UNIVERSIDAD TECNOLÓGICA EQUINOCCIAL
Campus Santo Domingo

FACULTAD DE CIENCIAS DE LA INGENIERÍA
CARRERA DE INGENIERIA ELECTROMECANICA Y AUTOMATIZACIÓN

Tesis de grado previa a la obtención de título de:
INGENIERO ELECTROMECANICO, MENCIÓN EN AUTOMATIZACIÓN INDUSTRIAL

SUPERVISIÓN Y CONTROL DE UN PROCESO INDUSTRIAL DIDÁCTICO DESDE UN DISPOSITIVO MÓVIL.

Estudiante:
CÉSAR FERNANDO ZAMBRANO MEDRANDA

Director de tesis
ING. VÍCTOR ARMIJOS

Santo Domingo – Ecuador
Enero, 2012

SUPERVISIÓN Y CONTROL DE UN PROCESO INDUSTRIAL DIDÁCTICO DESDE UN DISPOSITIVO MÓVIL.

Ing. Víctor Armijos _____
DIRECTOR DE TESIS

APROBADO

Ing. Marcelo Estrella _____
PRESIDENTE DEL TRIBUNAL

Ing. Jorge Terán _____
MIEMBRO DEL TRIBUNAL

Ing. Christian Macías _____
MIEMBRO DEL TRIBUNAL

Santo Domingo…....de……del 2012.

Institución: UNIVERSIDAD TECNOLÓGICA EQUINOCCIAL
Campus Santo Domingo

Tema: SUPERVISIÓN Y CONTROL DE UN PROCESO INDUSTRIAL DIDÁCTICO DESDE UN DISPOSITIVO MÓVIL.

Director de Tesis: Ing. Víctor Armijos

Fecha: Enero, 2012

Del contenido del presente trabajo se responsabiliza el autor

_____

César Fernando Zambrano Medranda
CI.0802513861

UNIVERSIDAD TECNOLÓGICA EQUINOCCIAL
Campus Santo Domingo

INFORME DEL DIRECTOR DE TESIS

Santo Domingo……de……..del 2012.

Ing. Marcelo Estrella
COORDINADOR INGENIERÍA ELECTROMECÁNICA
UNIVERSIDAD TECNOLÓGICA EQUINOCCIAL
CAMPUS SANTO DOMINGO

Por medio del presente tengo a bien informarle que la Tesis de Grado propuesta por el Sr. **César Fernando Zambrano Medranda**, previa a la obtención del Título de Ingeniería Electromecánica, Mención en Automatización Industrial, cuyo tema es **"SUPERVISIÓN Y CONTROL DE UN PROCESO INDUSTRIAL DIDÁCTICO DESDE UN DISPOSITIVO MÓVIL"**. Ha sido desarrollada en su totalidad bajo mi supervisión y autorizo la respectiva presentación.

Particular que informo para fines pertinentes.

Atentamente,

Ing. Víctor Armijos
DIRECTOR DE TESIS

DEDICATORIA

Dedico este proyecto a toda mi familia, quienes me han sabido dar todo su amor, cariño y todo su apoyo para salir siempre adelante encada una de mis metas, llenando mi vida de sabios consejos que me han permitido ser una persona de bien para la sociedad.

De manera especial a mi Padre, Madre, Hermanos y Abuelos quienes han sido parte de mi vida y que con su apoyo incondicional y el esfuerzo que han realizado he logrado culminar esta importante etapa de mi vida.

Finalmente a cada una de las personas que nunca han dejado de creer en mi y que ansiaban con mucho orgullo que termine mi carrera con éxito.

César Zambrano Medranda

AGRADECIMIENTO

En primer lugar quiero agradecer a mis padres que con su ejemplo de personas trabajadoras y de buen corazón, han sembrado en mí los valores éticos y morales que son el pilar fundamental para la formación integral de una persona.

Un agradecimiento cordial al Ing. Víctor Armijos director de mi tesis, quien con sus consejos y opiniones oportunas he podido terminar este proyecto.

A todos y a cada uno de los Docentes de la Universidad Tecnológica Equinoccial, en especial a los de la Facultad de Ingeniería Electromecánica quienes me impartieron sus conocimientos en las aulas para mi formación profesional.

Además quiero agradecer a cada uno de los miembros de mi familia, a mis compañeros y amigos que de una u otra forma siempre han estado a mi lado dándome todo su apoyo cuando lo necesitaba.

César Zambrano Medranda

ÍNDICE GENERAL.

Pág.

Portada…… i

Hoja de sustentación y aprobación de los integrantes del tribunal ……... ii

Hoja de responsabilidad el autor ……..……....… iii

Informe del director de tesis……..…… iv

Dedicatoria…… v

Agradecimientos…… vi

Índice general……..... vii

Índice de figuras…… xvi

Índice de tablas…… xx

Resumen Ejecutivo……..…..... xxi

Executive Summary..... xxiii

CAPÍTULO 1

CONTENIDO

1. Introducción…….……..... 1

1.1 Antecedentes…….…... 1

1.1.1 Antecedentes históricos…….…... 1

1.1.2 Antecedentes científicos…….……... 2

1.1.3 Antecedentes prácticos…….. 2

1.1.4 Importancia del estudio……..…….……..... 3

1.1.5 Situación actual del tema de investigación……..…… 3

1.2 Limitaciones del estudio…….…..... 4

1.3 Alcance del trabajo……..... 4

1.4 Objetivo del estudio…….…..... 4

1.4.1Objetivo general del estudio…….……...…..…... 4

1.4.2 Objetivos específicos……..…..…..... 5

1.5. Justificación……...……..… 5

1.6 Hipótesis……...……. 6

1.7 Metodología……...…….....…… 6

1.7.1 Unidad de análisis……...….. 6

1.7.2 Aspectos metodológicos generales del estudio……...…...…… 7

1.7.2.1Descriptiva…….……...……...…..… 7

1.7.2.2 Exploratoria…….……...……...…… 7

1.7.3 Método de estudio……...….... 7

1.7.3.1Deductivo…...……...……...…… 8

1.7.3.2 Analítico.……...…..……...……...…… 8

1.7.3.3 Sintético……...……...…….…...… 8

CAPÍTULO II

FUNAMENTOS TEORICOS

2.1 Estudio de los sistemas SCADA ……..... 9

2.1.1 Criterios de diseño del prototipo SCADA …… 9

2.1.2 Características del SCADA……..… 10

2.1.3 Esquema de un sistema SCADA……...… 10

2.1.4 Funciones de un sistema SCADA…… 12

2.1.5 Módulos de un sistema SCADA……. 13

2.2 Estudio de las interfaces de comunicación……..… 14

2.3 Intouch ……..……...… 16

2.3.1 Características…….…… 16

2.3.2 Programación……..…. 17

2.3.3 Comunicaciones…….…...… 18

2.4 Base De Datos……..……...…… 18

2.4.1 DBMS (DataBase Management System)…….…… 19

2.4.2 Microsoft SQL sever……... 19

2.4.2.1 Elementos de SQL Server…… 20

2.4.2.2 Seguridades…….. 21

2.5 INDUTRIALSQL SERVER……...… 22

2.5.1 InSQL……...……...…… 22

2.5.2 Consultas y Procedimientos Almacenados……..……...……. 26

2.6 Active Factory…… 28

2.6.1. Definición……..…….……..…..… 28

2.6.2. Aplicaciones de escritorio……...…… 29

2.6.3. Complementos de Microsoft Office……..…… 29

2.6.4. Controles…… 30

2.6.5. ActiveFactory Trend…… 30

2.6.6. ActiveFactory Query…… 30

2.6.7. ActiveFactory Workbook…… 31

2.6.8. ActiveFactory Report…… 31

2.7 Aplicaciones WEB…… 32

2.7.1 Arquitectura de las aplicacioness Web…… 33

2.8 PLC…… 33

2.8.1 PLCs…… 33

2.8.2 Estructura de un PLC…… 34

2.8.2.1 Fuente de alimentación…… 35

2.8.2.2 CPU (unidad central de procesamiento)…… 35

2.8.2.3 Módulos De Entradas y Salidas…… 36

2.8.3 Funcionamiento…… 36

2.8.4 Programación…… 37

2.8.5 Comunicación…… 37

2.9. Twido…… 38

2.9.1 Instalación del software…… 38

2.9.2 Registro de TwidoSuite…… 39

2.9.3 Inicio de programa…… 39

2.9.4 Configuración básica del Hardware Twido…… 41

2.10 Java……..…… 41

2.10.1 Introducción.…….. 41

2.10.2 J2ME (Java 2 Platform, Micro Edition)……..…… 43

2.10.3 Máquinas Virtuales J2ME…… 44

2.10.4 KVM……..…… 45

2.10.5 CVM……..…… 46

2.10.6 Configuraciones……..… 47

2.10.6.1 Configuración de dispositivos con conexión, CDC…...……...….. 48

2.10.6.2 Configuración de dispositivos limitados con conexión…..……..... 48

2.10.7 Perfiles……..… 50

2.10.8 J2ME y las comunicaciones……..……..... 51

CAPÍTULO III

METODOLOGÍA

3.1 Introducción.……..... 53

3.2 Nivel de la investigación…….....… 53

3.3 Tipo de investigación……..……..…. 54

3.3.1 Por el grado de abstracción ……..……..…..…… 54

3.3.2 Por la naturaleza de los datos……..……..…… 54

3.3.3 Por la dimensión cronológica……..…..… 54

3.3.4 Por el tipo de fuentes……..……..…..… 55

3.4 Diseño de investigación ……..……..… 55

3.4.2. Tipos de diseño……..….. 55

3.5. Técnicas de investigación…….…..…… 57

3.5.1. La observación…….…… 57

3.5.2 La revisión bibliográfica…….…… 58

3.6. Fuentes de datos……..……. 58

3.6.1. Fuentes primarias……..…… 58

3.6.2. Fuentes secundarias……...…… 58

CAPÍTULO IV

DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE SUPERVISIÓN Y CONTROL DE UN PROCESO INDUSTRIAL DIDÁCTICO DESDE UN DISPOSITIVO MÓVIL.

4.1 Análisis De Los Elementos Necesarios Para La Implementación Del Sistema……..……..… 60

4.1.1 Usuarios……..……..…... 61

4.1.1.1 Usuarios de la red Administrativa…….……..…... 61

4.1.1.2 Usuarios de la red Industrial……..……..…... 61

4.1.2 HARDWARE……...…..…… 61

4.1.2.1 Estación de trabajo…..……...…..…. 62

4.1.2.2 PLCs……...…. 62

4.1.2.3 Red Industrial…… 62

4.1.2.4 Data Base Server……...…….…..…..... 62

4.1.2.5 Web Server…….…… 63

4.1.3 SOFTWARE……..……..…… 63

4.1.3.1 Programación de PLCs (TWIDO)…….……..… 63

4.1.3.2 Base de Dato Industrial……..…. 63

4.1.3.3 SQL Server……..….. 63

4.1.3.4 Windows Server…… 64

4.2 Creación del sistema SCADA para el proceso a simular que es un

proceso de extracción de jugo de manzana…..... 64

4.2.1 Esquema del sistema SCADA que se va a implementar……..…..…. 65

4.2.2 Unidad de Control (PLC)…….…… 66

4.2.3 Creación de la pantalla HMI……..…...… 66

4.3 Selección de las variables que van a ser almacenadas en la base dez datos…..…. 71

4.3.1 Listado del direccionamiento de todas las variables……..……..… 71

4.4 Creación de la base de datos utilizando el software IndustrialSQL……...… 72

4.4.1. Introducción……..……..… 72

4.4.2. Instalación Microsoft SQL Server……..……..….. 73

4.4.2.1 Configuración de Microsoft SQL……..……..…. 78

4.4.3. Instalación de Industrial SQL……..……..….. 80

4.4.4 Configuración de la base de datos……..…….……..…… 86

4.4.5. Arrancar la base de datos……..……..…… 89

4.4.6. Importación de variables…….... 91

4.5 JavaScript ……... 100

4.5.1 Glosario básico……..… 101

4.5.2 Sintaxis…….……...1 101

4.5.3 Programación básica……..….. 103

4.5.3.1 Variables…….…... 104

4.5.3.2 Tipos de variables……..…… 106

4.5.3.3 .Operadores……..……..…..... 110

4.5.4 Estructuras de control de flujo……..…….….. 116

4.5.4.1 Estructura if…….……..…… 116

4.5.4.2 Estructura if...else……..…… 117

4.5.4.3 Estructura for……...……..…. 118

4.5.4.4 Estructura for...in…….…...……..…. 119

4.5.5 Funciones y propiedades básicas de JavaScript……..……..... 119

4.5.5.1 Funciones útiles para cadenas de texto…...…… 119

4.5.5.2 Funciones útiles para arrays……..…… 122

4.5.5.3 Funciones útiles para números……..… 125

4.5.6 Programación avanzada.……..……..…..….. 126

4.5.6.1 Funciones…….……. 127

4.5.6.2 Argumentos y valores de retorno…….……..…….….. 129

4.5.6.3 Ámbito de las variables……..…….…….....1 130

4.5.6.4 Sentencias break y continue…….……. 132

4.5.7 Otras estructuras de control…… 134

4.5.7.1 Estructura while……..…. 135

4.5.7.2 Estructura do...while……. 136

4.5.7.3 Estructura switch…….. 136

4.6 Netsbeans……...…. 138

4.6.1 Creación de un Proyecto……..……..... 146

4.7 Características del dispositivo móvil…….……. 168

4.8 Pruebas de funcionamiento……..…. 169

CAPÍTULO 5

CONCLUSIONES Y RECOMENDACIONES

5.1Conclusiones…….....……....., 176

5.2 Recomendaciones…….……...……....… 177

BIBLIOGRAFÍA.……..…… 178

ANEXOS……..... 180

ÍNDICE DE FIGURAS.

Fig. 2.1 Esquema de un SCADA…… 10

Fig. 2.2 Componentes de una HMI……..... 15

Fig. 2.3 Pantalla de Status del DBMS InSQL Server……..... 24

Fig. 2.4 Pantalla de Configuración de los Parámetros del InSQL Server …… 24

Fig. 2.5 Configuración para la comunicación con los I/O Server…… 25

Fig. 2.6 Configuración de los Tags…….……..... 25

Fig. 2.7 Almacenamiento de Datos en InSQL……..... 26

Fig. 2.8 Consola de SQL Server 2005…….…….. 27

Fig. 2.9 Búsqueda de variables…….…… 27

Fig.2.10 Resultados del Procedimiento Almacenado……..... 28

Fig.2.11 Capas de la Aplicación Web……..... 32

Fig. 2.12 Arquitectura de la Aplicación Web ……..... 33

Fig.2.13 Elementos del PLC ……. 34

Fig.2.14 Proceso del PLC…….. 36

Fig. 2.15 Entorno de ejecución…….. 44

Fig. 3.1 Diseño de investigación…… 56

Fig. 4.1 Esquema de SCADA…… 65

Fig. 4.2 Nueva aplicación de Intouch……..... 67

Fig. 4.3 Ventana del proceso…… 68

Fig. 4.4 Configuración del Access Na..…… 69

Fig. 4.5 Creación de variables…….…… 70

Fig.4.6 Selección Log Data…… 70

Fig. 4.7 Paso 1 y 2 de Instalación de Microsoft SQL Server……...…. 73

Fig. 4.8 Paso 3 y 4 de Instalación de Microsoft SQL Server…….. 74

Fig. 4.9 Paso 5 y 6 de Instalación de Microsoft SQL Server…… 75

Fig.4.10 Paso 7 y 8 de Instalación de Microsoft SQL Server……..…… 75

Fig. 4.11 Paso 9 y 10 de Instalación de Microsoft SQL Server……..…… 76

Fig. 4.12 Paso 11 y 12 de Instalación de Microsoft SQL Server……..…… 76

Fig. 4.13 Paso 13 y 14 de Instalación de Microsoft SQL Server……..… 77

Fig.4.14 SQL Server Service Manager……..... 77

Fig. 4.15 Consola de Microsoft SQL……...…… 78

Fig. 4.16 Crear nueva vista……..……..…. 78

Fig. 4.17 Propiedades de vista……..…… 79

Fig. 4.18 Nueva vista……..……..... 79

Fig.4.19 Paso 1 Instalación Industrial SQL……..…. 81

Fig. 4.20 Paso 2 Instalación Industrial SQL……... 82

Fig. 4.21 Paso 3 Instalación Industrial SQL……..…. 82

Fig. 4.22 Paso 4 Instalación Industrial SQL…… 83

Fig. 4.23 Paso 5 Instalación Industrial SQL……... 83

Fig.4.24 Paso 6 Instalación Industrial SQL……...… 84

Fig. 4.25 Paso 7 Instalación Industrial SQL……...… 84

Fig. 4.26 Paso 8 Instalación Industrial SQL…….……..… 85

Fig. 4.27 Paso 9 Instalación Industrial SQL……...… 85

Fig. 4.28 Paso 10 Instalación Industrial SQL……..... 86

Fig. 4.29 Consola administración de consola…….…… 87

Fig. 4.30 Arrancar el sistema…….…… 89

Fig. 4.31 Seguridad del sistema……..… 90

Fig. 4.32 Arrancando sistema…….…… 91

Fig. 4.33 Paso 1 Importar variables…… 92

Fig. 4.34 Paso 2 Importar variables……..... 92

Fig. 4.35 Paso 3 Importar variables……... 93

Fig. 4.36 Paso 4 Importar variables…… 93

Fig. 4.37 Paso 5 Importar variables……...…. 94

Fig. 4.38 Paso 6 Importar variables……... 94

Fig. 4.39 Paso 7 Importar variables……... 95

Fig.4.40 Paso 8 Importar variables……..... 96

Fig. 4.41 Paso 9 Importar variables…… 96

Fig. 4.42 Paso 10 Importar variables……..…… 97

Fig. 4.43 Paso 1confirmar cambios realizados……..... 97

Fig.4.44 Paso 2 Confirmar pasos realizados……..… 98

Fig.4.45 Paso 3 Confirmar cambios realizados…….. 98

Fig. 4.46 Visualizar variables importadas…….…. 99

Fig. 4.47 Propiedades de una variables……..…….…..... 100

Fig. 4.48 Paso 1 descargar Netbbeans…….…..... 139

Fig. 4.49 Paso 2 descargar Netbeans…….……...…... 140

Fig. 4.50 Paso 3 descargar Netbeans…….. 140

Fig. 4.51 Paso 1 instalar Netbeans……..……...…. 141

Fig. 4.52 Paso 2 instalar Netbeans……..... 141

Fig. 4.53 Paso 3 instalar Netbeans…….…… 142

Fig. 4.54 Paso 4 instalar Netbeans…….……...… 142

Fig. 4.55 Paso 5 instalar Netbeans…….……...… 143

Fig. 4.56 Paso 6 instalar Netbenas…….……. 143

Fig. 4.57 Paso 7 instalar Netbeans…….……..... 144

Fig. 4.58 Paso 8 instalar Netbeans…….…….…... 144

Fig. 4.59 Paso 9 instalar Netbeans…….……. 145

Fig. 4.60 Paso 10 instalar Netbeans…….…… 145

Fig. 4.61 Paso 1 Crear proyecto…….……. 146

Fig.4.62 Paso 2 Crear proyecto…….…….. 147

Fig. 4.63 Paso 3 Crear proyecto…… 147

Fig. 4.64 Paso 4 Crear proyecto…….....…. 148

Fig. 4.65 Paso 5 Crear proyecto…… 148

Figura 4.66 Esquema final de proyecto…… 149

Figura 4.67: Pantalla principal de aplicación…… 158

Figura 4.68**:** Primera etapa de proceso…… 160

Figura 4.69**:** Segunda etapa del proceso…… 161

Figura 4.70**:** Tercera etapa del proceso…… 162

Figura 4.71: Cuarta etapa del proceso…… 163

Figura 4.72: Acceso directo Toncat…… 170

Figura 4.73**:** Acceso directo página de Web Server…… 170

Figura 4.74: Arranque de base de datos virtual…… 171

Figura 4.75: Web Server conectado…… 171

Figura 4.76: Web Server desconectado …… 172

Figura 4.77: Pantalla de inicio en el celular…… 172

Figura 4.78: Aplicación abierta…… 173

Figura 4.79: Muestra 1 de comunicación…… 174

Figura 4.80: Muestra 2 de comunicación…… 174

Figura 4.81: Muestra 3 de comunicación …… 175

Figura 4.82: Muestra 4 de comunicación …… 175

ÍNDICE DE TABLAS.

Tabla. 4.1 Variables……71

Tabla 2: Conversión……108

RESUMEN EJECUTIVO

Ya que dentro de la formación académica en la Facultad de Electromecánica se presentan las puertas abiertas para la innovación en distintos campos técnicos y tecnológicos y optando por contemplar la iniciativa de fusionar la parte de Control Industrial con la investigación en Informática, el desarrollo de este proyecto lleva a cumplir con unos de los objetivos presentados a lo largo de la carrera: el desarrollo de tecnología de la mano de la investigación.

A continuación se presentará una breve descripción de cada uno de los capítulos que constituyen el presente trabajo de tesis.

En el primer capítulo y luego de un análisis riguroso se presentan los criterios, las ventajas, los motivos, los alcances y limitaciones, objetivos y justificativos que permiten cumplir rigurosamente con las expectativas enunciadas en el plan de tesis.

En el segundo capítulo se hace referencia a los fundamentos teóricos de sistemas SCADA, lenguaje SQL, bases de datos, y se hace énfasis en la información teórica del software IndustrialSQL y ActiveFactory que son programas especializados en la creación y manejo de bases de datos industriales, además se describen los fundamento teóricos de aplicaciones Web, del Controlador Lógico PLC, de lenguaje JAVA y del programa TWIDO.

En el tercer capítulo se hace constar la metodología que se va a utilizar para la realización de la tesis, dicha metodología consta del diseño de la investigación, la

selección del tipo de investigación y la especificación de las técnicas de investigación que se han utilizado en el estudio.

En el cuarto capítulo se ha documentado todo el proceso de diseño e implementación de un sistema SCADA para la simulación de un proceso de extracción de jugo, además en el sistema se ha implementado una base de datos utilizando el software IndustrialSQL para su creación y configuración, y se ha utilizado el programa ActiveFactory para generar tablas con valores actuales, también se detallara el proceso para crear una página Web, la programación y configuración del programas NetBeans.

En el quinto capítulo se muestran las conclusiones y recomendaciones extraídas luego de haber finalizado con éxito el presente trabajo de tesis.

EXECUTIVE SUMMARY

Since within the academic training at the School of Electromechanical doors are open to innovation in different technical and technological fields and opting for the initiative to merge contemplate the Industrial Control Computer research, the development of this project has to fulfill some of the objectives presented throughout the race: the development of technology of the hand of the investigation.

The following is a brief description of each of the chapters that constitute this thesis.

In the first chapter and then present a rigorous analysis criteria, benefits, motives, scope and limitations, and supporting objectives that enable rigorously fulfill the expectations set forth in the plan view.

In the second chapter refers to the theoretical foundations of SCADA systems, language, SQL, databases, and emphasizes the theoretical information and ActiveFactory IndustrialSQL software are programs that specialize in the creation and management of industrial databases, also describes the theoretical foundation of Web applications, the PLC Logic Controller, JAVA language and TWIDO program.

In the third chapter it is stated that the methodology to be used for the realization of the thesis, the methodology consists of the research design, selection of the type of research and specification of research techniques that were used in the study.

In the fourth chapter has documented the whole process of design and implementation of a SCADA system for the simulation of a juice extraction process, the system also has implemented a database using IndustrialSQL software for their creation and configuration and has used the program to generate tables ActiveFactory actual values

and spells out the process to create a Web page, programming and configuration of the NetBeans software.

In the fifth chapter lists the conclusions and recommendations taken after successful completion of this thesis.

CAPÍTULO I

INTRODUCCIÓN

1.1 Antecedentes

1.1.1 Antecedentes Históricos

Los primeros SCADAS eran simplemente sistemas de telemetría, que proporcionaban reportes periódicos de las condiciones de campo vigilando las señales que representaban medidas o condiciones de estados en ubicaciones de campo remotas. Estos sistemas ofrecían capacidades muy simples de monitoreo y control, sin funciones de aplicación ninguna. La visión del operador en el proceso estaba basada en los controladores y las lámparas detrás de tableros llenos de indicadores. Mientras la tecnología se desarrollaba, las computadoras asumieron el papel de manejar la recolección de datos, disponiendo de comandos de control y una nueva función donde presenta información sobre una pantalla de video.

En los albores de la Automatización, los sistemas que se utilizaban para el control de procesos eran tecnológicamente simples. Con el tiempo ha ido aumentando la complejidad de estos sistemas de forma exponencial, incorporando los últimos avances en visualización de datos, llegando a su máxima expresión al día de hoy, con los ordenadores y las pantallas de visualización como estrellas indiscutibles de la función de diálogo entre el Operador y el Sistema.

1.1.2 Antecedentes Científicos

Automatización Industrial es el uso de sistemas o elementos computarizados para controlar maquinarias y procesos industriales sustituyendo a operadores humanos.

El alcance va más allá que la simple mecanización de los procesos ya que ésta provee a operadores humanos mecanismos para asistirlos en los esfuerzos físicos del trabajo, la automatización reduce ampliamente la necesidad sensorial y mental del humano. La automatización como una disciplina de la ingeniería es más amplia que un mero sistema de control, abarca la instrumentación industrial, que incluye los sensores y transmisores de campo, los sistemas de control y supervisión, los sistema de transmisión y recolección de datos y las aplicaciones de software en tiempo real para supervisar y controlar las operaciones de plantas o procesos industriales.

1.1.3 Antecedentes Prácticos

Las comunicaciones móviles cada día tienen mayor importancia, es una tecnología en constante evolución, que ofrece cada día más y mejores servicios para la comunicación. Lo mismo sucede con los terminales, los bien conocidos “móviles”, los cuales avanzan acordes con los nuevos servicios que ofrecen los operadores. De este modo, actualmente hay terminales que son capaces de navegar por Internet, hacer llamadas de voz, videoconferencias en tiempo real, reproducción de música o de vídeos, etc.

Con el auge de los servicios de Internet en los operadores móviles y los nuevos terminales que ofrece actualmente el mercado, ya se puede pensar que el móvil puede servir para algo más que para hacer llamadas de voz, videoconferencias o navegar por

Internet. Actualmente, la mayoría de los móviles soportan aplicaciones programadas en Java MicroEdition (Java ME). La mayoría de ellas son juegos para entretener al propietario del móvil. Pero si utilizamos un poco la imaginación podríamos pensar en utilizar dicho lenguaje para realizar aplicaciones acordes a las necesidades de cualquier empresa. Un ejemplo claro, sería controlar un proceso industrial desde cualquier lugar del mundo a través del móvil.

1.1.4 Importancia del estudio

La automatización y la implementación de nuevas técnicas o tecnologías en las industrias es algo inevitable en la actualidad, por lo que resulta indispensable que los nuevos profesionales graduados en la Universidad Tecnológica Equinoccial (UTE), tengan conocimientos sobre las nuevas técnicas y tecnologías de automatización. Con estos antecedentes se ve la necesidad de tener un EQUIPO DIDACTICO en el que se demuestre una nueva tecnología para manejar procesos industriales de manera remota, dentro del Laboratorio de Automatización Industrial de la escuela de Ingeniería Electromecánica de la UTE, y de este modo poder complementar los conocimientos teóricos adquiridos.

1.1.5 Situación Actual del tema de investigación

En la actualidad en el sector industrial de la ciudad no existe un sistema de supervisión y control de procesos industriales vía Internet desde un dispositivo móvil. Existiendo una gran cantidad de industrias en la ciudad a las cuales se les puede implementar dicho sistema para que así tanto los accionistas o dueños y los encargados de la operación de los procesos de producción puedan visualizar y controlar sus procesos desde cualquier parte mediante su dispositivo móvil.

1.2 Limitaciones del Estudio

No poder obtener referencias de trabajos similares debido que en nuestro medio las industrias aún no aplican este tipo de tecnologías para el control y supervisión de procesos industriales.

Falta de bibliografía por parte de la universidad para obtener información de los programas InSQL y JAVA, que forman parte de los programas utilizados para el desarrollo del presente trabajo.

1.3 Alcance del trabajo

La implementación de un equipo didáctico en el que se emplee una nueva tecnología de control de procesos industriales en los laboratorios de Automatización Industrial de ingeniería en electromecánica de la UTE nos permitirá tener conocimientos en sistemas de este tipo, lo que nos permitirá el correcto desenvolvimiento en nuestras actividades profesionales dentro de las industrias. Esto nos ayudará a enfrentar los avances tecnológicos y por lo tanto las personas graduadas en la UTE vamos a ser más competitivas con relación a otros profesionales.

1.4 Objetivos del trabajo

1.4.1 Objetivo General

Diseñar e implementar un sistema de Supervisión y Control de un Proceso Industrial Didáctico desde un Dispositivo Móvil.

1.4.2 Objetivos Específicos

- Conocer las aplicaciones y características que tienen a su disposición los dispositivos móviles.
- Realizar la programación en JAVA2 de la Interface Hombre Máquina para luego descargarla al dispositivo móvil.
- Diseñar un sistema SCADA en INTOUCH para el control desde el PC y del Dispositivo Móvil.
- Realizar la comunicación del sistema didáctico con el dispositivo móvil.

1.5 Justificación

Este trabajo está enfocado en aplicar nueva técnicas, herramientas y tecnologías dentro de los controles y supervisión de procesos industriales. Al implementar este equipo se va a contribuir al desarrollo del laboratorio de Automatización Industrial, ya que va a estar a la par con el desarrollo tecnológico existente en el mercado.

La implementación de este equipo didáctico involucra directamente las áreas de Instrumentación Industrial, Control Automático y Automatización Industrial que son materias del pensum de la carrera de ingeniería Electromecánica, por lo que va a contribuir al desarrollo académico de los estudiantes de esta carrera. La implementación de este sistema didáctico nos ayudará a conocer el funcionamiento e interacción de los componentes de un sistema SCADA en dispositivos móviles.

Este método de control y supervisión de un proceso industrial es aplicable en todo tipo de industrias y procesos que tengan características similares, por lo cual es muy necesario tener conocimientos de estos tipos de sistemas y tecnologías empleadas.

El Gerente o Administrador de la planta, así como el personal de mantenimiento, tendrán acceso a los datos y estados de los equipos, dentro de la planta en tiempo real en todo momento y en cualquier lugar para poder mejorar la administración y a su vez mejorar los procesos.

En la actualidad y más aún en la profesión de automatización es de vital importancia conocer y aprovechar los avances tecnológicos que se puedan aplicar en los procesos industriales para así disminuir costos y aumentar beneficios dentro de una industria.

1.6 Hipótesis

La implementación de un Equipo didáctico de Supervisión y Control de un Proceso Industrial en el laboratorio de Automatización Industrial de la UTE, influenciará en el aprendizaje de los estudiantes.

1.7 Metodología

1.7.1 Unidad de Análisis o de Estudio

En la presente investigación la Unidad de Análisis es el sistema de supervisión y control desde un móvil para un proceso industrial en el laboratorio.

1.7.2 Aspectos Metodológicos generales del Estudio

Para el desarrollo del presente trabajo se aplicarán los siguientes tipos de investigación: descriptiva y exploratoria.

1.7.2.1. Descriptiva

Descriptiva porque se va a analizar las diferentes alternativas que existen para cumplir de manera eficaz los objetivos propuestos en este proyecto y luego se va a detallar los pasos que se efectuaron para dicho propósito.

1.7.2.2. Exploratoria

Exploratoria ya que no se va a utilizar ningún modelo anterior como base para este estudio, por lo cual se emplearan enfoques amplios y versátiles. Esto incluyen fuentes secundarias de información como: observación, entrevistas con expertos, entrevistas de grupos con especialistas e historias de casos.

1.7.3 Métodos de Estudio

En el desarrollo de este estudio se utilizarán los siguientes métodos de estudio: Deductivo, analítico y sintético.

1.7.3.1. Deductivo

Método deductivo porque se tiene una gran cantidad de teorías para los programas que se van a utilizar en el presente trabajo, de las cuales se tendrán que interpretar y aplicar las que realmente sean necesarias.

1.7.3.2. Analítico

Éste método porque se debe analizar toda la información obtenida para posteriormente hacer un resumen de lo más importante que se va a aplicar al estudio.

1.7.3.3. Sintético

Debido a que se adquiere el conocimiento de lo simple como es el manejo individual de cada uno de los programas a lo complejo que es la unión e interacción de varios programas.

CAPÍTULO II

FUNDAMENTOS TEÓRICOS

2.1 Estudio de los sistemas SCADA

"SCADA, acrónimo de Supervisory Control and Data Acquisition (Control Supervisor y Adquisición de Datos) es una aplicación de software diseñada para funcionar sobre computadores en el control de producción, proporcionando comunicación con los dispositivos de campo (controladores, autómatas programables, etc.)"[1] Y para posibilitar el control y supervisión desde el computador. Además, provee de toda la información que se genera en el proceso productivo a diversos usuarios, tanto del mismo nivel como de otros supervisores dentro de la empresa: control de calidad, supervisión, mantenimiento, etc.

2.1.1 Criterios de diseño del prototipo SCADA

El prototipo SCADA, básicamente hace referencia a un sistema que permite controlar y/o supervisar plantas o procesos por medio de una estación central, es decir una computadora central que efectúa las tareas de supervisión y gestión de alarmas de tal forma que tiene un control básico de los procesos. Todo esto se ejecuta en tiempo real y está diseñado para que el operador pueda tomar acciones sobre sus procesos.

[1] http://es.wikipedia.org/wiki/SCADA

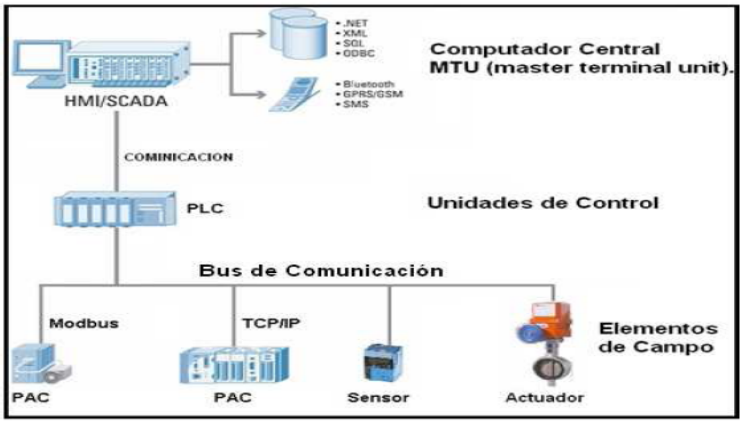
2.1.2 Características del SCADA

- ✓ Es un sistema de arquitectura abierta, capaz de crecer o adaptarse según las necesidades del usuario.
- ✓ Poder comunicarse con total facilidad, y de forma transparente al usuario, con el módulo y con el resto del mundo.
- ✓ Tener programas sencillos de instalar, sin excesivas exigencias de hardware, y fáciles de utilizar, con interfaces amigables con el usuario.

2.1.3 Esquema de un sistema SCADA

En la figura 2.1 se presenta un esquema básico pero representa una síntesis muy adecuada para determinar e identificar los componentes más importantes de la estructura de un sistema SCADA.

Figura 2.1 Esquema de un SCADA



Fuente: National Instrument - zone.ni.com/cms/images/devzone/tut/wer1.jpg
Elaborado por: César Zambrano / 2011

En el esquema anterior se resaltan tres elementos principales que son:

- ✓ **Computador Central (MTU).- "**Conocido como Unidad Maestra. Ejecuta las acciones de mando (programadas) en base a los valores actuales de las variables medidas. La programación se realiza por medio de bloques de programa en lenguaje de alto nivel (como C, Basic, etc.). También se encarga del almacenamiento y procesado ordenado de los datos, de forma que otra aplicación o dispositivo pueda tener acceso a ellos"[2].

- ✓ **La Unidad de Control**.- Este elemento es el que se encarga de captar las señales emitidas por los sensores que están en el campo y mediante una programación realiza el control de los otros elementos de campo. Las señales que recibe no solo son provenientes de los elementos de campo sino que también la Unidad Maestra (MTU) puede enviar señales de control para que la unidad de control realice los comandos que son necesarios para dicha señal. Las unidades controladoras pueden ser Controladores Lógicos Programables (PLC).

- ✓ **Elementos de campo**.- Son los elementos que permiten la conversión de una señal física en una señal eléctrica (y viceversa). Su calibración es muy importante para que no haya problema con la confusión de valores de los datos, entre los elementos de campo más comunes se puede mencionar: sensores, actuadores, electroválvulas, cámaras, etc.

Bus de comunicación.- Éste es el nivel que gestiona la información que los instrumentos de campo envían a la red de ordenadores desde el sistema. El tipo de BUS utilizado en las comunicaciones puede ser muy variado según las necesidades del sistema y del software escogido para implementar el sistema SCADA. Hoy en día, gracias a la estandarización de las comunicaciones con los dispositivos de campo, se puede implementar un sistema SCADA sobre prácticamente cualquier tipo de BUS. Se

---

[2] SISTEMAS SCADAS/ Ing. Henry Mendiburu Díaz/http/hamd.galeon.com

puede encontrar SCADAs sobre formatos estándares como los RS- 232, RS-422 y RS 485 a partir de los cuales, y mediante un protocolo TCP/IP.

2.1.4 Funciones de un sistema SCADA

Supervisión remota de instalaciones y equipos: Permite al operador conocer el estado de desempeño de las instalaciones y los equipos alojados en la planta, lo que permite dirigir las tareas de mantenimiento y estadística de fallas.

Control remoto de instalaciones y equipos: Mediante el sistema se puede activar o desactivar los equipos remotamente (por ejemplo abrir válvulas, activar interruptores, prender motores, etc.), de manera automática y también manual. Además es posible ajustar parámetros, valores de referencia, algoritmos de control, etc.

Procesamiento de datos: El conjunto de datos adquiridos conforman la información que alimenta el sistema, esta información es procesada, analizada, y comparada con datos anteriores, y con datos de otros puntos de referencia, dando como resultado una información confiable y veraz.

Visualización gráfica dinámica: El sistema es capaz de brindar imágenes en movimiento que representen el comportamiento del proceso, dándole al operador la impresión de estar presente dentro de una planta real. Estos gráficos también pueden corresponder a curvas de las señales analizadas en el tiempo.

Generación de reportes: El sistema permite generar informes con datos estadísticos del proceso en un tiempo determinado por el operador.

Representación de señales de alarma: A través de las señales de alarma se logra alertar al operador frente a una falla o la presencia de una condición perjudicial o fuera de lo aceptable. Estas señales pueden ser tanto visuales como sonoras.

Almacenamiento de información histórica: Se cuenta con la opción de almacenar los datos adquiridos, esta información puede analizarse posteriormente, el tiempo de almacenamiento dependerá del operador o del autor del programa.

Programación de eventos: Esta referido a la posibilidad de programar subprogramas que brinden automáticamente reportes, estadísticas, gráfica de curvas, activación de tareas automáticas, etc.

2.1.5 Módulos de un sistema SCADA

Configuración: Este módulo le permite al usuario definir el entorno de trabajo de su SCADA, adaptándolo a la aplicación particular que se desea desarrollar, es decir configurar todos los componentes del sistema para trabajar en coordinación entre todos los elementos. Por ejemplo la selección del software para la pantalla HMI, la selección del controlador, la selección del protocolo de comunicación, determinas si se va o no a implementar un sistema de adquisición y gestión de datos, etc.

Interfaz gráfico del operador: Proporciona al operador las funciones de control y supervisión de la planta Proyectados en una pantalla. El proceso se presenta mediante sinópticos gráficos a los cuales se les puede agregar muchos tipos de animaciones con la intención de hacer que el proceso que se visualiza sea lo parecido posible al proceso real.

Módulo de proceso: Ejecuta las acciones de mando pre programadas a partir de los valores actuales de variables leídas. La programación se la puede realizar en el

controlador que puede ser un PLC (Controlador Lógico Programable) o en el software mismo donde se realiza la HIM. es recomendable que el módulo de proceso tenga la programación en el controlador y no en la pantalla HMI.

Almacenamiento y Gestión de datos: Este módulo se encarga del almacenamiento y procesado ordenado de los datos, de forma que otra aplicación o dispositivo pueda tener acceso a ellos de forma fácil. Por lo general este módulo se maneja con una estructura de cliente – servidor, es decir que la base de datos se convierte en un servidor de datos y los clientes son los usuarios que pueden tener acceso a la información.

En cuanto a la gestión de los datos es una parte importante puesto que es la encargada de generar reportes que representan claramente el comportamiento de las variables de cualquier proceso industrial.

Comunicaciones: En este módulo se realiza la configuración de los diferentes módulos del sistema para establecer la transferencia de información entre la planta y la arquitectura hardware que soporta el SCADA, y entre ésta y el resto de elementos informáticos de gestión.

Se debe determinar cuales serán los protocolos de comunicación para que sean compatibles con todos los elementos inmersos en el sistema SCADA. Obviamente dichos protocolos están regidos por normas internacionales para que la selección de los mismos sea más fácil.

2.2 Estudio de las interfaces de comunicación

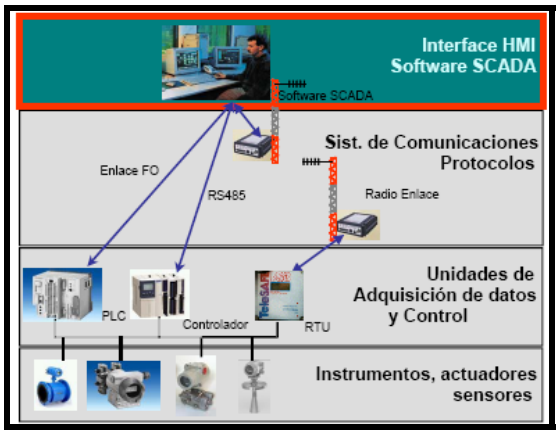
Las interfaces de comunicación se refieren a los mecanismos de hardware y software para que los dispositivos se comuniquen entre si. En el presente proyecto se realizó este

análisis para escoger la solución más adecuada en la comunicación del sistema a implementarse.

Una Interfaz Hombre – "Máquina (MMI) o Interfaz Humano - Máquina (HMI "Human Machine Interface") es el software o hardware que le permite a un operador humano interactuar con una máquina, proceso o sistema y determinar su estado (prendido/apagado) o magnitud de los dispositivos y/o variables físicas que estén presentes en una planta o proceso industrial, de manera amigable, simple y efectiva"[3].

Una HMI puede ser tan simple como las luces indicadoras de un tablero hasta el software de visualización en una computadora, para ello existen diferentes productos comerciales. Dentro de las plataformas de software para el desarrollo de una HMI se tiene por ejemplo el FIX de Intellution, InTouch de Wonderware, DSC de la National Instruments, etc.

Figura 2.2: Componentes de una HMI



Fuente: Sistemas SCADAS

Elaborado por: César Zambrano / 2011

[3] http://leonardo.uncu.edu.ar/catedras/electronica/archivos/Tema9_Scada.pdf

Para el desarrollo de ésta aplicación se escogió el software InTouch, ya que es el programa que tiene la universidad y además se cuenta con las licencias educativas.

2.3 Intouch

InTouch es un paquete de software utilizado para crear aplicaciones de interface hombre-máquina bajo entorno *PC*. **InTouch** utiliza como sistema operativo el entorno *WINDOWS* 95/98/NT/2000.

"Es un software mediante el cual se desarrollan aplicaciones HMI (Interfaces Machine Human) para el control y visualización de procesos de una forma gráfica. Intouch pertenece al paquete Wonderware Factory Suite de Invensys Corp. Y opera sobre los Sistemas Operativos de Windows; puede integrarse con algunos programas de Microsoft especialmente para la generación de reportes y almacenamiento de datos."[4]

2.3.1 Características

Intouch posee tres aplicaciones internas para el desarrollo de HMI que son:

Intouch Aplication Manager. Intouch Aplication Manager permite la creación y modificación de los proyectos, además de las configuraciones de red (Network Application Development).

Windows Maker. Windows Maker es el ambiente de desarrollo donde se utilizan gráficos orientados a objetos, para crear ventanas de visualizaciones animadas; y

[4] www.itba.edu.ar/capis/epg-tesis-y-tf/lopezfiguerola-tfe.pdf

permitirnos configurar la comunicación entre las gráficas, animaciones y dispositivos de control externo.

Windows Viewer. Windows Viewer es el ambiente de ejecución donde se visualiza las aplicaciones desarrolladas en Windows Maker.

Adicionalmente InTouch posee otros elementos muy útiles al momento de diseñar las interfaces:

- ✓ Sistema de alarmas distribuidas, que proporcionan a los operadores la capacidad de visualizar y reconocer simultáneamente información de alarmas desde múltiples ubicaciones remotas.
- ✓ Historial distribuido, que permite especificar en forma dinámica una fuente de datos de archivos históricos, diferente para cada serie de un gráfico de tendencia.
- ✓ El desarrollo de aplicación en red, permite crear una aplicación y ejecutarla en varios nodos de una red.

2.3.2 Programación

Intouch posee una amplia gama de elementos para el desarrollo y la programación de los HMIs.

- ✓ Los Wizards, son elementos inteligentes con los que se puede desarrollar aplicaciones de una forma simple y rápida.
- ✓ Animation Links, permiten dar animaciones a un gráfico o un símbolo creado.
- ✓ Los Quickscripts, son programas escritos por el usuario con la finalidad de manejar cálculos, condiciones de operación, activación y desactivación de alarmas, etc. Los Quickscripts pueden estar asociados a la aplicación, ventanas, teclas, condiciones, cambios de datos y funciones.

2.3.3 Comunicaciones

Intouch utiliza el protocolo DDE (Dynamic Data Exchange), desarrollado por Microsoft, para el intercambio de datos entre aplicaciones Windows. La Comunicación DDE se establece automáticamente entre programas que contemplan la estructura (cliente servidor); así entonces, podemos crear programas con gestiones especiales en VBASIC, EXCEL, etc., y pasar los datos a InTouch sin necesidad de crear un programa de comunicaciones.

Los servidores de autómatas (I/O Servers) que dispone WONDERWARE, prácticamente cubren la totalidad de los PLCs más conocidos del mundo, con comunicación tanto punto a punto como en red.

Intouch puede direccionar un servidor DDE a un puerto de comunicaciones y otro servidor a otro puerto, con esto se puede compartir información que venga de distintos PLCs o sistemas de campo.

2.4 Base De Datos

Una Base de Datos es un conjunto de archivos interrelacionados que contienen información importante de un proceso. Los componentes principales de una Base de Datos son:

- ✓ Hardware
- ✓ DBMS (DataBase Management System)
- ✓ Datos
- ✓ Usuarios

Toda Base de Datos debe poseer las siguientes características:

- ✓ Integración de toda la información.
- ✓ Disponibilidad de los datos en todo momento.
- ✓ Accesibilidad simultánea para distintos usuarios.
- ✓ Independencia de las aplicaciones respecto a la representación física de los datos.
- ✓ Mecanismos para asegurar la integridad y la seguridad de los datos.

2.4.1 DBMS (DataBase Management System)

El DBMS es un Software que permite definir, construir y manipular la información que posea la base de datos; para conseguir este objetivo el DBMS posee los módulos: DDL (Data Definition Lenguaje) que define la estructura de almacenamiento, DML (Data Manipulation Lenguaje) que recupera, elimina o inserta información en una base de datos y un lenguaje de consulta SQL (Structured Query Lenguaje) con el cual se extrae la información que posea la base de datos.

2.4.2 MICROSOFT SQL SEVER

"Microsoft SQL Server (SQL Server) es un DBMS muy utilizado para trabajar con grandes cantidades de datos, además de poseer una integración con aplicaciones Windows y Web"[5].

SQL Server trabaja en la plataforma Windows y con los Sistemas Windows Server, Windows NT, Windows Millenium, Windows 98 y XP. SQL Server posee varias bases de datos del sistema, a continuación se detalla sus características:

---

[5] www.itba.edu.ar/capis/epg-tesis-y-tf/lopezfiguerola-tfe.pdf

- ✓ (Master) Esta Base contiene tablas de sistema que realizan el seguimiento de la instalación del servidor y de todas las bases de datos que se creen posteriormente.
- ✓ (Tempdb) Es una base de datos temporal y posee las tablas temporales creadas por los usuarios y las tablas de trabajo que SQL Server necesita para el procesamiento y la ordenación de las consultas.
- ✓ (Model) Se utiliza como plantilla para todas las bases de datos creadas en un sistema.
- ✓ Msdb) Es empleada por el servicio SQL Server Agent, para guardar información respecto a tareas de automatización, seguridad, de duplicación y solución de problemas.
- ✓ (Distribution) Almacena toda la información referente a la distribución de datos basada en un proceso de replicación.

2.4.2.1 Elementos de SQL server

Los elementos que existen en cada base de datos de SQL Server son:

Tablas. Las Tablas son objetos de la base de datos que contienen la información ingresada por los usuarios y que se encuentran organizados en filas y columnas.

Vista. La Vista es un objeto similar a la tabla, a diferencia que en la vista no existen datos, ya que estos son obtenidos desde las tablas subyacentes a la consulta.

Procedimiento Almacenado. Los Procedimientos Almacenados son un conjunto de instrucciones de consultas precompiladas, las cuales llevan a cabo una operación de consulta o de control.

Desencadenador. El Desencadenador o Trigger es un Procedimiento Almacenado especial el cual se invoca automáticamente ante una operación sobre una tabla. Un Desencadenador puede controlar la integridad de los datos o preservar las relaciones definidas entre las tablas cuando se ingresa o borra datos de aquellas tablas.

Reglas. Las Reglas son objetos que especifican los valores aceptables que pueden ser ingresados dentro de una columna particular.

Restricciones. Las Restricciones son prohibiciones que se asignan a las columnas de una tabla y son controladas automáticamente por SQL Server.

Índices. Los Índices son similares a los índices de un libro, los índices de una tabla permiten buscar información rápidamente sin necesidad de recorrer registro por registro por toda la tabla.

2.4.2.2 Seguridades

Los datos dentro de una base deben estar protegidos para evitar su alteración no autorizada, destrucción, acceso no autorizado, etc., por lo tanto SQL Server brinda los siguientes niveles de seguridad:

- ✓ Nivel de SQL Server y
- ✓ Nivel de Base de Datos.

Un login permite conectarse a través del Nivel de SQL, mientras una cuenta de usuario accede al Nivel de Base de Datos. El Login puede ser un login de Windows NT/2000/2003 que ha sido concedido el acceso a SQL Server. Una cuenta de usuario es

específica en cada Base de Datos. Todos los permisos y los propietarios de los objetos son controlados por la cuenta de usuario.

Durante una solicitud de nueva conexión, SQL Server verifica el nombre del login suministrado, para asegurar que ese login está autorizado para acceder a SQL Server. Este proceso de verificación es llamado autenticación y puede ser:

Modo de Autenticación de Windows. Con la autenticación de Windows, no necesitamos especificar un nombre y contraseña de login para conectarse a SQL Server, debido a que el acceso a SQL Server es controlado por la cuenta de usuario de Windows NT/2000

Modo Mixto. Este modo de autenticación permite a los usuarios conectarse usando autenticación de Windows o autenticación de SQL Server. Para esto, se debe crear una cuenta y contraseña de login válida en SQL Server. Estas cuentas no están relacionadas a las cuentas de Microsoft Windows NT/2000/2003.

2.5 Indutrialsql Server

2.5.1 InSQL

IndustrialSQL Server (InSQL) es una base de Datos de Wonderware, diseñada para la recolección de información de los Sistemas de Producción, PLCs, etc., a una alta velocidad y con una gran capacidad de almacenamiento.

"InSQL está basada en la Base de Datos de Microsoft SQL Server, los datos están accesibles de un modo abierto utilizando funciones de SQL"[6].

[6] www.itba.edu.ar/capis/epg-tesis-y-tf/lopezfiguerola-tfe.pdf

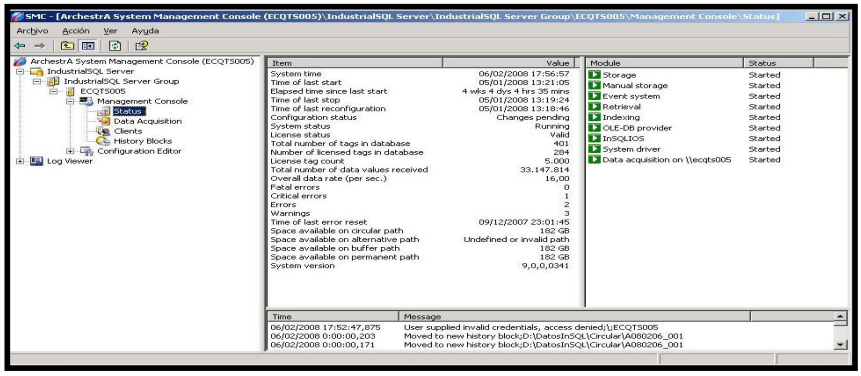
El Historiador **IndustrialSQL Server**, también conocido como InSQL, permite la adquisición de datos de planta en una base de datos de altas prestaciones histórica y a tiempo real. Combina la potencia y flexibilidad de una base de datos relacional con la velocidad y compresión de un sistema a tiempo real, integrando la parte empresarial y de gestión con los datos de planta de una industria. IndustrialSQL Server adquiere datos a muy alta velocidad reduciendo su volumen simultáneamente como extensión de Microsoft SQL Server. Además integra los datos de planta con eventos, resúmenes, datos de producción y configuración.

El historiador IndustrialSQL Server y sus herramientas de análisis asociadas proporcionan acceso detallado a los datos de planta a las personas que toman las decisiones, lo que revierte en una mejora significativa global. Facilita una imagen completa de los procesos de planta debido a que recoge los datos de producción automáticamente a tiempo real de múltiples fuentes simultáneamente, a muy alta velocidad y gran resolución. Gracias a este nivel de visibilidad se pueden corregir inmediatamente errores en procesos concretos o problemas de calidad de producto.

IndustrialSQL Server incorpora la tecnología de Microsoft SQL Server, por lo que puede integrar datos de otras plantas y de otras aplicaciones de gestión empresarial, facilitando un punto de acceso común y un mismo interface entre los sistemas de producción y de negocio. IndustrialSQL Server es escalable a múltiples niveles para dar respuesta a industrias de todos los tamaños y dispone de toda una colección complementaria de herramientas para el análisis, generación de informes y presentación de los datos adquiridos.

Para almacenar la Información proveniente de los PLC se utiliza el programa InSQL Server de Wonderware, este DBMS posee una interfase para configurar y monitorear el estado de los enlaces con cada PLC al igual que información sobre el acceso al sistema, número de Tag, errores, etc.

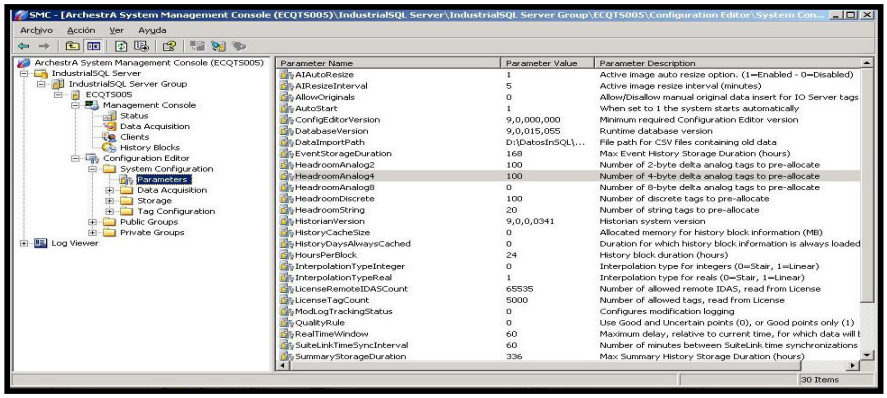
Figura 2.3: Pantalla de Status del DBMS InSQL Server



Fuente: Manual InSQL
Elaborado por: César Zambrano / 2011

Es posible además mostrar y configurar parámetros de este programa como son: el crecimiento de la base de datos, la versión del programa, el número de tag de la licencia, el modo de funcionamiento, etc.

Figura 2.4: Pantalla de Configuración de los Parámetros del InSQL Server



Fuente: Manual InSQL
Elaborado por: César Zambrano / 2011

Para configurar la adquisición de datos es necesario establecer los enlaces del InSQL Server con los PLCs, esto se realiza agregando los diferentes Topics en los cuales se especifica:

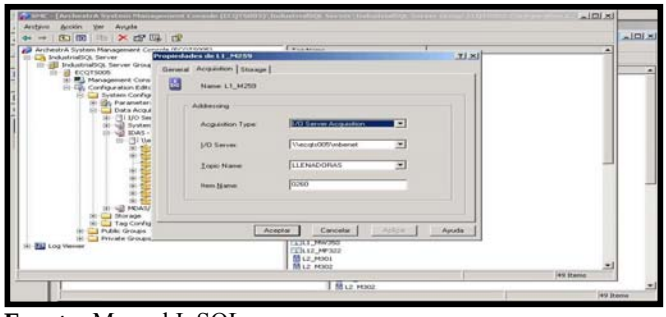
- ✓ El Computador o Servidor donde se encuentra instalado el I/O Server
- ✓ El Nombre del I/O Server
- ✓ El Nombre del Topic configurado en el I/O Server y
- ✓ El Tipo de Protocolo

Figura 2.5: Configuración para la comunicación con los I/O Server

Fuente: Manual InSQL
Elaborado por: César Zambrano / 2011

Adicionalmente se agrega las variables que serán almacenadas. InSQL Server puede almacenar variables discretas, analógicas, strings y eventos, debiendo en cada uno de los Tags declarados en la Base de Datos especificarse el nombre, la descripción, el tipo de adquisición, el I/O Server, el Topic Name y la dirección de memoria del PLC asignada a ese Tag.

Figura 2.6: Configuración de los Tags



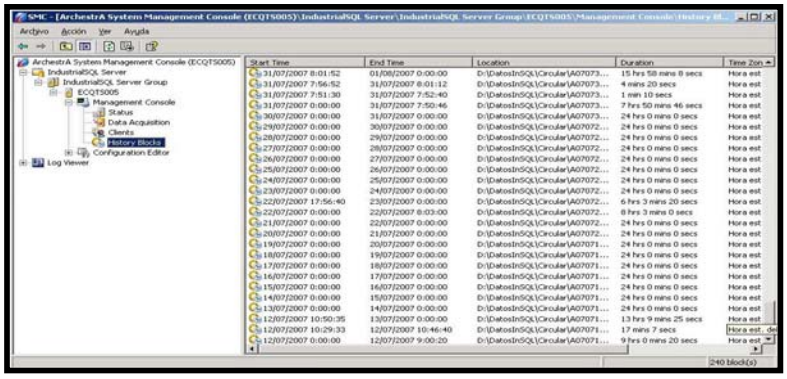
Fuente: Manual InSQL
Elaborado por: César Zambrano / 2011

2.5.2 Consultas y Procedimientos Almacenados

InSQL Server brinda la posibilidad de realizar consultas estructuradas en SQL, para esto se utiliza el DBMS comercial SQL Server 2005 compatible con la versión InSQL.

Los datos almacenados por InSQL Server se encuentran en archivos comprimidos denominados History Block, de esta forma, se optimiza el espacio requerido dentro del disco duro de servidor. Cada History Block es creado automáticamente cada día o cuando InSQL es iniciado.

Figura 2.7: Almacenamiento de Datos en InSQL

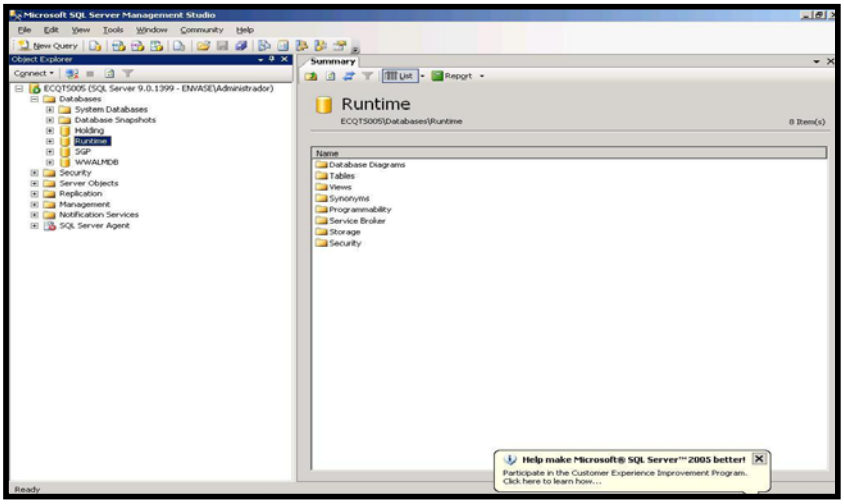


Fuente: Manual InSQL
Elaborado por: César Zambrano / 2011

Al instalar InSQL Server se crea automáticamente una base de datos denomina Runtime la cual posee la información de configuración de los variables creadas en la consola de InSQL. La base Runtime posee todos los elementos de una base de datos y a través de sus tablas o vistas se accede a la información almacenada en los History block.

Para acceder a la información almacenada se puede utilizar la consola de SQL Server (Microsoft SQL Server Managment Studio), desde la cual se accede a todas las bases creadas dentro del DBMS incluida la Runtime.

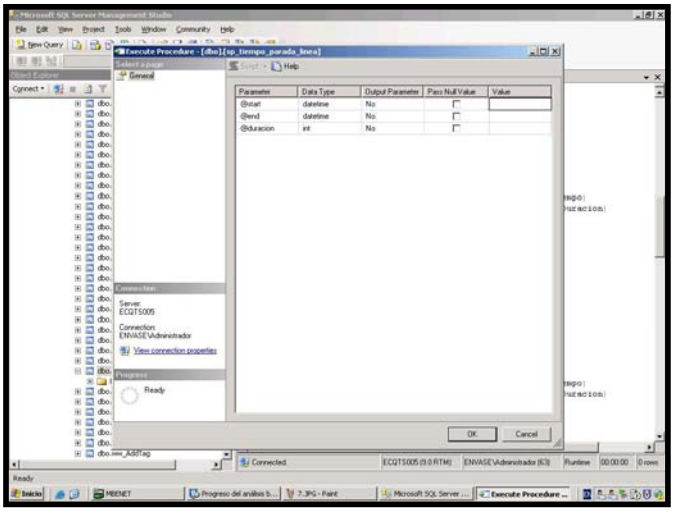
Figura 2.8: Consola de SQL Server 2005



Fuente: Manual InSQL
Elaborado por: César Zambrano / 2011

Los procedimientos almacenados pueden soportar variables de búsqueda que son especificadas por el usuario, de esta manera al llamar al procedimiento se ingresan los parámetros de entrada.

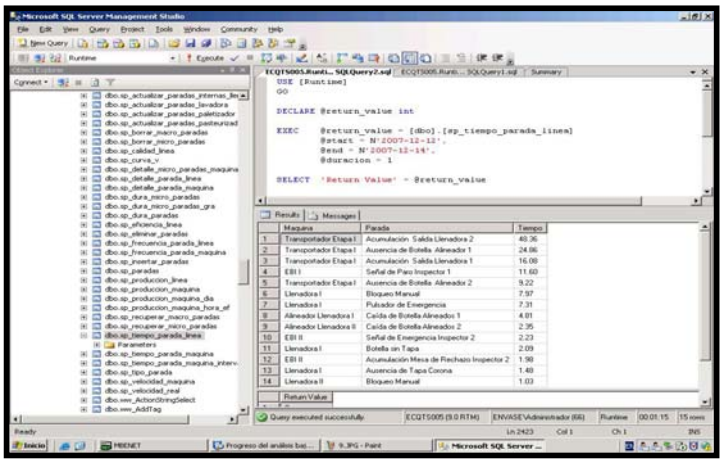
Figura 2.9: Búsqueda de variables



Fuente: Manual InSQL
Elaborado por: César Zambrano / 2011

Cuando el procedimiento es ejecutado se obtiene el resultado de la búsqueda que puede ser un valor o un conjunto de datos.

Figura 2.10: Resultados del Procedimiento Almacenado



Fuente: Manual InSQL
Elaborado por: César Zambrano / 2011

2.6 Active Factory

2.6.1. Definición

El software de análisis y de realización de gráficos de tendencias Wonderware Active Factory ofrece excelentes capacidades de generación de reportes y análisis de datos que maximizan el valor de los datos almacenados en Wonderware Historian. Active Factory ofrece análisis de tendencias de datos, un sofisticado análisis de los datos numéricos utilizando las hojas de cálculo Excel de Microsoft, generación de completos reportes de datos usando Microsoft Word, y la capacidad para publicar información de la planta histórica y en tiempo real a la Web o al sitio intranet de su compañía usando Wonderware Information Server. Los trabajadores con conocimiento de la planta que usan la información de Active Factory pueden identificar y solucionar problemas, estudiar ineficiencias de proceso potenciales y eliminar el proceso de localizar los datos.

2.6.2. Aplicaciones de escritorio

Herramientas ActiveFactory cliente se incluyen las siguientes aplicaciones independientes:

ActiveFactory Trend.- Permite graficar la tendencia de los datos en el tiempo histórico y tiempo real. Las potentes funciones permiten que los datos deben compararse con los otros de diferentes épocas. Alarmas y límite son fácilmente visibles. También es posible agregar anotaciones y ver en su evolución.

ActiveFactory Query.- Esta herramienta con solo apuntar y hacer clic permite búsquedas complejas a creado y ejecutado en contra de cualquier historiador Industrial SQL Server. El conocimiento de la estructura de base de datos o SQL no es necesario.

2.6.3. Complementos de Microsoft Office

El software Active Factory tiene herramientas que funcionan como complementos de Microsoft Office, los cuales se presentan a continuación:

ActiveFactory Workbook.- Este complemento para Microsoft Excel permite casi cualquier tipo de análisis y visualización de datos de un historiador Industrial SQL Server que utiliza el formato de hoja de cálculo Excel (.xls).

ActiveFactory Report.- Este complemento para Microsoft Word permite la presentación de informes sofisticados de un historiador Industrial SQL Server que utiliza el formato de documento Word (.doc)

2.6.4. Controles

Active Factory contiene las herramientas **aaHistClientTrend** y **aaHistClientQuery** que son los controles que proporcionan una funcionalidad esencial de ActiveFactory Trend y ActiveFactory Query para su uso en aplicaciones de envase, tales como InTouch HMI software e Internet Explorer. También puede utilizar ActiveFactory "bloques de construcción" los controles (tales como aaHistClientTagPicker, aaHistClientTimeRangePicker, etc) en sus aplicaciones personalizadas.

2.6.5. ActiveFactory Trend

Trend es una aplicación cliente que permite encontrar a las variables de consulta de base de datos de un servidor Industrial SQL y colócalas en una pantalla gráfica. Trend admite dos tipos de gráficos diferentes: una curva de tendencia regular y un gráfico de dispersión XY. Después de agregar las variables a un gráfico de tendencia, se puede manipular la pantalla en una variedad de maneras, incluyendo paneo, zoom, y la escala. Usted puede personalizar cualquier tendencia mediante la configuración de opciones de visualización y establecer las opciones generales.

2.6.6. ActiveFactory Query

Query es una aplicación cliente que le permite recuperar todo tipo de valores de una base de datos del servidor Industrial SQL historiador o cualquier base de datos SQL Server y devuelve los resultados en un formato de tabla. Si se está consultando la base de datos del servidor Industrial SQL historiador, se puede elegir entre varios tipos de consultas predefinidas y es fácil seleccionar las opciones de cada tipo, eliminando la necesidad de conocer la sintaxis SQL. La consulta SQL es creado para usuarios con poco conocimiento del lenguaje de programación SQL. También puede escribir consultas personalizadas si usted sabe de sintaxis SQL y el esquema de base de datos que está utilizando.

2.6.7. ActiveFactory Workbook

ActiveFactory Workbook es un complemento para Microsoft Excel que le permite consultar una o más Industrial SQL historiador Server o SQL Server bases de datos y devolver los resultados a una hoja de cálculo. Usando el ActiveFactory Workbook, usted puede crear fácilmente informes utilizando datos Industrial SQL historiador servidor sin necesidad de un conocimiento en profundidad de secuencias de comandos SQL. Los informes que se crean con ActiveFactory Workbook pueden ser salvados, lo que le permite ejecutar un informe nuevo en cualquier momento.

Si ActiveFactory Workbook está instalado en su computadora, un menú adicional se añade a Microsoft Excel. Este menú contiene todos los comandos de ActiveFactory los cuales se utilizan para crear un informe con los datos Industrial SQL. Además, puede utilizar la barra de herramientas ActiveFactory acceder a algunos de los comandos.

2.6.8. ActiveFactory Report

ActiveFactory Report permite realizar una consulta desde Microsoft Word de información dentro de la base de datos del Industrial SQL Server y regresa el resultado en forma de reporte a un documento de Word.

ActiveFactory Report es un "complemento" a Microsoft Word. Un complemento es un programa suplementario que se ejecuta dentro de la aplicación Microsoft Word y ofrece funciones personalizadas y comandos especializados.

Si el ActiveFactory Report está instalado, un menú adicional se añade a Microsoft Word. Después de que el complemento está cargado, el menú ActiveFactory contiene todos los comandos que se utiliza para crear un documento de informe o plantilla de

informe utilizando datos de un historiador Industrial SQL Server o una base de datos de SQL Server normal.

2.7 Aplicaciones WEB

Las Aplicaciones Web son un conjunto de páginas relacionadas a través de una estructura de navegación, desde las cuales se puede obtener información mediante una red y un navegador Web. Los Navegadores más conocidos son Explorer de Microsoft y Mozilla Firefox.

Generalmente las aplicaciones Web presentan tres niveles o capas básicas:

- ✓ **Capa de Acceso a Datos (DAL**). En la Capa de Acceso a Datos están todos los elementos que hacen posible la comunicación y la extracción de información desde una Base de Datos.

- ✓ **Capa Lógica de Negocio (BLL).** Capa Lógica de Negocios hace referencia al componente que encapsula toda la lógica empresarial de la aplicación.

- ✓ **Capa de Presentación.** Capa de Presentación contiene las páginas de aplicaciones Web que el Usuario generalmente ve.

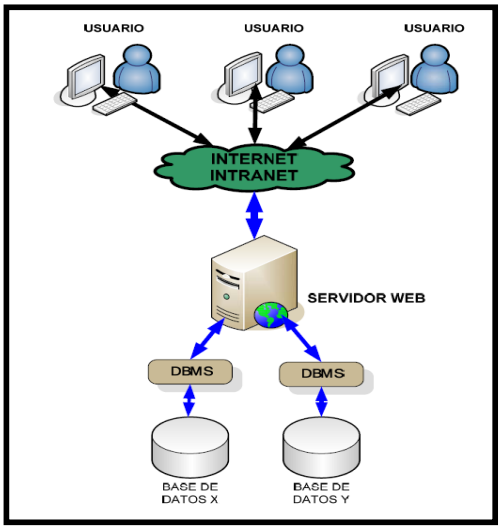
Figura 2.11: Capas de la Aplicación Web

Fuente: Página Web
Elaborado por: César Zambrano / 2011

2.7.1 Arquitectura de las aplicaciones Web

"El usuario utiliza un navegador (Internet Explorer) para conectarse a una aplicación Web instalada en algún servidor de la red, con el objetivo de ingresar o extraer información de una o varias bases de datos; para lo cual es preciso que el servidor Web posea la capacidad de manejar a varios usuarios simultáneamente"[7].

Figura 2.12: Arquitectura de la Aplicación Web



Fuente: Página Web
Elaborado por: César Zambrano / 2011

2.8 PLC

2.8.1 PLC (Controlador Lógico Programable)

El PLC (Controlador Lógico Programable) es un dispositivo electrónico programable creado a finales de los 60 y principios de los 70, fue diseñado para realizar diferentes

[7]http://bieec.epn.edu.ec:8180/dspace/bitstream/123456789/1134/4/T10991CAP%202.pdf

tareas de control en la Industria Automotriz además de soportar las más severas condiciones de trabajo.

Actualmente el PLC se ha extendido a varios procesos Industriales como:

- ✓ Plantas de Refinamiento
- ✓ Líneas de Embotellado
- ✓ Plantas de Producción de Alimentos, etc.

2.8.2 Estructura de un PLC

Básicamente en un PLC se encuentran los siguientes elementos:

- ✓ Fuente de Alimentación
- ✓ CPU (Unidad Central de Procesamiento)
- ✓ Módulos de Entrada y Salidas

Figura 2.13: Elementos del PLC

Fuente: Manuales
Elaborado por: César Zambrano / 2011

2.8.2.1 Fuente de alimentación

La Fuente de Alimentación provee voltaje para la CPU y los Módulos de Entradas y Salidas. Generalmente los PLCs trabajan con un voltaje estabilizado de 24 V. Dentro de la Fuente de Alimentación existe una Batería de Respaldo para la Memoria Ram que contiene el programa y algunas configuraciones.

2.8.2.2 CPU (unidad central de procesamiento)

La CPU es la parte más importante del PLC, internamente posee un Procesador y Memoria. El Procesador contiene un Microprocesador que se encarga de ejecutar las operaciones lógicas, aritméticas, trasferencias internas de información y comunicación. La Memoria es el elemento capaz de almacenar la información que el PLC requiere para su funcionamiento, existen varios tipos de memorias, siendo las más utilizadas:

RAM (Random Acces Memory) esta es una memoria que puede ser leída y escrita, sin embargo la información se pierde al existir cortes de energía.

ROM (Read Only Memory), este tipo de memoria está programada en el momento de la instalación en el PLC.

PROM (Programable ROM), memoria programable eléctricamente solo una vez.

EPROM (Erasable PROM) memoria programable eléctricamente y borrada con la ayuda de luz ultravioleta.

EEPROM (Electrically Erasable PROM) Memoria programable y borrable eléctricamente.

2.8.2.3 Módulos De Entradas y Salidas

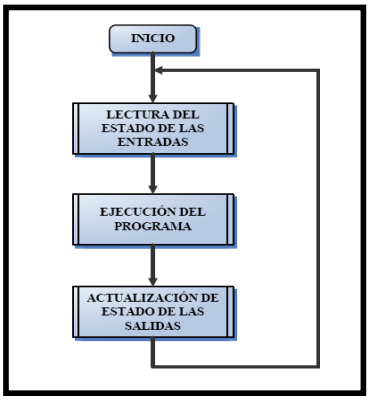
Los Módulos de Entradas y Salidas son los vínculos entre el PLC y el Proceso. Tanto las Entradas como las Salidas pueden ser del tipo Digital o Analógica. "Dentro de la estructura del controlador programable, las interfases o adaptadores de entradas y salidas cumplen la función de conectar el equipo con la vida exterior del CPU"[8].

2.8.3 Funcionamiento

El PLC realiza un proceso cíclico denominado SCAN, en este proceso existen varias etapas como son:

- ✓ Lectura de Entradas, verifica el estado de las entradas
- ✓ Ejecución del Programa contenido en la memoria y
- ✓ Actualización del Estado de las Salidas

Figura 2.14: Proceso del PLC



Fuente: Manual InSQL
Elaborado por: César Zambrano / 2011

[8] http://materias.fi.uba.ar/6722/apunteplc.pdf

2.8.4 Programación

Los programas de los PLCs son una secuencia lógica de eventos que el proceso puede presentar, de esta forma se puede controlar el funcionamiento y las acciones a tomarse en caso de paradas de emergencia, accionamiento de seguridades. La programación se la realiza generalmente desde un PC, para esto se debe contar con el Software indicado para la serie de PLC a programarse.

A fin de estandarizar los Lenguajes de programación de los PLCs, sin importar la marca, la IEC1 estableció el estándar IEC 1131-3, que regula los lenguajes de programación; siendo estos:

- ✓ Escalera (ladder)
- ✓ Lista de instrucciones (Assembler)
- ✓ Estructurado (Similar al Pascal)
- ✓ Texto
- ✓ Bloques de Función y
- ✓ Diagrama Secuencial de funciones (SFC).

2.8.5 Comunicación

La evolución de los PLCs en su parte estructural, ha llevado también a una evolución en su capacidad y forma de comunicación con los elementos externos, ya sea para Programación, Intercambio de Información, Monitoreo del Proceso, etc.

Una de las primeras formas de comunicación fue mediante señales analógicas, siendo las más comunes, las señales de voltaje (0 a 10V) y la de corriente (4 a 20 mA).

Adicionalmente, la comunicación de los PLC se ha desarrollado en el Área de la comunicación digital, siendo ésta la más utilizada actualmente por sus grandes ventajas

como velocidad de comunicación, inmunidad al ruido, cobertura para grandes distancias, etc.

2.9 Twido

El software de programación TwidoSuite está diseñado para ejecutarse en varios sistemas operativos Windows 2000/XP/Vista. En este capítulo se describen los requisitos de sistema del ordenador para instalar y ejecutar este software. También proporciona instrucciones para instalar o desinstalar e iniciar el software.

2.9.1 Instalación del software

Para instalar el software de programación TwidoSuite:

1. Insertar el CD de TwidoSuite en la unidad de CD-ROM del PC. Si está activada la función AUTORUN, la instalación comenzará automáticamente; a continuación, ir al paso 4. De lo contrario, ir al paso 2.
2. Si la instalación no empieza automáticamente, hacer clic en **Inicio** → **Ejecutar**. Aparecerá el cuadro de diálogo **Ejecutar**.
3. Introducir [Unidad:]\setup.exe; a continuación, hacer clic en **Aceptar**. Aparecerá la pantalla de selección de lenguaje.
4. Seleccionar un lenguaje y hacer clic en **Aceptar**. Aparece un mensaje preguntándole si desea registrar la copia del software.
5. Para registrar el software, haga clic en **Sí**. Para ejecutar el software sin registrarlo, haga clic en **No**.
6. Un paquete de software sin registrar funcionará durante 30 días, transcurridos los cuales dejará de hacerlo. Deberá desinstalar el software caducado antes de instalar una versión nueva. La instalación nueva debe registrarse antes de que funcione.
7. Seleccionar el lenguaje de instalación entre la lista de lenguajes disponibles.

8. El software se instalará en el lenguaje local seleccionado y en la versión en inglés. Después, podrá ejecutar el software en cualquiera de estos lenguajes seleccionando el lenguaje de ejecución en el Iniciador de la aplicación TwidoSuite.
9. Siga las instrucciones de instalación restantes que aparecen en pantalla.

2.9.2 Registro de TwidoSuite

Tómese unos minutos para registrar este software y convertirse en un miembro de la comunidad Schneider Automation. El registro es gratuito y le mantendrá informado de las noticias sobre los productos más recientes, actualizaciones de software y firmware para su controlador Twido.

Cómo registrar TwidoSuite

1. Vaya a **Misceláneo.**

2. Haga clic en **Acerca de** en la barra de tareas.

3. Rellene el formulario **Licencia** con la información «Empresa», «Usuario» y «Número de serie» y pulse **Ahora**.

Resultado: se inicia el **Asistente de registro TwidoSuite**.

4. Siga las instrucciones en pantalla para registrar la copia del software TwidoSuite.

Nota: Hay cuatro formas de registrarse: por la Web, el teléfono, el fax y por correo electrónico.

5. Lea las Condiciones del servicio; debe aceptarlas para poder continuar.

6. Envíe el registro.

2.9.3 Inicio de programa

Puede iniciar el programa mediante el **Iniciador de la aplicación TwidoSuite:**

Desde la ruta **Inicio → Programas → Schneider Electric → TwidoSuite → Iniciador de la aplicación**; o bien, desde el acceso directo del Iniciador de la aplicación del escritorio que se ha instalado con el programa:

Navegación por el espacio de trabajo de TwidoSuite

"La navegación por el interface del Twidosuite es muy intuitiva y gráfica ya que sigue los pasos de ciclo de desarrollo natural de una aplicación de automatización, por eso la navegación y la compresión de que se realiza en esa ventana es tan sencilla"[9].

En el espacio de trabajo general siempre tendremos una serie de barras, pestañas y menús que tendrán las siguientes funciones:

- ✓ **Barra de pasos de la aplicación:** Muestra los cuatro pasos de la aplicación TwidoSuite (Proyecto, Describir, Programar, Documentar).
- ✓ **Barra de subpasos del programa:** Muestra los tres subpasos del programa (Configurar, Programa, Depuración). Aparece únicamente cuando el paso Programa está seleccionado.
- ✓ **Barra de tareas:** Proporciona acceso a todas las tareas que puede realizar en el paso o subpaso seleccionado de la aplicación.
- ✓ **Barra de funciones:** Proporciona acceso a funciones especiales asociadas a la tarea seleccionada.
- ✓ **Barra de acceso rápido:** Muestra los comandos Anterior/Siguiente y los accesos directos a Guardar y a Analizar programa en todo momento.
- ✓ **Editores y visualizadores:** Se trata de ventanas de TwidoSuite que organizan los controles de programación y configuración de manera que las aplicaciones puedan desarrollarse correctamente.

[9] http://www.equiposdidacticos.com/pdf/catalogos/Manual_Twido.pdf

- ✓ **Barra del cuadro de lista Error:** Muestra información acerca de los posibles errores o advertencias de la aplicación.

2.9.4 Configuración básica del Hardware Twido:

"Lo primero que hay que hacer cuando se inicia la tarea de realizar un proyecto de automatización, es la configuración o descripción del hardware que se necesitará para dicho propósito, por lo tanto en función de ciertas premisas como son: El número de entradas y salidas (así como el tipo), la necesidad de memoria y velocidad en la CPU, necesidad de buses de comunicación...etc"[10].

Todo este proceso de descripción de la aplicación desemboca en la elección de un hardware determinado que se ajuste a las necesidades de la aplicación. Siendo distinto de una aplicación a otra.

2.10 Java

2.10.1 Introducción

La empresa Sun Microsystems lanzó a mediados de los años 90 el lenguaje de programación Java que, aunque en un principio fue diseñado para generar aplicaciones que controlaran electrodomésticos como lavadoras, frigoríficos, etc. debido a su gran robustez e independencia de la plataforma donde se ejecutase el código, desde sus comienzos se utilizó para la creación de componentes interactivos integrados en páginas Web y programación de aplicaciones independientes. Estos componentes se denominaron applets y casi todo el trabajo de los programadores se dedicó al desarrollo de éstos. Con los años, Java ha progresado enormemente en varios ámbitos como

[10] http://www.equiposdidacticos.com/pdf/catalogos/Manual_Twido.pdf

servicios HTTP, servidores de aplicaciones, acceso a bases de datos (JDBC)... Como **vemos Java se ha ido adaptando a las necesidades tanto de los usuarios como** de las empresas ofreciendo soluciones y servicios tanto a unos como a otros. Debido a la explosión tecnológica de estos últimos años Java ha desarrollado soluciones personalizadas para cada ámbito tecnológico. Sun ha agrupado cada uno de esos ámbitos en una edición distinta de su lenguaje Java. Estas ediciones son Java 2 Stantard Edition, orientada al desarrollo de aplicaciones independientes y de applets, Java 2 Enterprise Edition, enfocada al entorno empresarial y Java 2 Micro Edition, orientada a la programación de aplicaciones para pequeños dispositivos. En esta última edición de Java es en la que vamos a centrar todo nuestro estudio de ahora en adelante.

"La edición Java 2 Micro Edition fue presentada en 1999 por Sun Microsystems con el propósito de habilitar aplicaciones Java para pequeños dispositivos. En esta presentación, lo que realmente se enseñó fue una primera versión de una nueva Java Virtual Machine (JVM) que podía ejecutarse en dispositivos Palm. Para empezar podemos decir que Java Micro Edition es la versión del lenguaje Java que está orientada al desarrollo de aplicaciones para dispositivos pequeños con capacidades restringidas tanto en pantalla gráfica, como de procesamiento y memoria (teléfonos móviles, PDA`s, Handhelds, Pagers, etc)"[11]. La tardía aparición de esta tecnología, (hemos visto que la tecnología Java nació a mediados de los 90 y Java Micro Edition apareció a finales), puede ser debido a que las necesidades de los usuarios de telefonía móvil ha cambiado mucho en estos últimos años y cada vez demandan más servicios y prestaciones por parte tanto de los terminales como de las compañías.

Además el uso de esta tecnología depende del asentamiento en el mercado de otras, como GPRS, íntimamente asociada a J2ME y que ya está a nuestro alcance desde hace algunos años. J2ME es la tecnología del futuro para la industria de los dispositivos móviles. Actualmente las compañías telefónicas y los fabricantes de móviles están implantando los protocolos y dispositivos necesarios para soportarla.

---

[11] SERGIO GÁLVEZ ROJAS; LUCAS ORTEGA DIAZ- JAVA 2 Micro Edition, Pág. 1

2.10.2 J2ME (Java 2 Platform, Micro Edition)

Java 2 Platform, Micro Edition (J2ME): Esta versión de Java está enfocada a la aplicación de la tecnología Java en dispositivos electrónicos con capacidades computacionales y gráficas muy reducidas, tales como teléfonos móviles, PDAs o electrodomésticos inteligentes. Esta edición tiene unos componentes básicos que la diferencian de las otras versiones, como el uso de una máquina virtual denominada KVM (Kilo Virtual Machine, debido a que requiere sólo unos pocos Kilobytes de memoria para funcionar) en vez del uso de la JVM clásica, inclusión de un pequeño y rápido recolector de basura y otras diferencias. En este apartado vamos a ver cuales son los componentes que forman parte de esta tecnología.

- ✓ Por un lado tenemos una serie de máquinas virtuales Java con diferentes requisitos, cada una para diferentes tipos de pequeños dispositivos.
- ✓ Configuraciones, que son un conjunto de clases básicas orientadas a conformar el corazón de las implementaciones para dispositivos de características específicas. Existen 2 configuraciones definidas en J2ME: Connected Limited Device Configuration (CLDC) enfocada a dispositivos con restricciones de procesamiento y memoria, y Connected Device Configuration (CDC) enfocada a dispositivos con más recursos.
- ✓ Perfiles, que son unas bibliotecas Java de clases específicas orientadas a implementar funcionalidades de más alto nivel para familias específicas de dispositivos.

Un entorno de ejecución determinado de J2ME se compone entonces de una selección de:

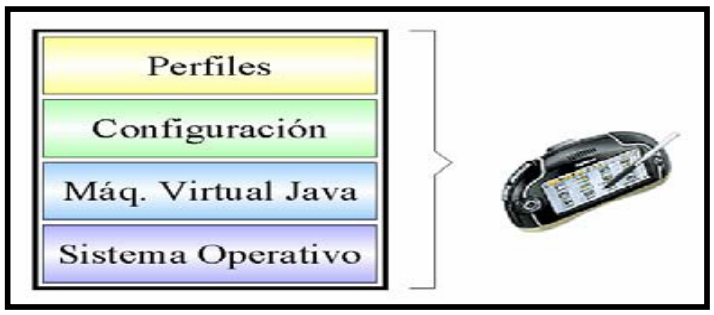
- ✓ Máquina virtual.
- ✓ Configuración.
- ✓ Perfil.

✓ Paquetes Opcionales.

2.10.3 Máquinas Virtuales J2ME

"Una máquina virtual de Java (JVM) es un programa encargado de interpretar código intermedio (bytecode) de los programas Java precompilados a código máquina ejecutable por la plataforma, efectuar las llamadas pertinentes al sistema operativo subyacente y observar las reglas de seguridad y corrección de código definidas para el lenguaje Java"[12]. De esta forma, la JVM proporciona al programa Java independencia de la plataforma con respecto al hardware y al sistema operativo subyacente. Las implementaciones tradicionales de JVM son, en general, muy pesadas en cuanto a memoria ocupada y requerimientos computacionales. J2ME define varias JVMs de referencia adecuadas al ámbito de los dispositivos electrónicos que, en algunos casos, suprimen algunas características con el fin de obtener una implementación menos exigente.

Figura 2.15 Entorno de ejecución



Fuente: Manual JAVA
Elaborado por: César Zambrano / 2011

Ya hemos visto que existen 2 configuraciones CLDC y CDC, cada una con unas características propias que veremos en profundidad más adelante. Como consecuencia, cada una requiere su propia máquina virtual. La VM (Virtual Machine) de la

[12] SERGIO GÁLVEZ ROJAS; LUCAS ORTEGA DIAZ- JAVA 2 Micro Edition, Pág. 5

configuración CLDC se denomina KVM y la de la configuración CDC se denomina CVM. Veremos a continuación las características principales de cada una de ellas:

2.10.4 KVM

"Se corresponde con la Máquina Virtual más pequeña desarrollada por Sun. Su nombre KVM proviene de Kilobyte (haciendo referencia a la baja ocupación de memoria, entre 40Kb y 80Kb)"[13]. Se trata de una implementación de Máquina Virtual reducida y especialmente orientada a dispositivos con bajas capacidades computacionales y de memoria. La KVM está escrita en lenguaje C, aproximadamente unas 24000 líneas de código, y fue diseñada para ser:

- ✓ Pequeña, con una carga de memoria entre los 40Kb y los 80 Kb, dependiendo de la plataforma y las opciones de compilación.
- ✓ Alta portabilidad.
- ✓ Modulable.
- ✓ Lo más completa y rápida posible y sin sacrificar características para las que fue diseñada.

Sin embargo, esta baja ocupación de memoria hace que posea algunas limitaciones con respecto a la clásica Java Virtual Machine (JVM):

- ✓ No hay soporte para tipos en coma flotante. No existen por tanto los tipos dobles ni float. Esta limitación está presente porque los dispositivos carecen del hardware necesario para estas operaciones.
- ✓ No existe soporte para JNI (Java Native Interface) debido a los recursos limitados de memoria.
- ✓ No existen cargadores de clases (class loaders) definidos por el usuario. Sólo existen los predefinidos.

---

[13] SERGIO GÁLVEZ ROJAS; LUCAS ORTEGA DIAZ- JAVA 2 Micro Edition, Pág. 6

- ✓ No se permiten los grupos de hilos o hilos daemon. Cuándo queramos utilizar grupos de hilos utilizaremos los objetos *Colección* para almacenar cada hilo en el ámbito de la aplicación.
- ✓ No existe la finalización de instancias de clases.
- ✓ No hay referencias débiles.
- ✓ Limitada capacidad para el manejo de excepciones debido a que el manejo de éstas depende en gran parte de las APIs de cada dispositivo por lo que son éstos los que controlan la mayoría de las excepciones.
- ✓ Reflexión

La KVM puede ser compilada y probada en 3 plataformas distintas:

- ✓ Solaris Operating Environment.
- ✓ Windows
- ✓ PalmOs

2.10.5 CVM

"La CVM (Compact Virtual Machine) ha sido tomada como Máquina Virtual Java de referencia para la configuración CDC y soporta las mismas características que la Máquina Virtual de J2SE. Está orientada a dispositivos electrónicos con procesadores de 32 bits de gama alta y en torno a 2Mb o más de memoria RAM"[14]. Las características que presenta esta Máquina Virtual son:

- ✓ Sistema de memoria avanzado.

---

[14] SERGIO GÁLVEZ ROJAS; LUCAS ORTEGA DIAZ- JAVA 2 Micro Edition, Pág. 8

- ✓ Tiempo de espera bajo para el recolector de basura.
- ✓ Separación completa de la VM del sistema de memoria.
- ✓ Recolector de basura modularizado.
- ✓ Portabilidad.
- ✓ Rápida sincronización.
- ✓ Ejecución de las clases Java fuera de la memoria de sólo lectura (ROM).
- ✓ Soporte nativo de hilos.
- ✓ Baja ocupación en memoria de las clases.
- ✓ Proporciona soporte e interfaces para servicios en Sistemas Operativos de Tiempo Real.
- ✓ Conversión de hilos Java a hilos nativos.
- ✓ Soporte para todas las características de Java2 v1.3 y librerías de seguridad, referencias débiles, Interfaz Nativa de Java (JNI), invocación remota de métodos (RMI), Interfaz de depuración de la Máquina Virtual (JVMDI).

2.10.6 Configuraciones

Ya se ha mencionado algo anteriormente relacionado con las configuraciones. Para tenerlo bien claro diremos que una configuración es el conjunto mínimo de APIs Java que permiten desarrollar aplicaciones para un grupo de dispositivos. Éstas APIs describen las características básicas, comunes a todos los dispositivos:

- ✓ Características soportadas del lenguaje de programación Java.
- ✓ Características soportadas por la Máquina Virtual Java.
- ✓ Bibliotecas básicas de Java y APIs soportadas.

Como ya hemos visto con anterioridad, existen dos configuraciones en J2ME:

CLDC, orientada a dispositivos con limitaciones computacionales y de memoria y CDC, orientada a dispositivos con no tantas limitaciones. Ahora veremos un poco más en profundidad cada una de estas configuraciones.

2.10.6.1 Configuración de dispositivos con conexión, CDC (Connected device Configuration).

"La CDC está orientada a dispositivos con cierta capacidad computacional y de memoria. Por ejemplo, decodificadores de televisión digital, televisores con internet, algunos electrodomésticos y sistemas de navegación en automóviles"[15]. CDC usa una Máquina Virtual Java similar en sus características a una de J2SE, pero con limitaciones en el apartado gráfico y de memoria del dispositivo. Ésta Máquina Virtual es la que hemos visto como CVM (Compact Virtual Machine). La CDC está enfocada a dispositivos con las siguientes capacidades:

- ✓ Procesador de 32 bits.
- ✓ Disponer de 2 Mb o más de memoria total, incluyendo memoria RAM y
- ✓ ROM.
- ✓ Poseer la funcionalidad completa de la Máquina Virtual Java2.
- ✓ Conectividad a algún tipo de red.

La CDC está basada en J2SE v1.3 e incluye varios paquetes Java de la edición estándar. Las peculiaridades de la CDC están contenidas principalmente en el paquete javax.microedition, que incluye soporte para comunicaciones http y basadas en datagramas.

2.10.6.2 Configuración de dispositivos limitados con conexión, CLDC (connected limited device configuration).

[15] SERGIO GÁLVEZ ROJAS; LUCAS ORTEGA DIAZ- JAVA 2 Micro Edition, Pág. 9

"La CLDC está orientada a dispositivos dotados de conexión y con limitaciones en cuanto a capacidad gráfica, cómputo y memoria. Un ejemplo de éstos dispositivos son: teléfonos móviles, buscapersonas (pagers), PDAs, organizadores personales, etc"[16].

Ya hemos dicho que CLDC está orientado a dispositivos con ciertas restricciones. Algunas de estas restricciones vienen dadas por el uso de la KVM, necesaria al trabajar con la CLDC debido a su pequeño tamaño. Los dispositivos que usan CLDC deben cumplir los siguientes requisitos:

- ✓ Disponer entre 160 Kb y 512 Kb de memoria total disponible. Como mínimo se debe disponer de 128 Kb de memoria no volátil para la Máquina Virtual Java y las bibliotecas CLDC, y 32 Kb de memoria volátil para la Máquina Virtual en tiempo de ejecución.
- ✓ Procesador de 16 o 32 bits con al menos 25 Mhz de velocidad.
- ✓ Ofrecer bajo consumo, debido a que estos dispositivos trabajan con suministro de energía limitado, normalmente baterías.
- ✓ Tener conexión a algún tipo de red, normalmente sin cable, con conexión intermitente y ancho de banda limitado (unos 9600 bps).

La CLDC aporta las siguientes funcionalidades a los dispositivos:

- ✓ Un subconjunto del lenguaje Java y todas las restricciones de su Máquina
- ✓ Virtual (KVM).
- ✓ Un subconjunto de las bibliotecas Java del núcleo.
- ✓ Soporte para E/S básica.
- ✓ Soporte para acceso a redes.
- ✓ Seguridad.

[16] SERGIO GÁLVEZ ROJAS; LUCAS ORTEGA DIAZ- JAVA 2 Micro Edition, Pág. 10

2.10.7 Perfiles

Acabamos de decir que el perfil es el que define las APIs que controlan el ciclo de vida de la aplicación, interfaz de usuario, etc. Más concretamente, un perfil es un conjunto de APIs orientado a un ámbito de aplicación determinado. Los perfiles identifican un grupo de dispositivos por la funcionalidad que proporcionan (electrodomésticos, teléfonos móviles, etc.) y el tipo de aplicaciones que se ejecutarán en ellos. Las librerías de la interfaz gráfica son un componente muy importante en la definición de un perfil. Aquí nos podemos encontrar grandes diferencias entre interfaces, desde el menú textual de los teléfonos móviles hasta los táctiles de los PDAs.

El perfil establece unas APIs que definen las características de un dispositivo, mientras que la configuración hace lo propio con una familia de ellos. Esto hace que a la hora de construir una aplicación se cuente tanto con las APIs del perfil como de la configuración. Tenemos que tener en cuenta que un perfil siempre se construye sobre una configuración determinada. De este modo, podemos pensar en un perfil como un conjunto de APIs que dotan a una configuración de funcionalidad específica.

Ya hemos visto los conceptos necesarios para entender cómo es un entorno de ejecución en Java Micro Edition. Este entorno de ejecución se estructura en capas, una construida sobre la otra.

Anteriormente vimos que para una configuración determinada se usaba una Máquina Virtual Java específica. Teníamos que con la configuración CDC usábamos la CVM y que con la configuración CLDC usábamos la KVM. Con los perfiles ocurre lo mismo. Existen unos perfiles que construiremos sobre la configuración CDC y otros que construiremos sobre la CLDC. Para la configuración CDC tenemos los siguientes perfiles:

- ✓ Foundation Profile.

- ✓ Personal Profile.
- ✓ RMI Profile.

Y para la configuración CLDC tenemos los siguientes:

- ✓ PDA Profile.
- ✓ Mobile Information Device Profile (MIDP).

2.10.8 J2ME y las comunicaciones

Ya hemos visto que una característica importante que tienen que tener los dispositivos que hagan uso de J2ME, más específicamente CLDC/MIDP (a partir de ahora nuestro estudio de J2ME se va a basar exclusivamente en el entorno CLDC/MIDP) es que necesitan poseer conexión a algún tipo de red, por lo que la comunicación de estos dispositivos cobra una gran importancia. En este apartado vamos a ver cómo participan las distintas tecnologías en estos dispositivos y cómo influyen en el uso de la tecnología J2ME. Para ello vamos a centrarnos en un dispositivo en especial: los teléfonos móviles. De aquí en adelante todo el estudio y la creación de aplicaciones se realizarán para este dispositivo. Esto es así debido a la rápida evolución que han tenido los teléfonos móviles en el sector de las comunicaciones, lo que ha facilitado el desarrollo, por parte de algunas empresas, de herramientas que usaremos para crear las aplicaciones.

Uno de los primeros avances de la telefonía móvil en el sector de las comunicaciones se dio con la aparición de la tecnología WAP. WAP proviene de Wireless Application Protocol o Protocolo de Aplicación Inalámbrica. Es un protocolo con el que se ha tratado de dotar a los dispositivos móviles de un pequeño y limitado navegador web. WAP exige la presencia de una puerta de enlace encargado de actuar cómo intermediario entre Internet y el terminal. Esta puerta de enlace o gateway es la que se encarga de convertir las peticiones WAP a peticiones web habituales y viceversa.

Las páginas que se transfieren en una petición usando WAP no están escritas en HTML, si no que están escritas en WML, un subconjunto de éste. WAP ha sido un gran avance, pero no ha resultado ser la herramienta que se prometía. La navegación es muy engorrosa (la introducción de URLs largas por teclado es muy pesada, además de que cualquier error en su introducción requiere que se vuelva a escribir la dirección completa por el teclado del móvil). Además su coste es bastante elevado ya que el pago de uso de esta tecnología se realiza en base al tiempo de conexión a una velocidad, que no es digamos, muy buena.

Otra tecnología relacionada con los móviles es SMS. SMS son las siglas de Short Message System (Sistema de Mensajes Cortos). Actualmente este sistema nos permite comunicarnos de una manera rápida y barata con quien queramos sin tener que establecer una comunicación con el receptor del mensaje. Con ayuda de J2ME, sin embargo, podemos realizar aplicaciones de chat o mensajería instantánea.

Los avances tecnológicos de la telefonía móvil, han pasados por varias generaciones como las conocidas 2G (GSM), 2.5G (GPRS), 2.75G (EDGE), 3G (UMTS), 3.75G(HSUPA) y la 4G denominada HSPA. La tecnología 4G nos permitirá navegar más rápido a menor latencia, esto significa, que el tiempo que tarda los datos en llegar desde un lugar a otro será mas rápido, esto permitirá una tasa de envíos más alta de hasta **50 Megabits por segundo**, siendo **25 veces más veloz** que la tecnología actual **3G**.

Otras tecnologías que favorecen la comunicación son Bluetooth y las redes inalámbricas que dan conectividad a ordenadores, PDAs y teléfonos móviles. De hecho, una gran variedad de estos dispositivos disponen de soporte bluetooth. Esto nos facilita la creación de redes con un elevado ancho de banda en distancias pequeñas (hasta 100 metros). Es de esperar que todas estas tecnologías favorezcan el uso de J2ME en el mercado de la telefonía móvil.

CAPÍTULO III

METODOLOGÍA

3.1 Introducción

En este capítulo se presentará un resumen de cómo se ha realizado todo el proceso de investigación para el desarrollo de esta tesis, dentro del contenido se describirá la profundidad del estudio, el tipo de investigación, el diseño de la investigación, y también los métodos y técnicas que se han utilizado para la recolección de información y datos.

3.2 Nivel de la investigación

Cuando se habla de nivel de investigación se refiere al grado de profundidad con que se trata un objeto o fenómeno. Aquí se indicará si se trata de una investigación exploratoria, descriptiva o explicativa.

Se ha determinado que el tipo de investigación es Exploratoria puesto que en la zona de Santo Domingo de los Tsáchilas la implementación de sistemas SCADAS en los procesos industriales es poco utilizado y mucho menos usando el tipo de tecnología que implemento en el desarrollo de este trabajo.

3.3 Tipo de investigación

Para determinar cual es el tipo de investigación que se ha realizado en el desarrollo de esta tesis se realizó un análisis desde distintos enfoques de la investigación, siendo así a continuación se presentan los tipos de investigación desde varios enfoques:

3.3.1 Por el grado de abstracción

Por el grado de abstracción la investigación se puede determinar que es **Aplicada** ya que en este tipo de investigación su principal objetivo es resolver problemas prácticos, con un margen de generalización limitado.

3.3.2 Por la naturaleza de los datos

De acuerdo con este enfoque, se determina que es una investigación de tipo **Cualitativa** puesto que este tipo de investigación se basa en el análisis subjetivo e individual, lo cual la hace una investigación interpretativa, referida a lo particular.

3.3.3 Por la dimensión cronológica

De acuerdo con la dimensión cronológica la investigación realizada en esta tesis es de tipo **Experimental** debido a que se está realizando algo que en la actualidad no existe

en esta zona del país, y la investigación experimental predice lo que sucederá si se produce alguna modificación en la condición actual de un objeto o hecho.

3.3.4 Por el tipo de fuentes

El presente trabajo es netamente una investigación **Bibliográfica** porque se ha recurrido a la indagación bibliográfica sobre el tema que se está tratando. Este tipo de investigación tiene como principal característica la búsqueda, recopilación, organización, valoración de información bibliográfica sobre un tema específico tomando en cuenta que las citas bibliográficas deben hacerse presentes en el trabajo para dar conocimiento de las fuentes de información que se han utilizado.

3.4 Diseño de investigación

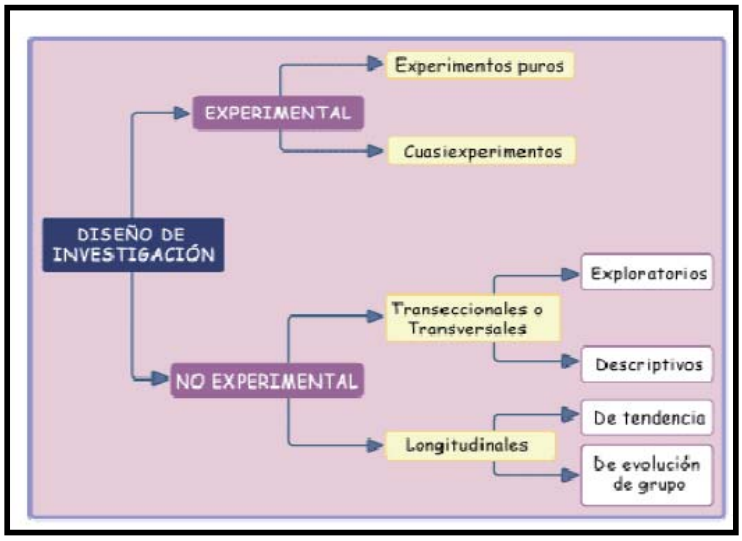
3.4.2. Tipos de diseño

"Con el fin de recolectar la información necesaria para responder a las preguntas de investigación (bien sea cualitativa o cuantitativa), el investigador debe seleccionar un diseño de investigación. Esto se refiere a la manera práctica y precisa que el investigador adopta para cumplir con los objetivos de su estudio, ya que el diseño de investigación indica los pasos a seguir para alcanzar dichos objetivos"[17].

[17] www.rena.edu.ve/cuartaEtapa/metodologia/tema5.html

La precisión, la profundidad, así como también el éxito de los resultados de la investigación dependen de la elección adecuada del diseño de investigación. He aquí un esquema donde se resumen los diferentes tipos de investigación.

Figura 3.1: Diseño de investigación



Fuente: Rena. "Diseño de investigación".Copyright 2008
Elaborado por: César Zambrano / 2011

Debido a que en la aplicación de esta tesis se determinó que el diseño de investigación utilizado es el **no Experimental** por lo tanto describimos los siguientes:

Transversal: Es el diseño de investigación que recolecta datos de un solo momento y en un tiempo único. El propósito de este método es describir variables y analizar su incidencia e interrelación en un momento dado. Este tipo de investigación puede ser: Exploratoria o Descriptiva.

Longitudinal: Es el diseño de investigación que recolecta datos a través del tiempo en puntos o períodos especificados, para hacer inferencias respecto al cambio, sus determinantes y consecuencias. Este tipo de investigación se divide en: de Tendencia y de Evolución de Grupo.

Puesto que el propósito del diseño no-experimental, transversal exploratorio, es explorar una comunidad, un contexto, una situación, una variable, un proceso o un fenómeno en un momento específico y que además el tema a tratarse no sea muy difundido es decir sea poco estudiado anteriormente. Además tomando en cuenta que el diseño exploratorio por lo general es muy usado dentro del enfoque Cualitativo, la investigación que se esta realizando en esta tesis tomará un diseño **No-experimental,**

Transversal Exploratorio, porque el uso de los programas usados para la realización de la tesis son muy pocos utilizados en la parte industrial.

3.5. Técnicas de investigación

En este apartado se va a establecer un listado de todas las técnicas que se ha utilizado en la realización de este trabajo de tesis.

3.5.1. La observación

A esta técnica siempre se la debe considerar como el punto de partida de toda investigación científica, por lo tanto en esta investigación se ha realizado la observación detenida del proceso industrial que se va a simular. Además se ha realizado la

observación muy minuciosa de los detalles que se van a presentar en el dispositivo móvil.

3.5.2 La revisión bibliográfica

Esta técnica ha sido la principal herramienta usada para la recolección de información en esta tesis puesto que se ha revisado la información de la ayuda de los programas informáticos que se utilizaron, libros de automatización que han sido de mucha ayuda, se ha revisado información proporcionada por profesionales en sistemas informáticos y principalmente se ha realizado una gran investigación con la ayuda del Internet.

3.6. Fuentes de datos

3.6.1. Fuentes primarias

En el presente estudio se obtendrán datos primarios cuando se realice las consultas a profesionales que hallan manejados los programas que se utilizaron y esto será de mucha ayuda ya que así se podrá configurar de manera más rápida dichos programas.

3.6.2. Fuentes secundarias

Las fuentes de información que se ha utilizado en esta tesis han sido las siguientes:

- ✓ Manuales de usuario de los programas Industrial SQL y Active Factory de la empresa norteamericana Wonderware, esta documentación esta incorporada en los programas de instalación de cada software.

- ✓ Libros recomendados y proporcionados por profesionales a los cuales se les ha realizado consultas.
- ✓ Amplia recolección de información descargada desde el Internet

CAPÍTULO IV

DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE SUPERVISIÓN Y CONTROL DE UN PROCESO INDUSTRIAL DIDÁCTICO DESDE UN DISPOSITIVO MÓVIL.

4.1 Análisis De Los Elementos Necesarios Para La Implementación Del Sistema

El sistema de Supervisión y Control de un Proceso Industrial Didáctico desde un Dispositivo Móvil, debe poseer los elementos necesarios para que de esa forma que brinde los siguientes beneficios:

- ✓ Adquisición continúa de la información existente en la línea del proceso.
- ✓ Confiabilidad de los datos almacenados
- ✓ Disponibilidad de la información almacenada en la base de datos, para las personas autorizadas a acceder a la misma.
- ✓ Facilidad para manejar el proceso desde cualquier lugar desde un móvil

Los elementos más importantes involucrados en el sistema a desarrollarse en el presente proyecto se describen a continuación:

4.1.1 Usuarios

Los usuarios serán aquellas personas que requieran tener acceso a la información del proceso que se va a simular, estos usuarios pueden corresponder a dos áreas.

4.1.1.1 Usuarios de la red Administrativa

Generalmente serán Analistas de Calidad, supervisores y gerentes o dueños de la empresa, quienes tienen asignado un usuario y contraseña que les permite ingresar y únicamente supervisar el proceso y visualizar la información.

4.1.1.2 Usuarios de la red Industrial

Serán Ingenieros de planta y programadores, de igual manera se les asignará un usuario y contraseña que les permite supervisar y controlar el proceso, además hacer modificaciones si es necesario.

4.1.2 Hardware

Aquí se detallan todos los elementos físicos que son necesarios para que los Operadores, Supervisores, Personal de Mantenimiento e Ingenieros de Planta puedan acceder a la información.

4.1.2.1 Estación de trabajo

Se va a considerar una estación de trabajo a la computadora principal o servidor, donde va a estar toda la información del proceso.

4.1.2.2 PLCs

Los PLCs además de controlar el funcionamiento de un proceso, proporcionan información del estado de las variables y eventos inherentes al proceso de Producción, además de las acciones realizadas por el Operador.

4.1.2.3 Red Industrial

La red Industrial permite la comunicación entre las estaciones de trabajo y la Base de Datos a través de un servidor WEB y constituyen el medio de transmisión de datos desde los PLCs.

4.1.2.4 Data Base Server

"El Servidor de Base de Datos (Data Base Server) almacenará toda la información ingresada por los operadores o adquirida desde los PLC, este equipo debe ser capaz de almacenar grandes cantidades de datos."[18]

---

[18] http://www.infomanuales.com/Manuales/SQLServer/SQLServer.asp

4.1.2.5 Web Server

El Web Server gestiona el acceso de todos los usuarios que desean acceder a la información del Data Base Server a través de páginas Web, ejecutadas mediante Internet Information Server.

4.1.3 Software

4.1.3.1 Programación de PLCs (TWIDO)

TWIDO es un Software de programación de los PLCs que permite editar y modificar el programa de control y configurar las variables de funcionamiento de cada máquina. "TwidoSoft le permite crear programas con distintos tipos de lenguaje"[19]

4.1.3.2 Base de Dato Industrial

El software Industrial InSQL Server es utilizado para recolectar y almacenar la información proveniente de los PLCs. Adicionalmente desde el programa SQL Server se puede acceder a la información almacenada en InSQL.

4.1.3.3 SQL Server

SQL Server es un DBMS comercial que sirve para guardar y gestionar la información almacenada en una base de datos. Permite además realizar tareas de búsqueda, programar ciertos cálculos u operaciones y asignar seguridades de acceso.

---

[19] http://ww2.estg.ipleiria.pt/~ftadeu/TwidoSoft_V20/Documentacao/spa/TwdoSW.pdf

4.1.3.4 Windows Server

Windows Server es el Sistema Operativo instalado en los servidores con las características especiales para el manejo de información y la administración de redes y usuarios.

4.2 Creación del sistema SCADA para el proceso a simular que es un proceso de extracción de jugo de manzana

En esta parte del capítulo se va a describir como se han realizado la implementación de un sistema SCADA para el proceso que se va a simular, el esquema de un sistema SCADA está conformado de tres partes claramente definidas, las cuales son:

Elementos de campo.- Son los elementos que permiten la conversión de una señal física en una señal eléctrica (y viceversa). Por ejemplo sensores, actuadores, electroválvulas, cámaras, etc. En este proyecto no se van a utilizar ninguno de ellos ya que solo se va a simular el proceso.

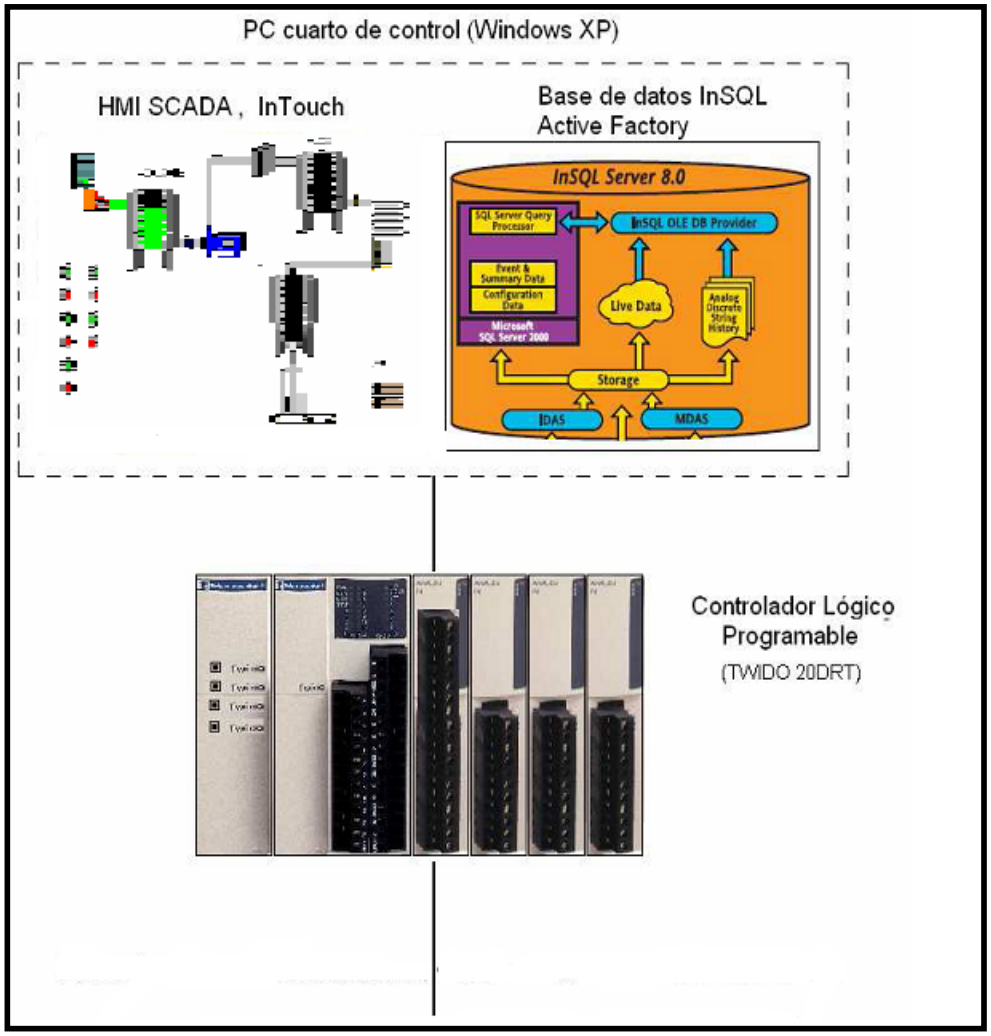
Unidad de Control.- Es la encargada de realizar el control del proceso mediante una programación establecida, esta unidad puede ser un Controlador Lógico Programable (PLC).

Computador Central.- En esta unidad se tiene la interface de usuario o HMI, además aquí se implementará la base de datos para el almacenamiento de los parámetros más importantes.

Por lo tanto este apartado se va a dividir dos grupos donde se presentará con detalle como se realiza la instalación del controlador "PLC" con su respectiva programación y la descripción de la creación de una pantalla HMI.

4.2.1 Esquema del sistema SCADA que se va a implementar

Figura 4.1: Esquema de SCADA



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

4.2.2 Unidad de Control (PLC)

Para el proceso que se va a simular se ha seleccionado que la unidad la unidad de control sea un controlador lógico programable o PLC, específicamente el modelo Twido TWDLMDA20DRT el cual consta de una base modular que posee un rack de 12 entradas digitales, 6 salidas vía relé de 2 Amperios y 2 salidas vía transistor de 0.3 amperios y la tensión de alimentación es de 24 voltios de corriente continua.

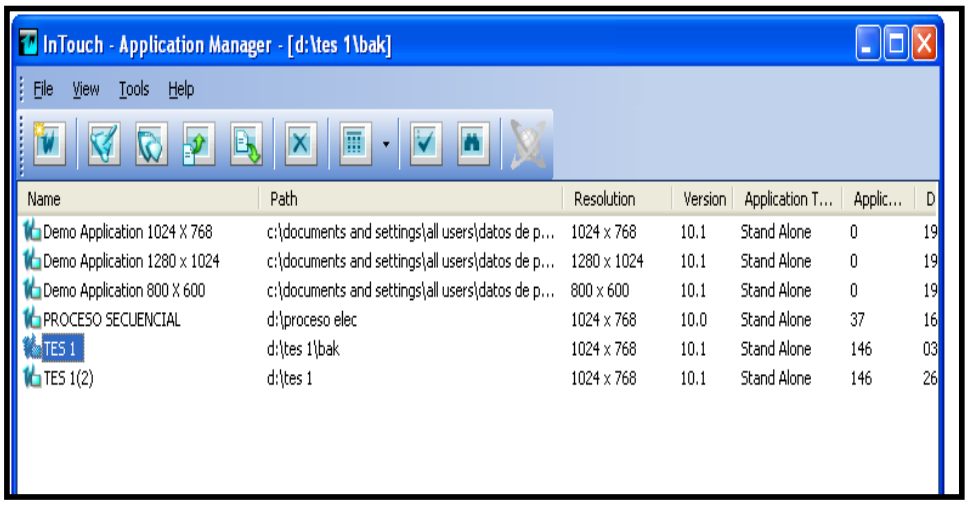
La programación del PLC es muy sencilla ya que es un proceso simple el que se va a simular y todas las señales de entradas y salidas están atadas a memorias internas del PLC para que puedan ser entrelazadas con la pantalla HMI. Dicha programación esta en el ANEXO 2, donde se presenta todo el circuito en lenguaje Laader para el proceso que se va a simular.

4.2.3 Creación de la pantalla HMI

Para la creación de la pantalla HMI se ha utilizado el software InTouch el cual también pertenece a la empresa americana Wonderware, todas las características que se realizan en el proceso de creación de la pantalla HMI tienen que estar en concordancia con la programación que se ha cargado en el controlador (PLC), por lo tanto a continuación se van a presentar los pasos que se siguieron durante la creación de la HMI para la simulación del proceso.

- ✓ Se crea una nueva aplicación dentro del programa InTouch, es este caso el nombre que se le ha dado a la aplicación es “TES 1” y esta ubicada en una carpeta dentro del disco D:\ de la computadora.

Figura 4.2: Nueva aplicación de Intouch



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

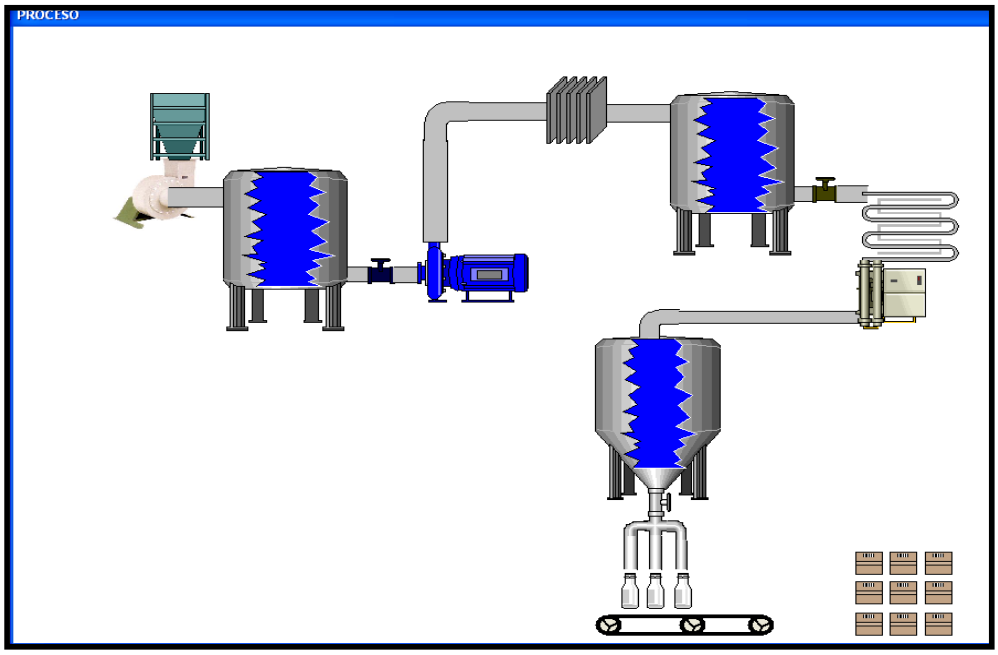
- ✓ Una vez creada la aplicación se inicia el programa donde se han creado diferentes ventana las cuales van a ser presentadas en los siguientes pasos en conjunto de una breve explicación de cada una de ellas.

- ✓ Para esta aplicación se crearán solo dos pantallas, la primera que es la de presentación y la segunda donde se va a visualizar el proceso de extracción de jugo siendo así un HMI muy básico que a su vez esa pantalla se va poder visualizar en el celular una vez realizada la aplicación.

- ✓ La primera pantalla es únicamente de presentación y tiene un botón para dirigirse a la pantalla del proceso.

✓ En la ventana “PROCESO” se visualiza todo el proceso gráficamente, así como también los valores numéricos de los niveles de líquido que existe en los tanques y la cantidad de producto terminado y además existen los botones de control.

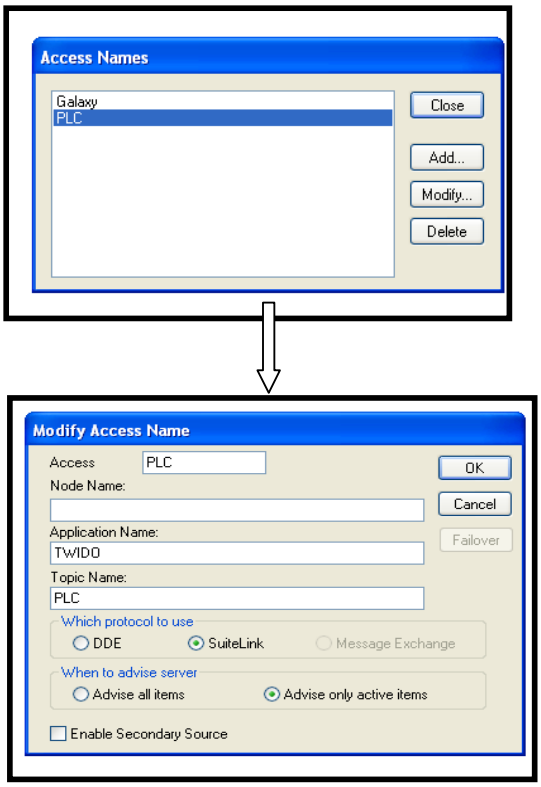
Figura 4.3 Ventana del proceso



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ Una vez creadas las pantallas o ventanas se debe crear el Acces Name el cual sirve de enlace para establecer la comunicación entre la pantalla HMI y el PLC, como se había mencionado, la comunicación se la realizará bajo estándares del protocolo MODBUS, para lo cual es necesario instalar el driver de comunicación el cual se encuentra en el paquete de Wonderware I/O Server para la comunicación con el PLC TWIDO, en la siguiente figura se muestra la configuración del Access Name para ingresar a estas ventana se da clic en la pestaña **Special** y luego en **Access Name.**

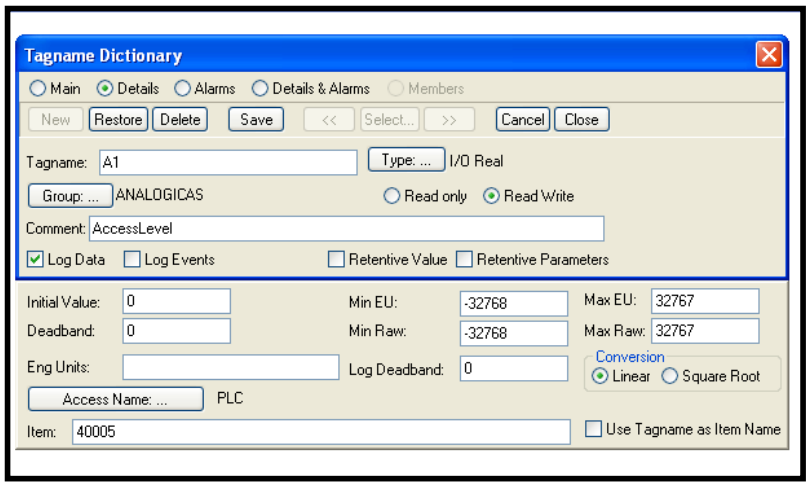
Figura 4.4: Configuración del Access Name



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

- ✓ Una vez configurado Access Name ya se pueden crear la variables dentro del programa de la siguiente forma: se ingresa en la pestaña **Special**, luego se da clic el **Tagname Dictionary** y aparece el cuadro de diálogo de las variables que tiene el sistema en la parte superior izquierda se tiene un botón para crear una variable nueva a la cual se le puede dar diferentes características como por ejemplo: el nombre de la variable, el tipo de variable, el grupo al que pertenecen, límites, valor inicial etc.

Figura 4.5: Creación de variables

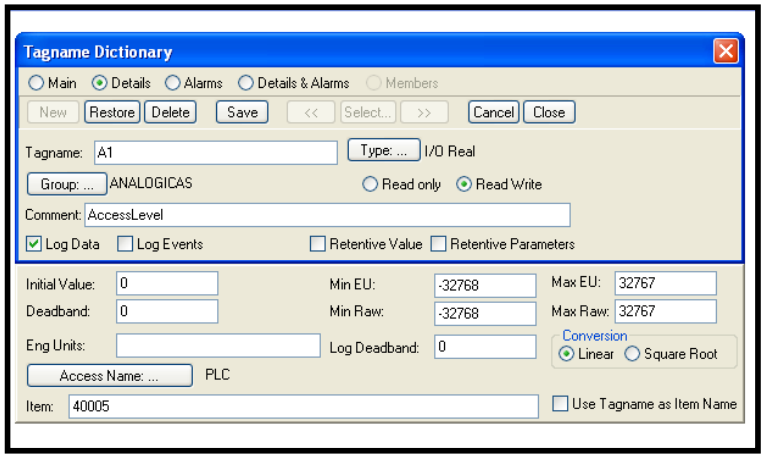


Fuente: César Zambrano / 2010

Elaborado por: César Zambrano / 2010

✓ En el siguiente paso se realizará la selección de las variables que van a ser almacenadas dentro de la base de datos, para dichas variables es necesario que en su configuración se seleccione la casilla de verificación de **LogData** la cual se especifica en la siguiente figura

Figura: 4.6 Selección Log Data



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

- ✓ Cuando todas las variables están creadas deben ser atadas a los objetos que se tengan dentro de las diferentes ventanas para que las animaciones representen cada uno de los parámetros del proceso, la forma de atar las variables es anotando el nombre de cada una en el objeto que corresponde. Todas las variables inmersas en el proceso con sus respectivas características se presentaran a continuación.

4.3 Selección de las variables que van a ser almacenadas en la base de datos

En esta sección se van a presentar todas y cada una de las variables que están inmersas dentro del proceso que se va a simular, dichas variables serán presentadas en una tabla donde se muestran sus principales características entre las cuales se pueden mencionar su símbolo, una breve descripción, las direcciones en el PLC y las correspondientes en la HMI, las memorias internas a las que están atadas, las direcciones MODBUS, etc.

4.3.1 Listado del direccionamiento de todas las variables

En la siguiente tabla se presentan todas las variables del proceso que se va a simular con sus respectivas características:

Tabla 4.1: Variables

SÍMBOLO	DESCRIPCIÓN	DIR. PLC	MEMORIAS	TAGNAMES	MODBUS
A1	LITROS DE JUGO-FRESCO	%IW2.0	%MW0	A1	40001
A2	LITROS DE JUGO-PASTEURIZADO	%IW2.1	%MW1	A2	40002
A3	LITROS DE JUGO-LISTO PARA ENVASAR	%IW2.2	%MW2	A3	40003
A6	NUMERO DE CAJAS DE PRODUCTO TERMINADO	%IW2.3	%MW3	A6	40004
D1	INICIO DE PROCESO	%Q0.0	%M1	D1	2

Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

4.4 Creación de la base de datos utilizando el software IndustrialSQL

4.4.1. Introducción

Para la creación de una base de datos utilizando el software IndustrialSQL se deben tener en cuenta algunas consideraciones y recomendaciones antes de proceder con la instalación del programa, dichas consideraciones se presentan a continuación:

- ✓ Debido a que Industrial SQL Server adquiere datos a muy alta velocidad reduciendo su volumen simultáneamente como extensión de Microsoft SQL Server, la primera consideración a tomar en cuenta es que el requisito importante para la instalación del software Industrial SQL es que Microsoft SQL Server esté instalado y funcionando en la máquina donde correrá la base de datos.
- ✓ Se debe considerar que por lo general una base de datos debe ser instalada en un equipo configurado como "Servidor" pero se determinó que para el proceso que se va a simular no es justificable realizar la adquisición de un servidor debido a su elevado costo, por lo tanto se ha decidido que la base de datos va a ser instalada en una computadora configurada bajo un sistema operativo Windows XP Profesional Service Pack 3.

- ✓ Es importante mencionar que el software Industrial SQL crea la base de datos al mismo tiempo de su instalación, es decir que únicamente se debe realizar la instalación del programa y automáticamente la base de datos estará creada y lista para funcionar.

- ✓ Otro de los pre-requisitos para la instalación de la base de datos es tener creada una cuenta de usuario de Windows que tenga privilegios de administrador, este paso se lo debe realizar porque el sistema de seguridad de la base de datos utiliza la misma cuenta de Windows para permitir en acceso a la gestión de los parámetros.

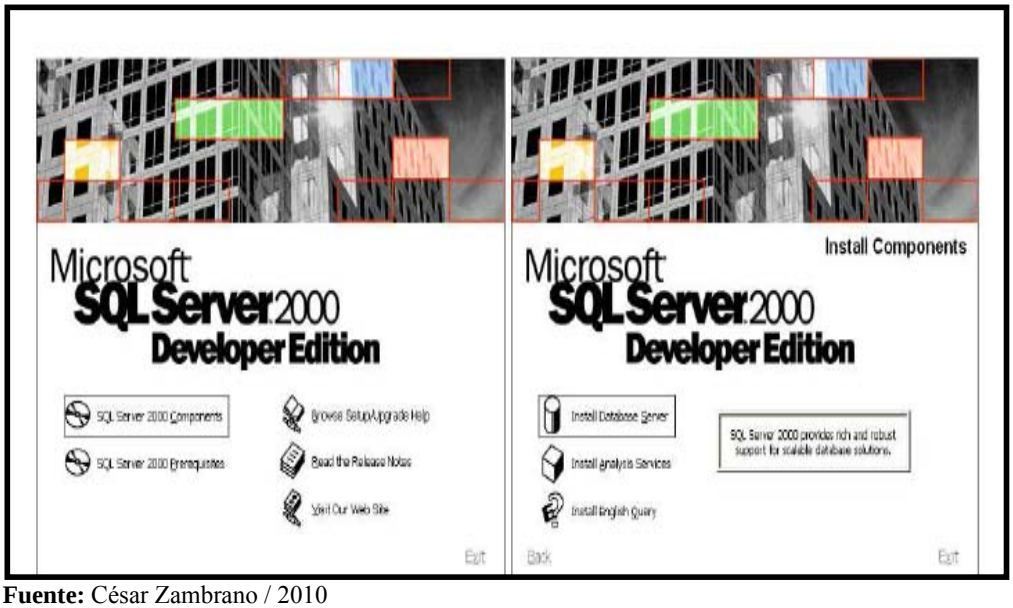
Si se toman en cuenta todas las recomendaciones anteriores ya se puede realizar la instalación de los programas sin tener ninguna complicación, a continuación se presentan la instalación de Microsoft SQL Server, de Industrial SQL y las configuraciones necesarias para que la base de datos industrial funcione correctamente.

4.4.2. Instalación Microsoft SQL Server

En este apartado se presenta paso a paso como se realizó la instalación del software Microsoft SQL Server Developer Edition para la base de datos.

- ✓ Cuando se introduce el CD de instalación del Microsoft SQL Server y se procede a la instalación del programa entrando en la opción de **Setup** aparece la primera ventana de la instalación en la cual se muestra la versión que se está por instalar, en esta ventada se debe seleccionar la opción **SQL Server 2000 Components** y aparece la segunda ventana de la instalación donde se debe dar clic en **Install Databases Server**, las dos primeras ventana se muestran en la siguiente figura.

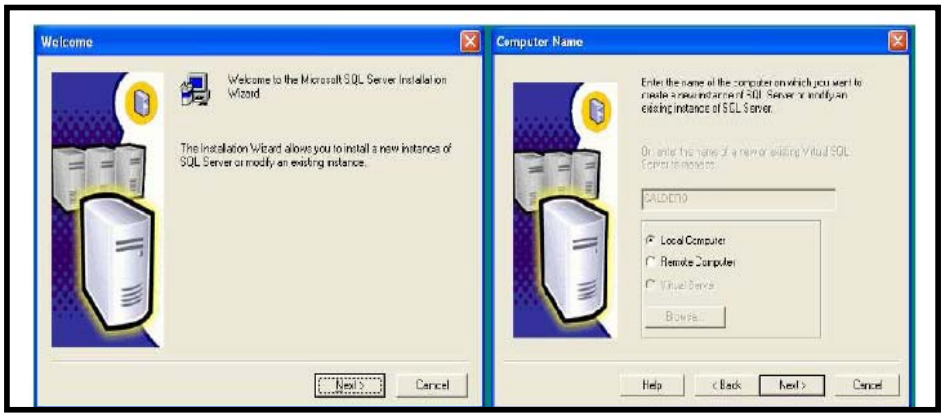
Figura 4.7: Paso 1 y 2 de Instalación de Microsoft SQL Server



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

- ✓ Una vez que se está dentro del asistente para la instalación del software se presenta la ventana de bienvenida al programa de instalación, se da clic en **Next** para continuar a la siguiente ventana en la que se debe seleccionar la computadora donde se pretende instalar el programa en este caso será en la misma máquina así que la opción que se debe seleccionar es la de **Local Computer** y se da clic en **Next**.

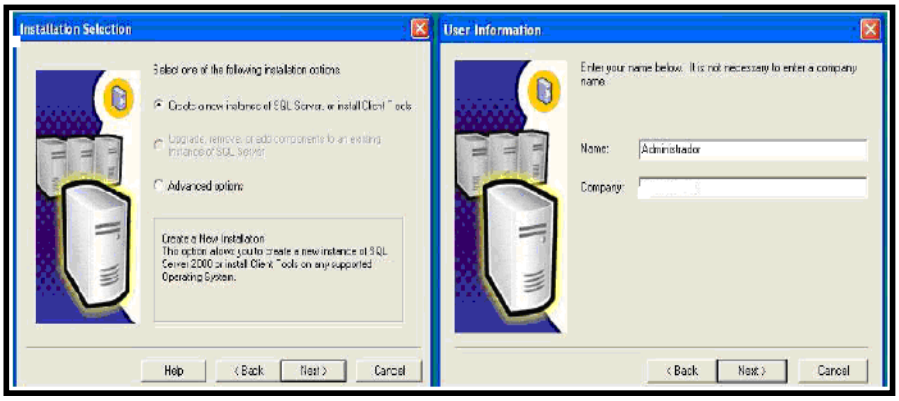
Figura 4.8: Paso 3 y 4 de Instalación de Microsoft SQL Server



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

- ✓ En la siguiente ventana de la instalación se selecciona las opciones de instalación donde se ha seleccionado la opción de **Create a new instance of SQL Server, or install Client Tools,** dar clic en **Next** y aparece la siguiente ventana para especificar el nombre del usuario y el nombre de la compañía, una vez llenado los datos se da clic en **Next**.

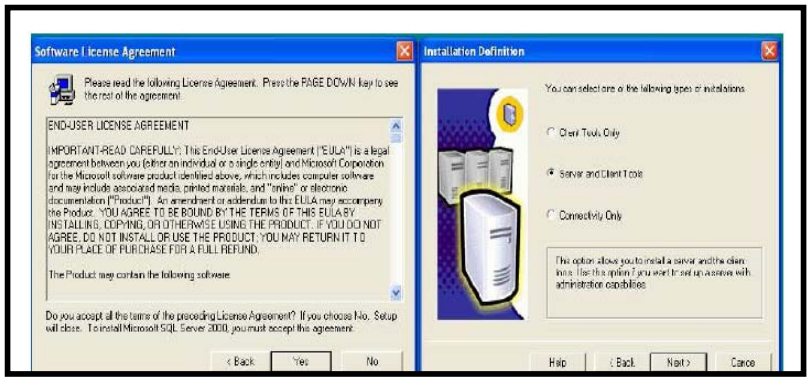
Figura 4.9: Paso 5 y 6 de Instalación de Microsoft SQL Server



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

- ✓ A continuación aparece una ventana que muestra el acuerdo de licencia del programa el cual se lo acepta dando clic en **Yes** y se sigue a la siguiente ventana que muestra los tipos de instalación que se pueden realizar, se selecciona la opción **Server and Client Tools** y se da clic en **Next**.

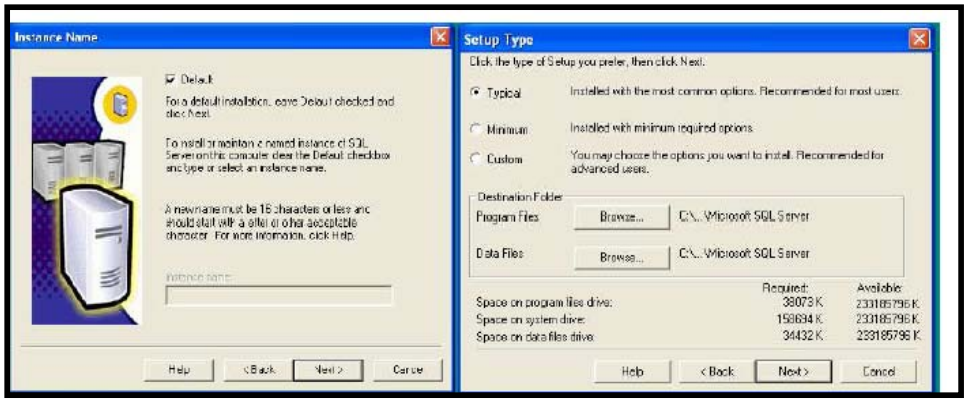
Figura 4.10: Paso 7 y 8 de Instalación de Microsoft SQL Server



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ En la siguientes dos ventanas se muestran el nombre de la instancia que se va a crear y el tipo de instalación en estos pasos se han seleccionado la opciones que el asistente ha establecido como predeterminados y se da clic en **Next**.

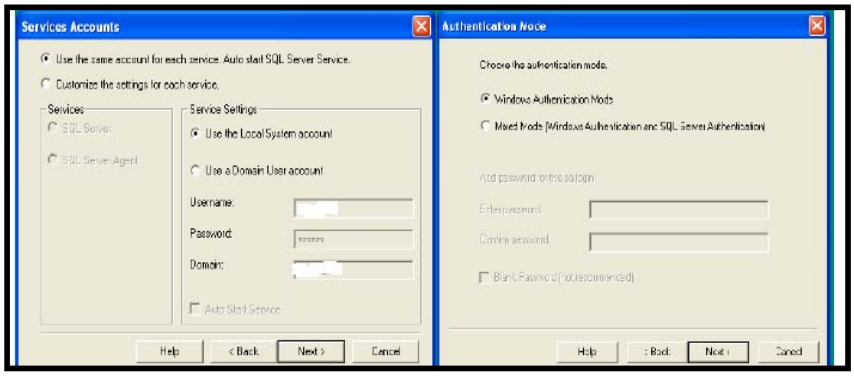
Figura 4.11 Paso 9 y 10 de Instalación de Microsoft SQL Server



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ En los siguiente dos paso se presentan opciones para la creación de cuentas de usuarios de la base de datos, se ha seleccionado la opción que utiliza la cuenta de usuario de Windows para acceso a la base de datos y para el modo de autenticación, asi que en la primera ventana se selecciona **Use the Local System account** y en la segunda **Windows Authentication Mode**, dar clic en **Next**.

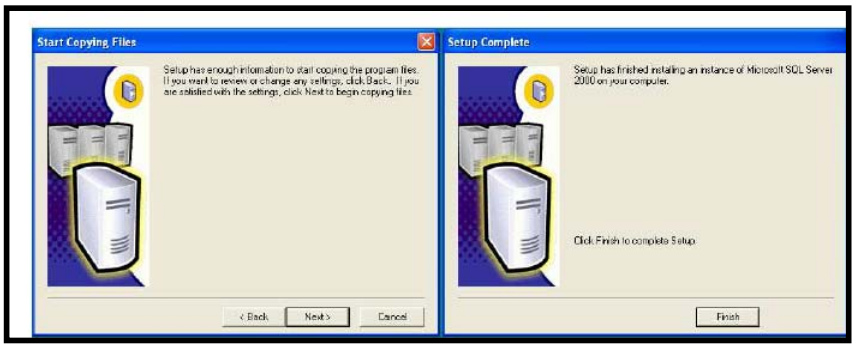
Figura 4.12: Paso 11 y 12 de Instalación de Microsoft SQL Server



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

- ✓ En la ventana siguiente se presenta un mensaje de que toda la información está lista para la instalación al dar clic en **Next** la instalación arranca, al finalizar el proceso aparece una ventana de confirmación de que la instalación ha finalizado con éxito se da clic en **Finish.**

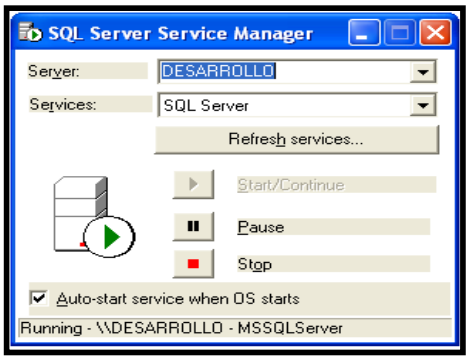
Figura 4.13: Paso 13 y 14 de Instalación de Microsoft SQL Server



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

Con el Microsoft SQL Server instalado en la máquina se debe comprobar que esté funcionando en el sistema accediendo a **Inicio**, **Todos los programas**, **Microsoft SQL Server** y dar clic en **Service Manager** aparece una ventana del administrador de funcionamiento, comprobar que el estado esté en **Runing**.

Figura 4.14: SQL Server Service Manager

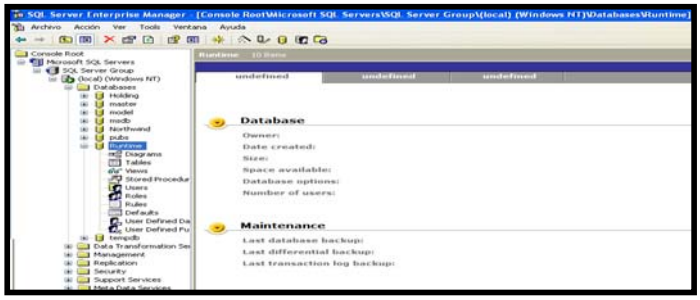


Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

4.4.2.1 Configuración de Microsoft SQL

Una vez que ya están todos los softwares instalados y listos para funcionar, al ejecutar Microsoft SQL aparece en su consola principal en la parte izquierda un árbol desplegable donde vamos a encontrar la base de datos que se creó automáticamente al instalar InSQL que se llama Runtime.

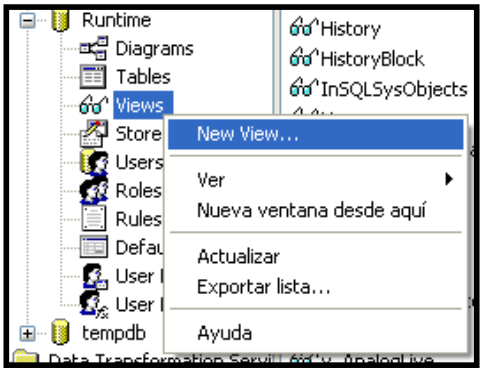
Figura 4.15: Consola de Microsoft SQL



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

Desplegamos la base de datos Runtime donde seleccionamos las vistas (Views), y damos clic derecho para seleccionar la opción vista nueva (New Views), para así crear una nueva vista donde se van a presentar las variables únicamente que se necesitan.

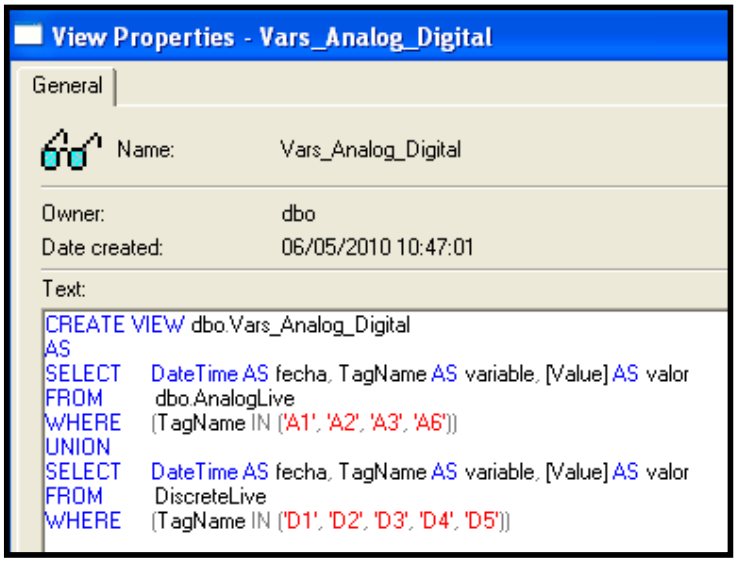
Figura 4.16: Crear nueva vista



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

Inmediatamente se abre una ventana para dar las características de la tabla y las variables que se van a presentar en esa vista.

Figura4.17: Propiedades de vista



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

Una vez creada la nueva vista, la podemos visualizar como en la siguiente figura.

Figura 4.18: Nueva vista

fecha	variable	valor
10/05/2010 10:22:	A3	0
10/05/2010 10:22:	A6	0
10/05/2010 10:22:	D3	0
10/05/2010 10:22:	D4	0
10/05/2010 10:22:	D5	0
10/05/2010 10:22:	D1	0
10/05/2010 10:22:	D2	1
10/05/2010 10:23:	A1	390
10/05/2010 10:23:	A2	210

Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

4.4.3. Instalación de Industrial SQL

Requisitos del servidor

Los requisitos mínimos de hardware y software para el historiador IndustrialSQL Server se basan en el número de variables y el tipo de datos de rendimiento esperado. Estos requisitos se dividen en tres niveles, en este proyecto describimos el que vamos a necesitar.

La configuración de memoria recomendada para SQL Server 2000 es sujetar su consumo de memoria en un 50% de la cantidad de memoria física instalada en el servidor o 512 MB, lo que sea menor. El valor recomendado memoria virtual de Windows es el doble de la cantidad de RAM física instalada en el servidor.

Un servidor de nivel 1.- "Un servidor de nivel 1 puede manejar una carga de alrededor de 5.000 variables. Por ejemplo, 2.600 analógicas, 2.200 discretas, 300 de cadenas de caracteres, y 20 variables I / O Server"[20]. Los requisitos mínimos son:

- P4 3.2 GHz CPU
- Cualquiera de los siguientes sistemas operativos:
- Windows 2000 Professional SP4
- Windows XP Professional SP2
- Windows 2000 Server SP4
- Windows 2000 Advanced Server SP4
- Windows Server 2003 Standard Edition
- 1 GB RAM
- GB tarjeta de interfaz de red (NIC)

[20] Industrial SQL Historian Installation Guide – pag. 8

- Microsoft SQL Server 2000 Standard Edition Service Pack 3a o Microsoft SQL Server 2000 Personal Edition Service Pack 3a, que es necesaria para Windows 2000 Professional y Windows XP Professional
- 270 MB de espacio libre en disco para instalar el historiador IndustrialSQL Server

Instalación del programa

Cuando se ingresa el CD de instalación del historiador Industrial SQL Server automáticamente el asistente de instalación inicia el proceso, caso contrario se debe ingresar a la ubicación del CD y dar doble clic en **Setup** para que el asistente arranque. Una vez iniciado el proceso aparece la primera ventana de la instalación la que muestra un mensaje de bienvenida y el nombre del software que se va a instalar, se da un clic en **Next**.

Figura 4.19: Paso 1 Instalación Industrial SQL

Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

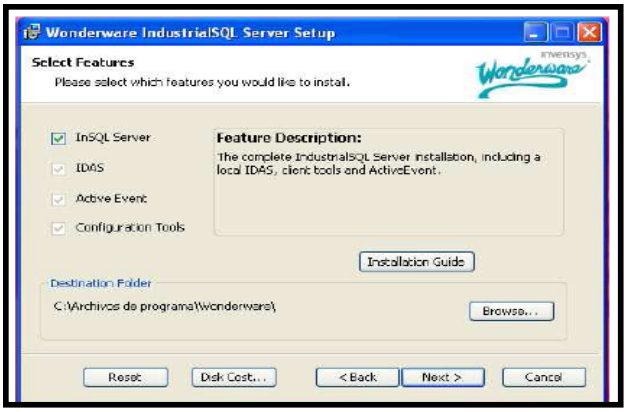
- ✓ En la segunda ventana del proceso se muestra el acuerdo de licencia del programa, se recomienda leer las cláusulas del documento y luego si aceptar dicho contrato seleccionando la casilla de verificación **I accept the License Agreement** y luego dar clic en **Next**.

Figura 4.20 Paso 2 Instalación Industrial SQL

Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

- ✓ En la siguiente ventana se debe escoger las características de las herramientas que se van a instalar, en este caso se ha seleccionado la característica de **InSQL Server** lo que hace que el resto de características se seleccionen por default puesto que ésta viene a ser la instalación más completa del software, además se puede determinar cual será la carpeta de destino del programa, a continuación se da clic en **Next**.

Fifura 4.21: Paso 3 Instalación Industrial SQL

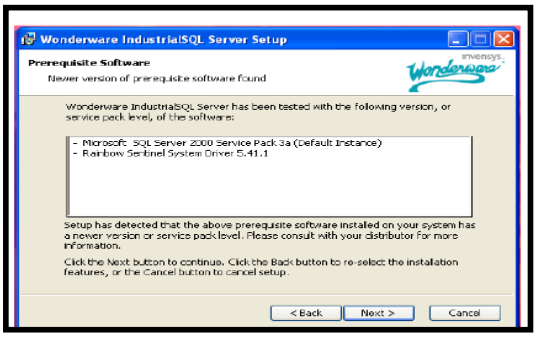


Fuente: César Zambrano / 2010

Elaborado por: César Zambrano / 2010

- ✓ Los pre-requisitos del software aparecen en la siguiente ventana, en el caso de que no estén instalados no se puede seguir con la instalación del programa pero cuando se los tiene instalados se da clic en **Next** para seguir.

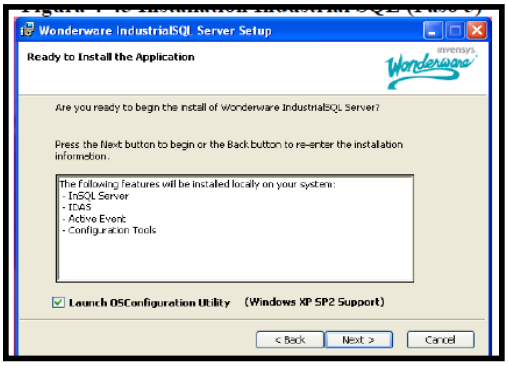
Figura 4.22: Paso 4 Instalación Industrial SQL



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

- ✓ Una vez que toda la información está lista, la siguiente ventana aparece para mostrar y verificar cuales componentes van a ser instalados, se comprueba que la casilla de verificación de **Launch OSConfiguration Utility (Windows XP SP2 Support)** esté seleccionada para que al finalizar la instalación aparezca el asistente de configuración de la base de datos que ha sido creada, se da clic en **Next**.

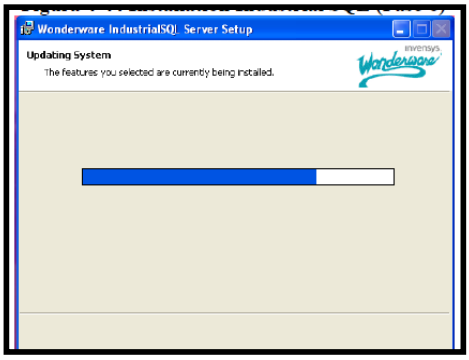
Figura 4.23: Paso 5 Instalación Industrial SQL



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ La siguiente ventana muestra el avance del proceso de instalación.

Figura 4.24: Paso 6 Instalación Industrial SQL



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

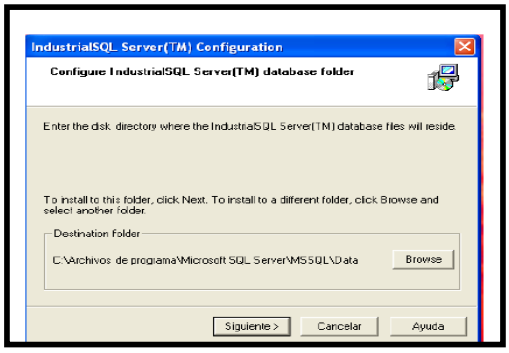
✓ Cuando la instalación del software ha sido finalizada una ventana de verificación aparece mostrando además la información de que el asistente de configuración se ejecutará cuando se cierre el programa de instalación del Industrial SQL, para terminar dar clic en **Finish**.

Figura 4.25: Paso 7 Instalación Industrial SQL

Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ Cuando el asistente la instalación de InSQL (Industrial SQL) ha finalizado aparece el asistente para configuración de la base de datos, la primera ventana permite determinar cual será la ubicación de los archivos de instalación de la base de datos del sistema, se debe verificar la dirección en el disco duro, se recomienda que la dirección sea la predeterminada, dar clic en **Siguiente**.

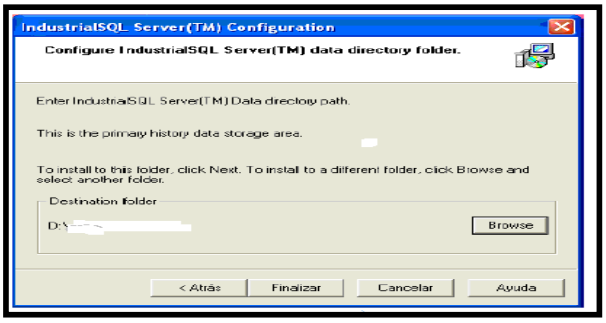
Figura 4.26: Paso 8 Instalación Industrial SQL



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ Una vez que se ha determinado donde se va a crear la base de datos se debe seleccionar la ubicación donde se van a almacenar los datos que se van a extraer del proceso, se ha seleccionado una ubicación dentro de la misma carpeta donde se ha guardado la aplicación de InTouch (pantalla HMI), una vez seleccionada la ubicación se da clic en **Finalizar**.

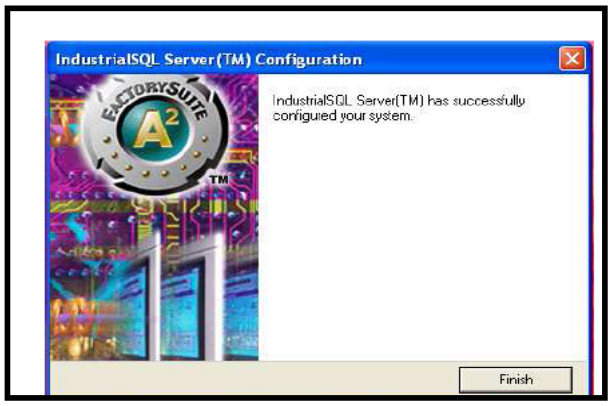
Figura 4.27: Paso 9 Instalación Industrial SQL



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

- ✓ Luego de que una ventana demuestra el desarrollo del proceso de configuración aparece la ventana final que verifica si la configuración se ha realizado con éxito. Para salir del asistente se da clic en **Finish.**

Figura 4.28 Paso 10 Instalación Industrial SQL



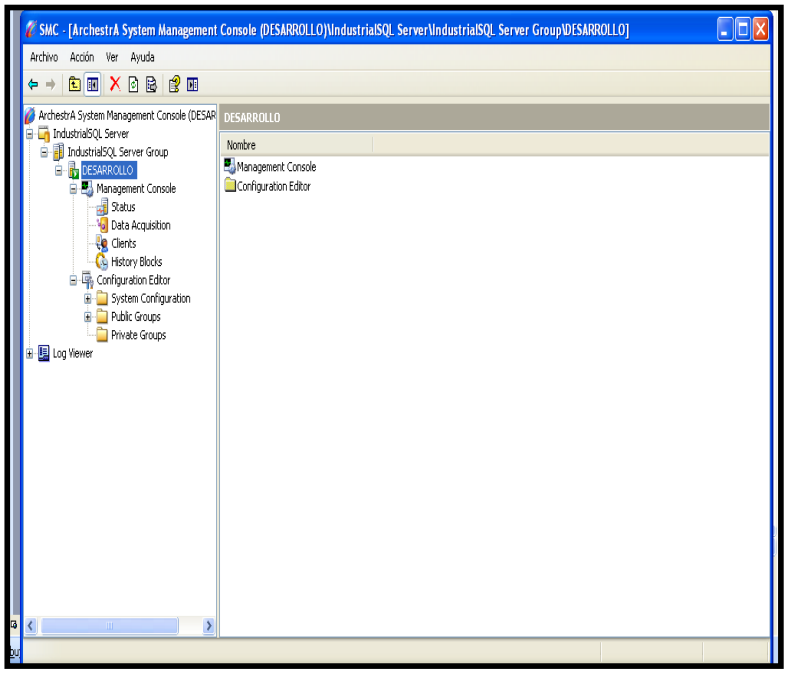
Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

Así se ha finalizado con la instalación del software Industrial SQL, la creación y configuración de la base de datos, es recomendable que se reinicie el sistema para que todos los componentes que han sido instalados corran con normal funcionamiento.

4.4.4 Configuración de la base de datos

La forma de acceder hacia la base de datos del sistema es mediante la consola de administración que es una herramienta del Industrial SQL. Para entrar a esta consola se debe ir a **Inicio**, **Todos los programas**, **Wonderware**, en la carpeta **Industrial SQL Server** dar clic en el icono **Industrial SQL Server**. En la siguiente figura se muestra la pantalla principal de la consola de administración del sistema con sus principales componentes.

Figura 4.29: Consola administración de consola



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

La parte izquierda de la pantalla de la consola de administración se tiene un navegador en donde se pueden encontrar los diferentes componentes de la consola que sirven para la configuración total de la base de datos industrial, entre estos componentes se tiene el nombre del servidor "DESARROLLO", la consola de administración (Management Console), el editor de configuración (Configuration Editor) y el Log Viewer. Para su mejor entendimiento se presenta una breve descripción de los principales elementos de la consola de administración:

Consola de administración (Management Console).- La consola de administración permite visualizar el comportamiento de la base de datos; ésta consta de cuatro componentes que son:

- ✓ **Status.** Muestra el estado de todos los componentes de la base de datos, se puede apreciar si el sistema está corriendo o está detenido, muestra que módulos están

funcionando correctamente, determina la cantidad de variables que se están almacenando, el tiempo de funcionamiento, es decir muestra un resumen general de todos los elementos de una base de datos.

- ✓ **Data Acquisition.** Permite visualizar el estado y las características de los servidores de entradas y salidas, en sí de todo el subsistema de adquisición de datos.

- ✓ **Clients.** Presenta los clientes que se están enlazando con el servidor de la base de datos.

- ✓ **History Blocks.** Este elemento muestra los bloques de datos que se van creando en el tiempo, cada vez que el sistema se inicia se crea una nueva carpeta dentro de la ubicación de la base de datos, en este caso la carpeta está en la siguiente dirección C:\InSQL\Data\Circular.

Editor de Configuración.Este componente permite modificar la configuración de los subsistemas de la base de datos, consta de tres elementos que se presentan a continuación:

- ✓ **System Configuration.** Permite visualizar y modificar los principales parámetros de la base de datos y sus subsistemas.

- ✓ **Public Groups.** Esta carpeta da el acceso a las diferentes variables que se tiene dentro de la base de datos, esto incluye a todas las variables del sistema operativo como también las variables que hayan sido importadas desde alguna aplicación HMI.

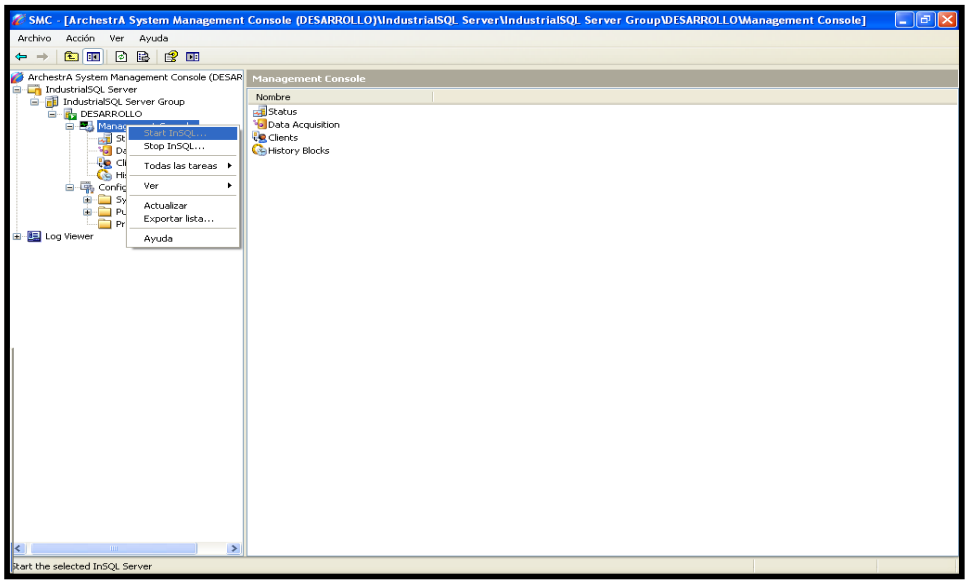
- ✓ **Private Groups.** Este elemento permite crear carpetas privadas donde se pueden ubicar las variables que se deseen, para introducirlas en una de esas carpetas basta con arrastrar las variables de una carpeta a otra.

Log Viewer.- Este componente lleva un registro de todos los sucesos que se han producido en el sistema, resaltando los errores que se hayan presentado.

4.4.5. Arrancar la base de datos

Para que la base de datos comience a funcionar completamente es necesario arrancarla para esto es necesario seguir la siguiente ruta dentro de la consola de administración: expandir la pestaña **IndustrialSQL Server**, luego expandir el servidor en este caso "**DESARROLLO**", dar clic derecho sobre **Management Console**, un pequeño cuadro de diálogo aparece y se da clic en **Start InSQL**, este proceso de muestra en la siguiente figura:

Figura 4.30: Arrancar el sistema



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

En el momento de dar clic en **Start InSQL** aparece un recuadro de la seguridad de la base de datos, en necesario tener permiso para poder realizar modificaciones en el sistema. Para tener acceso se debe ingresar los datos de la cuenta de usuario de

Windows que se había creado, recordar que la cuenta debe tener privilegios de administrador, en este caso la cuenta tiene los siguientes datos:

- ✓ Server = DESARROLLO
- ✓ Dominio = desarrollo
- ✓ Nombre de usuario = Usu1
- ✓ Contraseña = cesar

Cuando se ingresa los datos el sistema comienza el proceso de iniciación, este proceso puede tardar varios segundos, a continuación se muestra el cuadro de diálogo de seguridad:

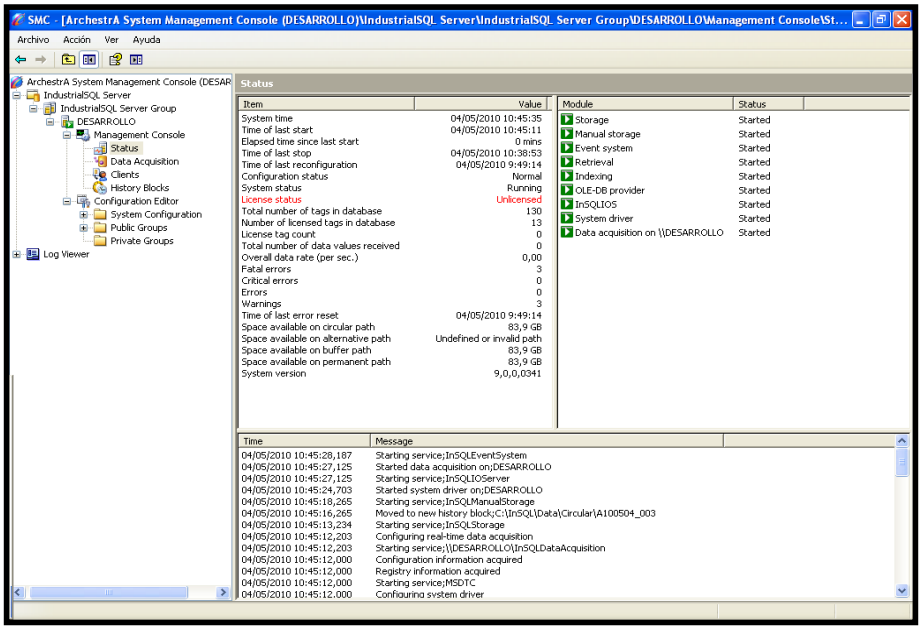
Figura 4.31: Seguridad del sistema

Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

Cuando el sistema está detenido si se ubica en la pestaña **Status** en el navegador se puede apreciar que los módulos están todos detenidos pero para verificar que el sistema está arrancando se recomienda situarse en Status para poder monitorear la inicialización

de cada uno de los módulos, a continuación se muestra en la parte derecha de la figura se puede visualizar el proceso de arranque de el InSQL.

Figura 4.32: Arrancando sistema



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

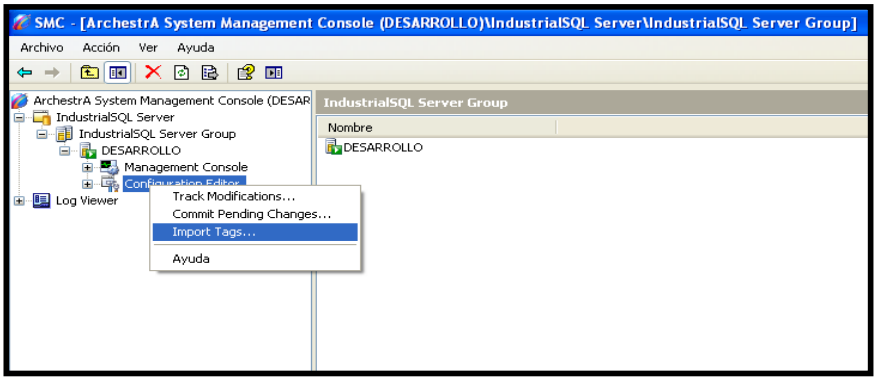
Cuando el sistema ya está corriendo está listo para importar las variables desde una aplicación de un HMI en este caso la que ha sido creada con el programa InTouch. En el siguiente paso se muestra como realizar la importación de las variables.

4.4.6. Importación de variables

Para la realización de la importación de variables hacia la base de datos se debe tomar en cuenta que dentro de la carpeta de la aplicación de InTouch se ha creado un archivo de texto llamado **tagname.x** donde están guardadas las características de todas la variables que tiene la aplicación HMI, para que Industrial SQL pueda importar este archivo se siguen pasos que se muestran a continuación:

✓ En la consola de administración se da clic derecho en cualquier parte del editor de configuración (**Configuration editor)** y luego se da clic en **Import Tags,** como se muestra en la figura.

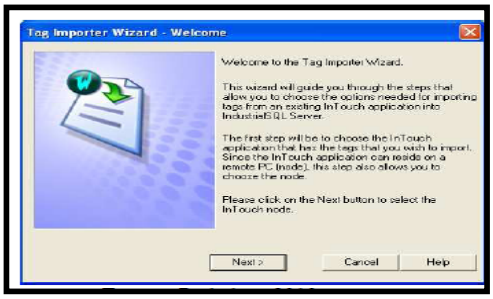
Figura 4.33: Paso 1 Importar variables



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ Una vez que se da clic en Inport Tags, aparece un cuadro de diálogo para la importación de las variables donde muestra una mensaje de bienvenida y un mensaje donde indica las funciones que se van a realizar, se da clic en **siguiente** para continuar.

Figura 4.34: Paso 2 Importar variables



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

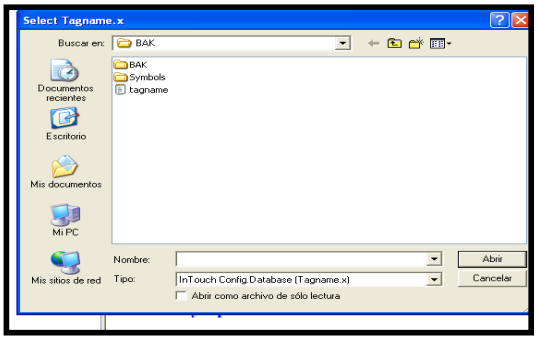
✓ En la siguiente ventana se presenta el listado de nodos de donde se han importado las variables a la base de datos, como en este caso es la primera vez que se va a realizar esta acción no aparece ningún nodo en la lista por lo tanto se debe dar clic en **Add** para agregar un nuevo nodo de importación

Figura 4.35: Paso 3 Importar variables

Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ Inmediatamente aparece el cuadro de diálogo para realizar la búsqueda de un archivo de Windows, el archivo que se debe buscar es **tagname.x** que está dentro de la carpeta de la aplicación de InTouch, se selecciona el archivo y da clic en **Abrir** para continuar.

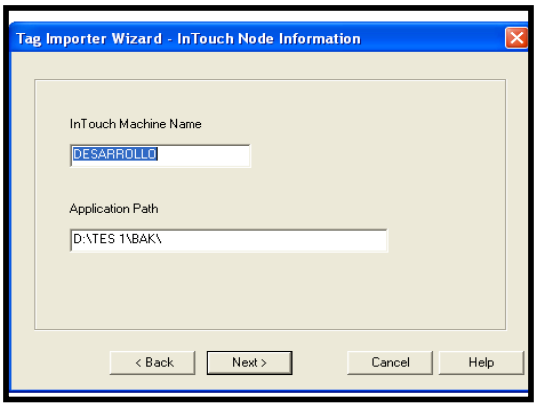
Figura 4.36: Paso 4 Importar variables



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ En la siguiente ventana se muestra el nombre del nodo de InTouch que es el nombre de la máquina donde está funcionando la aplicación HMI y la dirección del archivo tagname.x, para seguir se da clic en **Next**.

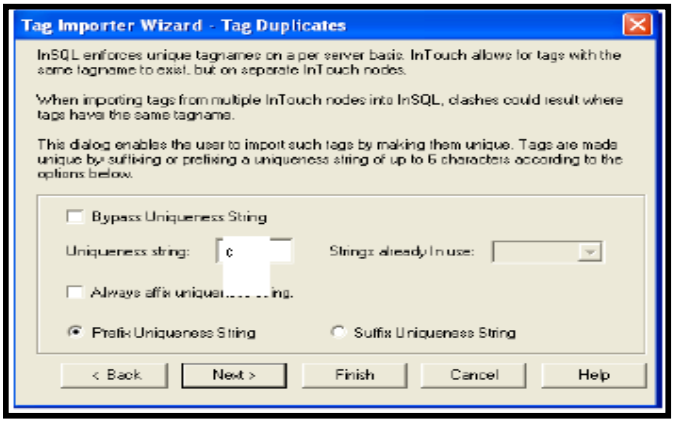
Figura 4.37: Paso 5 Importar variables



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ En la siguiente ventana se debe agregar un pequeño texto como prefijo para evitar problemas con la duplicación de variables que tengan el mismo nombre, se escribe el texto y de da clic en **Next**.

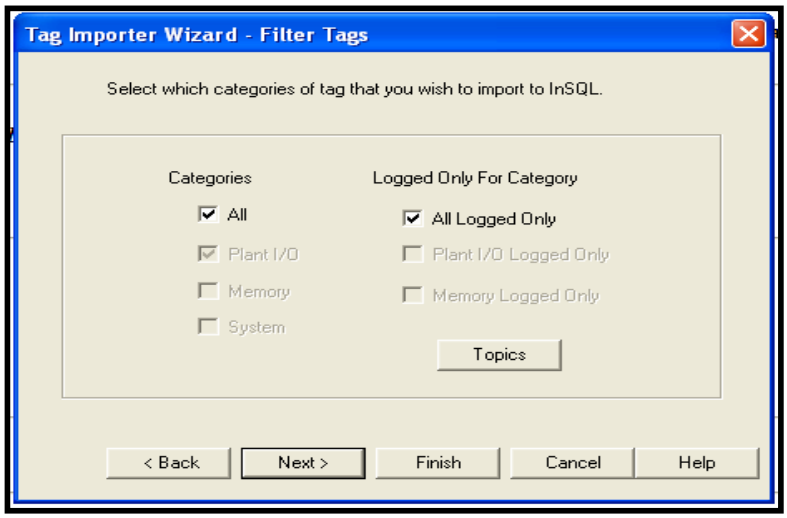
Figura 4.38: Paso 6 Importar variables



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ A continuación aparece una ventana para filtrar las variables que se van a importar utilizando como criterio la categoría de las variables. Como se desea importar todo tipo de variable se selecciona la opción **All** y se verifica que todas la variables van a ser almacenadas dentro de la base de datos, para continuar se da clic en Next.

Figura 4.39: Paso 7 Importar variables



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ La siguiente ventana que se muestra es para determinar el tipo de almacenamiento que van a tener las variables, el más recomendado y aparece como predeterminado por el asistente es el tipo **Delta**, este tipo tiene como característica que almacena los datos en el instante en que sensa una variación en el valor de una variable, se selecciona **Use Delta Storage For All Topics**, el resto de valores especifican las características que tendrá este tipo almacenamiento, es aconsejable que se mantengan sus valores por default y se da clic en **Next** para seguir con el siguiente paso.

Figura 4.40: Paso 8 Importar variables

Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

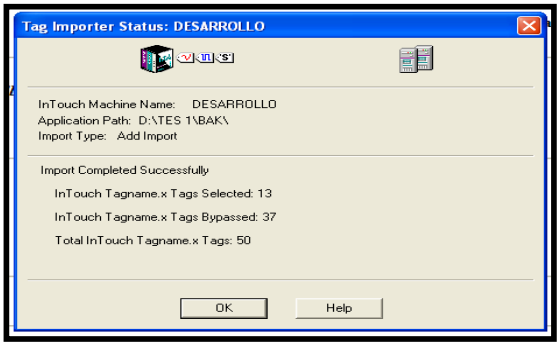
- ✓ Al momento toda la importación está lista por lo que aparece una ventana de confirmación mostrando las acciones que se van a realizar, se da clic en **Finish** para comenzar con la importación de las variables desde InTouch.

Figura 4.41: Paso 9 Importar variables

Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

- ✓ Luego de esperar por algunos segundos hasta que el proceso de importación haya terminado aparece una ventana de resumen donde se muestran los resultados de la importación incluyendo el número total de variables importadas, para terminar con la importación se da clic en **OK.**

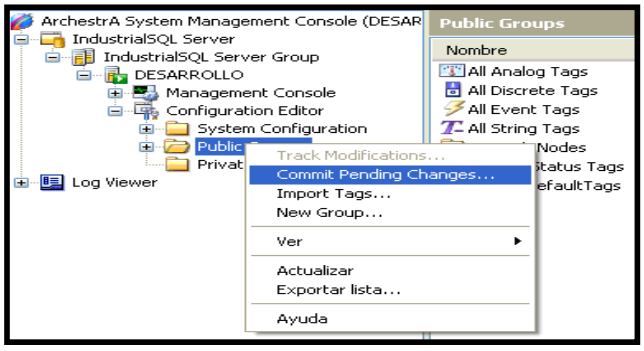
Figura 4.42: Paso 10 Importar variables



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ Cuando el proceso de importación ha finalizado todavía las variables no están listas para almacenar los datos porque el sistema requiere antes que verifique que las variables han sido importadas. Para esto nuevamente se da clic derecho en **Configuration Editor** y luego se da clic en **Commit Pending changes.**

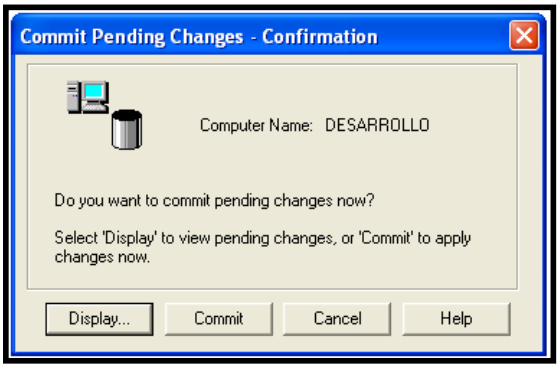
Figura 4.43: Paso 1 confirmar cambios realizados



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ El cuadro de diálogo para confirmación de cambios aparece mostrando algunas características de la base de datos.

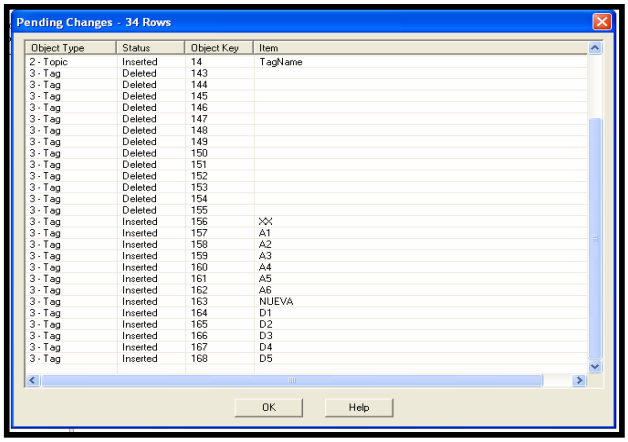
Figura 4.44: Paso 2 Confirmar pasos realizados



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

- ✓ Se recomienda dar clic en **Display** para visualizar los cambios que están pendientes y que se van a confirmar.

Figura 4.45: Paso 3 Confirmar cambios realizados

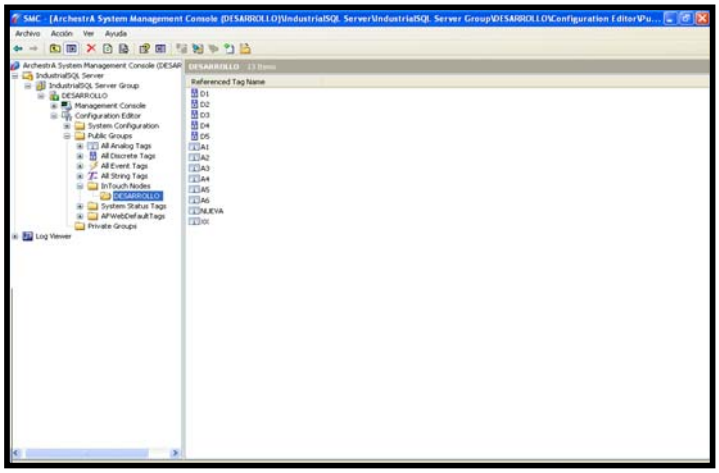


Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

- ✓ Luego de revisar si la información que se va a confirmar es la adecuada se cierra el cuadro dando clic en **OK** y luego en la ventana anterior del cuatro de diálogo dar clic en **Commit** y los cambios van a ser confirmados.

Recién cuando todos los cambios han sido confirmados es posible visualizar las variables que se han importado desde la aplicación de InTouch, para poder ver las variables de deben desplegar las siguientes pestañas: Configuration Editor, Public Groups, InTouch Nodes y dar clic en el nombre del nodo que es "DESARROLLO", de esta forma en la parte derecha de la ventana se visualizan todas las variables importadas que ven a almacenar sus valores el la base de datos.

Figura 4.46: Visualizar variables importadas

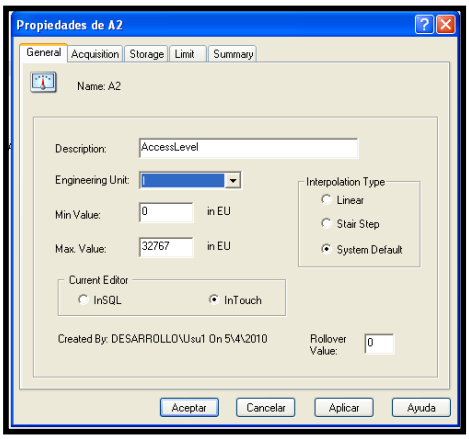


Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

Al dar doble clic sobre cualquiera de las variables importadas se abre un cuadro de diálogo que muestra todas la características de la variable como por ejemplo el nombre, el tipo de variable, su unidades de ingeniería, su descripción, sus características de adquisición, sus propiedades de almacenamiento, etc. Dichas características son vulnerables a ser modificadas, en esta aplicación no se han modificado ninguna de estas propiedades, es decir, todas las características fueron establecidas el momento de crear

la pantalla HMI en el software InTouch. La siguiente figura muestra el cuadro de propiedades de una variable.

Figura 4.47: Propiedades de una variables



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

Desde el instante en que las variables se muestran en la consola de administración dentro de la carpeta de **InTouch Nodes** y si el sistema está iniciado el almacenamiento de los valores de dichas variables ya se está realizando.

4.5 JavaScript

JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas.

Una página web dinámica es aquella que incorpora efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario.

Técnicamente, JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios.

4.5.1 Glosario básico

Script: Es cada uno de los programas, aplicaciones o código creados con el lenguaje de programación JavaScript. Unas pocas líneas de código forman un script y un archivo de miles de líneas de JavaScript también se considera un script.

Sentencia: cada una de las instrucciones que forman un script.

Palabras reservadas: son las palabras (en inglés) que se utilizan para construir las sentencias de JavaScript y que por tanto no pueden ser utilizadas libremente. Las palabras actualmente reservadas por JavaScript son: **break, case, catch, continue, default, delete, do, else, finally, for, function, if, in, instanceof, new, return, switch, this, throw, try, typeof, var, void, while, with**.

4.5.2 Sintaxis

La sintaxis de un lenguaje de programación se define como el conjunto de reglas que deben seguirse al escribir el código fuente de los programas para considerarse como correctos para ese lenguaje de programación.

La sintaxis de JavaScript es muy similar a la de otros lenguajes de programación. Las normas básicas que definen la sintaxis de JavaScript son las siguientes:

- ✓ **No se tienen en cuenta los espacios en blanco y las nuevas líneas**: como sucede con XHTML, el intérprete de JavaScript ignora cualquier espacio en blanco sobrante, por lo que el código se puede ordenar de forma adecuada para entenderlo mejor (tabulando las líneas, añadiendo espacios, creando nuevas líneas, etc.)
- ✓ **Se distinguen las mayúsculas y minúsculas**: al igual que sucede con la sintaxis de las etiquetas y elementos XHTML. Sin embargo, si en una página XHTML se utilizan indistintamente mayúsculas y minúsculas, la página se visualiza correctamente, siendo el único problema la no validación de la página. En cambio, si en JavaScript se intercambian mayúsculas y minúsculas el script no funciona.
- ✓ **No se define el tipo de las variables**: al crear una variable, no es necesario indicar el tipo de dato que almacenará. De esta forma, una misma variable puede almacenar diferentes tipos de datos durante la ejecución del script.
- ✓ **No es necesario terminar cada sentencia con el carácter de punto y coma (;)**: en la mayoría de lenguajes de programación, es obligatorio terminar cada sentencia con el carácter **(;)**. Aunque JavaScript no obliga a hacerlo, es conveniente seguir la tradición de terminar cada sentencia con el carácter del punto y coma **(;).**
- ✓ **Se pueden incluir comentarios**: los comentarios se utilizan para añadir información en el código fuente del programa. Aunque el contenido de los comentarios no se visualiza por pantalla, si que se envía al navegador del usuario junto con el resto del script, por lo que es necesario extremar las precauciones sobre la información incluida en los comentarios.

JavaScript define dos tipos de comentarios: los de una sola línea y los que ocupan varias líneas.

Ejemplo de comentario de una sola línea:

```
// a continuación se muestra un mensaje
alert("mensaje de prueba");
```

Los comentarios de una sola línea se definen añadiendo dos barras oblicuas (//) al principio de la línea.

Ejemplo de comentario de varias líneas:

```
/* Los comentarios de varias líneas son muy útiles
cuando se necesita incluir bastante información
en los comentarios */
alert("mensaje de prueba");
```

Los comentarios multilínea se definen encerrando el texto del comentario entre los símbolos **/*** y ***/.**

4.5.3 Programación básica

Antes de comenzar a desarrollar programas y utilidades con JavaScript, es necesario conocer los elementos básicos con los que se construyen las aplicaciones.

4.5.3.1 Variables

Una variable es un elemento que se emplea para almacenar y hacer referencia a otro valor. Gracias a las variables es posible crear *"programas genéricos"*, es decir, programas que funcionan siempre igual independientemente de los valores concretos utilizados.

De la misma forma que si en Matemáticas no existieran las variables no se podrían definir las ecuaciones y fórmulas, en programación no se podrían hacer programas realmente útiles sin las variables.

Aquí se presenta un ejemplo utilizando variables para almacenar y referirse dos números que van a sumarse:

```
numero_1 = 3

numero_2 = 1

resultado = numero_1 + numero_2
```

Los elementos numero_1 y numero_2 son **variables** que almacenan los valores que utiliza el programa. El resultado se calcula siempre en función del valor almacenado por las variables, por lo que este programa funciona correctamente para cualquier par de números indicado. Si se modifica el valor de las variables numero_1 y numero_2, el programa sigue funcionando correctamente.

Las variables en JavaScript se crean mediante la palabra reservada **var.** De esta forma, el ejemplo anterior se puede realizar en JavaScript de la siguiente manera:

```
var numero_1 = 3;
var numero_2 = 1;
var resultado = numero_1 + numero_2;
```

La palabra reservada **var** solamente se debe indicar al definir por primera vez la variable, lo que se denomina **declarar** una variable. Cuando se utilizan las variables en el resto de instrucciones del script, solamente es necesario indicar su nombre.

Si cuando se declara una variable se le asigna también un valor, se dice que la variable ha sido **inicializada**. En JavaScript no es obligatorio inicializar las variables, ya que se pueden declarar por una parte y asignarles un valor posteriormente. Por tanto, el ejemplo anterior se puede presentar de la siguiente manera:

```
var numero_1;
var numero_2;
numero_1 = 3;
numero_2 = 1;
var resultado = numero_1 + numero_2;
```

Una de las características más sorprendentes de JavaSript para los programadores relacionados con otros lenguajes de programación es que tampoco es necesario declarar las variables. En otras palabras, se pueden utilizar variables que no se han definido anteriormente mediante la palabra reservada **var**.

En todo caso, se recomienda declarar todas las variables que se vayan a utilizar. El nombre de una variable también se conoce como **identificador** y debe cumplir las siguientes normas:

- ✓ Sólo puede estar formado por letras, números y los símbolos $ (dólar) y _ (guión bajo).
- ✓ El primer carácter no puede ser un número.

4.5.3.2 Tipos de variables

Aunque todas las variables de JavaScript se crean de la misma forma (mediante la palabra reservada var), la forma en la que se les asigna un valor depende del tipo de valor que se quiere almacenar (números, textos, etc.)

Numéricas

Se utilizan para almacenar valores numéricos enteros (llamados *integer* en inglés) o decimales (llamados *float* en inglés). En este caso, el valor se asigna indicando directamente el número entero o decimal. Los números decimales utilizan el carácter **.**

(punto) en vez de , (coma) para separar la parte entera y la parte decimal. A continuación se presenta un ejemplo:

```
var iva = 12; // variable tipo entero

var total = 777.66; // variable tipo decimal
```

Cadenas de texto

Se utilizan para almacenar caracteres, palabras y/o frases de texto. Para asignar el valor a la variable, se encierra el valor entre comillas dobles o simples, para delimitar su comienzo y su final, como se muestra a continuación:

```
var mensaje = "Bienvenido a mi página web";

var nombreProducto = 'Artícula uno';

var letraSeleccionada = 'c';
```

En ocasiones, el texto que se almacena en las variables no es tan sencillo. Si por ejemplo el propio texto contiene comillas simples o dobles, la estrategia que se sigue es la de encerrar el texto con las comillas (simples o dobles) que no utilice el texto.

Pero, a veces las cadenas de texto contienen tanto comillas simples como dobles. Además, existen otros caracteres que son difíciles de incluir en una variable de texto (tabulador, ENTER, etc.). Para resolver estos problemas, JavaScript define un mecanismo para incluir de forma sencilla caracteres especiales y problemáticos dentro de una cadena de texto.

El mecanismo consiste en sustituir el carácter problemático por una combinación simple de caracteres. A continuación se muestra la tabla de conversión que se debe utilizar:

Tabla 2: Conversión

Si se quiere incluir...	Se debe incluir...
Una nueva línea	\n
Un tabulador	\t
Una comilla simple	\'
Una comilla doble	\"
Una barra inclinada	\\

Fuente: http://www.librosweb.es/javascript
Elaborado por: César Zambrano / 2010

Arrays

Un array es una colección de variables, que pueden ser todas del mismo tipo o cada una de un tipo diferente. Su utilidad se comprende mejor con un ejemplo sencillo: si una aplicación necesita manejar los días de la semana, se podrían crear siete variables de tipo texto.

```
var dia1 = "Lunes";
var dia2 = "Martes";
...
```

```
var dia7 = "Domingo";
```

En este tipo de casos, se pueden agrupar todas las variables relacionadas en una colección de variables o array. El ejemplo anterior se puede rehacer de la siguiente forma:

```
var días = ["Lunes", "Martes", "Miércoles", "Jueves", "Viernes", "Sábado",
"Domingo"];
```

Ahora, una única variable llamada **días** almacena todos los valores relacionados entre sí, en este caso los días de la semana. Para definir un array, se utilizan los caracteres [y] para delimitar su comienzo y su final y se utiliza el carácter , (coma) para separar sus elementos.

Una vez definido un array, es muy sencillo acceder a cada uno de sus elementos. Cada elemento se accede indicando su posición dentro del array. La única complicación, es que se debe tomar en cuenta, es que las posiciones de los elementos empiezan a contarse en el 0 y no en el 1.

```
var diaSeleccionado = dias[0]; // diaSeleccionado = "Lunes"
var otroDia = dias[5]; // otroDia = "Sábado"
```

Booleanos

Las variables de tipo *boolean* o *booleano* también se conocen con el nombre de variables de tipo lógico. Su funcionamiento básico es muy sencillo.

Una variable de tipo *boolean* almacena un tipo especial de valor que solamente puede tomar dos valores: true (verdadero) o false (falso). No se puede utilizar para almacenar números y tampoco permite guardar cadenas de texto, a continuación se muestra un par de variables de tipo booleano:

```
var clienteRegistrado = false;
var ivaIncluido = true;
```

4.5.3.3 Operadores

Las variables por sí solas son de poca utilidad. Hasta ahora, sólo se ha visto cómo crear variables de diferentes tipos y cómo mostrar su valor mediante la función alert(). Para hacer programas realmente útiles, son necesarias otro tipo de herramientas.

Los operadores permiten manipular el valor de las variables, realizar operaciones matemáticas con sus valores y comparar diferentes variables.

De esta forma, los operadores permiten a los programas realizar cálculos complejos y tomar decisiones lógicas en función de comparaciones y otros tipos de condiciones.

Asignación

El operador de asignación es el más utilizado y el más sencillo. Este operador se utiliza para guardar un valor específico en una variable. El símbolo utilizado es = (no confundir con el operador ==) como se muestra a continuación:

```
var numero1 = 3;
```

A la izquierda del operador, siempre debe indicarse el nombre de una variable. A la derecha del operador, se pueden indicar variables, valores, condiciones lógicas, etc:

Incremento y decremento

Estos dos operadores solamente son válidos para las variables numéricas y se utilizan para incrementar o decrementar en una unidad el valor de una variable. Ejemplo:

```
var numero = 5;
++numero;
alert(numero); // numero = 6
```

El operador de incremento se indica mediante el prefijo ++ en el nombre de la variable. El resultado es que el valor de esa variable se incrementa en una unidad.

De forma equivalente, el operador decremento (indicado como un prefijo -- en el nombre de la variable) se utiliza para decrementar el valor de la variable. Ejemplo:

```
var numero = 5;
--numero;
alert(numero); // numero = 4
```

Lógicos

Los operadores lógicos son imprescindibles para realizar aplicaciones complejas, ya que se utilizan para tomar decisiones sobre las instrucciones que debería ejecutar el programa en función de ciertas condiciones.

El resultado de cualquier operación que utilice operadores lógicos siempre es un valor lógico o booleano.

Negación

Uno de los operadores lógicos más utilizados es el de la negación. Se utiliza para obtener el valor contrario al valor de la variable:

```
var visible = true;

alert(!visible); // Muestra "false" y no "true"
```

La negación lógica se obtiene prefijando el símbolo **!** al identificador de la variable.

AND

La operación lógica AND obtiene su resultado combinando dos valores booleanos. El operador se indica mediante el símbolo && y su resultado solamente es true si los dos operadores son true, ejemplo:

```
var valor1 = true;

var valor2 = false;

resultado = valor1 && valor2; // resultado = false

valor1 = true;

valor2 = true;

resultado = valor1 && valor2; // resultado = trae
```

OR

La operación lógica OR también combina dos valores booleanos. El operador se indica mediante el símbolo || y su resultado es true si alguno de los dos operadores es true, ejemplo:

var valor1 = true;

var valor2 = false;

resultado = valor1 || valor2; *// resultado = true*

valor1 = false;

valor2 = false;

resultado = valor1 || valor2; *// resultado = false*

Matemáticos

JavaScript permite realizar manipulaciones matemáticas sobre el valor de las variables numéricas. Los operadores definidos son: suma (+), resta (-), multiplicación (*) y división (/). Ejemplo:

var numero1 = 10;

var numero2 = 5;

resultado = numero1 / numero2; *// resultado = 2*

resultado = 7 + numero1; *// resultado = 17*

resultado = numero2 – 3 *// resultado = 2*

resultado = numero1 * numero 2; *// resultado = 50*

Relacionales

Los operadores relacionales definidos por JavaScript son idénticos a los que definen las matemáticas: mayor que (>), menor que (<), mayor o igual (>=), menor o igual (<=), igual que (==) y distinto de (!=).

Los operadores que relacionan variables son imprescindibles para realizar cualquier aplicación compleja, El resultado de todos estos operadores siempre es un valor booleano.

var numero1 = 5;

var numero2 = 7;

resultado = numero1 > numero2; *// resultado = false*

resultado = numero1 < numero2; *// resultado = true*

numero1 = 9;

numero2 = 9;

resultado = numero1 >= numero2; *// resultado = true*

resultado = numero1 <= numero2; *// resultado = true*

resultado = numero1 == numero2; *// resultado = true*

resultado = numero1 != numero2; *// resultado = false*

Se debe tener especial cuidado con el operador de igualdad (==), el operador == se utiliza para comparar el valor de dos variables, por lo que es muy diferente del operador =, que se utiliza para asignar un valor a una variable.

Cuando se utilizan cadenas de texto, los operadores "mayor que" (>) y "menor que" (<) siguen un razonamiento no intuitivo: se compara letra a letra comenzando desde la izquierda hasta que se encuentre una diferencia entre las dos cadenas de texto.

4.5.4 Estructuras de control de flujo

Para realizar programas que no son tan simples como una sucesión lineal de instrucciones básicas, son necesarias las **estructuras de control de flujo,** que son instrucciones del tipo ″si se cumple esta condición, hazlo; si no se cumple, haz esto otro". También existen instrucciones del tipo "repite esto mientras se cumpla esta condición".

Si se utilizan estructuras de control de flujo, los programas dejan de ser una sucesión lineal de instrucciones para convertirse en programas *inteligentes* que pueden tomar decisiones en función del valor de las variables.

4.5.4.1 Estructura if

La estructura más utilizada en JavaScript y en la mayoría de lenguajes de programación es la estructura if. Se emplea para tomar decisiones en función de una condición. Su definición formal es:

```
if(condicion) {
...
}
```

Si la condición se cumple (es decir, si su valor es true) se ejecutan todas las instrucciones que se encuentran dentro de {...}. Si la condición no se cumple (es decir, si su valor es false) no se ejecuta ninguna instrucción contenida en {...} y el programa continúa ejecutando el resto de instrucciones del script.

4.5.4.2 Estructura if...else

En ocasiones, las condiciones suelen ser del tipo *"si se cumple esta condición, hazlo; si no se cumple, haz esto otro"*. Para este tipo de decisiones, existe una variante de la estructura **if** llamada **if...else**. Su definición formal es la siguiente:

```
if(condicion) {
...
}
else {
...
}
```

Si la condición se cumple (es decir, si su valor es true) se ejecutan todas las instrucciones que se encuentran dentro del if (). Si la condición no se cumple (es decir, si su valor es false) se ejecutan todas las instrucciones contenidas en else { }.

4.5.4.3 Estructura for

La estructura for permite realizar repeticiones (también llamadas bucles) de una forma muy sencilla. Su definición formal es la siguiente:

```
for(inicializacion; condicion; actualizacion) {
...
}
```

La idea del funcionamiento de un bucle for es la siguiente: "mientras la condición indicada se siga cumpliendo, repite la ejecución de las instrucciones definidas dentro del for. Además, después de cada repetición, actualiza el valor de las variables que se utilizan en la condición".

- ✓ La "inicialización" es la zona en la que se establece los valores iniciales de las variables que controlan la repetición.
- ✓ La "condición" es el único elemento que decide si continua o se detiene la repetición.
- ✓ La "actualización" es el nuevo valor que se asigna después de cada repetición a las variables que controlan la repetición.

4.5.4.4 Estructura for...in

Una estructura de control derivada de for es la estructura for...in. Su definición exacta implica el uso de objetos, Su definición formal adaptada a los arrays es:

```
for(indice in array) {
...
}
```

4.5.5 Funciones y propiedades básicas de JavaScript

JavaScript incorpora una serie de herramientas y utilidades llamadas funciones y propiedades, para el manejo de las variables. De esta forma, muchas de las operaciones básicas con las variables, se pueden realizar directamente con las utilidades que ofrece JavaScript.

4.5.5.1 Funciones útiles para cadenas de texto

A continuación se muestran algunas de las funciones más útiles para el manejo de cadenas de texto:

length, calcula la longitud de una cadena de texto (el número de caracteres que la forman), ejemplo:

```
var mensaje = "Hola César";

var numeroLetras = mensaje.length; // numeroLetras = 9
```

toUpperCase(), transforma todos los caracteres de la cadena a sus correspondientes caracteres en mayúsculas, ejemplo:

```
var mensaje1 = "Hola";

var mensaje2 = mensaje1.toUpperCase(); // mensaje2 = "HOLA"
```

toLowerCase(), transforma todos los caracteres de la cadena a sus correspondientes caracteres en minúsculas, ejemplo:

```
var mensaje1 = "HolA";

var mensaje2 = mensaje1.toLowerCase(); // mensaje2 = "hola"
```

charAt(posicion), obtiene el carácter que se encuentra en la posición indicada, ejemplo:

```
var mensaje = "Hola";

letra = mensaje.charAt(2); // letra = l
```

indexOf(caracter), calcula la posición en la que se encuentra el carácter indicado dentro de la cadena de texto. Si el carácter se incluye varias veces dentro de la cadena de texto, se devuelve su primera posición empezando a buscar desde la izquierda. Si la cadena no contiene el carácter, la función devuelve el valor -1, ejemplo:

```
var mensaje = "Hola";
var posicion = mensaje.indexOf('a'); // posicion = 3
```

lastIndexOf(caracter), calcula la última posición en la que se encuentra el carácter indicado dentro de la cadena de texto. Si la cadena no contiene el carácter, la función devuelve el valor -1:

```
var mensaje = "Hola";
var posicion = mensaje.lastIndexOf('a'); // posicion = 3
```

La función lastIndexOf() comienza su búsqueda desde el final de la cadena hacia el inicio, aunque la posición devuelta es la correcta empezando a contar desde el principio de la palabra.

substring(inicio, final), extrae una porción de una cadena de texto. El segundo parámetro es opcional. Si sólo se indica el parámetro inicio, la función devuelve la parte de la cadena original correspondiente desde esa posición hasta el final, ejemplo:

```
var mensaje = "Hola César";
```

```
var porcion = mensaje.substring(2); // porcion = "la César"
```

split(separador), convierte una cadena de texto en un array de cadenas de texto. La función parte la cadena de texto determinando sus partes a partir del carácter separador indicado, ejemplo:

```
var mensaje = "Hola César, soy una cadena de texto!";
var palabras = mensaje.split(" ");
// palabras = ["Hola", "César,", "soy", "una", "cadena", "de", "texto!"];
```

4.5.5.2 Funciones útiles para arrays

A continuación se muestran algunas de las funciones más útiles para el manejo de arrays:

length, calcula el número de elementos de un array, ejemplo:

```
var vocales = ["a", "e", "i", "o", "u"];
var numeroVocales = vocales.length; // numeroVocales = 5
```

concat(), se emplea para concatenar los elementos de varios arrays, ejemplo:

```
var array1 = [1, 2, 3];
```

array2 = array1.concat(4, 5, 6); *// array2 = [1, 2, 3, 4, 5, 6]*

join(separador), es la función contraria a split(). Une todos los elementos de un array para formar una cadena de texto. Para unir los elementos se utiliza el carácter separador indicado:

var array = ["hola", "césar"];

var mensaje = array.join(""); // mensaje = "holacésar"

pop(), elimina el último elemento del array y lo devuelve. El array original se modifica y su longitud disminuye en 1 elemento, ejemplo:

var array = [1, 2, 3];

var ultimo = array.pop();

// ahora array = [1, 2], ultimo = 3

push(), añade un elemento al final del array. El array original se modifica y aumenta su longitud en 1 elemento. (También es posible añadir más de un elemento a la vez), ejemplo:

var array = [1, 2, 3];

array.push(4);

// ahora array = [1, 2, 3, 4]

shift(), elimina el primer elemento del array y lo devuelve. El array original se ve modificado y su longitud disminuida en 1 elemento, ejemplo:

```
var array = [1, 2, 3];

var primero = array.shift();

// ahora array = [2, 3], primero = 1
```

unshift(), añade un elemento al principio del array. El array original se modifica y aumenta su longitud en 1 elemento. (También es posible añadir más de un elemento a la vez), ejemplo:

```
var array = [1, 2, 3];

array.unshift(0);

// ahora array = [0, 1, 2, 3]
```

reverse(), modifica un array colocando sus elementos en el orden inverso a su posición original:

```
var array = [1, 2, 3];

array.reverse();

// ahora array = [3, 2, 1]
```

4.5.5.3 Funciones útiles para números

A continuación se muestran algunas de las funciones y propiedades más útiles para el manejo de números.

NaN, (del inglés, *"Not a Number"*) JavaScript emplea el valor NaN para indicar un valor numérico no definido (por ejemplo, la división 0/0).

```
var numero1 = 0;
var numero2 = 0;
alert(numero1/numero2); // se muestra el valor NaN
```

isNaN(), permite proteger a la aplicación de posibles valores numéricos no definidos, ejemplo:

```
var numero1 = 0;
var numero2 = 0;
if(isNaN(numero1/numero2)) {
alert("La división no está definida para los números indicados");
}
else {
```

```
alert("La división es igual a => " + numero1/numero2);

}
```

Infinity, hace referencia a un valor numérico infinito y positivo (también existe el valor –Infinity para los infinitos negativos)

```
var numero1 = 10;

var numero2 = 0;

alert(numero1/numero2); // se muestra el valor Infinity
```

toFixed(digitos), devuelve el número original con tantos decimales como los indicados por el parámetro dígitos y realiza los redondeos necesarios, ejemplo:

```
var numero1 = 4564.34567;

numero1.toFixed(2); // 4564.35

numero1.toFixed(6); // 4564.345670

numero1.toFixed(); // 4564
```

4.5.6 Programación avanzada

Las estructuras de control, los operadores y todas las utilidades propias de JavaScript que se han descrito hasta ahora, permiten crear scripts sencillos y de mediana complejidad.

Sin embargo, para las aplicaciones más complejas son necesarios otros elementos como las funciones y otras estructuras de control más avanzadas, que se describen a continuación.

4.5.6.1 Funciones

Cuando se desarrolla una aplicación compleja, es muy habitual utilizar una y otra vez las mismas instrucciones. Cuando una serie de instrucciones se repiten una y otra vez, se complica demasiado el código fuente de la aplicación, ya que:

- ✓ El código de la aplicación es mucho más largo porque muchas instrucciones están repetidas.
- ✓ Si se quiere modificar alguna de las instrucciones repetidas, se deben hacer tantas modificaciones como veces se haya escrito esa instrucción, lo que se convierte en un trabajo muy pesado y muy propenso a cometer errores.

Las funciones son la solución a todos estos problemas, tanto en JavaScript como en el resto de lenguajes de programación. Una función es un conjunto de instrucciones que se agrupan para realizar una tarea concreta y que se pueden reutilizar fácilmente.

Las funciones en JavaScript se definen mediante la palabra reservada

function, seguida del nombre de la función. Su definición formal es la siguiente:

```
function nombre_funcion() {
...
}
```

El nombre de la función se utiliza para llamar a esa función cuando sea necesario. El concepto es el mismo que con las variables, a las que se les asigna un nombre único para poder utilizarlas dentro del código. Después del nombre de la función, se incluyen dos paréntesis, por último, los símbolos { y } se utilizan para encerrar todas las instrucciones que pertenecen a la función, ejemplo:

```
function suma_y_muestra() {
resultado = numero1 + numero2;
alert("El resultado es " + resultado);
}
var resultado;
var numero1 = 2;
var numero2 = 7;
suma_y_muestra();
numero1 = 11;
numero2 = 71;
suma_y_muestra();
numero1 = 55;
numero2 = 84;
suma_y_muestra();
...
```

4.5.6.2 Argumentos y valores de retorno

Las variables que necesitan las funciones se llaman argumentos. Antes de que pueda utilizarlos, la función debe indicar cuántos argumentos necesita y cuál es el nombre de cada argumento. Además, al invocar la función, se deben incluir los valores que se le van a pasar a la función. Los argumentos se indican dentro de los paréntesis que van detrás del nombre de la función y se separan con una coma (,).

En el ejemplo anterior, la función debe indicar que necesita dos argumentos, respectivamente para los dos números que tiene que sumar:

```
function suma_y_muestra(primerNumero, segundoNumero) { ... }
```

A continuación, para utilizar el valor de los argumentos dentro de la función, se debe emplear el mismo nombre con el que se definieron los argumentos:

```
function suma_y_muestra(primerNumero, segundoNumero) { ... }
var resultado = primerNumero + segundoNumero;
alert("El resultado es " + resultado);
}
```

Dentro de la función, el valor de la variable primerNumero será igual al primer valor que se le pase a la función y el valor de la variable segundoNumero será igual al

segundo valor que se le pasa. Para pasar valores a la función, se incluyen dentro de los paréntesis utilizados al llamar a la función:

```
// Definición de la función
function suma_y_muestra(primerNumero, segundoNumero) { ... }
var resultado = primerNumero + segundoNumero;
alert("El resultado es " + resultado);
}
// Declaración de las variables
var numero1 = 3;
var numero2 = 5;
// Llamada a la función
suma_y_muestra(numero1, numero2);
```

4.5.6.3 Ámbito de las variables

El ámbito de una variable (llamado *"scope"* en inglés) es la zona del programa en la que se define la variable. JavaScript define dos ámbitos para las variables: global y local.

El siguiente ejemplo ilustra el comportamiento de los ámbitos:

```
function creaMensaje() {
```

```
var mensaje = “Mensaje de prueba”;
}
creaMensaje();
alert(mensaje);
```

El ejemplo anterior define en primer lugar una función llamada creaMensaje que crea una variable llamada mensaje. A continuación, se ejecuta la función mediante la llamada creaMensaje(); y seguidamente, se muestra mediante la función alert() el valor de una variable llamada mensaje.

Sin embargo, al ejecutar el código anterior no se muestra ningún mensaje por pantalla. La razón es que la variable mensaje se ha definido dentro de la función creaMensaje() y por tanto, es una **variable local** que solamente está definida dentro de la función.

Cualquier instrucción que se encuentre dentro de la función puede hacer uso de esa variable, pero todas las instrucciones que se encuentren en otras funciones o fuera de cualquier función no tendrán definida la variable mensaje. De esta forma, para mostrar el mensaje en el código anterior, la función alert() debe llamarse desde dentro de la función creaMensaje():

```
function creaMensaje() {
var mensaje = “Mensaje de prueba”;
alert(mensaje);
}
```

```
creaMensaje();
```

Además de variables locales, también existe el concepto de **variable global**, que está definida en cualquier punto del programa.

Si una variable se declara fuera de cualquier función, automáticamente se transforma en variable global independientemente de si se define utilizando la palabra reservada var o no. Sin embargo, las variables definidas dentro de una función pueden ser globales o locales.

4.5.6.4 Sentencias break y continue

Las sentencias break y continue permiten manipular el comportamiento normal de los bucles for para detener el bucle o para saltarse algunas repeticiones. Concretamente, la sentencia break permite terminar de forma abrupta un bucle y la sentencia continue permite saltarse algunas repeticiones del bucle.

El siguiente ejemplo muestra el uso de la sentencia break:

```
var cadena = "En una ciudad de ecuador cuyo nombre no quiero acordarme...";
var letras = cadena.split("");
var resultado = "";
for(i in letras) {
```

```
if(letras[i] == 'a') {

break;

}

else {

resultado += letras[i];

}

}

alert(resultado);

// muestra "En un lug"
```

Si el programa llega a una instrucción de tipo break;, sale inmediatamente del bucle y continúa ejecutando el resto de instrucciones que se encuentran fuera del bucle for.

La utilidad de break es terminar la ejecución del bucle cuando una variable toma un determinado valor o cuando se cumple alguna condición.

En ocasiones, lo que se desea es saltarse alguna repetición del bucle cuando se dan algunas condiciones. Continuando con el ejemplo anterior, ahora se desea que el texto de salida elimine todas las letras "a" de la cadena de texto original:

```
var cadena = "En una ciudad de ecuador cuyo nombre no quiero acordarme...";

var letras = cadena.split("");

var resultado = "";
```

```
for(i in letras) {

if(letras[i] == 'a') {

continue;

}

else {

resultado += letras[i];

}

}

alert(resultado);

// muestra "En un lugr de l Mnch de cuyo nombre no quiero cordrme..."
```

En este caso, cuando se encuentra una letra "a" no se termina el bucle, sino que no se ejecutan las instrucciones de esa repetición y se pasa directamente a la siguiente repetición del bucle for.

La utilidad de continue es que permite utilizar el bucle for para filtrar los resultados en función de algunas condiciones o cuando el valor de alguna variable coincide con un valor determinado.

4.5.7 Otras estructuras de control

Las estructuras de control de flujo que se han descrito (if, else, for) y las sentencias que modifican su comportamiento (break, continue) no son suficientes para realizar algunas

tareas complejas y otro tipo de repeticiones. Por ese motivo, JavaScript proporciona otras estructuras de control de flujo diferentes y en algunos casos más eficientes.

4.5.7.1 Estructura while

La estructura while permite crear bucles que se ejecutan ninguna o más veces, dependiendo de la condición indicada. Su definición formal es:

```
while(condicion) {
...
}
```

El funcionamiento del bucle while se resume en: "mientras se cumpla la condición indicada, repite indefinidamente las instrucciones incluidas dentro del bucle".

Si la condición no se cumple ni siquiera la primera vez, el bucle no se ejecuta. Si la condición se cumple, se ejecutan las instrucciones una vez y se vuelve a comprobar la condición. Si se sigue cumpliendo la condición, se vuelve a ejecutar el bucle y así se continúa hasta que la condición no se cumpla.

Evidentemente, las variables que controlan la condición deben modificarse dentro del propio bucle, ya que de otra forma, la condición se cumpliría siempre y el bucle while se repetiría indefinidamente.

4.5.7.2 Estructura do...while

El bucle de tipo do...while es muy similar al bucle while, salvo que en este caso **siempre** se ejecutan las instrucciones del bucle al menos la primera vez. Su definición formal es:

```
do {
...
} while(condicion);
```

De esta forma, como la condición se comprueba después de cada repetición, la primera vez siempre se ejecutan las instrucciones del bucle.

Es importante no olvidar que después del while() se debe añadir el carácter ; (al contrario de lo que sucede con el bucle while simple).

4.5.7.3 Estructura switch

La estructura switch está especialmente diseñada para manejar de forma sencilla múltiples condiciones sobre la misma variable. Su definición formal puede parecer compleja, aunque su uso es muy sencillo:

```
switch(variable) {
case valor_1:
...
break;
case valor_2:
...
break;
...
case valor_n:
...
break;
default:
...
break;
}
```

La estructura switch se define mediante la palabra reservada switch seguida, entre paréntesis, del nombre de la variable que se va a utilizar en las comparaciones. Como es habitual, las instrucciones que forman parte del switch se encierran entre las llaves { y}.

Dentro del switch se definen todas las comparaciones que se quieren realizar sobre el valor de la variable. Cada comparación se indica mediante la palabra reservada case seguida del valor con el que se realiza la comparación. Si el valor de la variable

utilizada por switch coincide con el valor indicado por case, se ejecutan las instrucciones definidas dentro de ese case.

Normalmente, después de las instrucciones de cada case se incluye la sentencia break para terminar la ejecución del switch, aunque no es obligatorio. Las comparaciones se realizan por orden, desde el primer case hasta el último, por lo que es muy importante el orden en el que se definen los case.

Para realizar la programación, se tiene el programa NetBeans que incluye herramientas y un sinnúmero características para facilitar la generación de la documentación.

4.6 Netsbeans

El IDE NetBeans es un entorno de desarrollo integrado, una herramienta para programadores pensada para escribir, compilar, depurar y ejecutar programas. Está escrito en Java pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extender el IDE NetBeans. El IDE NetBeans es un producto libre y gratuito sin restricciones de uso.

Es un IDE de código abierto escrito completamente en Java usando la plataforma NetBeans. El NetBeans IDE soporta el desarrollo de todos los tipos de aplicación Java (J2SE, web, EJB y aplicaciones móviles).

Requerimientos mínimos de hardware:

Microsoft Windows XP Professional SP3

Procesador: 800MHz Intel Pentium III o equivalente

Memoria: 512 MB

Espacio en disco: 750 MB de espacio libre en disco

Requerimientos recomendados de hardware:

Microsoft Windows XP Professional SP3:

Procesador: 2.6 GHz Intel Pentium IV o equivalente

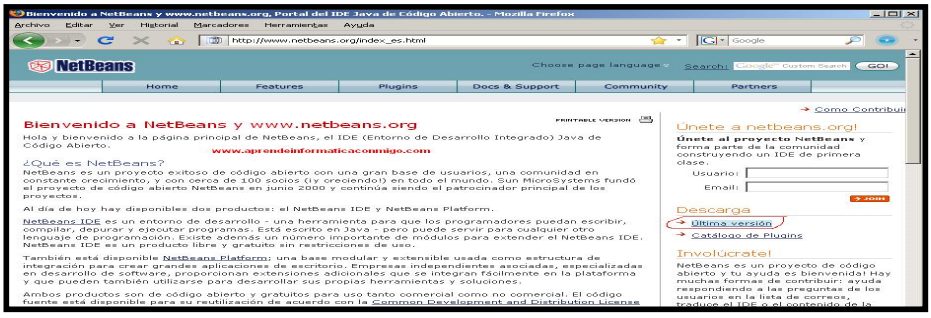
Memoria: 1 GB

Espacio en disco: 1 GB de espacio libre en disco

En primer lugar procedemos a bajarnos el software en cuestión. Para ello nos dirigimos a la página oficial de NetBeans en español donde se presentara su pantalla de bienvenida.

En la parte derecha tenemos un enlace que nos lleva directamente a la descarga de la última versión de este software. Podemos ver una pantalla como la siguiente:

Figura 4.48: Paso 1 descargar Netbeans

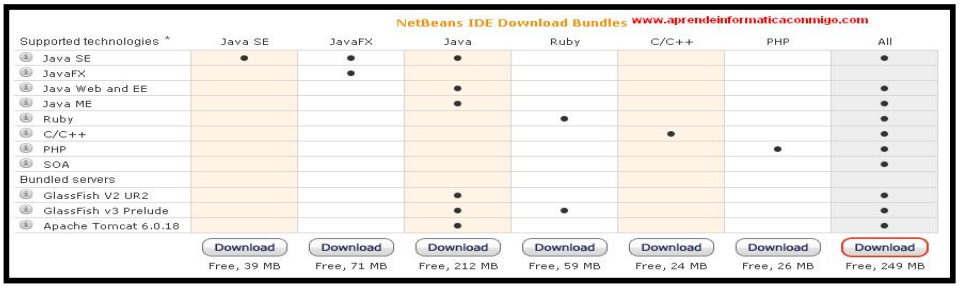


Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

Aquí tenemos las diferentes versiones de Netbeans, cada una con soporte para una serie de lenguajes de programación. Si sólo vamos a dedicar Netbeans para Java, podemos

escoger una versión específica para ello y con menos peso, pero en éste caso voy a elegir la versión en la que se engloban varios lenguajes de programación.

Figura 4.49: Paso 2 descargar Netbeans



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ Al iniciar la descarga nos da la opción de guardar el archivo para posteriormente ejecutarlo.

Figura 4.50: Paso 3 descargar Netbeans

Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

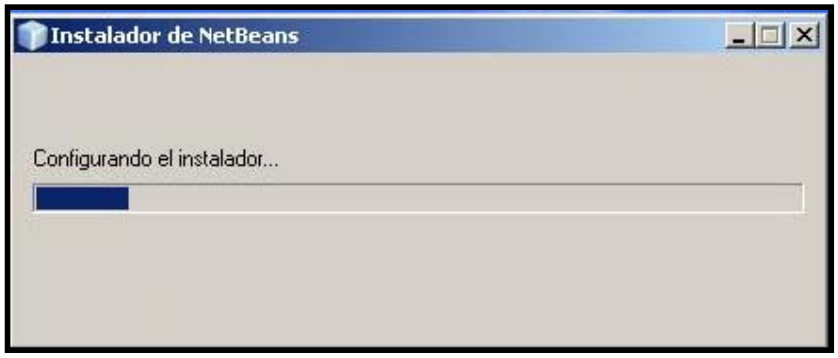
1. Una vez descargado el fichero lo ejecutamos.

Figura 4.51: Paso 1 instalar Netbeans

Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ Con eso se inicia el proceso de instalación.

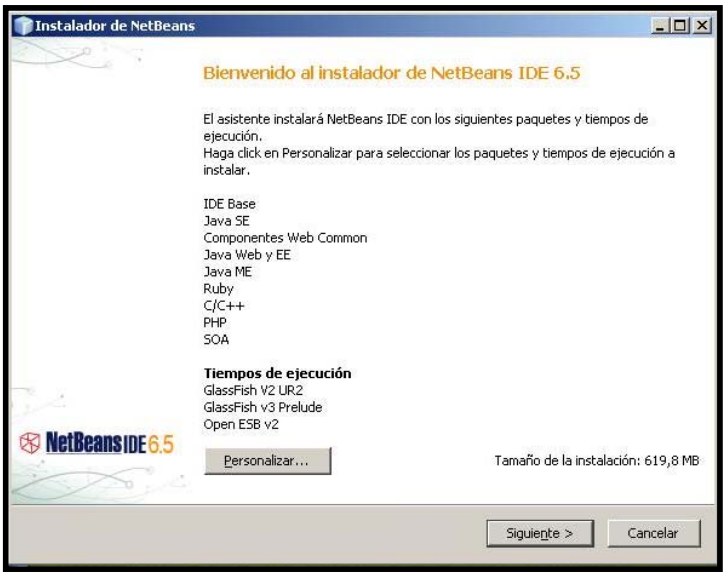
Figura 4.52: Paso 2 instalar Netbeans



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ Luego de esto se presenta la ventana de bienvenida del instalador de NetBeans, donde se indica los paquetes y tiempos de ejecución que se van a instalar, una vez leído los datos se da clic en siguiente.

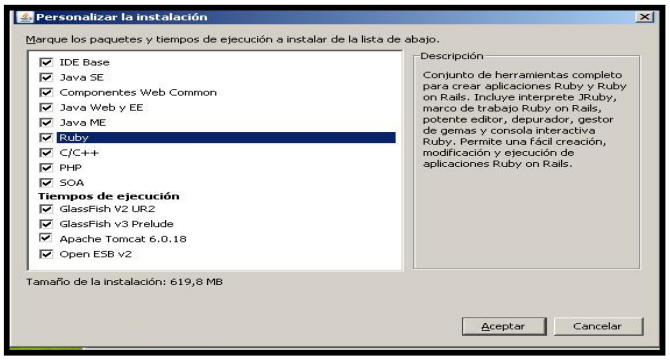
Figura 4.53: Paso 3 instalar Netbeans



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

- ✓ Se selecciona los lenguajes de programación que se va a utilizar para que se instalen los ficheros necesarios para cada uno de ellos y se da clic en aceptar.

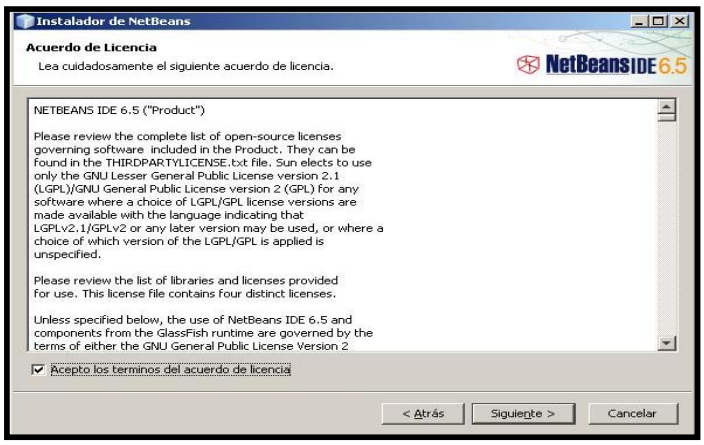
Figura 4.54: Paso 4 instalar Netbeans



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ Se lee los términos de acuerdo se licencia, se marca la casilla donde dice acepto los términos del acuerdo de licencia y se da clic en siguiente.

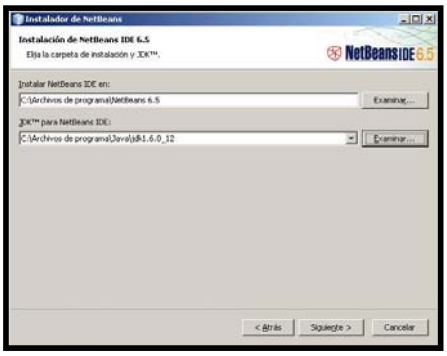
Figura 4.55: Paso 5 instalar Netbeans



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ En este paso da la opción para seleccionar la ruta donde quiere que se instalen los ficheros del programa, se recomienda dejar la preestablecida y dar clic en siguiente.

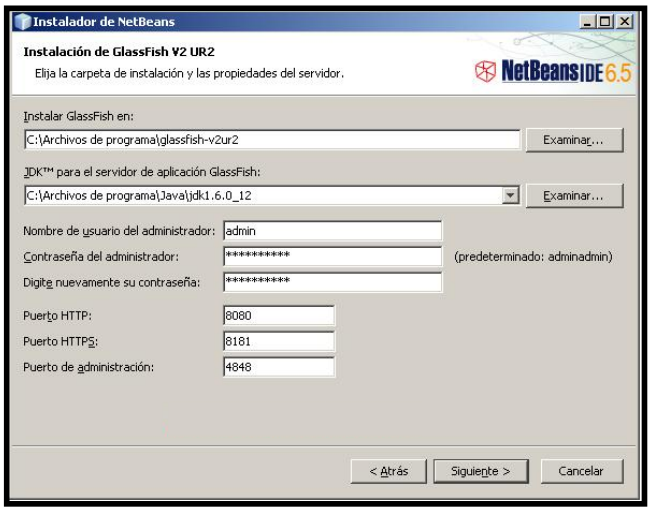
Figura 4.56: Paso 6 instalar Netbenas



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ En esta ventana se selecciona la carpeta de instalación y propiedades del servidor y luego dar clic en siguiente.

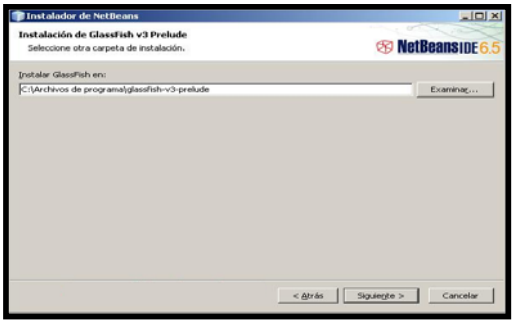
Figura 4.57: Paso 7 instalar Netbeans



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ En esta ventana se selecciona otra carpeta para la instalación de otro fichero y luego se da clic en siguiente.

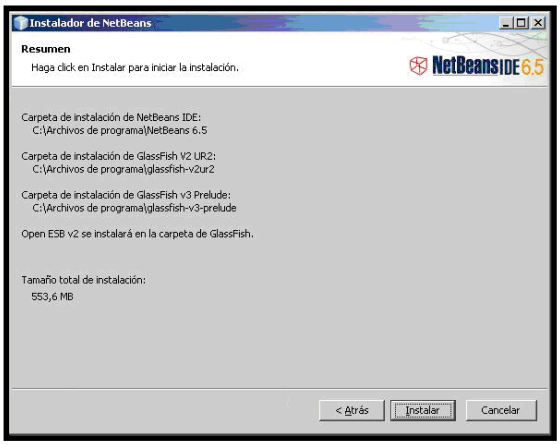
Figura 4.58: Paso 8 instalar Netbeans



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ En esta ventana se presenta el resumen de lo que se va a instalar si está de acuerdo con los datos presentado dar clic en instalar.

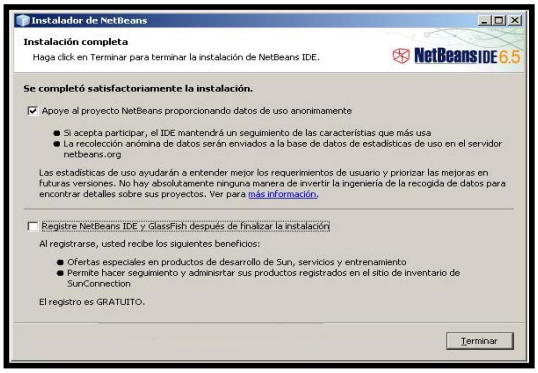
Figura 4.59: Paso 9 instalar Netbeans



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ Finalmente dar clic en terminar.

Figura 4.60: Paso 10 instalar Netbeans



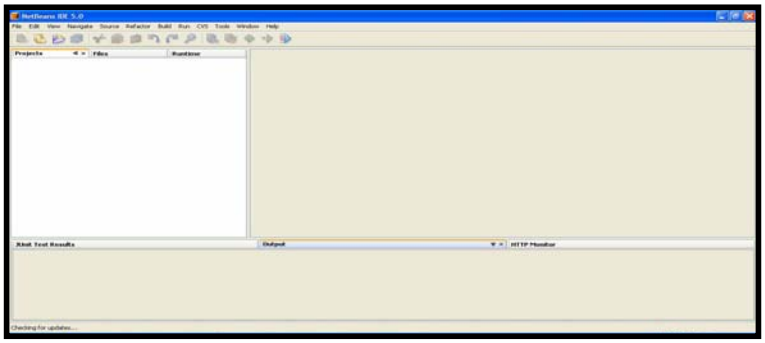
Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

4.6.1 Creación de un Proyecto

Para crear un programa de consola en Java utilizando **NetBeans** lo primero que hay que hacer es crear un proyecto. Un proyecto nos permite administrar los archivos con el código fuente y compilado de una aplicación. Para crear un proyecto se sigue el siguiente procedimiento:

- ✓ Ejecute el programa **NetBeans 5**. Al hacerlo aparecerá la ventana principal del programa como se ilustra en la figura.

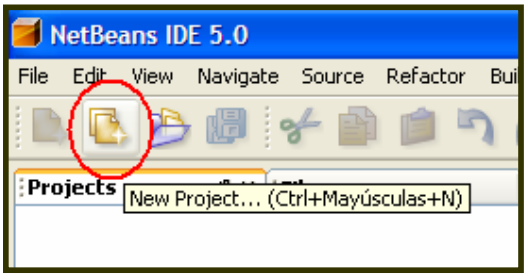
Figura 4.61: Paso 1 Crear proyecto



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

- ✓ Del menú principal de NetBeans, seleccione la opción **File/New Project,** presione las teclas **Ctrl+Mayúsculas+N** o haga clic en el icono **New Project** como se muestra en la figura.

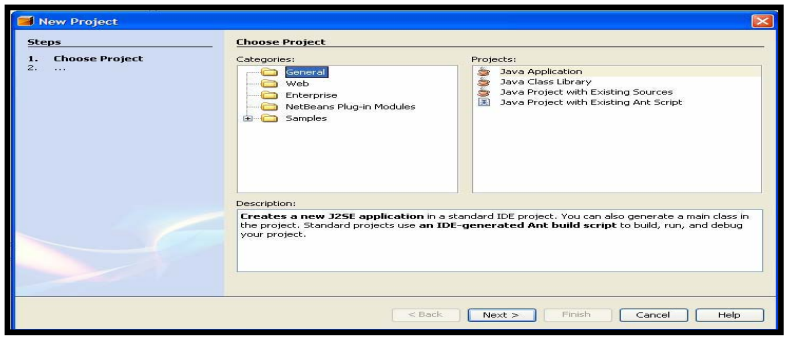
Figura 4.62: Paso 2 Crear proyecto



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ Aparecerá la primera ventana del asistente para crear un nuevo proyecto, figura. En esta ventana del asistente seleccionaremos el tipo de proyecto que deseamos crear. Como vamos a crear una aplicación de consola, seleccionaremos la opción **General** en el recuadro **Categories:** y la opción **Java Application** en el recuadro **Projets:**, y luego presionaremos el botón **Next>**.

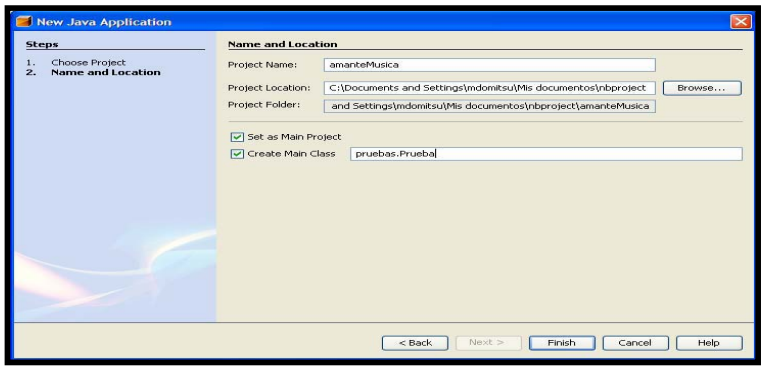
Figura 4.63: Paso 3 Crear proyecto



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ Aparecerá la segunda ventana del asistente para crear proyectos. En esta ventana seleccionaremos el nombre y la ubicación del proyecto, una vez seleccionado lo anterior dar clic en **Finish.**

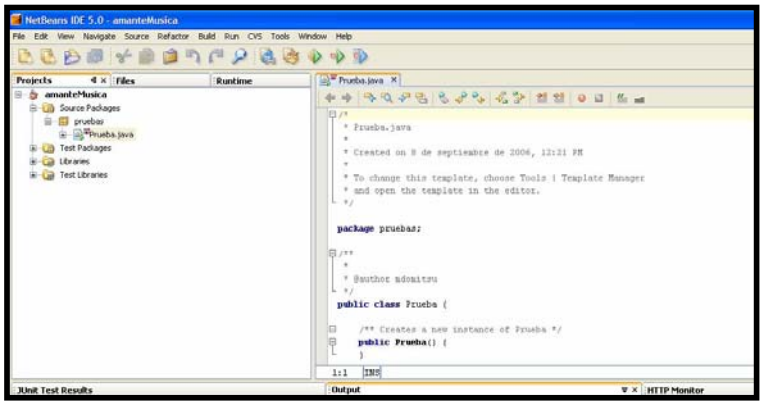
Figura 4.64: Paso 4 Crear proyecto



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ Desaparecerá el asistente para crear un nuevo proyecto y aparecerá lo mostrado en la figura siguiente. Del lado derecho aparece el editor de NetBeans con el esqueleto de la clase principal.

Figura 4.65: Paso 5 Crear proyecto

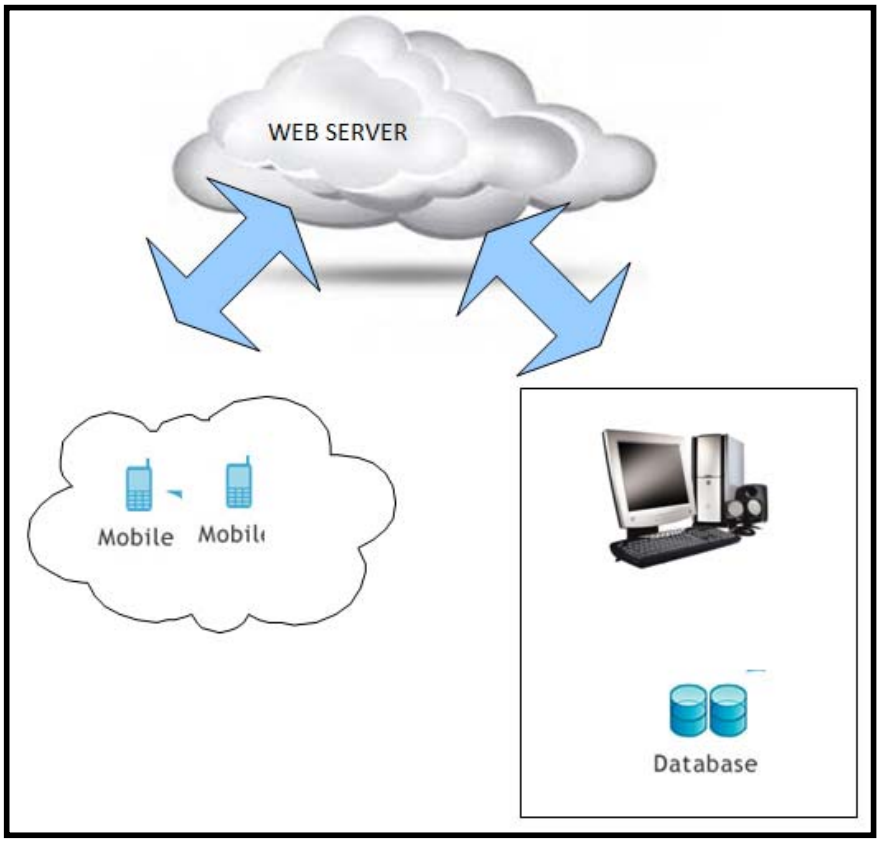


Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

Una vez creado el proyecto nuevo, en la figura anterior se presenta la consola donde se realizará el código fuente el mismo que se adjunta en los anexos.

El esquema final del proyecto queda como se muestra en la siguiente, donde claramente se identifican cuatro elementos macros del proyecto que son el servidor, la base de datos, el Web Server y el dispositivo móvil desde donde se va a acceder al proceso en general que se está simulando.

Figura 4.66: Esquema final de proyecto



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

La necesidad del proyecto es que se pueda visualizar desde cualquier punto del planeta la ejecución del proceso como tal, esto únicamente es posible presentando los datos a través de una red de alcance global tal y como es Internet. Definido esto y tomando en cuenta que se trata de un procedimiento en tiempo real, se han empleado para el desarrollo de la solución las tecnologías que actualmente dan mejores prestaciones para

este tipo de productos, como lo son php, http, java con sus derivaciones java script, j2me.

Para su consecución se segmentó al proyecto en tres etapas, considerando el ámbito en el que se desarrollan cada una, ejecución en la fábrica, servidor Web, presentación de datos. Esto permitió desarrollar una solución modular, es decir que aunque las etapas de la solución interactúan entre sí para la consecución del objetivo, cada una de ellas para su funciona de forma independiente, esto garantiza que si alguna de las partes dejase de funcionar de forma fortuita no sea terminado el proceso debido a esto.

A continuación se describen cada una de las etapas del proyecto:

Primera Etapa:

Esta etapa es la que comprende el desarrollo del proceso como tal, es decir es el lugar físico donde se lleva a cabo la extracción del jugo de manzana, es aquí donde están situados los equipos físicos empleados para el proceso completo como son: planta procesadora, equipos de control, servidores de BD, equipos de cómputo, entre otros.

Con lo que podemos decir que esta etapa será la encargada de generar los datos para el sistema, los cuales son almacenados en el servidor de BD de la planta, estos datos deben ser enviados a la red para su posterior presentación con este propósito se ha empleado lenguaje de programación y tecnología JAVA, para que sea enviada la información de la Base de Datos (BD) a la red. Ver código fuente en **ANEXO 2.**

//**Cabecera de datos**

```
<?xml version="1.0" encoding="UTF-8"?>
```

//**Inicio de la Aplicación a ejecutarse**

```
<application id="Application_ID" version="1.4"
xmlns="http://java.sun.com/xml/ns/j2ee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
http://java.sun.com/xml/ns/j2ee/application_1_4.xsd">
```

//**Nombre a desplegar**

```
    <display-name>
    SimuladorEAR</display-name>
```

    //**Indicación del archivo a ejecutar y ruta en la que se encuentra**

```
    <module id="WebModule_1315924743158">
        <web>
            <web-uri>Simulador.war</web-uri>
            <context-root>Simulador</context-root>
        </web>
    </module>
</application>
```

Segunda Etapa:

Esta etapa comprende al manejo de los datos enviados a la red, para lo que fue necesario contratar a un Proveedor de servicios Hosting y BD con lo que disminuimos y garantizamos la fiabilidad de la información a presentarse, ya que el hosting garantiza el poder ubicar al equipo desde cualquier lugar y el Servidor de BD nos permite almacenar la información de forma virtual para su envío con lo que si por cualquier razón se suspende o existe latencia de datos podemos seguir enviando información al dispositivo lector.

Luego de tener establecida la comunicación con el equipo y el respectivo respaldo de datos en nuestro servidor de BD virtual, el Servidor Web desarrollado procede a presentar los datos almacenados en la BD, enviándolos a través del protocolo http al dispositivo lector.

Como en toda comunicación, el canal a través del cual se envía la información es una de las partes sensibles y críticas para garantizar que los datos sean recibidos sin afectación alguna. Por lo cual se lo realiza a través de un canal javascript, esto por la gran fiabilidad que ofrece para estas soluciones, tanto en tiempo de respuesta como de integridad de datos muy necesarios entre las aplicaciones que utilicen el servicio Web para su funcionamiento. Ver el código fuente en el **ANEXO 3.**

Tercera Etapa:

Esta etapa comprende la visualización de los datos obtenidos desde la planta a través del Web Server, para su visualización y comunicación se han empleado tecnologías java micro edition (j2me) y ASP. Esto nos permite visualizar los datos de forma clara y

precisa por el tipo de tecnología empleada. Con esto se garantiza que la información pueda ser visualizada por cualquier equipo que cuente con los respectivos permisos, pueda acceder a Internet y que tenga soporte java

Para garantizar el acceso seguro a los datos se ha tomado en cuenta el acceso a través de validación por ingreso de clave, con lo que solo usuarios autenticados podrán ingresar a las prestaciones del sistema. Ver código fuente en **ANEXO 4.**

```
//Definición del nombre de la clase
package business;
//Importación de la clase dao ubicada en SimuladorDao
import dao.SimuladorDao;
//Declaración y codificación de la clase Simulador
public class Simulador {
        //Declaración de objeto llamado sim de tipo private de la clase dao ubicada en la propiedad SimuladorDao
        private dao.SimuladorDao sim;
        //Llamado a la clase Simulador() para su ejecución y presentación de datos
        public Simulador(){


        }
//Definición de la clase getData que será en donde se guarden los datos entregados o recopilados
        public String getData(){
//Instanciamiento de los objetos anteriormente Declarados
                sim = new SimuladorDao();
                return sim.getData();
```

```
        }
}
```

Las variables involucradas en el proceso que se está simulando se las crea en el Intouch, y son de tipo analógicas y digitales.

Las analógicas representan la cantidad de jugo que va a existir en cada uno de los tanques, y las digitales van a dar la señal para abrir o cerrar válvulas y para encender o apagar equipos, en un proceso real los valores de las variables serían generados por sensores y dispositivos de mando pero en éste caso se generan dentro del Intouch.

Base de datos. La base de datos se crea mediante el programa InSQL, en la cual se almacenan los valores de las variables que se crearon en el programa Intouch que son valores analógicos y digitales, los cuales indicarán los niveles de llenado de los tanques y si están encendidos o apagados los equipos respectivamente.

Ésta base de datos va a estar almacenada en la computadora portátil que a va a realizar la función de servidor, para así poder acceder a ella para ser enviada a la red lo cual se con la siguiente programación.

//**Cabecera de datos**

```
<?xml version="1.0" encoding="UTF-8"?>
```

//**Inicio de la Aplicación a ejecutarse**

```
<application id="Application_ID" version="1.4"
xmlns="http://java.sun.com/xml/ns/j2ee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
http://java.sun.com/xml/ns/j2ee/application_1_4.xsd">
```

//**Nombre a desplegar**

```
    <display-name>
    SimuladorEAR</display-name>
```

//**Indicación del archivo a ejecutar y ruta en la que se encuentra**

```
    <module id="WebModule_1315924743158">
        <web>
            <web-uri>Simulador.war</web-uri>
            <context-root>Simulador</context-root>
        </web>
    </module>
</application>
```

Servidor Web local. En ésta parte se lee y registra los datos es decir se maneja los datos enviados a la red, para lo cual se contrató un proveedor de servicios Hosting y base de datos.

El servidor web en el equipo se encuentra en el siguiente directorio:

C:\Archivos de programa\apache-tomcat6.0.33\webapps\simulador

La declaración de variables se realiza con las siguientes sentencias:

```
var fechas = new Array();
var variables=new Array();
var valores = new Array();
$.each(sim.data, function(i,item){
        fechas.push(item.fecha);
        variables.push(item.variable);
        valores.push(item.valor);
```

Para realizar el llamado al servidor Web y envío y recepción de datos se utiliza la siguiente dirección url:

"http://simulador.sc34.info/acciones.php?action=setData&fechas="+fechas+"&variables="+variables+"&valores="+valores,

A ésta dirección se le asigna un Hosting que es **simulador.sc.34.info** para poder acceder de una manera más rápida.

Para realizar el llamado y conexión a la base de datos local se utiliza las siguientes sentencias:

```
this.user = "apple";
this.password = "apple";
this.driver="com.microsoft.sqlserver.jdbc.SQLServerDriver";
this.url = "jdbc:sqlserver://localhost:1433;" +
        "databaseName=Runtime;";
```

Internet. En ésta etapa se almacena la base de datos virtual, el web Server, las imágenes en formato .gif y el código fuente de la aplicación.

Lo que se presenta en la página está guardado en el archivo de nombre **dao** que se encuentra en el siguiente directorio:

C:\Archivos de programa\apache-tomcat-6.0.33\webapps\simulador\WEB INF\classes\dao

Para lograr presentar la página de inicio de la aplicación la cual consta de dos cuadros de textos donde se deben ingresar un usuario y una contraseña respectivamente y un botón que dice **Enviar** el cual tiene la función de enviar y verificar la información antes ingresada y así poder acceder o no a la aplicación, se lo realiza mediante el siguiente segmento de código fuente:

```
<HTML>
<HEAD>
<TITLE>Proceso de obtencion de jugo de manzana</TITLE>
```

```
</HEAD>
<BODY>
<H2>Ingrese lo solicitado</H2>
<FORM ACTION='http://localhost:8080/ser_manzana/manzana' METHOD='POST'>
Usuario: <INPUT TYPE="TEXT" NAME="usuario" SIZE=8><BR>
Contraseña: <INPUT TYPE="TEXT" NAME="pass" SIZE=4><P><br>
<BR><INPUT TYPE="TEXT" NAME="tipo" SIZE=4 VALUE="1"><P><br>
<INPUT TYPE="SUBMIT" NAME="botonEnviar" VALUE="Enviar">
</FORM>
</BODY>
</HTML>
```

Figura 4.67: Pantalla principal de aplicación

Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

Una vez que se ingresa a la aplicación lo que se presenta es la etapa del proceso en la que se encuentra y el valor de llenado de cada tanque, para lo cual se dividió en cuatro etapas el proceso, la primera que representa el tanque de jugo extraído, la segunda el tanque del jugo pasteurizado, la tercera el jugo listo para envasar y la cuarta etapa representa el número de cajas de producto terminado.

La condición para que se presente la primera etapa en la aplicación es que la variable D1 sea igual a 1, tanto ésta variable como el valor de llenado del tanque se genera en el programa Intouch que es donde se esta simulando el proceso, las mismas que se almacenan en la base de datos para así poder acceder a las mismas.

Para presentar la segunda etapa en la aplicación la variable D2 debe ser igual a 1, para presentar la tercera etapa la variable D3 debe ser igual a 1 y para presentar la cuarta etapa la variable D4 debe ser igual a 4; de igual manera como en el caso de la primera etapa los valores de todas la variables se generan en el programa que se está simulando el proceso y se almacenan en la base de datos para poder acceder a sus valores.

Las condiciones para que se presente una u otra etapa del proceso, el valor de llenado de cada tanque y el valor de producto terminado, están dadas únicamente en el programa intouch que al realizar la simulación del proceso va a generar valores para cada una de las variables creadas y con dichos valores se crean las condiciones respectivas.

Ya en la aplicación lo que se hace es comparar los valores de las variables generados en la simulación para así poder saber la imagen de qué etapa del proceso se va a presentar y el valor de llenado del tanque que se presente en dicha etapa.

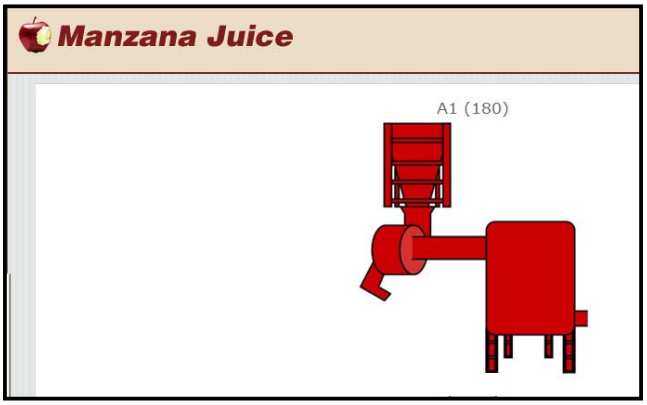
Los datos que se generen al simular el proceso automáticamente se cargan a la base de datos por lo tanto al realizar una consulta a ésta base de datos para realizar las comparaciones de los valores de las variables, siempre se va a tomar el último valor actualizado.

Por ejemplo para presentar la etapa uno se lo realiza con el siguiente segmento de código fuente:

```
if(iniciado.equals("1")){
                    if(consul_datos_man()){
                      presenta_datos(resp, tanque, llenado);
                    }else{
                       error(resp);
                    }
```

La siguiente figura muestra lo que presenta la aplicación en ésta etapa.

Figura 4.68: Primera etapa de proceso



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

Para las siguientes etapas el procedimiento es similar pero toca cambiar los nombres de las variables a comparar y el nombre de la etapa a presentar.

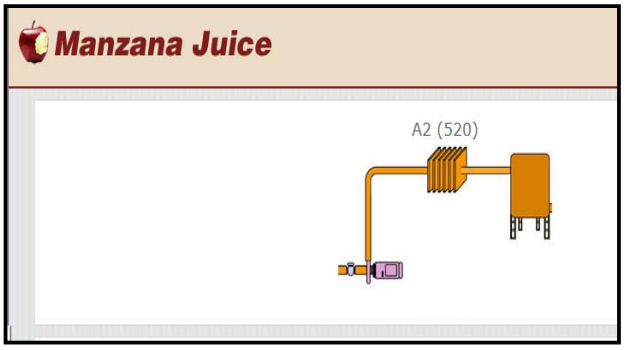
Para lograr que pase de presentar una etapa a la siguiente se utiliza el siguiente segmente de código fuente:

```
if(inicia_proceso()){//consultara si el proceso ha sido iniciado
                    norol(resp);//no posee los roles
                }else if(iniciado.equals("1")){
                    if(consul_datos_man()){
                      presenta_datos(resp, tanque, llenado);
                    }else{
                       error(resp);
                    }
```

El siguiente segmento de código fuente es para presentar la segunda etapa del proceso y la aplicación presentará la siguiente imagen.

```
                }else if(iniciado.equals("2")){
                  if(consul_datos_man()){
                    presenta_datos(resp, tanque, llenado);
                  }else{
                     error(resp);
                  }
```

Figura 4.69: Segunda etapa del proceso



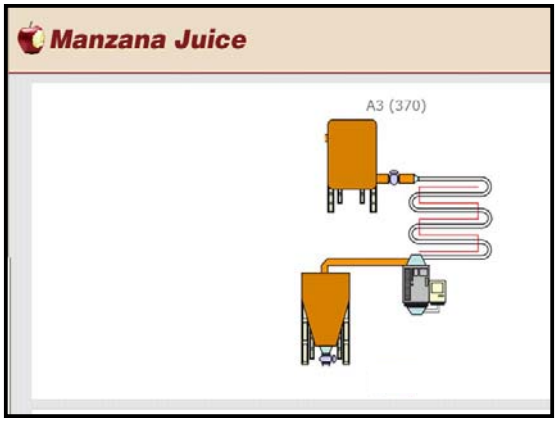
Fuente: César Zambrano / 2010

Elaborado por: César Zambrano / 2010

Para presentar la tercera etapa del proceso se usa el siguiente segmento de código y la aplicación presentara la imagen que se muestra a continuación.

```
}else if(iniciado.equals("3")){
 if(consul_datos_man()){
   presenta_datos(resp, tanque, llenado);
 }else{
    error(resp);
 }
```

Figura 4.70: Tercera etapa del proceso



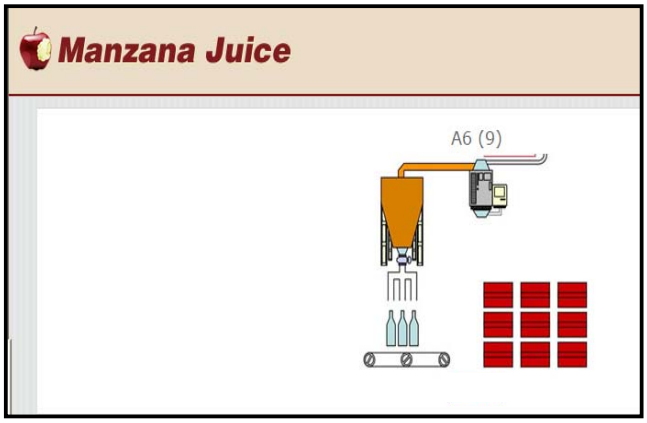
Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

Para presentar la cuarta y última etapa se utiliza el siguiente segmente de código fuente y la aplicación presenta la imagen que se muestra a continuación.

```
    }else if(iniciado.equals("4")){
    if(consul_datos_man()){
      presenta_datos(resp, tanque, llenado);
    }else{
       error(resp);

}//fin de inicia proceso
```

Figura 4.71: Cuarta etapa del proceso



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

Al momento que se presente la etapa en el cual está el proceso, la imagen de la misma va a estar cambiando de color y el valor de llenado del tanque va a estar actualizándose simultáneamente como lo hace en el programa simulador.

Las imágenes de cada una de las etapas que se van a presentar en la aplicación están almacenadas en el servidor web, las mismas que siempre van a estar disponibles cuando se cumplan las condiciones y se necesite visualizar cualquiera de las etapas del proceso.

Toda la programación se encuentra en un paquete llamado **dao** que se declara con las siguientes sentencias:

```
package dao;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
```

Con las siguientes sentencias se declara las clases que se encuentran en la página:

```
public class SimuladorDao {
        private String user = null;
        private String password = null;
        private String driver = null;
        private String url = null;
        private Connection con = null;
        private Statement stm = null;
        private ResultSet rs = null;
```

Para realizar la conexión a la base de datos se usa las siguientes sentencias:

```
public SimuladorDao(){
```

```
            this.user = "apple";
            this.password = "apple";
            this.driver="com.microsoft.sqlserver.jdbc.SQLServerDriver";
            this.url = "jdbc:sqlserver://localhost:1433;" +
                    "databaseName=Runtime;";
    }
    private void openConnection(){
            try{
                    //Registra el driver en la lista de drivers JDBC de la máquina
virtual
                    Class.forName(this.driver);
                    //Obtenemos la Connection (con)
                    this.con =
DriverManager.getConnection(url,this.user,this.password);
                    //la conexion nos da un Statement(stm)
                    this.stm =
this.con.createStatement(ResultSet.TYPE_FORWARD_ONLY,
ResultSet.CONCUR_UPDATABLE);
                    //this.stm =
this.con.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_UPDATABLE);
            }catch(java.lang.ClassNotFoundException e){
                    e.printStackTrace();
            }catch(SQLException e){
                    e.printStackTrace();
            };
    }
```

Las siguientes sentencias son para obtener respuestas al realizar la conexión:

```
public String getData(){
        String respuesta = "";
        openConnection();
        try{
```

Para llamar a las variables que se encuentran en la base de datos se utiliza las siguientes sentencias:

```
String sql="SELECT * FROM dbo.Vars_Analog_Digital WHERE valor!='0' AND
variable IN ('A1', 'A2', 'A3', 'A6')";
                this.rs = this.stm.executeQuery(sql);
                respuesta+="{'data':[ ";
                while (rs.next()){
                        respuesta+="{\"fecha\": \""+rs.getString("fecha")+"\", " +
                        "\"variable\":\""+rs.getString("variable")+
                        "\", \"valor\":\""+rs.getString("valor")+"\"},";
                }
                respuesta = respuesta.substring(0, respuesta.length()-1);
                respuesta+="]}";
        }catch(SQLException e){
                e.printStackTrace();
        }
        closeConnection();
```

```
            System.out.println(respuesta);
            return respuesta;
        }
        private void closeConnection(){
            try{
                //todos los recursos los liberamos en el finally
                if (rs!=null) rs.close();
                if (stm!=null) stm.close();
                if (con!=null) con.close();
            }catch(Exception e){
                e.printStackTrace();
            }
        }
}
```

Celular. En esta etapa se interpreta y se lee la información generada y almacenada en la base de datos, aquí se visualiza el proceso que se esta simulando.

Con las siguientes sentencias lo que se hace en importar o llamar lo que esta en **dao,** para poder visualizarlo en la página simulador.

```
package business;

import dao.SimuladorDao;
```

```
public class Simulador {
        private dao.SimuladorDao sim;

        public Simulador(){

        }
        public String getData(){
                sim = new SimuladorDao();
                return sim.getData();
        }
}
```

4.7 Características del dispositivo móvil

El dispositivo móvil que se usa para este proyecto es el Nokia 5800 XpressMusic es un teléfono inteligente de gama media. El 5800 cuenta con una pantalla táctil de 3.2″ con una resolución de 640×360 píxeles, la interfaz ha sido optimizada para manejarlo con una sola mano. Incluye una cámara de 3.2 mega píxeles, acelerómetro y soporta conexiones 3.5G.

La conexión a Internet también es muy importante, para lo que incorpora **conectividad HSDPA** y **Wi-Fi,** con ellas, podremos utilizar las diferentes aplicaciones que vienen preinstaladas, como el **navegador web**, el cual ofrece **soporte para Flash**, la aplicación de correo.

Incluye también un receptor GPS con soporte de A-GPS, que combinado con el software de Nokia Maps, que viene preinstalado, nos permitirá utilizarlo como un navegador GPS convencional, lo que viene facilitado por el tamaño de su pantalla.

4.8 Pruebas de funcionamiento

Una vez que esta lista la aplicación se procede a realizar las pruebas de funcionamiento de la misma para lo cual se detallan los pasos a seguir:

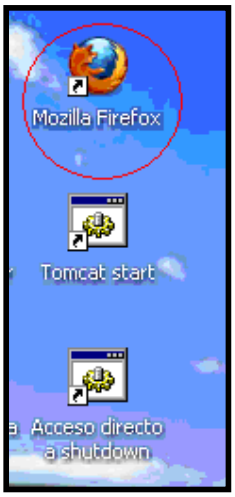
- ✓ Primero se abre la aplicación creada en Intouch y se la manda a correr, el procedimiento de esto ya está detallado anteriormente.
- ✓ Abrir el software InSQL y realizar los pasos para arrancar el servidor que ya se indicaron con anterioridad.
- ✓ Abrir el software Microsoft SQL y abrir la tabla que se creó para verificar los valores de las variables.
- ✓ Subir el servidor Tomcat, para lo cual se creo un acceso directo y solo es necesario dar doble clic sobre él.

Figura 4.72: Acceso directo Tomcat

Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

- ✓ Arrancar la base de datos virtual que está publicada en el Web Server, de igual manera se tiene un acceso directo para acceder a la página.

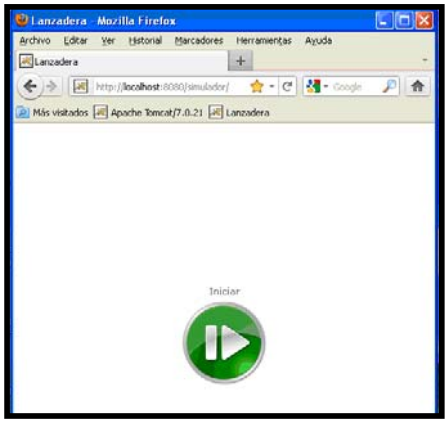
Figura 4.73: Acceso directo página de Web Server



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ La página se presenta como en la siguiente figura y solo de debe dar clic en el boton que se indica.

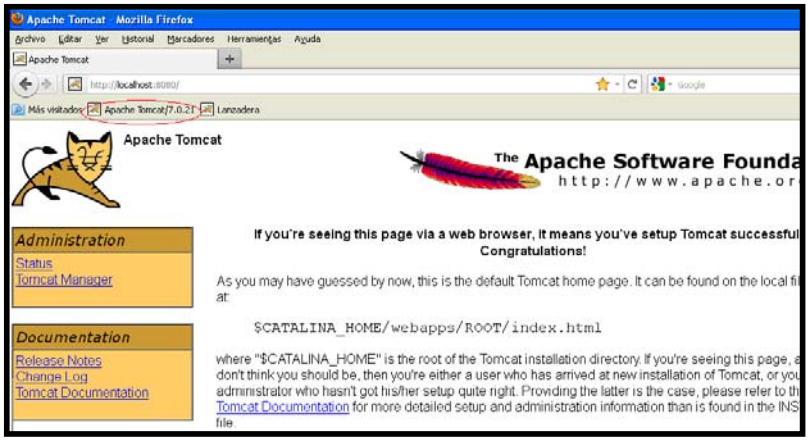
Figura 4.74: Arranque de base de datos virtual



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ Para comprobar si se conecto con el Web Server se da clic en la ventana que se indica en la figura y debe presentar lo siguiente.

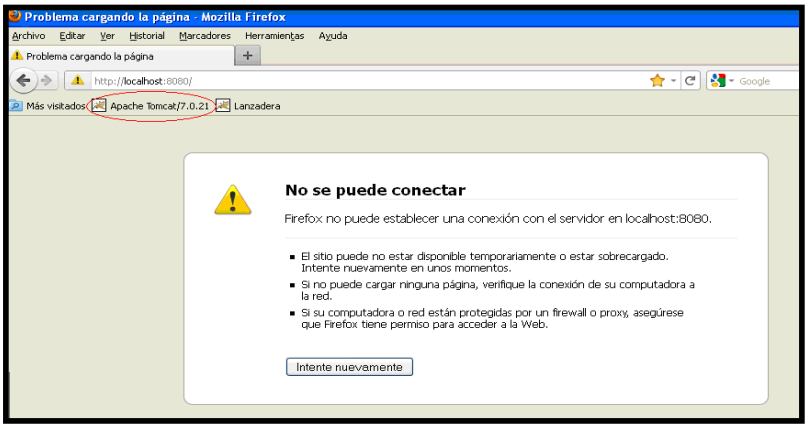
Figura 4.75: Web Server conectado



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ En caso de que no se conecte se presentará una pantalla como se muestra en la figura.

Figura 4.76 Web Server desconectado



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

✓ Para acceder a la aplicación desde el celular, se abre la misma donde presenta la pantalla como se muestra en la figura y se ingresa el usuario que es (**admin.)** y la contraseña que es **(123).**

Figura 4.77: Pantalla de inicio en el celular

Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

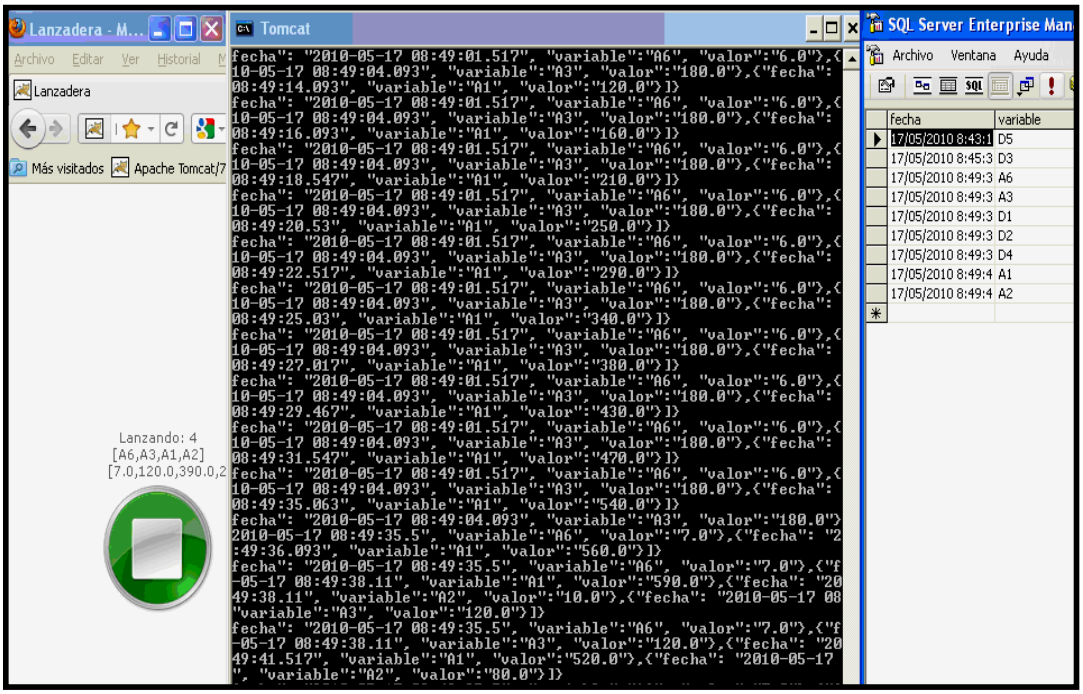
- ✓ Una vez ingresado esos datos ya podemos visualizar la etapa del proceso en la que se encuentra.

Figura 4.78: Aplicación abierta

Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

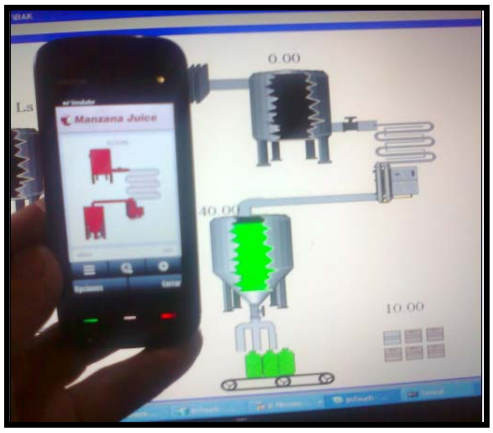
- ✓ Una vez realizado los pasos anteriores ya se puede verificar la transmisión de datos, a continuación se presenta varias figuras donde podemos verificarlo.
- ✓ Las figuras que se presentan a continuación son fotografías que muestran la comunicación que existe entre los software, base de datos, web Server y la aplicación del celular.

Figura 4.79: Muestra 1 de comunicación



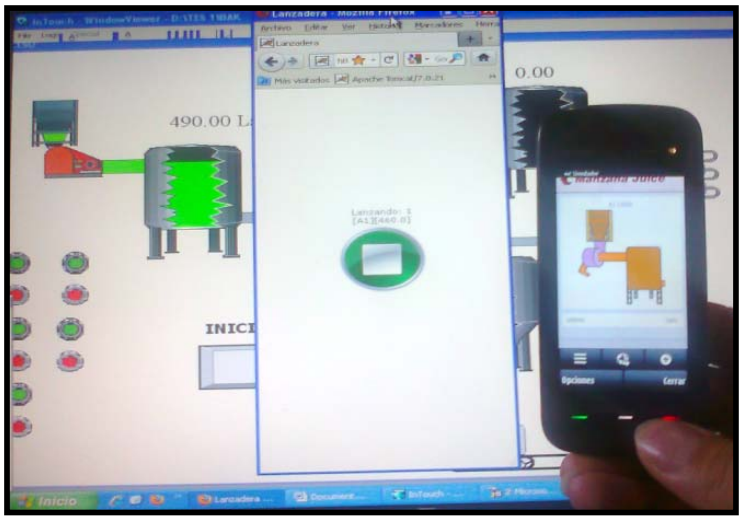
Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

Figura 4.80: Muestra 2 de comunicación



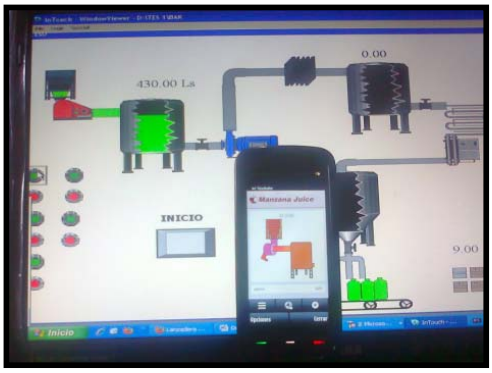
Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

Figura 4.81: Muestra 3 de comunicación



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

Figura 4.82: Muestra 4 de comunicación



Fuente: César Zambrano / 2010
Elaborado por: César Zambrano / 2010

CAPÍTULO V

CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

- ✓ Una vez concluido este trabajo de investigación se logró conocer y demostrar que un dispositivo móvil, como es un celular en este caso, nos brinda características adicionales de las habituales de llamar y enviar mensajes de texto y multimedia, como en este caso el soporte de tecnología JAVA y esto nos permite aprovechar al máximo dichas cualidades.

- ✓ Para la simulación del proceso de extracción de jugo de manzana se implementó un sistema SCADA con el software InTouch**.**

- ✓ En el desarrollo de este proyecto fue necesario crear una base de datos industrial para el proceso que se simuló, ésta se la realizó con la ayuda del software IndustrialSQL de Wonderware, la cual se instaló en la misma máquina donde se implementó el HMI del sistema SCADA.

- ✓ Para realizar la aplicación que se cargó en el celular se la acudió al uso del software Netsbeans el cual es un entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java.

- ✓ Queda demostrado que con la implementación de nuevas técnicas y tecnologías en lo que es el campo de la automatización industrial, se obtiene muchas ventajas como en este caso la optimización del uso de un celular.

5.2 Recomendaciones

- ✓ En la realización de este proyecto la base de datos se la creó en un ordenador, pero lo recomendable es que se lo haga en un servidor ya que éste equipo tiene las características adecuadas para garantizar y optimizar el funcionamiento de la misma.

- ✓ Se recomienda tener en cuenta que en este trabajo la base de datos no se la realizó en un servidor, por lo que se debe realizar ciertos pasos que se detallan en este documento antes de realizar las pruebas de funcionamiento.

- ✓ Se recomienda siempre tener conectada la llave física de la licencia del software INTOUCH, que es un dispositivo USB para así lograr el buen funcionamiento de la aplicación.

BIBLIOGRAFÍA

1. SERGIO GÁLVEZ ROJAS; LUCAS ORTEGA DIAZ- JAVA 2 Micro Edition

2. WIKIPEDIA ENCICLOPEDIA LIBRE, Sistema SACDA, http://es.

3. wikipedia.org/wiki/SCADA

4. ING. CARLOS DE CASTRO LOZANO E ING. CRISTÓBAL ROMERO
5. MORALES, Introducción a SCADA

6. ING. HENRY MENDIBURU DÍAZ, sistemas SCADA, http://hamd.galeon.com

7. INVENSYS, InTouch HMI 9.0-Basic Course, revision C febrero 2005

8. INVENSYS SYSTEMS, ActiveFactory User´s Guide, revision C 2006

9. INVENSYS SYSTEMS, IndustrialSQL Server Historian Administration Guide,
10. revision C 2005

11. PEDRO HERRARTE, Introducción a SQL, www.devjoker.com

12. SOFTWARE SOLUTIONS-www.wonderware-benelux.com/documents/products
13. WONDERWARE FACTORY SUITE, Industrial SQL Serve, Historian
14. Glossary. Invensys Systems, Inc. 2006
15. http://ww2.estg.ipleiria.pt/~ftadeu/TwidoSoft_V20/Documentacao/spa/TwdoSW.pdf

16. http://www.clikear.com/manuales/sql/sql1.aspx

17. http://www.infomanuales.com/Manuales/SQLServer/SQLServer.asp

18. http://www.equiposdidacticos.com/pdf/catalogos/Manual_Twido.pdf

19. http://ww2.estg.ipleiria.pt/~ftadeu/TwidoSoft_V20/Documentacao/spa/TwdoSW.pdf

20. http://www.logiteksa.com/wonderware/InSQL18.htm

21. http://es.wikipedia.org/wiki/SCADA

22. SISTEMAS SCADAS/ Ing. Henry Mendiburu Díaz/http/hamd.galeon.com

23. http://leonardo.uncu.edu.ar/catedras/electronica/archivos/Tema9_Scada.pdf

24. www.itba.edu.ar/capis/epg-tesis-y-tf/lopezfiguerola-tfe.pdf

25. http://bieec.epn.edu.ec:8180/dspace/bitstream/123456789/1134/4/T10991CAP%202.pdf

26. http://www.equiposdidacticos.com/pdf/catalogos/Manual_Twido.pdf

27. http://materias.fi.uba.ar/6722/apunteplc.pdf

28. http://www.disinel.com/Disinel%20Web/Wonderware/intouch_01.htm#seguridad

29. http://www.disinel.com/Disinel%20Web/Wonderware/insql_01.htm

30. http://www.logic-control.com/media/InSQL90Admin.pdf

31. http://bieec.epn.edu.ec:8180/dspace/bitstream/123456789/1134/3/T10991CAP%203.pdf

32. WONDERWARE PRODUCTOS, www.logiteksa.com/.../insql_activefactory

33. PEDRO HERRARTE, Introducción a las bases de datos, www.devjoker.com

ANEXOS

ANEXO 1

GLOSARIO DE TÉRMINOS

SCADA

Aacrónimo de Supervisory Control and Data Acquisition (Control Supervisor y Adquisición de Datos).

HMI

Interfase hombre máquina, (Human Machine Interface).

MTU

Unidad maestra.

PLC

Controlador Lógico Programable.

INTOUCH

Un paquete de software utilizado para crear aplicaciones.

DDE

Intercambio de datos dinámicos. (Dynamic Data Exchange).

INSQL

IndustrialSQL Server (InSQL) es una base de Datos de Wonderware, diseñada para la recolección de información de los Sistemas de Producción, PLCs, etc., a una alta velocidad y con una gran capacidad de almacenamiento.

BASE DE DATOS

Una Base de Datos es un conjunto de archivos interrelacionados que contienen información importante de un proceso.

DBMS

El DBMS (DataBase Management System)es un Software que permite definir, construir y manipular la información que posea la base de datos.

J2ME (Java 2 Platform, Micro Edition)

Java 2 Platform, Micro Edition (J2ME): Esta versión de Java está enfocada a la aplicación de la tecnología Java en dispositivos electrónicos con capacidades computacionales y gráficas muy reducidas, tales como teléfonos móviles, PDAs o electrodomésticos inteligentes.

Máquinas Virtuales J2ME

Una máquina virtual de Java (JVM) es un programa encargado de interpretar código intermedio (bytecode) de los programas Java precompilados a código máquina ejecutable por la plataforma, efectuar las llamadas pertinentes al sistema operativo subyacente y observar las reglas de seguridad y corrección de código definidas para el lenguaje Java.

KVM

Corresponde con la Máquina Virtual más pequeña desarrollada por Sun. Su nombre KVM proviene de Kilobyte (haciendo referencia a la baja ocupación de memoria, entre 40Kb y 80Kb). Se trata de una implementación de Máquina Virtual reducida y especialmente orientada a dispositivos con bajas capacidades computacionales y de memoria.

CVM

La CVM (Compact Virtual Machine) ha sido tomada como Máquina Virtual Java de referencia para la configuración CDC y soporta las mismas características que la Máquina Virtual de J2SE.

ANEXO 2

CÓDIGO FUENTE

//**Cabecera de datos**

```
<?xml version="1.0" encoding="UTF-8"?>
```

//**Inicio de la Aplicación a ejecutarse**

```
<application id="Application_ID" version="1.4"
xmlns="http://java.sun.com/xml/ns/j2ee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
http://java.sun.com/xml/ns/j2ee/application_1_4.xsd">
```

//**Nombre a desplegar**

```
    <display-name>
    SimuladorEAR</display-name>
```

//**Indicación del archivo a ejecutar y ruta en la que se encuentra**

```
    <module id="WebModule_1315924743158">
        <web>
            <web-uri>Simulador.war</web-uri>
            <context-root>Simulador</context-root>
        </web>
    </module>
</application>
```

ANEXO 3

CÓDIGO FUENTE DEL SERVIDOR WEB

//Cabecera de datos

```
<?xml version="1.0" encoding="UTF-8"?>

<xmi:XMI xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
xmlns:resources.j2c="http://www.ibm.com/websphere/appserver/schemas/5.0/resources.j2c.xmi"
xmlns:resources.jdbc="http://www.ibm.com/websphere/appserver/schemas/5.0/resources.jdbc.xmi"
xmlns:resources.jms="http://www.ibm.com/websphere/appserver/schemas/5.0/resources.jms.xmi"
xmlns:resources.mail="http://www.ibm.com/websphere/appserver/schemas/5.0/resources.mail.xmi"
xmlns:resources.url="http://www.ibm.com/websphere/appserver/schemas/5.0/resources.url.xmi">
```

//Definición de la cadena de conexión

```
  <resources.jdbc:JDBCProvider xmi:id="builtin_jdbcprovider" name="Derby JDBC Provider (XA)" description="Built-in Derby JDBC Provider (XA)"
providerType="Derby JDBC Provider (XA)"
implementationClassName="org.apache.derby.jdbc.EmbeddedXADataSource"
xa="true">
```

//ubicación de la Clase en la que se encuentra el driver

```
    <classpath>${DERBY_JDBC_DRIVER_PATH}/derby.jar</classpath>

    <factories xmi:type="resources.jdbc:DataSource"
xmi:id="DataSource_1315924746262" name="DefaultEJBTimerDataSource"
jndiName="jdbc/DefaultEJBTimerDataSource" description="Default data source for WebSphere EJB Timer Service" category="default"
```

authMechanismPreference="BASIC_PASSWORD"
relationalResourceAdapter="builtin_rra" statementCacheSize="10"
datasourceHelperClassname="com.ibm.websphere.rsadapter.DerbyDataStoreHelper"
>

<propertySet xmi:id="J2EEResourcePropertySet_1315924746262">

//**Definición de las propiedades para el string de conexión y definición de los posibles a emplearse si falla alguno (pool de conexión, driver)la Base de datos**

<resourceProperties xmi:id="J2EEResourceProperty_1315924746262"
name="databaseName" type="java.lang.String"
value="${USER_INSTALL_ROOT}/databases/EJBTimers/${SERVER}/EJBTimer
DB" description="Location of Derby default database for the EJB Timer Service."
required="true"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746263"
name="shutdownDatabase" type="java.lang.String" value=""/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746264"
name="dataSourceName" type="java.lang.String" value=""/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746265"
name="description" type="java.lang.String" value=""/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746266"
name="connectionAttributes" type="java.lang.String" value="upgrade=true"
description="Allows the Derby database to be upgraded."/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746267"
name="createDatabase" type="java.lang.String" value=""/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746268"
name="enableMultithreadedAccessDetection" type="java.lang.Boolean"
value="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746269"
name="preTestSQLString" type="java.lang.String" value=""/>

</propertySet>

<connectionPool xmi:id="ConnectionPool_1315924746262"
connectionTimeout="1800" maxConnections="30" minConnections="1"

reapTime="180" unusedTimeout="1800" agedTimeout="0"
purgePolicy="EntirePool"/>

</factories>

</resources.jdbc:JDBCProvider>

<resources.jms:JMSProvider xmi:id="builtin_jmsprovider" name="WebSphere
JMS Provider" description="V5 Default Messaging Provider"
externalInitialContextFactory="" externalProviderURL=""/>

<resources.jms:JMSProvider xmi:id="builtin_mqprovider" name="WebSphere MQ
JMS Provider" description="WebSphere MQ Messaging Provider"
externalInitialContextFactory="" externalProviderURL=""/>

<resources.j2c:J2CResourceAdapter xmi:id="builtin_rra" name="WebSphere
Relational Resource Adapter" description="Built-in Relational Resource Adapter for
WebSphere Persistence" archivePath="${WAS_LIBS_DIR}/rsadapter.rar">

<classpath>${WAS_LIBS_DIR}/rsadapter.rar</classpath>

<propertySet xmi:id="J2EEResourcePropertySet_1315924746263"/>

<deploymentDescriptor xmi:id="Connector_1315924746262"
displayName="WS_RdbResourceAdapter" vendorName="IBM" specVersion="1.5"
eisType="RRA" version="6.0">

<icons xmi:id="IconType_1315924746262" smallIcon="rdb_small_icon.jpg"
largeIcon="rdb_large_icon.jpg"/>

<displayNames xmi:id="DisplayName_1315924746262"
value="WS_RdbResourceAdapter"/>

<displayNames xmi:id="DisplayName_1315924746263"
value="WS_RdbResourceAdapter"/>

<descriptions xmi:id="Description_1315924746262" value="IBM Relational
ResourceAdapter"/>

<license xmi:id="License_1315924746262" required="false">

<descriptions xmi:id="Description_1315924746263" value="IBM Relational
ResourceAdapter"/>

</license>

<resourceAdapter xmi:id="ResourceAdapter_1315924746262" transactionSupport="NoTransaction" reauthenticationSupport="false" resourceAdapterClass="com.ibm.ws.rsadapter.spi.WSResourceAdapterImpl">

<outboundResourceAdapter xmi:id="OutboundResourceAdapter_1315924746262" reauthenticationSupport="false" transactionSupport="XATransaction">

<connectionDefinitions xmi:id="ConnectionDefinition_1315924746262" managedConnectionFactoryClass="com.ibm.ws.rsadapter.spi.WSManagedConnectionFactoryImpl" connectionFactoryInterface="javax.resource.cci.ConnectionFactory" connectionFactoryImplClass="com.ibm.ws.rsadapter.cci.WSRdbConnectionFactoryImpl" connectionInterface="javax.resource.cci.Connection" connectionImplClass="com.ibm.ws.rsadapter.cci.WSRdbConnectionImpl">

<configProperties xmi:id="ConfigProperty_1315924746262" name="ConnectionFactoryType" type="java.lang.Integer" value="2">

<descriptions xmi:id="Description_1315924746264" value="A constant indicating the type of connection factory: WSJdbcDataSource (1) or WSRdbConnectionFactory (2)"/>

</configProperties>

</connectionDefinitions>

//**Forma en la que se validará el password a ingresarse**

<authenticationMechanisms xmi:id="AuthenticationMechanism_1315924746262" authenticationMechanismType="BasicPassword" credentialInterface="javax.resource.spi.security.PasswordCredential">

<descriptions xmi:id="Description_1315924746265" value="BasicPassword authentication"/>

</authenticationMechanisms>

<authenticationMechanisms xmi:id="AuthenticationMechanism_1315924746263" authenticationMechanismType="Kerbv5" credentialInterface="javax.resource.spi.security.GenericCredential">

<descriptions xmi:id="Description_1315924746266" value="Kerbv5 Authentication"/>

</authenticationMechanisms>

</outboundResourceAdapter>

</resourceAdapter>

</deploymentDescriptor>

//

<connectionDefTemplateProps xmi:id="ConnectionDefTemplateProps_1315924746262" ConnectionDefinition="ConnectionDefinition_1315924746262"/>

</resources.j2c:J2CResourceAdapter>

<resources.mail:MailProvider xmi:id="builtin_mailprovider" name="Built-in Mail Provider" description="The built-in mail provider">

<protocolProviders xmi:id="ProtocolProvider_1315924746262" protocol="smtp" classname="com.sun.mail.smtp.SMTPTransport" type="TRANSPORT"/>

<protocolProviders xmi:id="ProtocolProvider_1315924746263" protocol="pop3" classname="com.sun.mail.pop3.POP3Store" type="STORE"/>

<protocolProviders xmi:id="ProtocolProvider_1315924746264" protocol="imap" classname="com.sun.mail.imap.IMAPStore" type="STORE"/>

</resources.mail:MailProvider>

<resources.url:URLProvider xmi:id="URLProvider_1" name="Default URL Provider" streamHandlerClassName="unused" protocol="unused"/>

<resources.j2c:J2CResourceAdapter xmi:id="J2CResourceAdapter_1250665795749" name="SIB JMS Resource Adapter" description="Default messaging provider" archivePath="${WAS_INSTALL_ROOT}/installedConnectors/sib.api.jmsra.rar">

<classpath>${WAS_INSTALL_ROOT}/installedConnectors/sib.api.jmsra.rar</classpath>

<propertySet xmi:id="J2EEResourcePropertySet_1315924746264"/>

<deploymentDescriptor xmi:id="Connector_1315924746263" displayName="WebSphere Default Messaging Provider" vendorName="IBM" specVersion="1.5" eisType="JMS Provider" version="0.3">

//**Definición de la ubicación de los archivos a utilizarse**

<displayNames xmi:id="DisplayName_1315924746264" value="WebSphere Default Messaging Provider"/>

<displayNames xmi:id="DisplayName_1315924746265" value="WebSphere Default Messaging Provider"/>

<displayNames xmi:id="DisplayName_1315924746266" value="WebSphere Default Messaging Provider"/>

<displayNames xmi:id="DisplayName_1315924746267" value="WebSphere Default Messaging Provider"/>

<displayNames xmi:id="DisplayName_1315924746268" value="WebSphere Default Messaging Provider"/>

//

<resourceAdapter xmi:id="ResourceAdapter_1315924746263" transactionSupport="NoTransaction" reauthenticationSupport="false" resourceAdapterClass="com.ibm.ws.sib.api.jmsra.impl.JmsJcaResourceAdapterImpl ">

<outboundResourceAdapter xmi:id="OutboundResourceAdapter_1315924746263" reauthenticationSupport="false" transactionSupport="XATransaction">

<connectionDefinitions xmi:id="ConnectionDefinition_1315924746263" managedConnectionFactoryClass="com.ibm.ws.sib.api.jmsra.impl.JmsJcaManaged QueueConnectionFactoryImpl" connectionFactoryInterface="javax.jms.QueueConnectionFactory" connectionFactoryImplClass="com.ibm.ws.sib.api.jms.impl.JmsQueueConnFactoryI

mpl" connectionInterface="javax.jms.QueueConnection"
connectionImplClass="com.ibm.ws.sib.api.jms.impl.JmsQueueConnectionImpl">

//**Definición de los objetos a utilizarse**

<configProperties xmi:id="ConfigProperty_1315924746263" name="BusName"
type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746264"
name="ClientID" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746265"
name="UserName" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746266"
name="Password" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746267"
name="NonPersistentMapping" type="java.lang.String"
value="ExpressNonPersistent"/>

<configProperties xmi:id="ConfigProperty_1315924746268"
name="PersistentMapping" type="java.lang.String" value="ReliablePersistent"/>

<configProperties xmi:id="ConfigProperty_1315924746269"
name="ReadAhead" type="java.lang.String" value="Default"/>

<configProperties xmi:id="ConfigProperty_1315924746270" name="Target"
type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746271"
name="TargetType" type="java.lang.String" value="BusMember"/>

<configProperties xmi:id="ConfigProperty_1315924746272"
name="TargetSignificance" type="java.lang.String" value="Preferred"/>

<configProperties xmi:id="ConfigProperty_1315924746273"
name="RemoteProtocol" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746274"
name="TargetTransportChain" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746275"
name="ProviderEndpoints" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746276" name="ConnectionProximity" type="java.lang.String" value="Bus"/>

<configProperties xmi:id="ConfigProperty_1315924746277" name="TemporaryQueueNamePrefix" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746278" name="ShareDataSourceWithCMP" type="java.lang.Boolean" value="false"/>

</connectionDefinitions>

<connectionDefinitions xmi:id="ConnectionDefinition_1315924746264" managedConnectionFactoryClass="com.ibm.ws.sib.api.jmsra.impl.JmsJcaManagedTopicConnectionFactoryImpl" connectionFactoryInterface="javax.jms.TopicConnectionFactory" connectionFactoryImplClass="com.ibm.ws.sib.api.jms.impl.JmsTopicConnFactoryImpl" connectionInterface="javax.jms.TopicConnection" connectionImplClass="com.ibm.ws.sib.api.jms.impl.JmsTopicConnectionImpl">

<configProperties xmi:id="ConfigProperty_1315924746279" name="BusName" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746280" name="ClientID" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746281" name="UserName" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746282" name="Password" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746283" name="NonPersistentMapping" type="java.lang.String" value="ExpressNonPersistent"/>

<configProperties xmi:id="ConfigProperty_1315924746284" name="PersistentMapping" type="java.lang.String" value="ReliablePersistent"/>

<configProperties xmi:id="ConfigProperty_1315924746285" name="DurableSubscriptionHome" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746286" name="ReadAhead" type="java.lang.String" value="Default"/>

<configProperties xmi:id="ConfigProperty_1315924746287" name="Target" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746288" name="TargetType" type="java.lang.String" value="BusMember"/>

<configProperties xmi:id="ConfigProperty_1315924746289" name="TargetSignificance" type="java.lang.String" value="Preferred"/>

<configProperties xmi:id="ConfigProperty_1315924746290" name="RemoteProtocol" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746291" name="TargetTransportChain" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746292" name="ProviderEndpoints" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746293" name="ConnectionProximity" type="java.lang.String" value="Bus"/>

<configProperties xmi:id="ConfigProperty_1315924746294" name="TemporaryTopicNamePrefix" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746295" name="ShareDataSourceWithCMP" type="java.lang.Boolean" value="false"/>

<configProperties xmi:id="ConfigProperty_1315924746296" name="ShareDurableSubscriptions" type="java.lang.String" value="InCluster"/>

</connectionDefinitions>

<connectionDefinitions xmi:id="ConnectionDefinition_1315924746272" managedConnectionFactoryClass="com.ibm.ws.sib.api.jmsra.impl.JmsJcaManagedConnectionFactoryImpl" connectionFactoryInterface="javax.jms.ConnectionFactory" connectionFactoryImplClass="com.ibm.ws.sib.api.jms.impl.JmsConnFactoryImpl" connectionInterface="javax.jms.Connection" connectionImplClass="com.ibm.ws.sib.api.jms.impl.JmsConnectionImpl">

<configProperties xmi:id="ConfigProperty_1315924746297" name="BusName" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746298" name="ClientID" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746299" name="UserName" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746300" name="Password" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746301" name="NonPersistentMapping" type="java.lang.String" value="ExpressNonPersistent"/>

<configProperties xmi:id="ConfigProperty_1315924746302" name="PersistentMapping" type="java.lang.String" value="ReliablePersistent"/>

<configProperties xmi:id="ConfigProperty_1315924746303" name="DurableSubscriptionHome" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746304" name="ReadAhead" type="java.lang.String" value="Default"/>

<configProperties xmi:id="ConfigProperty_1315924746305" name="Target" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746306" name="TargetType" type="java.lang.String" value="BusMember"/>

<configProperties xmi:id="ConfigProperty_1315924746307" name="TargetSignificance" type="java.lang.String" value="Preferred"/>

<configProperties xmi:id="ConfigProperty_1315924746308" name="RemoteProtocol" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746309" name="TargetTransportChain" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746310" name="ProviderEndpoints" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746311" name="ConnectionProximity" type="java.lang.String" value="Bus"/>

<configProperties xmi:id="ConfigProperty_1315924746312" name="TemporaryQueueNamePrefix" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746313" name="TemporaryTopicNamePrefix" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746314" name="ShareDataSourceWithCMP" type="java.lang.Boolean" value="false"/>

<configProperties xmi:id="ConfigProperty_1315924746315" name="ShareDurableSubscriptions" type="java.lang.String" value="InCluster"/>

</connectionDefinitions>

<authenticationMechanisms
xmi:id="AuthenticationMechanism_1315924746272"
authenticationMechanismType="BasicPassword"
credentialInterface="javax.resource.spi.security.PasswordCredential"/>

</outboundResourceAdapter>

<inboundResourceAdapter
xmi:id="InboundResourceAdapter_1315924746272">

<messageAdapter xmi:id="MessageAdapter_1315924746272">

<messageListeners xmi:id="MessageListener_1315924746272"
messageListenerType="javax.jms.MessageListener">

<activationSpec xmi:id="ActivationSpec_1315924746272"
activationSpecClass="com.ibm.ws.sib.api.jmsra.impl.JmsJcaActivationSpecImpl">

<requiredConfigProperties
xmi:id="RequiredConfigPropertyType_1315924746272" name="destination"/>

<requiredConfigProperties
xmi:id="RequiredConfigPropertyType_1315924746273" name="destinationType"/>

<requiredConfigProperties
xmi:id="RequiredConfigPropertyType_1315924746274" name="busName"/>

</activationSpec>

</messageListeners>

</messageAdapter>

</inboundResourceAdapter>

//**Definición y utilización de las características a emplearse en el software**

<adminObjects xmi:id="AdminObject_1315924746272"
adminObjectInterface="javax.jms.Queue"
adminObjectClass="com.ibm.ws.sib.api.jms.impl.JmsQueueImpl">

<configProperties xmi:id="ConfigProperty_1315924746316"
name="QueueName" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746317" name="DeliveryMode" type="java.lang.String" value="Application"/>

<configProperties xmi:id="ConfigProperty_1315924746318" name="TimeToLive" type="java.lang.Long"/>

<configProperties xmi:id="ConfigProperty_1315924746319" name="Priority" type="java.lang.Integer"/>

<configProperties xmi:id="ConfigProperty_1315924746320" name="ReadAhead" type="java.lang.String" value="AsConnection"/>

<configProperties xmi:id="ConfigProperty_1315924746321" name="BusName" type="java.lang.String"/>

</adminObjects>

<adminObjects xmi:id="AdminObject_1315924746273" adminObjectInterface="javax.jms.Topic" adminObjectClass="com.ibm.ws.sib.api.jms.impl.JmsTopicImpl">

<configProperties xmi:id="ConfigProperty_1315924746322" name="TopicSpace" type="java.lang.String" value="Default.Topic.Space"/>

<configProperties xmi:id="ConfigProperty_1315924746323" name="TopicName" type="java.lang.String"/>

<configProperties xmi:id="ConfigProperty_1315924746324" name="DeliveryMode" type="java.lang.String" value="Application"/>

<configProperties xmi:id="ConfigProperty_1315924746325" name="TimeToLive" type="java.lang.Long"/>

<configProperties xmi:id="ConfigProperty_1315924746326" name="Priority" type="java.lang.Integer"/>

<configProperties xmi:id="ConfigProperty_1315924746327" name="ReadAhead" type="java.lang.String" value="AsConnection"/>

<configProperties xmi:id="ConfigProperty_1315924746328" name="BusName" type="java.lang.String"/>

</adminObjects>

</resourceAdapter>

</deploymentDescriptor>

<connectionDefTemplateProps
xmi:id="ConnectionDefTemplateProps_1315924746272"
ConnectionDefinition="ConnectionDefinition_1315924746263">

<resourceProperties xmi:id="J2EEResourceProperty_1315924746272"
name="BusName" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746273"
name="ClientID" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746274"
name="ConnectionProximity" type="java.lang.String" value="Bus"
required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746275"
name="durableSubscriptionHome" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746276"
name="NonPersistentMapping" type="java.lang.String"
value="ExpressNonPersistent" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746277"
name="Password" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746278"
name="PersistentMapping" type="java.lang.String" value="ReliablePersistent"
required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746279"
name="ProviderEndpoints" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746280"
name="ReadAhead" type="java.lang.String" value="Default" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746281"
name="RemoteProtocol" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746282"
name="TargetTransportChain" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746283"
name="ShareDataSourceWithCMP" type="java.lang.Boolean" value="false"
required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746284"
name="shareDurableSubscriptions" type="java.lang.String" value="AsCluster"
required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746285" name="Target" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746286" name="TargetSignificance" type="java.lang.String" value="Preferred" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746287" name="TargetType" type="java.lang.String" value="BusMember" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746288" name="TemporaryQueueNamePrefix" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746289" name="temporaryTopicNamePrefix" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746290" name="UserName" type="java.lang.String" required="false"/>

</connectionDefTemplateProps>

<connectionDefTemplateProps xmi:id="ConnectionDefTemplateProps_1315924746273" ConnectionDefinition="ConnectionDefinition_1315924746264">

<resourceProperties xmi:id="J2EEResourceProperty_1315924746291" name="BusName" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746292" name="ClientID" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746293" name="ConnectionProximity" type="java.lang.String" value="Bus" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746294" name="DurableSubscriptionHome" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746295" name="NonPersistentMapping" type="java.lang.String" value="ExpressNonPersistent" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746296" name="Password" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746297" name="PersistentMapping" type="java.lang.String" value="ReliablePersistent" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746298" name="ProviderEndpoints" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746299" name="ReadAhead" type="java.lang.String" value="Default" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746300" name="RemoteProtocol" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746301" name="TargetTransportChain" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746302" name="ShareDataSourceWithCMP" type="java.lang.Boolean" value="false" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746303" name="ShareDurableSubscriptions" type="java.lang.String" value="InCluster" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746304" name="Target" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746305" name="TargetSignificance" type="java.lang.String" value="Preferred" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746306" name="TargetType" type="java.lang.String" value="BusMember" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746307" name="temporaryQueueNamePrefix" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746308" name="TemporaryTopicNamePrefix" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746309" name="UserName" type="java.lang.String" required="false"/>

</connectionDefTemplateProps>

<connectionDefTemplateProps
xmi:id="ConnectionDefTemplateProps_1315924746274"
ConnectionDefinition="ConnectionDefinition_1315924746272">

<resourceProperties xmi:id="J2EEResourceProperty_1315924746310"
name="BusName" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746311"
name="ClientID" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746312"
name="ConnectionProximity" type="java.lang.String" value="Bus"
required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746313"
name="DurableSubscriptionHome" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746314"
name="NonPersistentMapping" type="java.lang.String"
value="ExpressNonPersistent" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746315"
name="Password" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746316"
name="PersistentMapping" type="java.lang.String" value="ReliablePersistent"
required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746317"
name="ProviderEndpoints" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746318"
name="ReadAhead" type="java.lang.String" value="Default" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746319"
name="RemoteProtocol" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746320"
name="TargetTransportChain" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746321"
name="ShareDataSourceWithCMP" type="java.lang.Boolean" value="false"
required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746322"
name="ShareDurableSubscriptions" type="java.lang.String" value="InCluster"
required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746323" name="Target" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746324" name="TargetSignificance" type="java.lang.String" value="Preferred" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746325" name="TargetType" type="java.lang.String" value="BusMember" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746326" name="TemporaryQueueNamePrefix" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746327" name="TemporaryTopicNamePrefix" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746328" name="UserName" type="java.lang.String" required="false"/>

</connectionDefTemplateProps>

<activationSpecTemplateProps xmi:id="ActivationSpecTemplateProps_1315924746272" activationSpec="ActivationSpec_1315924746272">

<resourceProperties xmi:id="J2EEResourceProperty_1315924746329" name="acknowledgeMode" type="java.lang.String" value="Auto-acknowledge" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746330" name="busName" type="java.lang.String" required="true"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746331" name="clientId" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746332" name="destination" type="javax.jms.Destination" required="true"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746333" name="destinationType" type="java.lang.String" required="true"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746334" name="durableSubscriptionHome" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746335" name="maxBatchSize" type="java.lang.Integer" value="1" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746336" name="maxConcurrency" type="java.lang.Integer" value="10" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746337" name="messageSelector" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746338" name="password" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746339" name="readAhead" type="java.lang.String" value="Default" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746340" name="shareDataSourceWithCMP" type="java.lang.Boolean" value="false" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746341" name="shareDurableSubscriptions" type="java.lang.String" value="InCluster" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746342" name="subscriptionDurability" type="java.lang.String" value="NonDurable" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746343" name="subscriptionName" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746344" name="target" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746345" name="targetSignificance" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746346" name="targetTransportChain" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746347" name="targetType" type="java.lang.String" required="false"/>

<resourceProperties xmi:id="J2EEResourceProperty_1315924746348" name="userName" type="java.lang.String" required="false"/>

<complexPropertySet xmi:id="J2EEResourcePropertySet_1315924746272">

<resourceProperties xmi:id="J2EEResourceProperty_1315924746349" name="destination" type="javax.jms.Destination" required="false"/>

</complexPropertySet>

```
    </activationSpecTemplateProps>

    <adminObjectTemplateProps
xmi:id="AdminObjectTemplateProps_1315924746272"
adminObject="AdminObject_1315924746272">

      <properties xmi:id="J2EEResourceProperty_1315924746350"
name="BusName" type="java.lang.String" required="false"/>

      <properties xmi:id="J2EEResourceProperty_1315924746351"
name="DeliveryMode" type="java.lang.String" value="Application"
required="false"/>

      <properties xmi:id="J2EEResourceProperty_1315924746352"
name="forwardRoutingPath" type="[Ljava.lang.String;" required="false"/>

      <properties xmi:id="J2EEResourceProperty_1315924746353" name="Priority"
type="java.lang.Integer" required="false"/>

      <properties xmi:id="J2EEResourceProperty_1315924746354"
name="QueueName" type="java.lang.String" required="false"/>

      <properties xmi:id="J2EEResourceProperty_1315924746355"
name="ReadAhead" type="java.lang.String" value="AsConnection"
required="false"/>

      <properties xmi:id="J2EEResourceProperty_1315924746356"
name="reverseRoutingPath" type="[Ljava.lang.String;" required="false"/>

      <properties xmi:id="J2EEResourceProperty_1315924746357"
name="TimeToLive" type="java.lang.Long" required="false"/>

    </adminObjectTemplateProps>

    <adminObjectTemplateProps
xmi:id="AdminObjectTemplateProps_1315924746273"
adminObject="AdminObject_1315924746273">

      <properties xmi:id="J2EEResourceProperty_1315924746358"
name="BusName" type="java.lang.String" required="false"/>

      <properties xmi:id="J2EEResourceProperty_1315924746359"
name="DeliveryMode" type="java.lang.String" value="Application"
required="false"/>

      <properties xmi:id="J2EEResourceProperty_1315924746360"
name="forwardRoutingPath" type="[Ljava.lang.String;" required="false"/>
```

```
    <properties xmi:id="J2EEResourceProperty_1315924746361" name="Priority"
type="java.lang.Integer" required="false"/>

    <properties xmi:id="J2EEResourceProperty_1315924746362"
name="ReadAhead" type="java.lang.String" value="AsConnection"
required="false"/>

    <properties xmi:id="J2EEResourceProperty_1315924746363"
name="reverseRoutingPath" type="[Ljava.lang.String;" required="false"/>

    <properties xmi:id="J2EEResourceProperty_1315924746364"
name="TimeToLive" type="java.lang.Long" required="false"/>

    <properties xmi:id="J2EEResourceProperty_1315924746365"
name="TopicName" type="java.lang.String" required="false"/>

    <properties xmi:id="J2EEResourceProperty_1315924746366"
name="TopicSpace" type="java.lang.String" value="Default.Topic.Space"
required="false"/>

   </adminObjectTemplateProps>

  </resources.j2c:J2CResourceAdapter>

</xmi:XMI>
```

ANEXO **4**

CÓDIGO FUENTE APLICACION

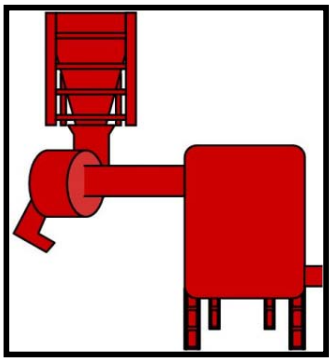
```
//Definición del nombre de la clase
package business;
//Importación de la clase dao ubicada en SimuladorDao
import dao.SimuladorDao;
//Declaración y codificación de la clase Simulador
public class Simulador {
        //Declaración de objeto llamado sim de tipo private de la clase dao ubicada en la propiedad SimuladorDao
        private dao.SimuladorDao sim;
        //Llamado a la clase Simulador() para su ejecución y presentación de datos
        public Simulador(){


        }
//Definición de la clase getData que será en donde se guarden los datos entregados o recopilados
        public String getData(){
//Instanciamiento de los objetos anteriormente Declarados
                sim = new SimuladorDao();
                return sim.getData();
        }
}
```

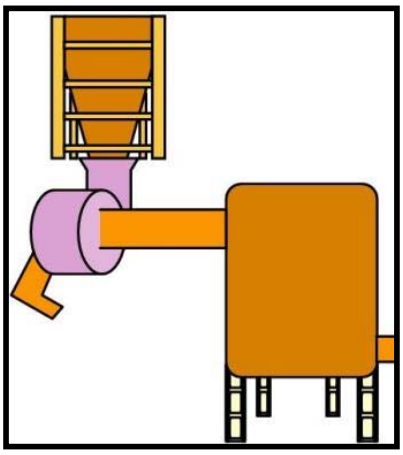
ANEXO 5

FIGURAS QUE SE PRESENTAN EN EL CELULAR

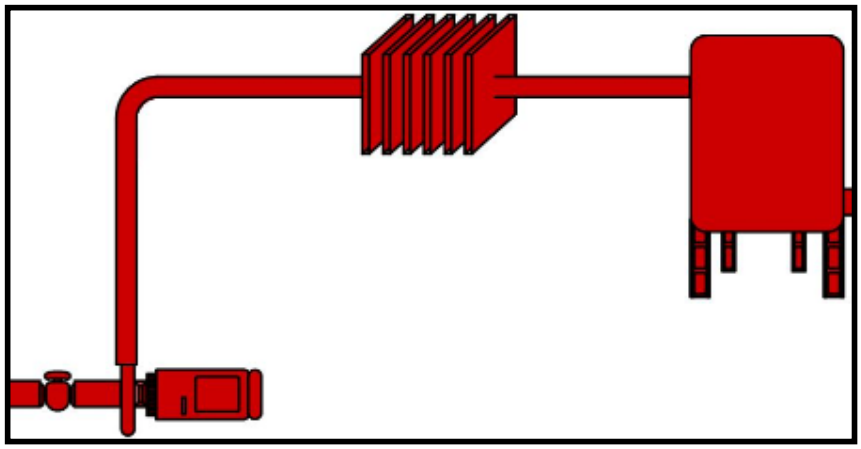
Etapa 1 del proceso cuando está en estado inactivo.



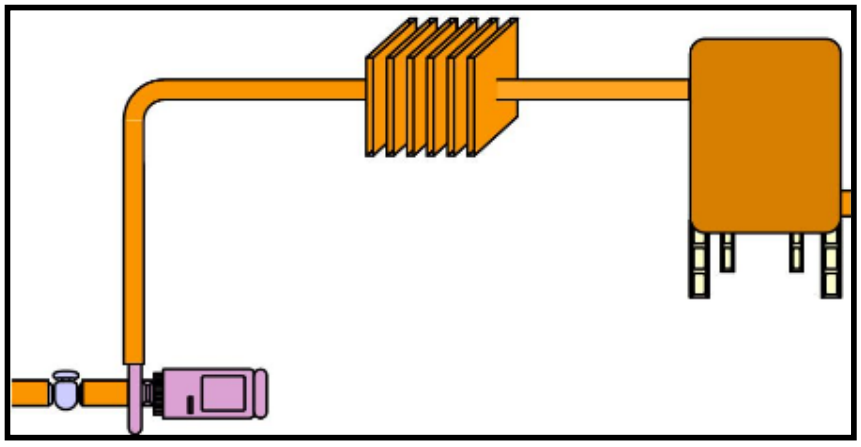
Etapa 1 del proceso cuando está en estado activo.



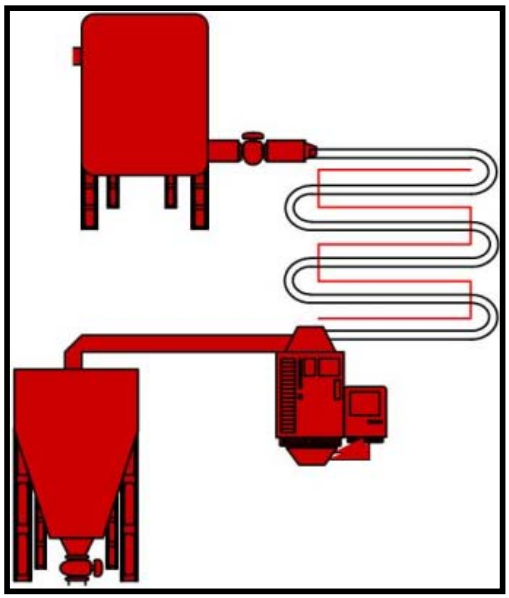
Etapa 2 del proceso en estado inactivo



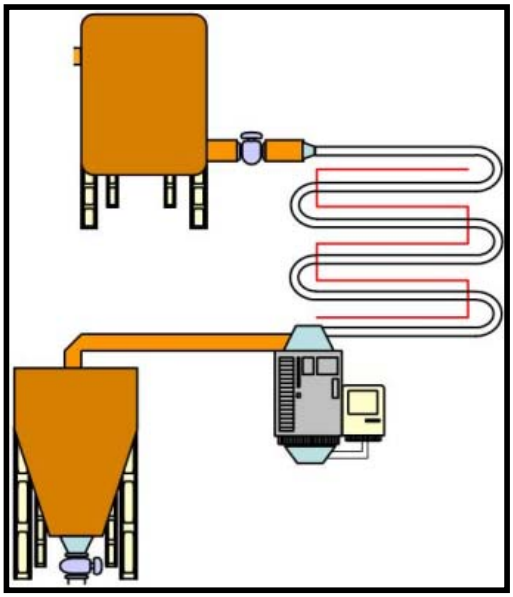
Etapa 2 del proceso en estado activo



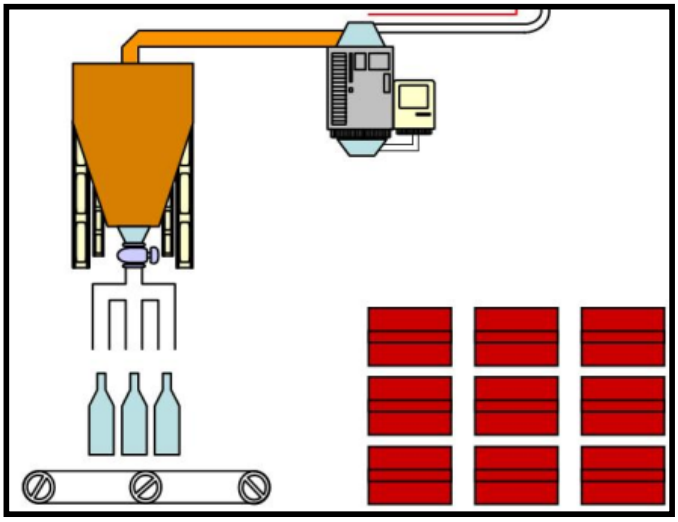
Etapa 3 del proceso en estado inactivo



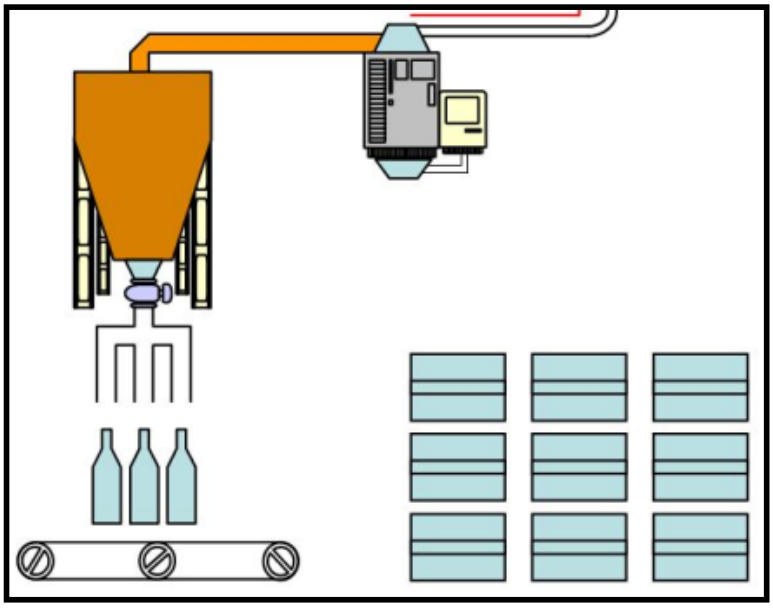
Etapa 3 del proceso en estado activo



Etapa 4 del proceso en estado inactivo

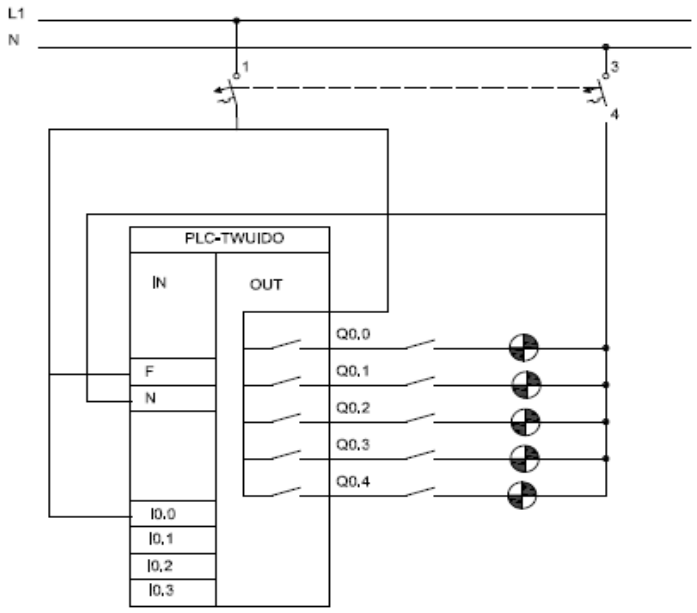


Etapa 4 del proceso en estado activo



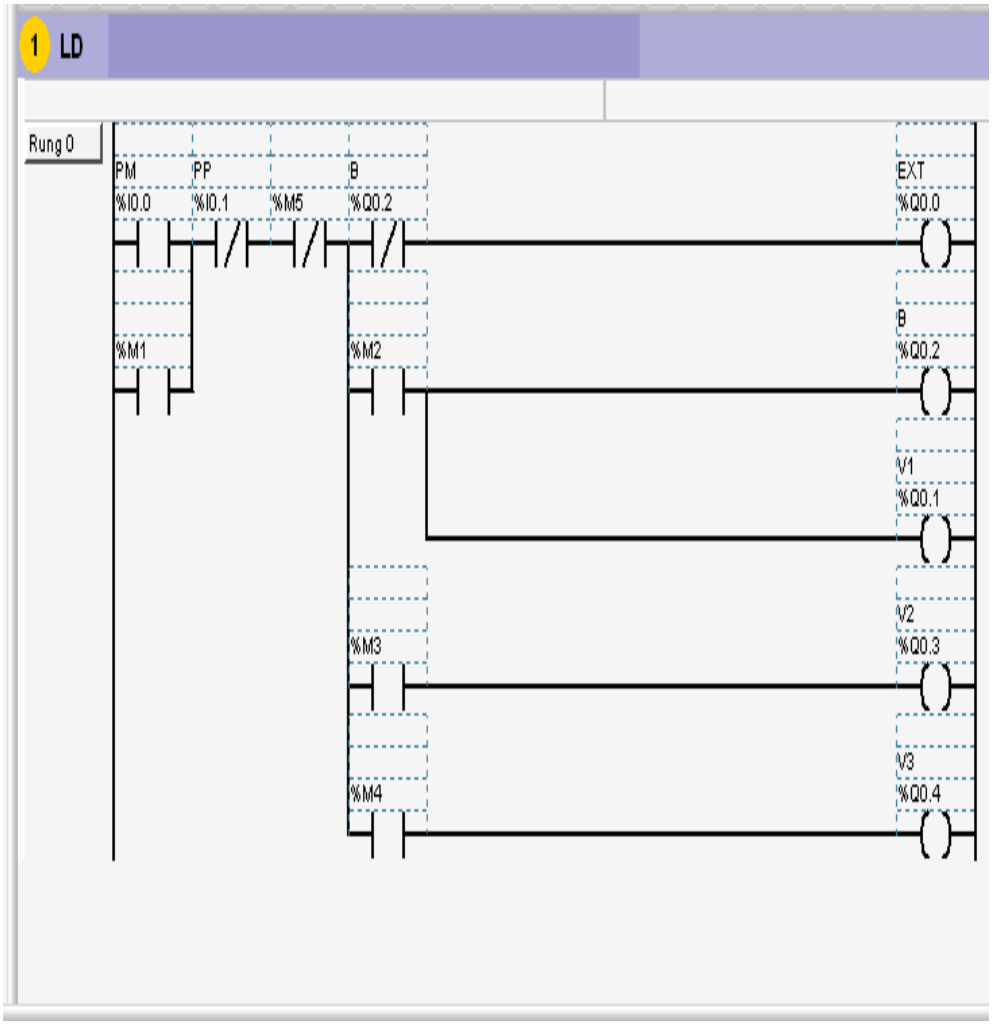
ANEXO **6**

DIAGRAMA DE CONEXIÓN DEL PLC



UNIVERSIDAD TECNOLOGICA EQUINOCCIAL		
Dibujo: C. Zambrano	ANEXO	Proyecto: Sistema Scada
Fecha: Octubre 2011		Escala:
Lamina:		Aprobo: Ing. V. Armijos

PROGRAMACIÓN DEL PLC



ANEXO 7

En este anexo se encuentran los ejecutables de la aplicación creada, los cuales están un CD que se entregará conjuntamente con el documento de la tesis.