Instituto Tecnológico de Costa Rica

Escuela de Ingeniería Electrónica

Diseño de un prototipo embebido para la gestión del padrón electrónico del T.S.E.

Informe de Proyecto de Graduación para optar por el título de Ingeniero en Electrónica con el grado académico de Licenciatura

Jorge Castro Murillo

Cartago, Noviembre del 2011

INSTITUTO TECNOLOGICO DE COSTA RICA

ESCUELA DE INGENIERIA ELECTRONICA

PROYECTO DE GRADUACIÓN

TRIBUNAL EVALUADOR

Proyecto de Graduación defendido ante el presente Tribunal Evaluador como requisito para optar por el título de Ingeniero en Electrónica con el grado académico de Licenciatura, del Instituto Tecnológico de Costa Rica.

Miembros del Tribunal

Ing. Eduardo Interiano Salguero

Profesor lector

Ing. Julio Stradi Granados

Profesor lector

Ing. Faustino Montes de Oca Murillo

Profesor asesor

Los miembros de este Tribunal dan fe de que el presente trabajo de graduación ha sido aprobado y cumple con las normas establecidas por la Escuela de Ingeniería Electrónica

Cartago, lunes 21 de noviembre del 2011

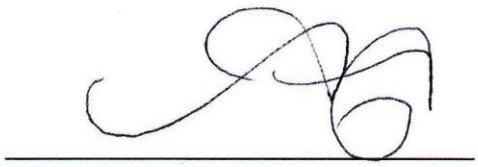
DECLARATORIA DE AUTENTICIDAD

Declaro que el presente Proyecto de Graduación ha sido realizado enteramente por mi persona, utilizando y aplicando literatura referente al tema e introduciendo conocimientos propios.

En los casos en que he utilizado bibliografía, he procedido a indicar las fuentes mediante las respectivas citas bibliográficas.

En consecuencia, asumo la responsabilidad total por el trabajo de graduación realizado y por el contenido del correspondiente informe final.

Cartago, 21 de noviembre del 2011



Firma del autor

Jorge Arturo Castro Murillo

Cédula: 111200900

Resumen

El presente documento describe los detalles del diseño e implementación de un prototipo embebido para ser empleado como alternativa de búsqueda en el padrón electoral durante los comicios en Costa Rica. El kit de desarrollo utilizado es el BeagleBoard-xM, el cual cuenta con un microprocesador de 1GHz ARM® Cortex™-A8, 512 MB de memoria RAM y dispone de puertos de expansión, USB, Ethernet, HDMI, RS232, audio y video para paneles LCD. El sistema trabaja bajo el sistema operativo Ångström, y las aplicaciones fueron desarrolladas mediante el ambiente Qt 4.7 C++, el cual cuenta con un cross-compilador que permite crear aplicaciones cuyo código es ejecutable en una arquitectura diferente a la plataforma en la que él se ejecuta. Cuenta además con una pantalla táctil LCD de colores de 7'', una impresora de matriz de puntos, teclado externo, y lector de cédulas en formato PDF417.

Se diseñó e implementó una aplicación de forma tal que al encender el equipo, el sistema realiza una auto-verificación de hardware y software para garantizar su integridad. Posteriormente, se autentica a los miembros de mesa presentes mediante la lectura del código de barras de la cédula de identidad, e imprime un acta de apertura. Una vez iniciado el periodo de votación permite la búsqueda de electores en el padrón de la Junta Receptora de Votos, indicando si dicha persona ha sufragado o no. De no haber emito el voto, el miembro de mesa confirma que dicha persona va a emitir el voto en la urna. Al finalizar el periodo electoral, el sistema imprime un acta de cierre y se apaga automáticamente.

A pesar de haberse completado los requerimientos funcionales especificados por el Tribunal, se concluye que este sistema no es técnicamente viable para darle continuidad al Proyecto Voto Electrónico debido a las limitaciones del sistema operativo.

Palabras claves: Angstrom, BeagleBoard-xM, Linux, sistemas embebidos, padrón electoral, Qt 4 C++, touchscreen, ULCD7.

Abstract

This document describes de details about the design and deployment of an embedded prototype to be used as an alternative to search in the electoral roll for elections in Costa Rica. The utilized development kit is the BeagleBoard-xM, which features a 1GHz ARM® Cortex™-A8 microprocessor, 512 RAM memory, an expansion port, USB, Ethernet, HDMI, RS232, audio and video for LCD panels. The system works under the operating system Ångström Distribution, and the applications where deployed trough de development environment Qt 4.7 C++, who has cross-compiler that allows to create applications that are executable under a different architecture to the one it is being executed. To be able to interact with de user, the system has a 7'' LCD color display with touchscreen, a dot matrix printer, an external keyboard, and an ID Card with PDF417 format reader.

At power up, the system starts the main application that performs a hardware and software self-test to ensure its integrity. Later, the system authenticates the authenticates de electoral board members by reading the bar code identification card, and prints a record of the opening. Once started the voting period, it allows the search of electors in the polling station, indicating whether the person has voted or not. By not issuing the vote, the board member confirms that this person will cast the vote in the ballot box. At the end of the election period, the system prints a closing act on and shutdowns automatically.

Despite the functional requirements have been completed, it is concluded that this system is not viable to also be used in subsequent projects of the Electronic Vote Project due to the operating system limitations.

Keywords: Angstrom, BeagleBoard-xM, electoral roll, embedded systems, Qt 4 C++, touchscreen, ULCD7.

Dedicatoria

Doy gracias a Dios por su amor y por estar siempre a mi lado.

A mis padres por su infinito apoyo incondicional, su cariño y comprensión.

INDICE GENERAL

INDICE DE FIGURAS

INDICE DE TABLAS

Capítulo 1: Introducción

El Tribunal Supremo de Elecciones –T.S.E.–, es el Órgano Constitucional Superior de la República de Costa Rica en materia electoral. Su principal tarea consiste en la organización, dirección y supervisión de los actos relativos al sufragio, para lo cual goza con el rango y total independencia propios de los Poderes del Estado; por lo que de él dependerán los demás organismos electorales del gobierno.

Como parte de su gestión, el Tribunal es la institución encargada de convocar a las elecciones populares así como de realizar el nombramiento de los miembros de las Juntas Electorales. Es además, quien ejecuta el escrutinio de los votos emitidos en las elecciones de Presidente y Vicepresidente de la República, de Diputados de la Asamblea Legislativa, de los miembros de las Municipalidades, además de los representantes de Asambleas Constituyentes.

1.1 Problema existente e importancia de la solución

Durante más de cinco décadas, como parte del proceso electoral, el gobierno de Costa Rica durante los periodos de votación ha dependido de mecanismos manuales en los centros de votación distribuidos a lo largo del país, que si bien es cierto han sido confiables y seguros, una serie de factores han servido como indicadores de que es necesario impulsar alternativas que permitan modernizar el sistema actual.

Parte de estos señalamientos consideran el crecimiento demográfico del país. Esto significa que al haber mayor cantidad de electores por atender, mayor será la demanda organizacional requerida para la integración de las juntas receptoras de votos, lo cual a su vez implica que el Estado deba designar cada vez mayores fondos económicos para satisfacer dicha necesidad.

Otro aspecto que el Tribunal ha recalcado es el referente al aumento de la cantidad de procesos electorales que se han generado últimamente, por lo cual deberá tenerse en cuenta no solamente las elecciones presidenciales, sino que ahora también las de regidores municipales, y prever eventuales segundas rondas así como referéndums. Sumando a esto, el grado de escolaridad es un factor inherente de quienes conforman las mesas de las juntas receptoras, teniendo una mayor repercusión en la eficiencia del funcionamiento de los procesos de cierre y escrutinio de votos.

Es por ello que el T.S.E. ha planteado un conjunto de estrategias para solventar dicha problemática. Esto involucra básicamente cinco aspectos, los cuales se resumen a continuación: (Sobrado, 2007)

a) Hacer un llamado al sector académico superior para incentivar el desarrollo de un prototipo que incluya hardware y software para implementar la votación electrónica.

b) El diseño del sistema de información deberá realizarse a la medida, bajo el esquema de ser administrado únicamente por el Tribunal, con altos niveles de seguridad física y lógica que garanticen los principios universales del sufragio.

c) Propiciar el uso de software de esta nueva forma de voto electrónico en centros educativos de primaria, secundaria, universitaria y colegios profesionales.

d) Procurar de que la ejecución del proyecto sea evolutiva; esto significa que deberá realizarse paulatinamente, iniciando con la búsqueda en el padrón electrónico en las mesas electorales, hasta finalizar con el escrutinio de los votos en su totalidad.

Sin embargo, el Tribunal aún continúa a la espera de prototipos electrónicos que cumplan con sus requerimientos, los cuales dentro de sus características no funcionales, deberán ser seguros, confiables, transparentes, integrales y de simplicidad en su manejo, para que así pueda agilizar la búsqueda en el padrón electoral.

Es por ello que el Centro de Investigaciones en Computación del Instituto Tecnológico de Costa Rica -C.I.C- busca ofrecer al Tribunal varias alternativas que durante los procesos de votación permitan la identificación del votante, la generación de actas de apertura y cierre, y validar que el elector haya emitido el voto una sola vez. En un mediano plazo, se investigarán otras alternativas relativas al proceso de votación electrónico propiamente en las urnas, una vez que el elector haya sido previamente identificado ante los miembros de mesa.

1.2 Solución seleccionada

La alternativa que será presentada al Tribunal consiste en un sistema electrónico embebido, en donde el objetivo principal por el momento será mostrar sus características funcionales. Para ello, se utilizó el kit de desarrollo *BeagleBoard-xM* propiedad del Centro de Investigaciones en Computación que contiene varios puertos mediante los cuales es posible conectarle otros periféricos que ya han sido adquiridos. Estos son: una impresora de matriz de puntos, un lector de código de barras de cédulas –en formato *PDF417*- y un teclado externo alfanumérico.

El aporte del autor se da en varios aspectos:

Dado que este kit no cuenta con una pantalla para mostrar la información de interés, se decidió adquirir un módulo externo *ULCD7 LITE* que contiene un display LCD de 7’’ a colores, el cual cuenta además con una pantalla táctil resistiva integrada. Este

módulo se comunica con el puerto LCD del *BeagleBoard-xM* y dispone de una réplica del puerto de expansión de 28 pines.

Puesto que uno de los requerimientos fundamentales indicados por el Tribunal es que el proyecto sea auditable, se procedió a seleccionar una distribución que fuese de código abierto y que utilizara un mínimo tiempo de carga. Se analizó la posibilidad de utilizar la distribución *Ubuntu 11.04*, sin embargo esta no es factible puesto que el módulo *ULCD7 LITE* está diseñado para funcionar exclusivamente con la distribución *Ångström*, quien a su vez cuenta con un *kernel* basado en el sistema operativo *Linux*.

A partir de estas herramientas, se atendieron ciertos casos de uso especificados en (Castro & Schmidt, 2011), documento en el cual se presentan los diferentes escenarios que ocurren durante el periodo de votación. Conociendo estos requerimientos, se creó la aplicación principal del proyecto mediante el ambiente de desarrollo integrado *Qt 4 C++.*

El diagrama de flujo general de la aplicación se muestra en la Figura 1.1.

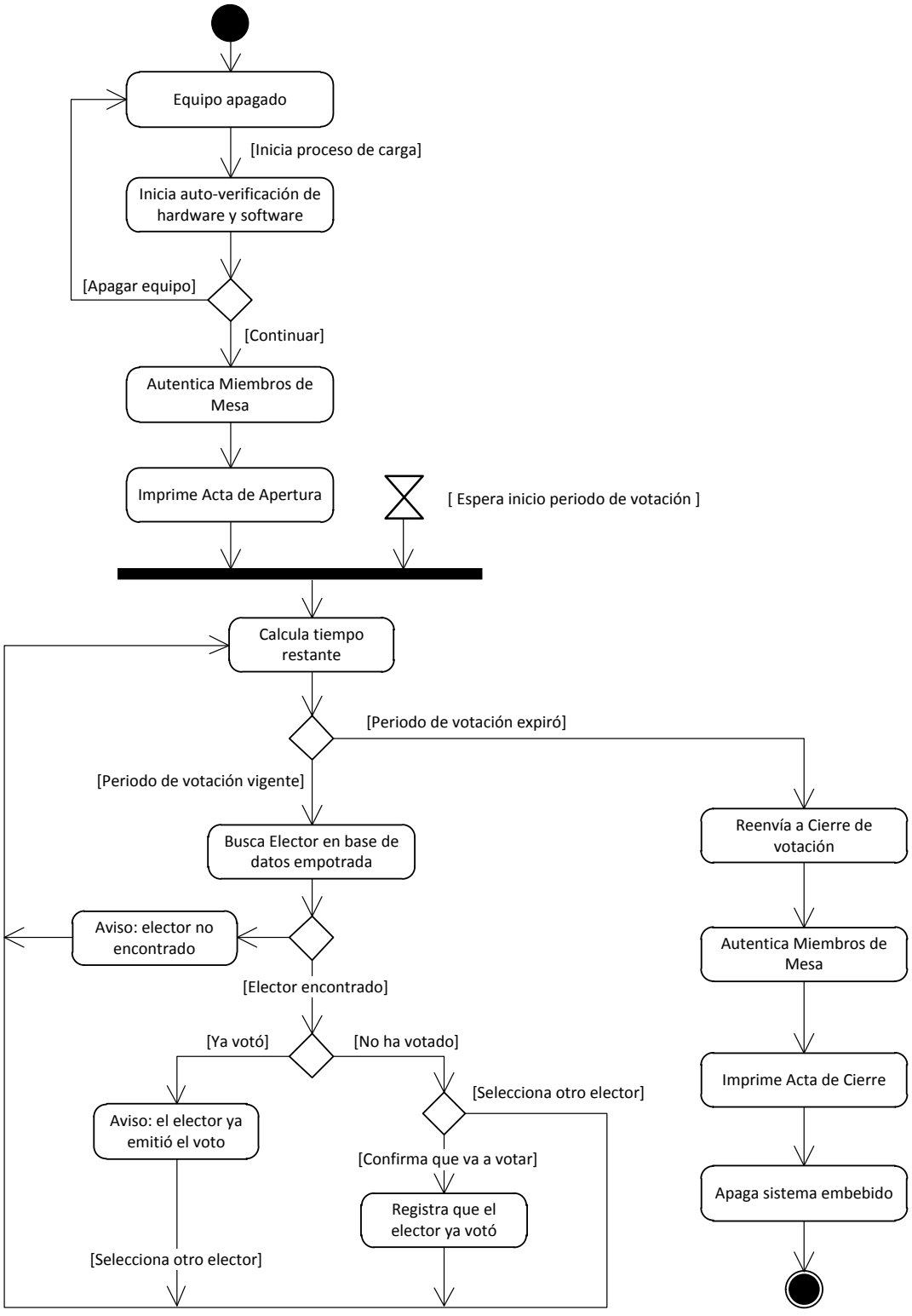


Figura 1.1 Diagrama general de la aplicación principal del sistema

Además, se optimizó el uso de recursos del sistema embebido de manera que el tiempo de carga del sistema embebido sea el mínimo posible; además de que las transiciones entre las ventanas de la aplicación sean desapercibidas por el usuario y se crearon las pantallas inicio y cierre –*bootloader*– del sistema operativo.

Finalmente, se analizó cuál es la viabilidad que tiene este sistema embebido para ser utilizado como base para proyectos subsecuentes que vayan a ser presentados al Tribunal.

Capítulo 2: Meta y Objetivos

2.1 Meta

Disponer de un sistema electrónico portable, seguro, autónomo y redundante, que facilite la logística, mejore la accesibilidad, disminuya la inversión y automatice el proceso de búsqueda de los electores en el padrón foto electoral.

2.2 Objetivo general

Diseñar e implementar un prototipo de arquitectura embebida que permita la gestión del padrón electrónico electoral.

2.3 Objetivos específicos

a) Analizar la viabilidad del kit de desarrollo para satisfacer los requerimientos funcionales especificados por el proyecto de investigación Voto Electrónico del C.I.C.

b) Disponer de un padrón electrónico con fotografía para verificar la condición del elector de la Junta Receptora de Votos.

c) Optimizar el uso de recursos del sistema para agilizar el proceso de apertura, verificación de electores y cierre de mesa.

d) Validar la viabilidad de uso del sistema en futuros proyectos de sistemas embebidos.

Capítulo 3: Marco teórico

3.1 Sistemas embebidos

Los sistemas embebidos pueden describirse, en términos generales, como aquellos sistemas electrónicos que están conformados por elementos tanto de hardware como de software, distinguidos por ser de dimensiones físicas reducidas, y que a diferencia de computadoras personales o supercomputadoras –*mainframes*-, están destinados a realizar tareas muy específicas.

Parte del diseño del embebido, involucra considerar cuáles son los requerimientos establecidos y determinar hasta qué punto es posible satisfacer esas necesidades de acuerdo con las herramientas tecnológicas que se disponen. Estas últimas han permitido el desarrollo de sistemas embebidos utilizados en soluciones médicas, aeronáuticas, telecomunicaciones, comercio, industriales, del consumidor, militares, automotrices, redes, entre otros. Ejemplos de ellos son dispositivos portátiles para monitoreo de pacientes en tiempo real con interfaz gráfica, adquisición y procesamiento digitales de imágenes en 3D y 4D (Intel Corporation), cámaras digitales, teléfonos móviles, terminales de puntos de ventas, analizadores de espectro, reproductores de sonido y video, video juegos, automóviles eléctricos, entre otros.

3.2 Sistemas operativos

Según el nivel de complejidad de las tareas que realicen, algunos embebidos requieren de un sistema operativo que permitan brindarle al dispositivo modularidad, escalabilidad, ser fácilmente configurable, soporte para controladores, soporte para múltiples microprocesadores, así como estabilidad y rendimiento aceptable, entre otros factores. A continuación se mencionan algunos de los sistemas operativos encontrados en dispositivos recientes:

a) Windows CE: sistema operativo propietario de Microsoft Corporation.
b) Android
c) Ubuntu
d) Ångström
e) MeeGo
f) Symbian
g) QNX

3.3 Kit de desarrollo Beagleboard-xM

Debido a la gran variedad de familias de microprocesadores y microcontroladores disponibles en el mercado, múltiples organizaciones han desarrollado kits de desarrollo, brindándoles a los usuarios la posibilidad de utilizar muchas de las funcionalidades con que cuentan estos circuitos integrados.

Cada uno de estos kits se enfoca en alguna o algunas de las características más importantes del chip, por lo que le son agregados los módulos necesarios para potenciar al máximo el uso de tales recursos. Por ejemplo algunos se interesan más en la parte de procesamiento digital de señales; otros en la parte gráfica al disponer de cámaras y procesadores de video; también están aquellos más enfocados al área de redes.

Uno de tales kits es el *Beagleboard-xM*. Este producto se distingue por su bajo costo –$149 USD-, por contar con el soporte de la comunidad *open-source* de *Beagleboard.org*, por su pequeño tamaño –8.5 cm x 8.8 cm-, así como tener un rendimiento muy cercano al de una computadora portátil tipo *laptop*.

Sus principales características de hardware se muestran en la Tabla 3.1 (BeagleBoard.org, 2010)

Tabla 3.1 Principales componentes de hardware del kit de desarrollo BeagleBoard-xM

Componente	**Características**	
Procesador	Texas Instruments Cortex A8 1GHz processor - DM3730CBP	
POP Memory	Micron 4Gb MDDR SDRAM (512MB) 200MHz	
PMIC TPS65950	Power Regulators	
	Audio CODEC	
	Reset	
	USB OTG PHY	
Soporte para depuración	14-pin JTAG	GPIO Pins
	UART	3 LEDs
PCB	3.1” x 3.0” (78.74 x 76.2mm)	6 capas
Indicators	Power, Power Error	2-User Controllable
	PMU	USB Power
HS USB 2.0 OTG Port	Mini AB USB connector	
	TPS65950 I/F	
USB Host Ports	SMSC LAN9514 Ethernet HUB	
	4 FS/LS/HS	Up to 500ma per Port if adequate power is supplied
Ethernet	10/100	From USB HUB
Audio Connectors	3.5mm L+R out	3.5mm L+R Stereo In
SD/MMC Connector	MicroSD	
User Interface	1-User defined button	Reset Button
Video	DVI-D	S-Video
Cámara	Connector	Supports Leopard Imaging Module
Power Connector	USB Power	DC Power
Overvoltage Protection	Shutdown @ Over voltaje	
Main Expansion Connector	Power (5V & 1.8V)	UART
	McBSP	McSPI
	I2C	GPIO
	MMC2	PWM
2 LCD Connectors	Access to all of the LCD control signals plus I2C	3.3V, 5V, 1.8V
Auxiliary Audio	4 pin connector	McBSP2
Auxiliary Expansion	MMC3	GPIO,ADC,HDQ

En la Figura 3.1 se muestra una fotografía del sistema embebido *Beagleboard-xM*

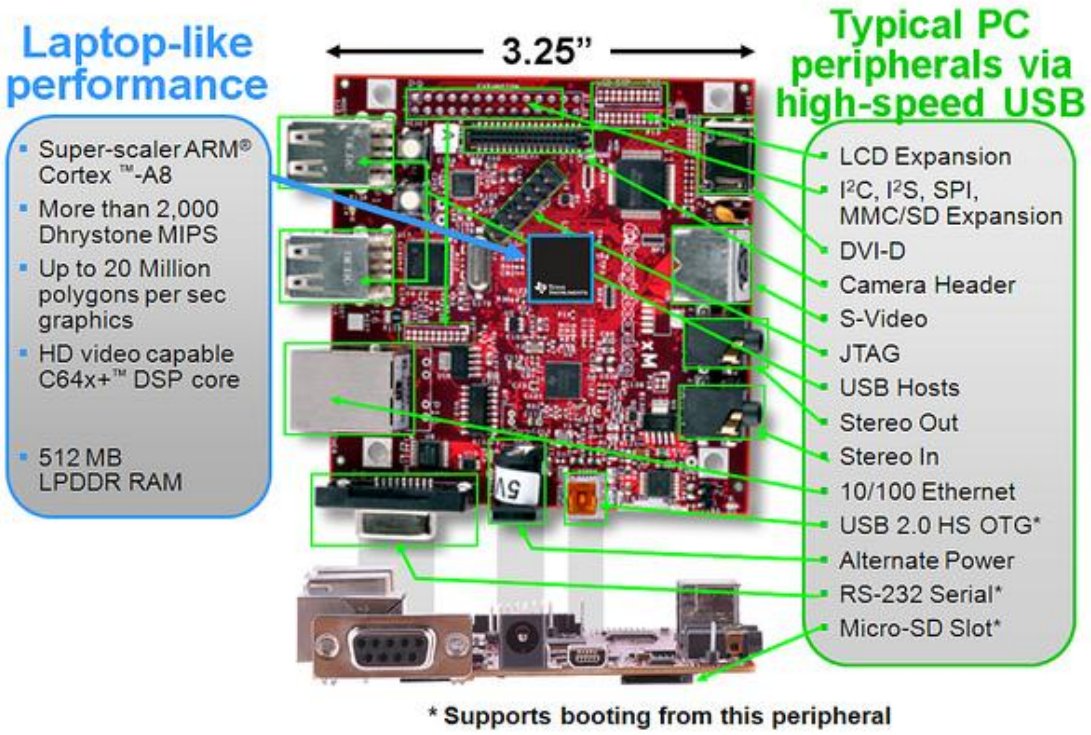


Figura 3.1 Fotografía del sistema embebido *Beagleboard-xM* destacando sus principales características.

Este embebido, el cual es fabricado por la empresa estadounidense *Circuitco Corporation*, ha lanzado al mercado varias versiones de la familia *BeagleBoard*. La primera de ellas, fue introducida en julio del 2008 (Digi-Key Corporation, 2008).

La segunda versión, la *BeagleBoard Rev C*., la cual fue estuvo a la venta a partir de mayo del 2009 (Digi-Key Corporation, 2009).

Al año siguiente, en setiembre del 2010 (Beagle Board, 2011), fue lanzada la versión más reciente: la *BeagleBoard-xM*. Todas ellas, siempre bajo el mismo costo de $149 al detalle.

Sin embargo, vale la pena dejar en claro que este kit de desarrollo es enviado a los clientes incluyendo únicamente como componente adicional una tarjeta de memoria tipo *Micro-SD* de 4 GB. Esta contiene una distribución del sistema operativo *Linux Ångström*, con la Rev. 4.25. El resto de herramientas, tales como cables Ethernet, cables mini-USB, fuente de alimentación externa de 5V, cable con conector DB9 macho para puerto RS-232, entre otros, deben ser adquiridos por el comprador por su propia cuenta.

3.4 Módulo Reloj en Tiempo Real (RTC)

En algunos kits para el desarrollo de sistemas embebidos, se presentan situaciones en las que cierta información que forma parte del proyecto debe mantenerse en continua actualización. En el caso específico de aquellos embebidos que requieren mantener la fecha y la hora ininterrumpidamente una vez que se retira la fuente de alimentación del sistema, es posible recurrir al uso de módulos de bajo consumo de potencia cuya fuente de energía sea independiente de todas las demás presentes.

Una muestra de dichos módulos es el *DS1307 RTC Module*, fabricado y distribuido por la empresa *SparkFun Electronics*. Básicamente está compuesto de tres elementos: una batería de litio de 3V –modelo CR1225-, un oscilador de cuarzo de 32 kH –32,768 Hz-, y el circuito integrado DS1307 –reloj de tiempo real-. En la Figura 3.2 se muestra dicho módulo.

Figura 3.2 Vista de frente y posterior del módulo de Reloj de Tiempo Real, RTC.

El protocolo de comunicación que se utiliza tanto para leer datos como para guardarlos en el RTC es el I^2C –del inglés *Inter-Integrated Circuit*-. Para ello, se define un modelo en el que uno de los circuitos integrados juega el papel de maestro –*master*- y los demás juegan el rol de esclavos –*slaves*-, en donde el maestro es quien administra la comunicación entre todos los chips conectados a este mismo bus de datos, determinando en qué momento un esclavo puede tomar el control de la línea y bajo qué condiciones.

Para realizar la conexión física, el fabricante del *DS1307* sugiere considerar en el diseño el diagrama que se muestra en la Figura 3.3.

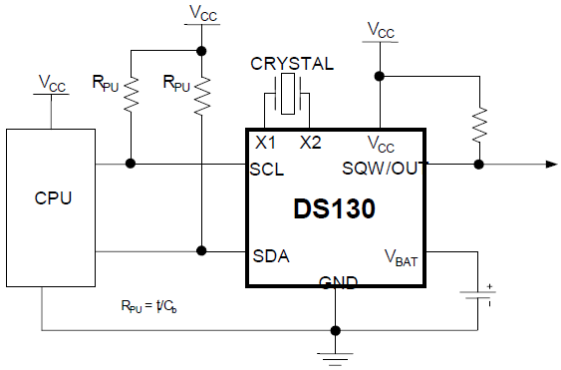


Figura 3.3 Conexión física entre el reloj de tiempo real *DS1307* y el dispositivo master para el uso del bus I^2C

Las principales características funcionales de este RTC, es que permite almacenar en el chip tanto la fecha como la hora –segundos, minutos, horas, día de la semana, día del mes, mes, y año-. Estos datos son guardados en sus registros en formato *BCD* –Código Binario Decimal- para facilitar su interpretación. Toma también en cuenta aquellos casos en los que el año es bisiesto y también permite que el formato de la hora almacenada sea de 12 o 24 horas con indicador de AM/PM.

Para realizar las operaciones de lectura y escritura, es necesario considerar la tabla de registros que indica el fabricante, la cual se muestra en la Tabla 3.2 (Maxim Integrated Products, 2008).

Tabla 3.2 Registros del chip reloj en tiempo real DS1307

ADDRESS	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	FUNCTION	RANGE
00h	CH	10 Seconds			Seconds				Seconds	00–59
01h	0	10 Minutes			Minutes				Minutes	00–59
02h	0	12	10 Hour	10 Hour	Hours				Hours	1–12 +AM/PM 00–23
		24	PM/ AM							
03h	0	0	0	0	0	DAY			Day	01–07
04h	0	0	10 Date		Date				Date	01–31
05h	0	0	0	10 Month	Month				Month	01–12
06h	10 Year				Year				Year	00–99
07h	OUT	0	0	SQWE	0	0	RS1	RS0	Control	—
08h–3Fh									RAM 56 x 8	00h–FFh

Por defecto, una vez alimentado el chip, la fecha y hora de inicio son: 1º de enero del 2000 y 00:00:00 en formato 24 horas, respectivamente.

Para su operación este chip requiere de una alimentación Vcc dentro del rango de 4.5V y 5.5V; sin embargo, en caso de que el valor de Vcc sea menor a 4.5V, se dispone de una entrada V_{BAT} que es utilizada como alternativa para mantener en ejecución el RTC sin que haya pérdida de información. Esta opción requiere que la tensión de V_{BAT} se encuentre dentro del rango de 2 V a 3.5 V con un valor nominal de

3 V. La condición necesaria para que un dispositivo externo pueda leer o escribir datos en los registros de este chip, es que la tensión mínima de alimentación sea de 1.25 x V_{BAT}.

3.5 Pantallas táctiles

Las pantallas táctiles o *touchscreens* son utilizadas en circunstancias en las que el usuario desee o requiera interactuar con el dispositivo a través de su pantalla de una manera directa y sencilla sin la necesidad de recurrir a otros instrumentos intermedios, tales como el teclado o el ratón, por ejemplo.

Existen principalmente cuatro tipos de tecnologías de pantallas:

a. Resistivas
b. Capacitivas
c. Infrarrojas

Entre otras se pueden encontrar las que utilizan tecnología SAW (*Surface Acoustic Waves*, el cual utiliza ondas ultrasónicas, en donde al presionar una parte de la superficie, una porción de las ondas son absorbidas lo que permite determinar su ubicación); APR (*Acoustic Pulse Recognition*, en donde al presionar una parte de la superficie se genera un sonido predefinido, con la ventaja de despreciar ruidos debidos a factores ambientales, por ejemplo).

Las láminas resistivas están conformadas por dos capas plásticas, cada una recubierta de una capa conductora de metal ultra delgada, y separadas entre sí por aire. En la Figura 3.4 se puede observar un diagrama de cómo está conformada la pantalla táctil de cuatro hilos (Analog Devices, Inc., 2011).

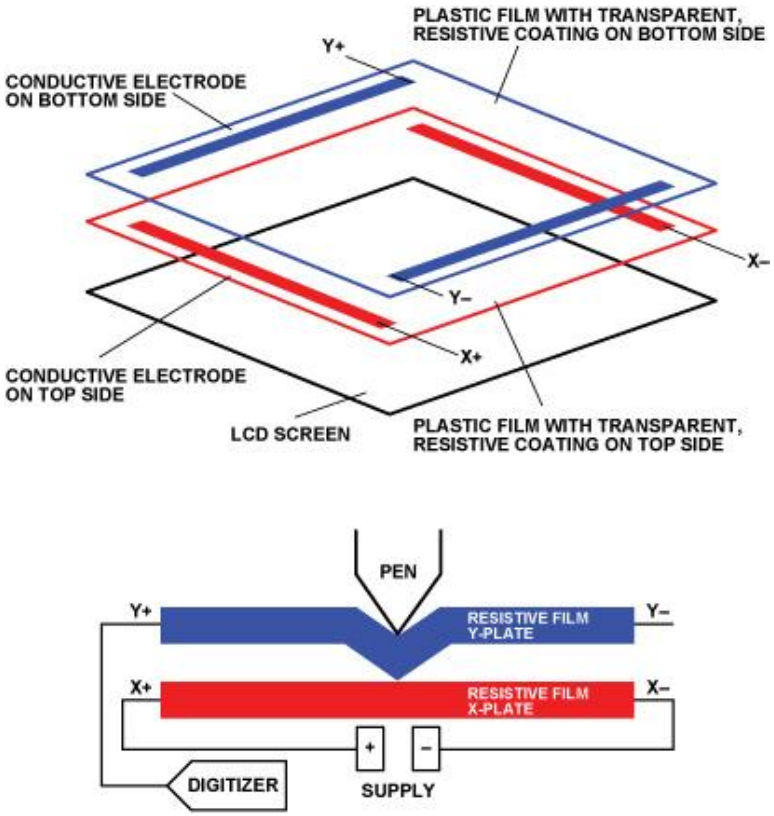


Figura 3.4 Componentes general para la detección de coordenadas XY de una *touchscreen*

En dicha figura puede observarse que cada lámina tiene conectados un par de electrodos en sus extremos (X+ y X- para la lámina inferior, e Y+ e Y- para la lámina superior). Al desplazarse de un electrodo a otro (por ejemplo, de X+ a X-) la lámina puede modelarse como una resistencia que varía linealmente con la distancia de separación. Para determinar cuál es la ubicación de un punto en el plano XY, primero se somete la lámina ±X a una diferencia de potencial V_{REF}; al ejercer presión sobre la lámina superior, ambas entran en contacto, por lo que utilizando el principio de división de tensión, se puede determinar a qué distancia sobre el eje X corresponde dicho punto. Utilizando la misma analogía, la posición sobre el eje Y se obtiene sometiendo la lámina ±Y a una diferencia de potencial V_{REF} y midiendo la tensión presente en el cable X+. Estos valores analógicos son muestreados mediante un convertidor analógico digital –ADC, por sus siglas en inglés- para ser procesados por

un microcontrolador –*driver* ó controlador de hardware- y enviados al CPU. En la Figura 3.5 se puede observar el esquema general para la conexión entre la lámina táctil resistiva y el microprocesador –CPU-.

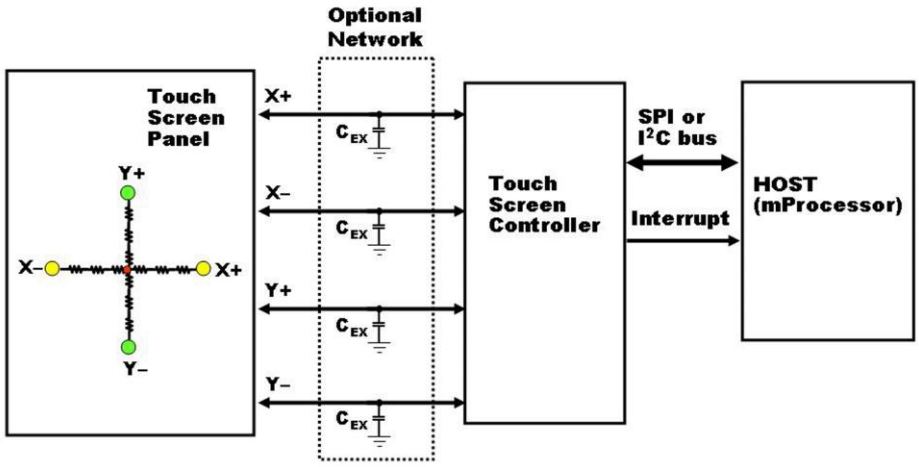


Figura 3.5 Esquema general de conexión entre la lámina táctil resistiva y el microprocesador

Algunos de los protocolos utilizados para la transmisión de datos entre el controlador de la pantalla táctil y el CPU son: *I^2C, SPI, USB y RS232*. Una vez recibidos estos datos en el CPU, es necesario utilizar un controlador de software que permita interpretar las coordinadas X y Y para que estas coincidan con el valor esperado; este último proceso se conoce como calibración de la pantalla táctil.

3.5.1 Controlador de pantalla táctil SLT-TP05-USB

El controlador de pantalla táctil modelo *SLT-TP05-USB*, es un dispositivo electrónico que permite la conexión entre una *touchscreen* de cuatro hilos y un *host* –CPU- que soporte el protocolo *USB 1.1*. La alimentación de +5V DC es tomada del puerto USB del host, con un consumo de 70 mA. Brinda soporte para una resolución de hasta 2048 x 2048 pixeles y garantiza un tiempo de respuesta menor a 24 ms. Cuenta también con controladores de software para los sistemas operativos Windows, Linux,

iMac y QNX, con lo cual es posible emular el uso de un ratón. En la Figura 3.6 se muestra una fotografía de dicho controlador.

Figura 3.6 Controlador de touchscreen SLT-TP05-USB

3.6 Ambiente de programación Qt4 C++

Debido al surgimiento de múltiples sistemas operativos en diferentes arquitecturas de hardware, los desarrolladores de software se han visto en la necesidad de buscar alternativas mediantes las cuales el producto sea compatible en la mayoría de los sistemas conocidos. Esto significa tomar en consideración aspectos relacionados con capacidad para lograr la correcta ejecución del software entre diferentes sistemas operativos y diferentes versiones, la dependencia que exista entre los paquetes instalados, la escalabilidad de los programas desarrollados y la seguridad, por mencionar algunos.

Uno de los ambientes de desarrollo de software ampliamente utilizados en la actualidad es el *Qt 4.7 C++*. Esta es una biblioteca multiplataforma que cuenta con herramientas para el diseño de aplicaciones que pueden ejecutarse en sistemas operativos de escritorio, así como en móviles. Algunos de ellos son el *Symbian* y el *MeeGo* –utilizados en *smartphones* ó teléfonos inteligentes-, *Microsoft Windows*, *Mac OS X*, y *Linux* (Nokia Corporation, 2011).

Entre sus clientes se encuentran: *Cadence Design Systems, Siemens, Sennheiser, Samsung, Walt Disney Animation Studios, Volvo Mobility Systems, National Instruments*, por mencionar algunos (Nokia Corporation, 2011).

Este ambiente se destacada también por disponer al usuario de una interfaz gráfica –el *Qt Creator*-, *toolchains* para realizar compilación cruzada, simuladores e interfaz de programación de aplicaciones (*APIs*, de sus siglas en inglés). Existen dos opciones de licencia: la comercial -*Qt Commercial License*- y la de código abierto -*GNU Lesser General Public License* (*LGPL*) versión 2.1- .

Capítulo 4: Procedimiento metodológico

4.1 Reconocimiento y definición del problema

Durante más de una década el Tribunal Supremo de Elecciones ha venido impulsando el desarrollo de alternativas tecnológicas que permitan agilizar los trámites de votación. Básicamente existen dos modalidades: los sistemas remotos de votación –realizados a través de *i-voting* o por vía telefónica- y los sistemas presenciales; es en este último, en el cual el Centro de Investigaciones en Computación –CIC- se ha enfocado a través de su Proyecto Voto Electrónico –VE-.

Uno de los limitantes que afronta el Tribunal es el factor económico. La opción de optar por una solución comercial requeriría de una inversión de recursos que se encuentra lejos del alcance previsto. De allí, la mejor opción como punto de partida, es diseñar e implementar un sistema electrónico confiable, auditable, seguro, que logre satisfacer los requerimientos funcionales y no funcionales y que supere en mucho las expectativas del Tribunal. Es a partir de esto que el C.I.C se ha dado la tarea de investigar sobre posibles escenarios que puedan ser planteados ante el Tribunal, de manera de esta última entidad se decida por la opción más conveniente o bien, indique aquellos cambios al sistema de votación electrónico para ajustarlo a sus necesidades.

Por tanto, durante el año 2010, el C.I.C realiza un análisis de requerimientos presentados por el TSE a partir de un estudio de factibilidad que esta última institución realizó entre el año 2008 y 2009. Con esto se logra concretar un primer marco de referencia que abarca las necesidades recopiladas para el desarrollo del prototipo electrónico.

4.2 Obtención y análisis de información

Para determinar cuáles habrían de ser los requerimientos del prototipo funcional que el C.I.C. planea presentar ante el Tribunal, se partió del uso del primer marco de referencia especificado por (Castro & Schmidt, 2011) como parte de su Proyecto Voto Electrónico. Este se enfoca en definir los requerimientos funcionales y los no funcionales del proceso de identificación de votantes en el padrón electoral.

El proceso de votación se subdivide principalmente en tres partes. La primera de ellas es cuando el elector se presenta ante los Miembros de mesa. Allí presenta su cédula de identidad y uno de los miembros lo busca en el Padrón Electoral, en donde corrobora su número de cédula, nombre, apellidos y su fotografía en blanco y negro. En la segunda parte, el elector se dirige a la urna en donde emite su voto en secreto; finalmente, se dirige nuevamente a la mesa de votación en donde deposita las papeletas, y en la tercera parte del protocolo firma el padrón como garantía de haber ejercido el voto presencialmente.

Este proyecto de investigación está enfocado exclusivamente a la primera de las tres etapas recién mencionadas, proceso en el cual, los miembros de mesa deben buscar al elector en el padrón; esta vez de una manera sencilla y mucho más rápida en comparación con la búsqueda manual.

Para conocer exactamente cuáles son los requerimientos funcionales y no funcionales del prototipo, es necesario recurrir a un par de secciones especificadas en el marco de referencia (Castro & Schmidt, 2011), los cuales son, respectivamente:

a) Sección: 2.1: Requerimientos funcionales

b) Sección: 2.2: Requerimientos no funcionales

Los contenidos de este par de secciones se pueden encontrar en el Anexo 1.

4.3 Evaluación de las alternativas y síntesis de una solución

Para cumplir con los requerimientos planteados en (Castro & Schmidt, 2011), se procedió a utilizar el kit para desarrollo de sistemas embebidos ya disponible en el C.I.C. Este equipo tiene la capacidad de soportar un sistema operativo embebido y lograr la comunicación con los periféricos presentes alimentado por una única fuente de 5V. Este sistema cumple con el criterio de auditabilidad solicitado por el Tribunal al contar con la licencia *Creative Common Works* (Creative Commons, 2011), por lo que también se recurrió al uso de paquetes de software cuyas licencias son de código abierto.

Por defecto, el *Beagleboard-xM* trae preinstalado el *sistema operativo Ångström, el cual es una distribución basada en el Debian* Project y está orientado para ser utilizado en sistemas embebidos. Tiene la ventaja de brindar soporte para interfaz gráfica DVI –*Digital Visual Interface*-. El mayor inconveniente de este sistema operativo es su limitada cantidad de paquetes de software disponibles y la gran cantidad tiempo que tarda en cargar el sistema –aproximadamente 2 minutos en modo consola y 5 minutos en modo gráfico-.

Otro posible sistema operativo evaluado es el *Ubuntu 11.04*, el cual también está basado en el *Debian Project*, pero está más orientado al uso en computadoras personales, de escritorio y servidores. Sin embargo cuenta con una cantidad mucho mayor de paquetes de software disponibles en comparación con el sistema *Ångström*. Adicionalmente, existe mayor soporte por parte de la comunidad de desarrolladores por lo que su estabilidad es destacable. Al no estar orientado para su uso en sistemas embebidos, se debe prestar atención a la demanda de recursos de hardware y software de los paquetes que se le instalen. A pesar de todo esto, el tiempo que tarda en cargarse el sistema en modo consola es menor a los 45 segundos.

La selección del sistema operativo va a depender mayoritariamente del escenario que se presente. Las dos posibles soluciones son los siguientes:

a) Primer solución: uso del *BeagleBoard-xM*, display de 10.4'' marca NEC, accesorios OEM como impresoras y lectores de código de barras PDF417 y touchscreen resistivo, operando en *Ubuntu 11.04.*

b) Segunda solución: uso del *BeagleBoard-xM* adjunto al módulo *ULCD7 LITE* que incluye display de 7'' operando en *Ångström*, impresoras y lectores de código de barras PDF41julio del uso general.

Ambas opciones están sujetas a la disponibilidad de los recursos económicos y su tiempo de entrega. Por tanto, dado que el C.I.C ya contaba con el kit de desarrollo *BeagleBoard-xM*, la impresora y el lector de código de barras PDF417, se procedió a investigar cuál display LCD sería compatible con el kit de desarrollo y por tanto se adquirió el módulo *ULCD7 LITE* para completar la segunda solución. La principal restricción de este módulo es que está diseñado para funcionar por defecto únicamente con la distribución Ångström.

4.4 Implementación de la solución

La solución seleccionada consiste en utilizar básicamente los siguientes elementos de hardware:

a) Kit de desarrollo *BeagleBoard-xM* –brindado por el C.I.C-

   Esta tarjeta está compuesta básicamente por cuatro puertos USB, uno de video DVI con conector HDMI –*High Definition Multimedia Interface*-, puerto Ethernet de 10/100 Mbps, puerto para comunicación con dispositivos LCD con interfaz LVDS, puerto serial RS232, puerto de expansión de 28 pines, así

como la ranura para un tarjeta de memoria *SD Card Mini* de 4GB*.* Se analizará la viabilidad sistema operativo *Ångström* en comparación con la de *Ubuntu 11.04.*

b) Módulo *ULCD7 LITE* –seleccionado por el autor-

Este módulo posee una pantalla de cristal líquido LCD de 7'' con soporte para *touchscreen* de tipo resistivo. Trae consigo los pines necesarios para conectarse directamente con el *BeagleBoard-xM* y además dispone de un duplicado del puerto de expansión de 28 pines de este último. Adicionalmente trae una base metálica removible en dos de sus extremos para brindarle inclinación de 45° a la pantalla y facilitar su visualización.

c) Módulo de Reloj en Tiempo Real (*Real Time Clock*, RTC) –seleccionado por el autor-.

Es necesario almacenar la fecha y hora del sistema operativo en un módulo externo puesto que el kit de desarrollo no cuenta con dicha opción una vez retirada la alimentación. Una vez encendido el embebido, se deben tomar los datos del RTC y a partir de ellos actualizar los del sistema operativo.

d) Accesorios tales como lector de código de barras en formato PDF417, impresora térmica, teclado estándar USB, fuente de alimentación de 5V, cable de red Ethernet, convertidor puerto RS232-USB, y memoria extraíble *USB Flash Card.* De todos ellos, los dos primeros fueron proporcionados por el C.I.C, el resto fueron seleccionados por el autor.

En cuanto a recursos de software, se decidió utilizar el ambiente de desarrollo para sistemas embebidos *Qt 4.7 C++* para crear la aplicación principal.

4.5 Reevaluación y rediseño

Si bien el módulo ULCD7 LITE al contar con una pantalla LCD y *touchscreen* incorporado es un buen punto de partida para cumplir con la mayoría de especificaciones solicitadas en el marco de referencia del Proyecto Voto Electrónico (Castro & Schmidt, 2011), el sistema operativo *Ångström* que provee el fabricante para uso exclusivo de este producto no representa una gran ventaja en cuanto a rendimiento y soporte para paquetes de software, en comparación con la distribución *Ubuntu 11.04*. La única ventaja es que el *kernel* ha sido configurado para utilizar el puerto LCD y el controlador del *touchscreen* ha sido instalado y calibrado adecuadamente.

La meta sería poder configurar la versión *Ubuntu 11.04* para que soporte el LCD de 10.4'', además de cross-compilar e instalar las bibliotecas para instalar el controlador del touchscreen de 10.4''. Esto brindaría mayor comodidad para el usuario al disponer de un área visual mucho mayor, y el tiempo de encendido y apagado se reduciría significativamente (disminuiría de 2 minutos a 45 segundos de espera para el encendido). Sin embargo, es poco viable darle continuidad al uso del *BeagleBoard-xM* junto con el módulo *ULCD7 LITE* debido a la enorme complejidad de modificar y asegurar el correcto funcionamiento de su *kernel*.

Capítulo 5: Descripción detallada de la solución

5.1 Análisis de soluciones y selección final

Se han planteado dos posibles soluciones que tienen en común el uso del mismo kit de desarrollo *BeagleBoard-xM* junto con los accesorios; sus diferencias significativas son el sistema operativo (en un caso *Ångström*, en el otro *Ubuntu*), las dimensiones y marca del display LCD, lo mismo que para el caso del *touchscreen*.

El factor determinante para efectos de este proyecto es el tiempo de adquisición de los componentes de hardware y su costo asociado. Por lo tanto, a continuación se describe la solución que se logró implementar utilizando componentes de menor tamaño, costo y pronta disponibilidad.

5.2 Selección del sistema operativo.

El kit de desarrollo *BeagleBoard-xM* es una plataforma de hardware que cuenta con la opción de funcionar con un sistema operativo de alto nivel que haya sido compilado para esta arquitectura embebida; algunos de ellos son el Windows CE, Linux, QNX, Symbian, entre otros.

Debido que el microprocesador que trae incorporado es de la familia ***ARM Cortex-A8*** (específicamente el modelo ***OMAP DM3730*** de 32 bits, producido por *Texas Instruments*), esto excluye el uso directo de aplicaciones que operan bajo otras arquitecturas como por ejemplo x86/64.

Para ampliar el uso de estos sistemas operativos en este tipo de embebidos, fue necesario recurrir a un cross-compilador (*cross-compiler* o *toolchain*), el cual es una aplicación que es capaz de producir código ejecutable para una arquitectura diferente a la que está ejecutándose. Así por ejemplo, si el cross-compilador fue instalado en

una computadora x86, es posible que este compile código que sea ejecutable exclusivamente en el sistema operativo Linux del kit *BeagleBoard-xM*.

Por defecto, el fabricante del módulo *ULCD7 LITE*, que es el que contiene el panel LCD de 7'' y trae integrada la pantalla *touchscreen* resistiva, brinda una imagen modificada del sistema operativo *Ångström* en su página web (BeagleBoard Toys Co., 2011). Una vez descargada en la PC, esta imagen se descomprimió y fue transferida a la memoria *SD Card mini* de 4GB, obteniéndose dos particiones: un sistema de archivos *Root File System* con formato *ext4*, y una segunda partición con formato *FAT32* que contiene los archivos de arranque del sistema –el *bootloader*-. Luego de encender el embebido, el *bootloader* realiza el mapeo de memoria y descomprime el *kernel* para luego inicializar los servicios del sistema según sean necesarios.

Por fortuna esta distribución no requirió modificaciones para poner en operación el LCD, el *touchscreen*, así como los demás puertos de entrada y salida –a diferencia de otras distribuciones en donde usualmente el *kernel* no detectaba el puerto Ethernet ni los puertos USB-. Sin embargo, para el caso de la distribución *Ubuntu 11.04* el soporte para video por defecto es mediante DVI y utiliza para ello un conector HDMI –la interfaz DVI es un subconjunto de la HDMI, ya que es solamente la parte digital de video-, pre-configurado para una resolución de 1024x768; la señal de video LCD no pudo ser procesada por el módulo *ULCD7 LITE* y el touchscreen lo detectó pero no dispone de la herramienta para calibración del área de emulación.

El *BeagleBoard-xM* cuenta con un conector de 2x10 pines, y es proveído para brindar conexión con adaptadores para paneles LCD, y dada la variedad de estos últimos, solamente se disponen de las señales sin procesar –*raw signals*-. De allí que el módulo *ULCD7 LITE* ya cuenta con los adaptadores electrónicos necesarios para

trasladar las señales de este puerto hacia la interfaz *RGB* que se requiere para brindarle funcionalidad al LCD de 7''.

Si se quisiese instalar la versión Ubuntu para funcionar con el módulo mencionado previamente, habría de recompilarse el *bootloader* para que este le pase los parámetros al *kernel* durante el proceso de arranque. Habría de ser necesario además, conocer muy a fondo el funcionamiento del *kernel*, lo cual reduce en cantidad la viabilidad de esta arquitectura.

Por tanto, se selecciona la distribución *Ångström* para efectos de este proyecto, y se invita a utilizar la distribución *Ubuntu 11.04* o posteriores para futuras investigaciones.

5.3 Selección del ambiente de desarrollo.

Qt4 C++ cuenta con la posibilidad de utilizar varios compiladores (*toolchains*) previamente instalados en la PC y dispone de la herramienta de diseño *Qt Creator* para crear aplicaciones gráficas (GUI, *Graphical User Interface*).

El ambiente de desarrollo puede ser descargado de (Nokia Corporation, 2011). La versión utilizada es el *Qt SDK offline version 1.1.3*, para arquitecturas Linux de 64-bits, y el sistema operativo de la PC es la distribución *Ubuntu 11.04 Natty 64-bit*.

El procedimiento de instalación de este SDK se describe en el Apéndice A.1.

5.4 Puesta en marcha del sistema embebido

Una vez transferida la imagen del sistema operativo a la memoria *SD Card Mini*, son creadas dos particiones: la que contiene el sistema de archivos en formato *ext4* y la de arranque en formato *FAT32*.

En la partición de arranque se encuentran los siguientes archivos, los cuales son generados a partir de la herramienta *U-Boot* (Osier-Mixon, 2010):

a) Bootloaders X-Loader (MLO) y *U-Boot (u-boot.bin)*

b) Kernel (*uImage*)

c) Script de arranque (*boot.scr* y *uEnv.txt*)

d) Sistema de archivo de la memoria RAM (*uInitrd*)

Adicionalmente habrá un archivo de texto plano *boot.cmd*, a partir del cual es posible volver a generar el archivo *boot.scr* mediante la herramienta *mkimage*. El archivo *boot.scr* contiene los parámetros que son enviados al *kernel*, dentro de los cuales se indica la resolución de la pantalla (por ejemplo 800x480MR-24@60, en donde el número 24 es el número de bits por pixel y 60 la frecuencia del display en Hertz), la cantidad de memoria volátil reservada (típicamente 12MB o 16MB), las direcciones en la tabla de memoria donde se ejecutan procesos subsecuentes (por ejemplo fatload mmc 0:1 0x80300000 *uImage*, lo cual indica que el *kernel* se ubica en la dirección 0x80300000 de la primera tarjeta multimedia encontrada), entre otras.

5.5 Segmentación de la aplicación según los casos de uso requeridos

Partiendo del marco de referencia especificado por (Castro & Schmidt, 2011), se crearon los formularios por los que habría de navegar el usuario. Estos fueron creados con la herramienta de diseño del *Qt4 C++ Creator*. En dicho documento, en

el apartado 2.1.3 se detallan los casos de uso funcionales que habrían de soportar la aplicación.

5.5.1 Casos de uso: Carga de Información de Votación (CU-06), Carga Padrón Electoral (CU-05), y Carga de Datos (CU-02).

En estos tres casos, la idea es transferirle al sistema embebido todos los datos que habrían de necesitarse una vez que entre en funcionamiento. Esto se logra mediante el protocolo de red SSH (*Secure Shell*), el cual brinda seguridad e integridad al transmitir la información de la computadora central hacia los nodos (entiéndase nodo como el sistema embebido). Allí se especifica los datos respectivos al número de mesa, los miembros que la integran, el tipo de elecciones, las fechas de apertura y cierre de las votaciones, así como la información del padrón parcial y completo. El diseño e implementación de estos casos de uso es realizado por los desarrolladores de software que integran el Proyecto Voto Electrónico del C.I.C., por lo cual se parte del hecho de que esta ya información se encuentra disponible en el embebido.

5.5.2 Caso de uso Comprobación del estado del sistema (CU-01)

Al iniciar el encendido del equipo embebido, la primera acción que deberá realizarse es corroborar que el sistema no presente alteraciones en sus componentes de hardware y software esenciales. Para lograr esto es posible recurrir a algunas herramientas de software tales como el *Hardware Listener* (**hwls**) y el *Hard Info* (**hardinfo**) que brindan información clasificada sobre las características de software y hardware presentes en el equipo en el momento de su ejecución. Cada una de ellas, internamente realiza un chequeo a través de otros servicios o archivos inherentes del sistema operativo. Por ejemplo, el archivo **/proc/cpuinfo** brinda información en texto plano sobre la familia que a la que pertenece el microprocesador, el nombre del modelo, la frecuencia de operación, el tamaño de la memoria *cache*, entre otros.

Debido a que el sistema operativo *Ångström* no dispone de ninguna de estas dos herramientas dentro de su lista de paquetes, resultó obligatorio realizar la comprobación manualmente. Los procesos de verificación se detallan en el Apéndice A2.

El diagrama de flujo general se muestra en la Figura 5.1

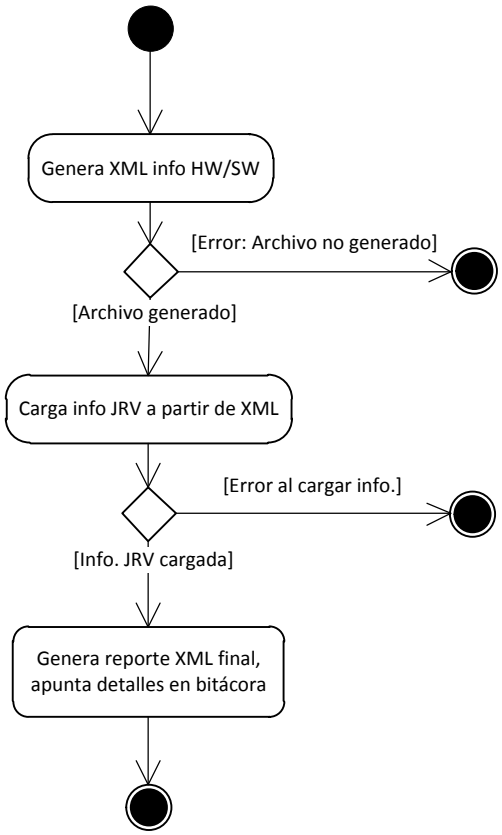


Figura 5.1 Diagrama de flujo para la generación de reporte de estado del sistema

En caso de que alguno de esos procesos de verificación devuelva un resultado negativo, se impide el avance hacia el siguiente caso de uso. De lo contrario se muestra el desglose parcial de estos detalles en una tabla tal y como se muestra en la Figura 5.2.

Figura 5.2 Captura de pantalla para el Caso de uso Comprobación del estado del sistema

A partir de este momento es posible apagar el equipo embebido o bien iniciar con el proceso de Autenticación de los miembros de mesa.

5.5.3 Caso de uso Autenticación de miembros de mesa (CU-03).

Una vez comprobado que la configuración de hardware y software del sistema es la correcta (a partir del caso de uso CU-01), se procede a autenticar los miembros de mesa presentes (caso de Uso CU-03).

Para ello, se muestra una ventana con un conjunto de pestañas, donde cada pestaña contiene la información prevista por el Tribunal para cada miembro de mesa. Por el momento, se han definido cinco roles:

a) Presidente Titular.
b) Secretaría.
c) Miembro Suplente.
d) Fiscal de la Junta Receptora de Votos (J.R.V.)
e) Fiscal General del Tribunal Supremo de Elecciones.

El procedimiento de autenticación puede representarse según se muestra en el diagrama de flujo de la Figura 5.3.

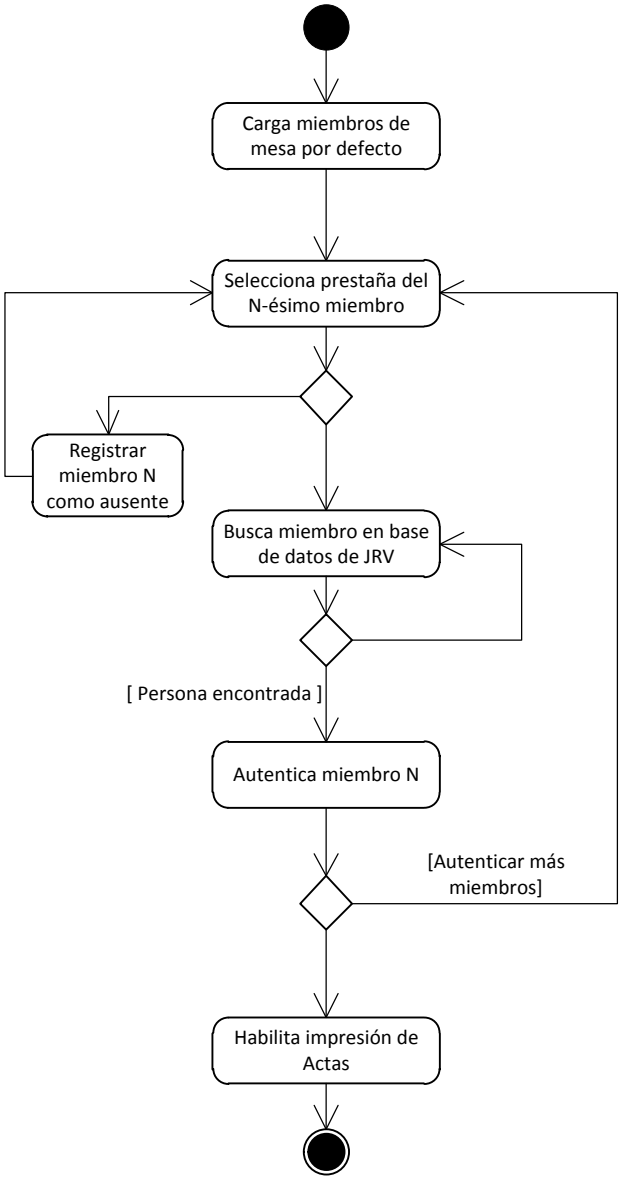


Figura 5.3 Diagrama de Flujo para la autenticación de Miembros de Mesa.

Como puede observarse, a pesar de tener el Tribunal previsto las personas que representan los Miembros de Mesa, es posible intercambiarlas en circunstancias inesperadas, siempre y cuando la autenticación sea válida.

En la Figura 5.4 se muestra una captura de pantalla del proceso de autenticación

Figura 5.4 Captura de pantalla para Caso de Uso de Autenticación de Miembros de Mesa.

Como puede observarse, además de la información personal del miembro de mesa (número de cédula de identidad, nombre, primer y segundo apellido), también se reserva un espacio para la fotografía. El sistema busca dicha imagen en la carpeta subcarpeta bajo el nombre de ***fotografias***. En caso de no disponerse de ella, se indica con el ícono de equis (ver figura X). Adicionalmente, en caso de que se haya decidido declarar a un miembro como ausente, en la pantalla se indica dicho mensaje (ver Figura 5.5).

Figura 5.5 Captura de pantalla indicando ausencia de Miembro de mesa.

Una vez que al menos un Miembro de mesa se haya autenticado, puede procederse a Imprimir el Acta de Apertura, o bien Apagar el equipo sin repercusión alguna la próxima vez que sea encendido.

5.5.4 Caso de uso Acta de Inicio y Acta de Cierre (CU-04 y CU-14).

El objetivo de estas etapa, es emitir un comprobante impreso que haga constar cuáles Miembros de Mesa han sido autenticados, que incluya la fecha, la hora, el número de mesa, la provincia, el cantón, y el distrito

Para efectos de este prototipo se recurrió al uso de la impresora de matriz de puntos, marca *Bixolon*, modelo *SRP-270D*, la cual tiene conexión directa mediante el puerto

USB. El ancho de la hoja de papel es de 76 mm, con un área efectiva de impresión de 65 mm, y con capacidad de abarcar como máximo 40 caracteres ASCII por línea.

Una alternativa para imprimir sin necesidad configurar previamente esta impresora, es que desde la aplicación se envíen los caracteres ASCII al puerto ***/dev/usb/lp0***. Sin embargo, algunas tareas como cortar la hoja de papel, detener las impresiones, hacer un salto de cierta cantidad de líneas, entre otros, requieren de ciertos comandos de control. En el Apéndice A3, se muestran los comandos utilizados.

Como parte de la secuencia de pasos que guían al usuario a conocer el estado actual del sistema, en la pantalla del embebido se imprime un mensaje de texto indicándole que el proceso de impresión se encuentra en curso. En la Figura 5.6 se muestra una captura de pantalla de muestra.

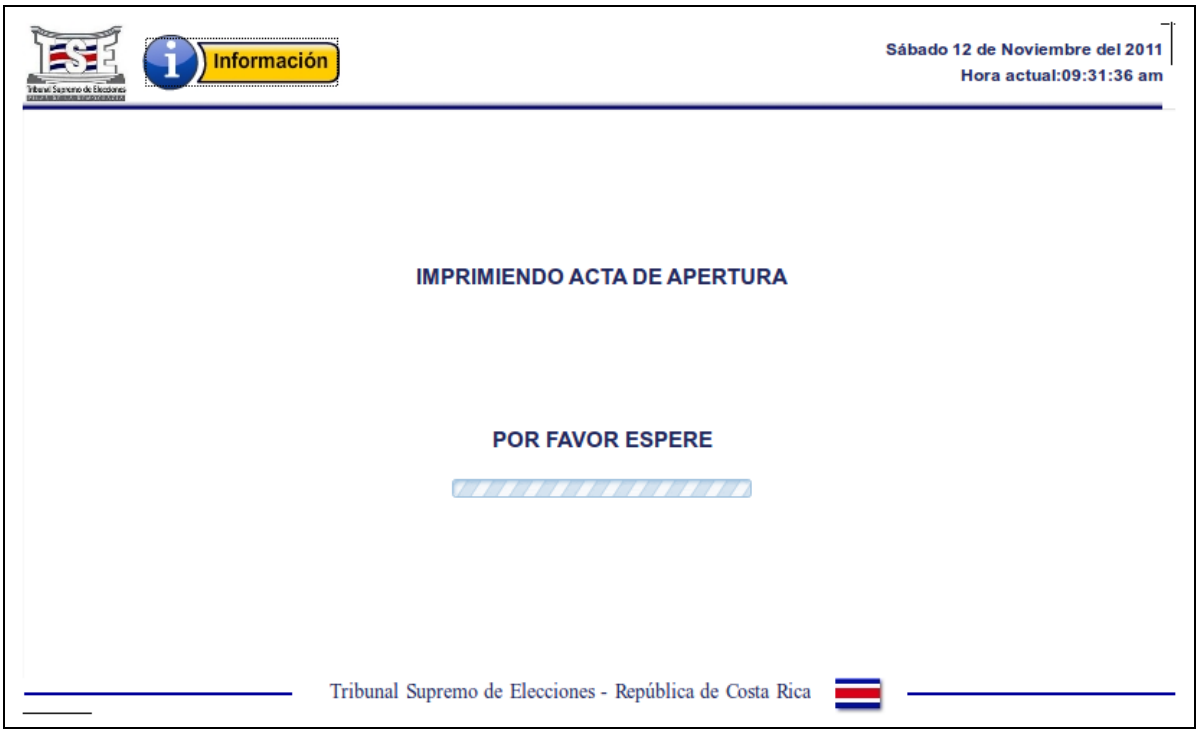


Figura 5.6. Captura de pantalla para el Caso de Uso Impresión de Acta de Apertura.

5.5.5 Caso de uso Apertura de Votación (CU-05).

En este proceso, el sistema se encuentra listo para dar inicio al proceso de identificación de electores, sin embargo deberá bloquearse mientras no sea la hora de inicio de votación. Para ello, se creó un reloj que indica al usuario que aún queda cierta cantidad de tiempo restante.

El diagrama de flujo que describe este proceso se muestra en la Figura 5.7.

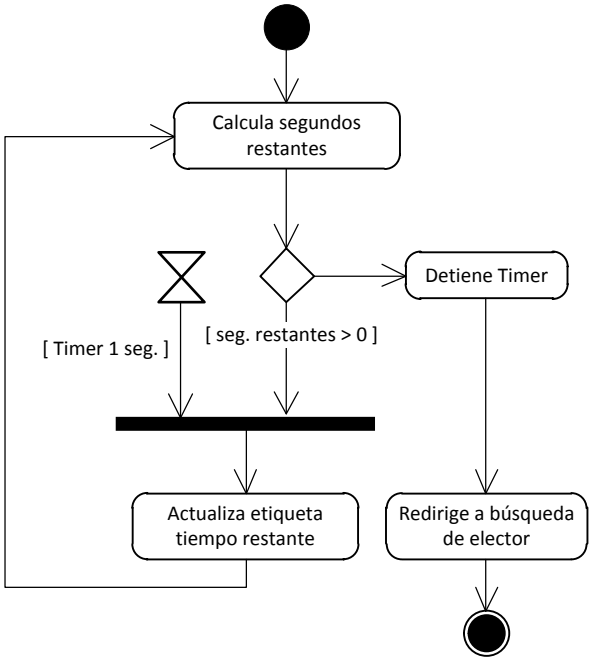


Figura 5.7 Diagrama de flujo para el Caso de uso Apertura de Votación.

En la Figura 5.8 además, se muestra una captura de pantalla del proceso de espera, indicando el tiempo restante para iniciar la búsqueda de electores. En caso de extenderse dicho tiempo a más de 24 horas, en la pantalla se le indica al usuario el número de días restantes y la fecha exacta de inicio de la votación.

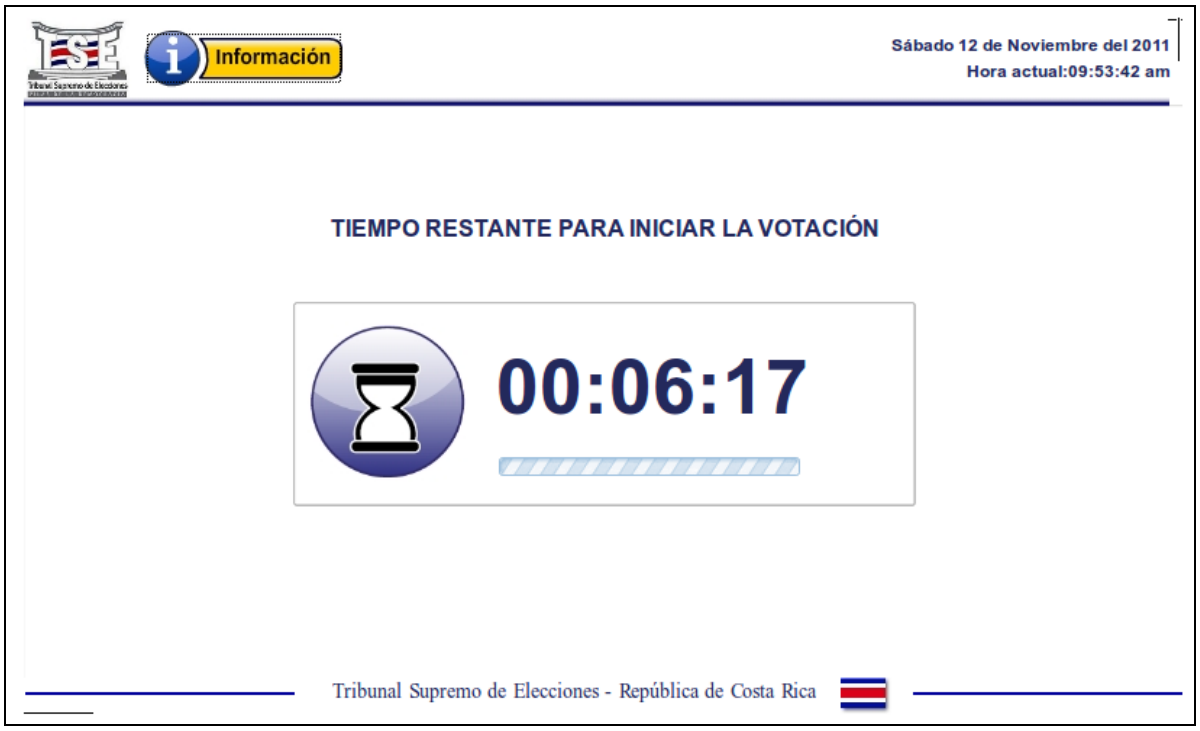


Figura 5.8 Captura de pantalla para el Caso de uso Apertura de Votación

5.5.6 Caso de uso Identificación del Elector (CU-06).

En esta etapa, se implementó el proceso de búsqueda previa del elector en la base de datos del sistema embebido.

La base de datos utilizada es *SQLITE3*, la cual es de código abierto y portátil (archivo único sin instalación previa). Esta base de datos contiene información de los electores que pertenecen a la Junta Receptora de Votos local, cuyos datos son asignados en la fase preparatoria del proceso de votación por los técnicos especializados del Tribunal. Se definieron dos bases de datos, una que contiene los

electores que fueron registrados como pertenecientes a la mesa actual, y una segunda base de datos que contiene todo el padrón nacional de Costa Rica.

El procedimiento consiste en que cuando un elector llega a la mesa, éste deberá presentar la cédula de identidad. La aplicación dispone de dos opciones para realizar la búsqueda en la base de datos: digitando manualmente el número de cédula o bien colocando la cédula de manera que sea reconocida por el lector de código de barras en formato *PDF417*. Para utilizar la primera opción, la aplicación dispone de un campo de texto en el que el número es digitado mediante el teclado externo. Para la segunda, el usuario primero debe presionar el botón etiquetado como “Leer mediante código de barras” y esperar a que la lectura del código se haya enviado del dispositivo externo hacia el embebido. Esta transmisión se realizada mediante el puerto serial RS232 que dispone el lector, y mediante un convertidor RS232-USB se emula el puerto RS232 en el sistema operativo del embebido.

Debido a que la cédula costarricense se encuentra cifrada bajo una llave exclusiva del Tribunal Supremo de Elecciones, la forma de descifrar la información del elector dependerá del uso de bibliotecas propiedad del Tribunal, para lo cual se ha autorizado a los desarrolladores de software de este Proyecto Voto Electrónico del CIC utilizar dichos recursos. Sin embargo, tal proceso aún queda pendiente, por lo que para efectos de este proyecto, se simula que los datos del elector se encuentran cifrados en formato *PDF417* sin requerir de ninguna llave para la protección de identidad.

La configuración y ubicación del puerto serial utilizado se muestra en el Apéndice A4.

El diagrama de flujo que representa los procesos realizados para este caso de uso se muestran en la Figura 5.9.

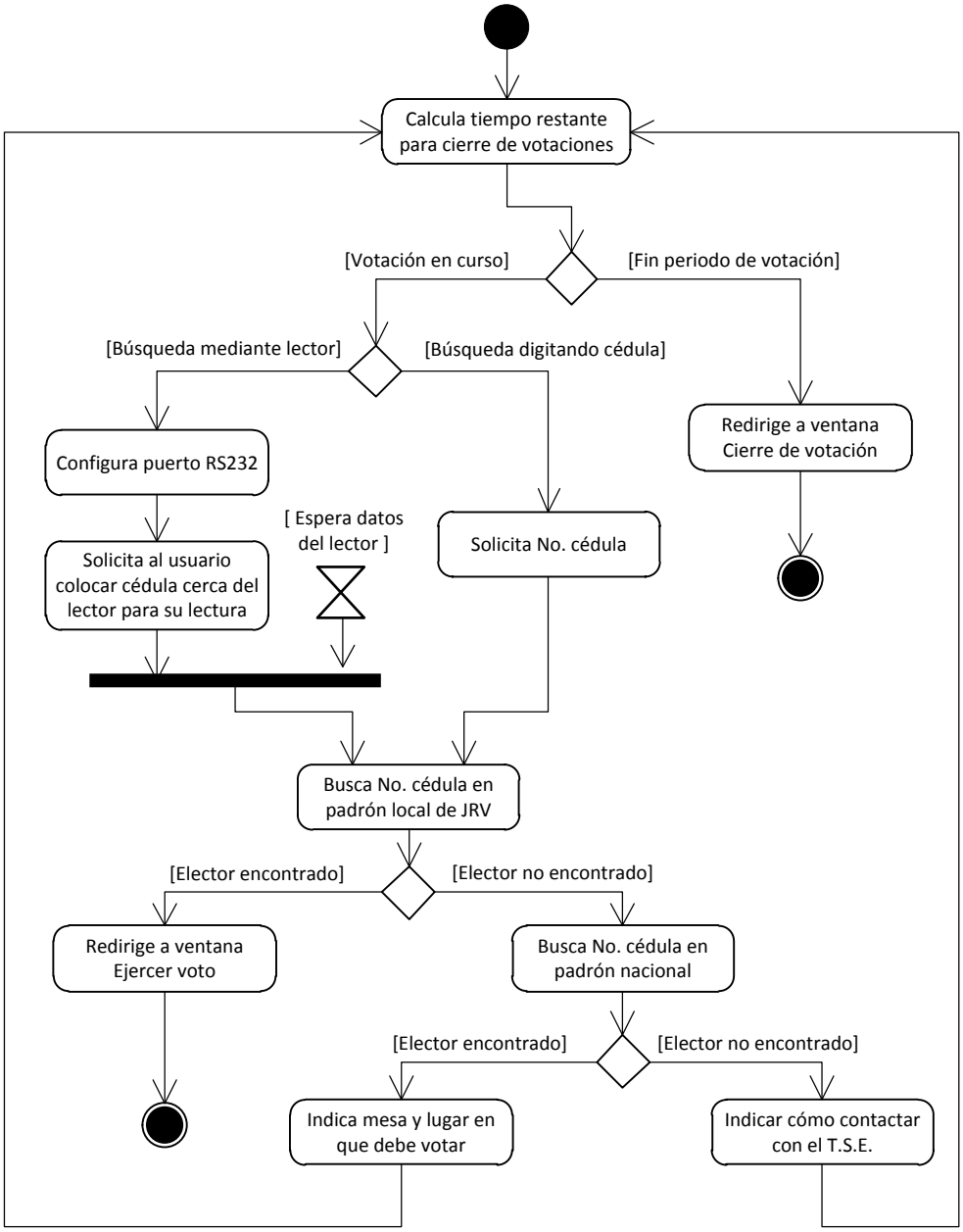


Figura 5.9 Diagrama de flujo para el Caso de uso Identificación del Elector (CU-06)

Debe recalcarse que el lenguaje *Qt C++* no dispone de bibliotecas que permitan la comunicación con el puerto serie RS232 del sistema. Una alternativa fue recurrir al uso de la biblioteca externa *QextSerialPort*, sin embargo esta no es compatible con la

versión más reciente de Qt 4.7; por lo que se decidió implementar una aplicación en lenguaje *Python* para configurar y capturar datos de dicho puerto enviándolos a un archivo temporal, para luego ser leídos por la aplicación principal.

En la Figura 5.10 se muestra una captura de pantalla de la ventana en la que se solicitan los datos del votante.

Figura 5.10 Captura de pantalla para el Caso de uso Identificación del elector (CU-06)

5.5.7 Caso de uso Ejercer voto (CU-07).

Una vez identificado el elector, se muestra en pantalla una ventana en donde se desglosan los datos personales de la persona, su domicilio electoral, y el número de Junta Receptora de Votos en la que se encuentra empadronado. Los datos del

domicilio electoral (provincia, cantón y distrito) son tomados de la base de datos en formato *SQLITE3* nombrada como ***codigos_electorales.sqlite3.***

En esta fase, el Miembro de mesa tiene la opción de confirmar de manera irreversible que el elector seleccionado va a proceder a emitir el voto, o bien puede cancelar la operación y seleccionar un elector diferente. Una vez confirmado, la aplicación actualiza la base de datos del padrón local (***padron_local_jrv.sqlite3***) de manera que el elector quede registrado como “ya votó”.

En caso de que el elector haya emitido su voto y se intente repetir el proceso, el sistema le indica al Miembro de mesa tal situación, indicándole la fecha y hora en la que el elector emitió su voto.

El diagrama de flujo que representa este proceso se muestra en la Figura 5.11.

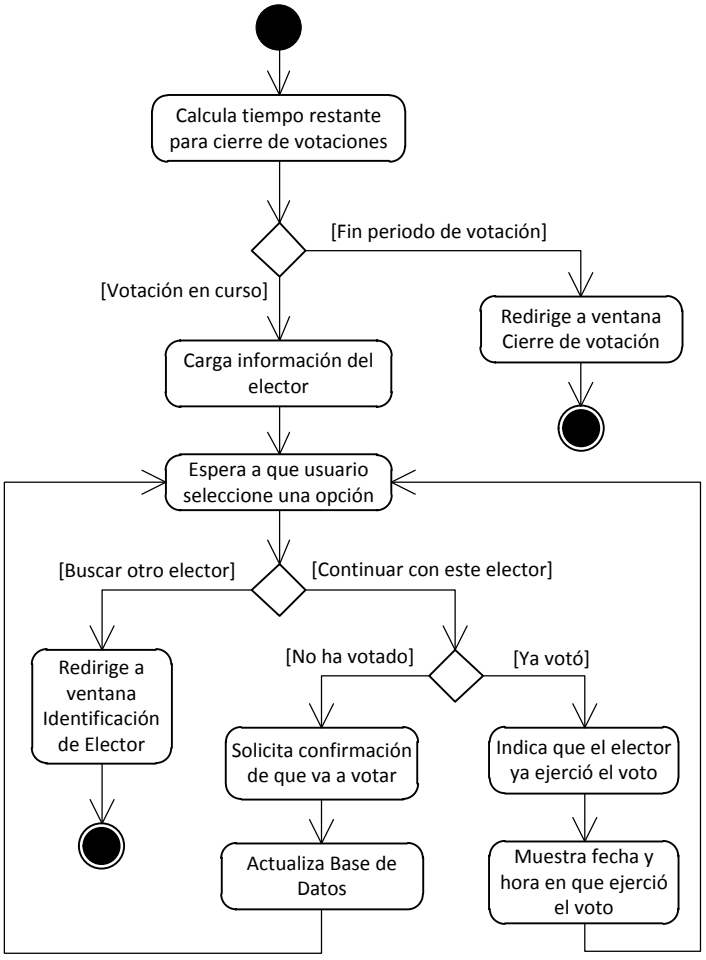


Figura 5.11 Diagrama de flujo para el Caso de uso Ejercer Voto (CU-07)

En la Figura 5.12 se muestra una captura de pantalla de la ventana en la que se solicita confirmar al Miembro de mesa que el elector va proceder el voto.

Figura 5.12 Captura de pantalla para el Caso de uso Ejercer voto (CU-07)

5.5.8 Caso de uso Información de mesa y estado de la votación (CU-08 y CU-12).

En todo momento se debió ubicar en la aplicación la opción de consultar la información general de la mesa. En esta se incluyen los datos que se muestran en el Apéndice A5, junto con su origen.

Esta información se desglosó en una ventana emergente, similar a la que se muestra en la Figura 5.13.

Figura 5.13 Captura de pantalla de información de la mesa de votación (CU-08 y CU-12)

5.5.9 Caso de uso Cierre de votación (CU-12).

Al vencer el periodo de votación definido en la fase preparatoria, el sistema reenvía al usuario a la ventana en donde se le indica que el tiempo ha expirado y que debe proceder a realizar el Acta de Cierre.

En la Figura 5.14 se muestra una captura de pantalla acerca de esta notificación.

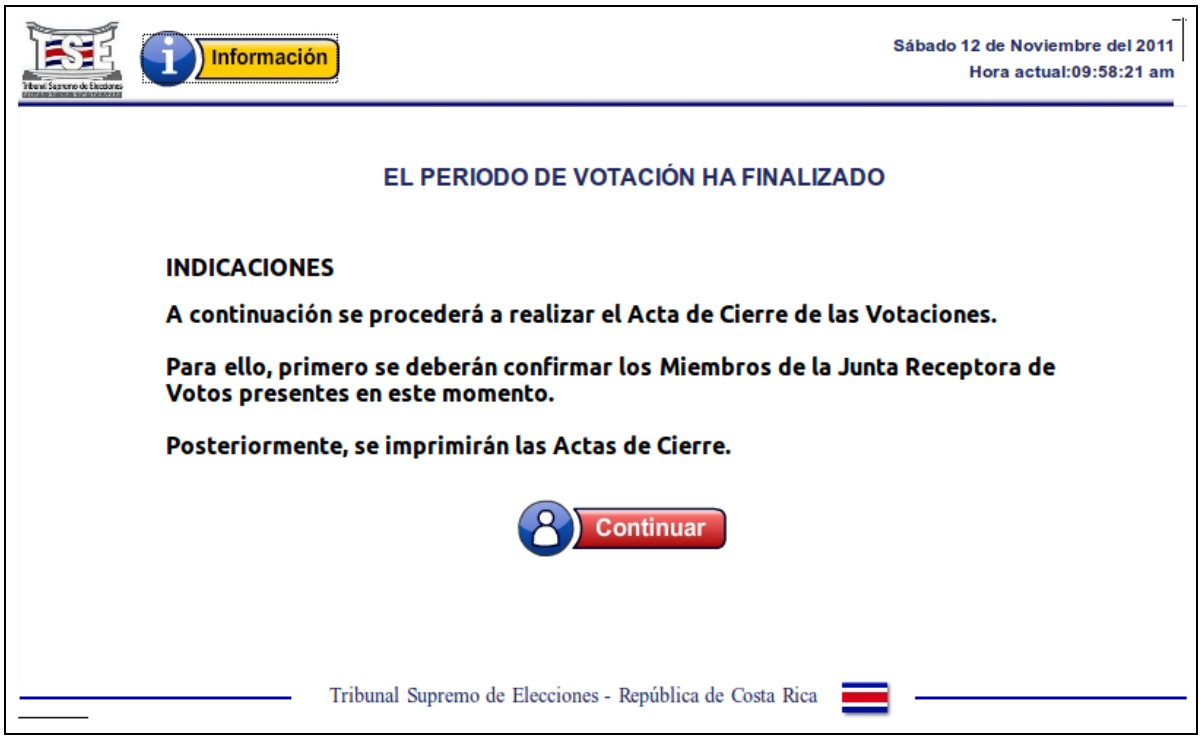


Figura 5.14 Captura de pantalla para el Cierre de votación (CU-13)

5.5.10 Ventanas complementarias (CU-12).

Aparte de las ventanas correspondientes a cada caso de uso, se implementó otra serie de ventanas intermedias con el objetivo de orientar aún más al usuario del embebido. Estas son:

a) Ventana de inicialización –*bootsplash*- del sistema embebido

El objetivo de esta ventana es brindar al usuario una breve información acerca del Proyecto Voto Electrónico. En la Figura 5.15 se muestra una captura de pantalla de

la ventana implementada. Una vez finalizada la carga del sistema operativo, esta desaparece e inicializa la aplicación principal.

Figura 5.15 Ventana de inicialización del sistema embebido.

b) Ventana de inicialización –*bootscreen*- de la aplicación principal.

El objetivo de esta ventana es brindar al usuario una breve información acerca de la aplicación que está siendo cargada. En la Figura 5.16 se muestra una captura de pantalla de la ventana implementada. Una vez finalizada la carga de la aplicación, esta se oculta y se procede con el primer Caso de uso: Verificación de estado del sistema (CU-01).

Figura 5.16 Ventana de inicialización de la aplicación principal.

5.6 Actualización de la hora del sistema embebido

Debido a que el sistema embebido debe tener una fecha y hora que debe estar sincronizada con la del Tribunal, debió asegurarse que esta información no se altere una vez retirada la fuente de alimentación del circuito. Para resolver este inconveniente, se utilizó un módulo de Reloj en Tiempo Real (RTC, por sus siglas en inglés). Este módulo se alimenta de una batería de litio de 3 V para almacenar en su memoria *RAM* los datos de la fecha y hora. La forma de comunicarse con otros dispositivos es mediante el protocolo I^2C, por lo que se debió buscar una alterativa sencilla para transferir estos datos hacia el sistema operativo del embebido.

Una primera alternativa era modificar el *kernel* de manera que durante el proceso de arranque, los controladores de hardware transmitieran esta información de forma

directa y automática al sistema Linux. Sin embargo, la tarea de recompilar el *kernel* de requiere de un profundo conocimiento dicha arquitectura de software, por lo que se optó por una segunda alternativa.

La segunda alternativa, consistía en establecer una comunicación entre capa de aplicación del sistema operativo y la capa de hardware, a través del espacio de usuario (*user space*), lo cual permitió liberarse de la necesidad de crear controladores de hardware. Sin embargo, la actualización requiere que cada vez que inicia el sistema operativo, se deba llamar al subproceso que efectúa dicha actualización a través de un servicio de Linux (*daemon*).

La implementación de la segunda alternativa, consistió en utilizar una herramienta de software llamada ***i2ctools***, el cual es un paquete de libre distribución, y fue comprobado su funcionamiento tanto en la versión cross-compilada *Ångström* como en la de *Ubuntu*. Más información sobre el uso de esta herramienta se puede encontrar en el Apéndice A6.

Capítulo 6: Análisis de Resultados

Una de las metas de este proyecto es contar con un prototipo que entre muchas de sus características no funcionales, sea capaz de reducir significativamente el tiempo de encendido del sistema embebido. A pesar de que la solución seleccionada de momento opera sin muchas modificaciones al sistema operativo *Ångström*, la idea es trasladarse a la distribución *Ubuntu 11.04* o superiores en un futuro. En la Tabla 6.1 se muestra una comparación entre el tiempo de encendido aproximado de cada distribución. Esta medición se realizó desde el momento en que arranca el *bootloader* inicia la descompresión del kernel hasta el momento en que el sistema operativo entre en el *runlevel* 3 (consola de Linux en la que solicita el nombre de usuario y contraseña para ingresar).

Tabla 6.1 Comparación de tiempos de encendido aproximado entre sistemas operativos *Ångström* y *Ubuntu11.04*

Sistema operativo	**Tiempo de encendido aproximado**
Ångström	2 minutos
Ubuntu 11.04	45 segundos

Como puede observarse en dicha tabla, la diferencia es bastante significativa (el tiempo de un sistema respecto al otro se reduce en un 38% aproximadamente).

Otro aspecto considerado es la latencia que tiene la aplicación principal (programada mediante *Qt 4 C*++) al momento de cargar las fotografías de los miembros de mesa y de los electores. A pesar de que el tamaño promedio de cada fotografía de 130x140 pixeles es de 12kB, se observó que hubo un pequeño retardo (menor a 1 s) durante esta transición. Esto se debe a que el microprocesador debe cargar un conjunto de bytes en el *framebuffer* (memoria reservada para el video) y desplegarlos en la

pantalla de manera dinámica. Un comportamiento similar ocurre al momento de ejecutarse por primera vez dicha aplicación; en este caso la latencia es mayor (aproximadamente 1.5 s) puesto que microprocesador requiere de mayor tiempo para completar el procesamiento de los datos necesarios.

En cuanto al consumo de potencia, se realizaron tres mediciones en distintos escenarios. En un primer caso, se midió el consumo de potencia para el *BeagleBoard-xM* funcionando aislado de los demás módulos; en el segundo caso se realizó la misma prueba pero esta vez para el caso del módulo *ULCD7 LITE* solamente; y finalmente, se midió la corriente total consumida de todo el sistema embebido, incluyendo las memorias de respaldo (*memory stick*), los cuatro puertos USB funcionando a la vez (uno para el teclado, otro para la impresora, el tercero para el convertidor USB-RS232 que se conecta con el lector de cédulas y el cuarto con la *memory stick*). Todo el sistema se alimentó con una fuente de 5V. Estas mediciones se muestran en la Tabla 6.2.

Tabla 6.2 Mediciones de corriente y consumo de potencia del sistema embebido

Dispositivo (s)	**Consumo de corriente (mA)**	**Potencia consumida (mW)**
BeagleBoard-xM	470	2350
Módulo ULCD7 LITE	1670	8350
Sistema embebido completo (BeagleBoard-xM, Módulo ULCD7 LITE, 4 puertos USB, memory stick)	2680	13400

Otro aspecto considerado fue la limitante que presenta el touchscreen resistivo que viene instalado sobre el LCD del módulo *ULCD7 LITE*. El inconveniente observado de esta tecnología es que al presionar uno de los botones que se muestran en la aplicación (por ejemplo con el dedo índice), la respuesta es prácticamente nula. Es decir, el hecho de ejercer presión con los dedos en un área relativamente grande de

la pantalla no es distinguible por el emulador del *touchscreen*. Esto se debe a que internamente se están presionando múltiples puntos en la lámina, por lo que las coordenadas XY percibidas por el controlador también son múltiples, y a pesar de que utiliza una probabilidad de acierto, esta configuración no se puede variar directamente desde el sistema operativo.

Capítulo 7: Conclusiones y recomendaciones

7.1 Conclusiones

1. Mediante este sistema embebido se logró cumplir los requerimientos funcionales especificados en el Proyecto Voto Electrónico del C.I.C.

2. Se logró completar la implementación del padrón electrónico con fotografía, optimizando el proceso de búsqueda de electores.

3. Los procesos de apertura, verificación de electores y cierre de mesa fueron optimizados mediante la reducción de recursos del sistema.

4. Se validó que el sistema embebido no es técnicamente viable para darle continuidad al Proyecto Voto Electrónico.

7.2 Recomendaciones

1. Se recomienda utilizar un sistema embebido que cuente con las siguientes características:

    a) Cuente con un Reloj de Tiempo Real
    b) La pantalla táctil sea capacitiva con el de facilitar la interacción entre el embebido y el usuario.
    c) Soporte más de un sistema operativo, en especial la distribución *Ubuntu*.

8 Bibliografía

[1] *BeagleBoard-XM with NEC LCD display.* (setiembre del 4 de 2011). Recuperado el 12 de setiembre del 2011, de http://bb-lvds.blogspot.com/2011/04/beagleboard-xm-with-nec-lcd-display.html

[2] *FlatLink TRANSMITTER (SN75LVDS83B).* (setiembre del 2011). Recuperado el 12 de setiembre del 2011, de http://www.ti.com/product/sn75lvds83b

[3] Analog Devices, Inc. (febrero del 2011). *New Touch-Screen Controllers Offer Robust Sensing for Portable Displays*. Recuperado el 30 de setiembre del 2011, de http://www.analog.com/library/analogDialogue/archives/44-02/touch_screen.html

[4] Angstrom Distribution. (2009). *Angstrom standalone toolchains and SDKs*. Recuperado el 2 de setiembre del 2011, de http://www.angstrom-distribution.org/toolchains/

[5] Beagle Board. (2011). *BeagleBoard Home Page*. Recuperado el 31 de agosto del 2011, de http://www.beagleboard.org

[6] Beagle Board. (1 de junio del 2011). *BeagleBoard-xM Product Details*. Recuperado el 15 de agosto del 2011, de http://beagleboard.org/hardware-xM

[7] BeagleBoard Toys Co. (2 de octubre del 2011). *ULCD7 LITE*. Recuperado el 20 de octubre del 2011, de http://beagleboardtoys.com/wiki/ULCD7/index.php?title=Main_Page#Resources

[8] Creative Commons. (2011). *About The Licenses*. Recuperado el 1 de noviembre del 2011, de http://creativecommons.org/licenses/

[9] Digi-Key Corporation. (28 de julio del 2008). *USB-powered Beagle Board from Digi-Key Unleashes Community Development with Laptop-like Performance and Expansion for $149*. Recuperado el 25 de octubre del 2011, de http://dkc1.digikey.com/us/en/mkt/Press/Beagle_Board.html

[10] Digi-Key Corporation. (13 de mayo del 2009). *Digi-Key Announces New Open Source BeagleBoard Development Board*. Recuperado el 25 de octubre del 2011, de http://dkc1.digikey.com/us/en/mkt/Press/BeagleBoardC.html

[11] eLinux.org. (14 de octubre del 2011). *BeagleBoardUbuntu.* Recuperado el 26 de julio del 2011, de http://elinux.org/BeagleBoardUbuntu

[12] Intel Corporation. (s.f.). *Intel Embedded*. Recuperado el 31 de agosto del 2011, de http://www.intel.com/p/en_US/embedded/applications/medical

[13] Litt, S. (2002). *Adding a Directory to the Path*. Recuperado el 6 de setiembre del 2011, de http://www.troubleshooters.com/linux/prepostpath.htm

[14] Maxim Integrated Products. (2008). I^2C Real-Time Clock (DS1307). California. Obtenido de http://datasheets.maxim-ic.com/en/ds/DS1307.pdf

[15] NEC LCD Technologies, Ltd. (2010). TFT Color LCD Module (NL10276BC20-10) LVDS Interface. Obtenido de http://www.spectrah.com/product/lcd_panel/nec_lcd_panel/nec-NL10276BC20-10.pdf

[16] Nokia Corporation. (2011). *Customer Stories*. Recuperado el 2 de noviembre del 2011, de http://qt.nokia.com/qt-in-use/story/customer

[17] Nokia Corporation. (2011). *Download Qt, the cross-platform application framework* . Recuperado el 3 de agosto del 2011, de http://qt.nokia.com/downloads/

[18] Nokia Corporation. (2011). *Qt Products*. Recuperado el 31 de agosto del 2011, de http://qt.nokia.com/products

[19] Nokia Corporation. (4 de mayo del 2011). *Qt Source Index*. Recuperado el 2 de agosto del 2011, de http://get.qt.nokia.com/qt/source/

[20] Osier-Mixon, J. (14 de diciembre del 2010). *Booting Linux on the BeagleBoard-xM.* Recuperado el 6 de agosto del 2011, de http://www.ibm.com/developerworks/linux/library/l-beagleboard-xm

Apéndices

A.1 Instalación y configuración del ambiente de desarrollo *Qt 4.7 C++.*

El *SDK de Qt 4.7 C++* se encuentra disponible en *http://qt.nokia.com/downloads.* Una vez descargado el software, deberá habilitarse los permisos de ejecución mediante el comando ***chmod***

$ chmod u+x Qt_SDK_Lin64_offline_v1_1_3_en.run

y ejecutar el instalador con las rutas de instalación por defecto:

$ ./Qt_SDK_Lin64_offline_v1_1_3_en.run

Al finalizar la instalación, el sistema se encuentra disponible para crear aplicaciones para la misma arquitectura del *host* (entiéndase como tal a la computadora que hospeda el ambiente de desarrollo *Qt C++*).

Adicionalmente, el cross-compilador que se utilizó es el *toolchain* de *Ångström*, el cual puede descargase de *http://www.angstrom-distribution.org/toolchains/.*

Una vez descargado descomprimirlo e instalarlo:

$ sudo tar -xvj -C / -f Angstrom-2011.03-x86_64-linux-armv7a-linux-gnueabi-toolchain-qte-4.6.3.tar.bz2

Ahora, para poder crear aplicaciones que sean ejecutables en el embebido, los desarrolladores de Nokia ponen a disposición de la comunidad una herramienta llamada *Qt Everywhere*, enfocada exclusivamente para crear aplicaciones ejecutables en otras arquitecturas de hardware. En (Nokia Corporation, 2011) puede descargarse el archivo ***qt-everywhere-opensource-src-4.6.2.tar.gz*** utilizado. Puesto que el microprocesador que el *BeagleBoard-xM* utiliza es el *DM3730* de

Texas Instruments, perteneciente a la familia *ARM 7a Core*, el *framework* que se utilizó es:

a) Para la versión Ubuntu de 32 bits:
Ångström-2011.03-i686-linux-armv7a-linux-gnueabi-toolchain-qte-4.6.3.tar.bz2

b) Para la versión Ubuntu de 64 bits:
Ångström-2011.03-x86_64-linux-armv7a-linux-gnueabi-toolchain-qte-4.6.3.tar.bz2

Seguidamente se descomprime este archive según sea el caso:

$ tar -xvzf qt-everywhere-opensource-src-4.6.2.tar.gz

con lo que se obtiene una carpeta llamada ***qt-everywhere-opensource-src-4.6.2*** en la misma carpeta de donde se descargó. El siguiente paso es crear un archivo de configuración dentro de esta carpeta, que va permitir que a la hora de instalar este *framework*, el sistema *host* pueda reconocer el cross-compilador de *Ångström* recién instalado, y poder así utilizarlo en el *Qt Creator*. Por tanto, dentro de esta carpeta ingresar a la subcarpeta ***mkspecs/qws/*** y duplicar la carpeta llamada ***linux-arm-g++*** ahora bajo el nombre de ***linux-dm3730-g++***. Allí dentro, abrir con un editor de texto el archivo *qmake.conf* y asegurarse de que los contenidos sean los siguientes:

```
include(../../common/g++.conf)
include(../../common/linux.conf)
include(../../common/qws.conf)

QMAKE_CFLAGS_RELEASE =   -O3 -march=armv7-a -mtune=cortex-a8 -
mfpu=neon -mfloat-abi=softfp

QMAKE_CXXFLAGS_RELEASE = -O3 -march=armv7-a -mtune=cortex-a8 -
mfpu=neon -mfloat-abi=softfp

QMAKE_CC = /usr/local/angstrom/arm/arm-angstrom-linux-gnueabi/bin/gcc
QMAKE_CXX = /usr/local/angstrom/arm/arm-angstrom-linux-gnueabi/bin/g++
QMAKE_LINK = /usr/local/angstrom/arm/arm-angstrom-linux-gnueabi/bin/g++
QMAKE_LINK_SHLIB = /usr/local/angstrom/arm/arm-angstrom-linux-
gnueabi/bin/g++

QMAKE_AR = /usr/local/angstrom/arm/arm-angstrom-linux-gnueabi/bin/ar cqs
QMAKE_OBJCOPY = /usr/local/angstrom/arm/arm-angstrom-linux-
gnueabi/bin/objcopy
QMAKE_STRIP = /usr/local/angstrom/arm/arm-angstrom-linux-gnueabi/bin/strip

load(qt_config)
```

Posteriormente, se procede a instalar el *framework* mediante el siguiente comando:

$ ./configure -opensource -confirm-license -no-qt3support -embedded arm -xplatform qws/linux-dm3730-g++ -platform /qws/linux-x86-g++ -no-xrandr -fast -no-cups -no-largefile -no-3dnow -no-mmx -little-endian -qt-mouse-tslib

$ make

$ make install

El siguiente paso consiste en configurar el *Qt Creator* para que reconozca el cross-compilador junto con el *framework* recién instalados. Primero deberá abrirse el programa e ir a la barra de *Tools*, y seleccionar *Options*. En la Figura A.1.1 se muestra una imagen de la ventana que debe abrirse.

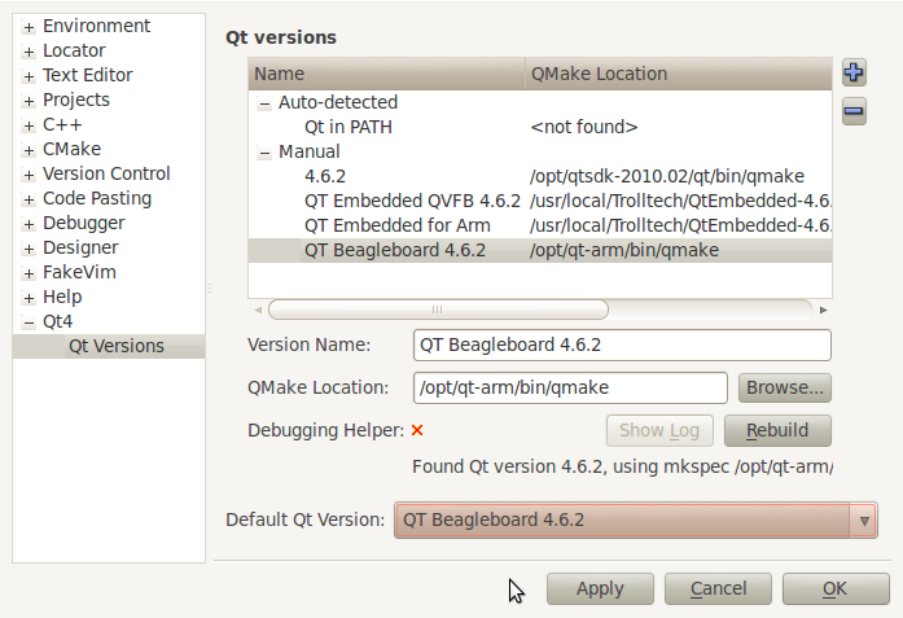


Figura A.1.1 Imagen de la ventana de opciones de *Qt Creator*

En el menú de la izquierda, se debe seleccionar la opción *Toolchains*. Estando allí habrá que indicar la ruta del framework recién instalado, que en este caso es ***/usr/local/Trolltech/Qt-Everywhere-4.7.3-arm/bin***

Como último paso es importante asegurarse que las variables de ambiente del *toolchain* y el *framework* estén presenten. Para ello, deben de modificarse los siguientes scripts según corresponda; en la Tabla A.1 se muestran los detalles (Litt, 2002).

Tabla A.1.1 Ubicación de los scripts de cada usuario que definen las rutas de los archivos de ejecución

Grupo de usuarios	**Script a modificar**
Un usuario en particular	$HOME/.bash_profile
Todos los usuarios excepto usuario root	/etc/profile
Root	/root/.bashrc_profile

El script debe contener el siguiente código:

$PATH=$PATH:/usr/local/angstrom/arm/arm-linux-gnueabi/bin/

export PATH

esto indica al sistema operativo, que agregue a la lista de rutas donde se encuentra los archivos de ejecución y otros scripts, las nuevas rutas indicadas cada una separada por dos puntos “:” cada vez que el usuario inicie sesión.

A.2 Procedimiento de verificación del estado del sistema embebido

Tabla A.2.1 Descripción de procesos de verificación de estado del sistema

Descripción del proceso	**Procedimiento de verificación**
¿Es esta la distribución del sistema operativo Linux, kernel 3.0.4+ y arquitectura ARM7?	Uso de comando **$ uname –a**
¿Existe conexión con el módulo ULCD7 LITE?	Utilizar la función interna de la aplicación principal ***BoolExisteConexionLCD()*** el cual intenta comunicarse con el LCD mediante el protocolo I^2C leyendo los registros ubicados en la dirección base 0x40 .
¿Está presente el teclado USB?	Uso de comando **$ lsusb**
¿Está presente la impresora térmica por USB?	Verifica presencia de ***/dev/usb/lp0***
¿Está presente el lector RS232?	Utiliza script programado en lenguaje *Python* llamado Verifica_RS232.py para corroborar que el puerto serial esté disponible
La integridad de ciertos archivos del programa permanece intacta	Utilizar hash de tipo MD5 aplicado sobre los archivos de la carpeta
¿Cuántas veces ha sido encendido el equipo?	Leer registro bitácora en formato XML.
¿Existe un Acta de Apertura registrada?	Leer registro bitácora en formato XML.
¿Existe información de la JRV?	Leer archivo de carga XML

Al ejecutarse todos estos procesos, se crea un nuevo reporte en formato XML, cuyo nombre incluye la fecha y hora en la que se realizó (reporte_encendido_dd_MM_yyyy_HH_mm_ss.xml).

A.3 Comandos utilizados para manejo de impresión

Tabla A.3.1 Comandos especiales para uso de la impresora

Descripción	**Comandos**	**Código hexadecimal**
Inicialización de la impresora	ESC, @	0x1B, 0x40
Salto de línea	NL	0x0A
Retorno de línea	CR	0x0D
Corte de papel	ESC, @, GS, V, 0	0x1B, 0x40, 0x1D, 0x56, 0x30

A.4 Configuración y ubicación del puerto serial RS232.

Tabla A.4.1 Configuración y ubicación del puerto serial RS232

Descripción	**Valor**
Ubicación del puerto en sistema operativo Ubuntu	*/dev/ttyUSB0*
Baudrate	9600
Bits de datos	8
Paridad	Ninguna
Bits de parada	1
Control de flujo	Ninguno

A.5 Origen de datos relativos a la información general de la mesa.

Tabla A.5.1 Origen de datos relativos a la información general de la mesa

Dato	Origen
Número de mesa	Archivo XML de carga (***info_precarga.xml***)
Provincia	
Cantón	
Distrito electoral	
Hora actual del sistema	Memoria RAM (*datetime* del sistema operativo)
Hora y fecha de apertura de votaciones	Archivo XML de carga (***info_precarga.xml***)
Hora y fecha de cierre de votaciones	
Total de electores pertenecientes a la mesa	Base de datos padrón JRV (***padron_local_jrv.sqlite3***)
Total de electores que han ejercido la votación	
Total de electores pendientes de ejercer el voto	

A.6 Uso del paquete de herramientas *i2ctools*

Esta herramienta está compuesta de varias sub-aplicaciones, las cuales son: **i2cdetect** (detecta dispositivos conectados al bus I^2C), **i2cget** (obtiene contenidos de los registros de un dispositivo I^2C) e **i2cset** (establece valores en los registros de un dispositivo I^2C).

Dos de las herramientas utilizadas fueron: **i2cget** e **i2cset**. La primera permitió leer cada uno de los bytes que se encuentran en el RTC, en donde la fecha y hora se subdividen en año, día, mes, día de la semana, hora, minuto, segundo, hora en formato AM/PM. Estos datos son representados en formato *BCD*. Por ejemplo, al leer el registro que corresponde al día 31 de diciembre del año 2011, el byte que representa al día es devuelto como 0x31 (b00110001), el que representa al mes es 0x12 (b00010010) y el que representa al año es 0x11 (b00010001). En el caso de los

años es importante notar que el RTC sólo devuelve los últimos dos dígitos, funcionando en el rango de años del 2000 al 2099.

Así pues, se creó una aplicación ejecutable por la consola de comandos, que es la encargada de leer los registros del RTC mediante la herramienta **i2cget**. Los comandos que se le envían a esta última incluyen: el número del bus de datos I^2C, la dirección del dispositivo de interés y la dirección del registro ubicado en memoria. En la Figura A.6.1 se ejemplifica el método para realizar la lectura de un registro mediante esta herramienta.

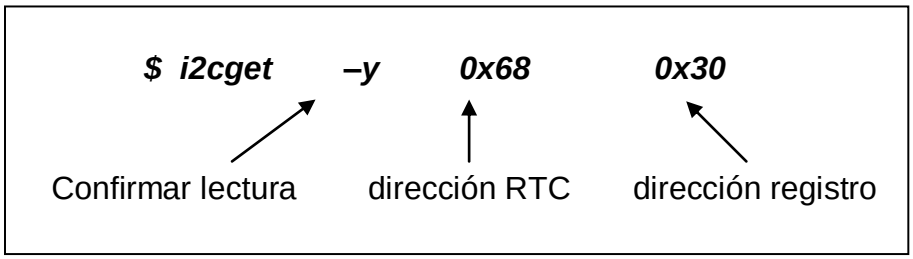


Figura A.6.1 Ejemplo de lectura de un registro del RTC

Para el caso particular de la *Beagleboard-xM*, el microprocesador dispone de tres buses I^2C, de los cuales el bus número 2 es el único que está disponible al usuario, los demás son reservados por el kernel para otras tareas.

Para realizar la lectura completa de la fecha y hora se creó una clase llamada ***RTC***. Esta contiene tres funciones las cuales son: **SetRTCDateTimeAlEmbebido()** y **SetGetVal()** y **SetDateTimeAlRTC()**. La primera de ellas tiene como objetivo leer la información del RTC y a partir de ellos actualizar la fecha y hora del sistema operativo del embebido. La segunda función es llamada por la primera y dependiendo de los parámetros que con que se invoque, hace la escritura o lectura directa en cada uno de los registros del RTC. La tercera función toma los parámetros enviados a la aplicación y a partir de ellos actualiza los datos del RTC. En la Tabla A.6.1 se muestran las direcciones de los registros del módulo de reloj en tiempo real,

su nombre definido en la clase, la descripción y el comando con que se invoca desde la *Shell* de Linux.

Tabla A.6.1 Constantes de la clase, registros en memoria del RTC

Constante definida en la clase	Dirección	Descripción
RTC_I2C_DEVICE_BUS	0x02	Dirección del bus I^2C en BeagleBoard-XM
RTC_I2C_DEVICE_ADDRESS	0x68	Dirección del RTC conectado al bus I^2C
RTC_SEGUNDOS_ADD	0x00	Dirección del registro para segundos
RTC_MINUTOS_ADD	0x01	Dirección del registro para minutos
RTC_HORAS_24_ADD	0x02	Dirección del registro para indicar si la hora del RTC está en formato de 24 horas
RTC_DIA_SEMANAL_ADD	0x03	Dirección del registro para indicar el día de la semana (0x01 corresponde a Domingo, 0x02 a Lunes, y así sucesivamente)
RTC_DIA_ADD	0x04	Dirección del registro para indicar el día del mes (varía de 0x01 a 0x31)
RTC_MES_ADD	0x05	Dirección del registro para indicar el mes (0x01 para Enero)
RTC_ANO_ADD	0x06	Dirección del registro para indicar los dos últimos dígitos del año (varía de 0x00 a 0x99)

Una vez conocidas todas las direcciones necesarias, la aplicación ejecuta el comando de ***i2cget*** respectivo mediante la función **QProcess::execute();**.

Para el caso complementario, es decir, cuando el sistema operativo debe actualizar la fecha del módulo RTC, el procedimiento es muy similar al mencionado previamente, y consiste básicamente en enviar un parámetro adicional a la hora de invocar esta vez la herramienta ***i2cset***. En este caso, el último parámetro sería el valor del byte que va a ser almacenado en el RTC. En la Figura A.6.2 se ejemplifica

el método para realizar la escritura en un registro del RTC mediante esta herramienta ubicada en el *userspace*.

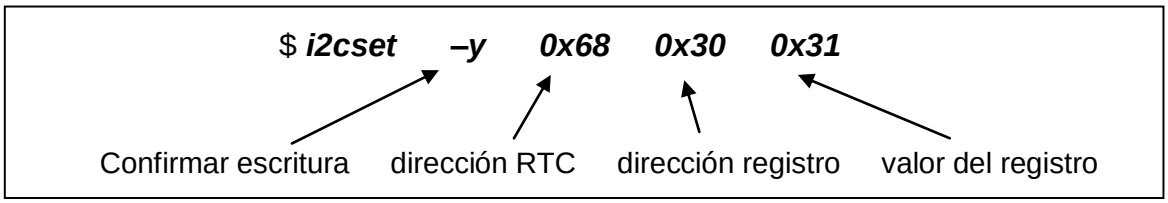


Figura A.6.2 Ejemplo de escritura en un registro del RTC

Anexos

TEC | Tecnológico de Costa Rica

Elaborado el 29 de junio, 2011

Informe de avance. Arquitectura de Software para un sistema de voto electrónico asistido por computadora.

Elaborado por

Jeff Schimdt Peralta[1] - jschmidtcr@gmail.com
Jose Castro[2] - jose.r.castro@gmail.com

Resumen: El voto electrónico es actualmente una opción que está evaluando el Tribunal Supremo de Elecciones para atender la necesidad diagnosticada de incrementar la cantidad de personas por mesa electoral sin incrementar la cantidad de oficiales por mesa, y para tener una respuesta más pronta del resultado de la elección. Actualmente el TSE ha llegado a la etapa de valorar opciones comerciales de voto electrónico, y ha concluido que su costo es prohibitivo. Por este motivo, el TSE esta considerando como principal opción el desarrollo de software autóctono para el sistema de voto electrónico, y dados los requerimientos estrictos de seguridad, transparencia, usabilidad etc. que un sistema de estos requiere.

Abstract: Electronic voting is an option that the National Elections Tribunal is evaluating to attend the diagnosed need of incrementing the number of voters per electorate table without incrementing the number of officials on each table. Another requirement of the Tribunal is that the new system should have the least possible impact on the current way voters emit their vote. By achieving this, the possibility of acceptance of the new system by the general population is incremented. The National Elections Tribunal has evaluated comertial solutions and found their price prohibitive. The inhouse development of software is currently the best option

Palabras clave: Voto electrónico, seguridad informática, máquinas receptoras de votos, identificación automática.

[1] Coordinador del proyecto - Investigador asociado.
[2] Director del CIC – Investigador asociado.

"Triste de los países que no utilicen a la ciencia como guías en sus empresas, se quedarán postergados y estarán supeditados al desarrollo de los demás, porque en las sociedades actuales, aquéllos que utilicen mayor conocimiento y sagacidad, serán los que logren ventajas sobre los otros..."

José María Castro Madriz, 1844

Distribución del documento		
Nº de Copia	Responsable	Organización
1	Jeff Schimdt Peralta	Centro de Investigaciones en Computación – ITCR
2		Tribunal Supremo de Elecciones.
		Total de copias: 2

TABLA DE CONTENIDO

[Esta página se dejó en blanco apropósito]

RESUMEN EJECUTIVO

El Tribunal Supremo de Elecciones costarricense (TSE) se encuentra actualmente en la etapa de evaluar opciones tecnológicas que le permitan modernizar su sistema de votación. Actualmente dicho tribunal ha efectuado varios estudios de la tecnología existente y ha determinado que el sistema más similar a las necesidades del tribunal, corresponde al actual sistema brasileño de voto. Sin embargo, el TSE considera que depender de la tecnología brasileña conlleva a ciertos riesgos que pueden comprometer la disponibilidad de los sistemas y las máquinas.

A partir de esto el TSE ha hecho durante el año 2008 y el 2009 un estudio de mercado para evaluar las opciones tecnológicas que ofrecen las casas comerciales que desarrollan soluciones de voto electrónico. Dicho estudio ha arrojado que los precios solicitados por estas casas son prohibitivos para el presupuesto nacional.

De esta forma el TSE ha tomado la decisión de evaluar opciones tecnológicas locales para efectuar el voto electrónico y partir de un análisis exhaustivo que nos permita a nosotros como costarricenses, generar nuestra propia plataforma para voto electrónico.

Durante el año 2010 se analizaron los requerimientos solicitados por el TSE con base al estudio de factibilidad del 2008. Estos requerimientos se ajustaron a un plan de desarrollo del software beta con el cual se determinaron nuevos requerimientos, flujos de procesos y arquitecturas recomendadas.

La administración del proyecto estuvo respaldada por un plan de gestión y comunicación que incluían varios aspectos para facilitar el desarrollo del software.

1. INTRODUCCIÓN

1.1 PROPÓSITO

En este documento se presenta un marco de referencia sobre las necesidades recopiladas para el desarrollo del voto electrónico en el país. En especial los requerimientos que estarán presentes en el prototipo del Sistema de Identificación de Votantes, los cuales son producto del análisis de factibilidad sobre implementación de requerimientos presentado por el Centro de Investigaciones en Computación del Instituto Tecnológico de Costa Rica al Tribunal Supremo de Elecciones en el año 2008, y la solicitud de esta última entidad en diciembre del 2009. En adición, se presentan algunos detalles sobre la gestión del proyecto.

Para facilitar el análisis este texto se ha estructurado de la siguiente forma:

1. Requerimientos del Producto. Agrupa conforme al estándar IEEE 830 -1998 las necesidades del voto electrónico como requerimientos.
2. Diseño de la Solución. Posee los modelos y estructuras utilizadas para solventar las necesidades planteadas por la especificación de requerimientos.
3. Requerimientos del Proceso. Detalla la organización y gestión del proyecto.

1.2 ALCANCE

Dado que el CIC pretende mostrar un conjunto de escenarios posibles para el voto electrónico, se tomo un subconjunto de requerimientos para desarrollar el prototipo funcional. En el Anexo 2 se puede encontrar la lista de casos de uso que se omitirán para el primer prototipo del sistema de identificación de votantes.

1.2.1 COMPONENTES DEL SISTEMA

Inicialmente se describen los componentes a nivel general del sistema, luego se explica cada uno de ellos, así como los objetivos que pretende resolver el sistema de Voto Electrónico en Costa Rica.

1.2.1.1 DESCRIPCIÓN GENERAL DEL SISTEMA

En el sistema de Voto Electrónico convergen elementos de hardware y software para lograr el objetivo de automatizar la emisión del voto por parte de los electores.

El sistema está formado por dos componentes: carga y configuración y sistema de Voto Electrónico.

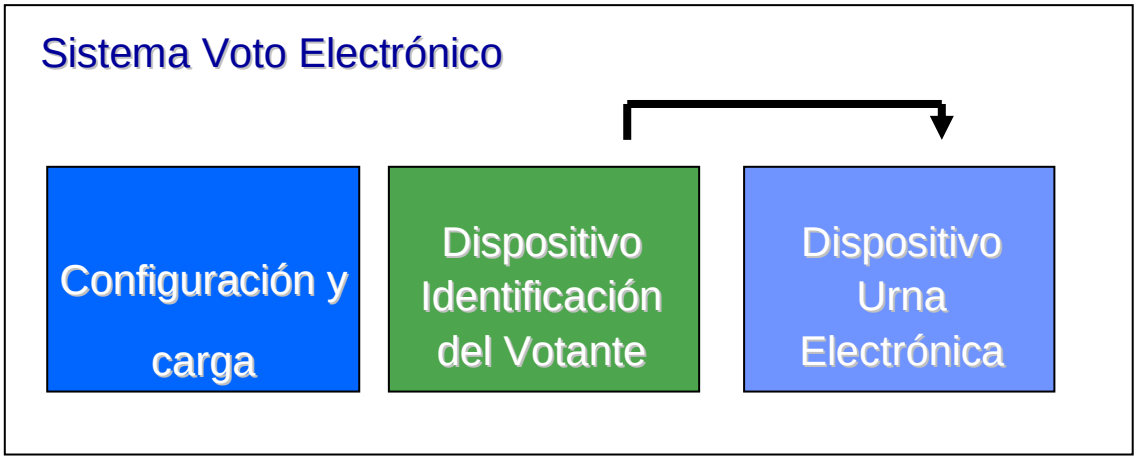


Figura No.1 Componentes del sistema de Voto Electrónico.

1.2.1.2 DESCRIPCIÓN DE LOS MÓDULOS

Se describen las principales características de cada uno de los componentes del Sistema de Voto Electrónico.

Carga y configuración.

Este componente comprende todas las tareas que se deben realizar en la fase preparativa de una elección, las cuales consisten en:

- Configurar los equipos: deben indicarse y activarse apropiadamente todos los parámetros necesarios para la elección que se va a realizar.
- Carga de datos: consiste en cargar a los equipos de votación los datos relacionados con padrón electoral y opciones de votación.

Sistema de Voto Electrónico (VE).

Este sistema va a contener mediante la utilización de un único dispositivo de hardware las funciones relacionadas con dos componentes: identificación del votante y urna electrónica.

El componente de identificación de votante pretende realizar la comprobación del elector. Esta comprobación persigue dos fines fundamentales:

- Asegurar que la persona que se presenta a la junta se encuentre asignada al padrón cargado en el equipo.
- Asegurar que el elector solo se pueda presentar a votar una única vez.

La comprobación de identidad se va a realizar por medio de la lectura de códigos de barra o digitación del número de cédula del elector.

El votante puede luego de que se la active la urna, emitir su voto en forma electrónica. Los votos son registrados directamente en la memoria del dispositivo de votación, mediante la utilización de un dispositivo externo de registro que también graba el voto, para luego ser leído en otro equipo.

1.2.2 Objetivos

Objetivo General.

- Diseñar e implementar diversas alternativas para automatizar el proceso de votación en Costa Rica.

Objetivos Específicos.

- Evaluar la utilización de diversas alternativas de identificación de votantes.
- Definir los requerimientos funcionales y no funcionales para el sistema de identificación de votantes.
- Generar al menos dos prototipos de alternativas para realizar la identificación de votantes en los procesos de votación.

1.2.3 Productos del Proyecto para el año 2011

1. Documento con especificación de requerimientos según estándar IEEE-8003-1998.
2. Desarrollo de software de aplicación para sistema de identificación de votantes.
3. Escogencia de hardware para el sistema de identificación de votantes

1.3 SIGLAS Y ACRÓNIMOS UTILIZADOS

BCCR – Banco Central de Costa Rica

CIC – Centro de Investigaciones

ITCR – Instituto Tecnológico de Costa Rica

SIV – Sistema de Identificación de Votantes

TSE – Tribunal Supremo de Elecciones

VE – Voto Electrónico

1.4 METODOLOGÍA DE DESARROLLO

El proyecto se desarrollará utilizando las siguientes etapas de la metodología de la ingeniería de software:

1. Establecimiento de los requerimientos del sistema
2. Diseño de la solución.
3. Programación de los componentes de hardware.
4. Realización de pruebas.
5. Evaluación de resultados.

2. REQUERIMIENTOS DEL PRODUCTO

Los requerimientos del producto son propiedades expuestas con el fin de resolver algún problema en el mundo real (SWEBOOK 2004 - IEEE). Estos se agrupan en funcionales y no funcionales.

Los funcionales describen las características que debe ejecutar el sistema, en cambio los no funcionales establecen restricciones sobre cómo se debe ejecutar. A estos últimos también se les conoce como requerimientos de calidad, pues la funcionalidad se mantiene.

Para la confección de los requerimientos se tomaron como base los documentos de Análisis de Factibilidad y "Requerimientos Funcionales para Padrón Registro Electrónico" del TSE. Para este último se enunciaron los objetivos del sistema:

- Que los miembros de las juntas receptoras de votos cuenten con un Padrón Registro con Fotografía electrónico por medio del desarrollo de un sistema informático para que mediante la utilización de una PC se pueda verificar la

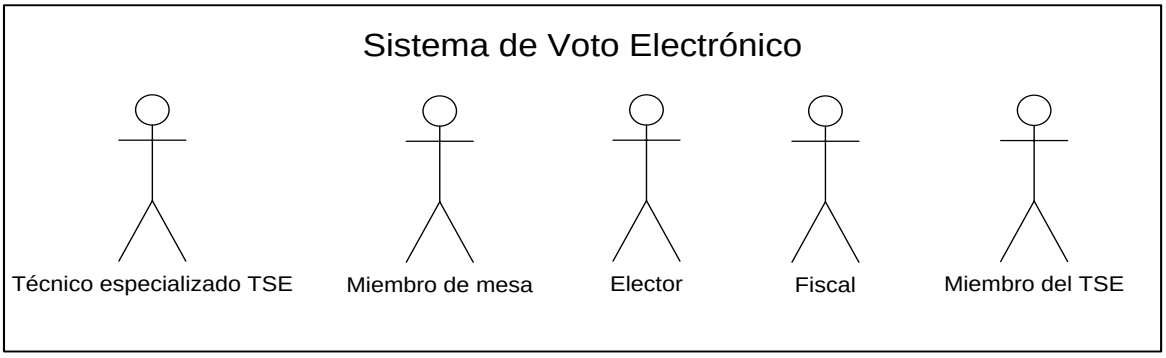
condición de elector e identificar a los votantes en cada junta receptora de votos de manera sencilla y expedita.

- Que facilite los procesos de apertura y cierre de la votación.
- Que el sistema cuente con las seguridades necesarias para garantizar la inalteración de la información del elector.
- Que el sistema cuente con módulos que permitan integrar funcionalidades de otras áreas del proceso electoral para maximizar la inversión del sistema.
- Que el sistema facilite la interacción entre éste y los miembros de las juntas receptoras de votos.
- Que disponga de seguridad en todos los componentes y funciones que garanticen inviolabilidad y sustracción de la información.

Luego se realizó una comparación entre ambos documentos y para el desarrollo del prototipo se optó por delimitar los requerimientos.

2.1 Requerimientos Funcionales

2.1.1 Catálogo de usuarios



Técnico especializado TSE.

Persona encargada de dar soporte durante el día de la votación a los equipos de votación y participar en la aparición de contingencias.

Miembro de mesa.

Es la persona encargada del manejo de la junta receptora de votos, tiene la responsabilidad de administrar la operación de los equipos de votación durante el día de las elecciones.

Elector.

Persona que se presenta a una junta receptora de votos para realizar la emisión del sufragio.

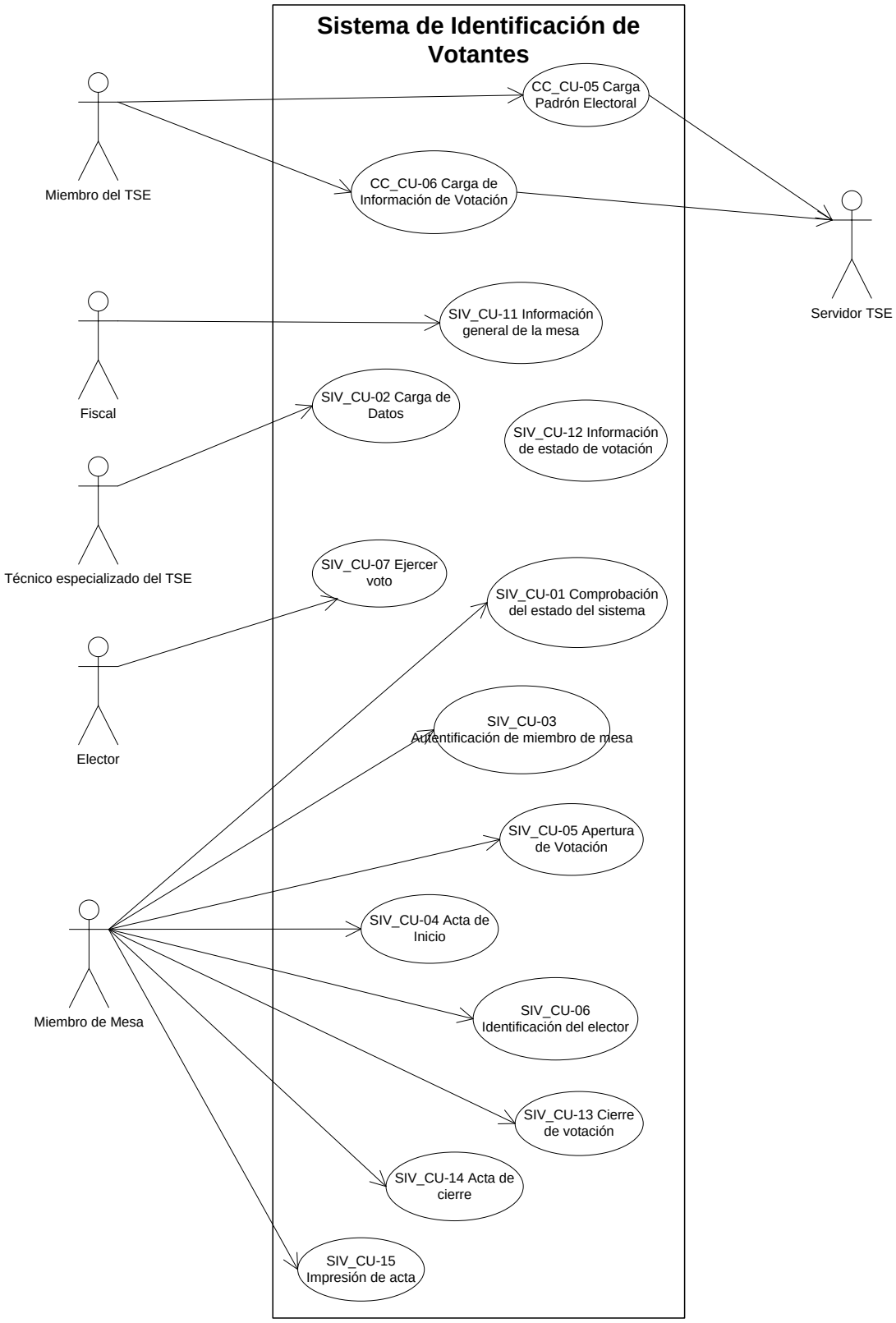
Fiscal.

Persona que se presenta a una junta receptora de votos para fiscalizar las labores del sufragio.

Miembro del TSE.

Es la persona autorizada para realizar las labores de configuración y carga de los equipos en las etapas previas al día de las elecciones.

2.1.2 DIAGRAMA DE CASOS DE USO



2.1.3 DESCRIPCIÓN DETALLADA DE CASOS DE USO

Caso de uso Carga de Información de Votación	**CC_CU-06**
Actor Principal	Miembro del tribunal
Resumen	Se recibe una lista con los asuntos y opciones a votar para cada centro de votación. Esta lista contiene la información de cada tipo de votación y asunto a votar. Además debe indicarse de manera general la fecha de apertura y cierre de la votación.
Pre condiciones	Debe haberse seleccionado los equipos que se van a utilizar.
Pos condiciones	Se cuenta con la información de la votación. El reloj del equipo SIV debe estar sincronizado con el equipo de un equipo del tribunal. Se crea la clase de configuración específica del equipo del SIV.

Escenario principal	
Acción de los actores	Respuesta del sistema
2. El miembro del tribunal selecciona el archivo con la información respectiva. 3. El miembro del tribunal indica al sistema que proceda con la carga de la información suministrada en un rango de mesas.	1. El sistema le indica al usuario que seleccione un archivo con la información de la votación.
	4. El sistema revisa que el archivo e información sean válidos. 5. El sistema carga la información de los miembros de mesa y la muestra en pantalla.
6. El miembro del tribunal acepta la configuración hecha.	6. El sistema carga la información. 7. Se repite el paso 4 al 6 según el rango de mesa solicitado.
Flujo alterno	
Línea 4: En caso de que no sea válido el archivo o la información se indicará un error al usuario y se devuelve a la línea 1.	
Especificaciones suplementarias	
▪ El acceso debe requerir autenticación del usuario al abrir la aplicación. ▪ La aplicación debe contar con control de cambios y configuración. ▪ Debe permitirse cerrar la aplicación y continuar en el paso y estado en que estaba.	

Caso de uso Carga Padrón Electoral	CC_CU-05
Actor Principal	Miembro del tribunal
Resumen	Se recibe una lista con los miembros del padrón asignados a cada junta.
Pre condiciones	La votación debe estar registrada.
Pos condiciones	El sistema cuenta con el padrón electoral para cada junta receptora de votos.

Escenario principal

Acción de los actores	Respuesta del sistema
	1. El sistema le indica al usuario que seleccione un archivo con la información del padrón electoral.
2. El miembro del tribunal selecciona el archivo con la información respectiva. 3. El miembro del tribunal indica al sistema que proceda con la carga de la información suministrada.	
	4. El sistema revisa que el archivo e información sean válidos. 5. El sistema carga la información. 6. Llama al siguiente caso de uso "Carga de Información de Votación".

Flujo alterno

Línea 4: En caso de que no sea válido el archivo o la información se indicará un error al usuario y se devuelve a la línea 1.

Especificaciones suplementarias

- Dentro del reporte esta información será indicada con cantidad total de votantes y cantidad total de mesas con información asociada.
- El acceso debe requerir autenticación del usuario al abrir la aplicación.
- Debe permitirse cerrar la aplicación y continuar en el paso y estado en que estaba.
- El padrón electoral debe contener los siguientes datos: nombres, apellidos, número de cédula, fecha de nacimiento, fotografía, sexo del elector, número de la junta receptora de votos y número del elector
- El padrón electoral debe realizar una carga completa del padrón o parcial por junta receptora de votos. Además debe considerarse la carga del padrón con foto y sin foto.
- Debe garantizarse que el proceso de carga de los dispositivos que se distribuirán en las juntas receptoras de votos tarde a lo más un mes, ya que es el tiempo que se cuenta a partir del cierre del padrón.

Caso de uso Comprobación del estado del sistema	SIV_CU-01
Actor Principal	Miembro de mesa
Resumen	Cada vez que el equipo inicia realizará un reporte con un chequeo total del sistema junto con la fecha y hora del reporte. Esta información es almacenada junto con los datos del sistema, de manera que permita un control de cuantas veces el equipo fue encendido. El chequeo revisa hardware (reconocer dispositivos y que respondan debidamente), software (sistema operativo y aplicaciones cargan debidamente) y configuración (información necesaria como el padrón de electores). Dicho chequeo deberá mostrarse en pantalla. Además, el reporte incluirá la versión de cada software y configuración utilizada.
Pre condiciones	Debe estar cargado el sistema.
Pos condiciones	Reporte digital del estado del sistema, que alimentará el acta de inicio respectiva.

Escenario principal

Acción de los actores	Respuesta del sistema
1. El miembro de mesa enciende el equipo respectivo.	2. El sistema inicia el chequeo de sus componentes, hardware, software y configuración, almacena los datos obtenidos en la bitácora interna, junto con la información pertinente para el control de veces que el equipo fue encendido. 3. Llama al siguiente caso de uso "Autenticación del miembro de mesa".

Flujo alterno

Línea 2: Las versiones de la configuración y/o software no son concordantes. El sistema mostrará un mensaje indicándole al usuario el problema. Impide el avance del sistema al siguiente paso.

Línea 2: El sistema no cuenta con una carga de configuración. Se procede a la carga de configuración (Caso de Uso "Carga de Datos").

Línea 2: Existe un acta de apertura registrada en el sistema. El sistema muestra al usuario un mensaje indicando que este sistema ya fue utilizado y que no puede volverse a iniciar un proceso de identificación con el mismo hasta que los datos sean extraídos oficialmente en el TSE.

Especificaciones suplementarias

- El sistema puede ser encendido en cualquier momento.
- El sistema puede apagarse al final de este caso de uso sin tener una implicación en el flujo de los eventos la próxima vez que sea encendido.

Caso de uso Carga de Datos	SIV_CU-02
Actor Principal	Técnico especializado del TSE
Resumen	En caso de que el equipo no cuente con una configuración el sistema deberá solicitar la carga de estos datos. Este caso se presentará en los equipos de contingencia, ya que estos contarán con el software instalado pero carecen de una configuración puesto que no se sabe a cual mesa estarán vinculados.
Pre condiciones	Debe estar cargado el sistema. El equipo no debe tener creada un acta de inicio.
Pos condiciones	El sistema estará listo para iniciar su funcionamiento.

Escenario principal

Acción de los actores	Respuesta del sistema
1. El técnico especializado del TSE introduce un medio de almacenamiento de datos que contiene el conjunto de todas las configuraciones de datos de todas las mesas correspondientes al centro de votación, estas configuraciones igualmente deben contar con un medio de identificación único.	
	2. El sistema mostrará en pantalla todas las configuraciones correspondientes al centro de votación.
3. El técnico del tribunal selecciona la configuración de datos que desea instalar en el sistema.	
	4. El sistema instalará la configuración en el equipo dejándolo listo para empezar a trabajar. Además dejará un registro en la bitácora del sistema para posteriormente poder identificar que es un equipo de contingencia.

Flujo alterno

Línea 2: Las versiones de la configuración y/o software no son concordantes. El sistema mostrará un mensaje indicándole al usuario el problema. Impide el avance del sistema al siguiente paso.

Especificaciones suplementarias

- La carga de una configuración debe realizarse una única vez, es necesaria una confirmación por parte del técnico especializado del TSE antes de realizar la carga. En caso de error debe registrarse en bitácora y volver a llamar al caso de uso "Carga de Datos".
- El sistema puede apagarse después de realizar este proceso.

Caso de uso Autentificación de miembro de mesa	SIV_CU-03
Actor Principal	Miembro de mesa
Resumen	El sistema al iniciar no puede ser utilizado hasta que un miembro de dicha mesa se autentique. Una vez autenticado, el sistema solo pedirá al miembro de mesa autenticarse nuevamente en tareas críticas (por ejemplo acta de inicio y acta de cierre). El mecanismo para autenticación de miembros de mesa es mediante código de barras de la cédula o clave de seguridad.
Pre condiciones	Debe estar cargado el sistema. Las contraseñas del Fiscal del TSE y presidente de mesa ya deben estar registradas.
Pos condiciones	El sistema estará listo para usarse.

Escenario principal	
Acción de los actores	Respuesta del sistema
2. El miembro de mesa se identifica mediante una contraseña o lectura del código de barras.	1. El sistema solicita la autenticación del miembro de mesa. 3. El sistema valida la información de identificación y registra en bitácora la autenticación del miembro de mesa. 4. Llama al siguiente caso de uso "Acta de Inicio".
Flujo alterno	
Línea 3: El código de identificación introducido no es válido. El sistema no podrá ser utilizado hasta que se introduzca un código correcto, sin embargo existirá un límite de oportunidades para realizar la autenticación, este valor se definirá en la fase preparativa.	
Especificaciones suplementarias	
• El sistema puede apagarse durante este proceso hasta el momento antes que de que se realice la validación sin tener una implicación en el flujo de los eventos la próxima vez que sea encendido.	

Caso de uso Acta de Inicio	SIV_CU-04
Actor Principal	Miembro de mesa
Resumen	El sistema genera un acta donde se registre información, imprime un número de actas según se defina en la fase preparativa y registra en la bitácora la acción. Este proceso debe realizarse de forma obligatoria, no deberá poderse continuar el proceso de votación hasta que esta acta sea impresa al menos una vez.
Pre condiciones	Debe estar cargado el sistema. El miembro de mesa debe estar validado.
Pos condiciones	El sistema estará listo para iniciar la votación.

Escenario principal

Acción de los actores	Respuesta del sistema
1. El miembro de mesa solicita la impresión de un acta de inicio.	2. El sistema valida que ningún elector ha emitido su voto. 3. El sistema registra la fecha, identificación única del equipo, versión y configuración de aplicación.
4. El miembro de mesa ingresa los datos producto de la revisión de las papeletas.	5. El sistema valida el total de papeletas contra la cantidad de electores de la junta.
6. El miembro de mesa emite comentarios.	7. El sistema llama al caso de uso "Impresión de Acta" y aumenta el contador de impresiones. 8. Registra la creación del acta en la bitácora de acción. 9. Llama al siguiente caso de uso "Apertura de votación".

Flujo alterno

Especificaciones suplementarias

- La identificación del equipo se puede hacer a través de la captura de Identificador Maquina (lshw) + Identificador de HDD + MAC Address + Información del CPU.
- Existen dos opciones a considerarse:

a) Una vez impresa el acta de inicio no será posible apagar el sistema hasta que termine el proceso de votación, en caso de apagar el sistema luego de realizar esta acción el sistema quedará bloqueado hasta que sea llevado al TSE para garantizar que los datos no puedan ser modificados. Deberá utilizarse un equipo de contingencia para poder continuar con el proceso.

b) El sistema podrá usarse a pesar de que se apague esto con el fin de que en caso de una contingencia se pueda restablecer el sistema y dar continuación al proceso; una vez que se reinicie el sistema este debería chequear sus componentes y configuración de nuevo, si todo esta correcto, el miembro de mesa se autentica para continuar el proceso normal, todo debe quedar registrado en bitácora.

Caso de uso Apertura de Votación	SIV_CU-05
Actor Principal	Miembro de mesa
Resumen	Es el proceso que permite dejar el sistema listo para iniciar el proceso de identificación de electores, sin embargo el sistema deberá bloquearse mientras no sea la hora oficial de inicio de votación, se le dará retroalimentación al usuario del tiempo restante para poder iniciar.
Pre condiciones	Debe estar cargado el sistema. El miembro de mesa debe estar validado. Se debe haber impreso al menos un acta de inicio.
Pos condiciones	El sistema estará listo para iniciar el proceso de identificación de electores.

Escenario principal	
Acción de los actores	Respuesta del sistema
1. El miembro de mesa solicita al sistema el inicio de un nuevo proceso de votación.	2. El sistema verifica que se haya impreso al menos un acta de inicio. 3. El sistema muestra un mensaje indicando el tiempo restante antes de que inicie oficialmente el proceso de votación. Cuando el tiempo llegue cero la aplicación llamará al caso de uso "Identificación de elector".
Flujo alterno	
Línea 2: En caso de que no se haya impreso un acta de inicio se llama al caso de uso "Acta de Inicio".	
Especificaciones suplementarias	

Caso de uso Identificación del elector	SIV_CU-06
Actor Principal	Miembro de mesa, Elector.
Resumen	Cuando un elector llega a la mesa, debe identificarse suministrando la cédula de identidad a los miembros de mesa. Se debe ingresar el número de cédula, o de manera automática mediante un dispositivo de lectura del código de barras de la cédula, el sistema muestra en pantalla los datos personales y foto del elector. Los miembros de mesa deben determinar la autenticidad y validez del elector mediante la información dada por el sistema.
Pre condiciones	Debe haberse iniciado un proceso de votación.
Pos condiciones	El elector se habrá identificado y estará listo para el siguiente paso de proceder a votar.

Escenario principal

Acción de los actores	Respuesta del sistema
1. El elector se valida en el sistema a través del método de identificación que se desee implementar, pudiendo este ser lectura del código de barras de la cédula o digitar número de cédula.	2. El sistema verifica la información del elector y valida su identidad con el padrón local. 3. El sistema muestra una pantalla con toda la información del elector contenida en el padrón tales como número de cédula, nombre completo y opcionalmente la foto.
4. El elector entrega la cédula de identidad al miembro de mesa. 5. El miembro de mesa verifica a través de la información mostrada la identidad del elector.	

Flujo alterno

Línea 1: El sistema indica al miembro de mesa que el elector ya ejerció el voto. Se ofrece una guía al miembro de mesa para comprobar que ya votó y para minimizar la desactivación de un elector que no ha votado.

Línea 2: Si no aparece en el padrón local se deberá buscarlo en el padrón nacional e indicar en que mesa y lugar vota. Si aun en el padrón nacional no aparece se deberá mostrar en pantalla un procedimiento para ubicar a la persona. Por ejemplo: Llamar a una línea de consulta del tribunal.

Especificaciones suplementarias

- El elector se podrá identificar a través de cualquier medio antes especificado. El hecho de que existan distintos métodos de identificación es con el fin de contar con distintas alternativas en caso de que un método de identificación no funcione. No es necesario identificarse a través de todos los medios, bastará con uno solo.
- El sistema debe validar cuales registros no tienen fotografía asignada en el SICI y asignarle un rotulo que indique esta situación.
- En caso de que se compruebe una suplantación de identidad se registrará una incidencia.

Caso de uso Ejercer voto	SIV_CU-07
Actor Principal	Elector
Resumen	En la pantalla de datos personales se muestra la información del elector, el miembro de mesa registra la acción correspondiente a la confirmación de votación. El elector debe firmar en una lista, inicialmente en blanco, previo a proceder con su voto.
Pre condiciones	El elector se ha identificado
Pos condiciones	El elector queda marcado como votado lo cual implica que no podrá volver a votar.

Escenario principal	
Acción de los actores	Respuesta del sistema
1. El elector pone en un listado su número de cédula y firma. 2. El miembro de mesa indica al sistema que la persona va a proceder a votar.	3. Se marca el elector como votado directamente en el padrón.
Flujo alterno	
Especificaciones suplementarias	

Caso de uso Información general de la mesa	SIV_CU-11
Actor Principal	Usuario autorizado.
Resumen	El sistema en todo momento debe permitir la consulta de información respecto a la mesa de votación. Esta información incluye centro de votación al que pertenece, provincia, cantón, distrito electoral, número de junta y cantidad de electores de mesa.
Pre condiciones	Debe haberse aprobado la comprobación del estado del sistema.
Pos condiciones	

Escenario principal	
Acción de los actores	Respuesta del sistema
1. El usuario solicita la información de la mesa de votación.	2. El sistema despliega en pantalla la un rótulo que indique la provincia, cantón, distrito electoral, número de junta y cantidad de electores de la junta, los datos de la cantidad de electores que han emitido el voto a cualquier hora durante el transcurso de la votación.
Flujo alterno	
Especificaciones suplementarias	

- Usuario autorizado depende del momento en que se utilice el sistema y no requiere autenticación alguna. Durante el periodo de votaciones solo los miembros de mesa se suponen como usuarios autorizados.
- La información debe ser tomada de la clase de configuración que pertenece al equipo.

Caso de uso Información de estado de votación	SIV_CU-12
Actor Principal	Sistema
Resumen	Durante el periodo comprendido entre la apertura y cierre de la votación debe mantenerse en pantalla información relevante para los miembros de mesa con respecto al estado de la votación. Esta es el total de electores para la mesa específica, hora oficial de inicio y cierre y hora actual del sistema.
Pre condiciones	Debe haber una apertura de votación.
Pos condiciones	

Escenario principal	
Acción de los actores	Respuesta del sistema
	1. El sistema muestra en pantalla la información respectiva.

Flujo alterno

Especificaciones suplementarias
▪ La información debe ser tomada de la clase de configuración que pertenece al equipo.

Caso de uso Cierre de votación		SIV_CU-13
Actor Principal	Miembro de mesa.	
Resumen	Indica al sistema que no se van a identificar más electores o no se registrarán más votos según corresponda el sistema. Este proceso se realizará de forma automática una vez que se cumpla con la hora de finalización que se especifico para el proceso electoral que está en curso. Una vez finalizado el proceso de cierre de votación se llamará al caso de uso "Acta de cierre".	
Pre condiciones	Apertura de votación.	
Pos condiciones	No se puede identificar más electores. Se puede proceder con el acta de cierre.	
Escenario principal		
Acción de los actores		Respuesta del sistema
2. El miembro de mesa indica a los electores que no se pueden emitir más votos. 4. El miembro de mesa llama al caso de uso "SIV_CU-14 Acta de cierre".		1. El sistema deshabilita la identificación de electores y registra la acción en bitácora. 3. El sistema muestra en pantalla las instrucciones para comenzar el acta de cierre.
Flujo alterno		
Especificaciones suplementarias		

Caso de uso Acta de cierre	SIV_CU-14
Actor Principal	Miembro de mesa.
Resumen	El sistema genera un acta donde se registre la siguiente información: fecha y hora del sistema; identificación única del equipo utilizado, aplicación y configuración utilizada, número de Junta, fecha y hora del cierre, lugar de votación, las categorías y las opciones de voto.
Pre condiciones	Apertura de votación. Estas solo podrán imprimirse si la elección ha sido finalizada.
Pos condiciones	El sistema está listo para apagarse.

Escenario principal	
Acción de los actores	Respuesta del sistema
1. El usuario ingresa de los nombres y datos de los miembros de la junta receptora de votos, fiscales y auxiliares electorales al momento del cierre. 2. Consignar los nombres de los partidos políticos y para ingresar el resultado de la votación por cada uno. Debe totalizar las cifras y validar contra la cantidad de electores de la junta receptora.	3. El sistema genera el acta de cierre con la información respectiva, registra la acción en bitácora, imprime el acta y la cantidad de copias que se requieran. 4. El sistema muestra los resultados finales los debe presentar con números y letras.
Flujo alterno	
Línea 2. En caso de existir electores pendientes el sistema indica un error y sale de la acción de acta de cierre.	
Especificaciones suplementarias	

Caso de uso Impresión de acta		SIV_CU-15
Actor Principal	Sistema	
Resumen	El caso de uso es invocado para la impresión de actas en formato de PDF y almacenarlas en un dispositivo de memoria no volátil para luego poder ser enviadas al TSE.	
Pre condiciones	El acta es creada por el caso de uso anterior.	
Pos condiciones	Se crea un archivo de extensión PDF con el nombre del acta.	
Escenario principal		
Acción de los actores	Respuesta del sistema	
	1. Se crea el archivo con el nombre del acta. 2. Se crea el archivo en formato PDF. 3. Se guarda el archivo.	
Flujo alterno		
Línea 1. El archivo ya existe con ese nombre.		
Especificaciones suplementarias		
▪ El formato PDF debe respetar el estándar ISO/IEC 32000-1:2008. ▪ Debe estar firmado digitalmente para garantizar su integridad.		

2.2 Requerimientos No Funcionales

2.2.1 Interfaces de Sistema

Se debe proveer de una interfaz de comunicación entre el sistema de Configuración y Carga.

2.2.2 Interfaz Gráfica de Usuario

El diseño de las pantallas debe respetar el estipulado en el Manual de identidad grafica institucional del Tribunal Supremo de Elecciones.

2.2.3 Almacenamiento de datos

Por ser definido.

2.2.4 Interfaces de comunicación

Por ser definido.

2.2.5 Dispositivos de Entrada

Por ser definido.

2.2.6 Dispositivos de Salida

Por ser definido.

2.2.7 Seguridad del dispositivo

No se puede extraer información sensible o privada del sistema por personas no autorizadas.

2.2.8 Alimentación de Energía

Por ser definido.

2.2.9 Licencias de dependencias

Utilización de software de fuente abierta.

Las licencias de Software Libre (SL) son muchos, y el proyecto considera una licencia de Software Libre a la intersección de las licencias aprobadas por la OSI (Open Source Initiative) y la FSF (Free Software Foundation):

http://www.opensource.org/licenses/category

http://www.gnu.org/licenses/license-list.html

Si bien todas las licencias de éste subconjunto son SL, no todas son compatibles entre si y no pueden/deben ser mezcladas en una aplicación:

http://www.gnu.org/licenses/gpl-faq.html#WhatIsCompatible

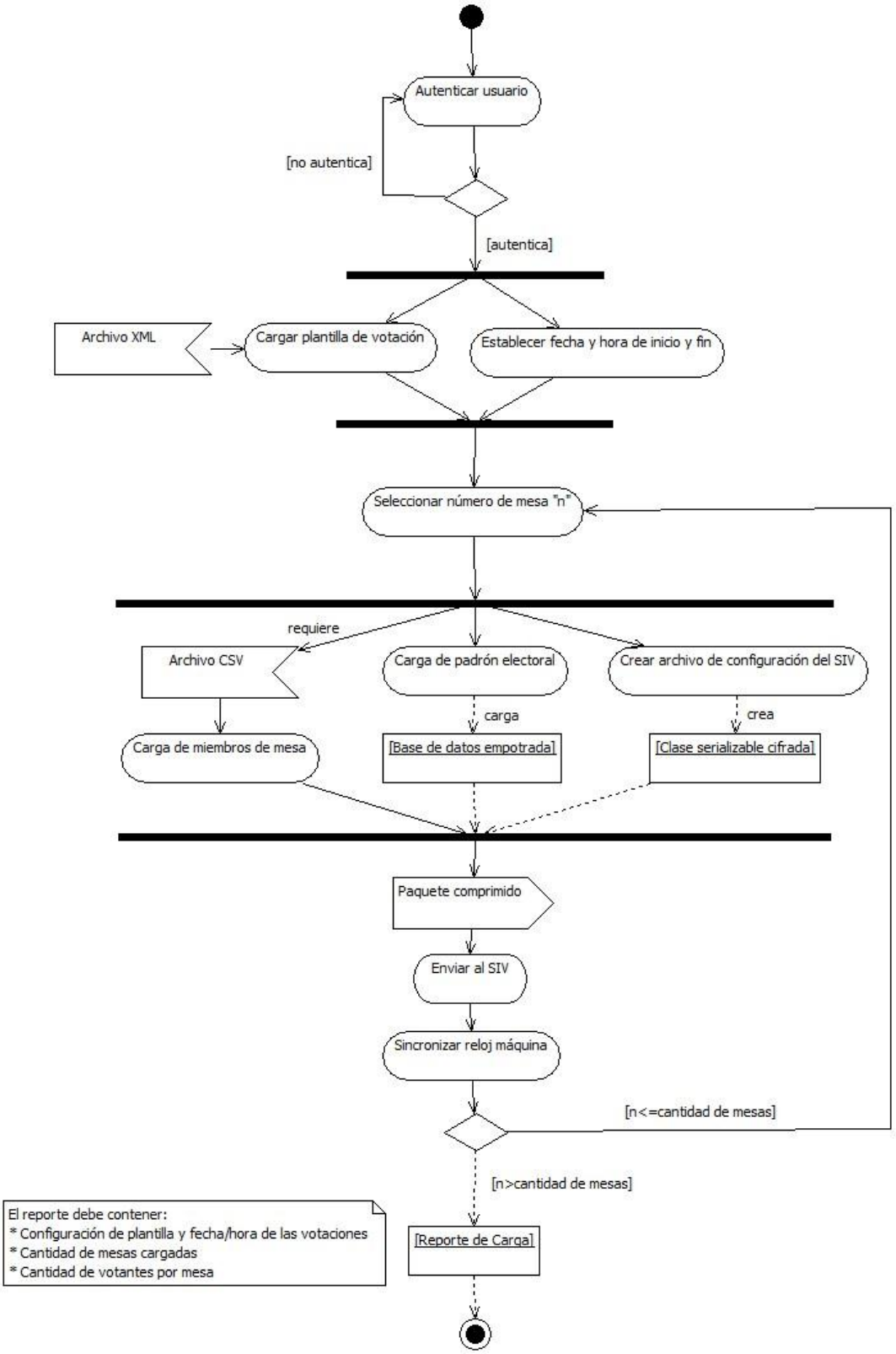
Dados los parámetros del proyecto, preferimos licencias que tengan:

- Un copyleft débil: es decir, se puede integrar las bibliotecas con código propietario sin ninguna responsabilidad sobre la publicación del código del proyecto. Algunos de los ejemplos más conocidos de éste tipo de licencias son: BSD License, Apache License, MIT License, EPL (Eclipse Public License). En caso de la eventual modificación de las bibliotecas, no es obligación legal pero si altamente recomendado por el proyecto la liberación de las modificaciones de la biblioteca para una máxima mantenibilidad.

- Excepción para el linkeo dinámico: son licencias con un fuerte copyleft, es decir, donde existe la obligación legal de publicar los eventuales cambios realizados al código de la biblioteca regida por dicha licencia. Sin embargo, se permite que la biblioteca sea utilizada en código propietario siempre y cuando éste la utilice con la técnica de linkeo dinámico, es decir, haga referencia de ella ("biblioteca de sistema") y no la empotre o la incorpore. El caso más emblemático de dicha licencia es la LGPL.

2.3 Diagrama de actividad de la aplicación.

2.3.1 Aplicación de Carga y Configuración



2.3.2 Aplicación de Identificación de Votantes