



Diseño de Aplicaciones 1 Obligatorio 2

Sebastián Uriarte N°: 194973

Juan Rodríguez N: 171983



JUNIO 2016

Índice

1. Descripción general	2
1.1 Supuestos.....	2
2. Justificación de Diseño	3
Primer obligatorio	3
Segundo obligatorio	4
Modelo Conceptual	7
3. Diagrama de Paquetes	8
4. Diagrama de Clases	9
4.1 Diagrama de Dominio.....	9
4.2 Diagrama de Excepciones.....	10
4.3 Diagrama de Interfaz.....	11
5. Diagramas de Interacción	14
6. Modelo de Tablas de la estructura de la Base de Datos	15
7. Evidencia de Clean Code.....	16
8. Pruebas	17
8.1 Datos de Prueba	17
8.2 Cobertura de Pruebas Unitarias.....	39

1. Descripción general

Como parte de este obligatorio se nos pidió implementar un sistema que monitoree el equipamiento instalado en plantas de producción. Estos sistemas SCADA generalmente utilizados en plantas industriales son los encargados de supervisar y controlar dichas instalaciones.

1.1 Supuestos

- Se asume que:
 - Las Plantas Industriales pueden tener el mismo nombre sin importar el dónde se encuentran.
 - Las Variables pueden tener el mismo nombre sin importar el nivel dónde se encuentran.
 - Los Dispositivos pueden tener el mismo nombre sin importar el nivel dónde se encuentran.
 - Los Tipos de Dispositivos no pueden tener el mismo nombre sin importar el Dispositivo al cual pertenecen.
- Se decidió que:
 - El Historial de Variables se creará a partir del segundo valor registrado, pero siempre guardando el anterior en forma desfasada al actual.
 - En caso que se decidiera borrar un Elemento “padre”, éste no puede tener “hijos”. Es decir, sólo se permitirá eliminar las “hojas” del árbol que contiene a todas las Plantas de Producción.
 - La única manera de poder Eliminar un Tipo de Dispositivo es si éste no se encuentra asociado a ningún dispositivo.
- Con respecto a la eliminación de Dispositivos, decidimos que se puede borrar cualquier Dispositivo sin importar si este se encuentra en uso o no.
Con esto nos referimos a si se encuentra asociado a una Planta Industrial o no.

- Se observó la posibilidad de la existencia de una herencia de Variable y Dispositivo hacia Componente. Con esto se evita duplicaciones de métodos y una mejor implementación del problema.

2. Justificación de Diseño

Primer obligatorio:

La solución concebida como parte del presente obligatorio está compuesta por, a grandes rasgos, cuatro paquetes que dividen el trabajo que debiera realizarse para poder resolver el problema planteado.

En primer lugar, existe un paquete denominado “Dominio”. En éste se almacenaron las distintas clases e interfaces vinculadas a las entidades de la realidad presentada en la propuesta y a los elementos que ésta involucra; a saber, “Instalación”, “Dispositivo”, etc. Se intentó lograr que, como regla general, cada clase fuera conocedora de sus datos, cuidando la exposición de los mismos a otras, y responsable de las validaciones vinculadas a la integridad y coherencia de los mismos. Como puntos particulares de interés a efectos de esta sección, en primer lugar, se decidió implementar una relación de generalización entre Dispositivo e Instalación, y un concepto nuevo denominado “Componente”, dado que se consideró inicialmente estas dos entidades compartían una serie de funcionalidades y atributos, como el manejo de un nombre, el poseer distintas variables, etc., si bien posteriormente en ciertos puntos del obligatorio se observó que era necesario utilizar casteos entre un tipo y otro para lograr implementar algunas funcionalidades correctamente, por lo que esta pudo no haber sido una decisión muy acertada.

Por otro lado, para manejar la cantidad de alarmas activas en una rama jerárquica de instalaciones, se le introdujo a la clase Variable un atributo del tipo Componente y a este uno de tipo Instalación, que representaran el padre al cual estos estarían asociados, como forma de generar una recursión hacia arriba que modificara los atributos necesarios en contraposición a posiblemente tener que efectuar una búsqueda en cierta estructura de datos, mecanismo evaluado como menos eficiente. Finalmente, se utiliza una clase “fachada” para manejar los tipos del

sistema, y las instalaciones y dispositivos de primer nivel, como forma de pasar a la interfaz un único objeto que almacene los datos manejados por el programa en ejecución.

Un segundo paquete es “Excepciones”, que contiene una serie de clases que heredan de las excepciones presentes en C# y no contienen código propio más que un constructor que llama al heredado. Se decidió incorporar éstas luego de una sugerencia docente, como forma de escribir sentencias *try-catch* de excepciones ajenas al lenguaje y así evitar capturar de forma inadvertida excepciones que podrían estar siendo producidas a raíz de un defecto de programación no contemplado.

En tercer lugar, existe un paquete denominado “PruebasUnitarias”, contenedor de, como su nombre lo indica, las distintas clases de pruebas unitarias encargadas de comprobar el correcto funcionamiento de los métodos implementados en las distintas clases del paquete Dominio, encargadas, como se mencionó anteriormente, de intentar modelar en código la realidad del problema planteado. Este adquiere una especial importancia dada la metodología de TDD (*Test-driven Development*) que se intentó aplicar en este obligatorio, mediante la cual el código funcional se genera luego de codificadas las pruebas unitarias vinculadas al mismo.

Finalmente, el último paquete contenido en la solución generada es “Interfaz”, que contiene las distintas clases y ventanas que permiten utilizar aquello codificado en Dominio, como nexo entre el usuario y la lógica del problema. Ésta fue implementada utilizando la biblioteca de clases Windows.Forms, y los distintos componentes que ésta ofrece como parte del *framework* .NET. Se intentó que ésta fuera similar al prototipo presentado en la propuesta y de generar una interfaz amigable para el usuario, dentro del requerimiento de usabilidad presente en este obligatorio.

Segundo obligatorio:

Para este segundo obligatorio hubieron de considerarse e implementarse numerosas modificaciones respecto de la situación considerada para el primero. En

primer lugar, con respecto de las decisiones de diseño manejadas para el primero, se continuó trabajando en la idea que guiaría en el primero el establecer la herencia anteriormente mencionada entre “Componente”, “Instalación” y “Dispositivo”, enmarcándose esta dentro de lo que sería algo similar a la aplicación del patrón *Composite*. Se estimó que las dificultades anteriormente mencionadas eran ineludibles e inherentes a la aplicación del mismo, que se extendió en este obligatorio a los tipos “Elemento SCADA” y “Planta Industrial”, nuevo elemento del dominio a ser manejado en el código. Ésta, además de tener sus atributos propios (a saber, *strings* representando la ciudad y la dirección de la misma), se observa en cuanto a las restricciones de los tipos posee la diferencia de no manejar Variables dentro de la misma. Es por esto que se decidió respetar la generalización original y extenderla de forma de que tanto “Planta Industrial” como “Componente” heredaran de “Elemento SCADA”, tipo genérico de entidades del problema a ser manejado dentro de las distintas listas y conjuntos de las mismas.

Por otro lado, otros cambios presentes dentro de la propuesta del presente obligatorio, como por ejemplo el pasarse a manejar intervalos de advertencia dentro de las variables y la cantidad de las mismas en elementos, no impactaron a nivel del diseño en cuanto a relaciones entre clases o tipos, sino más bien a nivel estructural de las mismas, añadiéndose nuevos atributos que permitieran llevar control de los datos requeridos. La propagación de la activación de alarmas y advertencias se implementaría aplicándose una variante del patrón *Observer*, con el cual la entidad Variable notifica a su padre que ha cambiado, mas no le indica que actualice su estado, sino que directamente se actualiza incrementando o decreciendo el atributo que corresponda, mediante métodos internos al paquete.

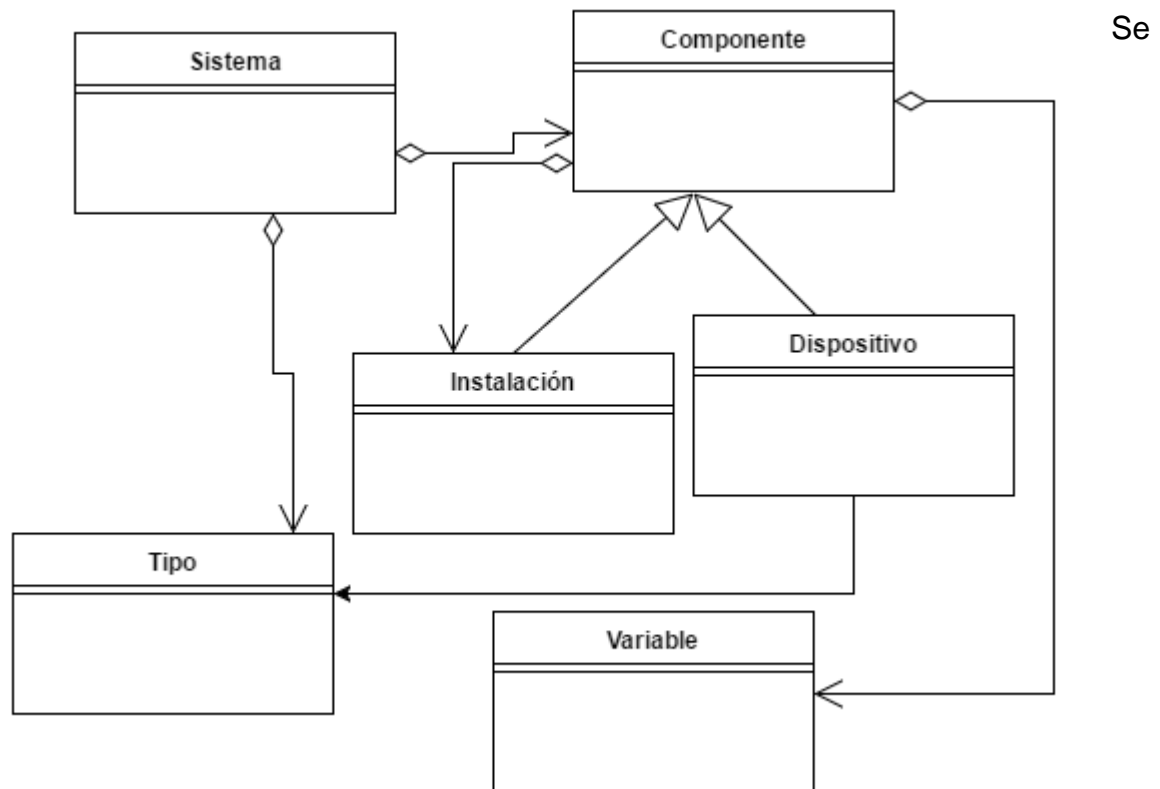
Con respecto al manejo de la persistencia de las entidades del problema, uno de los principales puntos de este obligatorio, éste, según indica la propuesta, se intentó resolver mediante *Entity Framework*, una librería provista para la plataforma *Microsoft .NET* cuyo objetivo es brindar herramientas para el manejo de la persistencia de entidades complejas en bases de datos. En este caso, se buscó implementar al patrón *Repository* en base a la guía que la *Microsoft Developer Network* provee para el mismo en base a este *framework*. Para esto, se crearon

clases de manejo de la persistencia para los distintos tipos de entidades, que heredaran de un tipo “Repositorio” genérico común, y sobrescribieran los métodos del mismo según fuera necesario.

En cuanto a esto, un punto que plantea posibilidades de mejora es el hecho de que, debido a nuestra inexperiencia con la herramienta y a las limitaciones de la misma al manejar ciertos tipos, fue necesario adaptar o condicionar el diseño de la solución planteada a su funcionamiento, sacrificando en ocasiones, mediante la inclusión de propiedades auxiliares o exposiciones innecesarias de los atributos de una clase, la correctitud del diseño en pos de lograr que ésta funcionara en conjunción con *Entity Framework*.

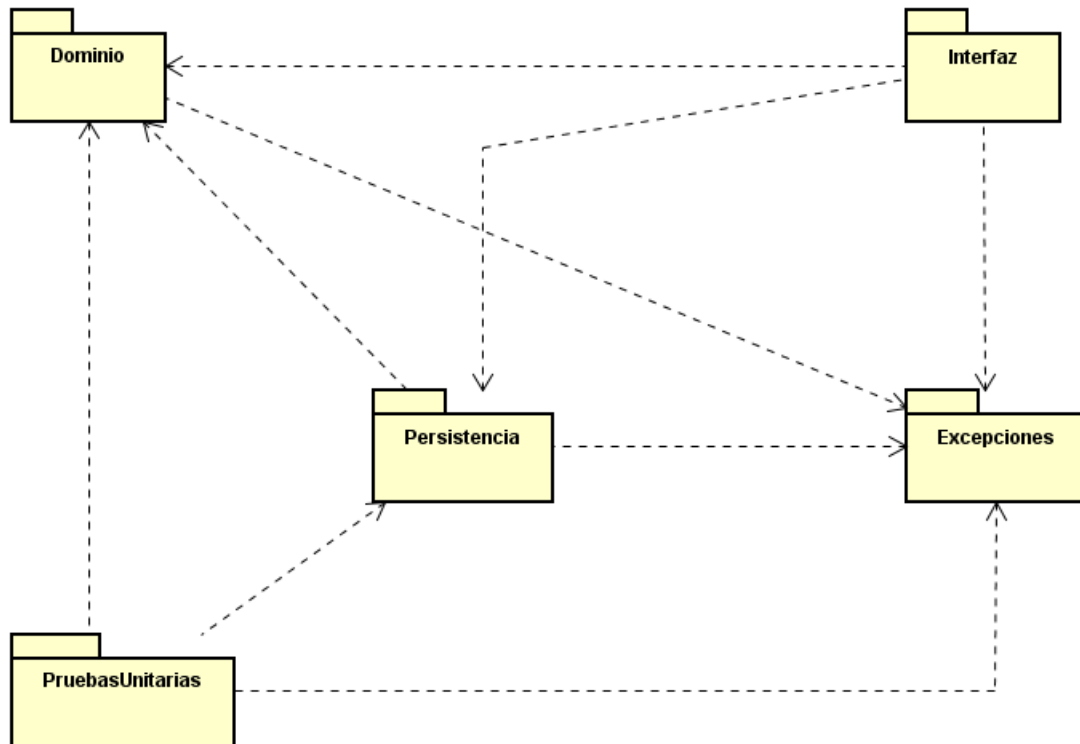
Asimismo, caso especial fue la persistencia de “Incidentes” en la aplicación, que según la propuesta plantearía esta sería implementada mediante la utilización del patrón “*Strategy*”, para permitir la extensibilidad en cuanto a mecanismos de guardado de los datos de los mismos, debiéndose permitir incluso en el contexto de esta situación cambiar en tiempo de ejecución entre dos implementados: archivos de texto y base de datos. Lo referente a la persistencia dentro de la solución sería colocado en un nuevo paquete lógico denominado “Persistencia”, dado que se consideró esto implicaba responsabilidades diferentes a las manejadas por las entidades dentro de la lógica del dominio y por ende resultaría conveniente separar las clases en estructuras distintas. En esto consistieron los principales cambios introducidos en el contexto de este obligatorio.

Modelo Conceptual



adjunta el modelo conceptual de la primera entrega del Obligatorio.

3. Diagrama de Paquetes

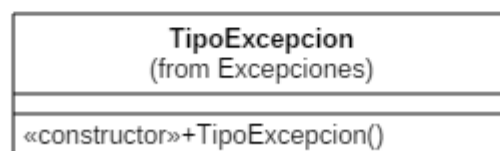
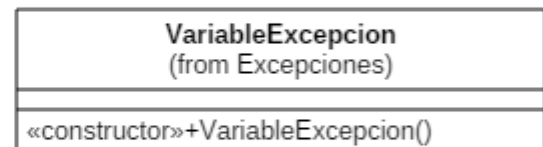
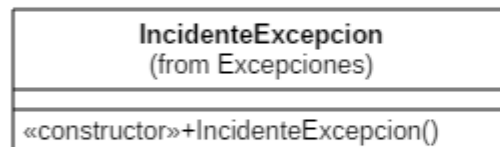
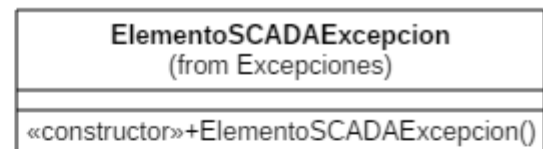
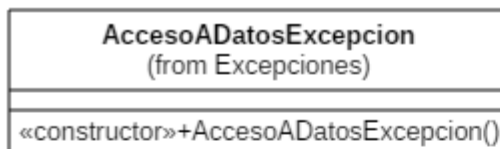
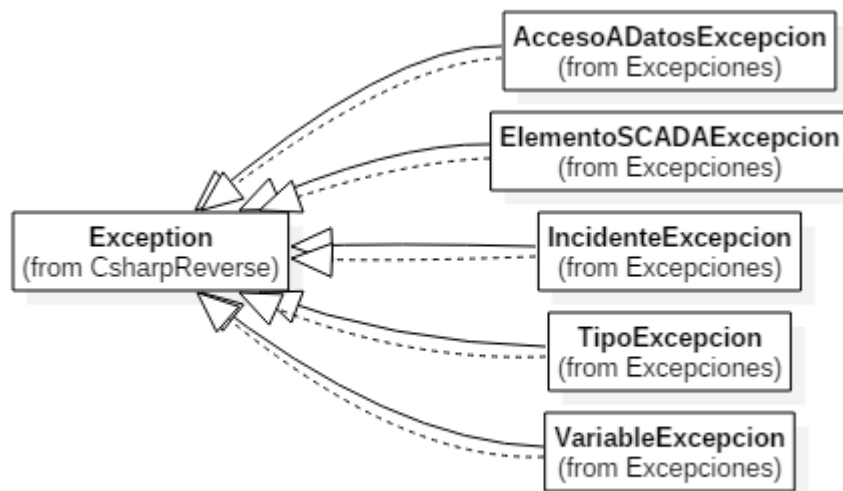


4.1 Diagrama de Dominio

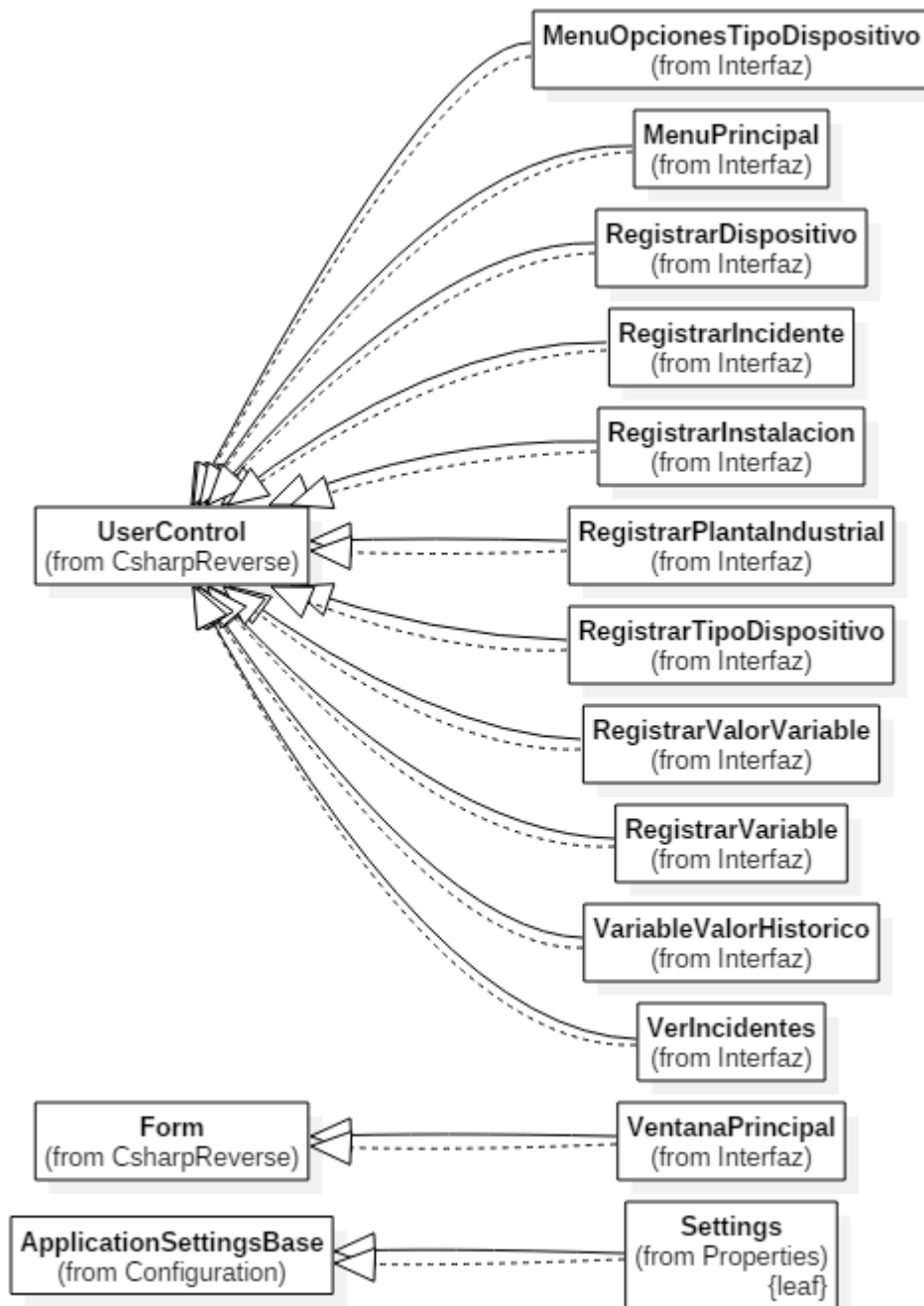
4.1 Diagrama de Dominio

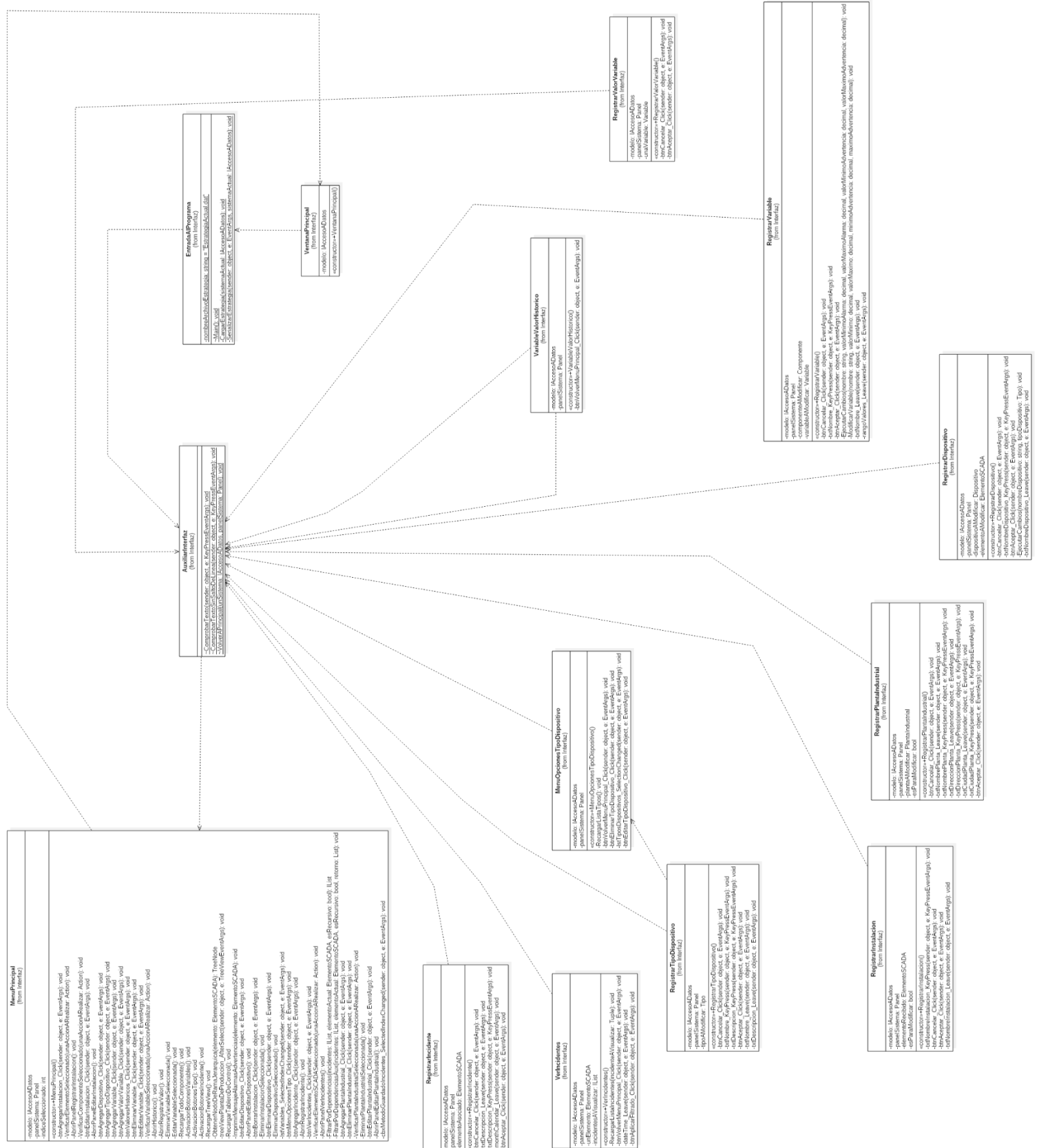


4.2 Diagrama de Excepciones

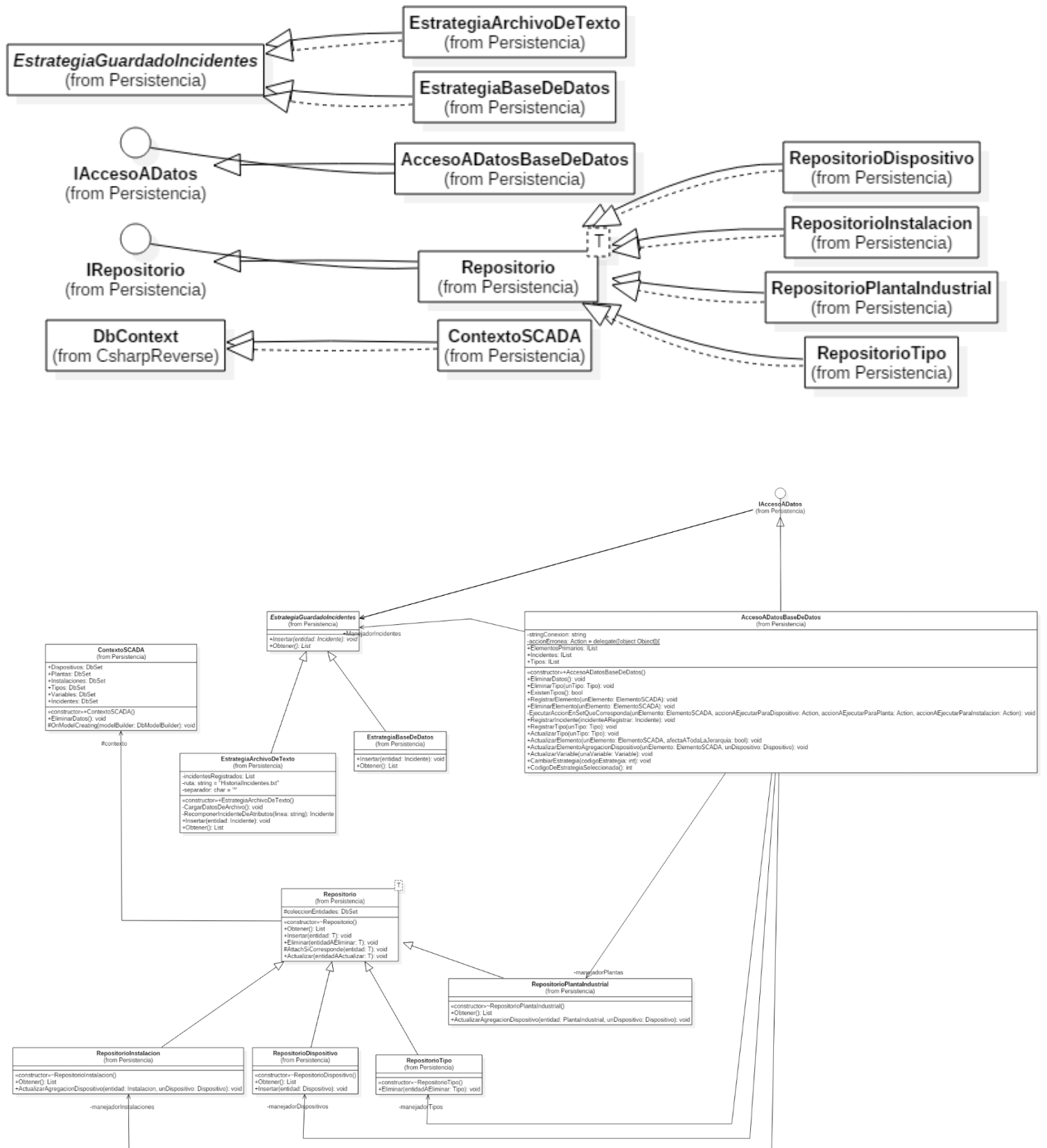


4.3 Diagrama de Interfaz



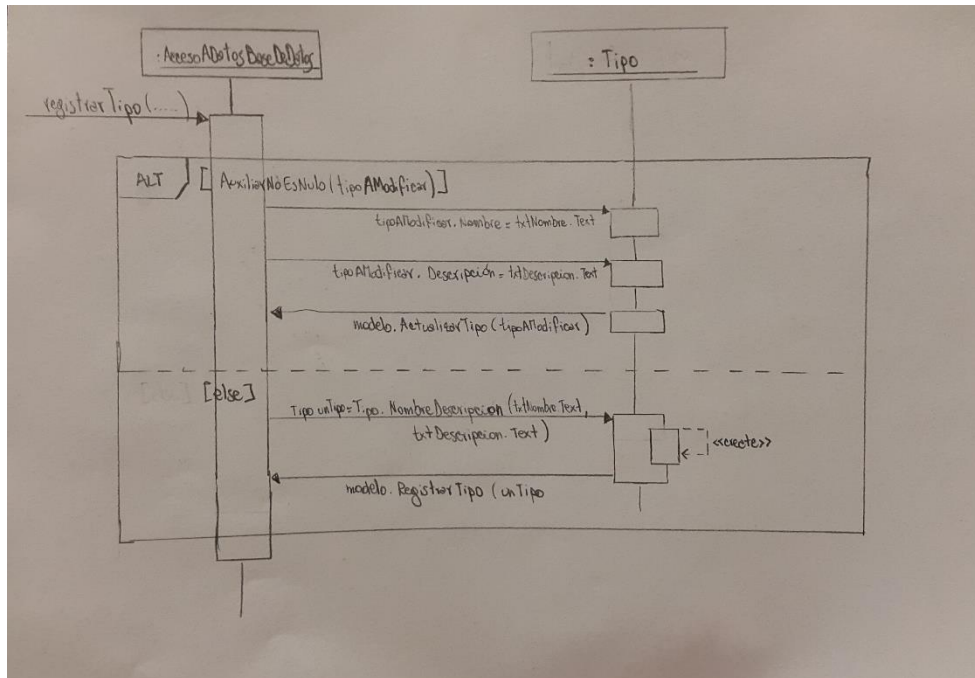


4.4 Diagrama de Persistencia

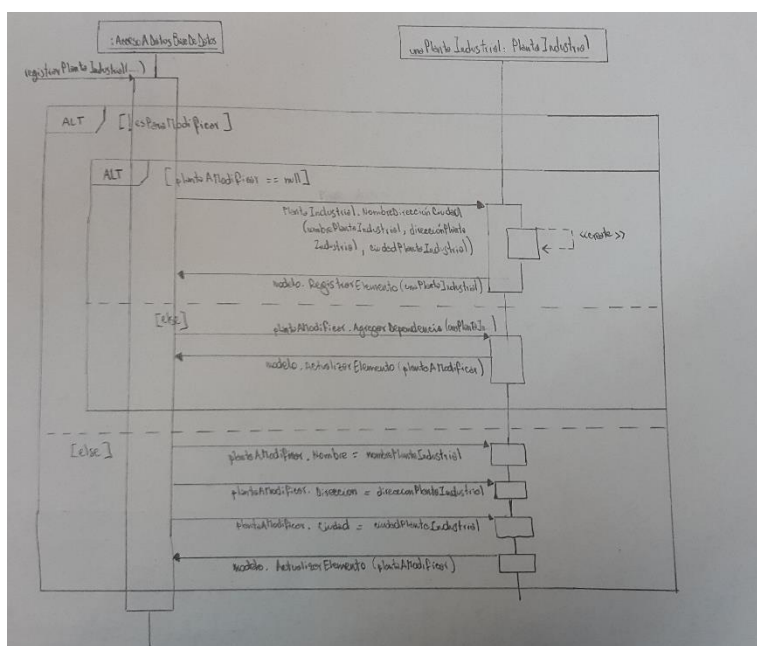


5. Diagramas de Interacción

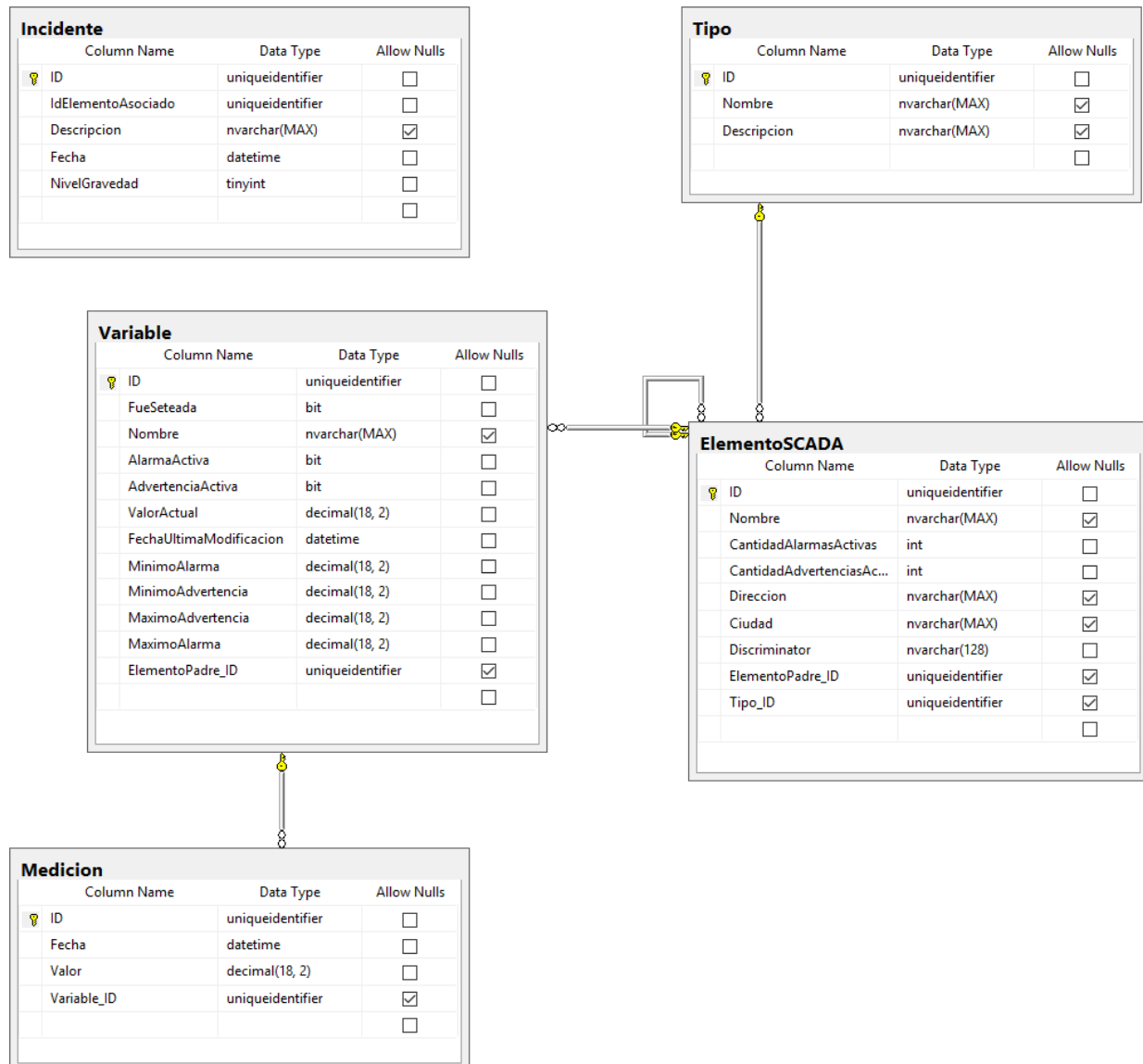
Registrar Tipo



Registrar Planta Industrial



















6. Modelo de Tablas de la estructura de la Base de Datos



7. Evidencia de Clean Code

Como parte de los requerimientos presentes en la propuesta para este obligatorio, en esta ocasión también, se planteó la necesidad de aplicar los distintos estándares que el libro “*Clean Code*”, de Robert C. Martin introduce. Así, a grandes rasgos al codificar las distintas funcionalidades del programa se intentó utilizar, por ejemplo, nombres de variables mnemotécnicos y representativos de lo que el objeto o método representara, así como el utilizar métodos estáticos que retornaran un objeto de cierta clase en lugar de simples constructores como forma de visualizar mejor los parámetros de los mismos, el evitar el uso de comentarios dentro del código, intentar reducir la profundidad de bloques *if* o *while* dentro de un mismo método, etc.

En cuanto a la metodología de desarrollo de TDD ya mencionada, ésta consistiría básicamente en implementar las pruebas unitarias de determinado método en forma previa a la codificación de este mismo, como forma de asegurar el buen funcionamiento del código escrito, que se ajuste precisamente a lo establecido para cierta función, y como forma también de documentar el comportamiento del mismo para posterior consulta. En comparación con el obligatorio anterior, como se verá, la aplicación de la metodología TDD fue parcial: fuertemente aplicada en la primera parte, más sobre el final del mismo dejada un tanto de lado por restricciones de tiempo y fechas de entrega, así por dificultades al momento de trabajar con la librería *Entity Framework*. Esto constituye un punto a mejorar para futuros

Graph	Description	Date	Author	Commit
	Se comenzó a hacer cambios en la interfaz para adecuar la misma a las nuevas funcionalidades requeridas	28 May, 2016 17:38	Juan Rodríguez <jr	3030f24
	Merge branch 'feature/AdvertenciasVariable' into develop	28 May, 2016 16:43	Sebastián Uriarte G	6ef5846
	[Refactoring]: Se intentó mejorar la calidad del código fuente, en cuanto a lo devuelto por la herramienta de análisis de código de VisualStudio.	28 May, 2016 16:38	Sebastián Uriarte G	61b6ceb
	[PruebasExitosas]: Se implementó el código necesario para hacer que el conjunto de pruebas anteriores fuera exitoso.	28 May, 2016 16:26	Sebastián Uriarte G	d4f584b
	[PruebasFallidas]: Se agregaron nuevas pruebas respecto de ciertos errores identificados a partir de la entrega anterior al modificar las Variables registradas,	28 May, 2016 15:49	Sebastián Uriarte G	0eabc5a
	Merge branch 'feature/RegistrarDatosIncidente' into develop	28 May, 2016 15:28	Juan Rodríguez <jr	de8a6f3
	Revert "[Refactoring]: Se arreglaron aspectos de claridad y calidad del código en la clase Incidente"	28 May, 2016 15:28	Juan Rodríguez <jr	579402d
	[Refactoring]: Se arreglaron aspectos de claridad y calidad del código en la clase Incidente	28 May, 2016 15:26	Juan Rodríguez <jr	e9c6052
	[Refactoring]: Correcciones a la prolijidad y claridad del código.	24 May, 2016 0:08	Sebastián Uriarte G	ebaf80b
	[PruebasExitosas]: Se implementó el código necesario para que el conjunto de pruebas anteriores fuera exitoso.	23 May, 2016 22:19	Sebastián Uriarte G	13b9386
	[NoCompila]: Se crearon pruebas orientadas al manejo de advertencias por componente, de forma similar a como funciona para alarmas.	22 May, 2016 18:59	Sebastián Uriarte G	3191163
	[Refactoring]: Se intentó hacer el código generado más legible, respetando los principios de Clean Code.	22 May, 2016 17:16	Sebastián Uriarte G	a111657
	[PruebasExitosas]: Se implementó el código necesario para que las pruebas anteriores pasaran.	21 May, 2016 18:45	Sebastián Uriarte G	48fafc6
	[Refactoring]: Se arreglaron aspectos de claridad y calidad del código en la clase Incidente	21 May, 2016 18:19	Juan Rodríguez <jr	a042fad
	[PruebasExitosas]: Se implementaron las funciones para el correcto funcionamiento del set de Incidente	21 May, 2016 17:57	Juan Rodríguez <jr	f20dc31
	[NoCompila]: Se realizaron cambios en las pruebas anteriores a raíz de una reinterpretación de lo requerido para implementar la funcionalidad pedida, en c	21 May, 2016 17:48	Sebastián Uriarte G	61b0279

Sebastián Uriarte Güimil, 194973

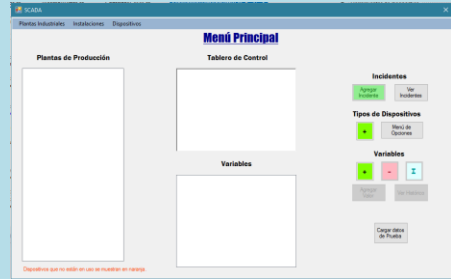



Juan Carlos Rodríguez Sappía, 171983

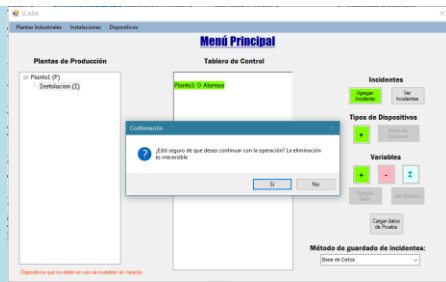
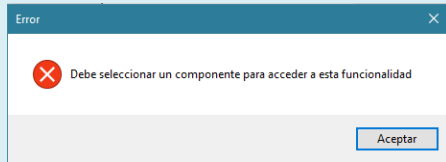
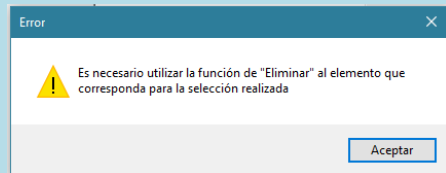
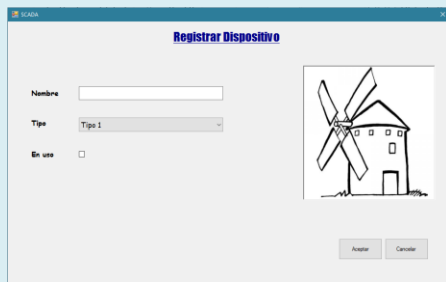
proyectos.

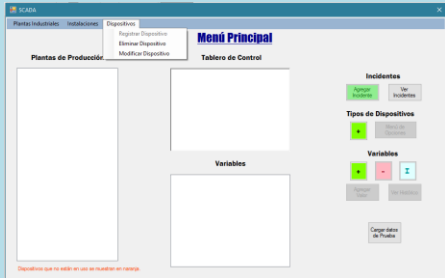
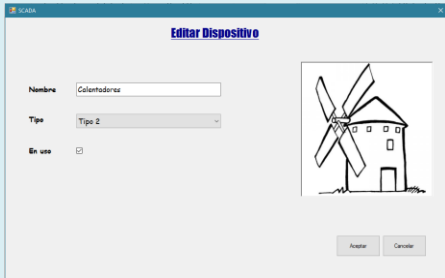
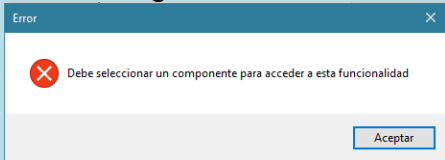
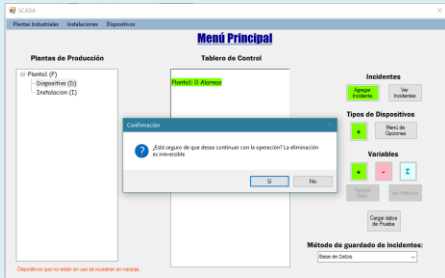
8. Pruebas

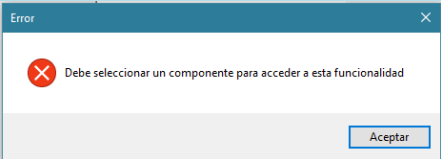
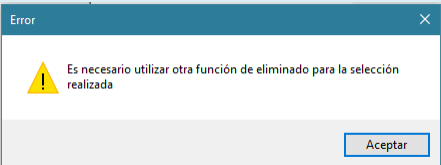
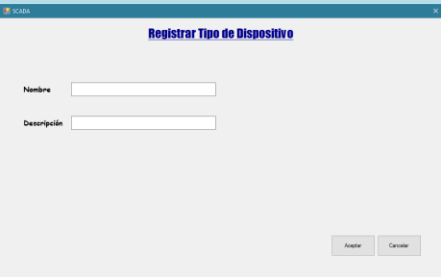

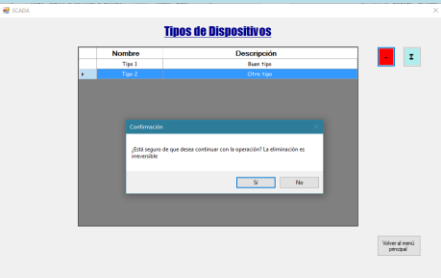
Funcionalidad	Variable Modificada	Dato Ingresado / Evento	Resultado Esperado	Resultado Obtenido
---------------	------------------------	----------------------------------	--------------------	-----------------------

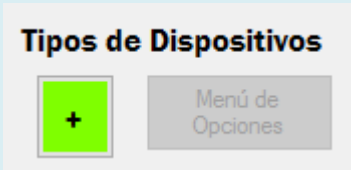
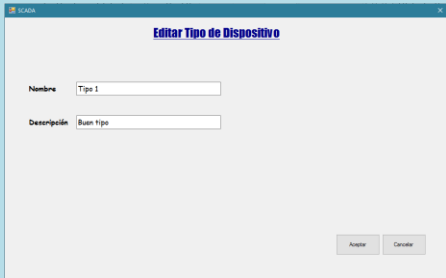
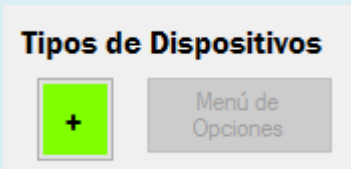
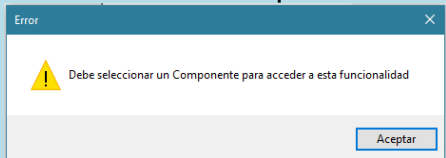
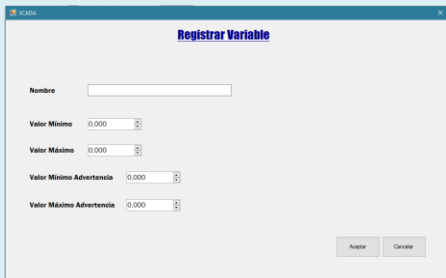
8.1 Datos de Prueba

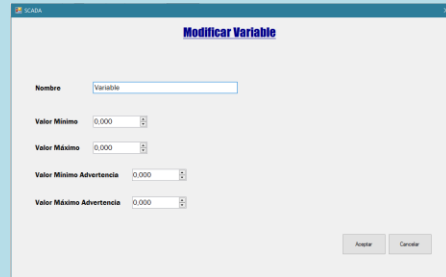

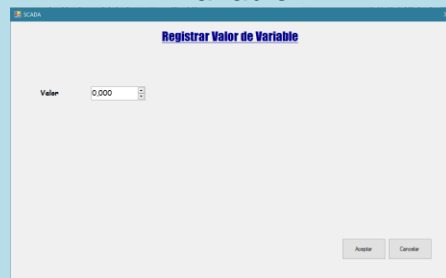

MenuPrincipal				OK
		Ingresar Instalación		OK
		Editar Instalación	Habiendo registrado por lo menos una Instalación 	OK
			Si no hay registrado una Instalación 	OK
		Eliminar Instalación	Habiendo registrado por lo menos una Instalación	OK

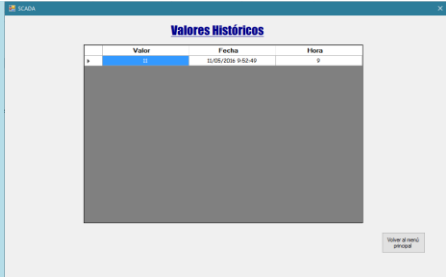
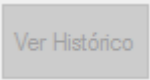
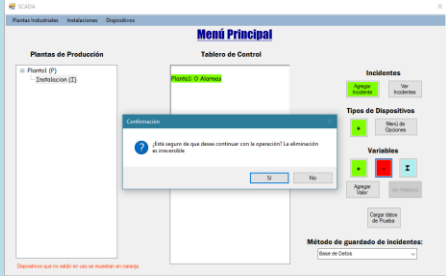

				
			<p>Si no hay registrada un Instalación</p>	OK
				
			<p>Si no hay seleccionada una Instalación</p>	OK
				
		<p>Ingresar Dispositivo</p>		OK

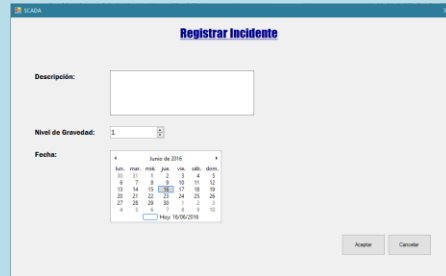
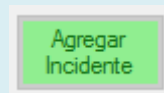

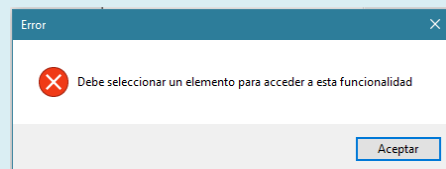
		<div>Si no hay registrado un Tipo de Dispositivo</div> <div></div> <div>(opción deshabilitada)</div> <div>OK</div>
		<div>Habiendo registro por los menos un Dispositivo</div> <div></div> <div>OK</div>
Editar Dispositivo		<div>Si no hay registrado un Dispositivo o no se seleccionó ningún elemento</div> <div></div> <div>OK</div>
		<div>Habiendo registrado por los menos un Dispositivo</div> <div></div> <div>OK</div>
Eliminar Dispositivo		<div>Si no hay registrado un Dispositivo o no se seleccionó</div> <div>OK</div>

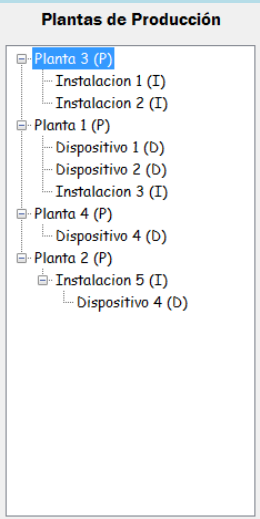
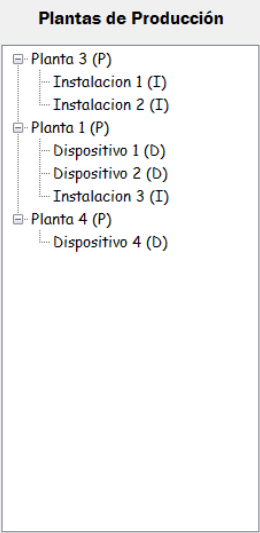
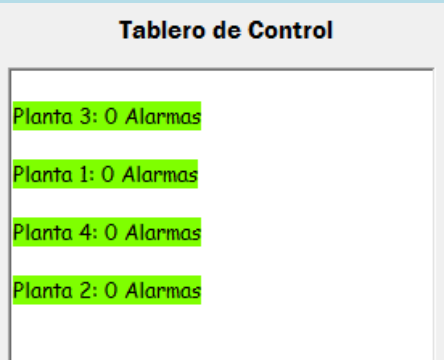
			ningún elemento		
			Si no hay seleccionado un Dispositivo		OK
		Insertar Tipo de Dispositivo			OK
		Menú de opciones			OK
		Eliminar Tipo de Dispositivo	Habiendo registrado por los menos un Tipo de Dispositivo		OK
			Si no hay registrado un Tipo de Dispositivo		OK

			 <p>(botón deshabilitado)</p>	
			<p>Habiendo registrado por lo menos un Tipo de Dispositivo</p> 	OK
		Editar Tipo de Dispositivo	<p>Si no hay registrado un Tipo de Dispositivo</p>  <p>(botón deshabilitado)</p>	OK
			<p>Si no hay registrado por lo menos un Componente</p>  <p>(botón deshabilitado)</p>	OK
		Ingresar Variable	<p>Habiendo registrado por lo menos un Elemento</p> 	OK

		Editar Variable	Habiendo seleccionado una Variable	OK
				OK
		Agregar Valor a Variable	Si no hay registrada una Variable o no se seleccionó una	OK
			 (botón deshabilitado)	OK
			Habiendo seleccionado una Variable	OK
				OK
		Ver Históricos Valores Variables	Si no hay registrada una Variable o no se seleccionó una	OK
			 (botón deshabilitado)	OK
			Habiendo registrado por lo menos dos Valores de Variables y seleccionado dicha Variable	OK

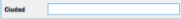

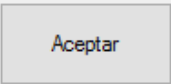
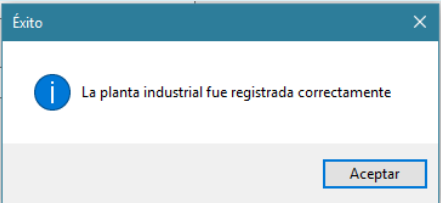
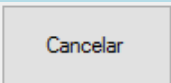
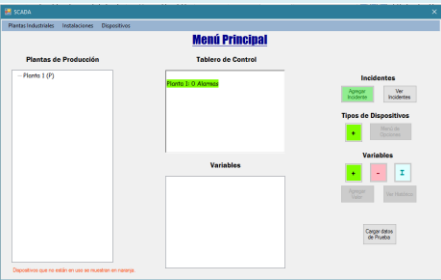
				
			<p>Si no hay registrados los Valores para dicha Variable</p>  <p>(botón deshabilitado)</p>	OK
		Eliminar Variable	<p>Habiendo registrado por lo menos una Variable</p> 	OK
			<p>Si no hay registrada una Variable o no se seleccionó una</p>  <p>(botón deshabilitado)</p>	OK

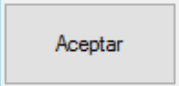
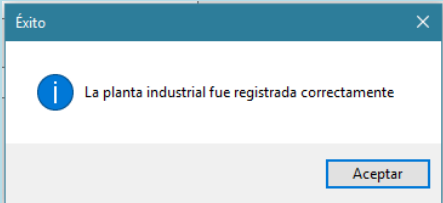
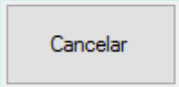
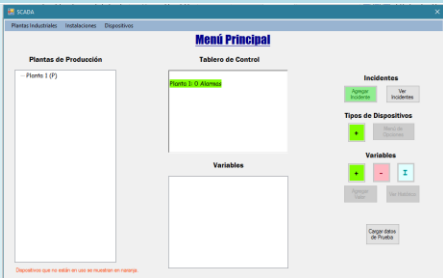
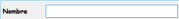
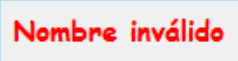
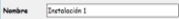
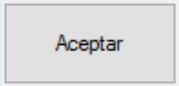
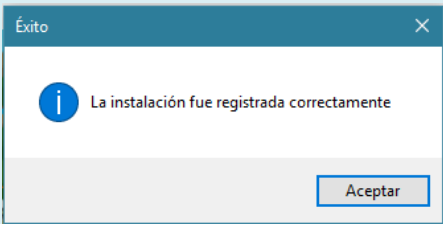
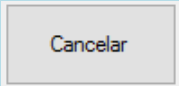

		Agregar Incidente	Habiendo seleccionado un elemento de la Planta de Producción		OK
			Si no hay seleccionado ningún elemento de la Planta de Producción	 (botón deshabilitado)	OK
		Ver Incidentes	Habiendo seleccionado un elemento de la Planta de Producción		OK
			Si no hay seleccionado ningún elemento de la Planta de Producción		OK

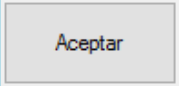
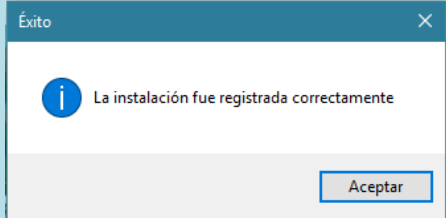
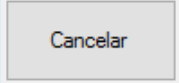
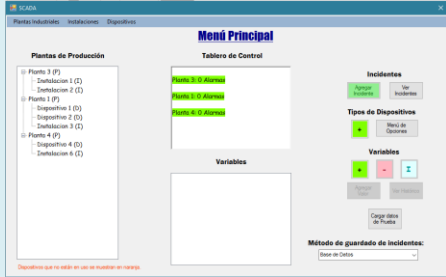
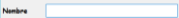


	treeView PlantaDe Produccion	Agregar Planta Industrial o Instalación o Dispositivo		OK
		Borrar una Planta Industrial o Instalación o Dispositivo		OK
	IstTableroDe Control	Habiendo registrado una Planta Industrial o una Instalación o un Dispositivo	<p>Sin Alarmas encendidas</p> 	OK

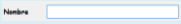

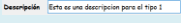
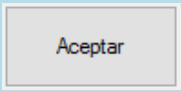
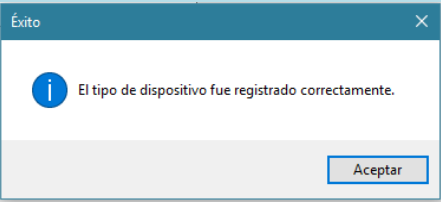
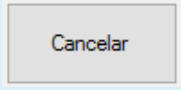
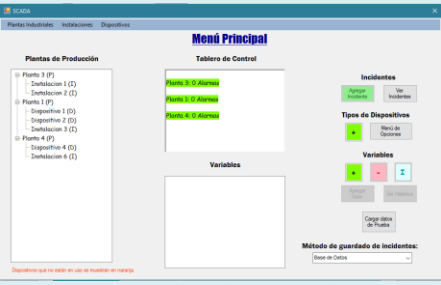
			Con Alarmas encendidas	
			<div> Tablero de Control <div> Extracción: 1 Alarmas Clarificación: 0 Alarmas Evaporación: 0 Alarmas Centrifugación: 0 Alarmas </div> </div>	OK
	IstVariables	Si no hay Planta Industrial o Dispositivo de primer nivel registrado	<div> Tablero de Control <div></div> </div>	OK
		Habiendo registrado una Variable	<div> Variables <div> Presión: N/A (10 - 30) Temperatura: N/A (0 - 200) </div> </div>	OK
		Si no hay registrada una Variable	<div> Variables <div></div> </div>	OK

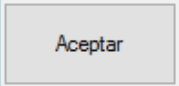
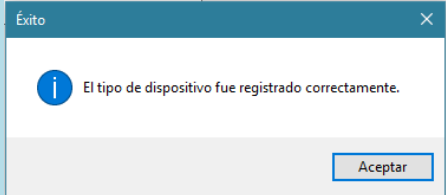
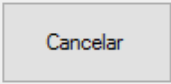


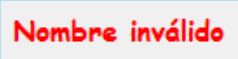

Registrar Planta Industrial		Si la Instalación o Dispositivo seleccionad o no tiene Variables	<div>Variables</div> <div>Sin datos a mostrar</div>	OK
	txtNombre Planta	<div>Nombre <input type="text"/></div>	Nombre inválido	OK
		“\$%\$”	No se va a permitir el ingreso de caracteres especiales	OK
		<div>Nombre <input type="text" value="Instalación 1"/></div>		OK
	txtDireccion Planta	<div>Dirección <input type="text"/></div>	Dirección inválida	OK
		“\$%\$”	No se va a permitir el ingreso de caracteres especiales	OK
		<div>Dirección <input type="text" value="Montevideo 1234"/></div>		OK

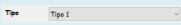
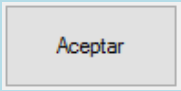
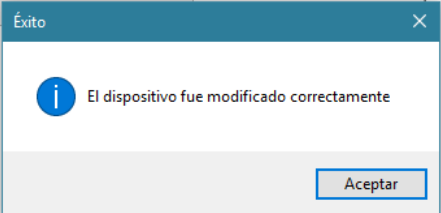
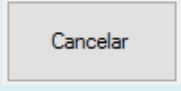
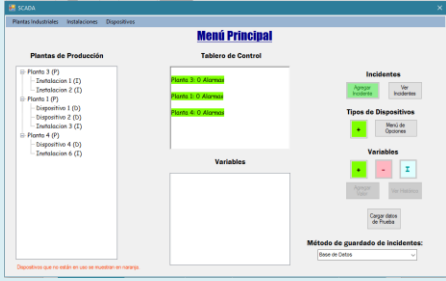
	txtCuidadPlanta		Ciudad inválida	OK
		“\$%\$”	No se va a permitir el ingreso de caracteres especiales	OK
				OK
	btnAceptar			OK
	btnCancelar			OK
Editar Planta Industrial	txtNombre Planta	Mismas que en Registrar Planta Industrial		OK
	txtDireccion Planta	Mismas que en Registrar Planta Industrial		OK

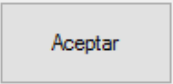
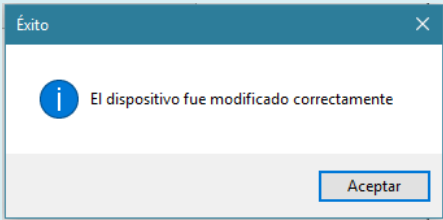
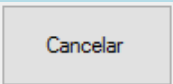
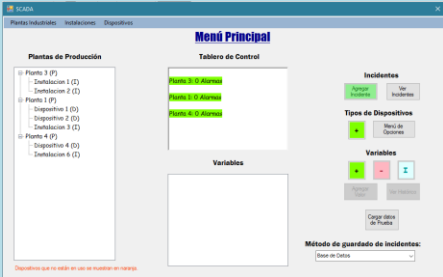


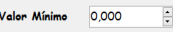
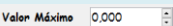
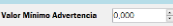


	txtCiudad Planta	Mismas que en Registrar Planta Industrial		OK
	btnAceptar			OK
	btnCancelar			OK
Registrar Instalación	txtNombre Instalacion			OK
		“\$%\$”	No se va a permitir el ingreso de caracteres especiales	OK
				OK
	btnAceptar			OK
	btnCancelar			OK


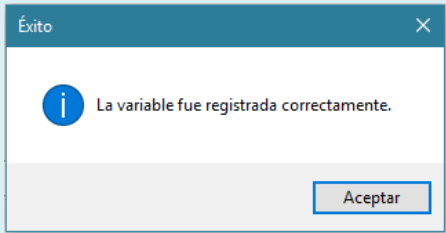
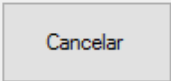

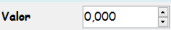
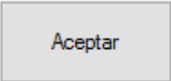
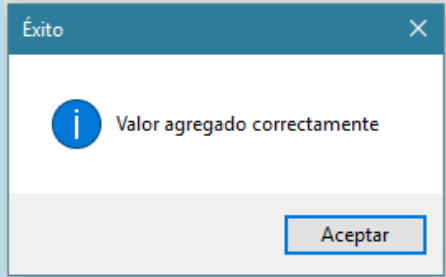
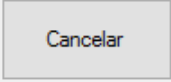
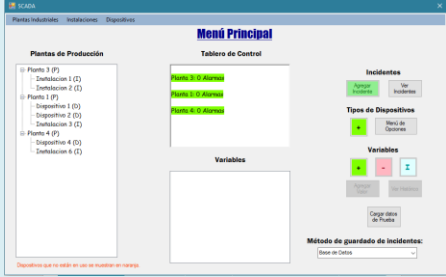
Editar Instalación	txtNombre Instalación	Mismas que en Registrar Instalación		OK
	btnAceptar			OK
	btnCancelar			OK
Registrar Tipo de Dispositivo	txtNombre			OK
		“\$%\$”	No se va a permitir el ingreso de caracteres especiales	OK
				OK

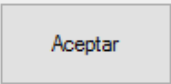
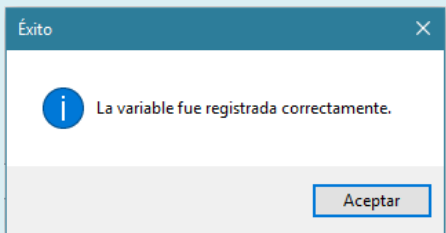
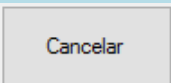
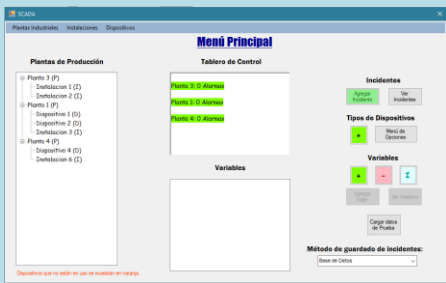
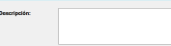


	txtDescripcion			OK
		“\$%\$”	No se va a permitir el ingreso de caracteres especiales	OK
				OK
	btnAceptar			OK
	btnCancelar			OK
Editar Tipo de Dispositivo	txtNombre	Mismas que en Registrar Tipo de Dispositivo		OK


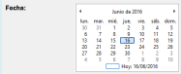
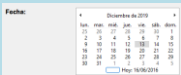
	txtDescripcion	Mismas que en Registrar Tipo de Dispositivo		OK
	btnAceptar			OK
	btnCancelar			OK
Registrar Dispositivo	txtNombre Dispositivo			OK
		“\$%\$”	No se va a permitir el ingreso de caracteres especiales	OK
				OK

	cbxTipo Dispositivo			OK
	btnAceptar			OK
	btnCancelar			OK
Editar Dispositivo	txtNombre Dispositivo	Mismas que en Registrar Dispositivo		OK
	cbxTipo Dispositivo	Mismas que en Registrar Dispositivo		OK
	chkEnUso	Mismas que en Registrar Dispositivo		OK

	btnAceptar			OK
	btnCancelar			OK
Registrar Variable	txtNombre		Nombre inválido	OK
		“\$\$\$”	No se va a permitir el ingreso de caracteres especiales	OK
				OK
	numMin			OK
	numMax			OK
	minAdv			OK
	maxAdv			OK
			El rango de valores es inválido.	OK

	btnAceptar			OK
	btnCancelar			OK
Agregar Valor A Variable	numValor			OK
	btnAceptar			OK
	btnCancelar			OK

Editar Variable	txtNombre	Mismas que en Registrar Variable		OK
	numMin	Mismas que en Registrar Variable		OK
	numMax	Mismas que en Registrar Variable		OK
	btnAceptar			OK
	btnCancelar			OK
Registrar Incidente	txtDescripcion			OK
		“\$%\$”	No se va a permitir el ingreso de caracteres especiales	OK
				OK

	numValor			OK
	monthCalendar			OK
			La fecha no puede ser posterior a la actual	OK

8.2 Cobertura de Pruebas Unitarias

A continuación, se muestran los resultados de la ejecución de las pruebas generadas como parte de este obligatorio:

alumnoFI_FIW10LC1111 2016-06-16 1...	171	17.45 %	809	82.55 %
dominio.dll	42	7.18 %	543	92.82 %
Dominio	42	7.18 %	543	92.82 %
Auxiliar	0	0.00 %	60	100.00 %
Componente	1	3.85 %	25	96.15 %
Dispositivo	6	13.95 %	37	86.05 %
ElementoSCADA	10	11.49 %	77	88.51 %
Incidente	0	0.00 %	39	100.00 %
Instalacion	1	4.55 %	21	95.45 %
ManejadorDependenciasC...	0	0.00 %	53	100.00 %
Medicion	9	56.25 %	7	43.75 %
PlantaIndustrial	1	2.17 %	45	97.83 %
Tipo	0	0.00 %	53	100.00 %
Variable	14	10.00 %	126	90.00 %
excepciones.dll	0	0.00 %	10	100.00 %
persistencia.dll	129	33.51 %	256	66.49 %

Pruebas: superadas (199)	
✓ ActivacionAdvertenciaMo...	< 1 ms
✓ ActivacionAlarmaModifica...	< 1 ms
✓ AgregarDependenciaCicli...	< 1 ms
✓ AgregarDependenciaCicli...	< 1 ms
✓ AgregarDependenciaDisp...	< 1 ms
✓ AgregarDependenciaPlant...	< 1 ms
✓ AgregarDependenciaPlant...	< 1 ms
✓ AgregarDependenciaTest1	1 ms
✓ AgregarDependenciaTest2	< 1 ms
✓ AgregarDependenciaTest3	< 1 ms
✓ AgregarDependenciaTest4	< 1 ms
✓ AgregarVariablePlantaInd...	< 1 ms
✓ AgregarVariableTest1	< 1 ms
✓ AgregarVariableTest2	< 1 ms
✓ AgregarVariableTest3	< 1 ms
✓ AgregaValoresAHistoricoT...	< 1 ms
✓ AgregaValoresAHistoricoT...	< 1 ms
✓ AgregaValoresAHistoricoT...	< 1 ms
✓ AlarmasActivasDispositivo...	< 1 ms
✓ AlarmasActivasDispositivoTe...	3 ms
✓ AlarmasActivasDispositivo...	< 1 ms
✓ AlarmasActivasDispositivo...	< 1 ms
✓ CompareToElementoSCA...	< 1 ms
✓ CompareToElementoSCA...	< 1 ms
✓ CompareToElementoSCA...	< 1 ms

En este caso, se observa que, si bien se estima dentro de parámetros aceptables la cobertura de pruebas unitarias conseguida, en comparación con el obligatorio anterior la misma ha descendido. Esto se considera, según se explicó en secciones anteriores, se debe a la complejidad anteriormente mencionada que tuvo el trabajar con *Entity Framework* que causó que el desarrollo del proyecto se desviara en las últimas etapas de lo que es la metodología TDD. Se agregaron una serie de propiedades o atributos, necesarios para que el mismo funcionara, que al no verse identificado su manejo con ninguna prueba funcional harían descender la cobertura. Sin embargo, se estima que esta está dentro de lo esperable y aceptado.