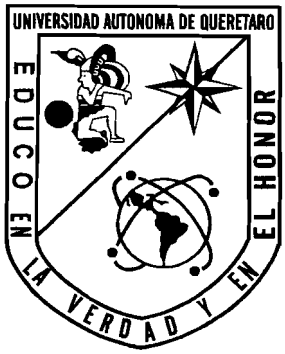


# UNIVERSIDAD AUTÓNOMA DE QUERÉTARO

## FACULTAD DE INFORMÁTICA

# AUTOMATIZACIÓN DE PROCESOS EXPERIMENTALES EN OVOCITOS Y FOLÍCULOS DE XENOPUS.

# TESIS

QUE PARA OBTENER EL TÍTULO DE

## INGENIERO EN COMPUTACIÓN

PRESENTA:

ALEJANDRO VALDEZ CHAPARRO

DIRIGIDO POR:

Dr. CARLOS SALDAÑA GUTIÉRREZ

C.U. SANTIAGO DE QUERÉTARO, QRO. AGOSTO 2007


*Universidad Autónoma de Querétaro*


*Instituto de Neurobiología UNAM campus Juriquilla*

# *Automatización de procesos experimentales en ovocitos y folículos de Xenopus.*

*Opción de titulación por tesis individual.*
*que para obtener el título de:*
***Ingeniero en Computación.***

***Director de de Tesis:***
*Dr. Carlos Saldaña Gutiérrez*

***Sinodales:***
*M.S.I. Gerardo Rodríguez Rojano*
*M en C. Ruth Angélica Rico Hernández.*
*L.I Ernesto Rubalcava Durán.*
*ISC. Elisa Morales Portillo.*

***Presentada por:***
*Alejandro Valdez Chaparro*
*Expediente: 93246*

*Santiago de Querétaro, Querétaro, Agosto de 2007.*

## Dedicatoria

*Este trabajo lo dedico a mis padres y mi hermano quienes me han apoyado de manera incondicional toda mi vida. A mis amigos y compañeros, especialmente Lule y Carlos por todas sus enseñanzas, criticas y sobre todo por su valiosa amistad, gracias también a los miembros del laboratorio en quienes encontré además de compañeros de labores grandes amigos.*

## Agradecimientos

*Agradezco a la Universidad Autónoma de Querétaro por la formación adquirida en esa gran institución, a la Universidad Nacional Autónoma de México y el Instituto de Neurobiolología Campus Juriquilla por la oportunidad brindad para el desarrollo de este proyecto, agradezco también el patrocinio otorgado por el Programa de Apoyo a Proyectos de Investigación e Innovación Tecnológica (PAPIIT, IN225107).*

*De manera muy especial agradezco la accesoria brindada por el Dr. Carlos Saldaña Gutiérrez, el Dr. Rogelio Arellano Ostoa, la Dra. Edith Garay Rojas, el M en CC. Alberto Lara Ruvalcaba, la I.S.C Elisa Morales Portillo, al M.S.I Gerardo Rodríguez Rojano, la M.C Ruth Angélica Rico Hernández y al L.I Ernesto Rubalcava Duràn. Gracias también a Horacio Ramírez Huerta y todos los miembros del laboratorio de Neurofisiología Celular por el apoyo brindado en la realización de este proyecto.*

## Prefacio.

Como parte de mi formación académica, tuve la oportunidad de realizar mi estancia profesional en el Instituto de Neurobiología UNAM-UAQ campus Juriquilla, en el área de cómputo, donde entre las tareas que me fueron asignadas se encontraba el soporte técnico a los distintos laboratorios que componen el instituto. En este lapso de tiempo pude acercarme al laboratorio de Neurofisiología Celular en donde el Dr. Carlos Saldaña Gutiérrez me cuestiono sobre la factibilidad de construir un sistema similar a un producto comercial que deseaba adquirir, el cual permitía automatizar el procedimiento experimental de perfusión celular.

Planteado este cuestionamiento decidí profundizar en las características de los productos comerciales que automatizan dicho procedimiento, en esta investigación encontré que la gama de productos ofertados es relativamente pequeña y que en su totalidad se trata de productos de origen extranjero con altos costos en su adquisición e importación. Así mismo, analizadas las características técnicas y funcionales encontré que dichos sistemas se encontraban construidos con tecnologías de bajo costo, por lo que a mi parecer existía una sobre valoración de los productos generada por la escasez de competitividad en el ramo.

Este panorama me motivo a iniciar el desarrollo de esta tesis, lo que me llevo a incursionar en un área de trabajo completamente ajena a mi perfil profesional, pero que me permitiría aplicar los conocimientos adquiridos en el aula para la creación de infraestructura novedosa para un sector de suma importancia para nuestro país, y al mismo tiempo sentar un precedente estatal al incursionar en el diseño y fabricación de equipos experimentales para las ciencias biológicas.

## Estructura de la tesis.

La presente tesis se encuentra organizada en tres partes de las cuales se derivan cuatro capítulos. La primera parte resume los conceptos básicos necesarios para comprender el material expuesto en este documento, la segunda parte presenta procedimientos realizados y finalmente se presentan los resultados y conclusiones así como las actividades propuestas para el futuro.

En el **Capítulo 1** se definen los conceptos teóricos más importantes, se detalla el proceso experimental así como los conceptos básicos para su entendimiento. Descrito este proceso se analizan los productos comerciales existentes con el fin de identificar sus ventajas y desventajas, también se describen los conceptos básicos de electrónica, programación y control que fueron necesarios para el desarrollo de este proyecto.

**Planteamiento general del desarrollo.**

En este apartado se plantea a manera de hipótesis un esquema de desarrollo general, este apartado tiene por objetivo presentar al lector un panorama general del trabajo realizado y que es descrito en los subsecuentes capítulos. Así mismo se describe la justificación para la adopción de este modelo y a manera de auto critica serán exponen las ventajas y desventajas del planteamiento.

En el **Capítulo 2** se presenta el trabajo realizado en el diseño y construcción de las partes mecánicas y eléctricas del sistema, diversos parámetros en su diseño son especificados y se detallan las pruebas realizadas para garantizar su correcto funcionamiento.

En el **Capítulo 3** se muestra el trabajo realizado para el desarrollo y programación del software, se detallan los flujos funcionales del software y la caracterización del proceso experimental, así como los algoritmos, métodos y procedimientos utilizados para su programación. De igual manera, se detallan procedimientos fundamentales en la temporización del software y la inclusión de parámetros necesarios para realizar ajustes de retardo.

En el **Capítulo 4** se presenta el prototipo del sistema desarrollado y un análisis de su funcionamiento. También se valora el costo de su desarrollo y finalmente se detallaran las problemáticas encontradas en el desarrollo de este proyecto, se propone también el trabajo a futuro y las posibles mejoras.

## Abreviaturas.

**PC:**
Acrónimo Inglés de "Personal Computer", o Computadora Personal
**TTL:**
Acrónimo Inglés de Transistor-Transistor Logic o Lógica Transistor a Transistor.
**CPU:**
Acrónimo Inglés de Central Processing Unit, o Unidad de Procesamiento Central.
**API:**
Acrónimo Inglés de Application Programming Interface o Interfaz de Programación de Aplicaciones.
**SPP:**
Acrónimo Inglés de Standard Parallel Port, o Puerto paralelo estándar.
**EPP:**
Acrónimo Inglés de Enhaced Parallel Port o Puerto paralelo mejorado.
**ECP:**
Acrónimo Inglés de Extended Capability Port o Puerto paralelo de capacidad extendida.
**Front End:**
Unidad que manipula y ordena los datos e instrucciones que serán enviadas al procesador.
**RAM:**
Acrónimo Inglés de Random Acces Memory o Memoria de Acceso Aleatorio.
**USB:**
Acrónimo Inglés de Universal Serial Bus, o Bus Serie Universal.
**RS232:**
Norma internacional conocida como Electronic Industries Alliance RS-232C para el intercambio de datos binarios.

# Tabla de contenido

Prefacio. ....4
Introducción. ....13
Justificación. ....14
Objetivo general. ....15
Objetivos específicos. ....15
Capítulo 1 Generalidades. ....16
1.0 Introducción. ....16
1.1 Farmacología. ....16
1.2 Mecanismos de acción Fármaco-Receptor. ....16
1.3 Cuantificación de la interacción receptor – fármaco. ....17
1.4 Curva Dosis Respuesta. ....18
Figura 1: Curva Dosis – Respuesta ....19
1.5 Análisis funcional de receptores. ....19
Figura 2: Ovocitos de la Rana Xenopus *laevis* ....20
1.6 Técnica de control de voltaje. ....20
Figura 3: Técnica de control de voltaje ....21
Figura 4: Set de experimentación Laboratorio Neurofisiología Celular ....22
1.7 Sistemas Comerciales. ....23
1.8 Complete 8-Channel Fast Miniature Perfusion System ....23
1.8.1 Fabricante: Bioscience Tools. ....23
1.8.2 Descripción. ....23
1.8.3 Características físicas. ....23
Figura 5: Sistema de perfusión Bioscentific tools. ....24
1.8.4 Características eléctricas. ....24
1.8.5 Funcionalidades. ....24
1.8.6 Costos. ....25
Tabla 1: Precios sistemas Bioscience Tools. ....25
1.9 ValveBank® 8 Teflon Perfusion Systems ....25
1.9.1 Fabricante: AutoMate Scientific. ....25
1.9.2 Descripción. ....25
1.9.3 Características físicas. ....25
Figura 6: Sistema de perfusión ValveBank®8 Teflón. ....26
1.9.4 Características eléctricas. ....26
1.9.5 Funcionalidades. ....26
1.9.6 Costos. ....27
Tabla 2: Precios sistemas AutoMate Scientific. ....27
1.10 Ventajas. ....27
1.11 Desventajas. ....27
1.12 Tecnología TTL. ....29
Tabla3: Características y versiones de la familia TTL. ....29
1.13 Solenoides. ....30
Figura 7: Válvula de solenoide ....30
1.14 Comportamiento funcional del solenoide. ....30
1.15 El puerto paralelo ....31
Tabla 4: Puerto paralelo, características. ....31

1.16 Nomenclatura del conector. ....32
Tabla 5: Nomenclatura del conector puerto paralelo. ....32
Figura 8: Diagrama conexiones Puerto Paralelo. ....32
1.17 Acceso y control del puerto paralelo. ....32
Tabla 6: Direcciones comunes para puerto paralelo. ....32
1.18 Inpout32.dll. ....33
1.19 Ejecución multiproceso. ....33
Figura 9: Multiprocesamiento Virtual ....34
Figura 10: Procesamiento Multihilo. ....35
1.20 Ejecución en tiempo real. ....36
Resultados y procedimientos realizados en esta tesis. ....38
Requerimientos y recursos. ....38
Tabla 7: Recursos del Set1 Laboratorio de Neurofisiología Celular. ....39
Tabla 8: Recursos del Set2 Laboratorio de Neurofisiología Celular. ....39
Tabla 9: Recursos del Set3 Laboratorio de Neurofisiología Celular. ....39
Retos y problemáticas. ....40
Hipótesis. ....40
Ventajas. ....41
Desventajas. ....41
Figura 11: Esquema general del desarrollo. ....41
Capitulo 2. Diseño y desarrollo de sistema. ....42
2.0 Introducción. ....42
2.1 Selección de materiales de construcción. ....42
2.2 Diseño físico del sistema. ....42
Figura 12: Sistema prototipo. ....43
Figura 13: Soporte de contenedores. ....44
Figura 14: Soporte de contenedores. ....44
2.3 Características de los solenoides. ....44
Tabla 10: Tabla de valores operativos nominales de los solenoides. ....45
Figura15: Válvulas de solenoide implementadas en sistema. ....45
2.4 Análisis funcional de los solenoides. ....45
2.4.1 Consumo de potencia. ....45
Figura 16: Niveles operativos válvulas de solenoide. ....46
2.4.2 Retardos de apertura y cierre. ....46
Figura 17: Retardo de apertura de solenoides ....47
Figura 18: Retardo de cierre de solenoides ....48
2.5 Diseño de las placas de control. ....48
Figura 19: Plano de conexiones para placas de control. ....49
Figura 20: Placas de control, Fotografía de las placas de control fabricadas. ....50
2.6 Método de fabricación de placas. ....50
Capitulo 3. Desarrollo del software controlador. ....51
3.0 Introducción. ....51
3.1 Selección de plataformas. ....51
3.2 Caracterización del proceso experimental. ....51
Figura 21: Flujo general del proceso experimental de perfusión. ....52
3.3 Descripción de las fases. ....52
Figura 22: Flujo operativo para la ejecución de una curva de concentraciones. ....53

Figura 23: Flujo operativo para la ejecución de una aplicación unitaria. ....54
Figura 20: Flujo operativo para la ejecución de aplicaciones empalmadas. ....55
Figura 24: Flujo operativo para la ejecución de corrientes mecánicas. ....55
Figura 25: Flujo operativo para la ejecución manual ....56
Figura 26: Flujo operativo para el modo independiente de ejecución. ....57
Figura 27: Flujo operativo para el modo sincronizado de ejecución. ....58
3.8 Modelado General del software de control. ....60
Figura 28: Diagrama de flujo detallado del sistema de perfusión. ....60
3.9 Modelado de base de datos. ....60
Figura 29: Modelo Entidad – Relación base de datos del sistema. ....61
3.10 Diccionario de datos. ....61
3.11 Programación del sistema. ....61
3.11.1 Control y acceso al puerto paralelo. ....62
3.11.2 Carga en RAM de rutina de ejecución ....65
Figura 30: Flujo para la carga de rutinas de aplicación. ....65
3.11.3 Ejecución de rutina y temporización de solenoides. ....66
Figura 31: Retardo de respuesta mecánico del solenoide ....67
Figura 32: Ajuste en software del retardo mecánico ....68
Figura 33: Posibles configuraciones del sistema. ....68
Figura 34: Ejecución del proceso intercepción. ....70
3.11.4 Finalización de la aplicación e inserción a base de datos. ....72
3.11.5 Carga y prueba de parámetros de retardo. ....76
3.12 Programación de Hilos y prioridad de ejecución. ....77
Tabla 11: Prioridades de ejecución de los hilos. ....78
Capitulo 4.Resultados. ....80
4.0 Introducción. ....80
4.1 Prototipo del sistema. ....80
Figura 32: Fotografía sistema prototipo. ....81
4.2 Software de control. ....81
Figura 33: Pantalla principal del sistema. ....82
Figura 34: Flujo de ventanas general del sistema. ....82
4.3 Conclusiones ....85
4.4 Trabajo a Futuro. ....86
Apéndices. ....88
1 Bibliografía ....95

## Resumen.

"*Automatización de procesos experimentales en ovocitos y folículos de Xenopus*", tesis de nivel licenciatura Universidad Autónoma de Querétaro 2007.

En esta Tesis se presenta el desarrollo e implementación de un sistema automatizado de perfusión celular que llamamos "*Cronos*", el cual consiste en un software desarrollado para la creación de rutinas que controlan la apertura y cierre de un arreglo de válvulas, que aplican diferentes fármacos a una cámara de perfusión.

La naturaleza del procedimiento experimental así como la infraestructura donde será utilizado el sistema delimitó el desarrollo del proyecto, obligando a realizar una selección específica de la plataforma de desarrollo, puertos de comunicación, niveles de señalización y materiales de construcción. De igual manera, su desarrollo requirió diseñar procedimientos de ajuste tanto en software como en hardware con el objetivo de asegurar la reproducibilidad de los procedimientos realizados, así como la inexistencia de influencias eléctricas y mecánicas generadas por el sistema que afecten los resultados de dicho proceso experimental.

El sistema está compuesto por dos elementos, el software de control y la interfaz mecánica. El software de control es el resultado de la caracterización del método experimental y el desarrollo de una interfaz gráfica sencilla, que permiten la planificación de un protocolo de experimentación, así como la modificación en tiempo real de la aplicación de fármacos. Esta caracterización permitió también diseñar diferentes módulos correspondientes a las variantes del proceso, permitiendo crear un sistema apegado a las necesidades reales del experimentador. Adicionalmente, fue incluido el manejo de una base de datos para almacenar los protocolos, resultados e información generada por el proceso para su posterior ejecución y consulta.

La interfaz mecánica del sistema se compone de circuitos de control, circuitos de amplificación y cableado eléctrico, los cuales fueron diseñados con el fin de permitir a una PC controlar una serie de válvulas de solenoide mediante un puerto simple de comunicación. Esta interfaz también se compone de una estructura física, la cual fue

diseñada para permitir que sistema se pueda alojar en una mesa de trabajo convencional mediante un mecanismo sencillo para su instalación y remoción.

## Introducción.

Las Neurociencias son, sin duda alguna, una de las ramas de la ciencia mayormente desarrolladas en nuestro país, prueba de ello es la existencia de centros de investigación como el Instituto de Neurobiología (INB) de la Universidad Nacional Autónoma de México (UNAM) campus Juriquilla, centro que cuenta entre sus miembros con destacadas personalidades en el área como el Doctor *Honoris Causa* de la UAQ y de la UNAM Ricardo Miledi, acreedor del Premio Príncipe de Asturias en 1999.

En dicho Instituto se estudia la estructura y las funciones del cerebro mediante una plataforma multidisciplinaria dividida en tres departamentos, permitiendo así el estudio de los aspectos moleculares, celulares, conductuales y cognitivos del cerebro. Las actividades de investigación del Instituto se realizan en laboratorios especializados que cuentan con equipos, metodologías y personal especializado.

Uno de los departamentos de estudio de este instituto es el departamento de Neurobiología Celular y Molecular, en donde se investigan los diversos procesos neurobiológicos de la comunicación celular, tanto a nivel eléctrico como químico, así como sus implicaciones clínicas, zootécnicas y farmacológicas. Este departamento se encuentra integrado por trece laboratorios.

Entre estos laboratorios se encuentra el laboratorio de Neurofisiología Celular (D-13).
En dicho laboratorio es analizada la comunicación entre células durante diferentes procesos fisiológicos. La comunicación celular es estudiada con técnicas de experimentación electrofisiología utilizando como sistema de expresión folículos de la rana *Xenopus laevis*, el cual permite estudiar los aspectos químicos y eléctricos de la señalización celular, permitiendo así describir las vías de comunicación de distintos modelos celulares desde los elementos moleculares que participan, hasta las consecuencias fisiológicas de su activación.

## Justificación.

El estudio de las Neurociencias requiere en muchos casos de sistemas experimentales automatizados por varias razones, por ejemplo 1)Reducir al máximo alguna alteración por parte del observador, en eventos en donde el tiempo de reacción y/o algún otro parámetro se encuentra fuera del control humano, 2) eventos donde la manipulación mecánica se realiza de forma repetida y en dimensiones micrométricas (e.g., microscopia confocal ), ó 3) aquellos en los que el volumen de análisis es muy grande y costoso por lo que requiere mayor eficiencia (e.g., farmacología) y 4 ) eventos en donde establecer condiciones experimentales reproducibles es fundamental para la obtención de resultados(e.g, registro electrofisiológico).

Desafortunadamente la investigación en nuestro país se encuentra limitada por presupuestos insuficientes que impiden adquirir equipamiento que facilite la labor del investigador, obligándolo a realizar una serie de controles experimentales necesarios para verificar la veracidad de sus experimentos.

Para este caso en particular, este proyecto de Tesis se centra en la construcción de un sistema de perfusión celular. (El cual consiste en un recambio constante de solución, ya sea para mantener las condiciones experimentales, o bien, para el caso opuesto, donde el investigador requiere del cambio de soluciones de manera programada y secuencial). Es importante destacar que en la actualidad existen comercialmente equipos automatizados de origen extranjero que tiene costos que van desde los 2,000 a los 5,000 Dólares, suma que es considerablemente alta y que nos obliga a ser dependientes de tecnologías importadas.

## Objetivo general.

Este proyecto de tesis tiene como objetivo la **construcción de un sistema automatizado de perfusión,** con un costo de desarrollo menor al de un producto comercial.

## Objetivos específicos.

Con el fin de alcanzar el objetivo general de este proyecto la resolución de las siguientes premisas será fundamental:

1. Comprender en su totalidad el proceso experimental.
2. Realizar un análisis de reingeniería a los sistemas comerciales.
3. Diseñar los flujos que intervendrán en el sistema
4. Diseñar un modelo entidad-relación para la implementación de la base de datos del sistema.
5. Programar el software de control e interfaz gráfica.
6. Diseñar físicamente el sistema.
7. Seleccionar los materiales para su construcción.
8. Diseñar los circuitos de control necesarios.
9. Asegurar la inexistencia de influencias eléctricas y mecánicas en los resultados.
10. Implementar y depurar el sistema.
11. Aplicación del sistema en diversos procesos biológico como:
    a) Curvas dosis respuesta
    b) Estimulación mecánica

# Capítulo 1 Generalidades.

## 1.0 Introducción.

En este capítulo se presenta un panorama general de los conceptos más importantes de Biología, Electrónica y Programación, necesarios para el entendimiento del proyecto desarrollado.

## 1.1 Farmacología.

La farmacología constituye la base científica que determina si una sustancia puede ser usada con fines terapéuticos y en relación con la salud denominada fármaco. Esta rama de la ciencia se encarga de estudiar las propiedades, efectos, y facetas de toda sustancia que interactúa con sistemas biológicos por medio de procesos químicos que inhiben o activan procesos corporales normales. Los estudios de la farmacología incluyen conocimientos de la historia, origen, propiedades físicas y químicas, efectos bioquímicos y fisiológicos, los mecanismos de acción, absorción, distribución, excreción, así como los usos terapéuticos de estas substancias (Cedric M. Smith 1995).

La determinación del uso terapéutico o la utilidad de la sustancia ante una enfermedad se encuentra estrechamente relacionando con la dosis y concentración administrada a un paciente. En este sentido, la farmacología define dos áreas principales de estudio, 1) la farmacocinética, que estudia el aspecto cuantitativo del comportamiento de los fármacos, es decir lo que el organismo hace con el fármaco administrado y 2) la farmacodinámica, que estudia los efectos bioquímicos y fisiológicos de las drogas así como el de sus mecanismos de acción, a través del efecto que tiene sobre las funciones del organismo (Katzung 2005).

## 1.2 Mecanismos de acción Fármaco-Receptor.

Receptor: componente molecular de un sistema biológico, con el cual diferentes fármacos reaccionan y producen cambios en las funciones celulares específicas (Katzung 2005).

La denominación de una sustancia como fármaco se encuentra estrechamente relacionado con la interacción que estas sustancias tienen con las moléculas corporales, es decir, es el resultado de la interacción de dos moléculas (o grupos). De esta interacción se derivo el concepto de blanco molecular, que puede definirse como el componente de un organismo con el cual interactúa el fármaco, iniciando la serie de fenómenos bioquímicos que llevan a la consecución del efecto.

Este esquema que define el concepto de blanco molecular es de gran importancia pues permite de manera práctica identificar:

a) La relación cuantitativa entre la concentración de un fármaco y sus efectos.

b) La selectividad de acción farmacológica que dependerá de la interacción con los fármacos.

c) La forma en que actúa el fármaco, ya sea en una acción de antagonista (moléculas capaces de unirse a un receptor sin alterar su función, solamente lo bloquean), o en forma de agonista (moléculas que son capaces de determinar un cambio mediado por su previa unión).

Por lo general estos receptores o blancos moleculares son proteínas de membrana celular. Al estar formadas de múltiples aminoácidos las posibilidades de interacción son sumamente altas. De igual manera la localización de estos receptores es heterogénea, es decir, las zonas en donde se encuentran están delimitadas, definiendo de manera directa los sitios de acción de un fármaco así como la concentración a la que el receptor deberá ser expuesto.

## 1.3 Cuantificación de la interacción receptor – fármaco.

La interacción entre receptor – fármaco puede ser representada por medio de la ecuación:

$$[R] + [X] = [RX]$$

Donde [R] es la concentración de receptores, [X] es la concentración del fármaco en la vecindad de los receptores y [RX] es la concentración del complejo fármaco-receptor.

Puesto que la mayor parte de las acciones de un fármaco se pueden explicar como la consecuencia de su unión a un receptor, se puede expresar la consecución del efecto

farmacológico de la siguiente manera:

$$E = \frac{E_{max} * Dosis}{EC_{50} + Dosis}$$

En donde $E_{max}$ se refiere al efecto máximo a conseguir y $EC_{50}$ a la dosis requerida para lograr la mitad de dicho efecto. La forma grafica de esta ecuación se conoce como curva dosis respuesta (Meek ME 2002) (Tozer 2006).

## 1.4 Curva Dosis Respuesta.

Esta representación gráfica nos permite conocer el grado del efecto de una sustancia sobre un receptor. La dosis (eje x) representa la concentración del fármaco, es decir, la dosis necesaria para alcanzar una respuesta determinada (eje y). Esta gráfica nos permite conocer también la *afinidad* de un fármaco por su receptor, así como la eficacia o efecto mayor de este (figura 1).

Si bien esta representación corresponde a la respuesta que una célula presenta frente a un fármaco, es evidente que esta respuesta solamente puede aplicarse a dicho organismo, hecho que se conoce como la variabilidad biológica.

Este parámetro obliga entonces a determinar un nivel efectivo de dosis que podría ser aplicada a organismos mediante la elaboración de curvas dosis respuesta a un número amplio de variedades celulares. La dosis en donde el 50% de los complejos celulares presentan un efecto estudiado es denominado $DC_{50}$, valor que es utilizado comúnmente como la concentración adecuada de administración, al ser el umbral en donde se presenta una respuesta beneficiosa sin efectos tóxicos o letales (D´Argenio 2004).

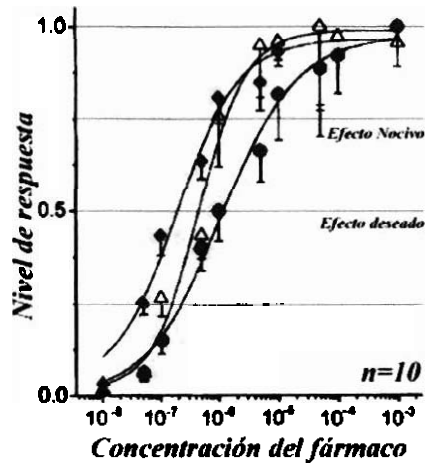


**Figura 1: Curva Dosis – Respuesta**, grafica representativa para el análisis del comportamiento de una respuesta eléctrica celular frente a tres fármacos en diferentes concentración dentro de un rango señalado. Los fármacos fueron ATP ( ●), UTP (△), 2-MeSATP (◆) para un grupo de estudio de diez células. (Modificada de Arellano, y cols. 1998).

## 1.5 Análisis funcional de receptores.

El modelo más ampliamente utilizado para el estudio de la estructura y funciones de muchas proteínas humanas, es el del ovocito de la rana *Xenopus laevis*. Este modelo comenzó a ser utilizado en 1982 cuando el Dr. Ricardo Miledi y su grupo de investigación demostraron que la inyección de RNA mensajero extraído del cerebro conducía a la síntesis y a la incorporación de muchos tipos de receptores y canales iónicos en la membrana del ovocito, demostrando incluso que este modelo podía llevarse directamente al estudio de patologías al utilizarse receptores de pacientes con desordenes cerebrales y mediante la utilización de técnicas de electrofisiología realizar además estudios sobre el comportamiento y la afinidad de fármacos en estado experimental (Ricardo Miledi 2002).

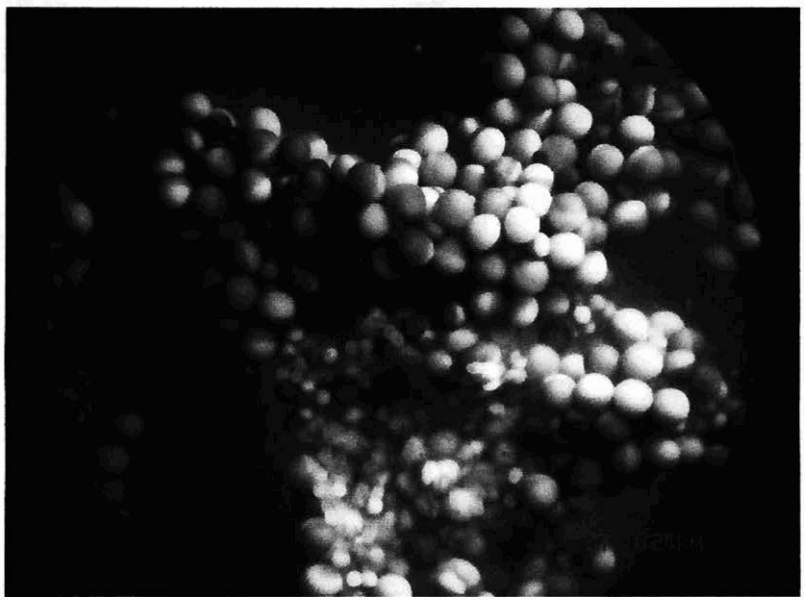

**Figura 2: Ovocitos de la Rana Xenopus *laevis*, fotografías amplificadas de un ovocito, estas células esféricas tiene un diámetro aproximando de 1mm y son identificables a simple vista.**

## 1.6 Técnica de control de voltaje.

El control de voltaje, es una técnica de registro electrofisiológico ampliamente utilizada para el estudio de corrientes que fluyen a través de canales iónicos en una membrana. Esta técnica consiste en controlar el valor de voltaje transmembranal de la célula bajo estudio. De esta manera puede ser medida la corriente requerida para mantener el voltaje constante, y saber la amplitud y cinética de la corriente que fluye a través de la membrana que es proporcional a su conductancia determinada por la apertura y cierre de los canales iónicos.

El sistema utilizado para la medición de estas corrientes está formado por dos microelectrodos intracelulares (Figura 3), uno de los cuales registra el voltaje (E) y otro inyecta corriente (I) a la célula. Debido a que el orden de magnitud de los voltajes registrados es muy pequeño, el electrodo de voltaje se conecta a un amplificador de transimpedancia. Esta señal es comparada con un voltaje comando (E) que es el voltaje al que se desea mantener la membrana, la diferencia entre ellos es compensada por una inyección de corriente a través del electrodo (I), cuya magnitud es determinada a través de

un circuito de retroalimentación negativo (FBA feedback amplifier). En la cámara de perfusión se encuentra un tercer electrodo que mide a través de un amperímetro la corriente (I) que cruza la membrana de la célula. La cámara de perfusión está conformada por un receptáculo que alberga a la célula en estudio con el objetivo de disminuir su movilidad. La cámara de perfusión es llenada de manera constante y de modo unidireccional gracias a una succión negativa que permite regular el flujo de solución a la que será expuesta la célula en estudio, a lo que se conoce como sistema de perfusión (Sakmann 1995).

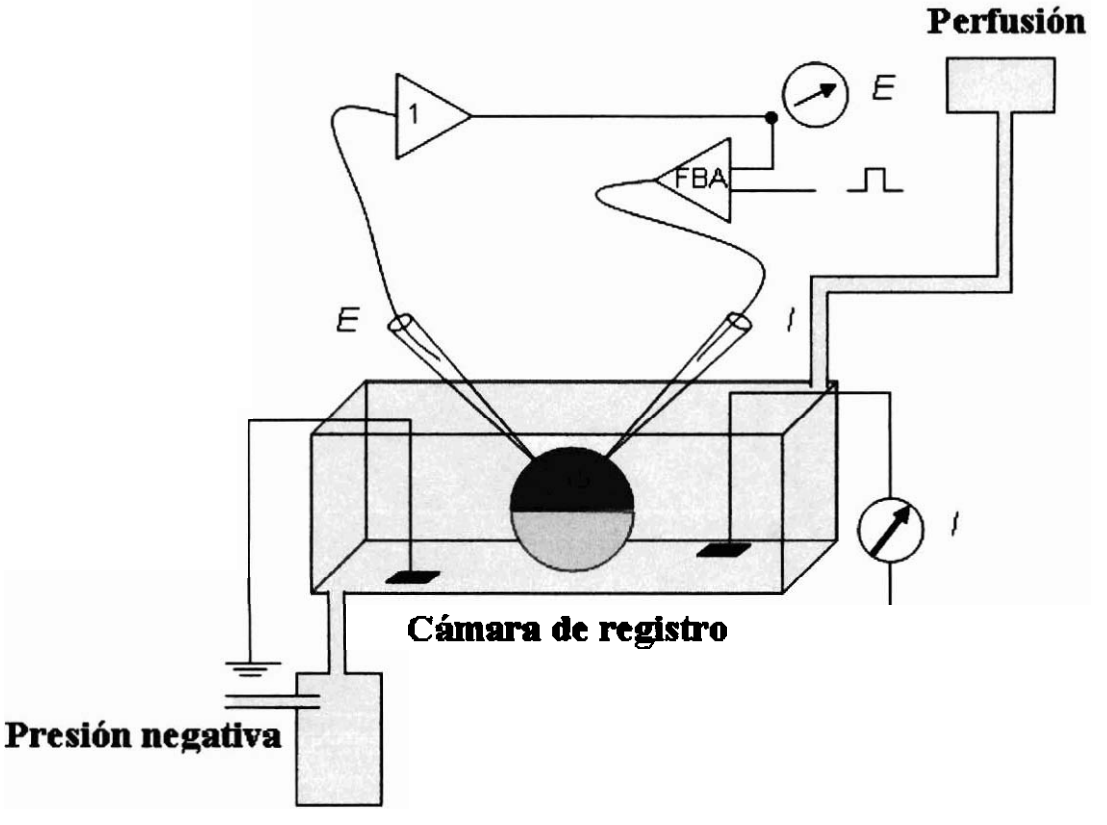


**Figura 3: Técnica de control de voltaje,** Caricatura representativa de la técnica con ovocito de Xenopus, donde **E´** es el electrodo de voltaje. **X1** es el amplificador. **E** es el voltaje de reforzamiento o mantenimiento. **FBA** amplificador de retroalimentación negativa. **I´** es el electrodo intracelular de corriente **I** es el amperímetro que se encuentra en el baño y censa la corriente transmembranal.

Dentro del laboratorio de Neurofisiología celular esta técnica es implementada mediante un amplificador de voltaje controlable Geneclamp 500B Axón Instrumentes, un osciloscopio marca Gould (D50-602), una tarjeta de adquisición de datos Digidata 1200A controlada por un software de registro pClamp 9 de la marca Axón Instrumentes, así como un generador de pulsos GRASS Astro-Med. La perfusión y selección de las soluciones se realiza de manera manual mediante flujo por gravedad y un matraz para cada una de las soluciones a perfundir.


**Figura 4: Set de experimentación Laboratorio Neurofisiología Celular**, (A) es la cámara de experimentación montada mediante amplificadores HS-2ª Axón Instrumentes, (B) es el amplificador de voltaje controlado Geneclamp 500B Axón Instrumentes, (C) es el osciloscopio Gould (D50-602), (D) es el generador de pulsos GRASS Astro-Med, (E) es el sistema de registro implementado en una PC mediante una tarjeta de adquisición de datos Digidata 1200A y el software de registro pClamp 9.

## 1.7 Sistemas Comerciales.

Existe una amplia diversidad de fabricantes de sistemas automatizados de perfusión celular. En su mayoría los fabricantes de estos sistemas son proveedores de tecnologías del sector farmacéutico. Como punto de partida de este proyecto fueron analizados distintos productos de numerosos fabricantes, con el fin de identificar las ventajas, desventajas, tecnologías y diseños que existen en el mercado, algunos de los productos más destacados son:

## 1.8 Complete 8-Channel Fast Miniature Perfusion System

### 1.8.1 Fabricante: Bioscience Tools.

### 1.8.2 Descripción.

Es un sistema miniatura de perfusión con 8 canales, permite ser utilizado para cambiar la solución alrededor de una muestra, o para aplicar compuestos de prueba localmente.

### 1.8.3 Características físicas.

Los canales están implementados en válvulas de solenoide con un mecanismo de cierre por constricción de manguera o de "pinch", estas válvulas se encuentran alojadas en una caja de aluminio que permite montarlas en un soporte extensible de hasta 3 pies, permitiendo así colocar las válvulas cerca de la muestra y reducir al mínimo el volumen muerto en tubería. Se incluye un sostenedor magnético y contenderos de fármacos con capacidad de 50 ml con llave de paso y tubería de $1\frac{1}{16}$ de diámetro interno de teflón (Bioscience Tools Inc. 2007).

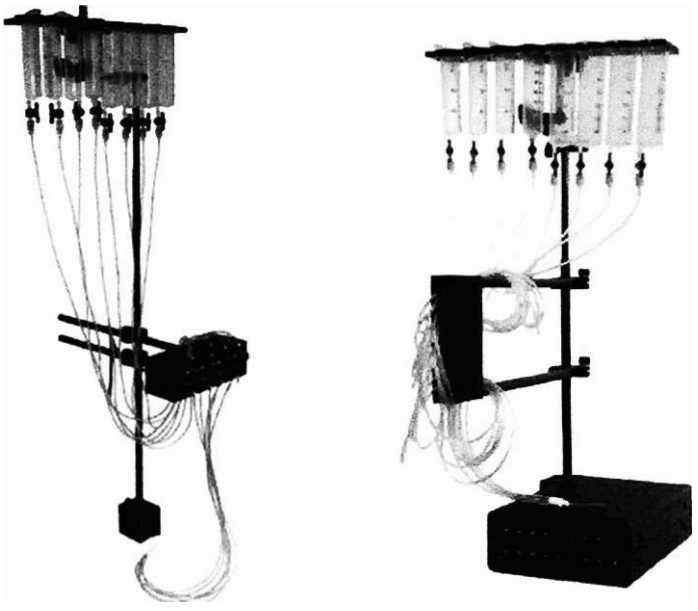

**Figura 5: Sistema de perfusión Bioscentific tools.**

### 1.8.4 Características eléctricas.

El sistema incluye un controlador de 16 canales implementados en interruptores manuales o por señales TTL generadas por computadora mediante puerto RS232 o conexión USB. Las válvulas tienen un tiempo de respuesta de 1.5ms y operan con 12VDC. El sistema eléctrico de las válvulas se encuentra en cajas de metal y sus conexiones se realizan a través de cables blindados, su fuente de alimentación es externa y opera en el rango de 120/220V (Bioscience Tools Inc. 2007).

### 1.8.5 Funcionalidades.

El sistema puede controlar hasta 16 canales, es incluido un modo de funcionamiento automático el cual realiza una apertura secuencial de válvulas de izquierda a derecha. El modo manual permite controlar las válvulas mediante los controles externos, y el modo codificado controla automáticamente la conmutación de las válvulas mediante entradas digitales provenientes de un puerto RS232 de una PC. (Bioscience Tools Inc. 2007)

### 1.8.6 Costos.

El sistema se puede adquirir en diferentes presentaciones que se diferencian por el número de canales que implementan y la calidad de los solenoides, algunos de los modelos y costos representativos son (Bioscience Tools Inc. 2007):

| Modelo. | Nombre | Precio |
|---|---|---|
| PS15-8 | Complete 8-Channel Fast Miniature Perfusión System | 3,210.00Dll |
| PS15-4 | Complete 4-Channel Fast Miniature Perfusión System | 2,740.00Dll |
| PS25-2 | Complete 2-Channel Miniature Perfusión System | 1,930.00Dll |

**Tabla 1: Precios sistemas Bioscience Tools.**

## 1.9 ValveBank® 8 Teflon Perfusion Systems

### 1.9.1 Fabricante: AutoMate Scientific.

### 1.9.2 Descripción.

Es un sistema de perfusión de 8 canales con equipo de procesamiento independiente de fácil uso y fácil instalación.

### 1.9.3 Características físicas.

Los canales están implementados en válvulas de solenoide con un mecanismo de cierre de "pinch", estas válvulas están alojadas en una caja de aluminio y su soporte permite montarlas en diferentes posiciones, son incluidos contenedores de fármacos con capacidad de 60 ml y tubería de $1\frac{1}{16}$ de diámetro interno de teflón. (AutoMate Scientific Inc. 2007)

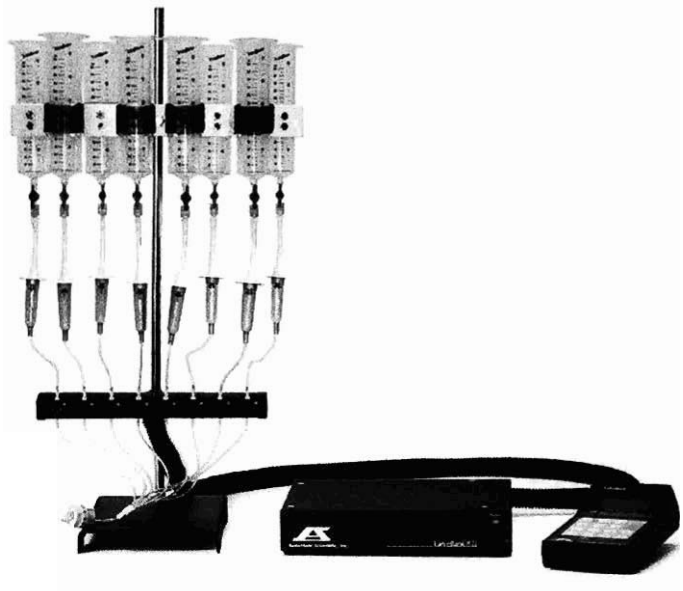

**Figura 6: Sistema de perfusión ValveBank®8 Teflón.**

## 1.9.4 Características eléctricas.

El sistema incluye una consola con microcontrolador, que mediante señales TTL controla las válvulas, permite enviar la secuencia de aplicación mediante puerto RS232 o conexión USB. Las válvulas tienen un tiempo de respuesta de 1.4ms y operan con 12VDC con una baja impedancia, el cableado eléctrico es blindado, su fuente de alimentación es externa y opera en el rango de 120/220V (AutoMate Scientific Inc. 2007).

## 1.9.5 Funcionalidades.

El sistema controla 8 canales mediante un microcontrolador con memoria, que le permite operar de manera independiente hasta por 99 horas, también puede ser controlado mediante hardware Digidata, ITC-16 o tarjetas Nacional Instruments mediante el software pClamp, o LabView mediante señales TTL. También incluye un software propietario llamado EasyCode que permite diseñar de manera sencilla las secuencias de aplicación (AutoMate Scientific Inc. 2007).

### 1.9.6 Costos.

El sistema se puede adquirir en diferentes presentaciones que se diferencian por el número de canales que implementan y la velocidad de reacción de las válvulas (AutoMate Scientific Inc. 2007).

| Modelo. | Nombre | Precio |
|---|---|---|
| 13-01-23 | Valve Bank*4 Perfusión System Valves 4ms delay. | 2,395.00Dll |
| 13-01-53 | Valve Bank *8 Teflon Perfusión System Valves 4ms delay. | 3,995.00Dll |
| 13-21-57 | Valve Bank *8 Teflon Perfusión System Lee Valves 1.4ms delay. | 4,385.00Dll |

**Tabla 2: Precios sistemas AutoMate Scientific.**

## 1.10 Ventajas.

Para ambos casos se encontró que los sistemas tienen diseños físicos muy depurados, que permiten montarlos y desmontarlos con facilidad, así como realizarles mantenimiento y cambio de piezas de manera sencilla. En ambos casos se encontró también que los parámetros eléctricos y descripciones son muy apropiados para su aplicación final además de contar con mecanismos de apertura manual y controlada por PC que permiten tener variantes interesantes en su uso. Adicionalmente como valor agregado a la gama de productos, los fabricantes permiten elegir la calidad de las válvulas del sistema, la velocidad de respuesta, el número de canales implementados ajustando así el sistema a los requerimientos del cliente.

## 1.11 Desventajas.

La desventaja principal de estos sistemas son los elevados costos de adquisición e importación, así como los posibles costos que representaría la compra de repuestos y reparaciones pues en su totalidad de trata de productos fabricados en el extranjero. Muy posiblemente, los elevados costos de adquisición obedecen a una relación de oferta - demanda del mercado, sin embargo es fácilmente apreciable que la mayoría de estos equipamientos fueron diseñados para la industria farmacéutica y posteriormente adaptados

al campo de la investigación. Por ejemplo, si bien en los productos posteriormente analizados en ambos casos los sistemas permiten controlar la apertura de las válvulas mediante una PC, también en ambos casos se incluye un medio mecánico como botones o switches que permiten operar de manera manual el sistema sin una PC, implementación que obviamente impacta el costo final del producto, otro ejemplo es que en ambos casos se permite conectar el sistema con software y hardware de adquisición de datos muy especifico que en algunos casos aun no arriban al área de la investigación (LabView).

Otro punto importante que no es completamente claro, es la información y descripciones técnicas ofrecidas al consumidor. Si bien son presentados números y datos sobre el desempeño de los productos, en ningún momento son presentadas hojas de datos de componentes ni prueba alguna que demuestre la veracidad de los parámetros expuestos, por ejemplo el tiempo de respuesta de los solenoides que ofrecen estos productos se encuentra en el orden de los 1.4ms, sin embargo no se especifica si este tiempo de respuesta es un tiempo de respuesta eléctrico o mecánico, de igual manera para los valores de consumo de potencia y emisión de ruido no se especifica dato alguno de su construcción ni componentes y mucho menos se especifica si los valores de consumo de potencia y emisión de ruido corresponde a valores nominales, o bajo condiciones especificas.

Si bien los posibles problemas de carácter técnico que cualquier sistema presenta son relativamente sencillos de superar, la lógica del diseño al que fueron sujetos es difícilmente adaptable en la marcha de su implementación, por ejemplo para el producto ValveBank®8 Teflón Perfusión System (Figura6) se presupone un escenario en donde el trabajo del usuario consta en cargar una rutina de aplicaciones y esperar un resultado, este panorama es muy contrastante al trabajo realizado dentro del Laboratorio de Neurofisiología Celular, en donde en la mayoría de los casos, son tomadas decisiones que modifican el diseño experimental como reiniciar el experimento, abortarlo, modificar alguna aplicación subsecuente, escenarios que requerirían en los sistemas comerciales detener el trabajo, rediseñar un protocolo e iniciar nuevamente el experimento.

## 1.12 Tecnología TTL.

TTL es una tecnología de construcción de circuitos electrónicos digitales, en la que los elementos de entrada y salida son transistores. Esta tecnología fue introducida por Signetics en 1961 y rápidamente popularizada y adoptada por fabricantes como Texas Instruments convirtiéndose así en un estándar en la industria bajo el nombre de la familia 74xx (Malvino 1991).

Al paso del tiempo esta tecnología ha sufrido diversas modificaciones en sus prestaciones por lo que se han añadido diversas versiones de esta familia, las diferentes variantes de la familia 74 son:

| Parámetros de funcionamiento | Parámetros de Voltaje |
|---|---|

| Familia | Retardo (ns) | Disipación (mW) | Frecuencia(MHz) | VOH | VOL | VIH | VIL |
|---|---|---|---|---|---|---|---|
| Serie 74 | 33 | 1 | 3 | 2.4 | 0.4 | 2.0 | 0.8 |
| Serie 74L, TTL de bajo consumo de potencia | 6 | 23 | 50 | 2.4 | 0.4 | 2.0 | 0.7 |
| Serie 74H, TTL de alta velocidad | 3 | 20 | 125 | 2.7 | 0.5 | 2.0 | 0.8 |
| Serie 74S, TTL Schottky | 9.5 | 2 | 45 | 2.7 | 05 | 2.0 | 0.8 |
| Serie 74LS (LS-TTL), TTL Schottky de bajo consumo de potencia | 1.7 | 8 | 200 | 2.5 | 0.5 | 2.0 | 0.8 |
| Serie 74AS (AS-TTL), TTL Schottky avanzada | 4 | 1 | 70 | 2.5 | 0.5 | 2.0 | 0.8 |

**Tabla3: Características y versiones de la familia TTL.**

## 1.13 Solenoides.

Es una de las múltiples aplicaciones derivadas del descubrimiento del electromagnetismo y la ley de Ampere, la cual demuestra que es posible generar campos magnéticos ordenados mediante la conducción de corrientes. Al igual que un electroimán un solenoide es una bobina conformada por varias espiras de alambre conductor arrolladas una al lado de la otra según una hélice de paso constante en una o varias capas sobre una superficie cilíndrica. A diferencia de un electroimán, en el solenoide el núcleo cilíndrico tiene libertad de movimiento, de esta manera mediante la polarización, y despolarización de la bobina el núcleo se mueve de un polo magnético a otro, haciendo la función de pistón pudiendo realizar tareas mecánicas como interruptor o compuerta. (Edminister 1999)

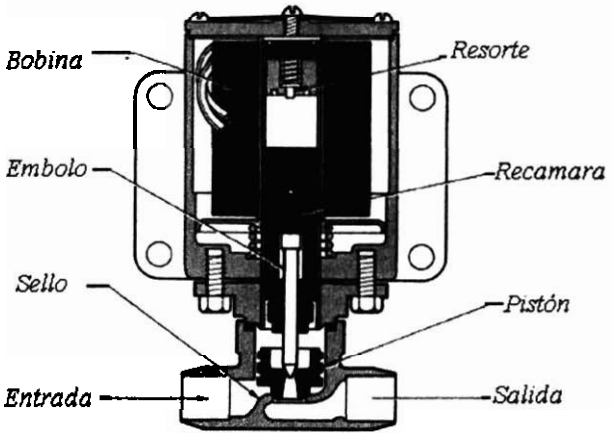


**Figura 7: Válvula de solenoide,** caricatura representativa de la composición física y componentes de un sistema de válvula de solenoide.

## 1.14 Comportamiento funcional del solenoide.

Al igual que un electroimán, el solenoide está sujeto a las mismas leyes físicas que determinan la intensidad de la carga. En el caso especifico del solenoide, el nivel de carga determinará el momento en que el embolo será movido y el funcionamiento mecánico será realizado. Este comportamiento puede ser clasificado como, nivel de polarizaron y nivel de

despolarización, variables que son determinadas completamente por el tipo de construcción y materiales empleados (Kraus 2000).

## 1.15 El puerto paralelo

El puerto paralelo estándar SPP apareció con la primera computadora personal lanzada por IBM en 1981 y fue introducido como una alternativa al bajo rendimiento del puerto serial existente. Desde su aparición, el puerto ha sufrido varias modificaciones para hacerlo más veloz y bidireccional ya que originalmente era unidireccional.

Su estandarización se logró hasta 1994 con la aparición de la norma IEEE 1284, la cual, establece que el puerto paralelo funciona bajo la lógica y niveles de señalización TTL a través de ocho líneas de datos paralelos, cinco líneas de estado y cuatro líneas de control; además de cinco modos de transmisión: modo de compatibilidad, modo byte, modo bidireccional, modo EPP.

Cada uno de estos modos provee un método de transmisión en una de las siguientes direcciones: PC-periférico, periférico-PC o bidireccional. El modo de compatibilidad es soportado por la mayoría de las computadoras existentes. Por definición es unidireccional y provee el método más simple en la transmisión de los datos ya que no requiere algún tipo de protocolo de comunicación. Es importante mencionar que todos los modos son dependientes del software, por lo que la velocidad de transmisión se ve fácilmente afectada por la velocidad de procesamiento y la carga de trabajo del procesador. En la siguiente tabla se muestran los puertos paralelos utilizados actualmente y sus conectores (Institute of Electrical and Electronics Engineers 1992).

| Modo | SSP | PS/2 | EPP | ECP |
|---|---|---|---|---|
| Fabricante | IBM | IBM | Intel | Hewlett Packard |
| Bidireccional | No | Sí | Sí | Sí |
| Acceso Directo a memoria. | No | Sí | Sí | Sí |
| Velocidad | 150 kB/s | 150 kB/s | 2 MB/s | 2 MB/s |
| Conector | DB25 | PS/2 | DB25 | DB25 |

**Tabla 4: Puerto paralelo, características.**

## 1.16 Nomenclatura del conector.

| Pin | Descripción |
|---|---|
| 1-14-16-17 | Líneas de control(C) |
| 2 ~ 9 | Líneas de datos (D) |
| 13-12-11-10-1510 | Líneas de estado.(S) |
| 18 ~ 25 | Tierra eléctrica. |

**Tabla 5: Nomenclatura del conector puerto paralelo.**

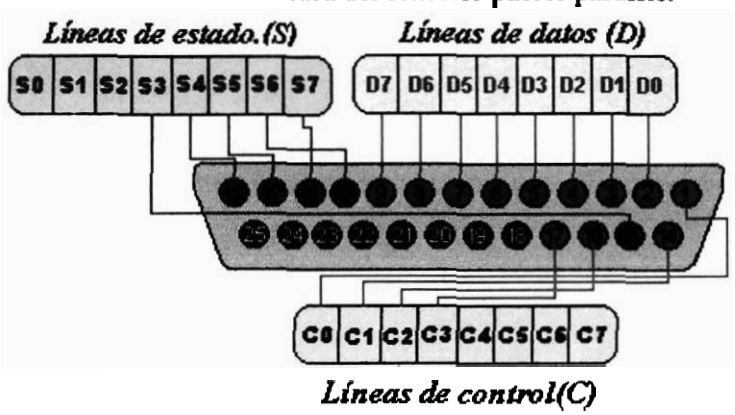


**Figura 8: Diagrama conexiones Puerto Paralelo.**

## 1.17 Acceso y control del puerto paralelo.

El control y acceso a puertos en sistemas operativos como Win95/98 es una rutina relativamente sencilla, pues puede ser operado mediante funciones convencionales de software como Inporb, outportb o _inp (), _Outp haciendo referencia directa al valor numérico de puerto con el que se desea trabajar (Axelson 1997).

| Dirección Inicial | Función |
|---|---|
| 0000:0408 | Dirección base para LPT1 |
| 0000:040A | Dirección base para LPT2 |
| 0000:040C | Dirección base para LPT3 |
| 0000:040E | Dirección base para LPT4 |

**Tabla 6: Direcciones comunes para puerto paralelo.**

Sin embargo para las generaciones de sistemas operativos basados en NT y sus clones como WIN NT4, WIN2000, WINXP, estas rutinas son inoperantes debido a que el sistema operativo asigna una serie de privilegios y restricciones a distintos tipos de programas, creando así una clasificación en dos categorías, Modo Usuario y Modo Kernel. Obviamente el modo Kernel hacer referencia las funciones y directivas del sistema operativo y el modo Usuario al resto del software, así el sistema operativo controla el acceso a recursos de entrada y salida a puertos permitiéndole detectar, bloquear y finalizar la ejecución de todo software que haga uso de estas. Para el caso de estas plataformas el acceso y control del puerto se logra mediante la programación o utilización de drivers de comunicación con puertos que permiten utilizar las funciones convencionales.

## 1.18 Inpout32.dll.

Es un driver de comunicación con puertos para sistemas operativos basados en NT. Funciona como un gestor de puertos entre el sistema operativo y el programador, permitiendo al programador utilizar las sentencias básicas de entrada y salida sin necesidad de escribir un código de mayor complejidad para la gestión de estos recursos, algunas de las ventajas de utilizar este driver son (LOT Logix Group 2005):

- Opera bajo todas las versiones de Windows (WIN 98, NT, 200 y XP) por lo que no es necesario modificar el código fuente para migrarlo de plataforma.
- Acceso a los privilegios del modo Kernel mediante este gestor.
- No se requiere software especial ni adicional.
- El driver es cargado automáticamente cuando la dl les cargada.
- No se requiere una API especifica las funciones Inp, Out pueden ser utilizadas.
- Fácil uso en Visual C++ y Visual Basic.

## 1.19 Ejecución multiproceso.

La arquitectura clásica de una PC, consta de un solo microprocesador (CPU) que trabaja una instrucción a la vez, por lo que siendo realistas, es imposible que una PC con esta arquitectura pueda ejecutar dos aplicaciones al mismo tiempo.

Sin embargo para algunos sistemas operativos es posible crear un efecto de paralelismo gracias a que realizan un multiprocesamiento virtual que se basa en la ejecución de cada instrucción en un tiempo muy breve, pero muchas veces por segundo. El tiempo o la rebanada de tiempo (TimeSlice) en que cada una de estas aplicaciones se ejecuta en el CPU es controlado enteramente por el software del sistema operativo responsable de gestionar los recursos de hardware (Kernel), generando así lo que puede ser conocido como el efecto rendimiento – sensación, es decir, si bien el CPU realiza una tarea a la vez para el usuario existe la sensación que todas las aplicaciones se ejecutan al mismo tiempo (National Instruments Corporation 2007) (Microsoft Corporation 2007).

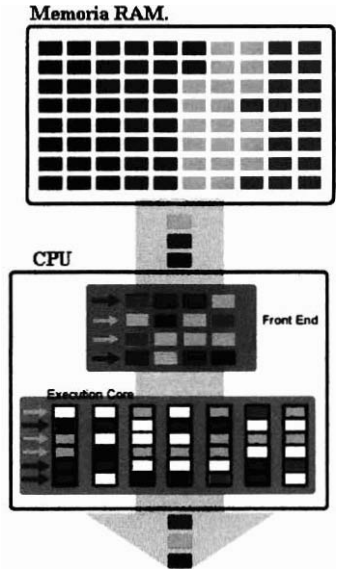


**Figura 9: Multiprocesamiento Virtual,** caricatura representativa, las tareas a ejecutarse son almacenadas como instrucciones en la memoria RAM distinguidas entre sí por colores verde, rojo, amarillo y morado, estas instrucciones son enviadas de manera ordenada al procesador, en donde son recibidas por el Front End del procesador y ordenadas según su naturaleza,(datos, instrucción, salto),de ahí son enviadas al núcleo de ejecución (Execute Core) para ser ejecutadas de una a la vez, de esta manera el CPU ejecuta diferentes instrucciones que diferentes tareas creando así un efecto de paralelismo real.

Los programas de hoy en día llevan acabo diferentes funciones, las cuales en ocasiones pueden requerir de la espera de resultados, la finalización de otra función, acceder a una localidad de memoria, etc. Para mejorar este esquema de paralelismo, el sistema operativo

crea con cada proceso en ejecución uno o varios hilos de ejecución (*threads*), los cuales permiten que un proceso sea ejecutado como un conjunto de unidades de ejecución relacionadas, lo cual es mucho más eficiente que una colección de procesos. Estos *threads* comparten entre si los recursos del proceso, poseen mecanismos de comunicación y llamado a ejecución, se alojan en el mismo espacio de direcciones y tiene acceso a los mismos datos, de esta manera cuando un hilo modifica un dato en memoria los otros utilizan el resultado cuando acceden al dato. Este esquema también es optimizado por el *kernel* al crear una tabla de prioridades (*sheduler*) y así decidir qué *thread* es ejecutado en cada *timeslice*. De esta manera el sistema operativo permite contar con un esquema que, en base a fenómenos de concurrencia (varios procesos compitiendo por los recursos de un solo procesador) permite crear un modelo virtual de paralelismo (Microsoft Corporation 2007).

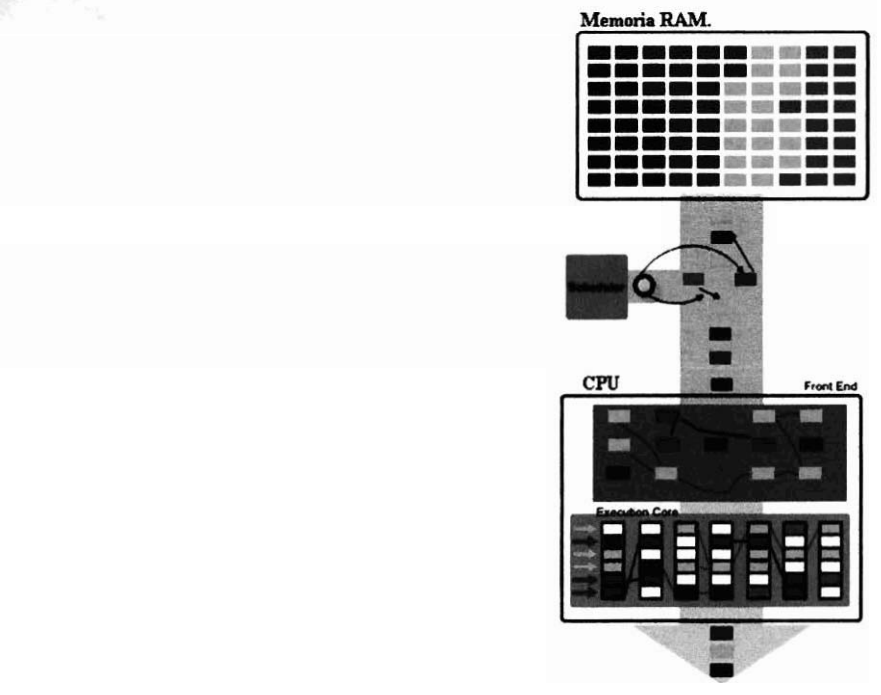


**Figura 10: Procesamiento Multihilo**, Esquema representativo del procesamiento multihilo de un CPU, las tareas a ejecutarse son almacenadas como instrucciones en la memoria RAM distinguidas entre sí por colores verde, rojo, amarillo y morado, estas instrucciones son enviadas al Sheduler que determina el orden en que se enviaran al CPU, dentro del CPU las características de los Threads permiten que diferentes tareas correspondientes a un mismo proceso conozcan el estado de sus variables, soliciten información e incluso soliciten sea ejecuta una instrucción pariente, este mecanismo permite realizar un multiprocesamiento virtual regido por esquemas de prioridades.

Lamentablemente el rendimiento de este paralelismo está determinado por el tipo de sistema operativo y el tipo de Kernel implementado en éste. Para el caso de las plataformas Windows que cuentan generalmente con Kernels híbridos, se tiene la desventaja de carecer de documentación y de privilegios para su modificación. Para el caso de las plataformas Linux se cuenta con Kernels monolíticos, los cuales pueden ser modificados o reemplazados, además de contar con gran documentación al respecto (Joseph 2001).

## 1.20 Ejecución en tiempo real.

Una ejecución en tiempo real es requerida para aplicaciones que deben ejecutar determinísticamente una tarea en software sin interrupciones o interferencias de otras tareas no-críticas. Las aplicaciones que se ejecutan determinísticamente muchas veces realizan una tarea crítica en iteraciones y estas iteraciones consumen siempre la misma cantidad de tiempo del procesador. De esta manera las aplicaciones determinísticas son apreciadas no tanto por su velocidad, sino por su confiabilidad a responder con consistencia a entradas y generar salidas con muy poco "parpadeo" (o Jitter). Un ejemplo claro de una aplicación determinística es un lazo de control de tiempo crítico. Un lazo de control de tiempo crítico adquiere información de un sistema físico y responde a esa información con una salida de tiempo muy precisa. Por ejemplo, en la industria del petróleo en donde cientos de metros de pipas se ensamblan al día. A medida que dos pipas son unidas desde sus dos puntas, un torque es requerido para girar las pipas, que embobinen y queden interconectadas. Suponga que la máquina que conecta las pipas usa un lazo de control que responde a los incrementos en la resistencia entre las pipas aplicando más torque. Ahora suponga que cuando se llega a un nivel crítico de torque, el lazo de control responde terminando la tarea. Bajo estas condiciones, el lazo debe ejecutarse determinísticamente porque un retraso en el software puede causar daños severos a las pipas y al equipo. Es necesario usar ejecución en tiempo real cuando se requiere garantizar que una tarea crítica se ejecutará de manera precisa en el horario establecido, comúnmente esto puede ser logrado al modificar las prioridades de ejecución del sistema operativo, obligándolo así a que tareas no criticas como el despliegue

grafico, no interfieran con el proceso de tiempo crítico (Joseph 2001) (National Instruments Corporation 2007) (Microsoft 2003) (Jim Beveridge 1996).

## Resultados y procedimientos realizados en esta tesis.

Como un apartado introductorio al trabajo realizado, en esta sección se presenta al lector de manera específica las problemáticas a solucionar en el desarrollo de este proyecto. Este apartado pretende también plantear a manera de hipótesis un bosquejo de diseño del sistema que en base a requerimientos y recursos pretende desarrollar una solución óptima para la creación del sistema.

## Requerimientos y recursos.

Como punto de partida para la elaboración de un modelo general de sistema, fueron considerados los recursos tecnológicos al interior del Laboratorio D-13 y la funcionalidad deseada por el usuario para el sistema, aspectos que delimitaran de manera directa el planteamiento general del proyecto.

Como un requerimiento funcional, es necesario que el sistema desarrollado sea sumamente flexible y amigable en su presentación con el fin de ser aplicado de manera transparente a esquemas experimentales de electrofisiología, microscopía y farmacología. De igual manera, el sistema deberá permitir al usuario contar con herramientas que agilicen el diseño de sus experimentos y al mismo tiempo le permitan tomar decisiones en tiempo real que modifiquen el flujo de sus experimentos, sin que estas tareas le requieran perder tiempo ni detalle de su labor experimental.

Todas estas funcionalidades deberán ser diseñadas tomando como base los recursos tecnológicos con los que se cuenta en el interior del Laboratorio de Neurofisiología Celular, el cual cuenta con 3 sets de experimentación, a continuación se presentan detalladamente los recursos en hardware y software con que cuenta cada uno de estos sets.

Set 1: Set de registro electrofisiológico.

| Elemento. | Descripción |
|---|---|
| Sistema operativo | Windows 98. |

| | |
|---|---|
| Microprocesador | Pentium III 900Mhz. |
| Memoria RAM | 256 MB |
| Puertos de comunicación libres. | LTP2. |
| Software implementado. | Pclamp9. |

**Tabla 7: Recursos del Set1 Laboratorio de Neurofisiología Celular.**

Set 2: Set de registro electrofisiológico 2.

| **Elemento.** | **Descripción** |
|---|---|
| Sistema operativo | Windows 98. |
| Microprocesador | Pentium III 600Mhz. |
| Memoria RAM | 128 MB |
| Puertos de comunicación libres. | LTP2. |
| Software implementado. | Pclamp9. |

**Tabla 8: Recursos del Set2 Laboratorio de Neurofisiología Celular.**

Set 3: Set de microscopia.

| **Elemento.** | **Descripción** |
|---|---|
| Sistema operativo | Windows XP. |
| Microprocesador | Intel Xeon 3.0Ghz Dual. |
| Memoria RAM | 2 GB |
| Puertos de comunicación libres. | LTP2, Serial, USB. |
| Software implementado. | Image Pro Plus. |

**Tabla 9: Recursos del Set3 Laboratorio de Neurofisiología Celular.**

Contemplados estos recursos, se hace evidente que el sistema desarrollado deberá ser compatible con Win32 y utilizar algún puerto de comunicación simple como LTP1, SERIAL o USB.

## Retos y problemáticas.

Una vez descrito el proceso experimental y analizados los productos comerciales, es evidente que la temporización y la inocuidad funcional del sistema son características que determinarán el éxito o fracaso de este proyecto.

Para el caso de la temporización, la problemática a resolver es asegurar que el mecanismo que realiza el control lógico de la aplicación de fármacos realice este control sin retraso ni variación en su comportamiento. Trasladando esto al terreno del software el problema a resolver es asegurar que el control de soluciones se realice cada segundo exacto, sin retraso ni modificación en su funcionamiento, en estos mismo términos existe una limitante pues hay que recordar que la mayoría de los sistemas operativos no son ejecutados en tiempo real y a su vez los programas son ejecutados según un orden de procesamiento, de igual manera es necesario recordar que toda interfaz mecánica tiene tiempos de retardo y respuesta que requieren ser contemplados.

Para el caso de la inocuidad funcional la problemática a solucionar es asegurar que los materiales seleccionados para la fabricación sean materiales inocuos que no interactúen con los fármacos utilizados y que de alguna manera alteren los resultados.

## Hipótesis.

Es posible crear un sistema de perfusión controlado por software, centralizando el control de válvulas en una PC. Diversas técnicas y metodologías como la programación de hilos de ejecución, modificación de la prioridad de ejecución del programa, inclusión de parámetros mecánicos y algoritmos de compensación de retardos pueden otorgar al sistema un alto grado de certeza en su ejecución temporizada. Este software puede ser desarrollado enteramente en C++ y mediante un entorno visual como CBuilder otorgarle una apariencia sencilla, y al mismo tiempo otorgarle valor agregado incluyendo herramientas como bases de datos. La utilización de estas plataformas de programación permiten crear un programa enteramente compatible con Win32 asegurando que pueda ser implementado en cualquier set experimental del Laboratorio.

## Ventajas.

El modelo propuesto (Figura 11) pretende crear una solución confiable en su programación para la temporización de las aplicaciones, utilizando los recursos existentes al interior del laboratorio, evitando así la erogación de gastos adicionales para su implementación. Así mismo, este modelo permite la creación de múltiples rutinas y herramientas en programación que flexibilicen de manera amistosa el trabajo del experimentador, creando así una herramienta, segura, económica y fácil de usar.

## Desventajas.

Este modelo fue diseñado de manera obligatoria bajo una plataforma Windows (Win32) puesto que utilizar otro sistema operativo implicaría realizar la compra de licencias de software que en algunos casos representan costos de hasta 5000 Dólares. Si bien es posible crear aplicaciones en tiempo real con un buen nivel de certidumbre en plataformas Win32 (Jim Beveridge 1996), la naturaleza del sistema operativo no es la optima para estas tareas e incluso pueden verse afectadas por el usuario mismo al exponer el equipo a cargas de trabajo superiores a los recursos con que cuenta, por lo tanto este modelo presupone que los equipos de experimentación son utilizados de manera racional y dedicada a las tareas de experimentación, y que el usuario no interferirá con el flujo normal de carga de trabajo.

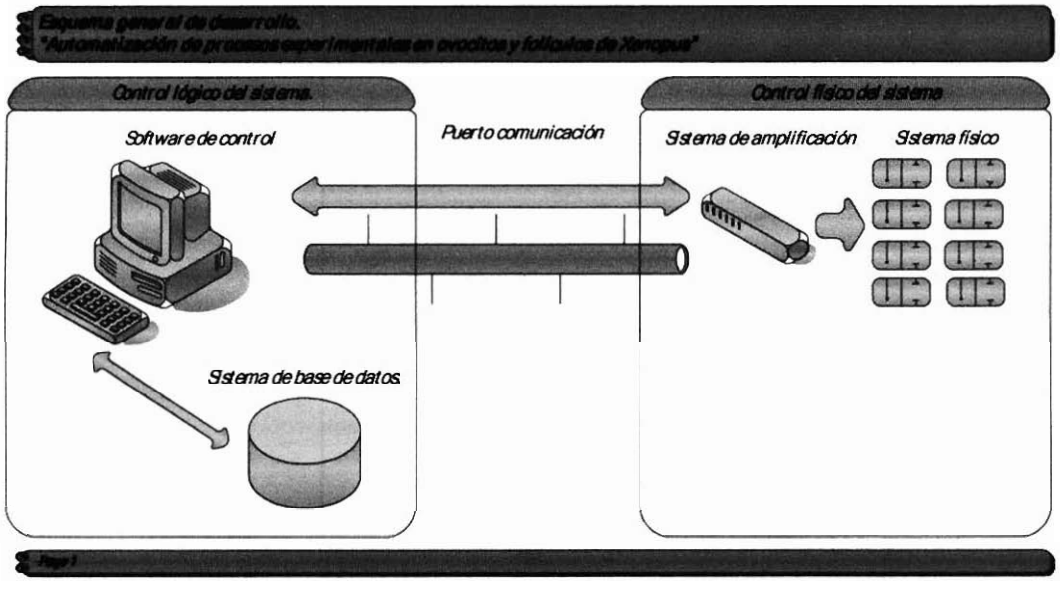


**Figura 11: Esquema general del desarrollo.**

# Capitulo 2. Diseño y desarrollo de sistema.

## 2.0 Introducción.

En este capítulo se presentan el trabajo realizado en el diseño y construcción física del sistema, se detallan además la selección de materiales y sus características. De igual manera se presenta el diseño electrónico del sistema de control que permiten realizar la interfaz PC – Válvulas.

## 2.1 Selección de materiales de construcción.

Los materiales seleccionados para la construcción del sistema fueron aluminio y MDF, para el caso de este último se deseaba contar con materia fácil de modificar en su diseño debido a que la versión del sistema correspondía a un prototipo. Una vez depurado el diseño se procederá a construirlo completamente en aluminio. Para el caso de la tubería de fármacos se implementará en manguera de $1\frac{1}{16}$ de diámetro interno de teflón, los contenedores de fármacos serán implementados en jeringas de 50 ml.

Para el caso de las válvulas, se selecciono equipo del fabricante McMaster-Carr, equipo que cuenta con certificación sanitaria y del cual se tiene un referente de calidad al interior del laboratorio.

## 2.2 Diseño físico del sistema.

El sistema fue diseñado buscando fabricar una pieza liviana, fácil de instalar y fácil de remover de las mesas de experimentación. Este diseño se dividió en cuatro piezas fundamentales, el soporte de los contenedores, el soporte de las válvulas, el pedestal del sistema y la base del sistema. Los planos para su construcción se encuentran en el Apéndice (A).

**Soporte de contenedores.**

Esta pieza está diseñada para colocar ocho jeringas de 50 ml en su estructura, al mismo tiempo que permite ser montada y ajustada a una altura deseada en el pedestal.

**Soporte de las válvulas.**

Esta pieza está diseñada para alojar las válvulas de solenoide y la circuitería de control, al mismo tiempo que permite sea montada y ajustada a una altura deseada en el pedestal.

**Pedestal.**

Esta pieza está diseñada para alojar los contenedores de válvulas y fármacos, está ranurada en su cuerpo para permitir sea modificada la altura de alojamiento, también se encuentra ranurada en su parte inferior para ser anclada a la base del sistema.

**Base del sistema.**

Se empleó un imán electromagnético como base del sistema, permitiendo así instalar y remover de manera sencilla el sistema en una mesa de trabajo estándar.

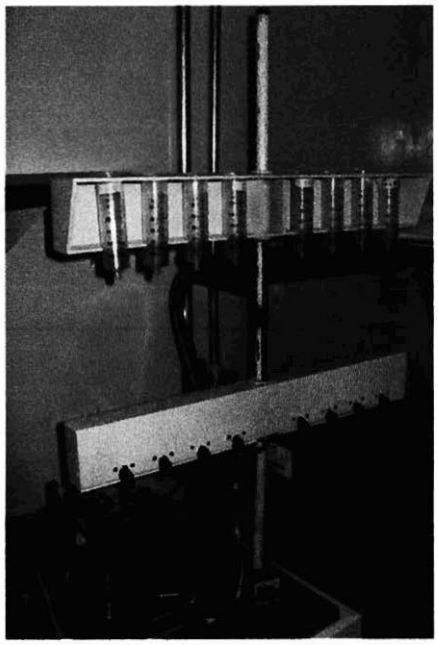

**Figura 12: Sistema prototipo, fotografía del sistema diseñado.**

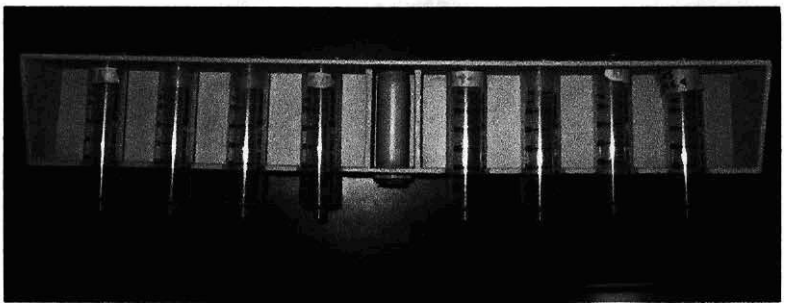

**Figura 13: Soporte de contenedores, fotografía de la pieza que aloja los contenedores de fármacos.**

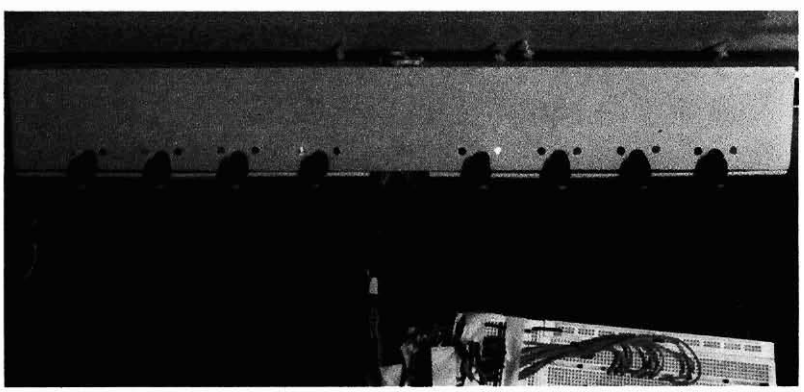

**Figura 14: Soporte de contenedores, fotografía de la pieza que aloja el sistema de válvulas y circuitos de control.**

## 2.3 Características de los solenoides.

Las válvulas de solenoide seleccionadas para la fabricación del sistema implementan un mecanismo básico de embolo que abre y cierra el flujo de solución, están fabricados en teflón y su diseño permite realizar mantenimientos de rutina de manera sencilla desatornillando la parte superior. Los valores operativos nominales descritos por el fabricante corresponden a (Tabla 10).

| Característica. | Valor |
|---|---|
| Voltaje Operativo nominal | 12V DC. |
| Consumo de corriente nominal | 400mA. |
| Tipo de flujo | Normalmente cerrada. |

**Tabla 10: Tabla de valores operativos nominales de los solenoides.**

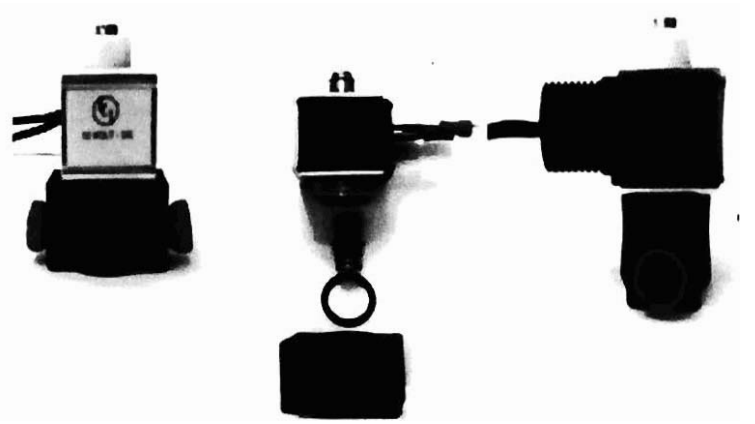

**Figura15: Válvulas de solenoide implementadas en sistema.**

## 2.4 Análisis funcional de los solenoides.

Con el fin de contar con la suficiente información para el diseño de la interfaz de potencia PC – Válvula y el sistema que provea de energía a las válvulas, fueron aplicadas diferentes metodologías que permiten conocer los niveles operativos de los solenoides, así como el consumo de potencia en cada uno de estos niveles. De la misma manera estas metodologías permiten identificar el tiempo de retardo al cierre y el tiempo de retardo a la apertura, valores que fueron sumamente importantes para el diseño del software controlador.

### 2.4.1 Consumo de potencia.

Mediante un circuito de alimentación compuesto por una fuente variable, un multimetro y un osciloscopio, fueron determinados los valores para el consumo de potencia de los solenoides, si bien esta grafica (Figura 16) nos indica la corriente que consume el solenoide en determinado voltaje, también nos permite identificar el voltaje y corriente necesaria para

que el solenoide realice el trabajo mecánico de apertura o cierre. Estas pruebas determinaron que los solenoides realizan la apertura al nivel de 6.8 Volts con un consumo de 0.24mA, y el cierre a un nivel descendente de 1.6 Volts con un consumo de 0.06mA.

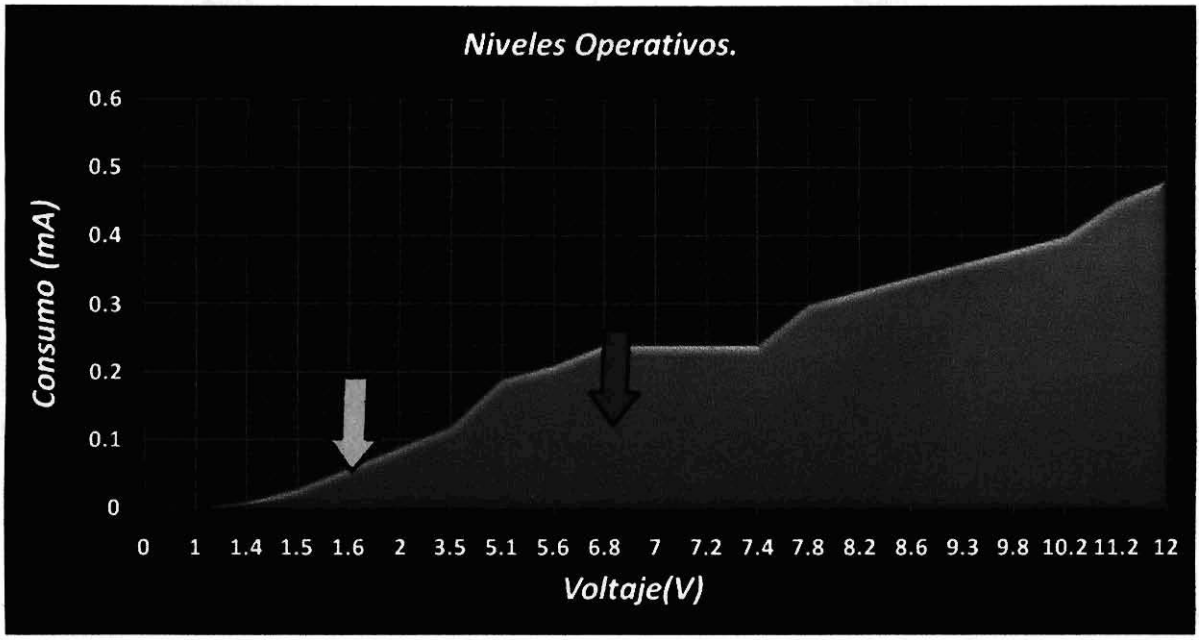


**Figura 16: Niveles operativos válvulas de solenoide.** La grafica identifica los niveles operativo de voltaje y tensión necesarios para que los solenoides realicen el trabajo mecánico de apertura (⬇) y cierre (⬇).

## 2.4.2 Retardos de apertura y cierre.

Como ya fue expuesto en el Capítulo 1, la naturaleza electromagnética de un solenoide acarrea retardos para la respuesta mecánica del mismo, conocidos los niveles operativos de los solenoides (Figura 16) y teniendo como base la teoría electromagnética, fue posible determinar el lapso de tiempo necesario para realizar una polarización y apertura así como una despolarización y cierre.

Estos tiempos fueron encontrados mediante una fuente de voltaje y un osciloscopio, el cual permitió contar con una grafica que describe la polarización y despolarización del solenoide. Esta metodología arrojó que el tiempo teórico que el solenoide requiere para realizar una apertura es de 35 ms, y para el caso del cierre es de 314ms.

Si bien estos valores encontrados identifican el tiempo total para que el solenoide realice una polarización y despolarización total, es necesario recordar que el trabajo mecánico no se realiza en magnitudes totales, por lo que realizando una equivalencia entre el tiempo de polarización y los niveles operativos se puede determinar que el tiempo real de retardo de apertura es de 18ms (Figura 17) y el tiempo real de retardo de cierre es de 283ms (Figura 18).

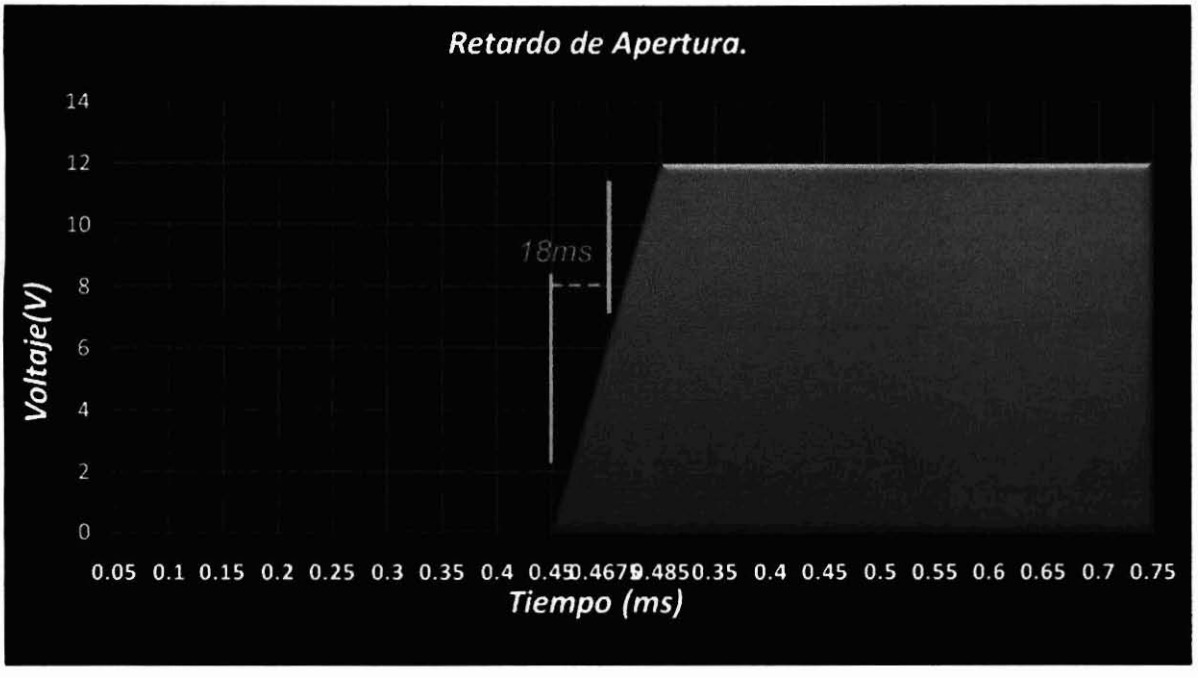


**Figura 17: Retardo de apertura de solenoides, la grafica identifica el tiempo necesario para que el solenoide alcance un consumo de voltaje y corriente necesario para realizar el trabajo mecánico de apertura.**

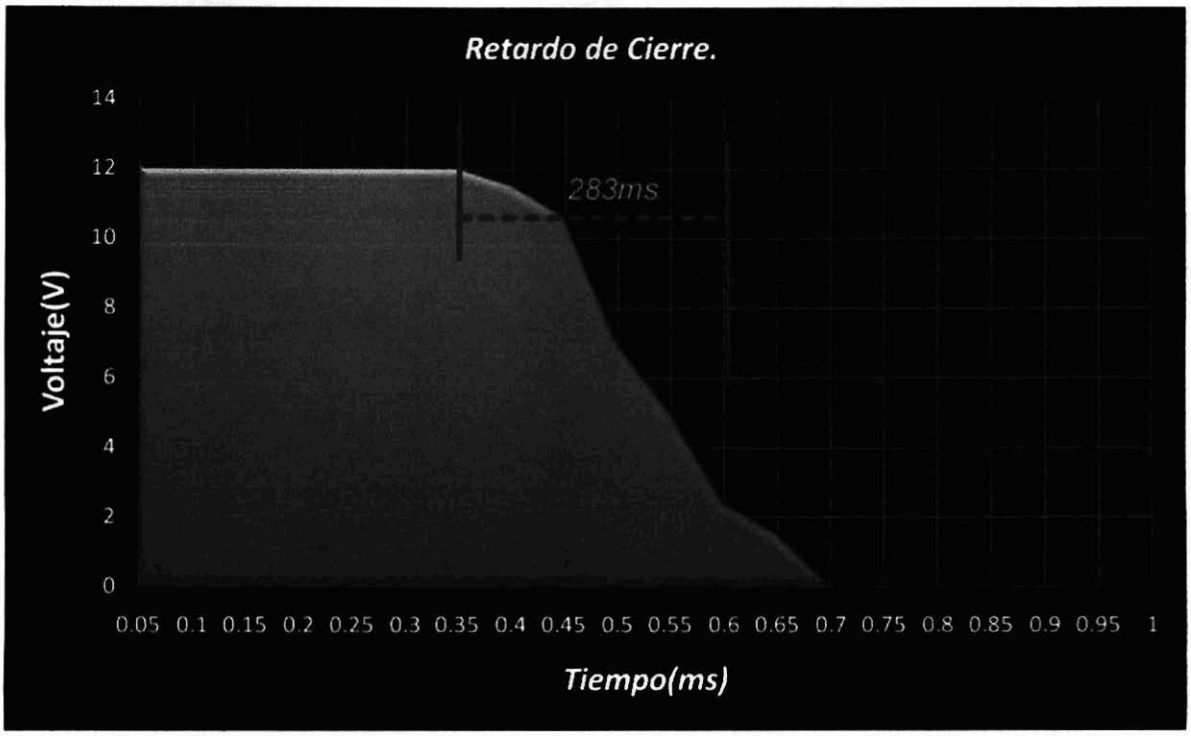


**Figura 18: Retardo de cierre de solenoides, la grafica identifica el tiempo necesario para que el solenoide se despolarice y el trabajo mecánico de cierre sea realizado.**

## 2.5 Diseño de las placas de control.

Fueron diseñadas y fabricadas dos placas de circuitos que funcionan como equipo de control y amplificación. Estas placas a grandes rasgos permiten mediante niveles de voltaje de 0-5 V de un puerto paralelo de una PC, controlar los solenoides que operan en rangos de 12 V, al mismo tiempo estos circuitos realizan una tarea de buffer de protección evitando que corrientes altas y cortocircuitos lleguen al puerto. Los circuitos integrados utilizados en este diseño son:

- SN7404.
- ULN2803A.
- Resistencias 330Ω.
- Clemas de 8 pines.

- Diodos.
- Bases de circuitos.

Para el caso del SN7404, cuya función es de un inversor binario (Texas Instruments 1983), se utilizó para implementar indicadores luminosos "LED,s" para el estatus de cada válvula, verde para encendida, rojo para apagada.

En el caso del ULN2803A, cuya función es la de driver (Motorola Inc. 1996), es utilizado como circuito de amplificación y buffer de protección, el resto de los componentes como clemas y bases tienen la finalidad de contar con mecanismos prácticos para su instalación y posterior mantenimiento.

Estas placas se encuentran interconectadas entre sí por los voltajes GND, 5V Y 12V. Estas conexiones así como el diseño completo de las placas pueden identificarse en la Figura 19.

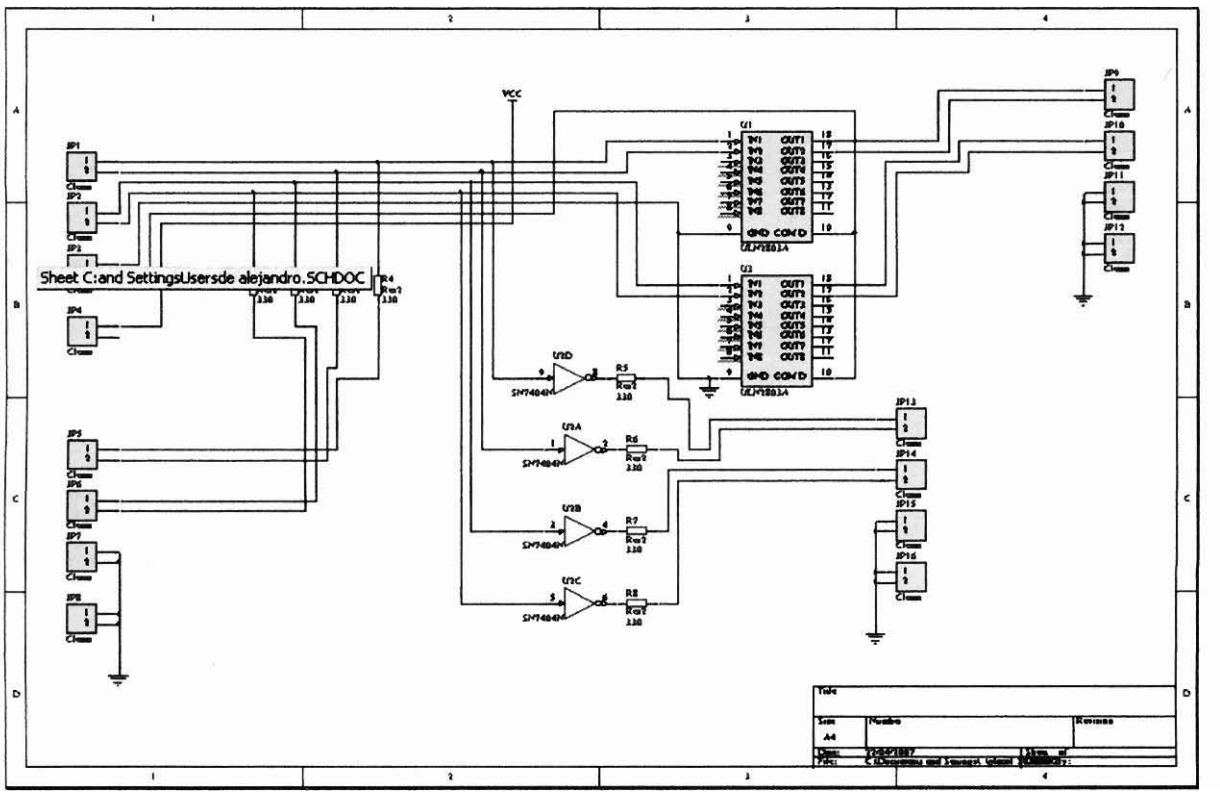


**Figura 19: Plano de conexiones para placas de control.**

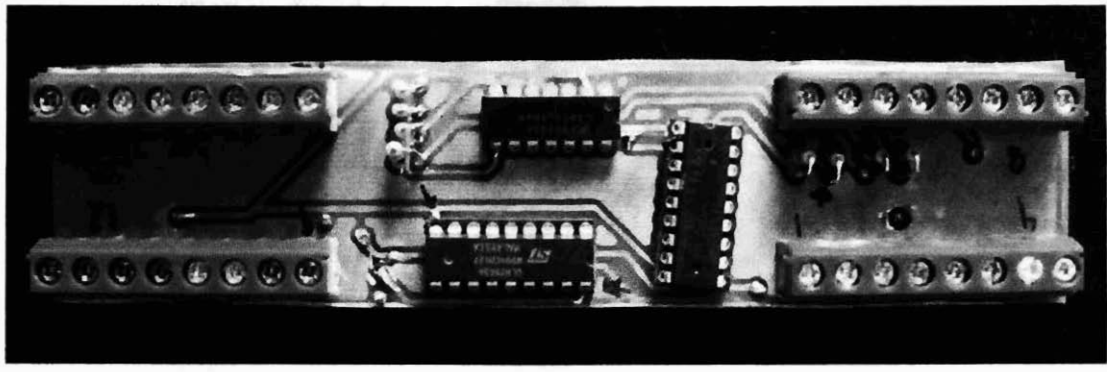

**Figura 20: Placas de control, Fotografía de las placas de control fabricadas.**

## 2.6 Método de fabricación de placas.

Utilizando software de diseño electrónico (Prótel 2005) se diseñó el modelo PCB para la fabricación de placas mediante la técnica de serigrafía y barrido de acido, estos diseños pueden ser consultados en el Apéndice B.

# Capitulo 3. Desarrollo del software controlador.

## 3.0 Introducción.

En este capítulo se presentan los puntos más destacados en la caracterización, diseño y programación del software controlador, cuyo objetivo es permitir al experimentador elaborar de manera ágil una rutina de aplicaciones de fármacos que determina la apertura y cierre de válvulas. Diversas consideraciones técnicas fueron implementadas con el fin de asegurar que mecanismos de reproducibilidad e inclusión de parámetros fueran adoptados.

## 3.1 Selección de plataformas.

Fue seleccionado Cbuilder 4 como plataforma de desarrollo, Paradox como sistema de base de datos y el puerto LTP1 como el puerto interfaz con el sistema de válvulas. Esta selección se basó en la premisa de crear un sistema capaz de ejecutarse en sets de experimentación con recursos en hardware limitados.

En el caso del puerto LTP1, su selección obedece a evitar una problemática con el software de experimentación que es ejecutado actualmente en los sets, el cual en la mayoría de los casos utiliza sistemas de validación mediante llaves en puertos, sin embargo considerando las perspectivas de implementación futura fueron estudiadas y desarrolladas diversas opciones de comunicación y control mediante otra interfaz como RS232 y USB.

## 3.2 Caracterización del proceso experimental.

El proceso experimental fue modelado en conjunto con el experimentador mediante diagramas de flujo con el fin de comprender en su totalidad sus variantes y procedimientos, en este modelado se encontró que el proceso experimental puede dividirse en cuatro grandes fases correspondientes a:

1. Creación de un protocolo.
2. Creación de una rutina de aplicaciones.
3. Ejecución de rutina.
4. Almacenamiento de protocolo.

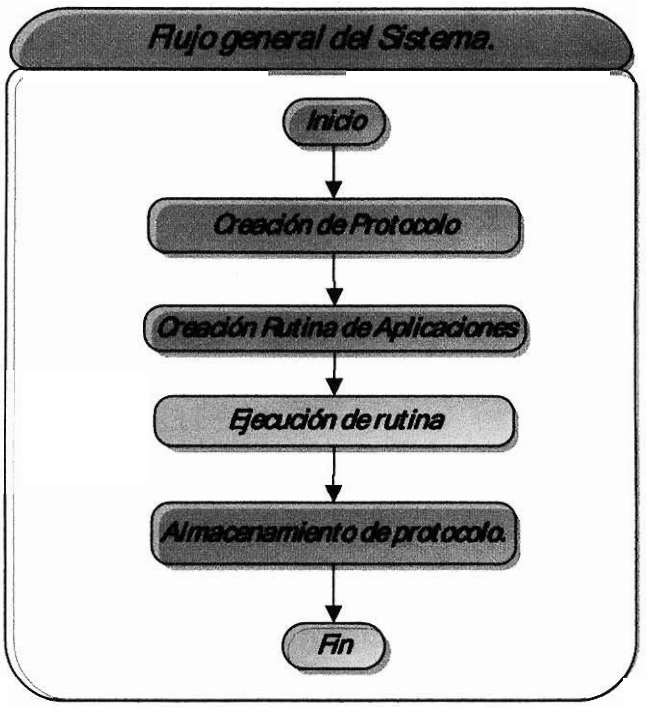


**Figura 21: Flujo general del proceso experimental de perfusión.**

## 3.3 Descripción de las fases.

### 3.3.1 Creación de un protocolo.

En esta fase la información general del proceso experimental es ingresada. Esta información corresponde a datos como el nombre asignado al protocolo, el nombre de la persona que lo genera, la velocidad de perfusión a la que operara, el número de canales que utilizará así como su contenido y concentración.

### 3.3.2 Creación de una rutina de aplicaciones de fármacos.

En esta fase es creada una relación tiempo – fármaco que regirá la apertura y cierre de las válvulas. Debido a que existen algunas variantes del proceso experimental, esta fase fue

dividida en modalidades, de esta manera fue posible crear mecanismos que agilicen la creación de las rutinas mediante módulos de software específicos a cada modalidad.
Experimentalmente encontramos que la creación de rutinas de aplicaciones puede ser diseñada bajo cuatro modalidades:

- El modo curva de concentraciones.
- El modo aplicación unitaria.
- El modo de aplicaciones empalmadas.
- El modo de corrientes mecánicas.

### 3.3.3 El modo curva de concentraciones.

En esta modalidad existe un fármaco base que tiene la función de solución de lavado y siete soluciones conteniendo el fármacos y que tienen una secuencia lógica en su aplicación según su concentración. La forma de ejecución de esta modalidad se basa en la aplicación lineal de un lavado-concentración repitiendo esta operación hasta terminar con la secuencia de concentraciones.

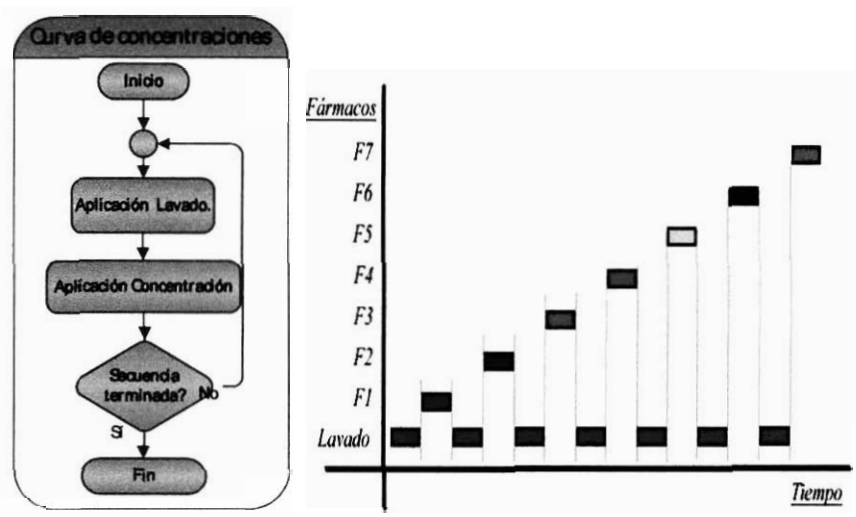


**Figura 22: Flujo operativo para la ejecución de una curva de concentraciones.**

### 3.3.4 El modo aplicación unitaria.

En esta modalidad existe un fármaco base que tiene la función de solución de lavado, pero el resto de los fármacos no tiene una secuencia regida por algún parámetro lógico, es decir la secuencia de aplicaciones se decide de manera arbitraria. La forma de ejecución de esta modalidad se basa en la aplicación lineal de un lavado - secuencia.

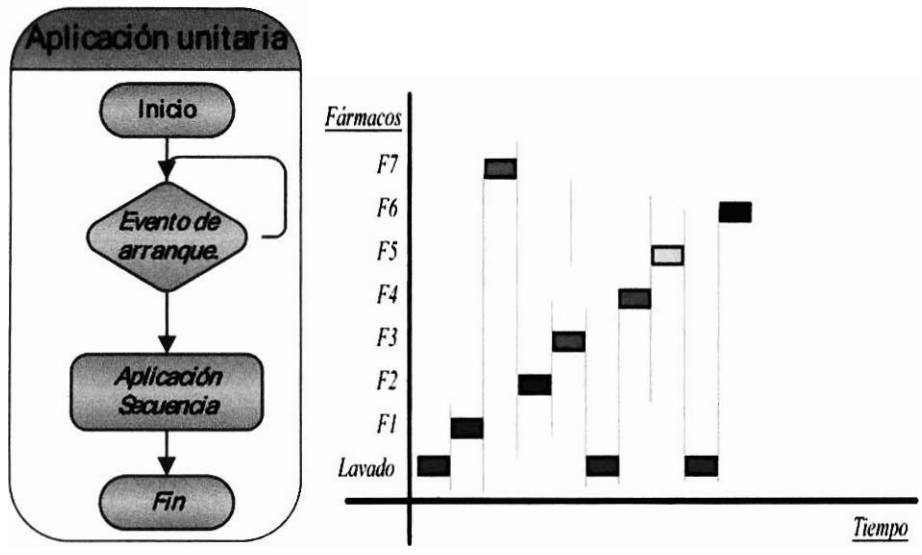


**Figura 23: Flujo operativo para la ejecución de una aplicación unitaria.**

### 3.3.5 El modo de aplicaciones empalmadas.

Es una modalidad semejante en funcionamiento al modo de aplicación unitaria pero con la variante de que es posible programar la secuencia de aplicaciones de manera no lineal, es decir aplicando dos o más fármacos al mismo tiempo.

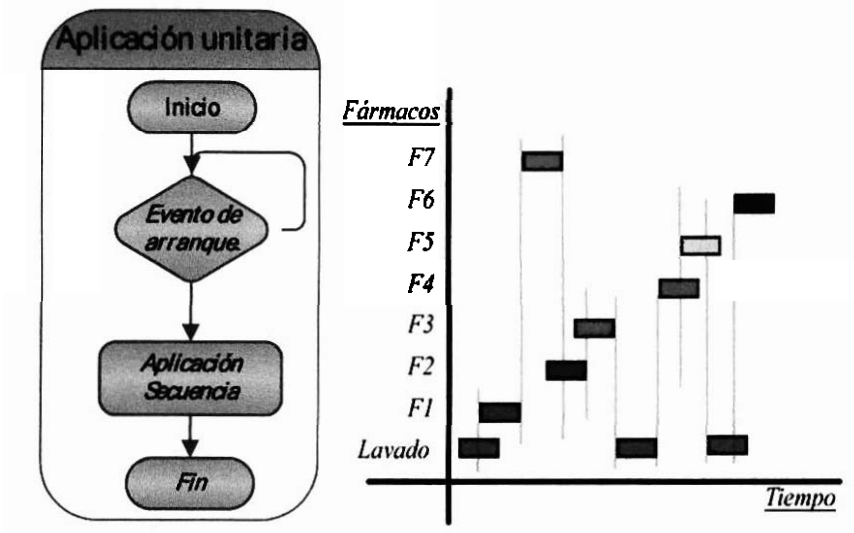


**Figura 20: Flujo operativo para la ejecución de aplicaciones empalmadas.**

## 3.3.6 El modo de corrientes mecánicas.

En este modo solamente es utilizado un fármaco base que tiene la función de solución lavado. En este caso en particular, la secuencia de aplicaciones es en realidad una serie de cortes en la aplicación de dicho lavado, la forma de ejecución de esta modalidad se basa en la aplicación de un lavado-corte repitiendo la operación el número de veces deseado.

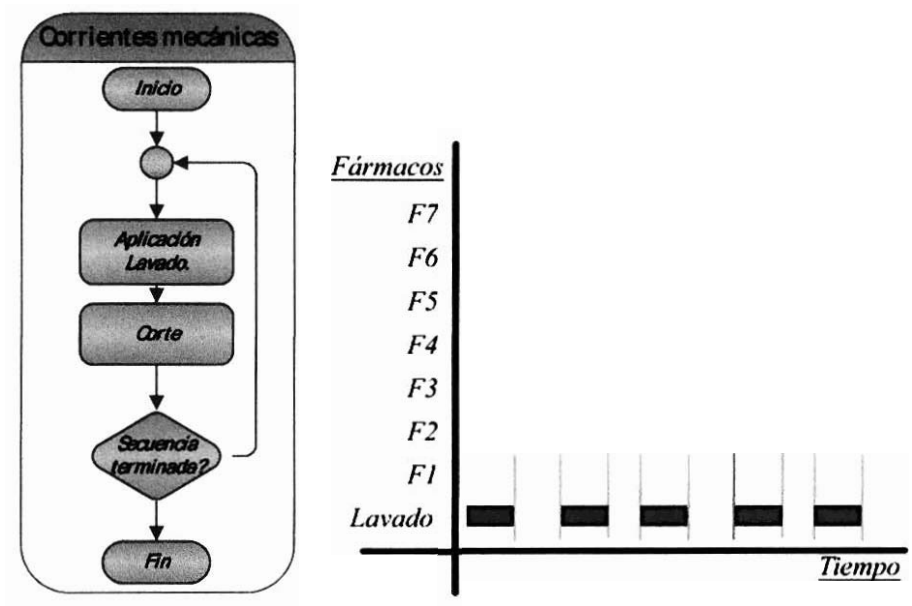


**Figura 24: Flujo operativo para la ejecución de corrientes mecánicas.**

## 3.4 Ejecución de rutina.

En esta fase, la rutina diseñada es ejecutada y se procede a controlar la apertura y cierre del sistema de válvulas, al igual que la creación de rutinas de aplicación, existen variantes en la ejecución del proceso experimental. Fueron identificadas tres formas básicas de ejecución las cuales son:

- Modo manual de ejecución.
- Modo independiente de ejecución.
- Modo sincronizado de ejecución.

### 3.4.1 Modo manual de ejecución.

Esta modalidad no requiere la creación previa de una rutina de aplicaciones, sino que la apertura y cierre de las válvulas es decide de manera arbitraria al momento mediante un mecanismo manual como oprimir una tecla.

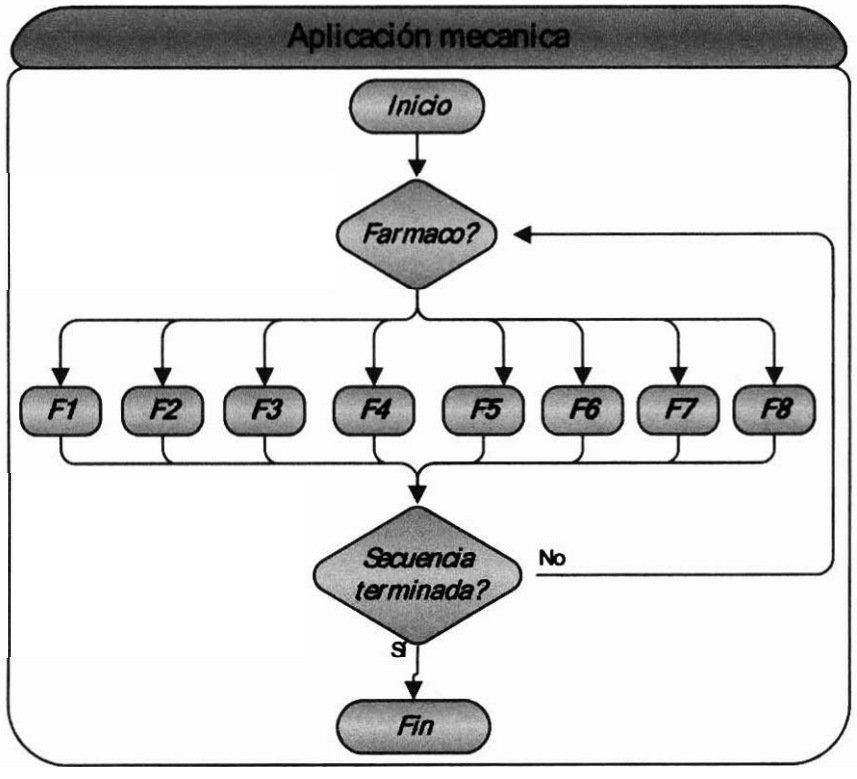


**Figura 25: Flujo operativo para la ejecución manual.**

### 3.4.2 Modo simple de ejecución.

Esta modalidad procede a ejecutar la secuencia de aplicaciones previamente planificada de manera directa al momento en que el usuario lo indica.

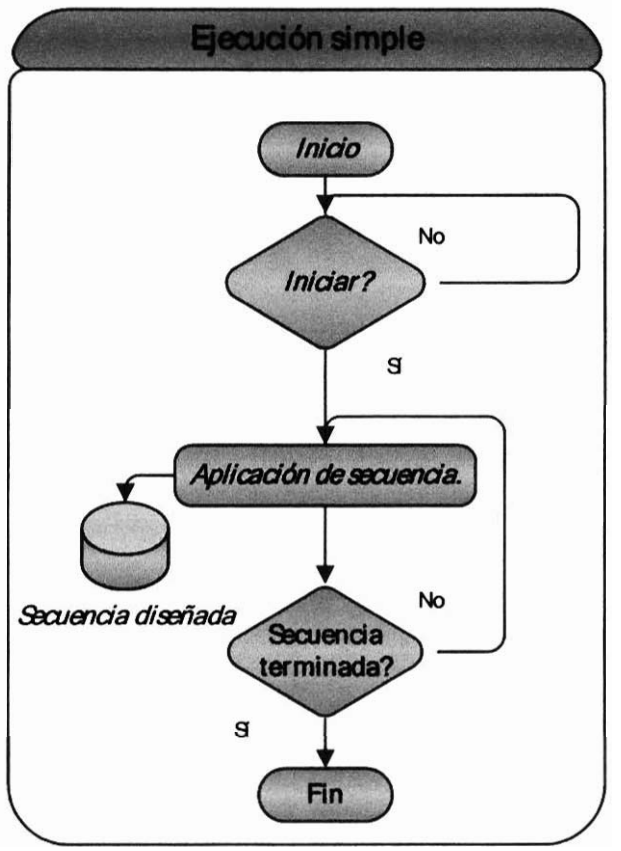


**Figura 26: Flujo operativo para el modo independiente de ejecución.**

### 3.4.3 Modo sincronizado de ejecución.

Esta modalidad ejecuta la secuencia de aplicaciones de manera sincronizada a un evento especifico, en este caso en particular, el objetivo de esta sincronización es permitir al usuario la ejecución de la secuencia de aplicaciones a la par del arranque de otros sistemas de software como sistemas de adquisición de datos o sistemas de captura de imágenes.

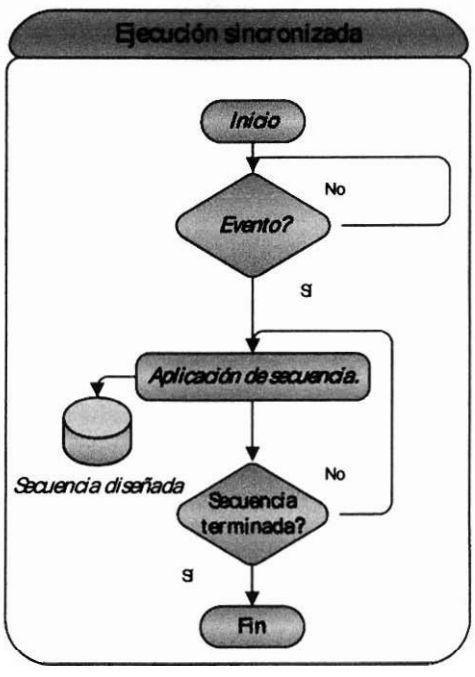


**Figura 27: Flujo operativo para el modo sincronizado de ejecución.**

## 3.5 Almacenamiento de protocolo.

En esta fase es almacenado el protocolo diseñado así como sus aplicaciones y resultados generados.

## 3.6 Funcionalidades.

Diversas funcionalidades fueron incluidas en el desarrollo con el fin de flexibilizar el funcionamiento del sistema, estas funcionalidades son:

- Búsqueda de protocolos.
    - Vista del diseño del protocolo.
    - Vista de resultados obtenidos.
- Lavado del sistema.
- Calibración de canales.

### 3.6.1 Búsqueda de protocolos.

Esta funcionalidad permite al usuario consultar sus protocolos previamente diseñados, consultar su composición, consultar sus resultados y si lo desea ejecutar nuevamente dicho protocolo.

### 3.6.2 Lavado del sistema.

Esta funcionalidad permite al usuario realizar la apertura y cierre de válvulas a libertad con el fin de realizar una limpieza en sus componentes físicos.

### 3.6.3 Calibración de canales.

Esta funcionalidad permite al usuario realizar una estimación de la velocidad de desplazamiento de fármacos, el sistema realizará la apertura controlada para un volumen determinado permitiendo así conocer la velocidad a la que se encuentra calibrado el sistema.

### 3.6.4 Requerimientos funcionales.

Descrito el proceso experimental y sus variantes, algunas de las funciones que el sistema obliga a desarrollar son:

- Parámetros del sistema.
    - Visualización de los parámetros físicos del sistema.
    - Modificación de los parámetros físicos del sistema.
    - Pruebas de rendimiento para parámetros.

## 3.7 Parámetros del sistema.

Esta funcionalidad permite al sistema incluir como variables de funcionamiento diversos parámetros físicos del sistema, en algunos de estos casos estos parámetros modifican el funcionamiento del software como es el caso de los retrasos de los solenoides, y en otros se trata de información que le permite al experimentador contar con mecanismos de reproductibilidad al conocer la configuración establecida para un experimento en particular, ejemplo, altura de los contenedores, altura de las válvulas, diámetros de la tubería, etc.

## 3.8 Modelado General del software de control.

Descrito el proceso experimental y contempladas las funcionalidades requeridas, fue posible modelar el sistema de control de una manera más específica (Figura 28) y planificar los flujos de operación así como describir las variables que en cada flujo serán requeridas.

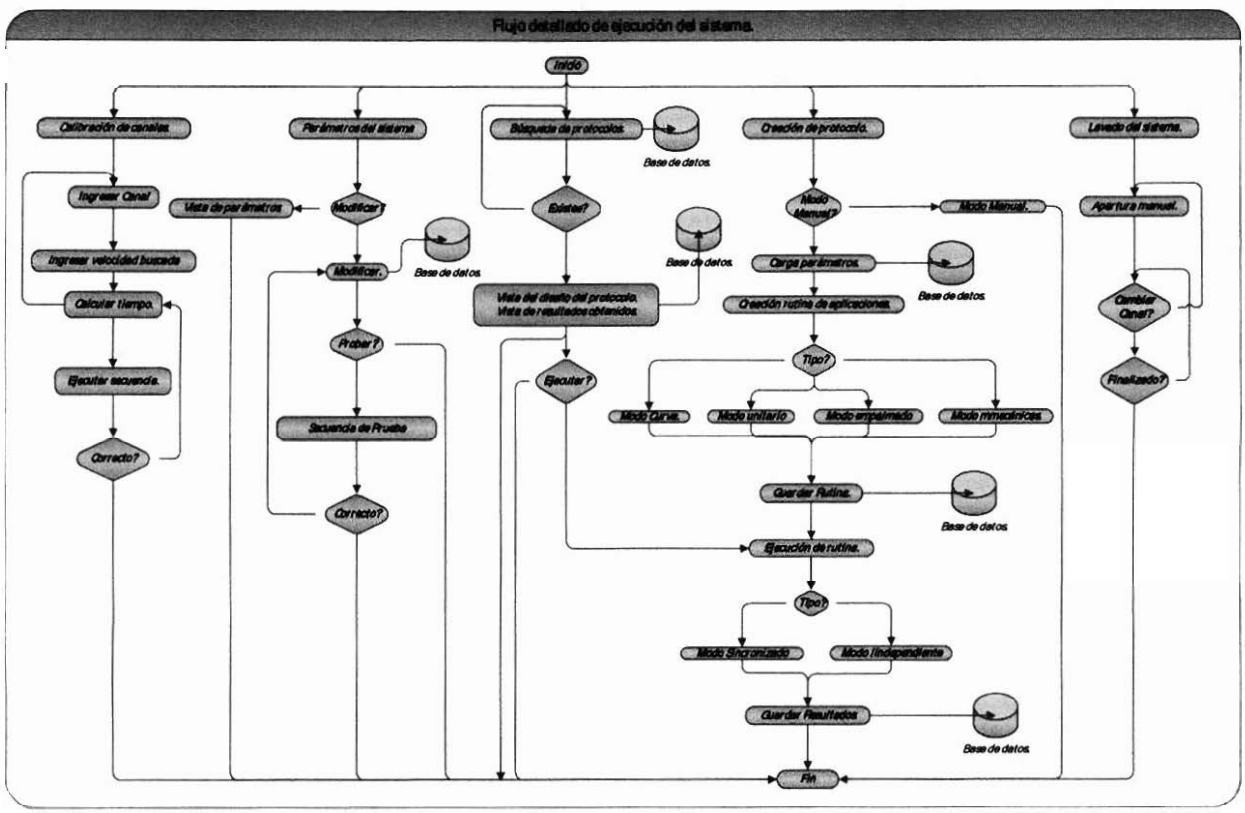


**Figura 28: Diagrama de flujo detallado del sistema de perfusión.**

## 3.9 Modelado de base de datos.

Una vez realizado el modelado funcional del software fue diseñada y construida la base de datos del sistema. El modelo pretende permitir a cada experimentador contar con sus propios diseños de protocolos, permitiéndole consultar su diseño, modificarlo, así como acceder a ellos y ejecutarlos nuevamente:

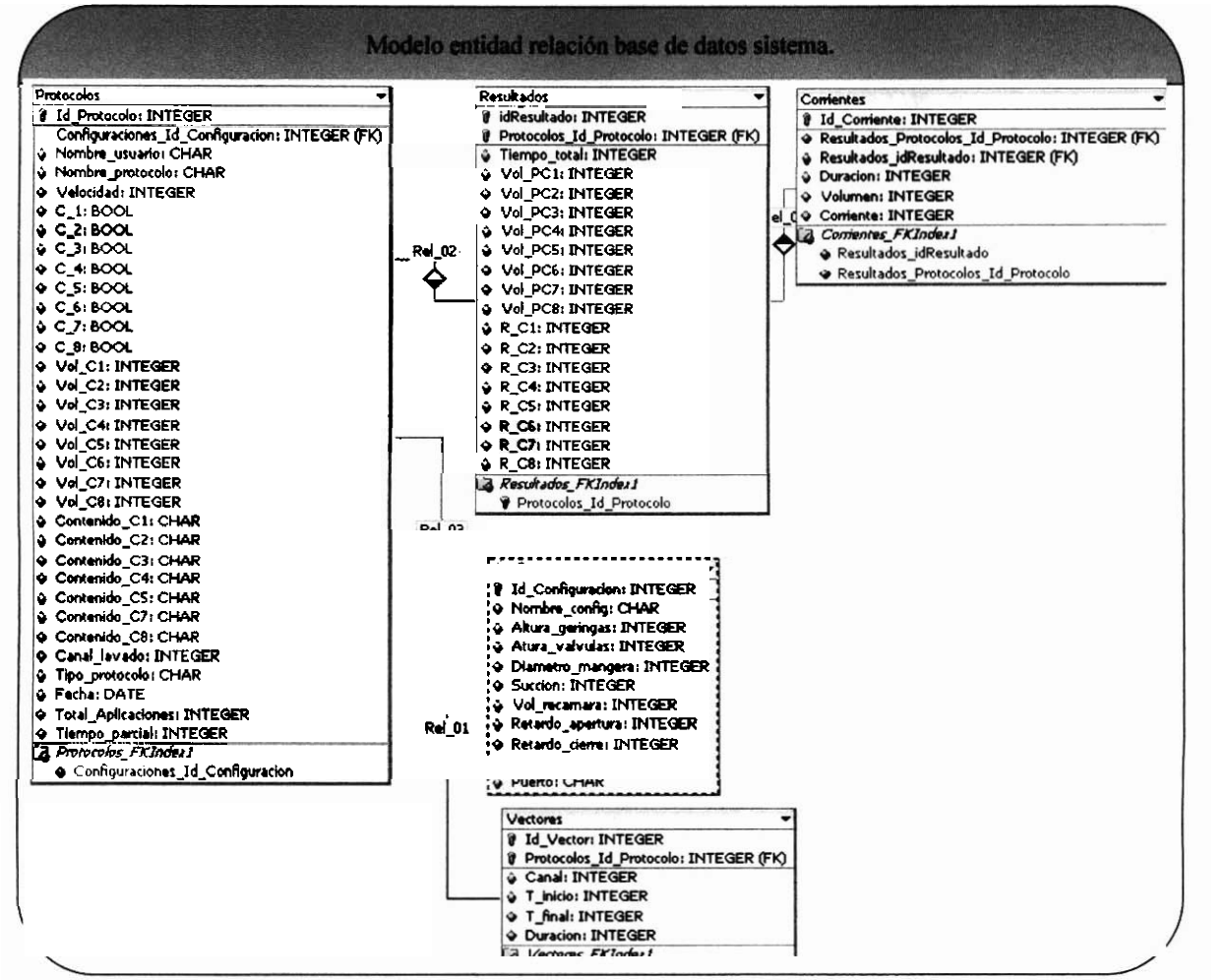


**Figura 29: Modelo Entidad – Relación base de datos del sistema.**

## 3.10 Diccionario de datos.

La información guardada en la base de datos así como la función de cada uno de estos campos está descrita en el diccionario de datos que se encuentra en el Apéndice C.

## 3.11 Programación del sistema.

Modelado el proceso experimental y creada la base de datos del sistema, se inició la programación de los módulos correspondientes a cada funcionalidad, los siguientes apartados corresponden a los puntos más destacados dentro del proceso de programación.

### 3.11.1 Control y acceso al puerto paralelo.

El acceso y control del puerto paralelo es la función básica que el sistema debía realizar, su programación y control se basó en la utilización de la Inpout32.dll. Si bien esta DLL nos permite realizar acceso a cualquier puerto de la PC mediante funciones input, output, su utilización requiere la definición de algunas variables, el instanciamiento de librerías y un método especifico para el envió de datos.

Para su utilización dentro de CBuilder es necesario:

- Definir dos tipos de datos correspondientes a las funciones de entrada y salida.

```
typedef short _stdcall (*inpfuncPtr)(short portaddr);
typedef void _stdcall (*oupfuncPtr)(short portaddr, short datum);
```

- Instanciar las librerías de la Inpout32.dll.

```
HINSTANCE hLib;
inpfuncPtr inp32;
oupfuncPtr oup32;
```

- Definir una variable tipo short que tiene la función de indicar el valor que desea enviarse al puerto:

```
short k;
```

- Definir una variable de tipo entero que tiene la función de indicar el valor de registro del puerto al que deseamos acceder, en el caso del puerto paralelo típicamente es utilizado el 0408.

```
int j;
```

Una vez definidas las variables e instanciadas las librerías se proceden a cargarlas y a realizar la comunicación con la DLL de la siguiente manera:

```
hLib = LoadLibrary("inpout32.dll");
inp32 = (inpfuncPtr) GetProcAddress(hLib, "Inp32");
 if (inp32 == NULL)
 {
  Application->MessageBox("ERRO AL ACCEDER A PUERTO PARALELO","",MB_OK);
```

```
  };
oup32 = (oupfuncPtr) GetProcAddress(hLib, "Out32");
if (oup32 == NULL)
{
Application->MessageBox("ERRO AL ACCEDER A PUERTO PARALELO","",MB_OK);
};
```

Si el proceso de comunicación con la dll fue exitoso es posible enviar y recibir valores al puerto de la siguiente manera:

Enviar valores:

(oup32)(j,k);

Leer valores:

K=(inp32)(i);

El siguiente de código corresponde a una forma en Cbuilder que permite mediante la selección de distintos combobox el envió de diferentes valores a puerto paralelo.

```
//---------------------------------------Forma lavado -----------------
#include <vcl.h>
#pragma hdrstop
#include "Lavado.h"
//---------------------------------------------------------------------
----
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm6 *Form6;
     typedef short _stdcall (*inpfuncPtr)(short portaddr);
     typedef void _stdcall (*oupfuncPtr)(short portaddr, short datum);
//--DECLARACION PARA PUERTO
     HINSTANCE hLib;
     inpfuncPtr inp32;
     oupfuncPtr oup32;
     short k;
     int j;
int salida=0;
//---------------------------------------------------------------------
----
__fastcall TForm6::TForm6(TComponent* Owner)
        : TForm(Owner)
{
hLib = LoadLibrary("inpout32.dll");
        j=0x378;
        inp32 = (inpfuncPtr) GetProcAddress(hLib, "Inp32");
        if (inp32 == NULL)
                {
```

```
                Application->MessageBox("ERRO   AL   ACCEDER   A   PUERTO
PARALELO","",MB_OK);
                };

        oup32 = (oupfuncPtr) GetProcAddress(hLib, "Out32");
        if (oup32 == NULL)
                {
                Application->MessageBox("ERRO   AL   ACCEDER   A   PUERTO
PARALELO","",MB_OK);
                };
        k=0x00;
        (oup32)(j,k);
}
//---------------------------------------------------------------------------
----
void __fastcall TForm6::SpeedButton1Click(TObject *Sender)
{
salida=0;
if(ComboBox1->Text== "On")
        {
        salida=salida+1;
        }
if(ComboBox2->Text=="On")
        {
        salida=salida+2;

if(ComboBox3->Text== "On")
        {
        salida=salida+4;
        }
if(ComboBox4->Text== "On")
        {
        salida=salida+8;
        }
if(ComboBox5->Text== "On")
        {
        salida=salida+16;
        }
if(ComboBox6->Text== "On")
        {
        salida=salida+32;
        }
if(ComboBox7->Text== "On")
        {
        salida=salida+64;
        }
if(ComboBox8->Text== "On")
        {
        salida=salida+128;
        }
        k=salida;
        (oup32)(j,k);
        }
```

### 3.11.2 Carga en RAM de rutina de ejecución

El diseño de las rutinas de aplicación fue agilizada mediante módulos específicos, que incluyen métodos propios para la generación automática del tipo de rutina. Creada esta rutina es necesario almacenarla en la base de datos y cargarla en memoria RAM para su ejecución. Para este fin fue creada una matriz binaria de (8 * n), que representa las válvulas (filas 0-7) y su activación o desactivación en cada segundo de la ejecución (columnas 0-n). La última fila corresponde a la línea de tiempo que le permite identificar al programa el momento en que un switcheo de válvulas se llevará a cabo y será necesario un ajuste de tiempo, esta línea de tiempo corresponde al tiempo de cierre de cada aplicación y es calculado he incluido en la tabla de aplicaciones por cada una de las modalidades..

Este procedimiento se adoptó como todas las modalidades de creación de rutinas, así como para las modalidades de búsqueda.

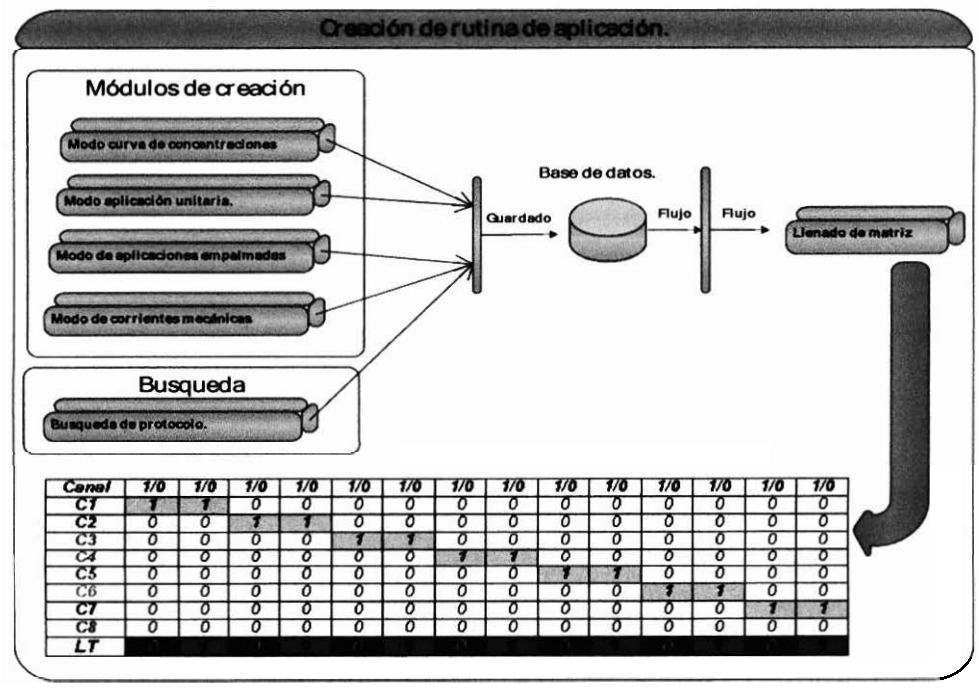


**Figura 30: Flujo para la carga de rutinas de aplicación.**

El código fuente para el llenado de la matriz corresponde al siguiente.

```
//-----LLENADO DE MATRIZ SEGUN VECTORES DEL PROTOCOLO Y DESPLGIEGE DE
RESULTADOS-------------
//-- select Canal,T_inicio,T_Final
//--from Vectores
//--where Id_Protocolo=:IDP;
Query1->Close();
Query1->ParamByName("IDP")->AsFloat=Form1->Id_Protocolo;
Query1->Open();
nres=Query1->RecordCount;
if (nres==0)
Application->MessageBox("Error: Vector no valido","",MB_OK);
else
{
for (i=0;i<nres;i++)
        {
        qvalvula=Query1->FieldByName("Canal")->AsInteger;
        q_inicio=Query1->FieldByName("T_Inicio")->AsInteger;
        q_final=Query1->FieldByName("T_Final")->AsInteger;
        cambio=Query1->FieldByName("Cambio")->AsInteger;
        for(f=q_inicio; f<=q_final; f++)
                {
                matriz[qvalvula-1][f]=1;
                }
        matriz[8][cambio]=1;
        Query1->Next();
        }
}
```

### 3.11.3 Ejecución de rutina y temporización de solenoides.

Una vez cargada la rutina de aplicaciones e iniciada su ejecución, es necesario realizar un ajuste de tiempo en los ciclos en que un switcheo de válvulas se realizará.

Es necesario recordar que los solenoides tienen tiempos de retardo en su respuesta mecánica y por lo general el cierre es más lento que una apertura. Es por ello que delegar su apertura y cierre a un proceso ejecutado en intervalos fijos, puede generar la existencia de aplicaciones simultáneas de soluciones.

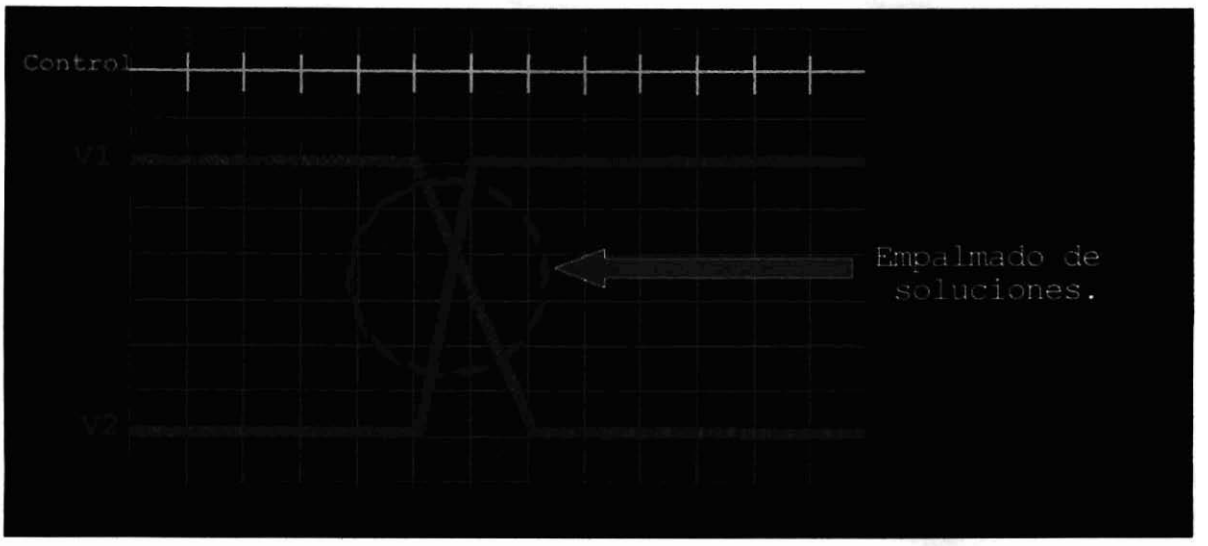


**Figura 31: Retardo de respuesta mecánico del solenoide,** la grafica muestra como el retardo mecánico de los solenoides puede generar la aplicación de múltiples soluciones.

Si bien este error se genera en niveles de milisegundos, dentro del proceso experimental representaría la mezcla de dos soluciones que podrían potenciar o disminuir la respuesta eléctrica de la célula en estudio. Para reducir al máximo este problema fue diseñado un procedimiento que consta de dos procesos temporizados de manera independiente:

Proceso de ejecución.

Este proceso se ejecuta en intervalos de mil milisegundos. Este proceso realiza la apertura y cierre de los solenoides indicados en la matriz de control, de la misma manera revisa si en el instante de ejecución actual se realizará un switcheo de solenoides, en caso de existir, realiza un llamado a ejecución del proceso de intercepción, de lo contrario su ejecución continua de manera normal.

Proceso de intercepción.

El proceso de intercepción se encuentra temporizado a razón de un segundo menos el tiempo de retardo de apertura y cierre de los solenoides. La función de este proceso es la de enviar un cierre de puerto prematuro al solenoide que en un instante determinado realizará un switcheo, de esta manera es posible realizar el cierre del solenoide en el momento necesario para compensar el retardo del solenoide y asegurar que su tiempo efectivo de apertura sea de un segundo.

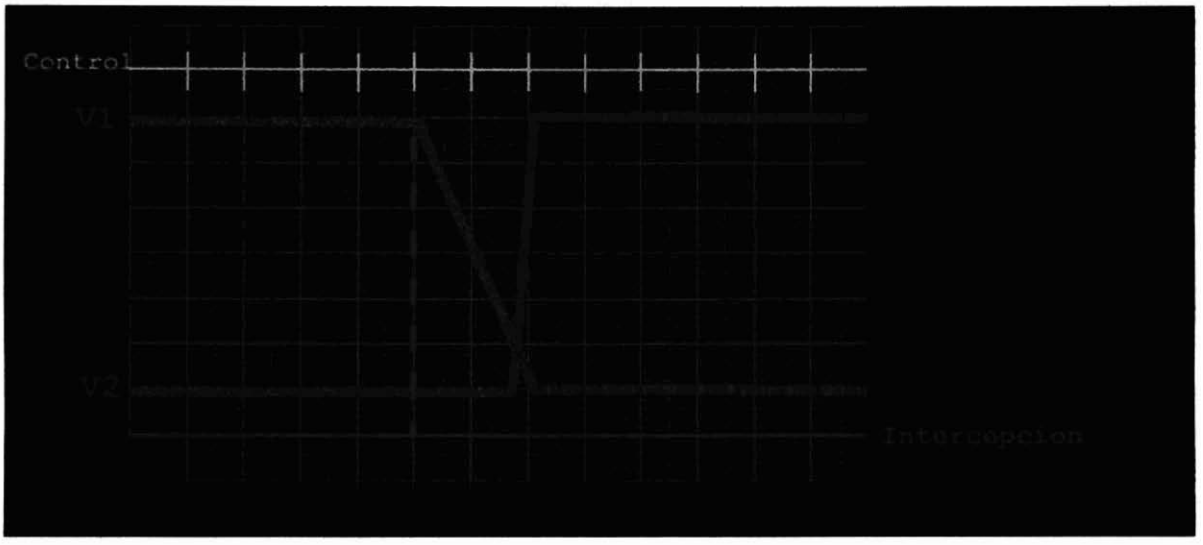


**Figura 32: Ajuste en software del retardo mecánico,** la grafica muestra como el proceso de intercepción realiza el cierre de la válvula el tiempo previo necesario para que no exista empalamiento de soluciones.

Este procedimiento también fue ideado con el objetivo de permitir al sistema modificar su configuración, permitiendo así al investigador jugar con los parámetros de retardo de apertura y cierre para ajustarse a procesos experimentales con respuestas rápidas o respuestas lentas.

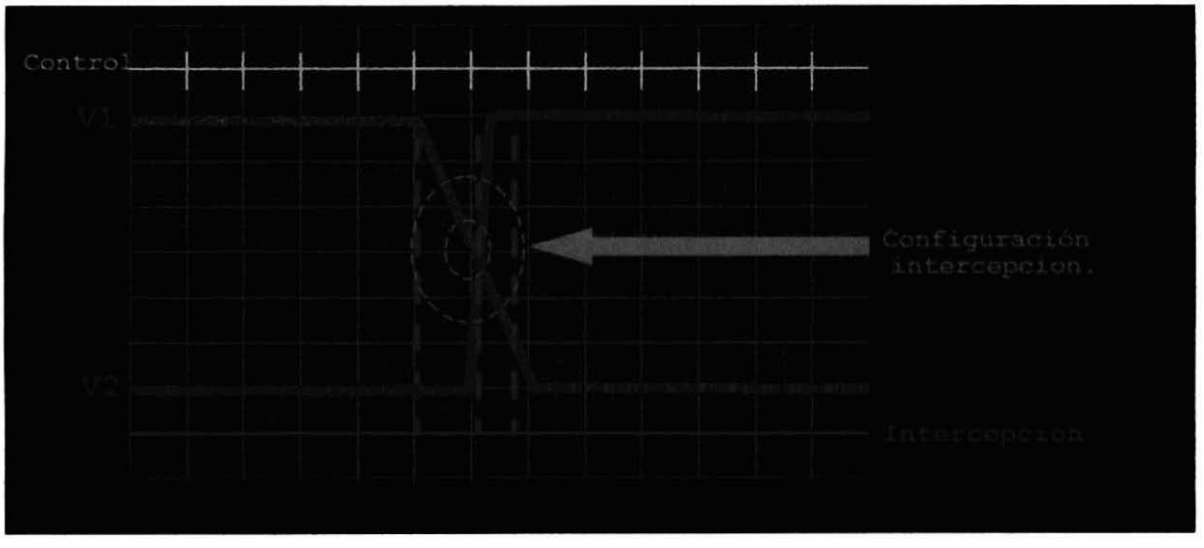


**Figura 33: Posibles configuraciones del sistema,** la grafica permite identificar como la modificación de los retardos mecánicos de los solenoides permiten modificar el comportamiento del sistema.

El procedimiento de compensación así como el Cronómetro total del sistema se implemento mediante cuatro objetos Timer de Cbuilder, estos objetos utilizan las Windows API Timer Functions que permiten ejecutar continuamente un código después de un intervalo de tiempo medido en ms, los Timers utilizados realizan las siguientes funciones:

- Timer 1: Cronómetro general de experimentación
- Timer 2: Cronómetro de control.
- Timer 3: Cronómetro volumen
- Timer 4: Cronómetro de intercepción.

**Timer 1: Cronómetro general de experimentación.**

Realiza un conteo del tiempo total que el ovocito se encuentra en la cámara bajo perfusión, es importante recalcar que el tiempo total de experimentación puede variar del tiempo calculado para el protocolo si existen eventualidades.

```
void __fastcall TForm5::Timer1Timer(TObject *Sender)

t_total++;
}
```

**Timer 2: Cronómetro de control.**

Las funciones de este Timer consisten en realizar el cálculo del valor de control que se envía al puerto y la revisión de la línea de tiempo para activación del proceso de intercepción para un switcheo de válvulas (Figura 34), una vez finalizado el experimento este también realiza la inserción en la base de datos de los tiempos he información del experimento.

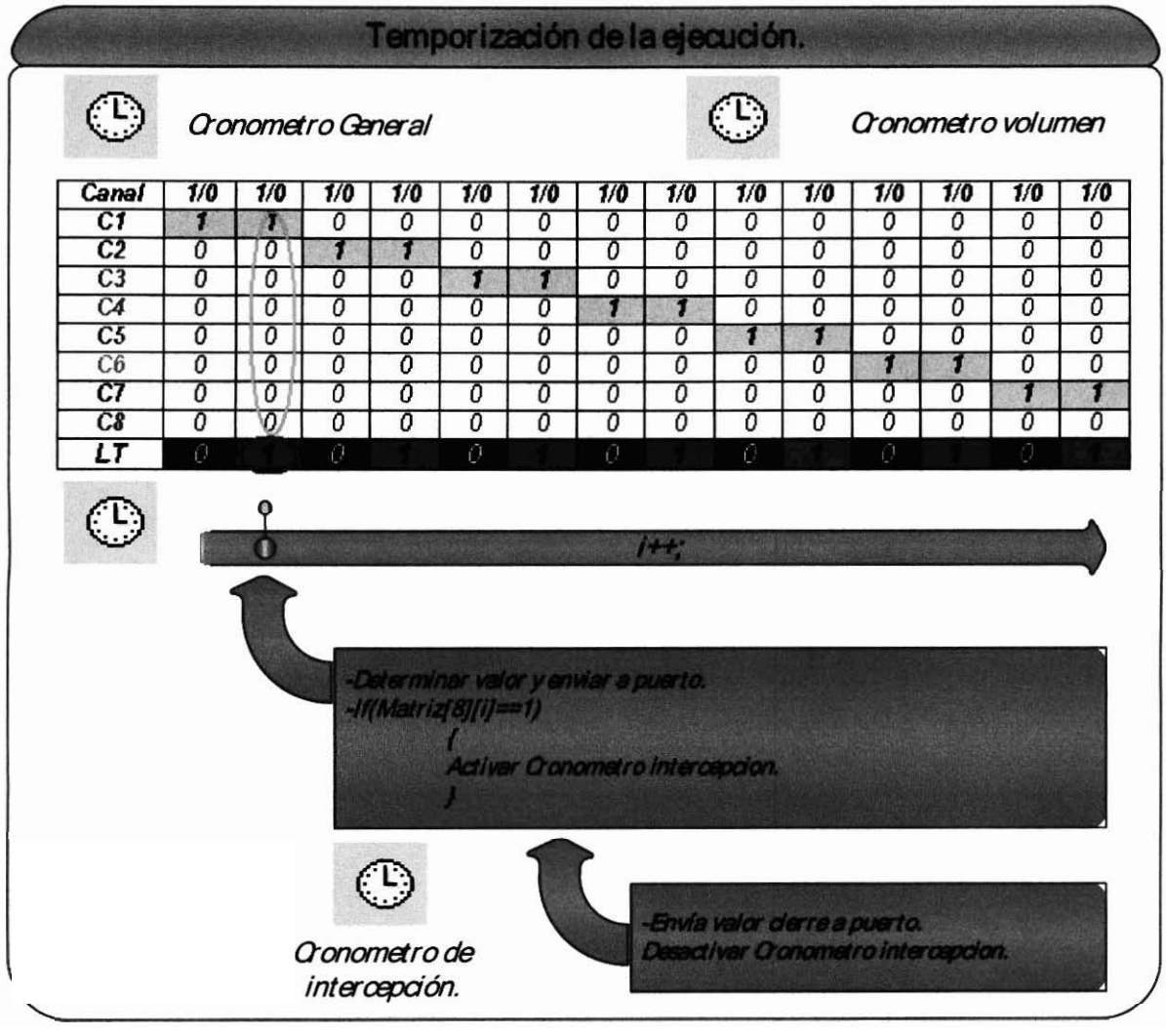


**Figura 34: Ejecución del proceso intercepción.**

```
void __fastcall TForm5::Timer2Timer(TObject *Sender)
{
if(contador<=Form1->T_Parcial)

      salida=0;
      }
if (matriz[0][contador]==1)
      {
            salida=(salida+1);
            }
if (matriz[1][contador]==1)

            salida=salida+2;
```

```
if (matriz[2][contador]==1)
      {
            salida=salida+4;
            }

if (matriz[3][contador]==1)
      {
      salida=salida+8;
      }

if (matriz[4][contador]==1)
            {
            salida=salida+16;
      }

if (matriz[5][contador]==1)
      {
            salida=salida+32;
             }

if (matriz[6][contador]==1)
      {
            salida=salida+32;
       }

if (matriz[7][contador]==1)
      {
      if(Form1->Vol_8<pcritico)
             {
            salida=salida+128;
      }

k=salida;
contador++;
CGauge1->AddProgress(1);
}

else
{
//TEMINA LA EJECUCION
Timer2->Enabled=false;
Timer1->Enabled=false;
Timer3->Enabled=false;
k=0x00;
(oup32)(j,k);
}
(oup32)(j,k);
if (matriz[8][contador]==1)
{
Timer4->Enabled=true;
}
}
```

**Timer 3: Cronómetro volumen**

Este Timer realiza el cálculo de desplazamiento de volúmenes para el canal de lavado, básicamente realiza una disminución del volumen perfundido por segundo al volumen inicial de la solución. También realiza una sumatoria por segundo del volumen que fue enviado a la recamara de experimentación.

```
Vol_c=(Vol_v-disminucion);
V_c=(V_c)+(disminucion);
```

Timer 4: Cronómetro de temporización.

Este Timer realiza el cierre de válvulas en un lapso de tiempo controlado por los retardos físicos del solenoide.

```
Timer4->Interval=1000-(Retardo_Cierre-Retardo_Aoertura);
void __fastcall TForm5::Timer4Timer(TObject *Sender)
{
k=0x00;
(oup32)(j,k);
Timer4->Enabled=false;
}
```

### 3.11.4 Finalización de la aplicación e inserción a base de datos.

Terminada la ejecución de las aplicaciones, es necesario almacenar en la base de datos los resultados experimentales como volúmenes perfundidos, tiempo de experimentación, etc. De igual manera es necesario crear las tablas correspondientes a las corrientes resultantes para cada aplicación, al igual que la generación de la rutina de aplicación la generación de tablas de resultados se ve modificada según el tipo de rutina de aplicación. Dentro del proceso de generación de tablas es identificado a qué tipo de aplicación corresponde y es adoptado un método de generación específico:

```
//INCERTAMOS A LA TABLA DE RESULTADOS LA INFORMACION
//INSERT INTO
Resultados(Id_protocolo,Tiempo_Total,Vol_PC1,Vol_PC2,Vol_PC3,Vol_PC4,Vol_
PC5,Vol_PC6,Vol_PC7,Vol_PC8,Fecha)
//VALUES (:IDP,:TTO,:VP1,:VP2,:VP3,:VP4,:VP5,:VP6,:VP7,:VP8,:DAT)
Query2->Close();
Query2->ParamByName("IDP")->AsFloat=Form1->Id_Protocolo;
Query2->ParamByName("TTO")->AsFloat=t_total;
Query2->ParamByName("VP1")->AsFloat=V_c1;
Query2->ParamByName("VP2")->AsFloat=V_c2;
Query2->ParamByName("VP3")->AsFloat=V_c3;
Query2->ParamByName("VP4")->AsFloat=V_c4;
```

```
Query2->ParamByName("VP5")->AsFloat=V_c5;
Query2->ParamByName("VP6")->AsFloat=V_c6;
Query2->ParamByName("VP7")->AsFloat=V_c7;
Query2->ParamByName("VP8")->AsFloat=V_c8;
Query2->ParamByName("DAT")->AsDate=Date();
Table1->Active = true;
Query2->ExecSQL();
Table1->ApplyUpdates();
Table1->CommitUpdates();
Table1->Refresh();
//-------RECUPERAMOS EL ID DEL EL RESULTADO PARA DESPLEGAR TABLA Y ESE
PEDO
//Select MAX(Id_Resultado) As Maximo
//from Resultados
Query3->Close();
Query3->Open();
nres=Query3->RecordCount;
if (nres==0)
        Application->MessageBox("Error: Vector no valido","",MB_OK);
else
        {
        id_resultado=Query3->FieldByName("Maximo")->AsInteger;
        Label16->Caption=id_resultado;
        }

//----RECUPERADO EL RESULTADO PROCEDEMOS A COPIAR VECTORES
REPRESENTATIVOS SEGUN SEA EL TIPO DE PROTOCOLO
if(Form1->Tipo_Protocolo=="CURVA DE CONCENTRACIONES")
       {
        //Select canal,duracion
        // from Vectores
        //Where Id_Protocolo=:IDP and Id_Vector=!IDV
        //RECUPERAMOS datos del vector.
        Query4->Close();
        Query4->ParamByName("IDV")->AsFloat=Form3->C_Lavado;
        Query4->ParamByName("IDP")->AsFloat=Form1->Id_Protocolo;
        Query4->Open();
        nres=Query4->RecordCount;
        if (nres==0)
                Application->MessageBox("Error: Vector no
valido","",MB_OK);
        else
                {
                int canal,duracion;
                double volumen;
                AnsiString contenido;
                Application->MessageBox("OK: SI SE ENCONTRO LA
QUERY","",MB_OK);
                Label17->Caption=nres;
                Label18->Caption=canal;
                Label19->Caption=duracion;
                Label20->Caption=volumen;
                for (int i=0;i<nres;i++)
                        {
                        canal=Query4->FieldByName("Canal")->AsInteger;
                        duracion=Query4->FieldByName("Duracion")-
>AsInteger;
```

```
                        volumen=disminucion*Form1->velocidad;
                        if (canal==1)
                                {
                                contenido=Form1->Contenido1;
                                }
                        if (canal==2)
                                {
                                contenido=Form1->Contenido2;
                                }
                        if (canal==3)
                                {
                                contenido=Form1->Contenido3;
                                }
                        if (canal==4)
                                {
                                contenido=Form1->Contenido4;
                                }
                        if (canal==5)
                                {
                                contenido=Form1->Contenido5;
                                }
                        if (canal==6)
                                {
                                contenido=Form1->Contenido6;
                                }
                        if (canal==7)
                                {
                                contenido=Form1->Contenido7;
                                }
                        if (canal==8)
                                {
                                contenido=Form1->Contenido8;
                                }
                        //HE INCERTAMOS EN LA BASE DE DATOS
                        //Insert into
Corrientes(Id_Resultado,Canal,Contenido,Duracion,Volumen)
                        //VALUES(:IDR,:CNL,:CON,:DRC,:VOL)
                        //------------PRIMER VECTOR LAVADO--////
                        Query5->Close();
                        Query5->ParamByName("IDR")->AsFloat=
id_resultado;
                        Query5->ParamByName("CNL")->AsFloat=canal;
                        Query5->ParamByName("CON")->AsString=contenido;
                        Query5->ParamByName("DRC")->AsFloat=duracion;
                        Query5->ParamByName("VOL")->AsFloat=volumen;
                        Table2->Active = true;
                        Query5->ExecSQL();
                        Table2->ApplyUpdates();
                        Table2->CommitUpdates();
                        Table2->Refresh();
                        Query4->Next();
                        }
                }
        }

        //---SI NO ES ESE AGREGAMOS TODOS LOS VECTORES
        else
```

```
        {
         //Select canal,duracion
         // from Vectores
         //Where Id_Protocolo=:IDP
         //RECUPERAMOS datos del vector.
         Query7->Close();
         Query7->ParamByName("IDP")->AsFloat=Form1->Id_Protocolo;
         Query7->Open();
         nres=Query7->RecordCount;
         if (nres==0)
                 Application->MessageBox("Error: Vector no
valido","",MB_OK);
         else
                 {
                 int canal,duracion;
                 double volumen;
                 AnsiString contenido;
                 Application->MessageBox("OK: SI SE ENCONTRO LA
QUERY","",MB_OK);
                 Label17->Caption=nres;
                 Label18->Caption=canal;
                 Label19->Caption=duracion;
                 Label20->Caption=volumen;
                 for (int i=0;i<nres;i++)
                         {
                         canal=Query7->FieldByName("Canal")->AsInteger;
                         duracion=Query7->FieldByName("Duracion")-
>AsInteger;

                         volumen=disminucion*Form1->velocidad;
                         if (canal==1)
                                 {
                                 contenido=Form1->Contenido1;
                                 }
                         if (canal==2)
                                 {
                                 contenido=Form1->Contenido2;
                                 }
                         if (canal==3)
                                 {
                                 contenido=Form1->Contenido3;
                                 }
                         if (canal==4)
                                 {
                                 contenido=Form1->Contenido4;
                                 }
                         if (canal==5)
                                 {
                                 contenido=Form1->Contenido5;
                                 }
                         if (canal==6)
                                 {
                                 contenido=Form1->Contenido6;
                                 }
                         if (canal==7)
                                 {
                                 contenido=Form1->Contenido7;
                                 }
```

```
                        if (canal==8)
                                {
                                contenido=Form1->Contenido8;
                                }
                        //HE INCERTAMOS EN LA BASE DE DATOS
                        //Insert into
Corrientes(Id_Resultado,Canal,Contenido,Duracion,Volumen)
                        //VALUES(:IDR,:CNL,:CON,:DRC,:VOL)
                        //------------PRIMER VECTOR LAVADO--////
                        Query5->Close();
                        Query5->ParamByName("IDR")->AsFloat=
id_resultado;
                        Query5->ParamByName("CNL")->AsFloat=canal;
                        Query5->ParamByName("CON")->AsString=contenido;
                        Query5->ParamByName("DRC")->AsFloat=duracion;
                        Query5->ParamByName("VOL")->AsFloat=volumen;
                        Table2->Active = true;
                        Query5->ExecSQL();
                        Table2->ApplyUpdates();
                        Table2->CommitUpdates();
                        Table2->Refresh();
                        Query7->Next();
                        }
                }
        }
```

### 3.11.5 Carga y prueba de parámetros de retardo.

Diseñado el procedimiento de ajuste de tiempo en el switcheo de válvulas, fue necesario crear un modulo que permitiera realizar pruebas repetidas de la modificación de estos parámetros para posteriormente ser guardados y utilizados en la ejecución del sistema.

Para este fin, fue diseñado un procedimiento que solicita de manera visual sean asignados dos canales, un tiempo de aplicación, el momento de corte y los retardos de apertura y cierre de los solenoides. Este procedimiento se incluyó como una herramienta en el modulo de configuraciones. Al igual que el proceso de ejecución, este procedimiento implementa un Timer de ejecución y un Timer de intercepción que realiza el cierre del solenoide en el tiempo requerido.

```
void __fastcall TForm30::Timer1Timer(TObject *Sender)
{
if(contador<=d_prueba)
        {
         if(contador<t_cierre)
         {
```

```
            k=salida_a;
            }
          }
else
{
Timer1->Enabled=false;
contador=0;
Application->MessageBox("Ok: prueba finalizadfa","",MB_OK);
}
(oup32)(j,k);
if(contador==t_cierre)
{
Timer2->Enabled=true;
}
contador++;
}

void __fastcall TForm30::Timer2Timer(TObject *Sender)
{
k=0x00;
(oup32)(j,k);
Timer2->Enabled=false;
}
```

## 3.12 Programación de Hilos y prioridad de ejecución.

Con el fin de otorgar al software certeza en su ejecución temporizada, fueron albergados al interior de un Hilo los procedimientos de temporización y control, la plataforma utilizada CBuilder proporciona ya clases abstractas para estas tareas, estas clases corresponden al TThread.

Para definir un hilo se debe crear una clase derivada *TThread*. Al ser *TThread* una clase abstracta posee métodos puros que funcionan como métodos virtuales, ejemplo Execute(). Como sucede con cualquier otra clase, en una clase derivada de *TThread* se pueden añadir todos los miembros (propiedades y métodos) que se necesiten. Sin embargo para su utilización es necesario implementar:

- Un **constructor** de la clase, que define e inicializa el hilo.
- El **método *Execute,*** *que contiene el código asociado a la tarea que tiene que realizar el hilo.*

La inicialización del hilo se realiza en su constructor, donde se establecen los valores iníciales de cuantas propiedades sean necesarias, antes de invocar al constructor de una clase derivada de *TThread*, se invoca al constructor de su clase base (*TThread*):

```
__fastcall TThread(bool CreateSuspended);
```

Éste recibe como parámetro **false** si se desea que el hilo ejecute su tarea inmediatamente después de ser creada, o **true** si se desea este suspendido temporalmente. Dos características de las hebras conviene establecerlas en el constructor: su prioridad y cuándo debe liberarse la hebra.

La prioridad de un hilo indica qué grado de preferencia tiene cuando el sistema operativo reparte el tiempo de CPU entre los distintos hilos que se ejecutan en el sistema, para indicar la prioridad del objeto hilo hay que fijar el valor de la propiedad ***Priority***:

| Valor | Prioridad |
|---|---|
| *tpIdle* | Se ejecuta cuando el sistema está inactivo. |
| *tpLowest* | Dos puntos por debajo del valor normal. |
| *tpLower* | Un punto por debajo del valor normal. |
| *tpNormal* | El Hilo tiene la prioridad normal. |
| *tpHigher* | Un punto por encima del valor normal. |
| *tpHighest* | Dos puntos por encima del valor normal. |
| *tpTimeCritical* | El hilo tiene la prioridad más alta. |

**Tabla 11: Prioridades de ejecución de los hilos.**

Ejemplo de definición de hilo de ejecución:

Programa .h:

```
public:
  Thilo *Hilo;
  __fastcall THiloprueba(TPaintBox *PaintBox);
  __fastcall ~ THiloprueba(void);
```

Programa.cpp:

```
#include <stdlib.h>

__fastcall THiloprueba::THiloprueba
        (TPaintBox *PaintBox)
        : TThread(false)
{
```

```
  Priority       = tpNormal;
  FreeOnTerminate = true;

  Hilo = new Hilo (   // Objeto Hilo
//CODIGO DE TAREA EN EJECUCION TAREA EN EJECUCION
k=0x00;
(oup32)(j,k);
}

__fastcall THiloprueba::~THiloprueba (void)
{
  delete Hilo;
}
```

# Capitulo 4.Resultados.

## 4.0 Introducción.

En este capítulo se presentara el sistema prototipo, se describirán sus características y se realizara una breve introducción al funcionamiento del programa de control. También serán detallados los resultados experimentales obtenidos con el funcionamiento del sistema. Finalmente serán descritas las conclusiones y propuestas las mejoras futuras para el sistema.

## 4.1 Prototipo del sistema.

El prototipo del sistema se encuentra actualmente construido en MDF, material utilizado para realizar maquetas de piezas mecánicas, el sistema cuenta con 8 canales de perfusión regulados por válvulas de solenoide, estas válvulas se encuentran alojadas en una caja diseñada para incluirlas junto a sus circuitos de control y reducir así la longitud del cableado de alta impedancia, la base del sistema fue implementada mediante un electroimán que permite alojar el sistema a una mesa de trabajo y retirarlo con facilidad. Los solenoides, así como el resto del sistema se alimentan de una fuente de 120/220V externa, la parte frontal del sistema identifica mediante etiquetas de colores el contenedor y la válvula de control para cada canal, así como dos leds rojo y verde que indican la activación o cierre de cada una de estas válvulas.

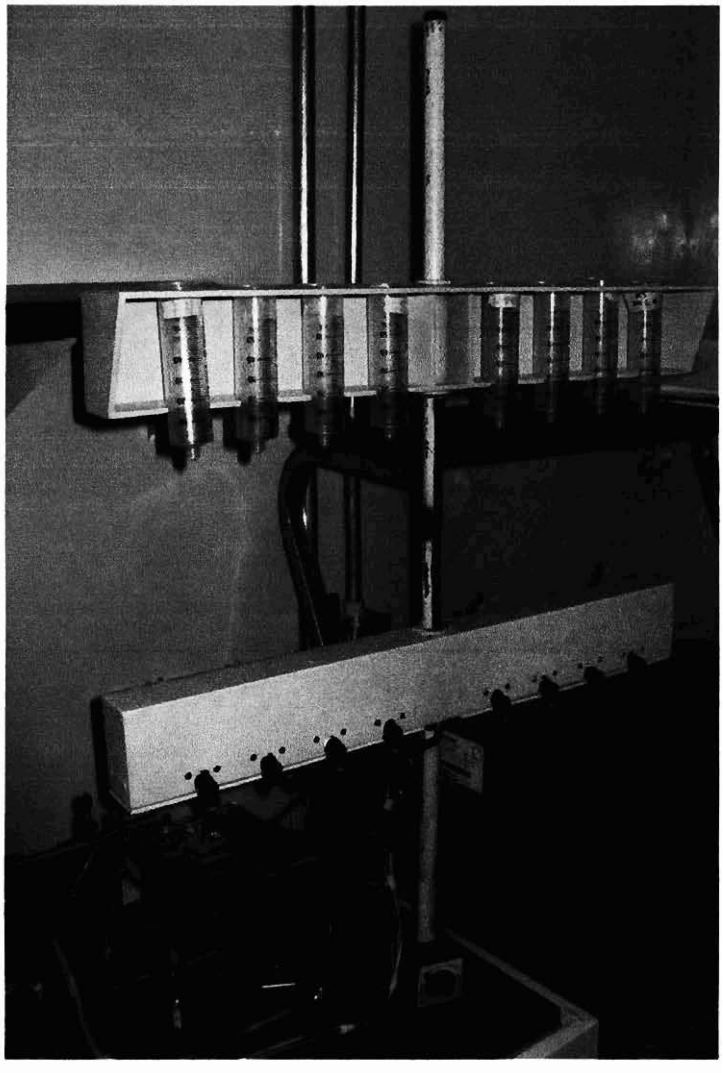

**Figura 32: Fotografía sistema prototipo.**

## 4.2 Software de control.

El software desarrollado se llamado "Cronos", esta aplicación concentra en una ventana principal las diferentes funciones del sistema permitiendo de manera grafica iniciar el diseño de un protocolo de experimentación.

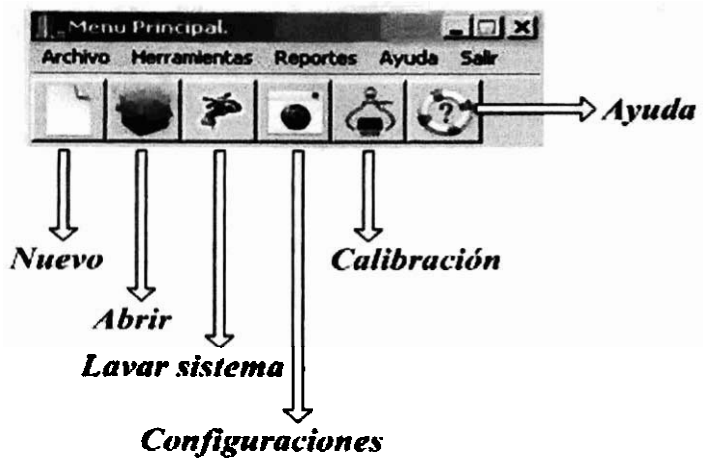


**Figura 33: Pantalla principal del sistema.**

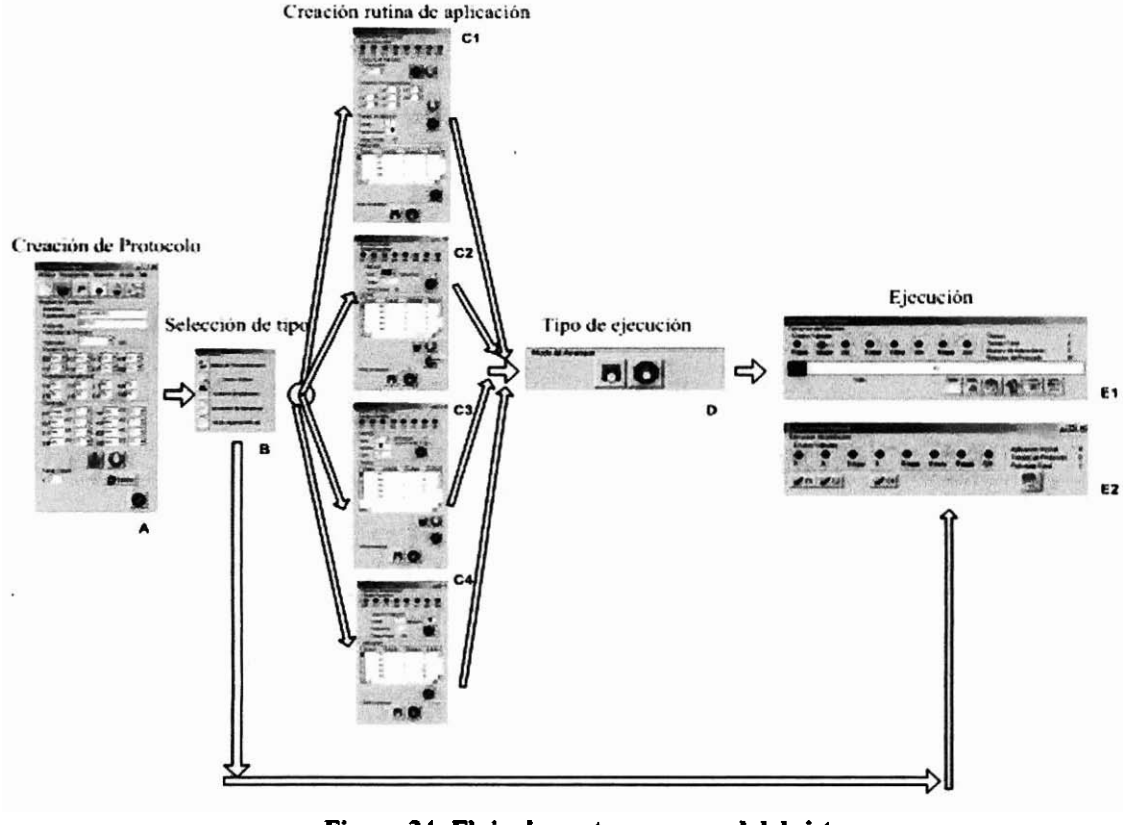


**Figura 34: Flujo de ventanas general del sistema.**

El flujo general de operación consta de cuatro pasos que el usuario debe finalizar antes de iniciar la ejecución de un protocolo, el primer paso consiste en la creación de un protocolo Forma(A) en la cual el usuario ingresará la información general del protocolo, como los canales válidos, contenidos, concentraciones e información general del experimento, los datos ingresados en esta etapa corresponden a:

- Nombre de usuario
- Nombre de protocolo
- Velocidad de Perfusión
- Validación de canal y descripción de contenido
- Establecimiento de canal de lavado

Ingresada la información general del protocolo de experimentación se procede a seleccionar un modo de diseño para la creación de la rutina de aplicación de fármacos "forma (B)", entre los modos de diseño se encuentra:

- Modo Curva de Concentraciones
- Modo Vector Unitario
- Modo Vectores Empalmados
- Modo Generador de Respuestas Mecánicas
- Modo de Apertura Mecánica.

Modo Curva de Concentraciones

Esta modalidad solicita al usuario sea definido un canal de lavado, una vez establecido el canal de lavado es solicitado al usuario defina una secuencia lógica de aplicación 1-n para el resto de los canales, establecida la secuencia es solicitada la duración del lavado así como la duración de las aplicaciones, una vez establecido estos tiempos el programa genera de manera automática el diseño de la secuencia de aplicaciones necesarias para completar el experimento.

Modo Vector Unitario

Esta modalidad solicita al usuario especifique el canal y la duración para cada aplicación que desee agregar a su diseño, en el caso especifico de esta modalidad, se cuenta con herramientas que permiten eliminar o modificar una aplicación especifica alterando el resto de las aplicaciones de manera automática si es necesario.

Modo Vectores Empalmados

Esta modalidad solicita al usuario especifique el canal, el instante de inicio y el instante de fin para cada aplicación que desee agregar a su diseño, de esta manera diferentes vectores pueden ser agregados con tiempos de inicio y fin variados, en el caso de esta modalidad también se cuenta con herramientas de modificación y eliminación de aplicaciones pero estas no alteran el resto del diseño.

Modo Generador de Respuestas Mecánicas

Esta modalidad solicita al usuario un tiempo de lavado, un tiempo de paro y un número de repeticiones que desea ejecutar, establecidos estos datos el programa procede a crear el diseño de las aplicaciones de manera automática.

Modo de Apertura Manual.

Este tipo de ejecución permite al usuario mediante las teclas 1-8 solicitar al sistema realice una apertura de un canal especifico cerrando el resto. Ya que este tipo de ejecución no pose control alguno, ni requiere planificación, el usuario determina el momento en que desea terminar con el experimento.

Ejecución de aplicaciones.

Diseñadas las aplicaciones el programa solicita sea definida una forma de iniciar su ejecución. La modalidad manual inicia el sistema en el momento que el usuario indique un clic sobre el arranque del sistema, la modalidad sincronizada el programa inicia el lavado del ovocito e inicia las aplicaciones al primer clic o tecla oprimida en la PC.

Ejecución manual

Este tipo de ejecución permite al usuario inicia el sistema con un lavado constante, una vez que el experimentador considera que las condiciones para el inicio del protocolo están dadas indicara el arranque de las aplicaciones. Iniciado este procedimiento el usuario puede realizar también las siguientes funciones

- Abortar el protocolo sin vaciar la cámara de experimentación
- Parar el sistema vaciando por completo la cámara de experimentación
- Reiniciar por completo el protocolo.
- Moverse a un punto especifico del protocolo
- Visualizar el protocolo.
- Visualizar el cálculo de desplazamiento.

## 4.3 Conclusiones

Como conclusión de este trabajo es posible mencionar que se logro la construcción de un sistema automatizado de perfusión, el cual implementa en software diversas funcionalidades que agilizan la labor del investigador en la planeación, ejecución, almacenamiento y resolución de contingencias.

Mediante la inclusión de parámetros fue posible la implementación de un control dinámico de retardos en los solenoides, lo que permite al investigador modificar a voluntad el tiempo y volumen muerto que el switcheo de solenoides implica. El diseño físico y eléctrico del sistema basado en circuitos de bajo consumo de potencia y material inocuo permitieron la eliminación de alteraciones eléctricas o de carácter mecánico en los resultados experimentales.

Las características del sistema desarrollado permiten al laboratorio contar con la certidumbre que el proceso de perfusión es reproducible con exactitud. De igual manera la automatización de este proceso permite incrementar la productividad en el tiempo de trabajo del investigador liberándolo del trabajo físico y permitiéndole enfocarse en el análisis de datos.

A la par de los beneficios funcionales que el desarrollo de este proyecto acarrean, es posible mencionar que los gastos realizados en el desarrollo corresponden a una pequeña parte del costo de un equipo comercial, por lo que es posible asegurar que la realización de este y

otros proyectos de la misma índole tiene asegurada su rentabilidad para el instituto, para la versión prototipo de este sistema los gastos realizados corresponden a:

| Concepto: | Costo |
|---|---|
| 8 Válvulas de solenoide | 120 DLL. |
| Acoples de canalización. | 128 DLL |
| Materiales y construcción: | 230 DLL |
| Partes electrónicas. | 100 DLL |
| Total: | 578 DLL |

## 4.4 Trabajo a Futuro.

E objetivo general en la realización de este proyecto fue alcanzado con la construcción de un sistema que automatiza el proceso de perfusión celular, sin embargo durante el desarrollo del proyecto diversas modificaciones en su diseño y construcción fueron realizadas con el fin de ajustar su funcionamiento. Algunas de estas medidas adoptadas pueden ser mejoradas por lo que en los siguientes aparatados se describirán los aspectos que son susceptibles de mejora.

**Implementación de sensores volumétricos.**

Si bien fue implementada una funcionalidad en software que permite realizar un cálculo de la velocidad de vaciado de los contenedores, es obvio que este parámetro dependerá del usuario, por lo que el sistema no asegura la valides del parámetro. Esta problemática podría ser resuelta integrando sensores volumétricos acoplables a los contenedores. Estos sensores ofrecerían una lectura real del volumen disponible y la velocidad de vaciado, estas lecturas podrían ser adquiridas por el software mediante la implementación de un convertidor ADC.

**Presurización del sistema.**

Una de las problemáticas detectadas en el funcionamiento del sistema es la aparición de burbujas en el flujo de fármacos, problemática que es generada por la existencia de conexiones defectuosas y por la entrada de aire a los contenedores, si bien es posible reducir al máximo este problema con los materiales adecuados, la presurización del sistema permitiría asegurar en un 100% un flujo constante y libre entradas de aire. Además otros beneficios de esta implementación seria la ampliación del rango de velocidad de perfusión.

**Rediseño físico del sistema.**

El prototipo del sistema construido puede ser mejorado, si bien el diseño toma características funcionales de los sistemas comerciales, la implementación de mecanismos como broches, seguros e indicadores le otorgaría al sistema una utilidad de producto terminado y no la de equipo prototipo.

**Construcción física del sistema.**

Finalmente el último punto susceptible de mejora es la finalización de la construcción física del sistema con las mejoras antes señaladas, si bien el modelo prototipo con el que contamos es funcional en un 100% es deseable implementarlo en los materiales señalados, específicamente la caja que contiene los solenoides debe ser construida en aluminio así como el soporte del sistema construido en el mismo material.

**Apéndices.**

# APÉNDICE A. PLANOS DE CONSTRUCCIÓN DEL SISTEMA.

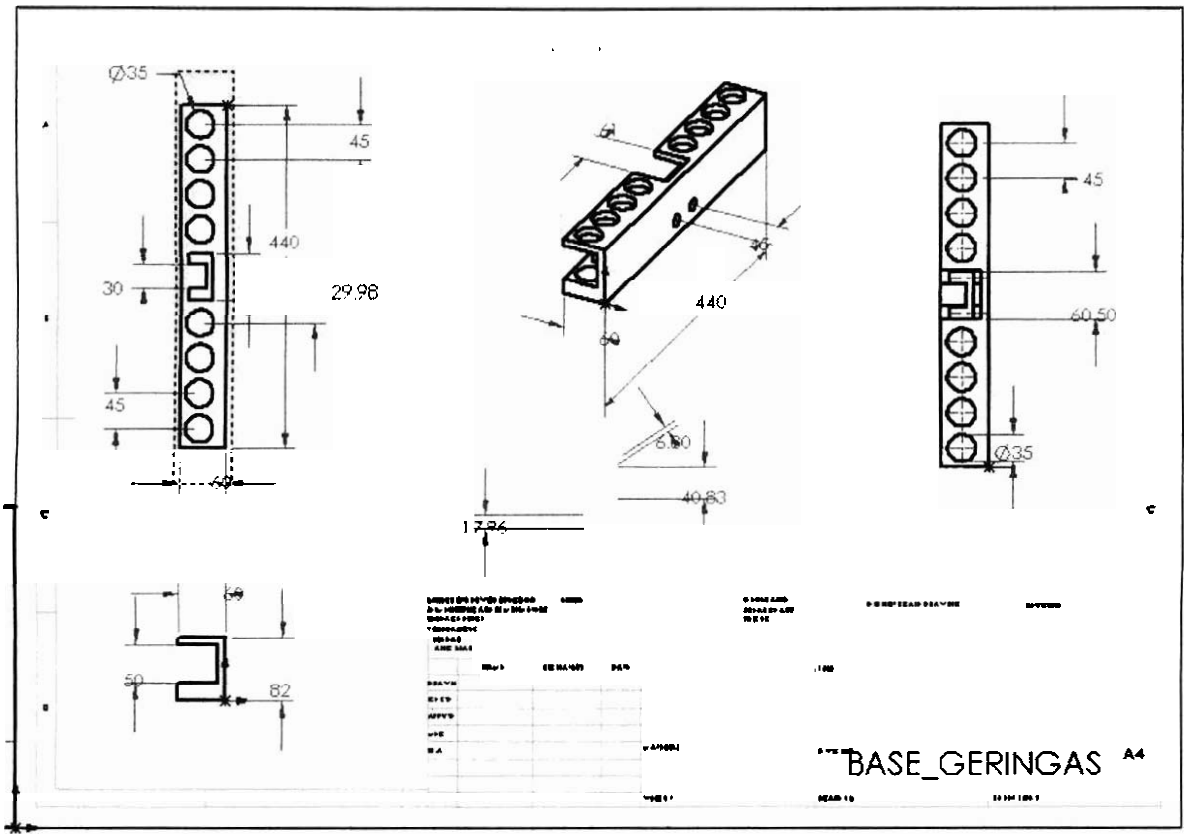


Figura A.1: Plano para el soporte de contenedores.

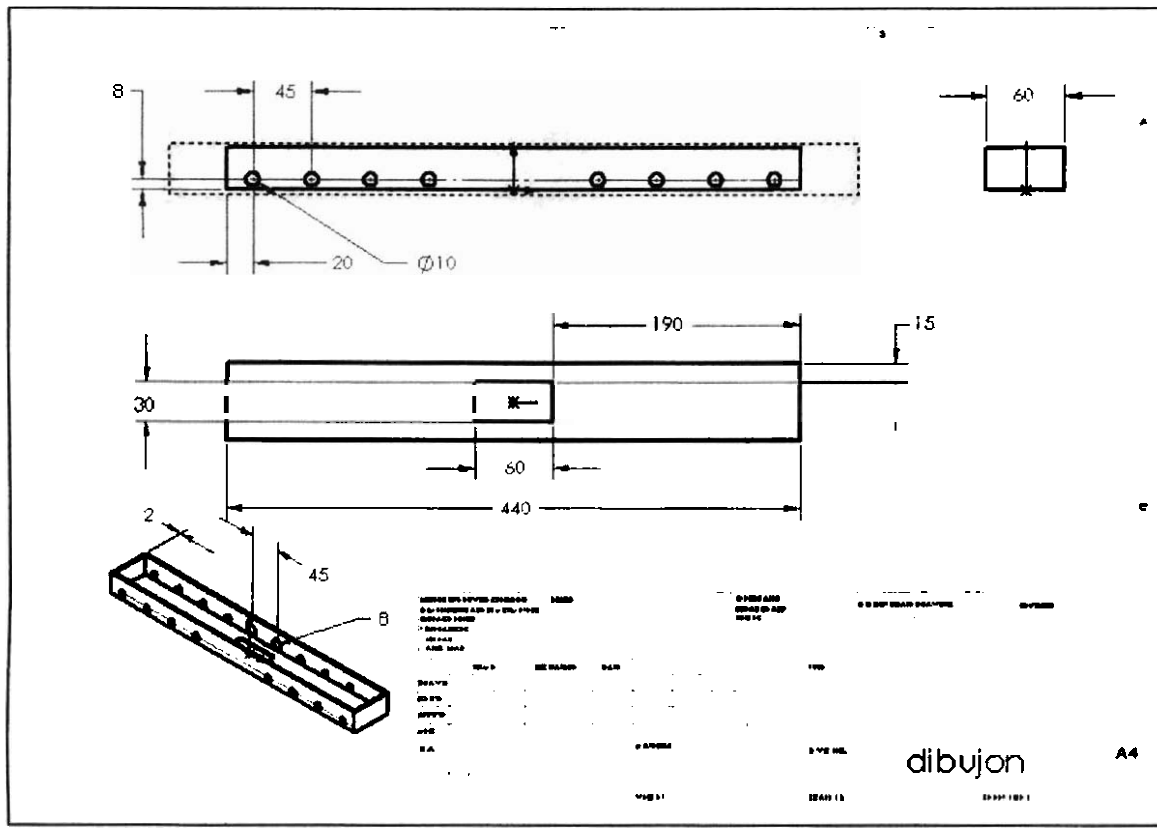


Figura A.2: Plano para el soporte de válvulas.

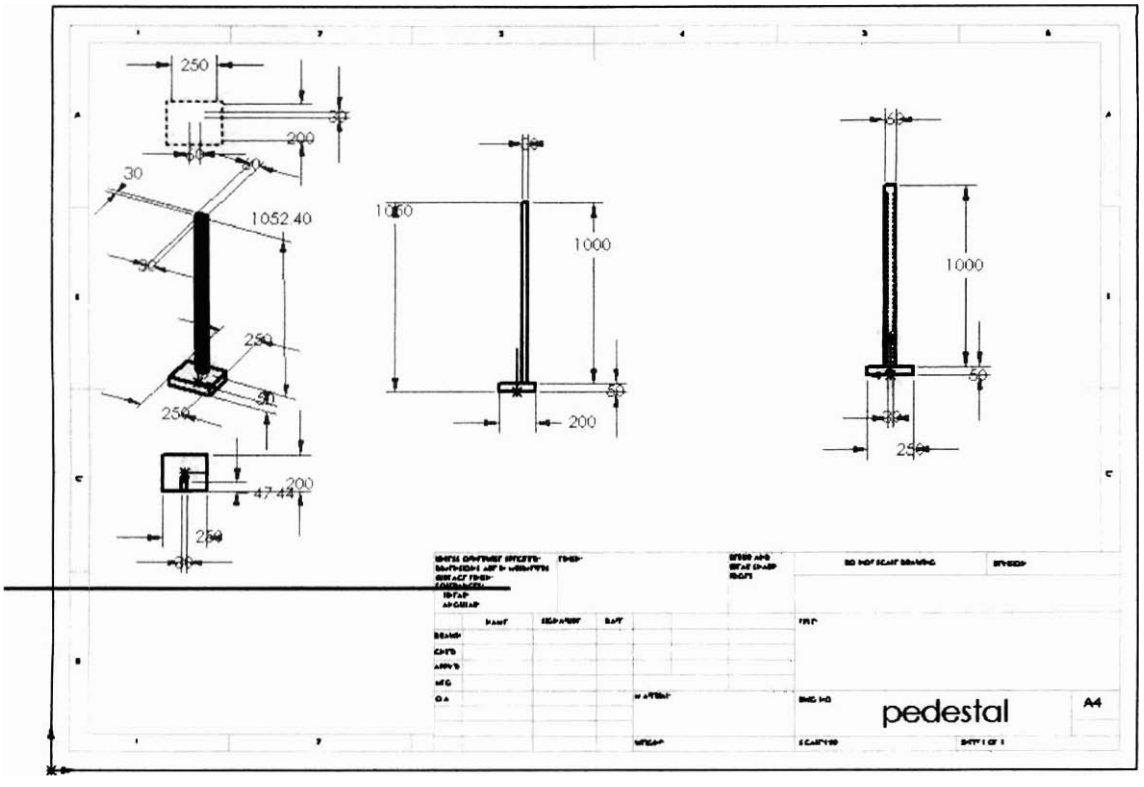


Figura A.3: Plano para el pedestal del sistema.

## Apéndice B.

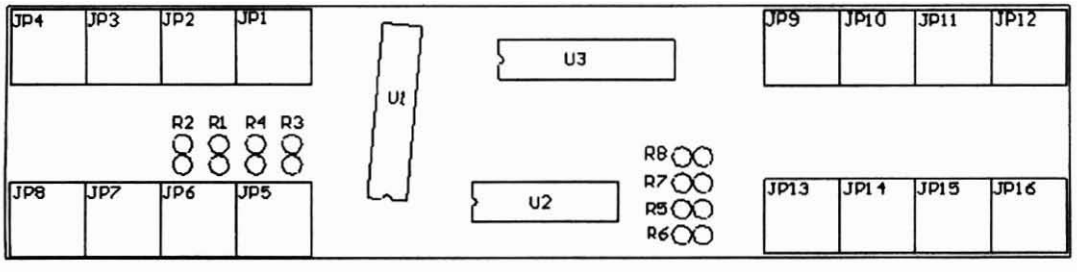


Figura B.1: Plano general de la placa de control.

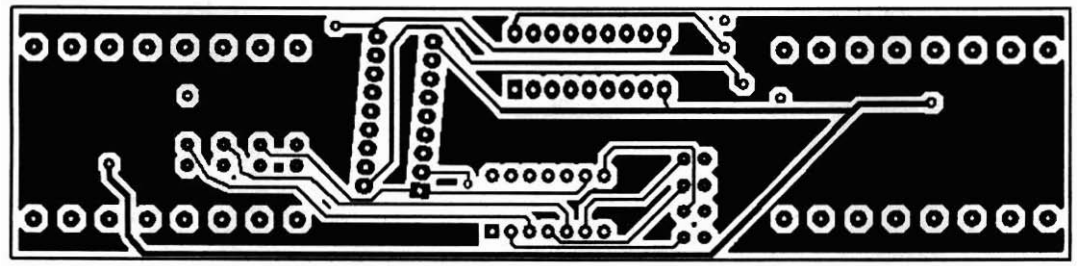

Figura B.2: Cara superior de la placa de control.

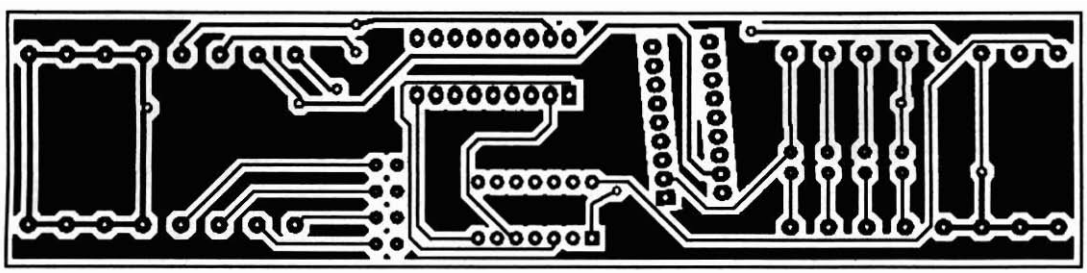

Figura B.3: Cara inferior de la placa de control.

## Apéndice C.

Tabla: Protocolos.

| Dato. | Tipo de dato | Función | Intervalo | Descripción. |
|---|---|---|---|---|
| Id_Protocolo | Autonumerico | PK | Continuo | Numero identificador de protocolo |
| Id_Configuracion | Entero | FK | Continuo | Identifica la configuración del sistema |
| Nombre_Usuario | Char(75) | Almacenar datos usuario. | Texto | Identifica por nombre de usuario un protocolo |
| Nombre _ protocolo | Char(25) | Almacenar datos del protocolo | Texto | Identifica por nombre un protocolo |
| Fecha | Date | Almacenar datos del protocolo | Continuo | Identifica la fecha de creación del protocolo |
| Velocidad | Entero | Almacenar datos del protocolo | Continuo | Identifica la velocidad del flujo de fármacos. |
| C_1 | Binario | Almacenar datos de canales. | Discreto | Identifica la disponibilidad del canal 1 en protocolo |
| C_2 | Binario | Almacenar datos de canales. | Discreto | Identifica la disponibilidad del canal 2 en protocolo |
| C_3 | Binario | Almacenar datos de canales. | Discreto | Identifica la disponibilidad del canal 3 en protocolo |
| C_4 | Binario | Almacenar datos de canales. | Discreto | Identifica la disponibilidad del canal 4 en protocolo |
| C_5 | Binario | Almacenar datos de canales. | Discreto | Identifica la disponibilidad del canal 5 en protocolo |
| C_6 | Binario | Almacenar datos de canales. | Discreto | Identifica la disponibilidad del canal 6 en protocolo |
| C_7 | Binario | Almacenar datos de canales. | Discreto | Identifica la disponibilidad del canal 7 en protocolo |
| C_8 | Binario | Almacenar datos de canales. | Discreto | Identifica la disponibilidad del canal 8 en protocolo |

| | | | | |
|---|---|---|---|---|
| Vol_1 | Entero | Almacenar datos de fármacos. | Continuo | Identifica el volumen inicial del canal 1 |
| Vol_2 | Entero | Almacenar datos de fármacos. | Continuo | Identifica el volumen inicial del canal 2 |
| Vol_3 | Entero | Almacenar datos de fármacos. | Continuo | Identifica el volumen inicial del canal 3 |
| Vol_4 | Entero | Almacenar datos de fármacos. | Continuo | Identifica el volumen inicial del canal 4 |
| Vol_5 | Entero | Almacenar datos de fármacos. | Continuo | Identifica el volumen inicial del canal 5 |
| Vol_6 | Entero | Almacenar datos de fármacos. | Continuo | Identifica el volumen inicial del canal 6 |
| Vol_7 | Entero | Almacenar datos de fármacos. | Continuo | Identifica el volumen inicial del canal 7 |
| Vol_8 | Entero | Almacenar datos de fármacos. | Continuo | Identifica el volumen inicial del canal 8 |
| Contenido_1 | Char(25) | Almacenar datos de fármacos. | Texto | Identifica contenido del canal 1 |
| Contenido_2 | Char(25) | Almacenar datos de fármacos. | Texto | Identifica contenido del canal 2 |
| Contenido_3 | Char(25) | Almacenar datos de fármacos. | Texto | Identifica contenido del canal 3 |
| Contenido_4 | Char(25) | Almacenar datos de fármacos. | Texto | Identifica contenido del canal 4 |
| Contenido_5 | Char(25) | Almacenar datos de fármacos. | Texto | Identifica contenido del canal 5 |
| Contenido_6 | Char(25) | Almacenar datos de fármacos. | Texto | Identifica contenido del canal 6 |
| Contenido_7 | Char(25) | Almacenar datos de fármacos. | Texto | Identifica contenido del canal 7 |
| Contenido_8 | Char(25) | Almacenar datos de fármacos. | Texto | Identifica contenido del canal 8 |
| Conc1 | Char(2) | Almacenar datos de fármacos. | Texto | Identifica la concentración de la solución del canal 1 |
| Conc2 | Char(2) | Almacenar datos de fármacos. | Texto | Identifica la concentración de la solución del canal 2 |
| Conc3 | Char(2) | Almacenar datos de fármacos. | Texto | Identifica la concentración de la solución del canal 3 |
| Conc4 | Char(2) | Almacenar datos de fármacos. | Texto | Identifica la concentración de la solución del canal 4 |
| Conc5 | Char(2) | Almacenar datos de fármacos. | Texto | Identifica la concentración de la solución del canal 5 |
| Conc6 | Char(2) | Almacenar datos de fármacos. | Texto | Identifica la concentración de la solución del canal 6 |
| Conc7 | Char(2) | Almacenar datos de fármacos. | Texto | Identifica la concentración de la solución del canal 7 |
| Conc8 | Char(2) | Almacenar datos de fármacos. | Texto | Identifica la concentración de la solución del canal 8 |
| Canal_Lavado | Entero | Almacenar datos del protocolo | Continuo | Identifica un canal como canal de lavado |
| Tipo_Protocolo | Char(25) | Almacenar datos del protocolo | Discreto | Identifica la modalidad en que fueron planificadas las aplicaciones |
| Total_Aplicaciones | Entero | Almacenar datos del protocolo | Continuo | Identifica el numero de aplicaciones realizadas en el protocolo |
| Tiempo_parcial. | Entero | Almacenar datos del protocolo | Continuo | Identifica el tiempo requerido para la ejecución del protocolo |

Tabla Vectores:

| Dato. | Tipo de dato | Función | Intervalo | Descripción. |
|---|---|---|---|---|
| Id_vector | Autonumerico | PK | Continuo | Numero identificador de vector |
| Id_Protocolo | Entero | FK | Continuo | Identifica a que protocolo pertenece la aplicación |
| Canal | Entero | Almacenar datos de la aplicación | Continuo | Identifica el canal de aplicación |
| T_inicio | Entero | Almacenar datos de la | Continuo | Identifica el momento de apertura de |

| | | | | |
|---|---|---|---|---|
| | | aplicación | | válvula |
| T_Final | Entero | Almacenar datos de la aplicación | Continuo | Identifica el momento de cierre de válvula |
| Duración | Entero | Almacenar datos de la aplicación | Continuo | Identifica el tiempo de aplicación |
| Cambio | Entero | Almacenar datos de la aplicación | Continuo | Identifica en momento en que es iniciada esta aplicación |

Tabla Configuraciones:

| Dato. | Tipo de dato | Función | Intervalo | Descripción. |
|---|---|---|---|---|
| Id_Configuracion | Autonumerico | Pk | Continuo | Numero identificador de configuración |
| Nombre_Config | Char(25) | Almacenar datos de las configuraciones | Texto | Identificar por nombre una configuración |
| Altura_Geringas | Entero | Almacenar datos de las configuraciones | Continuo | Identificar la altura del sistema de contenedores |
| Altura_Valvulas | Entero | Almacenar datos de las configuraciones | Continuo | Identificar la altura del sistema de válvulas |
| Diametro_Manguera | Entero | Almacenar datos de las configuraciones | Continuo | Identifica el diámetro interior de la manguera utilizada |
| Succión | Entero | Almacenar datos de las configuraciones | Continuo | Identifica la succión aplicada a la cámara |
| Vol_Camara | Entero | Almacenar datos de las configuraciones | Continuo | Identifica el volumen de la cámara de experimentación |
| Retardo_Apertura | Entero | Almacenar datos de las configuraciones | Continuo | Identifica el retardo de apertura de los solenoides |
| Retardo_Cierre | Entero | Almacenar datos de las configuraciones | Continuo | Identifica el retardo de cierre de los solenoides |
| Vol_Geringas | Entero | Almacenar datos de las configuraciones | Continuo | Identifica la capacidad de los contenedores |
| Nivel_Critico | Entero | Almacenar datos de las configuraciones | Continuo | Identifica un nivel critico de vaciado |

Tabla Resultados:

| Dato. | Tipo de dato | Función | Intervalo | Descripción. |
|---|---|---|---|---|
| Id_Resultado | Autonumerico | Pk | Continuo | Numero identificador de resultado |
| Id_Protocolo | Entero | Almacenar resultados del experimento | Continuo | Identifica el protocolo al que pertenece el resultado |
| Tiempo_Total | Entero | Almacenar resultados del experimento | Continuo | Identifica el tiempo total que duro el experimento |
| Vol_PC1 | Entero | Almacenar resultados del experimento | Continuo | Identifica el volumen total que prefundió el canal 1 |
| Vol_PC2 | Entero | Almacenar resultados del experimento | Continuo | Identifica el volumen total que prefundió el canal 2 |
| Vol_PC3 | Entero | Almacenar resultados del experimento | Continuo | Identifica el volumen total que prefundió el canal 3 |
| Vol_PC4 | Entero | Almacenar resultados del experimento | Continuo | Identifica el volumen total que prefundió el canal 4 |
| Vol_PC5 | Entero | Almacenar resultados del experimento | Continuo | Identifica el volumen total que prefundió el canal 5 |
| Vol_PC6 | Entero | Almacenar resultados del experimento | Continuo | Identifica el volumen total que prefundió el canal 6 |
| Vol_PC7 | Entero | Almacenar resultados del experimento | Continuo | Identifica el volumen total que prefundió el canal 7 |
| Vol_PC8 | Entero | Almacenar resultados del experimento | Continuo | Identifica el volumen total que prefundió el canal 8 |

Tabla Corrientes:

| Dato. | Tipo de dato | Función | Intervalo | Descripción. |
|---|---|---|---|---|
| Id_Corriente | Autonumerico | PK | Continuo | Numero identificador de corriente |
| Id_Resultado | Entero | Almacenar las respuestas a aplicaciones | Continuo | Identifica el resultado al que pertenece la corriente |
| Canal | Entero | Almacenar las | Continuo | Identifica el canal que aplico el |

| | | | | |
|---|---|---|---|---|
| | | respuestas a aplicaciones | | fármaco |
| Contenido | Char(25) | Almacenar las respuestas a aplicaciones | Texto | Identifica la solución prefundida |
| Duración | Entero | Almacenar las respuestas a aplicaciones | Continuo | Identifica la duración de la aplicación |
| Volumen | Entero | Almacenar las respuestas a aplicaciones | Continuo | Identifica el volumen aplicado, |
| Corriente | Entero | Almacenar las respuestas a aplicaciones | Continuo | Almacena la respuesta a la aplicación |

# 1 Bibliografía


AutoMate Scientific Inc. «AutoMate Scientific.» 2007. http://www.autom8.com/main_products.html (último acceso: 16 de Mayo de 2006).

Axelson, Jan. *Parallel Port Complete: Programming, Interfacing & Using the PC'S Parallel Pinter Port.* Lakeview Research, 1997.

Axon Instruments Inc. *pClamp 6 User´s Guide to Fetchan and pSTAT.* 1999.

Bioscience Tools Inc. *Bioscience Tools.* 2007. http://www.biosciencetools.com/catalog/perfusionMINI.htm (último acceso: 26 de Mayo de 2006).

Cedric M. Smith, Alan M,Reynard. *Essentials of Pharmacology.* New York: W.B. Saunders Company, 1995.Cseri, James Beu. *Windows XP Embedded Step by Step.* Annabooks/Rtc Books, 2003.

D´Argenio, David Z. *Advanced Methods of Pharmacokinetic and Pharmacodynamic Systems Analysis.* New York: Springer, 2004.

Eckel, Bruce. *Thinking in C++.* Indiana: Prentice Hall, 1999.

Edminister, Joseph A. *Electromagnetismo.* México: McGraw-Hill , 1999.

Herbert Schildt, Gregory L. Guntle. *Borland C++ Builder: The Complete Reference.* McGraw-Hill , 2001.

Hollingworth, Jarrod. *C++ Builder 5 Developer´s Guide.* Indiana: SAMS, 2001.

Institute of Electrical and Electronics Engineers. «IEEE Standard Digital Interface for Programable Instrumentation.» 1992.

Institute of Electrical and Electronics Engineers. «IEEE Standard Signaling Method for a Bi-directional Parallel Peripherial Interface for Personal Computers.» 1992.

Jim Beveridge, Robert Wiener. *Multithreading Applications in Win32: The Complete Guide to Threads.* Addison-Wesley Professional, 1996.

Joseph, Mathai. *Real Time Systems Specification, Verificaction and Analysis.* Londres: Prentice Hall, 2001.

Kalant, Harold. *Principios de Farmacología Médica.* Oxford University Press , 2002.

Katzung, Bertham. *Farmacología Básica y Clínica.* México, D.F.: Editorial El Manual Moderno, 2005.

Kraus, John. *Electromagnetismo: Con Aplicaciones* . McGraw-Hill , 2000.

Lázaro, Antonio Mánuel. *Lab View6i Programación Gráfica para el Control de Instrumentación.* Madrid: PARANINFO, 2001.

LOT Logix Group. *Inpout32.dll.* 2005. http://www.logix4u.cjb.net/ (último acceso: 22 de Enero de 2007).

Malvino, A. *Principios de Electrónica.* México: McGraw-Hill, 1991.

Meek ME, Renwick A, Et al. «Guidelines for application of chemical-specific adjustment factors in dose/concentration.» 2002.

Microchip Technology Incorporated. *PIC16F87X Reference Manual.* California: Microchip, 2001.

Microsoft Corporation. «Microsoft Developer Network.» 2007. http://msdn2.microsoft.com/en-us/library/ms681917.aspx (último acceso: 12 de Diciembre de 2006).

Microsoft. «Real-Time Operating Systems INtime Architecture.» *TenAsys Corporation.* 1 de September de 2003. http://msdn2.microsoft.com/en-us/library//ms838597.aspx (último acceso: 8 de Enero de 2007).

Motorola Inc. «ULN2803 Reference Manual.» 1996.
National Instruments Corporation. *Zone NI.* 20 de Mayo de 2007. http://zone.ni.com/devzone/cda/tut/p/id/3105 (último acceso: 12 de Febrero de 2007).
Paul Horowitz, Winfield Hill. *The Art of Electronics.* Cambridge University Press, 1989.
Ricardo Miledi, *† Et al. «Expression of functional neurotransmitter receptors in Xenopus oocytes after injection of human brain membranes.» *Pubmed*, 2002.
Schildt, Herb. *Borland C++ Builder: The complete Reference.* New York: McGraw-Hill, 2001.
Texas Instruments. «SN7404 Reference Manual.» 1983.
Tozer, Thomas N. *Introduction to Pharmacokinetics and Pharmacodynamics.* Lippincott Williams & Wilkins , 2006.
Umran S. Inan, Aziz S. Inan. *Engineering Electromagnetics.* Mexico: Prentice Hall, 1998.
William H, Hayat. *Teoría Electromagnética.* Mexico: MC Graw Hill , 2000.