*Serie Científica de la Universidad de las Ciencias Informáticas*
Vol. 9, No. 3, Marzo, 2016
ISSN: 2306-2495 | RNPS: 2343
http://publicaciones.uci.cu
Pág.149-163

Tipo de artículo: Artículo original
Temática: Realidad virtual
Recibido: 28/02/2016 | Aceptado: 28/03/2016

# Versión 2.0 del módulo Snap de la herramienta AsiXMec

# *Version 2.0 of the Snap module of the AsiXMec tool*

**Midiala Baños Artigas [1*], Marvyn Amado Rodríguez [1]**

[1] Universidad de las Ciencias Informáticas. Carretera San Antonio Km 2 ½. Boyeros. CP.:19370. {midi, mamarquez}@uci.cu

[*] Autor para correspondencia: midi@uci.cu

**Resumen**

El diseño computarizado ha tomado gran auge en todas las esferas sociales. Es por eso que las grandes y pequeñas empresas dedicadas al desarrollo de software, hacen todo lo posible por llevar al mercado herramientas de Diseño Asistido por Ordenador (CAD por sus siglas en inglés) con las mejores funcionalidades, con el objetivo de satisfacer el mayor número de necesidades de los clientes. El presente trabajo está orientado al desarrollo de una segunda versión del módulo *Snap* de la herramienta de diseño computarizado *AsiXMec*, que se desarrolla en el proyecto *Diseño y Simulación de Estructuras Mecánicas* de la Universidad de las Ciencias Informáticas. Este software tiene como propósito apoyar a los ingenieros mecánicos en el diseño de prototipos virtuales de estructuras mecánicas, para su posterior ensamble en empresas manufactureras. La nueva versión surge debido a que al adicionar nuevos *snaps* o modificar los existentes, es necesario compilar toda la aplicación, debido al alto acoplamiento que existe entre el módulo *Snap* y el núcleo del sistema. La propuesta de solución se basa en el uso de *plugins* para reducir el acoplamiento en el software *AsiXMec*, obteniéndose una versión mejorada que permite a *AsiXMec* cargar dinámicamente los *snaps* una vez que se inicializa. Esto le permite al usuario seleccionar desde una interfaz visual los *snaps* que desee que se ejecuten durante la etapa de diseño.

**Palabras clave:** acoplamiento; AsiXMec; Diseño Asistido por Ordenador; plugin; *snap*

***Abstract***

*The computerized design has exploded in all social spheres. That is why large and small companies engaged in software development, make every effort to bring to market tools of computer aided design (CAD for its acronym in english) with the best features, in order to satisfy the greatest number of customer needs. This work is aimed at developing a second version of snap module AsiXMec computerized design tool, developed in the project design and simulation of mechanical structures of the university of information science. This software aims to support*

*Serie Científica de la Universidad de las Ciencias Informáticas*
Vol. 9, No. 3, Marzo, 2016
ISSN: 2306-2495 | RNPS: 2343
http://publicaciones.uci.cu
Pág.149-163

*mechanical engineers in the design of virtual prototypes of mechanical structures for subsequent assembly in manufacturing companies. The new version arises because snaps to add new or modify existing ones, you need to compile the entire application due to high coupling between the snap module and the kernel. The proposed solution is based on the use of plugins to reduce coupling in AsiXMec software, resulting in an improved version that allows dynamically load AsiXMec snaps once initialized. This allows the user to select from a visual interface the snaps you want to run during the design stage.*

***Keywords:*** *coupling; AsiXMec; Computer Aided Design; plugin; snap*

## Introducción

En la actualidad, el diseño de nuevos productos o la modificación de los ya existentes, se ha convertido en un elemento fundamental para la mejora de la capacidad de innovación y competitividad de las empresas industriales. La forma de lograr un diseño con mayor precisión, a un menor costo y que mejore los estándares de producción, ha ido incrementándose en los últimos años (Saldaña Pomazunco, 2012). Tradicionalmente se hacía uso de un lápiz y papel para realizar los dibujos, y si en algún momento el documento no tenía la calidad requerida, era necesario reemplazarlo por otro, lo que traía como consecuencia la pérdida de recursos y tiempo de trabajo (Salomon, 2011). Sin embargo, al desarrollarse las Tecnologías de la Informática y la Comunicación (TIC), fueron apareciendo nuevos enfoques para resolver ese tipo de problemas.

Con el objetivo de ayudar a profesionales o aficionados en la creación de diseños más eficientes, fueron creadas las herramientas de Diseño Asistido por Ordenador (CAD por sus siglas en inglés), las cuales proporcionan una interfaz gráfica para que el usuario interactúe con ellas a través de dispositivos de entrada, como un teclado, un mouse o un lápiz óptico. Los campos de aplicación de los sistemas CAD pueden ser variados, como la ingeniería, la arquitectura, el diseño automotriz, el diseño gráfico, la publicidad, el diseño de moda y la medicina (Jackson, y otros, 2013).

Varias empresas se han destacado a través de los años por crear aplicaciones CAD con un gran número de funcionalidades, con el objetivo de apoyar a los usuarios en el diseño computarizado. Entre ellas se encuentran *Parametric Technology Corporation (*PTC*), Dassault Systèmes* y Autodesk. Esta última se funda en la década de los años 80 del siglo pasado y es en la actualidad una de las empresas líderes en el mercado de herramientas CAD. Autodesk se dedica al desarrollo de software de diseño en 2D y 3D para las industrias manufactureras, de infraestructuras, de construcción, así como en la realización de productos para el entretenimiento, destacándose en este último campo las herramientas de diseño Maya, 3ds Max *y* Blender, siendo la última mencionada una alternativa libre.

*Serie Científica de la Universidad de las Ciencias Informáticas*
Vol. 9, No. 3, Marzo, 2016
ISSN: 2306-2495 | RNPS: 2343
http://publicaciones.uci.cu
Pág.149-163

## Técnicas de posicionamiento en herramientas de diseño gráfico computarizado

Un elemento que tienen en común varias de las herramientas de diseño gráfico computarizado, son las técnicas para posicionar elementos en el área de trabajo, las que permiten al usuario diseñar objetos con mayor calidad y exactitud. Entre estas técnicas se destacan las guías, las reglas, las restricciones, el grid o cuadrícula, y los snaps. Cada una de las herramientas CAD que se mencionaron con anterioridad hacen uso de algunas de estas técnicas; sin embargo, por el auge que han tomado y las ventajas que facilitan a un usuario, otras herramientas de diseño como CorelDRAW, una aplicación informática de diseño gráfico vectorial, el editor de imágenes Adobe Photoshop y la suite de oficina LibreOffice, también las emplean.

- Guías

Las guías son una técnica para ajustar los objetos con precisión, las cuales se emplean en herramientas de diseño y tratamiento de imágenes como Adobe Photoshop. Las operaciones con guías se pueden realizar de diferentes maneras, por lo que hay varias formas para realizar una misma operación. Se presentan como líneas verticales u horizontales que el usuario puede crear o desplazar, teniendo en cuenta lo que esté realizando (Editorial Vértice, 2010).

- Reglas

Las reglas se utilizan para colocar los objetos de forma precisa (Figura 1). Por lo general, se muestran en la parte superior y lateral izquierda del espacio de trabajo. Al mover el mouse, se puede apreciar una línea discontinua tanto vertical como horizontal, que indica el origen de coordenadas del puntero (Editorial Vértice, 2010).

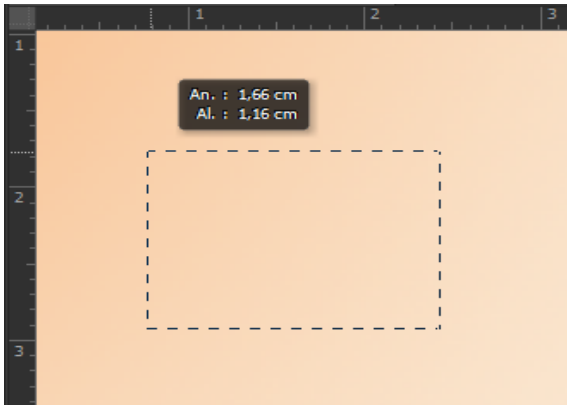


Figura 1. Reglas en Adobe Photoshop

*Serie Científica de la Universidad de las Ciencias Informáticas*
Vol. 9, No. 3, Marzo, 2016
ISSN: 2306-2495 | RNPS: 2343
http://publicaciones.uci.cu
Pág.149-163

- Restricciones

Las restricciones ayudan en el proceso de manipulación directa de un objeto, ya que proporcionan al usuario otra vía para ubicar los objetos de forma precisa, lo cual constituye una funcionalidad útil en los sistemas de diseño y modelación. Estas pueden aparecer en un menú (Figura 2) donde es posible seleccionar la restricción concreta con la que se desea trabajar o en forma de símbolos (Figura 3) (Shih, 2014; Lockhart, 2013).

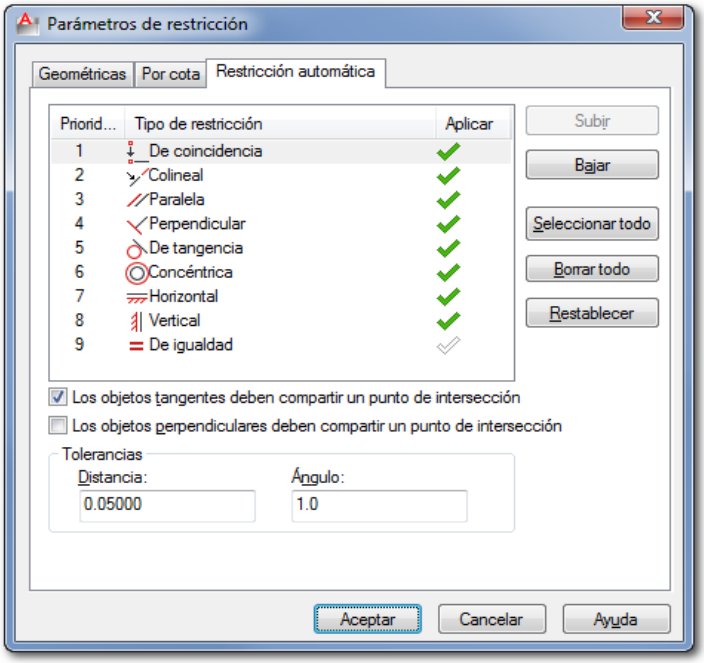


Figura 2. Menú de restricciones en la herramienta Autodesk AutoCAD


Figura 3. Restricciones en la herramienta Autodesk AutoCAD

*Serie Científica de la Universidad de las Ciencias Informáticas*
Vol. 9, No. 3, Marzo, 2016
ISSN: 2306-2495 | RNPS: 2343
http://publicaciones.uci.cu
Pág.149-163

- *Grid*

El grid o cuadrícula es un elemento que ayuda a colocar objetos de manera simétrica (Figura 4). Su representación es una rejilla que divide el área de trabajo en rectángulos de iguales dimensiones. El usuario puede definir la dimensión de cada rectángulo, y puede seleccionar si desea mostrar el *grid* mientras está trabajando. Algunas de las propiedades de la cuadrícula que se pueden configurar son el color, el estilo de las líneas (continua o discontinua) y el número de divisiones (Editorial Vértice, 2010; Lockhart, 2013).

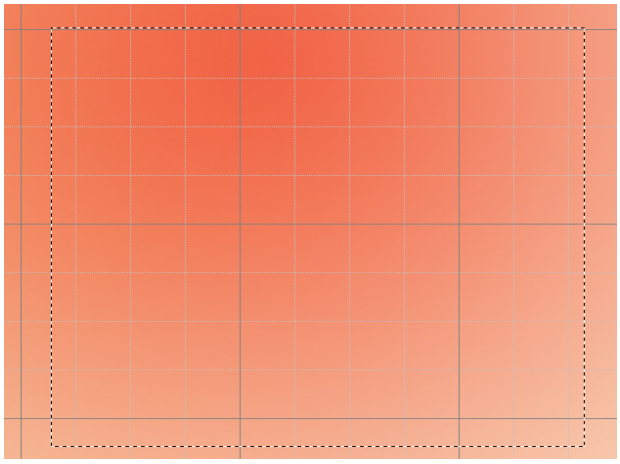

Figura 4. Grid o cuadrícula en Adobe Photoshop

## Los *snaps* en herramientas de diseño gráfico computarizado

En las herramientas CAD una de las técnicas empleadas por los usuarios para lograr precisión en sus diseños, son los *snaps* o puntos de referencia a objetos. Un sistema CAD permite a un usuario emplear los *snaps* para facilitar la colocación precisa al mover un objeto a una ubicación deseada. El término objeto se refiere a las entidades con las cuales puede trabajar un diseñador; así como su combinación para la creación de otras entidades (Jackson, y otros, 2013). Algunas herramientas de diseño computarizado como AutoCad e Inventor, utilizan la expresión *object snap* (Shih, 2014) para referirse a los *snaps*, que será el término adoptado en el presente trabajo.

Con el objetivo de que exista una mejor comprensión del término, se describe a continuación el comportamiento del *snap* perpendicular de la herramienta AsiXMec (Figura 5). Cuando un usuario está diseñando, al dar clic sobre el *sketcher*, se captura el punto del cursor y se guardan sus coordenadas. Posteriormente se proyecta ese punto sobre la entidad que ya se dibujó en el *sketcher*, con el objetivo de buscar aquel punto en el cual se obtendrá un objeto que cumpla con las características deseadas, adquiriendo así un comportamiento de atracción y facilitando la exactitud del

*Serie Científica de la Universidad de las Ciencias Informáticas*
Vol. 9, No. 3, Marzo, 2016
ISSN: 2306-2495 | RNPS: 2343
http://publicaciones.uci.cu
Pág.149-163

boceto realizado. En este ejemplo la entidad sobre la cual se proyecta el punto es la entidad recta, y el punto que se desea encontrar es aquel donde la entidad que se desea dibujar, es perpendicular a la primera.

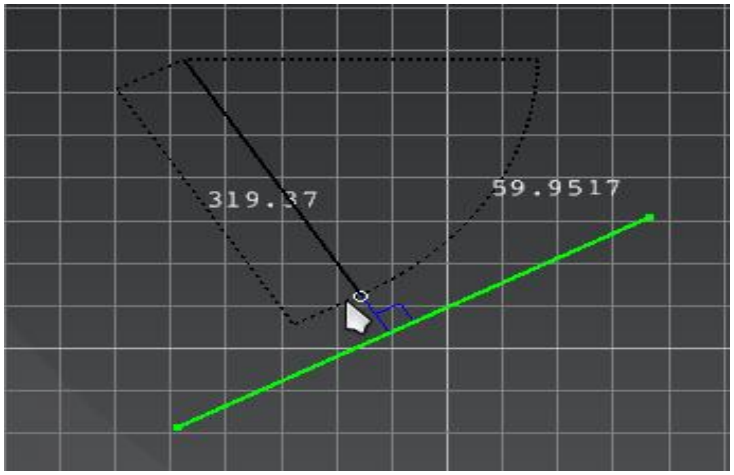


Figura 5. Visualización del *snap* paralelo en la herramienta *AsiXMec*

Muchos de los software CAD que se dedican al diseño y modelado de piezas mecánicas, edificaciones, carrocerías o productos cinematográficos como películas son privativos, lo cual dificulta a los especialistas su acceso, pues sus licencias comerciales son de un alto costo. Ante esta situación surgen nuevos sistemas de diseño que son de código abierto, como es el caso de las herramientas LibreCAD y Blender. Otra herramienta que se desea que en un futuro sea de código abierto, pues aun su licencia no se ha definido es *AsiXMec*, un sistema que se desarrolla en el proyecto Diseño y Simulación de Estructuras Mecánicas (DISEM) en el Centro de Entornos Interactivos 3D (VERTEX), perteneciente a la Facultad 5 de la Universidad de las Ciencias Informáticas (UCI). Este software surge con el propósito de apoyar el diseño de prototipos virtuales de estructuras mecánicas, como elemento de apoyo para su posterior construcción en empresas manufactureras.

*AsiXMec* es un modelador geométrico y paramétrico basado en historial de operaciones. Algunas de las funcionalidades que brinda son manipular, crear, editar, importar o exportar archivos compatibles con otros sistemas para Linux como Salome-Meca, DraftSight, *LibreCAD*, o para Windows como AutoCAD, SolidWorks e Inventor. Entre los módulos que lo componen se encuentra el *sketcher* o boceto, el cual consiste en un área de trabajo donde se crean o modifican entidades geométricas de dos dimensiones (arco, línea, polígono, círculo, punto, rectángulo y elipse). Es en el *sketcher* donde un diseñador define las características que puede tener una pieza mecánica.

*Serie Científica de la Universidad de las Ciencias Informáticas*
Vol. 9, No. 3, Marzo, 2016
ISSN: 2306-2495 | RNPS: 2343
http://publicaciones.uci.cu
Pág.149-163

Un ingeniero mecánico que se encuentre en pleno proceso de diseño de una pieza, puede hacer uso de las funcionalidades que brindan otros de sus módulos, como es el caso del módulo *Snap*. Actualmente, este módulo en el software *AsiXMec* presenta ciertas dificultades que atentan contra la calidad del software. Una de ellas consiste en la inserción de un nuevo *snap* (como es el caso del *snap* tangente), que son de gran utilidad para asistir a los usuarios en sus diseños o mejorar los que ya existen; para realizar el proceso de inserción es necesario modificar el núcleo y compilar completamente el sistema. Esto trae consigo, que, si en un futuro se brindan actualizaciones del módulo, es obligatorio proveer el software en su totalidad. Por tal motivo el objetivo de esta investigación es desarrollar la versión 2.0 del módulo *Snap* para apoyar el proceso de creación de entidades 2D en el *sketcher* del producto *AsiXMec*.

## Materiales y métodos

Entre las funcionalidades que brinda el sistema *AsiXMec* se encuentra el soporte para plugins, además de contar con una interfaz *ribbon* en la cual el especialista puede seleccionar la opción que desee, para el diseño y modelado de una pieza. Cuenta con un *solver* de restricciones, permite el diseño asistido por snaps e identificación automática de caras. Su interfaz gráfica es sencilla lo cual la hace de fácil manejo por los usuarios, además de brindar soporte de internacionalización para los idiomas Inglés y Español. Entre los módulos que componen la herramienta se encuentra el módulo *Snap*, el cual contiene actualmente 15 snaps que ayudan al usuario a lograr mayor precisión en sus bocetos, y de los cuales solo se muestran 11 al usuario. Estos *snaps* están implementados haciendo uso de las funcionalidades que brinda la biblioteca *Open CASCADE Community Edition* (OCE), un paquete de desarrollo de software robusto y avanzado, orientado a la visualización y modelado en 3D asistido por ordenador. Incluye un conjunto de bibliotecas implementadas en C++ que prestan servicios para la superficie 3D y modelado de sólidos, visualización, intercambio de datos y la creación rápida de aplicaciones (Arcia Salazar, 2011; Open CASCADE SAS, 2015).

El módulo en cuestión presenta dificultades que atentan contra la calidad del software, debido a que existe una gran relación entre el mismo y otros de la herramienta, como es el caso de los módulos de las entidades y las restricciones. Así, por ejemplo, al pintarse en el *sketcher* un *snap*, este devuelve tres elementos con los cuales trabaja el módulo de las restricciones. Estos elementos son el tipo de entidad, el tipo de restricción que se puede usar y el punto que se proyecta sobre la entidad con la cual se trabaja.

*Serie Científica de la Universidad de las Ciencias Informáticas*
Vol. 9, No. 3, Marzo, 2016
ISSN: 2306-2495 | RNPS: 2343
http://publicaciones.uci.cu
Pág.149-163

Actualmente, todos los *snaps* se cargan al iniciar la aplicación, por lo que un usuario no puede decidir si los puede cargar o no, ya que esta funcionalidad se encuentra fusionada con el código del proyecto. Teniendo en cuenta las condiciones actuales del módulo *Snap*, es necesaria la búsqueda de una vía factible que permita reducir el acoplamiento en este software con el propósito de aumentar su calidad.

### *Plugins*

El término *plugin* no forma parte del diccionario de la Real Academia Española (RAE). Se trata de un concepto de la lengua inglesa que puede entenderse como inserción y que se emplea en el campo de la informática. Un *plugin* es un complemento el cual una vez adicionado a un software, le incorpora una nueva funcionalidad. Lo habitual es que el *plugin* se ejecute mediante el software principal, con el que interactúa a través de una interfaz (IBM Corporation , 2012; Godoy, 2011).

Algunas ventajas que ofrecen estos complementos es que facilitan la colaboración de desarrolladores externos con el software, además de que reducen su acoplamiento, pues eliminan y reducen relaciones innecesarias. Como resultado se obtiene un software donde se evidencia con mayor claridad las relaciones existentes entre los módulos. Sin embargo, una de sus desventajas es que en ocasiones puede surgir un conflicto entre un *plugin* y la aplicación principal, lo cual provoca diversos fallos en el sistema. En estos casos, por lo general, el software brinda la opción de desactivar el *plugin* de manera temporal o desinstalarlo.

Varias de las herramientas CAD actuales hacen uso de esta técnica, ya que permite a los usuarios instalar o desintalar aquellas funcionalidades que sean necesarias en tiempo de desarrollo. Algunas de estas herramientas son Autodesk AutoCAD (Lockhart, 2013), Autodesk Inventor (Hansen, 2013; CadStock, 2010), Solid Work (Dassault Systèmes, 2014), FreeCAD (Community, 2014) o *Computer Aided Three-dimensional Interactive Application* (*CATIA* ) (Dassault Systèmes, 2014).

## Arquitectura de *plugins* que ofrece *QT Framework*

Entre los *Integrated Develoment Environment (IDE)* que permiten la creación de plugins se encuentra *Qt Creator*, mediante el cual los desarrolladores pueden crear plugins de forma rápida e integrarlos a un sistema de forma segura, utilizando el *framework Qt* (Nokia Corporation, 2015).

*Qt* provee dos *API* para la creación de plugins, una de alto nivel para extender *Qt* como *framework*; es decir, para adicionar funcionalidades al *framework Qt* y otra de bajo nivel, mediante la cual se puede extender otras aplicaciones. Hacer una aplicación extensible utilizando *Qt Framework*, involucra varias etapas. En un primer momento se debe

*Serie Científica de la Universidad de las Ciencias Informáticas*
Vol. 9, No. 3, Marzo, 2016
ISSN: 2306-2495 | RNPS: 2343
http://publicaciones.uci.cu
Pág.149-163

definir un conjunto de interfaces que permitan la comunicación entre el sistema del cual se extiende y los *plugins* que se crean, para luego mediante la macro *Q_INTERFACES*(), a través de los metadatos, informarle a la aplicación sobre las interfaces que fueron creadas (Blanchette, y otros, 2006; Stroustrup, 1995).

*Qt* realiza el proceso de detección y carga de un *plugin* a través de la clase *QPluginLoader*(), quien verifica que el *plugin* está enlazado con la misma versión de *Qt* que la aplicación (Nokia Corporation, 2015).

## Propuesta de solución

En el desarrollo de dicha solución se hará uso de la arquitectura de *plugins* que brinda *Qt Framework*, donde mediante las macros que emplea para el desarrollo de interfaces, se realizará la conversión a *plugins* de cada uno de los *snap* que contiene actualmente el módulo. Una vez culminado este proceso se deben integrar con el software *AsiXMec*, teniendo en cuenta los restantes módulos con los cuales está estrechamente vinculado, permitiendo cargarlos complementos dinámicamente cuando el software inicialice. El sistema debe mostrar una interfaz gráfica con la cual el usuario puede interactuar, compuesta por varios *checkbox* que contienen solo aquellos *plugins* que se pudieron cargar una vez ejecutada la aplicación por un usuario. Este último debe tener la posibilidad de activar o desactivar, uno o varios *plugins* a la misma vez, dependiendo de los que desee que estén habilitados cuando se encuentre diseñando una estructura mecánica.

Pasos:

1. Convertir los *snap* a *plugins*: Para esto se hace uso de las macros *Q_DECLARE_INTERFACE(ClassName, Identifier)*, *Q_IMPORT_PLUGIN( PluginName)* y *Q_PLUGIN_METADATA()*, que ofrece *Qt Framework* para el desarrollo de *plugins*.
2. Integrar los *plugins* con el software *AsiXMec*: Una vez desarrollados los plugins, es necesario modificar el funcionamiento del módulo e integrarlo a la herramienta. Con tal propósito se deben incluir algunas funciones en la clase *plugin*-manager, para que así pueda cargarlos.
3. Cargar los *plugins*: Empleando la clase *QPluginLoader* () que provee *Qt Framework*, se cargan automáticamente los *plugins* desarrollados.
4. Mostrar *plugins* cargados: Los *plugins* que se pudieron cargar, deben ser visualizados en una interfaz que los contiene.
5. Controlar activación de los *plugins*: Debe permitir que un usuario pueda seleccionar del menú de *plugins* que se muestra en la interfaz, aquellos que desee que estén habilitados, y de igual forma poder deshabilitarlos si así lo desea.

*Serie Científica de la Universidad de las Ciencias Informáticas*
Vol. 9, No. 3, Marzo, 2016
ISSN: 2306-2495 | RNPS: 2343
http://publicaciones.uci.cu
Pág.149-163

## Arquitectura del módulo *Snap*

Para el desarrollo de la aplicación se escogió la arquitectura basada en capas, definiéndose 2 capas donde cada una realiza operaciones que progresivamente se aproximan más al cuadro de instrucciones de la máquina. En la capa de presentación se encuentran las clases que sirven a las operaciones de interfaz de usuario, mientras que en la capa lógica se encuentran las clases que realizan operaciones de interfaz del sistema (Figura 6).

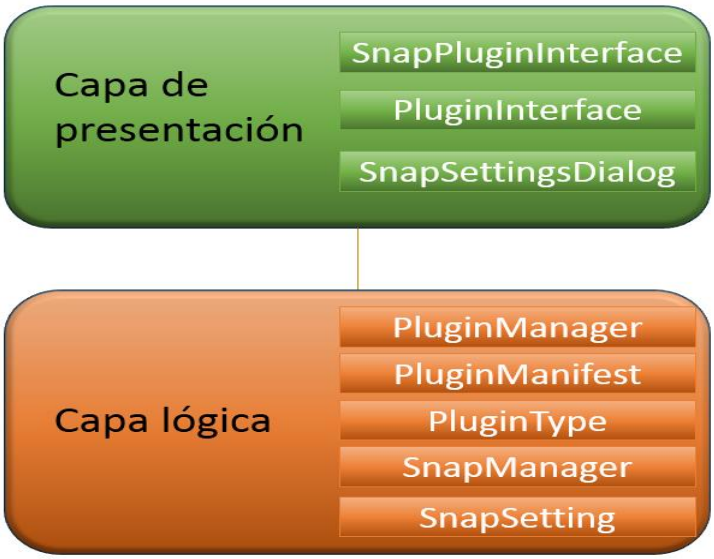


Figura 6. Arquitectura en capas del módulo *Snap*

Capa de presentación

- *SnapPluginInterface*: Esta clase brinda una interfaz de la cual heredan todos los *snaps*, ofreciendo las funciones necesarias para calcular si es posible realizar el *snap*, cómo se debe mostrar y cómo finalmente se hará.
- *PluginInterface*: Esta clase ofrece una interfaz de la cual heredan todas las interfaces de los *plugins*, brindándole a la aplicación *AsiXMec* un mecanismo estándar para el manejo de sus *plugins*.
- *SnapSettingsDialog*: Esta clase muestra una interfaz que permite la activación y desactivación de los snaps que fueron cargados.

*Serie Científica de la Universidad de las Ciencias Informáticas*
Vol. 9, No. 3, Marzo, 2016
ISSN: 2306-2495 | RNPS: 2343
http://publicaciones.uci.cu
Pág.149-163

Capa lógica

- *PluginManifest*: Es la clase encargada de capturar los datos que se encuentran en el archivo *xml* de cada *plugin*, los cuales le permiten que se carguen. Estos datos son el tipo de *plugin*, el nombre y una descripción de cada uno.
- *PluginManager*: Es la clase encargada de cargar cada *plugin*, una vez que haya identificado su tipo. Asigna a cada uno la interfaz mediante la cual se podrá interactuar con él.
- *PluginType*: Esta clase contiene los tipos de *plugins*.
- *SnapSetting*: Es la clase encargada de gestionar la activación y desactivación de los *plugins*.
- *SnapManager*: Esta clase es la encargada de obtener todos los *snaps* que fueron cargados, con el objetivo de gestionarlos. Se ejecuta cuando se crea una entidad en el *sketcher*.

Se emplea esta arquitectura ya que facilita la migración, pues el acoplamiento con el entorno está localizado en la capa inferior. Esta capa es la única a reimplementar en caso de migrar a un entorno diferente. Permite además la reutilización, pues como cada una implementa interfaces claras y lógicas, pueden intercambiarse; además del mantenimiento del sistema, ya que los cambios en una capa apenas afectan a la otra.

## Resultados y discusión

Una vez concluido el ciclo de desarrollo al sistema, así como las pruebas de calidad correspondientes, se obtuvo una segunda versión del módulo *Snap* que se integra sin errores al sistema *AsiXMec*. El componente permite adicionar al sistema nuevos *snaps* o realizar modificaciones en los que ya existe, sin la necesidad de recompilar la aplicación completa. Para ello solo basta compilar el *snap* en cuestión, lo cual trae consigo que no sean afectadas las funcionalidades de los módulos relacionados. El módulo carga dinámicamente cada uno de los *snaps* en forma de *plugins* (Figura 7), además de brindar al ingeniero mecánico la opción de controlar su activación al ejecutarse el software, pues puede decidir mediante una interfaz visual cuáles son los que desea activar o desactivar (Figura 8). Lo anterior permite que los módulos que conforman la aplicación tengan un menor acoplamiento y un mayor rendimiento en tiempo de ejecución del sistema.

*Serie Científica de la Universidad de las Ciencias Informáticas*
Vol. 9, No. 3, Marzo, 2016
ISSN: 2306-2495 | RNPS: 2343
http://publicaciones.uci.cu
Pág.149-163

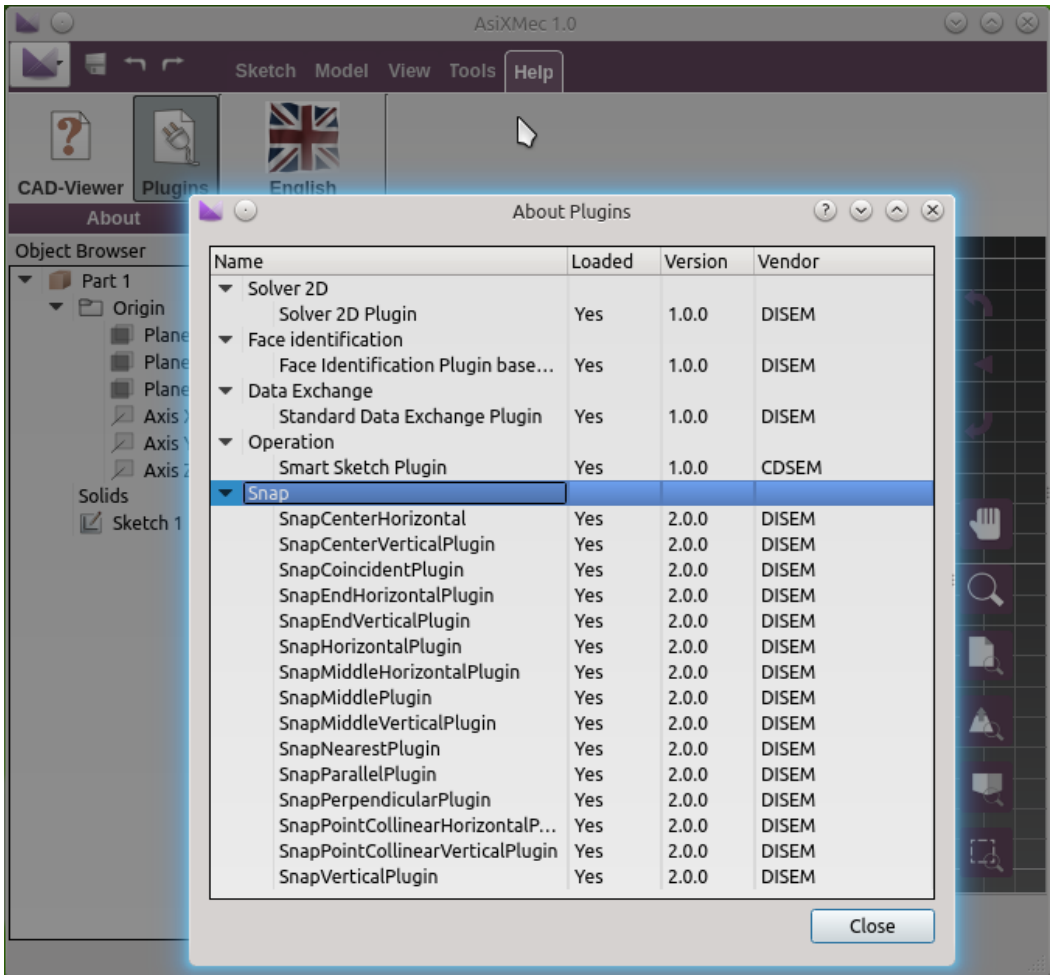


Figura 7. Plugins cargados al ejecutar *AsiXMec*

*Serie Científica de la Universidad de las Ciencias Informáticas*
Vol. 9, No. 3, Marzo, 2016
ISSN: 2306-2495 | RNPS: 2343
http://publicaciones.uci.cu
Pág.149-163

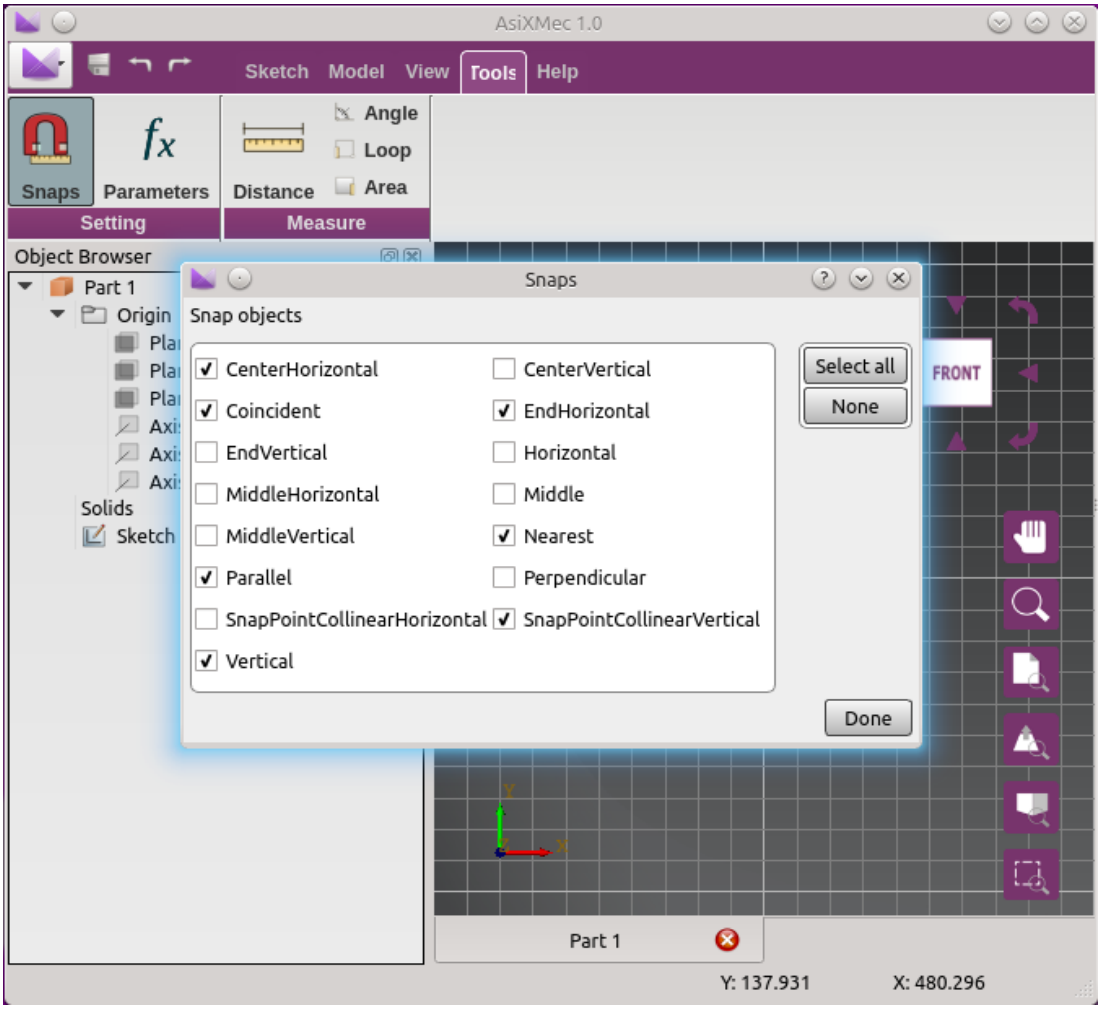


Figura 8. Interfaz visual del Módulo *Snap*

## Conclusiones

- La nueva versión del módulo *Snap* del software *AsiXMec* permite al equipo de desarrollo incorporar nuevos *snaps* o modificar los que ya existen sin la necesidad de compilar todo el proyecto, reduciendo así el acoplamiento.
- Los usuarios una vez inicializada la herramienta, permite cargar, mostrar y seleccionar los *snaps* en forma de *plugins*, los cuales se pueden habilitar o deshabilitar teniendo en cuenta las necesidades del diseño.

## Referencias

**Arcia Salazar, Miguel. 2011.** *Desarrollo de un componente visual para Qt utilizando el framework OpenCascade.* 2011.

**Blanchette, Jasmin y Summerfield, Mark. 2006.** *C++ GUI programming with Qt 4.* s.l. : Prentice Hall Professional, 2006.

**CadStock. 2010.** CadStock. *CadStock.* [En línea] Febrero de 2010. [Citado el: 30 de Octubre de 2015.] http://cadstock.com/noticias/420-screenshot-plugin-para-capturar-piezas-o-ensamblajes-en-autodesk-inventor.

**Community, LibreCAD. 2014.** LibreCAD. *LibreCAD.* [En línea] 2014. [Citado el: 30 de Octubre de 2015.] http://librecad.org/cms/home.html.

**Dassault Systèmes. 2014.** Dassault Systèmes. *Dassault Systèmes.* [En línea] 2014. [Citado el: 30 de Octubre de 2015.] http://www.3ds.com/products-services/catia/xtmc=catia&xtcr=1.

**2014.** SolidWorks. *SolidWorks.* [En línea] 2014. [Citado el: 30 de Octubre de 2015.] http://www.solidworks.com.

**Editorial Vértice. 2010.** *Photoshop.* s.l. : Vértice, 2010.

*Generación de recomendaciones para terapias antivirales del VIH.* **Andres Peralta Godoy, Ezequiel S. Velez. 2011.** Córdoba, Argentina : Master's thesis, FaMAF-UNC, 2011, Vol. 3.

**Peralta Godoy, Andres y S. Velez, Ezequiel . 2011.** Córdoba, Argentina : Master's thesis, FaMAF-UNC, 2011, Vol. 3.

**Hansen, Scott. 2013.** *Autodesk Inventor 2014: A Tutorial Introduction.* s.l. : SDC Publications, 2013.

**IBM Corporation . 2012.** *Plug-In for Informatica PowerCenter.* 2012.

**Jackson, P., y otros. 2013.** *Method of creating a snap point in a computer-aided design system. US20130055125 A1* US, 28 de Febrero de 2013.

**Lockhart, Shawna. 2013.** *Tutorial Guide to AutoCAD 2014.* s.l. : SDC Publications, 2013.

**Nokia Corporation.** Qt Reference Documentation. *Qt Reference Documentation.* [En línea] [Citado el: 1 de Enero de 2015.] http://doc.qt.digia.com/qtcreator-2.3/creator-os-supported-platforms.htm.

**2015.** Qt Reference Documentation. *Qt Reference Documentation.* [En línea] 2015. [Citado el: 30 de Octubre de 2015.] http://doc.qt.digia.com/qtcreator-2.3/creator-os-supported-platforms.htm.

**Open CASCADE SAS. 2015.** OpenCASCADE. *OpenCASCADE.* [En línea] 2015. [Citado el: 30 de Octubre de 2015.] http://www.opencascade.org/.

**Saldaña Pomazunco, Jorge Luis. 2012.** *Creación de Estándares para planos mediante el software CAD Autodesk Inventor Professional 2011.* Universidad Nacional Mayor de San Marcos, Perú : s.n., 2012.

**Salomon, David. 2011.** *The computer graphics manual.* s.l. : Springer, 2011.

**Schroff Development Corporation. 2014.** *Tools for Design Using AutoCAD 2014 and Autodesk Inventor 2014.* 2014.

**Shih, Randy H. 2014.** *Schroff Development Corporation: Tools for Design Using AutoCAD 2014 and Autodesk Inventor 2014.* 2014.

**Stroustrup, Bjarne . 1995.** *The C++ programming language.* s.l. : Pearson Education India, 1995.