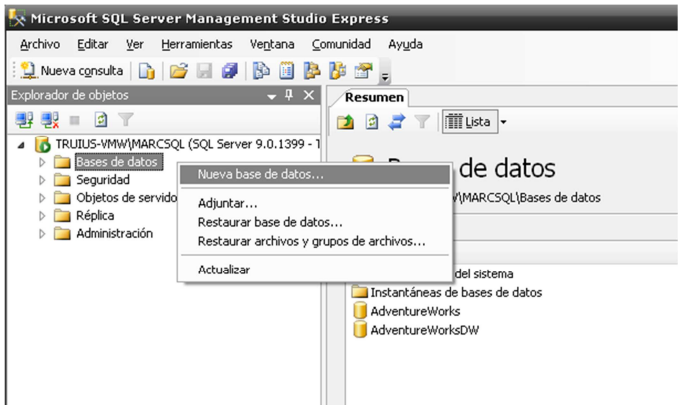


Figura 35. *Creant una base de dades.*

Observem així que la nova base de dades ja està disponible (Figura 36. Nova BBDD creada.). El pas següent serà crear una nova taula dins de la BBDD creada anteriorment. Com abans, amb el botó dret del ratolí accedim a les diferents opcions dins de les taules (*Tablas*), en aquest cas *Nueva tabla* (Figura 37. Creant nova taula.)

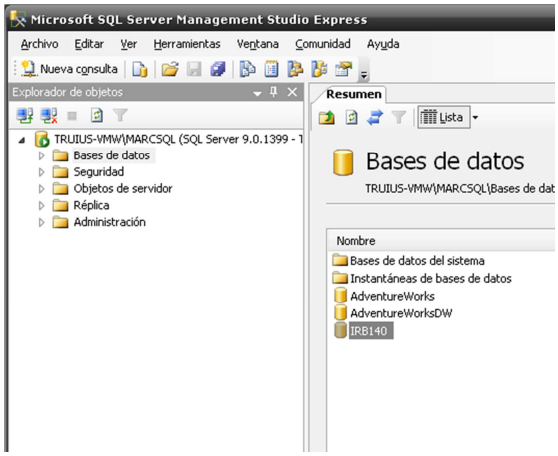


Figura 36. *Nova BBDD creada.*

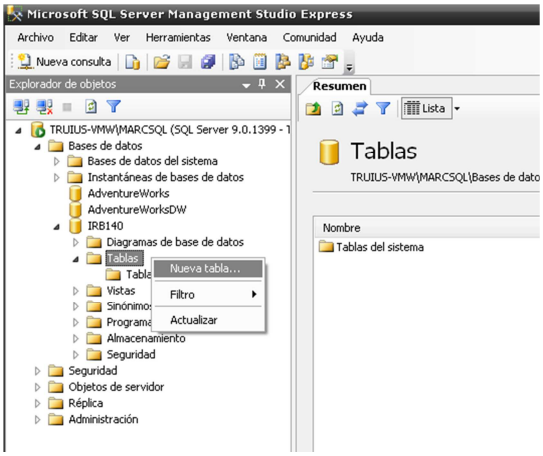


Figura 37. *Creant nova taula.*

Amb la nova taula, el primer pas serà donar-li un nom que s'ajusti a la seva funció des de la pestanya de la finestra principal *Propiedades*. A continuació afegirem tantes columnes com variables desitgem enregistrar col·locant-nos en la primera posició de cada fila, lloc on indiquem el nom, i en la segona posició el tipus de dada.

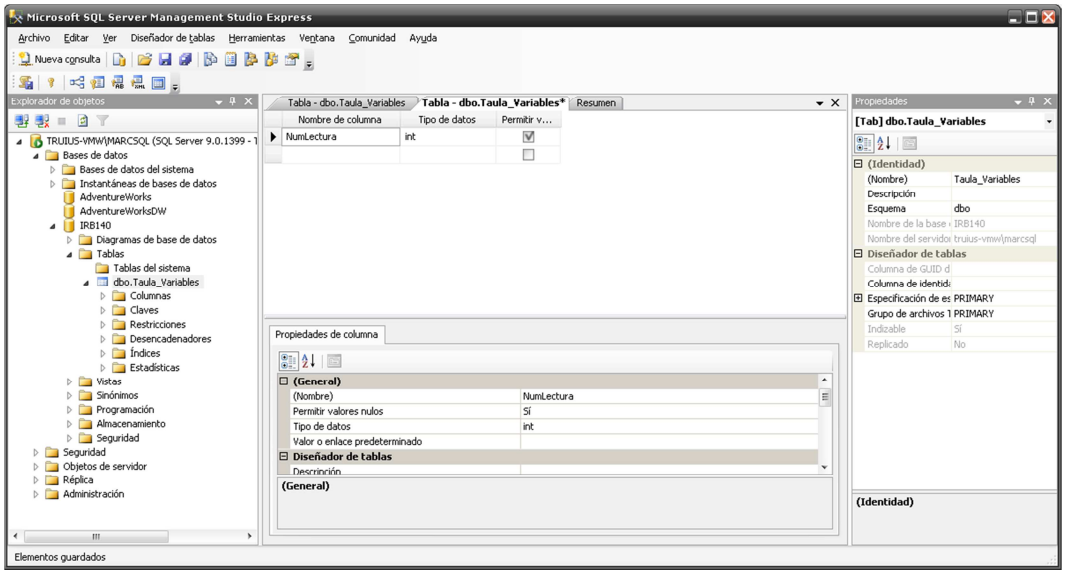


Figura 38. *Escriptura i configuració de les columnes de la taula de la BBDD.*

Si, per exemple, es desitgessin separar cada una dels diferents tipus principals de variables tractades per la controladora del robot: digitals d'entrada, digitals de sortida i variables persistents; faríem el procés de creació de taula 3 cops, afegint per a cada una tantes columnes com número d'elements tingui cada tipus.

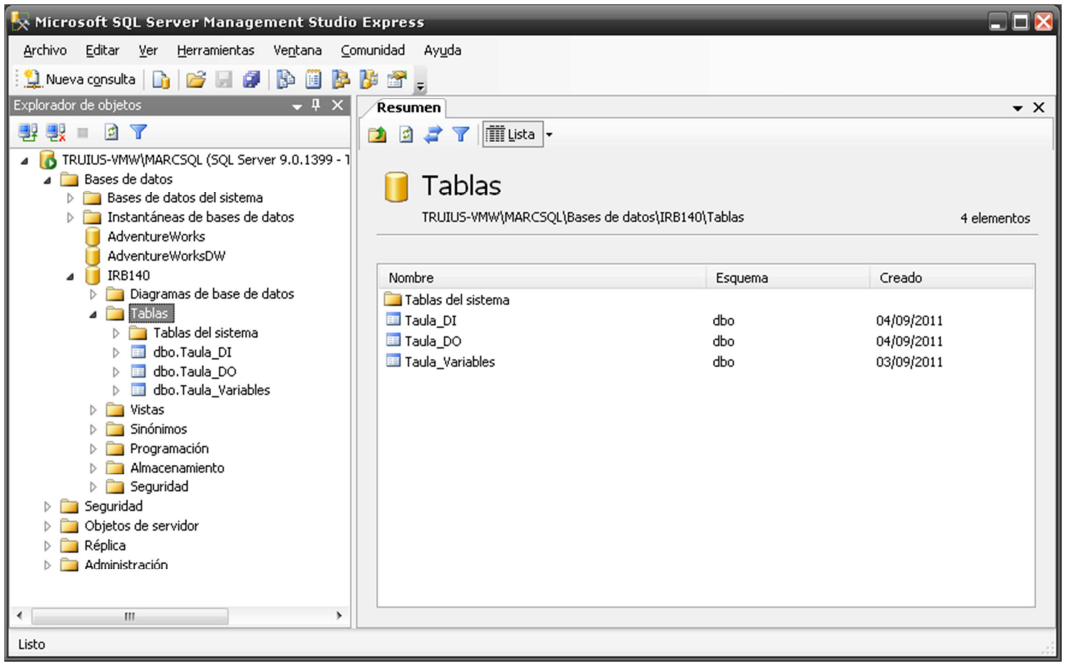


Figura 39. *BBDD utilitzada amb les 3 taules creades i configurades adequadament.*

Per al nostre cas, les taules utilitzades per a emmagatzemar la informació dels ítems afegits a l'aplicació de monitorització de dades es poden crear des de la pròpia aplicació client. No obstant, se n'ha creat una per defecte, *Taula_exemple*, i que serveix de punt de partida per si l'usuari no vol crear noves taules i/o desitja únicament fer petites proves.

En tots els casos, les taules tindran un total de 5 columnes:

Taula 1. *Columnes contingudes en les taules.*

Taula per desar dades des del client OPC		
Columna	**Tipus de dada**	**Descripció**
NumLectura	Int	Emmagatzema el número de fila, utilitzada principalment per a conèixer la cel·la a modificar
Dia_Hora	Datetime	Conté la data i hora en que s'ha efectuat l'enregistrament
ItemID	Text	Conté l'ídentificador o nom complet de l'ítem: "Canal.Dispositiu.Ítem"
Valor	Text	Conté el valor actualitzat de l'ítem.
Qualitat	Text	Conté la qualitat de la comunicació amb l'ítem en el moment del canvi de valor.

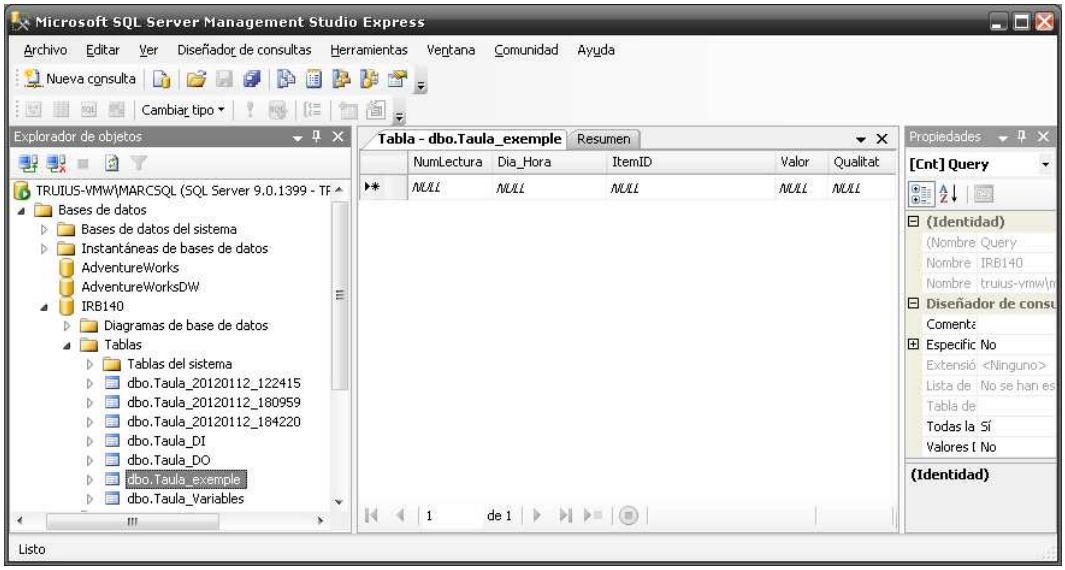


Figura 40. *BBDD utilitzada amb la taula per defecte creada i oberta amb les cinc columnes buides.*

10. MONITORITZACIÓ DE VARIABLES

Per a desenvolupar l'aplicació client que ha de realitzar la monitorització de les variables del robot IRB140 i de qualsevol altre servidor OPC connectat a la xarxa del laboratori farem ús del programari Visual Basic 6. Així mateix, l'aplicació també es comunicarà amb una Base de Dades on s'enregistraran els valors de totes les variables monitoritzades.

L'aplicació està dividida en dos formularis principals des d'on farem la monitorització dels ítems per una part i l'accés a la BBDD per l'altre. Al mateix temps, es farà ús d'un mòdul *.bas* i de quatre objectes *.Clss* els quals ens ajudaran a construir d'una forma més còmode la nostra aplicació.

Un mòdul .bas és aquell que conté bàsicament variables i funcions que seran públiques per a tots els formularis d'una aplicació . En aquest cas, nosaltres l'utilitzarem per a declarar la major part de tipus de variables utilitzades durant la comunicació client-servidor OPC i per a algunes variables de la base de dades utilitzades en els diferents formularis.

Els mòduls de classe .cls definirem els trs tipus de classes que tenim en el porjecte: servidor, grup i ítems. La última de les classes la utilitzarem per acabar de configurar l'objecte *browse*.

Per iniciar l'aplicació caldrà obrir l'executable Cient "*OPC.exe*" (Figura 41. Executable de l'aplicació.). Un cop oberta l'aplicació ens trobarem en la finestra principal (Figura 42. Finestra del formulari principal: "Aplicació client OPC".).

Des d'aquesta podrem realitzar totes les funcions possibles de l'aplicació, exceptuant la visualització de les dades enregistrades en una base de dades que es realitzarà des d'una altra finestra.

Figura 41. *Executable de l'aplicació.*

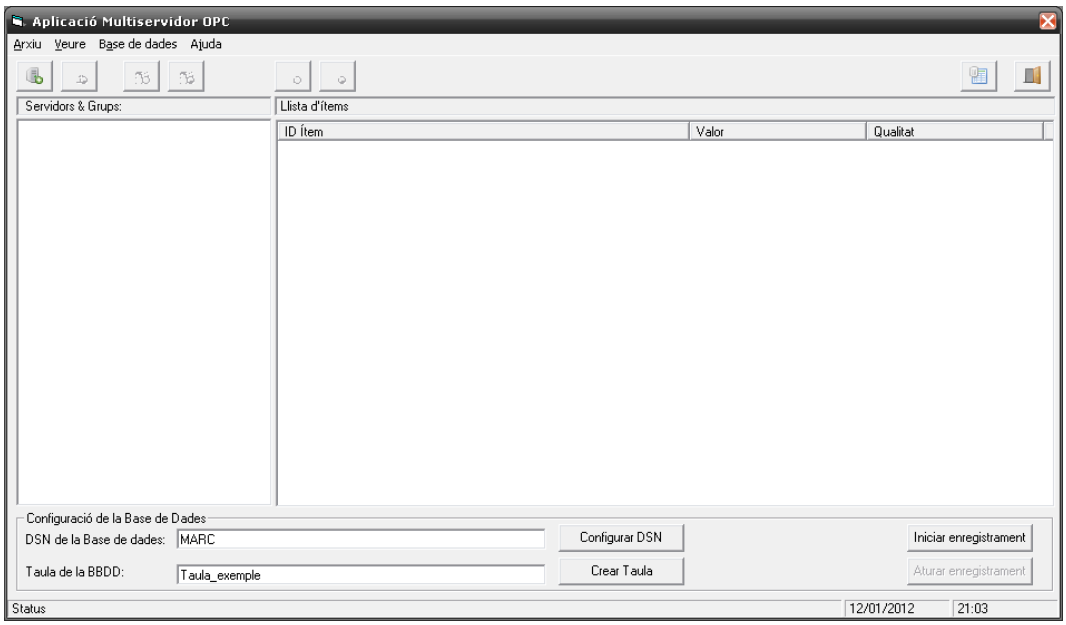


Figura 42. *Finestra del formulari principal: "Aplicació client OPC".*

Per començar a utilitzar l'aplicació cal afegir les variables o ítems dels servidors dins la llista. Des del moment en que s'afegeixi la primera de les variables, la llista d'ítems inserits s'anirà actualitzant periòdicament cada un cert temps marcat, el qual podrà ser modificat durant els transcurs de la monitorització.

Un cop afegits tots els ítems desitjats, l'aplicació ens permet desar aquest client en un format propi de forma que pugui ser carregat en posteriors sessions sense necessitat d'afegir un per un totes les variables que s'han afegit. Una altre possibilitat de l'aplicació és netejar el client actual per començar una nova sessió sense la necessitat de tancar i tornar a obrir l'aplicació.

Des d'aquesta finestra, podrem també iniciar la connexió amb la base de dades així com indicar a quina taula d'aquesta base de dades volem desar les dades o, si no existeix la taula, crear-la i començar l'enregistrament.

Finalment, des d'aquesta finestra es podrà fer la crida de la pantalla utilitzada per a visualitzar les dades desades en una taula d'una base

de dades (Figura 43. Finestra de visualització d'una taula d'una BBDD.) a la qual se li han afegit prèviament la informació.

Figura 43. *Finestra de visualització d'una taula d'una BBDD.*

10.1. Afegir ítems al client

El primer pas a realitzar en el nostre client és afegir tots aquells ítems que es desitgin visualitzar. Per a fer-ho tenim dues possibilitats:

1. Afegir els ítems d'un en un buscant primer el seu servidor i indicant a continuació a quin grup afegir-lo
2. Carregar un client prèviament desat. Aquesta segona opció s'explicarà detalladament en apartats posteriors d'aquest mateix capítol.

Si afegim els ítems d'un en un, l'adició tant de servidors com de grups com ítems es realitzarà mitjançant el conjunt de botons situats en la part superior esquerra de la finestra, sota el menú de l'aplicació.

Primer de tot haurem d'afegir el servidor OPC en el qual està allotjat. Per a tal fi cal prémer el botó destinat a tal fi, *Afegir servidor*. S'obrirà a continuació una nova finestra en la qual, després de carregar tots els servidors OPC disponibles, podrem seleccionar un i afegir-lo al nostre client.

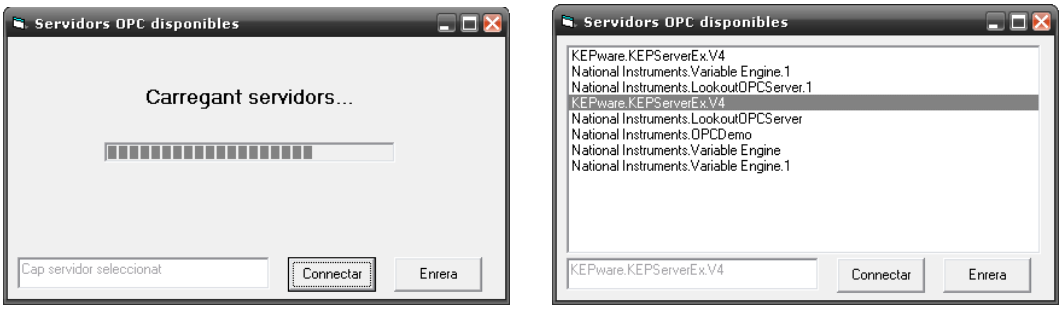


Figura 44. *Afegint un servidor. Carregant un servidor (esquerra) i seleccionant un servidor disponible (dreta).*

A continuació, cal afegir un grup al servidor. Primer hem de seleccionar el servidor que s'acaba d'afegir en el panell de l'esquerra de l'aplicació i, d'una manera semblant a l'anterior, mitjançant el botó en aquest cas *Afegir Grup* se'ns obrirà una nova finestra on haurem d'omplir els camps següents:

- Nom de grup: indica el nom del grups que afegirem
- UpdateRange: temps d'actualització del grup
- Deadband: valor mínim en que ha de variar una dada per a que pugui saltar l'esdeveniment DataChange (canvi en valor de dada)

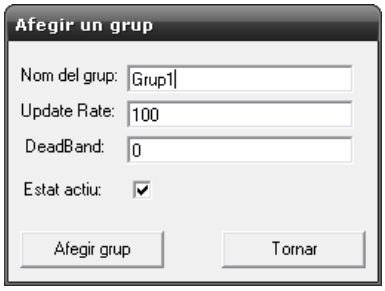


Figura 45. *Afegint un grup.*

Un cop ja tenim el servidor i el grup on afegirem l'ítem desitjat, cal seleccionar el grup primer, i fent ús del botó *Afegir Ítem* podrem seleccionar la variable a afegir al client OPC mitjançant una nova finestra, on a través d'un panell desplegable, podrem seleccionar l'ítem o bé, si es coneix la ruta d'accés directe, indicant aquesta en el camp "ITEM ID".

Aquesta finestra utilitzada per a incorporar els ítems al client disposa d'un botó per verificar que l'ítem està disponible i la comunicació amb

aquest és correcte, botó *Validar*. El resultat d'aquesta comprovació es pot visualitzar en el camp *Resposta de validació*.

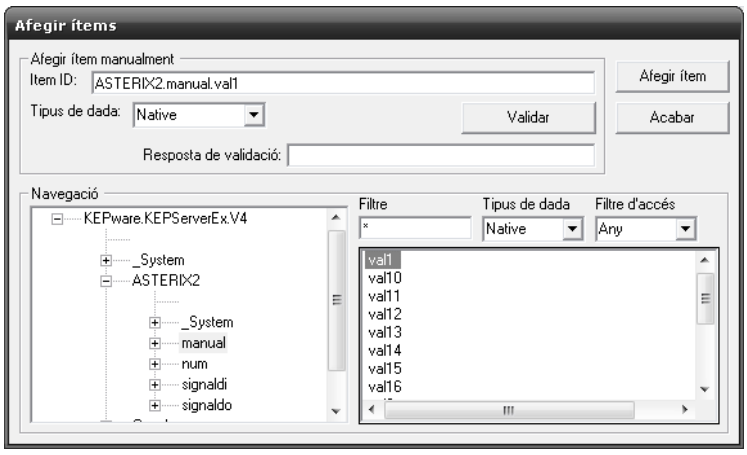


Figura 46. *Afegint un ítem.*

Anàlogament als botons per a inserir servidors, grups i ítems es disposa dels botons pertinents per a treure'ls o eliminar-los de l'aplicació client. Aquests estan situats al costat dels botons d'adició i en parelles, de forma que afegir/treure servidor OPC estan situats un al costat de l'altre, i el mateix succeeix amb grups i ítems.

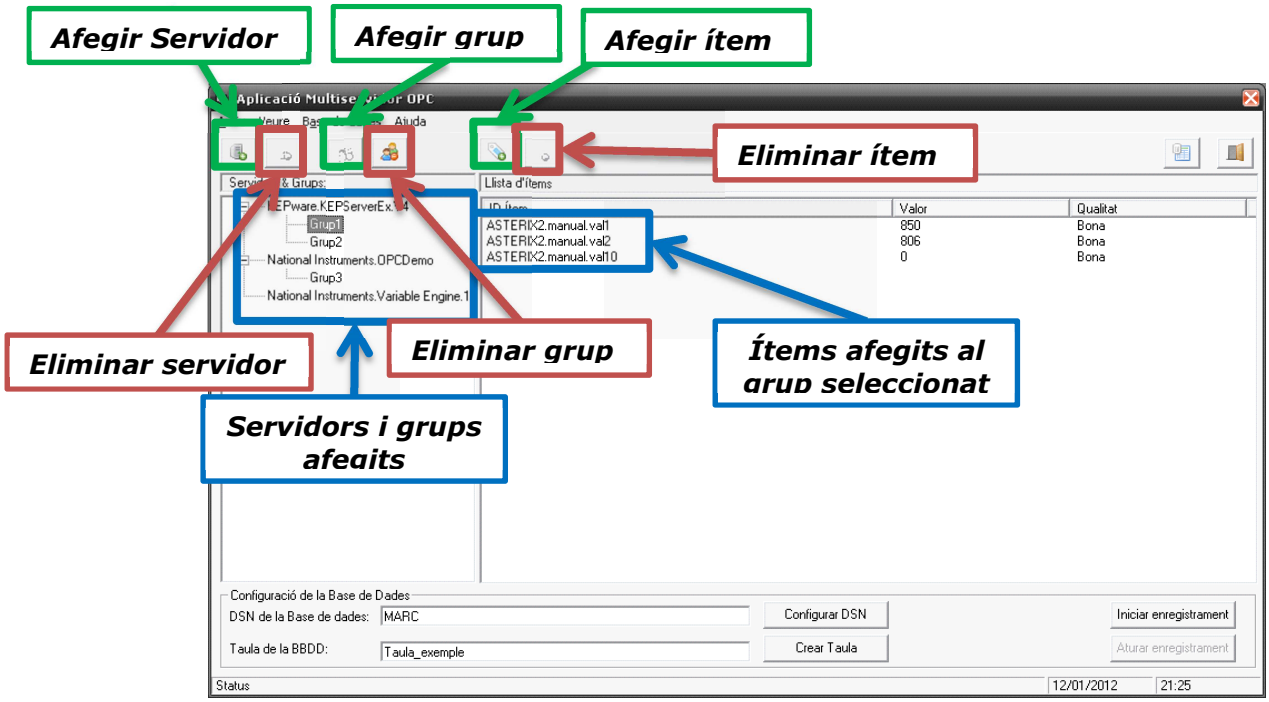


Figura 47. *Client amb servidors, grups i ítems afegits.*

Un cop afegit el primer dels diferents ítems que es desitgin visualitzar, el client anirà refrescant per pantalla el valor dels ítems amb un valor

de temps determinat, per defecte 50 ms. Aquest temps pot ser modificat mitjançant l'opció del menú *Temps d'actualització de dades* de la pestanya *Veure*. S'obrirà una finestra on indicarem el nou valor de refresc. Els valors permesos oscil·len entre els 50 ms i 1 s.

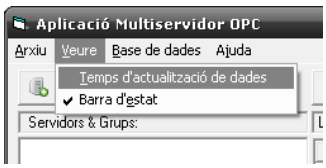


Figura 48. *Pestanya "Veure" del menú.*

10.2. Carregar/Desar/Iniciar un nou client

Un cop hem afegit tots els ítems desitjats, l'aplicació permet desar el client configurat fins aleshores de forma que en una sessió posterior no s'hagin d'afegir tots els ítems un per un sinó que permeti fer una càrrega d'un fitxer on es troba el client OPC desat.

Per a guardar el client serà suficient en clicar en l'opció *Desar client* del menú des de la pestanya *Arxiu* indicant el directori on es desitja desar l'arxiu. El format de l'arxiu és propi d'aquesta aplicació i es desarà amb l'extensió *.clo (Client OPC).

Un cop desat, podrem seguir amb el funcionament normal de l'aplicació, ja sigui continuant amb la monitorització dels ítems, afegir-ne de nous o treure'n els existents, o iniciar o obrir un nou client.

Si es desitja fer neteja de tota la informació inserida en el client i iniciar-ne'n un de nou, només cal prémer l'opció *Nou client* del menú des de la pestanya *Arxiu*. El client quedarà com si s'hagués obert per primer cop i llest per tornar a ser utilitzat.

Si el que es vol es fer una càrrega d'un client prèviament desat, caldrà accedir a l'opció *Obrir client* del menú des de la pestanya *Arxiu* indicant el directori on es troba l'arxiu del client. Aquesta acció provocarà que s'elimini qualsevol ítem afegint al client que teníem en funcionament abans de carregar el nou client OPC.

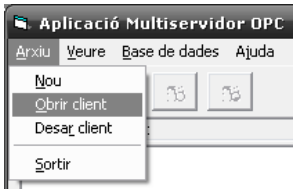


Figura 49. *Pestanya "Arxiu" del menú.*

10.3. Enregistrament de les dades a una base de dades.

Una altra opció que permet l'aplicació es la d'enregistrar tots els valors que prenguin els diferents ítems del client un una taula d'una base de dades. Per a tal fi es disposa dels camps de configuració situats en la part inferior de l'aplicació (Figura 50. Camps de configuració per a la base de dades de la finestra principal de l'aplicació.)

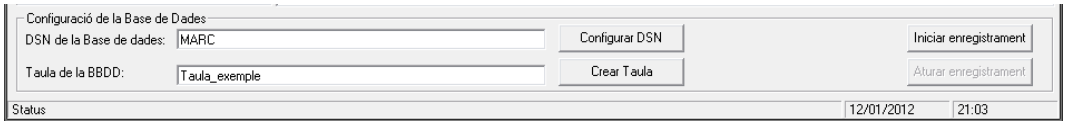


Figura 50. *Camps de configuració per a la base de dades de la finestra principal de l'aplicació.*

El primer camp s'utilitzarà per indicar la DSN a la qual tenim lincada la nostra base de dades. La DSN ha de ser configurada prèviament a fer l'enregistrament. En cas que l'usuari no hagi creat la DSN abans d'iniciar l'aplicació, aquesta incorpora un botó el qual cridarà a la funció de configuració de DSN de Windows. Aquesta funció també es pot cridar des de el menú principal dins la pestanya BBDD.

Figura 51. *Pestanya "Base de dades" del menú.*

Des d'aquí l'usuari podrà configurar la DSN per a ser utilitzada a continuació en l'aplicació client OPC. Els principals paràmetres que ha de passar l'usuari són la base de dades a la qual es farà l'accés i la

qual ha d'haver estat prèviament creada[1], el servidor on està allotjat la base de dades i el seu registre d'accés (nom d'usuari i contrasenya si en té).

Si ja es té la DSN configurada o bé s'acaba de configurar, cal introduir el nom amb el qual hem desat la configuració en el camp pertinent, *DSN de la base de dades*. A continuació cal indicar el nom de la taula on desarem el valor de les dades.

Si no tenim creada la taula o bé es vol crear una diferent per a noves dades caldrà prémer el botó *Crear Taula*. Tant si es desitja crear la taula com indicar a quina s'accedirà per a l'enregistrament, el no d'aquesta caldrà introduir-lo en el camp *Taula de la BBDD*.

Un cop introduïts el nom de la configuració DSN així com el nom de la taula a la qual hem d'accedir, per iniciar l'enregistrament caldrà prémer el botó *Iniciar enregistrament*. Un cop iniciat, no està permès fer canvis en cap dels dos camps anteriors i, per tant, quedaran inhabilitats.

Per aturar l'enregistrament, ja sigui per que no es vol continuar desant els valors a la base de dades com si es per a canviar la configuració de la DSN o de la taula amb la qual s'interactua, caldrà prémer el botó *Aturar enregistrament*. Tot i que s'aturi l'enregistrament, l'aplicació continuarà refrescant els valors mostrats per pantalla.

L'acció d'emmagatzematge s'executa únicament quan es detecta un canvi de valor en una variable mitjançant l'esdeveniment *DataChange*, ja que de fer-se de forma periòdica podríem omplir la taula d'informació redundant o bé perdre informació.

Els camps que s'emmagatzemen a la taula són el dia i hora en que ha canviat el valor de la dada, a quin ítem pertany la dada, el seu valor i la qualitat de la comunicació en el moment del canvi de valor.

10.4. Altres funcions

La resta de funcions que es poden realitzar des d'aquesta finestra són consultar la informació sobre l'autor de l'aplicació, cridar al manual

[1] Els passos per crear una BBDD es poden trobar al Capítol 9: Base de dades.

d'ajuda de l'aplicació o bé accedir a la finestra de visualització de les dades enregistrades en una taula d'una base de dades.

Figura 52. *Pestanya "Ajuda" del menú.*

10.5. Visualització d'una taula d'una base de dades.

Per accedir-hi cal prémer el botó *Visualitzar taula de BBDD* del formulari principal. En aquesta nova finestra podrem cridar i visualitzar una taula on haguem enregistrat anteriorment les dades del client OPC (Figura 43. Finestra de visualització d'una taula d'una BBDD.). El primer que s'ha de fer es indicar on ha d'anar a buscar i quina és la taula de la base de dades.

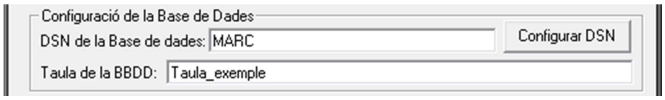


Figura 53. *Camps de configuració per a la base de dades de la finestra de visualització d'una taula d'una BBDD.*

Com en el cas per a configurar a quina taula desar la informació, tenim un espai reservat a complimentar els paràmetres per indicar la base de dades on està la taula desitjada mitjançant la DSN de connexió i el nom d'aquesta taula. Com passava en la finestra principal de l'aplicació, es disposa d'un botó per a cridar a la funció de configuració de DSN de Windows per si es necessita fer algun tipus d'ajust.

Un cop hem completat aquests dos camps ja només resta connectar-nos amb la taula i visualitzar-la en la graella situada a la meitat inferior de la finestra. Per a connectar-nos amb la taula serà suficient en prémer el botó *Connectar*.

La informació que es mostrarà en aquesta graella són el dia i hora en que ha canviat el valor de la dada, de quin ítem es tracta, el seu valor i la qualitat de la comunicació en el moment del canvi de valor en aquest ordre.

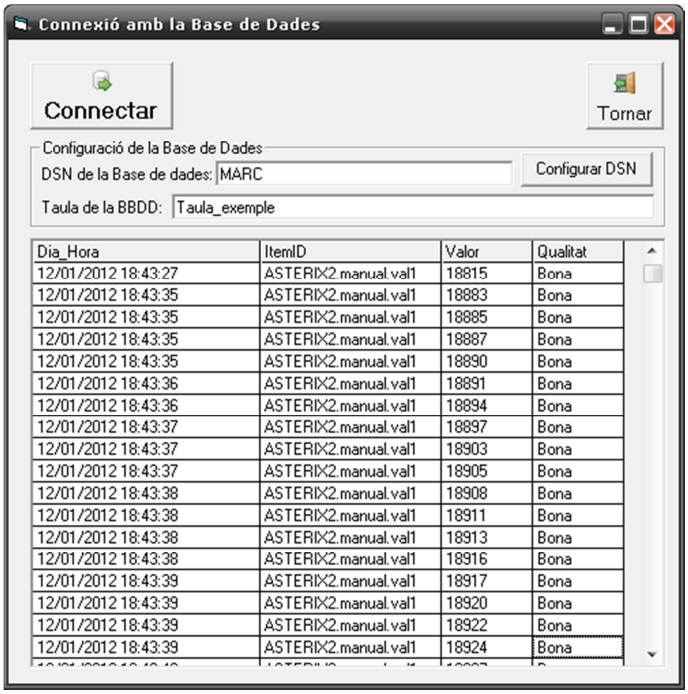


Figura 54. *Finestra de visualització d'una taula d'una BBDD amb la graella d'informació omplerta.*

Un cop hem acabat de revisar i/o mirar la informació proporcionada per la taula podrem tornar a la finestra principal mitjançant el botó Tornar.

11. MANIPULACIÓ DEL ROBOT

El desenvolupament del conjunt de l'aplicació es pot diferenciar en dues parts: una dedicada a la programació del robot mitjançant el llenguatge RAPID, específica d'ABB, i la part d'entorn virtual desenvolupat amb Visual Basic.

Amb Visual Basic es desenvoluparà tota la interfície gràfica des d'on demanar la petició per que el robot atengui a cada una de les màquines i de mostrar per pantalla les peces processades per cada una de les màquines i quin és l'estat tant del robot com de les màquines.

També permetrà fer un test de les comunicacions, per verificar que la interfície i robot es comuniquin, i reiniciar el compte de peces de les màquines.

La programació en RAPID estarà dedicada a controlar únicament el robot i de generar la cua d'atenció i portar el compte de peces processades a mode de simular un control des del propi robot, entenent que en una aplicació real les màquines, en comptes d'estar comandades des d'un únic PC, cada màquina tindria el seu i en la controladora del robot es tractaria la informació que anés arribant.

El comportament del robot serà sempre el mateix: anar a buscar una peça a qualsevol de les màquines disponibles i descarregar-la a una cinta transportadora. En cas de no atendre a cap màquina, el robot haurà de quedar en la posició de repòs.

A continuació s'exposa amb més detall la zona de treball suposada a l'hora de desenvolupar l'aplicació, la relació entre les variables tractades des de l'aplicació VB6 i el programa desenvolupat en llenguatge RAPID.

11.1. Zona de treball

Per a l'aplicació, es suposarà un entorn de treball específic amb una situació determinada dels diferents elements que hi participaran.

La zona de treball es plantejarà de forma que es tinguin 3 estacions: una de fresat, una d'escatat i una de soldadura. Cada una d'aquestes màquines se li suposarà un emplaçament físic fixe. Junt amb les màquines, també hi haurà una cinta per on extreure les peces processades per cada estació.

El robot estarà situat al mig de la zona imaginària de treball, de forma que pugui arribar sense complicacions a les tres estacions i a la cinta transportadora.

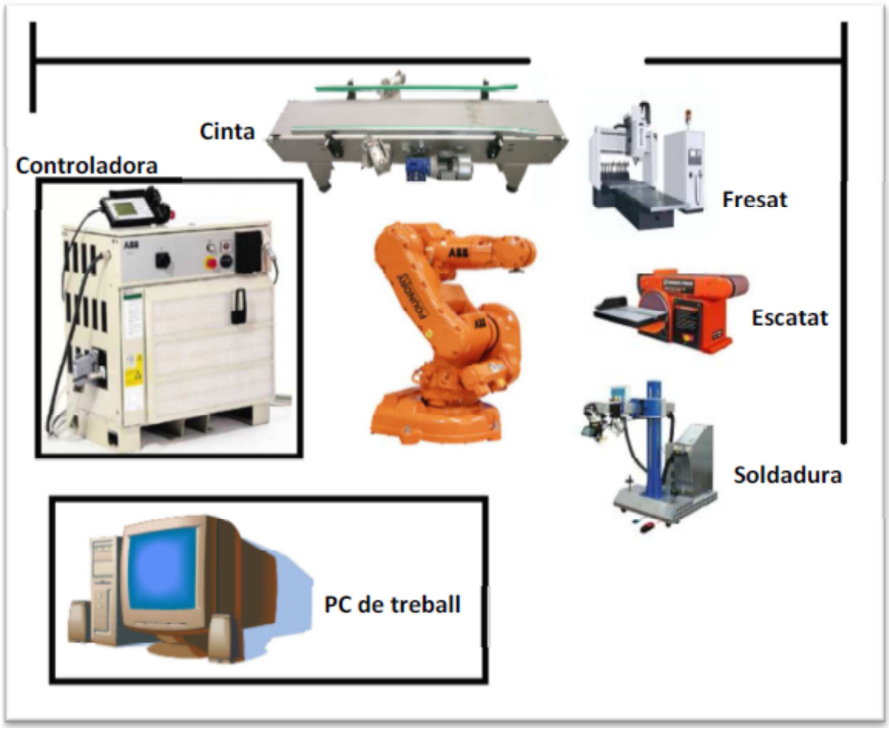


Figura 55. *Zona de treball del robot.*

Degut a l'espai limitat disponible al laboratori, les tres màquines estaran situades a la zona més amplia del laboratori, mentre que la cinta transportadora quedarà situada en paral·lel a la paret del despatx interior del laboratori. Els altres cantons del robot estan ocupats per la pròpia controladora del robot i per el PC des d'on llençarem la interfície amb Visual Basic.

11.2. Relació de variables VB-RAPID

A continuació, fent ús d'una taula, es mostra les diferents variables utilitzades per comunicar dades de Visual Basic a RAPID i a la inversa, de forma que sigui més fàcil entendre les parts de codi que puguin aparèixer d'aquí en endavant.

Podrem distingir les variables segons pertanyin a un llenguatge o a un altre ja que en el cas de RAPID, totes les variables porten la terminació "RAP" mentre que les de VB no porten cap indicatiu. Amb el sentit s'indica si la informació es passa de RAPID a VB o a l'inrevés, de VB a RAPID.

Taula 2. *Relació entre variables del programa RAPID amb variables de l'aplicació de manipulació del robot.*

Variable VB	Variable RAPID	Descripció	Sentit
comunicació	comunicacionRAP	S'utilitza per a testejar les comunicacions entre VB i RAPID.	De VB a RAPID
estat_test	comunicacionRAP	A la variable VB s'emmagatzema el valor retornat per la variable de RAPID quan es proven les comunicacions.	De RAPID a VB
peces_fresat	pFresadoraRAP	Indiquen el nombre de peces processades per la màquina de fresat.	De RAPID a VB
peces_escatat	pEscatatRAP	Indiquen el nombre de peces processades per la màquina d'escatat.	De RAPID a VB*
peces_soldadura	pSoldaduraRAP	Indiquen el nombre de peces processades per la màquina de soldadura.	De RAPID a VB*
estat_fresat	estatFreRAP	Indica l'estat de la estació de fresat.	De RAPID a VB*
estat_escatat	estatEscRAP	Indica l'estat de la estació d'escatat.	De RAPID a VB
estat_soldadura	estatSolRAP	Indica l'estat de la estació de soldadura.	De RAPID a VB

Variable VB	Variable RAPID	Descripció	Sentit
Estat_Robot	Estat_Robot_RAP	Indica l'estat del robot: a quina màquina s'està atenent o si s'està deixant una peça a la cinta transportadora.	De RAPID a VB

*Nota: per a aquestes variables, la comunicació es farà generalment tal i com s'indica, a excepció del cas de reiniciar el compte de peces moment en el qual es reiniciarà enviant un 0 des de la variable de VB a RAPID.

11.3. Interfície gràfica en Visual Basic

Des de la interfície gràfica que desenvoluparem en Visual Basic, podrem habilitar cada una de les tres màquines, així com fer la petició d'atenció de cada una d'elles, provar si s'estableix comunicació amb el robot i reiniciar el nombre de peces comptades que hagi processat cada màquina. A més, serà l'encarregada de mostrar la informació de quines màquines estan habilitades, l'estat de cada màquina i del robot.

Per a les màquines se'ls hi donarà 3 estats possibles:

- Parada: se li suposarà que ja ha processat la peça següent i encara no s'ha demanat la descàrrega de la peça al robot,
- Esperant: s'ha demanat petició per que el robot reculli la peça de la màquina però encara no s'està atenent, i
- Descarregant: el robot està descarregant la peça de la màquina.

Per al robot, s'ha fixat 5 estats possibles:

- Parat: el robot no està atenent a cap màquina ni deixant la peça a la cinta, està en posició de repòs,
- Atenent a Fresat: el robot està agafant la peça de la màquina de fresat,
- Atenent a Escatat: : el robot està agafant la peça de la màquina d'escatat,
- Atenent a Soldadura: el robot està agafant la peça de la màquina de soldadura,

- Descarregant a cinta: el robot està descarregant la peça a la cinta.

L'aplicació s'ha dividit en dos finestres o formularis: una finestra principal on es mostrarà tota la informació, i una última des d'on fer el test de comunicació i el reinici del comptes de peces.

11.3.1. Finestra principal

En aquesta finestra és on es fa la major part d'instruccions i serà des d'on l'usuari farà les peticions per atendre a les diferents màquines. La finestra la podem dividir en diferents zones:

- a la part esquerra trobem tota la informació de cada una de les tres màquines disponibles,
- a la part dreta superior trobem les opcions per habilitar cada una de les estacions,
- a la dreta i centrada trobem la informació sobre l'estat de la comunicació i si s'ha reiniciat o no els comptadors,
- i finalment a la part inferior-dreta trobem l'estat del robot i la màquina a la qual s'està atenent.

Figura 56. *Finestra principal de l'aplicació de manipulació del robot.*

Si ens fixem en la part esquerra de la finestra, tenim la informació de les tres màquines ordenades com estació de fresat, estació d'escatat i estació de soldadura. Per a cada una d'elles es mostra si està habilitada o no, el tipus de funcionament en que es troba (un dels 3 estats possibles comentats anteriorment: parada, esperant o descarregant) i el número de peces descarregades des de l'últim reinici del comptador respectiu.

El botó de descàrrega només s'habilitarà un cop habilitem la màquina. També s'ha forçat que un cop premem el botó de descàrrega, s'inhabiliti fins que el robot no hagi descarregat la peça de la màquina. Un cop la peça ha estat descarregada, el botó estarà habilitat novament de forma automàtica.

Per poder iniciar les descàrregues de peces haurem d'habilitar primer les màquines. Això ho podrem fer marcant les caselles destinades a tal fi en la part superior-dreta de la finestra actual.

Un cop estigui marcada una estació, aquesta apareixerà habilitada junt amb la informació corresponent a si la màquina està habilitada, en una altra etiqueta l'estat de la màquina (parada, esperant a ser atesa o si el robot està descarregant la seva peça) i en la última el nombre de peces.

En la Figura 57. Estació habilitada. podem veure el cas particular d'una de les màquines quan aquesta està habilitada. Com es pot veure, apareix en les 3 etiquetes de cada màquina la informació pertinent i el botó de descàrrega habilitat.

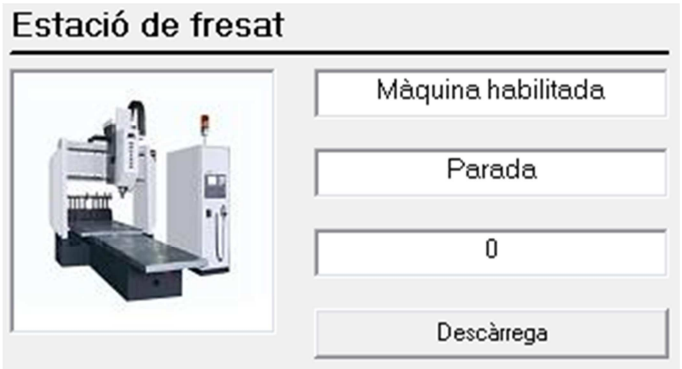


Figura 57. *Estació habilitada.*

A continuació es mostra la part de codi per a una de les màquines per al tractament de la informació que ha d'aparèixer en cada una de les

etiquetes, així com aspectes visuals addicionals com el canvi de format del punter al passar per sobre el botó de descàrrega depenent de l'estat en que es trobi la màquina :

Codi 9. *Tractament de la informació a mostrar en les etiquetes per a cada una de les màquines.*

```
'Estació Fresat
If Hab_fresat.Value = 0 Then 'fresadora inhabilitada...
      Desc_fresat.Enabled = False 'No boto descarrega
      Hab_fres_label.Caption = "Màquina inhabilitada"
      Estat_fresat_label.Caption = "--------------------"
      Desc_fresat.MousePointer = 12 'Mouse amb senyal de prohibit
Else     'fresadora habilitada (Hab_fresat.Value = 1)...
      Hab_fres_label.Caption = "Màquina habilitada"
      Desc_fresat.MousePointer = 0 'Mouse amb senyal fletxa
      Status = Helper1.S4ProgramNumVarRead("estatFreRAP", 0,
            estat_fresat)     'Llegim estat de màquina a RAPID
      Select Case (estat_fresat) 'Mostrar estat Fresadora
        Case 0: Estat_fresat_label.Caption = "Parada"
            Desc_fresat.MousePointer = 0 'Mouse amb senyal
                                         'prohibit
            Desc_fresat.Enabled = True 'boto descàrrega habilitat
        Case 1: Estat_fresat_label.Caption = "Esperant"
            Desc_fresat.MousePointer = 11 'Mouse amb senyal
                                          'rellotge
            Desc_fresat.Enabled = False
        Case 2: Estat_fresat_label.Caption = "Descarregant..."
            Desc_fresat.MousePointer = 12 'Mouse amb senyal fletxa
            Maq_atesa_label.Caption = "Descarregant peça d'estació
                                                      de Fresat"
      End Select
End If
...
{manca part de codi per a les altres màquines}
...
'Lectura i mostreig de peces processades per màquina
Status = Helper1.S4ProgramNumVarRead("pFresadoraRAP", 0,
peces_fresat)
Peces_fres_label.Caption = peces_fresat
Status = Helper1.S4ProgramNumVarRead("pEscatatRAP", 0,
peces_escatat)
Peces_esc_label.Caption = peces_escatat
Status = Helper1.S4ProgramNumVarRead("pSoldaduraRAP", 0,

peces_soldadura)
Peces_sold_label.Caption = peces_soldadura
```

Primer es fa tot el tractament oportú segons l'estat llegit de RAPID de la màquina, i un cop fet això per a les 3 màquines, es fa una lectura de les peces processades per cada màquina de forma seguida.

Si ens fixem ara en la part inferior dreta de la finestra, tal i com s'ha comentat anteriorment, hi trobem la informació de l'estat del robot: quin tipus de moviment està fent i la màquina a la qual està atenent.

En aquest cas, llegim al programa fet en RAPID les variables que indiquen l'estat de les màquines i la de l'estat del robot. Com la lectura de l'estat de les màquines ja s'ha fet per mostrar aquesta informació en les etiquetes de cada una de les màquines, en aquesta part només ens caldrà llegir l'estat del robot.

Com s'indica a l'inici d'aquest apartat, els possibles estats del robot són 5: estar en moviment cap a les màquines de fresat, escatat o soldadura, cap a la cinta transportadora, o bé estar en la situació de repòs; i que tenen els valors 1, 2, 3, 4 i 0 respectivament.

Per mostrar per pantalla la informació de l'estat del robot, s'utilitzarà l'estructura d'un *Select Case*, i mostrarem la informació pertinent per a l'etiqueta dedicada a tal fi.

Codi 10. *Detecció i visualització de l'estat del robot.*

```
'Mostrar on està el Robot: moviment
Status = Helper1.S4ProgramNumVarRead("Estat_Robot_RAP",0,
Estat_Robot)
Select Case (Estat_Robot)
      Case 1: Mov_robot_label.Caption = "Atenent a màquina fresat"
      Case 2: Mov_robot_label.Caption = "Atenent a màquina
escatat"
      Case 3: Mov_robot_label.Caption = "Atenent a màquina
soldadura"
      Case 4: Mov_robot_label.Caption = "Robot descarregant en
cinta"
      Case 0: Mov_robot_label.Caption = "Robot parat"
End Select
```

Per mostrar la màquina a la qual està atenent, vegis que una part es fa dins del propi tractament d'informació per a les màquines, dins del Select Case quan aquest val 2 (màquina descarregant). Això provocarà que si una de les màquines s'està descarregant automàticament es mostri el text de la màquina que s'està atenent per l'etiqueta de màquina atesa.

El problema encara no resolt és quan no s'està atenent a cap màquina, ja que no es mostraria cap text o en el pitjor dels casos seguiria impresa l'última màquina atesa. Per a solucionar-ho serà suficient en introduir el text de cap màquina atesa en cas que no hi hagi cap descarregant-se. La part de codi per a mostrar aquesta informació és el següent:

Codi 11. *Detecció i visualització de màquina atesa.*

```
...  {manca part de codi}  ...
Case 2: Estat_fresat_label.Caption = "Descarregant..."
  Desc_fresat.MousePointer = 12 'Mouse amb senyal fletxa
  Maq_atesa_label.Caption = "Descarregant peça d'estació de
Fresat"
...  {manca part de codi}  ...
Case 2: Estat_escatat_label.Caption = "Descarregant..."
  Desc_escatat.MousePointer = 12
  Maq_atesa_label.Caption = "Descarregant peça d'estació
d'Escatat"
...  {manca part de codi}  ...
Case 2: Estat_soldadura_label.Caption = "Descarregant..."
  Desc_soldadura.MousePointer = 12
  Maq_atesa_label.Caption = "Descarregant peça d'estació de
Soldadura"
...  {manca part de codi}  ...
'Si no s'està atenent a cap màquina
If (estat_fresat<>2) And (estat_escatat<>2) And
(estat_soldadura<>2)
                        Then
      Maq_atesa_label.Caption = "No s'atén a cap màquina"
End If
```

Tota aquesta actualització d'informació, la part d'informació de les màquines, el nombre de peces processades per cada màquina, l'estat del robot i la màquina atesa, es farà dins d'un Timer. Per tant, tota aquesta part de codi es farà de forma continuada i periòdicament fins que es tanqui l'aplicació.

Per a la petició de descàrrega, disposem d'un botó per a cada màquina dins de la seva zona d'informació. Mitjançant aquest botó, activarem un TRAP del programa escrit en RAPID per a que la màquina sigui atesa un cop el robot estigui lliure. El codi per activar el TRAP és el mostrat a continuació i igual per a cada una de les màquines variant únicament el nom del TRAP:

Codi 12. *Tractament dels botons de descàrrega.*

```
'************** Tractament dels botons de descàrrega **************
'Peticio atenció fresat
Private Sub Desc_fresat_Click()
      Status =
Helper1.S4DigitalWrite("DO10_11",1,ResultSpec,ResultID) End Sub
'Peticio atenció escatat
Private Sub Desc_escatat_Click()
      Status =
Helper1.S4DigitalWrite("DO10_12",1,ResultSpec,ResultID) End Sub
'Peticio atenció soldadura
Private Sub Desc_soldadura_Click()
      Status =
Helper1.S4DigitalWrite("DO10_13",1,ResultSpec,ResultID)
End Sub
```

El tractament de cua per a les màquines es tracta dins de RAPID i s'explicarà en el corresponent apartat.

Finalment, a la part central al cantó dret i trobem la part d'informació de l'estat de les comunicacions i el reinici dels comptadors de peces. En aquest cas, la informació es recull de la finestra ja comentada de *Test&Reset* a la qual podrem accedir clicant sobre el botó 

.

Si l'usuari desitja sortir, només caldrà prémer el botó *Sortir de l'aplicació*. No obstant, per seguretat, s'ha forçat a que les màquines estiguin inhabilitades i que, de no ser així, surti un missatge d'error evitant el tancament de l'aplicació. En cas de tenir les màquines parades i inhabilitades, l'aplicació enviarà un nou TRAP al codi RAPID per a activar la senyal de sortida del sistema i possibilitar un tancament adequat.

Codi 13. *Codi per sortir de l'aplicació amb màquines inhabilitades.*

```
'******************** Sortir de l'aplicació **********************
Private Sub Sortir_principal_Click()
  If Hab_fresat.Value = 1 Or Hab_escatat.Value = 1 Or

Hab_soldadura.Value = 1 Then
      Dim Missatge, Estil, Titol, Resposta, MiString As String
      Missatge = "S'han d'inhabilitar les màquines i que totes
elles                                                        restin
parades."
      Estil = vbOKOnly + vbCritical + vbDefaultButton2 'Defineix
                                                       'els botons.
      Titol = "Compte!!"
      Resposta = MsgBox(Missatge, Estil, Titol)
  Else
      'Peticio TRAP de sortida
      Status =
Helper1.S4DigitalWrite("DO10_15",1,ResultSpec,ResultID)
      Unload Inici
      Unload Test_Reset
      Unload Reiniciar
      Unload Principal
  End If
End Sub
```

11.3.2. Finestra de Test&Reset

La utilitzarem per testejar la comunicació entre l'aplicació i el robot i per a reiniciar els comptadors de les peces descarregades des de cada una de les diferents màquines. A la mateixa finestra, es mostrarà per pantalla l'estat de les comunicacions i si s'han reiniciat els comptadors.

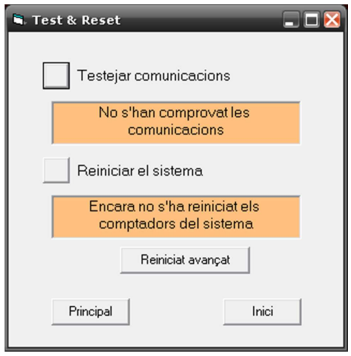


Figura 58. *Finestra de Test & Reset.*

Per a cada un dels possibles estats de les comunicacions es donarà el missatge en text i amb el fons acolorit depenent de cada un dels 3 estats possibles, i que són següents:

- que no s'hagi testejat,
- que el test sigui correcte i la comunicació es realitzi de forma satisfactòria, o
- que fallin les comunicacions.

Respecte al codi, el que es fa en aquest cas es escriure directament a la variable de comunicació *comunicacionRAP* un valor diferent de l'inicial (enviem variable amb valor igual a 1), i s'espera retornar un tercer valor diferent als altres dos (esperem rebre un 2, valor sobreescrit en el programa desenvolupat en RAPID).

Ambdues rutines apareixen el codi següent marcades en color blau. Es comprova així que funciona tan l'escriptura com la lectura en comunicacions.

Codi 14. *Codi per testejar comunicacions.*

```
comunicacio = 1   'Posem comunicació amb valor 1

Status = Principal.Helper1.S4ProgramNumarWrite("comunicacionRAP",
    0, comunicacio, ResultSpec, ResultID) 'Escribim a RAPID

Status = Principal.Helper1.S4ProgramNumVarRead("comunicacionRAP",
    0, estat_test)'Llegim de RAPID

If estat_test = 2 Then 'Si llegim valor 2 (correcte)
     Test_label.Caption = "Test realitzat de forma correcte"
     Test_label.BackColor = &H80FF80 '* Fons Test = verd
     Principal.Test_label.Caption = "Test realitzat de forma
correcte"
     Principal.Test_label.BackColor = &H80FF80 '* Fons Test = verd
Else        '..en cas contrari error en comunicacions
     Test_label.Caption = "Errata en les comunicacions"
     Test_label.BackColor = &HFF& '* Fons Test = vermell
```

```
     Principal.Test_label.Caption = "Errata en les comunicacions"
     Principal.Test_label.BackColor = &HFF& '* Fons Test = vermell
End If
```

Per al cas de la posta a 0 dels comptadors de peces, a més de l'opció de reiniciar tots al mateix temps, hom podrà triar fer el reinici del comptador d'una o més màquines de forma individual mitjançant la nova finestra dedicada a tal fi prement el botó "*Reinici avançat*".

Figura 59. *Finestra per al reiniciar els comptadors individualment.*

Amb l'opció de *Reiniciar comptadors* es reinicien tots els comptadors seleccionats i es passa de forma automàtica a la finestra principal. En cas de prémer a *Tornar*, serem enviats a la finestra de *Test & Reset* sense reiniciar els comptadors de peces encara que hagin estats seleccionats.

Sigui quin sigui el cas per a la posta a 0 dels comptadors de peces (automàtica o individual), el mètode seguit és escriure directament a les variables del programa RAPID i després fer la lectura del valor per a comprovar que el reinici es fa correctament.

Les línies de codi que permeten reiniciar els comptadors de forma individual són les mostrades a continuació, mentre que per al cas de reiniciar-los tots junts es farà únicament l'escriptura i lectura i d'una tirada:

Codi 15. *Codi per reiniciar comptadors.*

```
If R_Fre.Value = 1 Then            'Reiniciem estació Fresat
      Principal.peces_escatat = 0
      Status =
Principal.Helper1.S4ProgramNumPersWrite("pEscatatRAP",
      0, Principal.peces_escatat, 0, ResultID)
      Status =
Principal.Helper1.S4ProgramNumVarRead("pEscatatRAP", 0,
      Principal.peces_escatat)
      Principal.Hab_fresat = False
End If
If R_Esc.Value = 1 Then            'Reiniciem estació Escatat
```

```
      Principal.peces_fresat = 0
      Status =
Principal.Helper1.S4ProgramNumPersWrite("pFresatRAP",
      0, Principal.peces_fresat, 0, ResultID)
      Status = Principal.Helper1.S4ProgramNumVarRead("pFresatRAP",
0,               Principal.peces_fresat)
      Principal.Hab_escatat = False
End If
If R_Sol.Value = 1 Then            'Reiniciem estació Soldadura
      Principal.peces_soldadura = 0
      Status =
Principal.Helper1.S4ProgramNumVarWrite("pSoldaduraRAP",
      0, Principal.peces_soldadura, 0, ResultID)
      Status =
Principal.Helper1.S4ProgramNumVarRead("pSoldaduraRAP",
      0, Principal.peces_soldadura)
      Principal.Hab_soldadura = False
End If

'Amb les següents línies mostrem informació per pantalla Principal
Principal.Reset_label.Caption = "Comptadors reiniciats"
Principal.Reset_label.BackColor = &H80FF80 '* Fons Reset = verd
Test_Reset.reset_var = True
Test_Reset.Reset_label.Caption = "Comptadors reiniciats"
Test_Reset.Reset_label.BackColor = &H80FF80 '* Fons Reset = verd
```

Notis que, un cop reiniciem els comptadors, al mateix temps inhabilitem les estacions, de manera que l'usuari hagi de tornar ha habilitar-les en cas que les vulgui manipular-les.

11.4. Programació en RAPID

El llenguatge RAPID és un llenguatge estructural format per un programa i el seu conjunt de mòduls per a realitzar les diferents tasques que es vulguin programar.

En aquest mòdul hauran d'aparèixer primer les dades del programa, utilitzades per a definir les diferents variables que es faran servir, la rutina principal (*main*), on s'inicia i es fa l'execució del programa, i les sub-rutines, parts d'execució de programa que es cridaran des de la rutina principal i que ens permeten dividir el programa en parts més petites obtenint així un programa modular més senzill i fàcil d'entendre.

El programa RAPID utilitzat manipula el robot de forma que aquest atengui a les diferents màquines un cop l'usuari ens demani petició des del programa en Visual Basic, al mateix temps que haurà de portar el compte de peces processades.

Permetrà també el reinici dels comptadors quan se li demani, i saber i comunicar l'estat del robot, tant en tipus de moviment com de

màquina atesa. Des de l'aplicació RAPID, també es portarà la gestió de la cua per atendre a totes les màquines.

No obstant, abans de començar a generar l'aplicació RAPID, primer haurem d'agafar els punts ficticis de posició de les màquines i de la cinta, així com un punt de repòs per al robot quan aquest no efectuï cap tipus de moviment. Un cop agafats, ens quedarà unes instruccions com les mostrades a continuació, les quals introduirem dins de les dades de programa, dins del mòdul principal:

Codi 16. *Declaració dels punts de posició utilitzats com a constants.*

```
! Constants de posició:
! ---------------------
! Repòs
CONST robtarget pCal:=[[516.64,0.37,703.33],...,9E+09,9E+09]];
! Punts màquines: mXY (X: tipus de màquina
                      !Y: punt d'aproximació (1)/ punt final(2))
CONST robtarget m12:=[[-628.26,-341.99,419.63],...,9E+09,9E+09]];
CONST robtarget m11:=[[-622.31,-364.84,513.68],...,9E+09,9E+09]];
CONST robtarget m22:=[[-120.48,-837.61,446.82],...,9E+09,9E+09]];
CONST robtarget m21:=[[-138.62,-714.31,475.13],...,9E+09,9E+09]];
CONST robtarget m32:=[[385.96,-644.56,480.64],...,9E+09,9E+09]];
CONST robtarget m31:=[[406.61,-612.01,689.37],...,9E+09,9E+09]];
! Punts cinta
CONST robtarget c3:=[[726.14,21.21,213.75],...,9E+09,9E+09]];
CONST robtarget c1:=[[723,89.56,425.29],...,9E+09,9E+09]];
```

Per a la nomenclatura dels punts, pCal és el punt de repòs del robot quan no tingui que atendre a cap màquina, c1 i c3 són els punts on haurà d'anar el robot per deixar les peces a la cinta, essent c3 el punt per situar-se i c1 el punt on deixar la peça.

Finalment, mXY indica que s'està atenent a una màquina, on el número X indica la màquina en concret (3.-Fresat/2.-Escatat/1.-Soldadura) i la Y indica el punt d'aproximació fins a la màquina o si es tracta del punt precís on agafar la peça (valors 1 i 2 respectivament).

11.4.1. Variables utilitzades en l'aplicació

Un cop tenim tant la capçalera com les instruccions pertinents per a declarar el mòdul principal i els punts declarats, ja podem començar amb la resta de codi. Primer definirem les diferents variables que necessitarem per a la nostra aplicació, de les quals destaquen les destinades a fer el compte de peces i declarar l'estat de màquines i robot, ja que hauran de ser del tipus persistent per a mantenir el

nombre de peces produïdes i permetre la seva posterior connectivitat amb el servidor OPC.

La resta de variables, o bé ja s'han descrit en l'apartat anterior 11.2 Relació de variables VB-RAPID o bé s'utilitzaran de forma interna per a la correcta gestió de l'aplicació. A continuació es mostra tota la declaració de les variables que s'utilitzaran agrupades segons la seva funció.

Codi 17. *Declaració de les variables utilitzades en aplicació RAPID.*

```
    ! Variables numèriques per a comptar peces
    PERS num pFresadoraRAP:=0;
    PERS num pEscatatRAP:=0;
    PERS num pSoldaduraRAP:=0;

    !Variables seqüencia a seguir i peticions
    VAR num seguent1:=0;      !Primer element en cua
    VAR num seguent2:=0;      !Segon element en cua
    VAR num seguent3:=0;      !Tercer element en cua
    VAR num peticioFre :=0;   !Petició Fresat
    VAR num peticioEsc :=0;   !        Escatat
    VAR num peticioSol :=0;   !        Soldadura

    ! Variables comunicació
    PERS num comunicacionRAP:=0;    !var de comunicació
    VAR num sortirRAP:=0;   !var per executar mòdul principal o
tancar
    PERS num Estat_Robot_RAP:=0;    !0->Parat a repòs, 1->Fresat,
                                    !2->Escatat, 3->Soldadura,
                                    !4->Cinta

    ! Variables estat màquines
    PERS num estatFreRAP :=0; !0->Parada,1->Espera,2->Descarregant
    PERS num estatEscRAP :=0;
    PERS num estatSolRAP :=0;

    ! Variables TRAP
    VAR intnum maqFre;
    VAR intnum maqEsc;
    VAR intnum maqSol;
    VAR intnum acabar;
    ! ---------------------------
    ! Fi declaració variables
```

Un punt a destacar són les variables introduïdes en el programa RAPID, que són les mateixes que les subscrites en el servidor. A continuació es mostra una taula amb totes les variables numèriques utilitzades des del programa RAPID amb la seva correspondència tant en el servidor com en la base de dades:

Taula 3. *Correspondència entre variables del programa RAPID amb variables en el servidor i de la BBDD.*

Variable RAPID	Variable en servidor	Variable en BBDD
pEscatatRAP	ASTERIX2.num.pEscatatRAP	Peces_Escatat
pFresadoraRAP	ASTERIX2.num.pFresadoraRAP	Peces_Fresadora
pSoldaduraRAP	ASTERIX2.num.pSoldaduraRAP	Peces_Soldadura
Estat_Robot_RAP	ASTERIX2.num.Estat_Robot_RAP	Estat_Robot
estatEscRAP	ASTERIX2.num.estatEscRAP	Estat_Escatat
estatFreRAP	ASTERIX2.num.estatFreRAP	Estat_Fresat
estatSolRAP	ASTERIX2.num.estatSolRAP	Estat_Soldadura

11.4.2. Rutines del mòdul principal

Un cop definides les variables, posarem les instruccions de les diferents subrutines que s'utilitzaran. Concretament es farà ús de 6 rutines: les dues primeres per a les seqüències d'agafar la peça de la màquina i deixar-la a la cinta respectivament, tres més per atendre a cada una de les diferents màquines, i una última per anar a la posició de repòs.

Notis que, per al cas de les rutines d'agafar i deixar peça, es podrien haver agrupat en una única rutina, però s'ha preferit deixar-les com dues rutines diferents ja que depenent de la màquina d'on s'extregui la peça ens pot interessar una posició diferent del canell per a cada cas o bé que es necessitin velocitats diferents, o qualsevol altra necessitat.

Per a la rutina d'agafar peça, primer ens aproximarem a la màquina (*pos_allunyament*), moviment el qual es pot fer amb una velocitat més alta i menor precisió.

A continuació ens moure'm a la posició on agafar la peça amb menor velocitat i major precisió (*pos_aproximacio*) i ens esperarem un temps suficient per agafar la peça. Finalment tornem a la primera de les posicions per treure el robot amb la peça de la màquina.

Notis que per a aquesta rutina, haurem d'indicar ambdues posicions de la màquina: aproximació a màquina i punt d'acció sobre la màquina.

Per a la rutina per deixar la peça a la cinta transportadora, els moviments són anàlegs, però en aquest cas no caldrà entrar a la rutina

les dues posicions, aproximació i punt d'acció, ja que els punts de la cinta seran fixes. En aquest últim cas, indicarem també que l'estat del robot és el de deixant peça a cinta.

Codi 18. *Rutines per agafar o deixar peça.*

```
! Rutines agafar i deixar peça
    ! ----------------------------
    PROC agafar_peca(robtarget pos_aproximacio, robtarget

      pos_allunyament)
      MOVEJ pos_allunyament,V800,fine,tool0;
      MOVEJ pos_aproximacio,v50,fine,tool0;
      WaitTime 2; !Espera 2 segons per agafar peça
      MOVEJ pos_allunyament,V50,fine,tool0;
    ENDPROC

    PROC deixar_peca()
            Estat_Robot_RAP:=4;
            MOVEJ c1,V800,fine,tool0;
            MOVEJ c3, v50, fine, tool0;
            WaitTime 1; !Espera 1 segon per deixar peça
            MOVEJ c1,V50,fine,tool0;
    ENDPROC
```

Per al cas de les rutines d'atenció a màquina, primer actualitzarem el valor de l'estat del robot per a mostrar per el programa en Visual Basic la màquina a la qual s'està atenent. A continuació es crida a la rutina per agafar la peça, indicant les dues posicions necessàries, i seguidament cridem a la rutina per deixar la peça a la cinta transportadora. Finalment, augmentem en un el valor del comptador de peces, indicant que hem descarregat una peça més.

A continuació es presenta només les instruccions per a una de les màquines. Per a la resta de màquines, només variaran les posicions i el comptador a actualitzar.

L'última de les rutines, pertany al moviment del robot a la posició de repòs i l'actualització de la variable d'estat del robot amb el valor per indicar que aquest resta parat.

Codi 19. *Rutines d'atenció a màquines i posició de repòs.*

```
! Rutines atenció màquines
    ! ---------------------------
    PROC go_fresadora()
            Estat_Robot_RAP:=1;
            agafar_peca m32, m31;
            deixar_peca;
            pFresadoraRAP:=pFresadoraRAP+1;
    ENDPROC
```

```
    ... {rutines d'atenció a les altres màquines} ...

PROC go_repos()
        Estat_Robot_RAP:=0;
        MoveJ pCal, v150, z10, tool0;
ENDPROC
! ---------------------------
! Fi rutines atenció màquines
```

11.4.3. Rutina principal

A la rutina principal és on s'executarà el nostre codi i des d'on es cridaran a les altres rutines. Les primeres instruccions serà moure el robot al punt de repòs i iniciar i generar les TRAP necessàries que s'utilitzaran a l'aplicació:

Codi 20. *Inici rutina principal.*

```
! Rutina principal
    ! ---------------------------
    PROC main()
            !Primer moure a punt de calibracio
            comunicacionRAP := 0;
            go_repos;

            !TRAP
            IDelete maqFre;
            IDelete maqEsc;
            IDelete maqSol;
            IDelete acabar;
            CONNECT maqFre WITH TrapFresat;
            CONNECT maqEsc WITH TrapEscatat;
            CONNECT maqSol WITH TrapSoldadura;
            CONNECT acabar WITH TrapAcabar;
            ISignalDO DO10_11,1,maqFre;
            ISignalDO DO10_12,1,maqEsc;
            ISignalDO DO10_13,1,maqSol;
            ISignalDO DO10_15,1,acabar;
            SetDO DO10_11,0;
            SetDO DO10_12,0;
            SetDO DO10_13,0;
            SetDO DO10_15,0;

      ...
      {codi restant}
      ...
    ENDPROC
```

Per acabar de fer el complet tractament de les interrupcions, TRAPs, cal indicar les seves rutines. En total, tindrem fins a 4 TRAPs: una per a cada una de les tres màquines i una per a indicar que l'usuari vol sortir de l'aplicació.

Donat a que actuaran com interrupcions, el codi ha de ser el més breu possible. Per això, l'únic tractament que farem en cada interrupció serà posar a '1' les variables de petició d'atenció per a les màquines per al cas de les TRAP de màquines o posar a '1' la variable de sortida de l'aplicació.

En tots els casos, caldrà sempre reiniciar la TRAP recent activada i posar-la amb valor '0' per a que pugui ser atesa altre cop. A continuació només s'indica el codi per a una de les màquines, ja que per a les quatre interrupcions el funcionament és anàleg.

Codi 21. *Instruccions per a les interrupcions TRAP.*

```
    ! Interrupcions TRAP
    ! --------------------------
    TRAP TrapFresat
      peticioFre:=1;    !Activem variable relacionada amb la
petició
      SetDO DO10_11,0;  !Posem a '0' de nou la interrupció
    ENDTRAP
```

El següent pas, dins de la rutina principal i a continuació de les anteriors instruccions, posarem dins d'un bucle del tipus *while* totes aquelles seqüències que volem repetir mentre l'usuari no desitgi sortir de l'aplicació.

Codi 22. *Bucle de repetició.*

```
!Loop de repetició mentre programa en funcionament
      WHILE sortirRAP = 0 DO
      ...
      {codi a executar de forma continuada}
      ...
      ENDWHILE  !Final loop infinit de programa
```

Dins d'aquest bucle posarem primer les instruccions per al test de comunicació, de forma que quan s'iniciï el test i es modifiqui el valor de la variable de comunicació des de Visual Basic, l'aplicació en RAPID respon donant un nou valor a la variable. Com ja s'ha comentat en l'apartat d'explicació del programa en VB, es permet així verificar la comunicació tant de lectura com d'escriptura.

Codi 23. *Instruccions per a test de comunicació.*

```
      !Test de comunicacio
      if comunicacionRAP = 1 then          !Si es rep variable de
                                           !comunicació de VB...
```

```
            comunicacionRAP := 2;       !...contestem incrementant
                                          ! la variable.
            go_repos;          !Anar a posició de repòs
            WaitTime 5;        !Espera 5 segons
      endif
      !-----------------------------------
```

El següent pas és generar la cua de peticions d'atenció. El funcionament és idèntic per a cada una de les màquines. Mirem si està activa la variable corresponent a l'activació de la petició (posada a '1' a l'activar-se la seva interrupció pertinent).

Si es dóna aquesta primera condició, mirem si la primera posició de la cua està lliure i, en cas d'estar-ho, posarem un 1 com a valor. Si està ocupada, fem el mateix procediment amb la segona posició, i si també està col·locada posarem el valor 1 a la tercera posició.

Notis que aquest tractament de la cua el podem fer ja que com a premissa del sistema, s'ha especificat que no podem generar una nova atenció a la màquina fins que aquesta hagi estat ja descarregada. Per tant, com a pitjor cas, serà quan totes tres màquines demani'n atenció del robot.

Si s'ampliés el nombre de màquines, o bé fem un procés iteratiu i senzill com aquest però llarg, o es canvia l'estratègia de generació de la cua mitjançant vectors. En un primer moment es va intentar solucionar amb vectors, però degut a diferents problemes al escriure al codi i davant la dificultat de *debugar* el codi *in situ* amb el poc temps disponible per a fer les proves oportunes es va canviar a l'actual estratègia finalment adoptada.

De nou, com el procés és el mateix per a cada una de les màquines només s'inclou les instruccions d'una de les màquines en aquest cas la de Fresat i a la qual li correspon el nombre 1. Per a les màquines d'Escatat i de Soldadura, el valor a introduir en la posició de cua serà 2 i 3 respectivament.

Codi 24. *Generació de la cua d'atenció a màquines.*

```
      !Control i generació de la cua de peticions
      if peticioFre = 1 then
            if seguent1 = 0 then
                  seguent1 := 1;
            elseif seguent2 = 0 then
                  seguent2 := 1;
            else
                  seguent3 := 1;
```

```
            endif
            peticioFre := 0;
            estatFreRAP := 1; !en espera
      endif
```

Un cop generada la cua o actualitzada amb les noves peticions d'atenció, només quedarà decidir el tipus de moviment a indicar al robot depenent del valor de la primera posició de la cua: seguent1.

Finalment, només caldrà actualitzar les posicions de la cua de peticions d'atenció. El procediment es senzill, posar a la primera posició el valor de la segona, a la segona posició posar la de la tercera, i a la tercera posició reiniciar-la amb un 0, de forma que aquesta última posició quedi lliure.

Codi 25. *Atenció del robot en funció de la primera posició de la cua i actualització de la nova cua.*

```
!Actuacio segons cua maquina
        TEST seguent1   !Primer valor de cua es ...
            CASE 1:     !...1 -> Actuar a Fresat
                  estatFreRAP :=2;  !Descarregant Fresat
                  go_fresadora;
                  estatFreRAP :=0;  !Màquina Parada
            CASE 2:     !...2 -> Actuar a Escatat
                  estatEscRAP :=2;  !Descarregant Escatat
                  go_escatat;
                  estatEscRAP :=0;  !Màquina Parada
            CASE 3:     !...3 -> Actuar a Soldadura
                  estatSolRAP :=2;  !Descarregant Soldadura
                   go_soldadura;
                  estatSolRAP :=0;  !Màquina Parada
            DEFAULT:    !...cap dels anteriors -> Anar a repòs
                  go_repos;
        ENDTEST

        !Tractament següent
        seguent1:=seguent2;
        seguent2:=seguent3;
        seguent3:=0;
```

Si sortim del bucle d'operació, significa que l'usuari ha demanat per mitjançant la TRAP corresponent i des de Visual Basic tancar l'aplicació, i acabar amb l'execució del programa. En aquest punt, només caldrà esborrar les interrupcions per a no deixar-les penjades i que la resta de grups de pràctiques de l'assignatura les puguin fer servir sense problemes.

Codi 26. *Esborrar TRAPs generats.*

```
!Borrar DO i TRAPs
IDelete maqFre;
IDelete maqEsc;
IDelete maqSol;
IDelete acabar;
```

12. CONCLUSIONS

El projecte en conjunt resol i compleix el problema i les fites marcades com objectius a assolir. A partir de la resolució i confecció del present s'ha aconseguit implementar la comunicació client-servidor OPC entre un PC on monitoritzar un conjunt de dades i la controladora S4 del robot IRB140 del laboratori de robòtica i CIM de l'edifici TR11.

Amb la conseqüent resolució que s'aporta en aquest treball, no només permet realitzar la comunicació amb la controladora S4 sino que, coneixent el servidor d'altres unitats que incorporin connexió OPC, podem utilitzar l'aplicació presentada per a comunicar-nos amb totes les facilitats que aquest tipus de comunicació ens ofereix.

Aquests avantatges davant d'altres tipus de comunicació son, per citar-ne alguns, l'estandardització la qual permet a partir d'un sol desenvolupament la connexió amb gran quantitat d'unitats dins una mateixa xarxa o la senzillesa d'implementació que permet, un cop fet un, crear-ne tots els necessaris.

Però no únicament ens hem de fixar en el tipus de comunicació plantejat, sino que l'enregistrament de les dades en diferents taules dins d'una mateixa base de dades dóna un toc més professional i de caràcter industrial al projecte, doncs a petita escala intenta simular el funcionament del sistema d'històrics i control de dades d'una empresa real ja sigui de petita escala com multinacional.

A més, el treball no té per què acabar-se aquí. Tant pot ser reaprofitat per a possibles expansions de l'àmbit d'utilització dins el mateix laboratori, com pot ser utilitzat per altres alumnes que hagin de tractar amb el robot o, donada la seva versatilitat, en qualsevol tipus de pràctica que faci ús de la comunicació client-servidor OPC.

Per tant, es prega al lector que no es quedi només amb la conclusió de que s'ha realitzat un treball en camp pràctic en la realització d'una comunicació entre un ordinador i un altre element, en aquest cas la controladora S4 d'ABB.

Res més lluny de la realitat, sinó es demana que s'entengui com l'aprofundiment i aportació de coneixements en aquest tipus de

comunicació la qual ens permet establir connexió de forma fàcil i intuïtiva amb un notori nombre de dispositius, i de com efectuar el tractament de dades sobre una base de dades la qual també utilitza l'estàndard SQL. Així doncs, al utilitzar estandarditzacions, evitem fer desenvolupaments específics per a determinats elements.

13. BIBLIOGRAFIA

13.1. Referències bibliogràfiques

ABB Automation. 2003. "WebWare SDK Controls Reference: User's Guide". United States of America.

Barranco Marin, Antonio and Güell Brocal, Marc. 2010. Informe de pràctiques de Control i Programació de Robots. Terrassa.

13.2. Bibliografia de Consulta

Aguilera, Jóse. et al. "Implementació d'un sistema MES sobre la cèl·lula de fabricació del laboratori de CIM". Terrassa (Gener 2011).

Mathworks. "OPC Toolbox: Data Change Events and Subscriptions" http://www.mathworks.es/help/toolbox/opc/ug/f6-6349.html (accès Gener, 2012).

Microsoft. "SQL Server 2005: Date y Time (Transact-SQL)" http://msdn.microsoft.com/es-es/library/ms187819%28v=sql.90%29.aspx (accès Agost, 2011).

Microsoft. "Visual Studio: propietats FlexGrid" http://msdn.microsoft.com/en-us/library/aa228008%28v=VS.60%29.aspx (accès Agost, 2011).

Microsoft. "Visual Studio: llistat d'instruccions" http://msdn.microsoft.com/es-es/library/2f43da0y.aspx (accès Agost, 2011).

Orozco Celaya, Leonel. "Control Microsoft Hierarchical FlexGRID" http://whitesharkfishing-huatulco.com/descargas/profe231_mshflexgrid_parte1.pdf (accès Agost 2011).

Paredes, J. "Visual Basic – Guia del estudiante: teoria de objetos ADO". http://es.scribd.com/doc/55786178/VBTEORIAADO (accès Juliol 2011).

Planas, Rita M. and Hernández, Juan Carlos. "Pràctica de Control i Programació de Robots." ETSEIAT (Febrer 2010).

Rodríguez Bucarelly, Carlos Manuel et al. "Visual Basic 6.0. Orientacción a bases de datos". Distribució digital gratuita. 2008.

13.2.1. Fòrums de consulta

Compunauta. "Fòrum de conuslta sobre programació en Visual Basic 5.0". http://www.compunauta.com/forums/visualbasic/ (últim accès Juny 28, 2011).

Rdgz Pro. "Fòrum de conuslta sobre programació en Visual Basic 6.0 i bases de dades mitjançant SQL Server". http://rdgz.wordpress.com/ (últim accès Agost 23, 2011).

Grupo de Usuarios Microsoft – MUG. "Fòrum de conuslta sobre programació en Visual Basic 6.0 i bases de dades mitjançant SQL Server". http://www.mug.org.ar/default.aspx (últim accès Agost 29, 2011).

VB-mundo. "Fòrum de conuslta sobre programació en Visual Basic 6.0". http://www.foro.vb-mundo.com/ (últim accès Septembre 2, 2011).

VB6. "Tutorials and source code samples" http://www.vb6.us/ (últim accès Agost 17, 2011).

Recursos Visual Basic "Tutoriales básicos" http://www.recursosvisualbasic.com.ar/htm/ /tutoriales/tutorial-basico5.htm (últim accès Gener 09, 2012)

ANNEX A – CODI DE L'APLICACIÓ

Tot el codi utilitzat per a la realització de la pràctica es pot trobar en el CD que acompanya a aquesta entrega. El fitxers de codi es troben en format Visual Basic (*.vb) i contenen els pertinents i concrets comentaris com per entendre el funcionament de l'aplicació i per a cada un dels passos seguits en el seu desenvolupament.

ANNEX B – MANUAL WEBWARE SDK CONTROLS REFERENCE

El manual WebWare SDK Controls Reference es pot trobar també en el CD que acompanya a aquesta entrega així com a l'ordinador *CIM21* del Laboratori CIM de l'edifici TR11 en l'apartat d'ajuda del programa *WebWare SDK* o bé a la pàgina d'internet de la companyia ABB.

ANNEX C – MANUAL D'AJUDA DE L'APLICACIÓ DESENVOLUPADA

A continuació es mostra el manual d'ajuda de l'aplicació desenvolupada en el present projecte:

Benvingut a l'ajuda

Benvingut a l'ajuda de l client OPC multiservidor. Mitjançant aquest manual d'ajuda es pretén facilitar la manipulació del client. Que necessiteu saber?

Que pot fer aquest client?

Amb aquest client podreu monitoritzar les variables de qualsevol servidor OPC que permeti l'accés a dades. També podreu enregistrar tots els canvis que es produeixin en els ítems afegits a aquest client OPC dins una taula d'una base de dades i, posteriorment, visualitzar els resultats desats per pantalla.

Com afegir un servidor/grup/ítem?

El primer pas a realitzar en el nostre client és afegir tots aquells ítems que es desitgin visualitzar. Per a fer-ho tenim dues possibilitats:

1. Afegir els ítems d'un en un buscant primer el seu servidor i indicant a continuació a quin grup afegir-lo
2. Carregar un client prèviament desat. Aquesta segona opció s'explicarà detalladament en apartats posteriors d'aquest mateix capítol.

Si afegim els ítems d'un en un, l'adició tant de servidors com de grups com ítems es realitzarà mitjançant el conjunt de botons situats en la part superior esquerra de la finestra, sota el menú de l'aplicació.
Primer de tot haurem d'afegir el servidor OPC en el qual està allotjat. Per a tal fi cal prémer el botó destinat a tal fi, *Afegir servidor*. S'obrirà a continuació una nova finestra en la qual, després de carregar tots els servidors OPC disponibles, podrem seleccionar un i afegir-lo al nostre client.

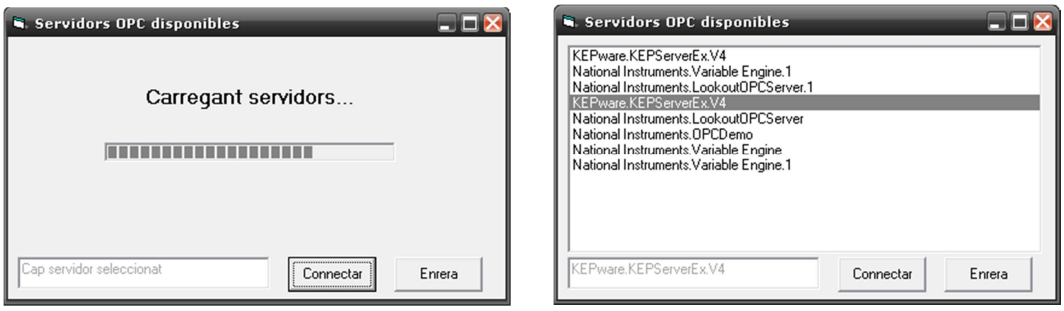


Figura: Afegint un servidor. Carregant un servidor (esquerra) i seleccionant un servidor disponible (dreta).

A continuació, cal afegir un grup al servidor. Primer hem de seleccionar el servidor que s'acaba d'afegir en el panell de l'esquerra de l'aplicació i, d'una manera semblant a l'anterior, mitjançant el botó en aquest cas *Afegir Grup* se'ns obrirà una nova finestra on haurem d'omplir els camps següents:

- Nom de grup: indica el nom del grups que afegirem
- UpdateRange: temps d'actualització del grup
- Deadband: valor mínim en que ha de variar una dada per a que pugui saltar l'esdeveniment DataChange (canvi en valor de dada)

Figura: Afegint un grup.

Un cop ja tenim el servidor i el grup on afegirem l'ítem desitjat, cal seleccionar el grup primer, i fent ús del botó *Afegir Ítem* podrem seleccionar la variable a afegir al client OPC mitjançant una nova finestra, on a través d'un panell desplegable, podrem seleccionar l'ítem o bé, si es coneix la ruta d'accés directe, indicant aquesta en el camp "ITEM ID".
Aquesta finestra utilitzada per a incorporar els ítems al client disposa d'un botó per verificar que l'ítem està disponible i la comunicació amb aquest és correcte, botó *Validar*. El resultat d'aquesta comprovació es pot visualitzar en el camp *Resposta de validació*.

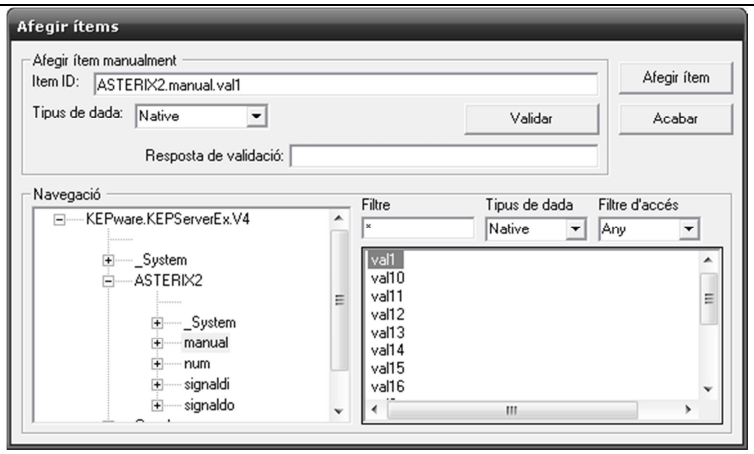
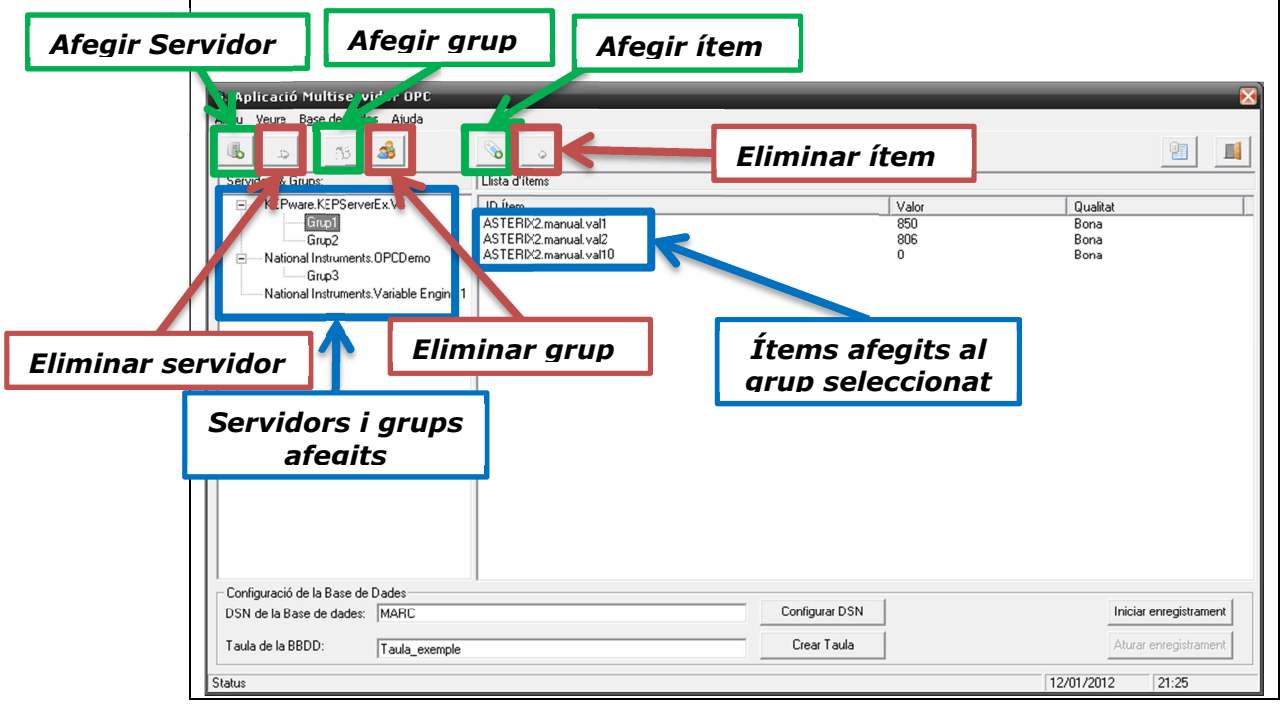


Figura: Afegint un ítem.

Anàlogament als botons per a inserir servidors, grups i ítems es disposa dels botons pertinents per a treure'ls o eliminar-los de l'aplicació client. Aquests estan situats al costat dels botons d'adició i en parelles, de forma que afegir/treure servidor OPC estan situats un al costat de l'altre, i el mateix succeeix amb grups i ítems.



Iniciar/Obrir/Desar un client

L'aplicació permet desar el client configurat de forma que en una sessió posterior no s'hagin d'afegir tots els ítems un per un sinó que permeti fer una càrrega d'un fitxer on es troba el client OPC desat.
Per a guardar el client serà suficient en clicar en l'opció *Desar client* del menú des de la pestanya *Arxiu* indicant el directori on es desitja desar l'arxiu. El format de l'arxiu és propi d'aquesta aplicació i es desarà amb l'extensió *.clo (Client OPC).
Un cop desat, podrem seguir amb el funcionament normal de l'aplicació, ja sigui continuant amb la monitorització dels ítems, afegir-ne de nous o treure'n els existents, o iniciar o obrir un nou client.
Si es desitja fer neteja de tota la informació inserida en el client i iniciar-ne'n un de nou, només cal prémer l'opció *Nou client* del menú des de la pestanya *Arxiu*. El client quedarà com si s'hagués obert per primer cop i llest per tornar a ser utilitzat.
Si el que es vol es fer una càrrega d'un client prèviament desat, caldrà accedir a l'opció *Obrir client* del menú des de la pestanya *Arxiu* indicant el directori on es troba l'arxiu del client. Aquesta acció provocarà que s'elimini qualsevol ítem afegint al client que teníem en funcionament abans de carregar el nou client OPC.

Figura: Pestanya "Arxiu" del menú.

Enregistrar dades en una taula d'una Base de Dades

Una altra opció que permet l'aplicació es la d'enregistrar tots els valors que prenguin els diferents ítems del client un una taula d'una base de dades. Per a tal fi es disposa dels camps de configuració situats en la part inferior de l'aplicació.

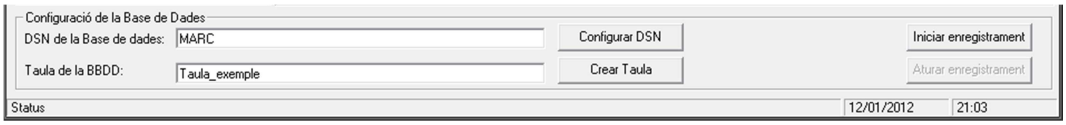


Figura: Camps de configuració per a la base de dades de la finestra principal de l'aplicació.

El primer camp s'utilitzarà per indicar la DSN a la qual tenim lincada la nostra base de dades. La DSN ha de ser configurada prèviament a fer l'enregistrament. En cas que l'usuari no hagi creat la DSN abans d'iniciar l'aplicació, aquesta incorpora un botó el qual cridarà a la funció de configuració de DSN de Windows. Aquesta funció també es pot cridar des de el menú principal dins la pestanya BBDD.

Figura: Pestanya "Base de dades" del menú.

Des d'aquí l'usuari podrà configurar la DSN per a ser utilitzada a continuació en l'aplicació client OPC. Els principals paràmetres que ha de passar l'usuari són la base de dades a la qual es farà l'accés i la qual ha d'haver estat prèviament creada, el servidor on està allotjat la base de dades i el seu registre d'accés (nom d'usuari i contrasenya si en té).
Si ja es té la DSN configurada o bé s'acaba de configurar, cal introduir el nom amb el qual hem desat la configuració en el camp pertinent, *DSN de la base de dades*. A continuació cal indicar el nom de la taula on desarem el valor de les dades.
Si no tenim creada la taula o bé es vol crear una diferent per a noves dades caldrà prémer el botó *Crear Taula*. Tant si es desitja crear la taula com indicar a quina s'accedirà per a l'enregistrament, el no d'aquesta caldrà introduir-lo en el camp *Taula de la BBDD*.
Un cop introduïts el nom de la configuració DSN així com el nom de la taula a la qual hem d'accedir, per iniciar l'enregistrament caldrà prémer el botó *Iniciar enregistrament*. Un cop iniciat, no està permès fer canvis en cap dels dos camps anteriors i, per tant, quedaran inhabilitats.
Per aturar l'enregistrament, ja sigui per que no es vol continuar desant els valors a la base de dades com si es per a canviar la configuració de la DSN o de la taula amb la qual s'interactua, caldrà prémer el botó *Aturar enregistrament*. Tot i que

s'aturi l'enregistrament, l'aplicació continuarà refrescant els valors mostrats per pantalla.
L'acció d'emmagatzematge s'executa únicament quan es detecta un canvi de valor en una variable mitjançant l'esdeveniment *DataChange*, ja que de fer-se de forma periòdica podríem omplir la taula d'informació redundant o bé perdre informació.
Els camps que s'emmagatzemen a la taula són el dia i hora en que ha canviat el valor de la dada, a quin ítem pertany la dada, el seu valor i la qualitat de la comunicació en el moment del canvi de valor.

Consultar una taula d'una Base de Dades

Per accedir-hi cal prémer el botó *Visualitzar taula de BBDD* del formulari principal. En aquesta nova finestra podrem cridar i visualitzar una taula on haguem enregistrat anteriorment les dades del client OPC. El primer que s'ha de fer es indicar on ha d'anar a buscar i quina és la taula de la base de dades.

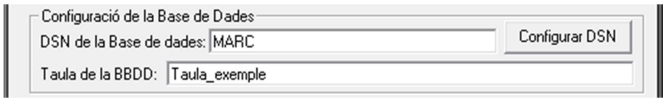


Figura: Camps de configuració per a la base de dades de la finestra de visualització d'una taula d'una BBDD.

Com en el cas per a configurar a quina taula desar la informació, tenim un espai reservat a complimentar els paràmetres per indicar la base de dades on està la taula desitjada mitjançant la DSN de connexió i el nom d'aquesta taula. Com passava en la finestra principal de l'aplicació, es disposa d'un botó per a cridar a la funció de configuració de DSN de Windows per si es necessita fer algun tipus d'ajust.
Un cop hem completat aquests dos camps ja només resta connectar-nos amb la taula i visualitzar-la en la graella situada a la meitat inferior de la finestra. Per a connectar-nos amb la taula serà suficient en prémer el botó *Connectar*.
La informació que es mostrarà en aquesta graella són el dia i hora en que ha canviat el valor de la dada, de quin ítem es tracta, el seu valor i la qualitat de la comunicació en el moment del canvi de valor en aquest ordre.

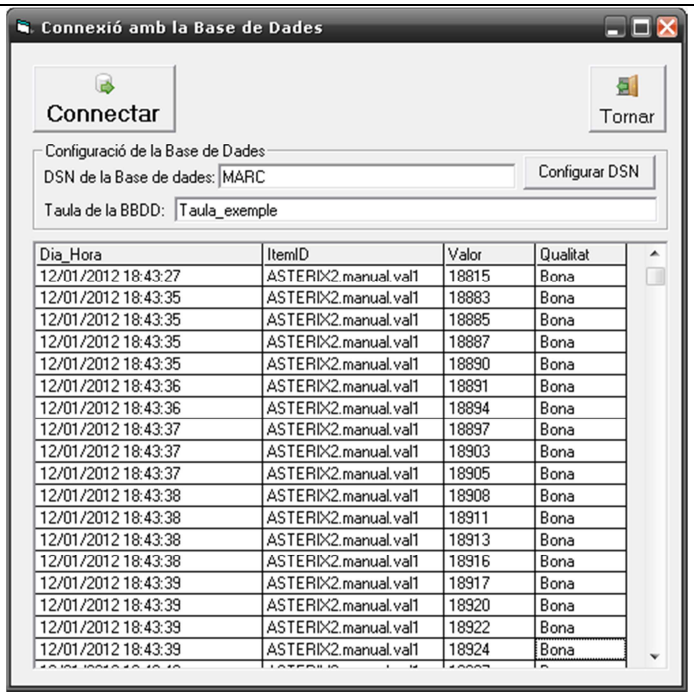


Figura: Finestra de visualització d'una taula d'una BBDD amb la graella d'informació omplerta.

Un cop hem acabat de revisar i/o mirar la informació proporcionada per la taula podrem tornar a la finestra principal mitjançant el botó Tornar.

Autor

L'autor d'aquesta aplicació és en Marc Güell Brocal, alumne de la ETSEIAT en la titulació Enginyeria en Automàtica i Electrònica Industrial, qui ha desenvolupat aquesta aplicació com a projecte final de carrera.

Per a qualsevol consulta de l'aplicació, com d'altres configuracions del programari utilitzat així com del desenvolupament del projecte en si es pot consultar el projecte amb títol *Disseny d'una estructura de recol·lecció de dades d'un robot ABB140.*

Titulació:

Enginyeria en Automàtica i Electrónica Industrial

Alumne:
Marc Güell Brocal

Títol PFC:
Disseny d'una estructura de recol·lecció de dades d'un robot ABB140

Director del PFC:
Juan Carlos Hernández Palacín

Convocatòria de lliurament del PFC
Gener 2012

Contingut d'aquest volum: **-PRESSUPOST ECONÒMIC-**

-PLEC DE CONDICIONS-

INDEX GENERAL

Escola Tècnica Superior d'Enginyeries
Industrial i Aeronàutica de Terrassa
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Disseny d'una estructura de recol·lecció de dades d'un robot ABB140

VOLUM II – PRESSUPOST ECONÒMIC

Alumne: **Marc Güell Brocal**

Director de Projecte: **Juan Carlos Hernández Palacín**

Gener 2012

TAULA DE CONTINGUTS

TAULA DE TAULES

1. INTRODUCCIÓ

Per a aquest capítol, entenem que a l'hora de pressupostar un projecte com aquest, el client ja disposa d'un seguit d'eines i material prèviament. Sent així, es fixa per a fer el pressupost de desenvolupament del projecte únicament el cost dels enginyers per a desenvolupar l'aplicació, i cost derivat tant del software necessari per al desenvolupament de les aplicacions com al hardware necessari per implantar la solució proposada al client.

Per tant, no constarà en l'estudi econòmic el cost del conjunt robòtic (robot ABB IRB140 i controladora ABB S4), ni de l'emplaçament del mateix, així com del software previ i necessari per a treballar satisfactòriament amb el robot.

A continuació es mostra una llista detallada del material amb el qual no es comptarà per fer el pressupost:

- Robot ABB IRB140,
- Controladora ABB S4,
- Programari WebWare SDK amb llicència,
- Material necessari per a les comunicacions del robot abans d'iniciar-se el projecte (com per exemple connexió LAN),
- Habitació o planta (zona) de treball.

Si que serà requisit d'entrar a pressupost, com ja s'ha comentat, els costos d'enginyeria, software i hardware addicional.

2. DESPESES EN RECURSOS HUMANS

En aquesta part es realitza l'estudi del cost que equivaldria pagar a un enginyer per al temps dedicat a l'estudi previ, al disseny tant de hardware com de software, al muntatge i altres operacions. En resum, el cost dels recursos humans per al desenvolupament del projecte.

Notis que, per a calcular el preu/hora de l'enginyer tenim dues possibles opcions:

1. Enginyer junior: en aquest cas, el preu/hora és baix, entre 20 i 30 €/hora. Equivaldria a ser un enginyer que s'inicia en el món laboral o que hi fa poc que està immers, el qual necessitaria una gran quantitat d'hores per al desenvolupament de totes les parts deguda a la seva poc experiència laboral.

2. Enginyer sènior: en aquest cas, el preu/hora és més alt, entre 50 i 70 €/hora. No obstant, el nombre d'hores que utilitzarà serà menor que en el cas anterior, ja que disposa d'experiència, un cert "know how", i el seu desenvolupament seria més ràpid.

Generalment un projecte sol portar-lo un enginyer sènior (o més) complimentant-se amb altres enginyers com ell i d'enginyers júnior. Per ajustar-nos el més possible a la realitat, i tenint en compte el treball fet per el tutor, entendrem que l'enginyer sènior serà el tutor efectuant el paper de cap de projecte i l'autor actuarà com un enginyer junior el qual desenvoluparà la major part del projecte fent les consultes apropiades amb el sènior.

Tot i que no s'ha portat un control d'hores exhaustiu, si és cert que el temps de desenvolupament del projecte s'aproxima al barem estipulat per la universitat ETSEIAT en la planificació i normativa de Projectes Finals de Carrera.

Per tant, el temps utilitzat per a la realització del projecte final de carrera està fixat a 225 hores per a l'alumne, l'equivalent als 9 crèdits de duració del projecte a raó de 25 hores per crèdit. Per a estimar un

nombre d'hores per al tutor, s'estableix un 10% d'hores de dedicació del total de l'alumne, el que suposa 22 hores per part del tutor.

Per al preu dels enginyers es farà una mitja aproximada del cost de cada un dels diferents enginyers, posant un cost hora de 22 € per a l'enginyer júnior i 65€ per al sènior.

A continuació es mostra una taula amb les hores dedicades per l'enginyer sènior (tutor) i el júnior (autor del projecte):

Taula 1. *Cost en recursos humans.*

Tipus Enginyer	Preu/Hora	Hores dedicades	Cost total
Enginyer sènior	22 €	225	4950 €
Enginyer júnior	65 €	22	1430 €
		TOTAL	6380 €

3. DESPESES EN SOFTWARE

Aquí es desglossa el cost originat per el programari utilitzat per al desenvolupament del projecte, tant de la part pràctica per a realitzar les aplicacions com el programari necessari per a la redacció i confecció de la memòria.

Notis que només es necessita un programa de cada tipus atès que aquests s'instal·laran en un únic ordinador. Aquest ordinador, serà el que teòricament deixaríem en una empresa per a poder executar les aplicacions desenvolupades.

No obstant, com no serà el mateix PC on treballarà l'enginyer júnior, es sumarà un 20% a la quantitat del material utilitzat en concepte d'adquisició del mateix programari però que serà amortitzat també amb altres projectes. En total, per tant, podem suposar que la quantitat de recursos per a cada programari serà de 1,20 unitats.

Taula 2. *Cost en material software.*

Material	Quantitat	Preu unitat	Cost total
Microsoft Office 2010, Hogar y Pequeña Empresa	1,20	199,99 €	239,99 €
Microsoft SQL Server 2005, Workgroup Edition (1 servidor – 5 clients)	1,20	438,80 €	526,56 €
Microsoft Visual Basic 6.0	1,20	156,73 €	188,08 €
Microsoft Windows XP Professional SP2	1,20	208,96 €	250,75 €
		TOTAL	1205,38 €

4. DESPESES EN HARDWARE

En aquest apartat fem constar tot el cost produït per material hardware necessari per implementar les aplicacions, que aquestes funcionen, coordinin i comuniquin correctament amb el material ja disponible.

Per al cas dels ordinadors, com ja s'ha introduït en l'apartat de costos software, n'imputarem un que conté les aplicacions i programari necessari instal·lat i el qual s'espera vendre al client; i un segon ordinador, el qual serà personal de l'enginyer júnior amb el qual treballarà des de casa.

Per al PC per a client, se'n triarà un de gamma alta ja que degut al software utilitzat necessita d'un processador d'alta velocitat i amb una quantitat de memòria RAM elevada per a poder treballar amb agilitat i que el client no vegi provocar parades en les aplicacions ni en el propi ordinador.

La següent taula mostra de forma desglossada el cost total en material hardware:

Taula 3. *Cost en material hardware.*

Material	**Quantitat**	**Preu unitat**	**Cost total**
Monitor Asus VH197D LED de 18,5''	1	99 €	99 €
PC genèric per a client	1	700 €	700 €
		TOTAL	799,00 €

5. COST TOTAL PRESSUPOSTAT

Atenent als diferents costos que s'han anat pressupostant, a continuació es mostra mitjançant una taula el cost total del desenvolupament d'aquest projecte.

Taula 4. *Cost total del projecte.*

Tipus de cost	**Cost**
Despeses en recursos humans	6380,00 €
Despeses en software	1205,38 €
Despeses en hardware	799,00 €
	8384,38 €

Disseny d'una estructura de recol·lecció de dades d'un robot ABB140

VOLUM II – PLEC DE CONDICIONS

Alumne: **Marc Güell Brocal**

Director de Projecte: **Juan Carlos Hernández Palacín**

Gener 2012

TAULA DE CONTINGUTS

1. DISPOSICIONS GENERALS

1.1. Objecte del plec de condicions

El present Plec de Condicions constitueix el conjunt d'instruccions, normes i especificacions que juntament amb l'especificat en la memòria descriptiva del projecte, defineixen tots els requisits tècnics per a fer ús de l'aplicació.

L'objectiu del mateix és definir les obligacions del client durant la realització de la màquina, complint totalment amb els articles del present plec obligant-se a complir les ordres formulades per l'enginyer, des de l'inici de la màquina, fins la recepció final.

1.2. Documents que defineixen el client OPC

L'aplicació client OPC és definida pel Plec de Condicions i pels documents constitutius del projecte: Memòria, Pressupost i Annexes.

Són documents contractuals els documents Plec de Condicions i Pressupost, que s'inclouen en el present Projecte. Qualsevol canvi i/o modificació en el plantejament que impliqui un canvi substancial en l'aplicació i/o el seu funcionament haurà de posar-se en coneixement a l'autor de la mateixa.

2. RESPONSABILITATS

2.1. Responsabilitats de l'autor.

Les responsabilitats de l'autor sobre l'aplicació es limiten a la correcta instal·lació en el lloc de treball final així com comprovar el seu correcte funcionament en el moment de la posta a punt en el lloc definitiu. Si serà de responsabilitat de l'autor respecte del lloc de treball subministrar l'ordinador on correrà l'aplicació desenvolupada tal i com està pressupostat en el capítol econòmic d'aquest projecte.

Serà responsabilitat de l'autor proveir de tot el programari necessari i pressupostat tal i com es mostra en el document de pressupost econòmic.

L'autor no serà responsable de qualsevol canvi i/o manipulació tant sobre l'aplicació com en el sistema operatiu o qualsevol programari necessari per al correcte funcionament de l'aplicació. Tampoc es fa responsable de l'ús inadequat d'aquesta.

L'autor s'eximeix de qualsevol mal funcionament de l'aplicació en l'apartat de comunicacions amb el conjunt robòtic i possibles perjudicis derivats tant en l'aplicació com del propi conjunt robòtic si aquest no està format per els elements utilitzats durant l'elaboració del present projecte: una controladora S4 i un robot IRB140.

2.2. Responsabilitats de l'usuari.

Queda com a responsabilitat de l'usuari fer un correcte funcionament i ús de l'aplicació així com de tot el programari que acompanya l'aplicació i que és facilitat en el moment de l'adquisició d'aquesta.

També serà responsabilitat de l'usuari facilitar les instal·lacions i el conjunt robòtic format per la controladora S4 més el robot IRB140 i de totes les seves respectives llicències actualitzades i funcionals, així com la xarxa de comunicació que sigui necessària.