



Trabajo de Diploma para optar por el título de Ingeniero Informático.

Título: Entidad arco de elipse para el modelador geométrico de AsiXMec 2.0.

Autor: Henry González Olivera

Tutor: Ing. Ángel Ulise Tabares González.

Año 58 de la Revolución.

La Habana, Marzo 2016

DECLARACIÓN DE AUTORÍA

Declaro ser autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas con los derechos patrimoniales de la misma, con carácter exclusivo. Autorizo a dicho centro para que haga el uso que estime pertinente con este trabajo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Firma del autor

Firma del tutor

DEDICATORIA

Dedico el presente trabajo mis padres por siempre estar ahí cuando lo necesitaba, a mi abuela Celia, que donde quiera que este vea que cumplí con mi promesa y a todos aquellos que de una manera u otra me ayudaron a conseguirlo.

AGRADECIMIENTOS

A mis padres Rosabel y Jorge por siempre ser mi guía y mi sostén, por siempre estar cuando los necesitaba, por darme la oportunidad y confianza que necesitaba para estudiar.

A mi abuela Cela, que aunque no se encuentra hoy entre nosotros siempre me apoyó para que terminara mi carrera.

A mi hermano Harold, que siempre me daba consejos e ideas.

A mi tutor por siempre guiarme por el camino correcto y ayudarme cada vez que iba a pedir su ayuda y por soportarme por ser tan insistente y recalcitrante.

A mis amigos Peter, Ale, Javier, Pedro (Dundune), Adrián por siempre estar ahí para escuchar mis historias y compartir conmigo.

A Juan Carlos y Anelis por siempre estar apoyándome y halándome las orejas cuando lo necesitaba.

A José Miguel y Guille por ser siempre mis compañeros de armas dentro y fuera del aula.

A Danier y Abel que de una manera u otra siempre lograban sacarme una sonrisa.

A Martha por guiarme por el camino del liderazgo cuando más nadie podía.

A Yeinelis y Ofelia por siempre escucharme cuando más nadie lo hacía.

A David, Andy y Frank (Tito) por siempre mantenerme en la última moda y actualizado.

A todos los del grupo 5504 por estos 5 años, por soportarme, por dejarme entrar en sus vidas y ser partícipe de ellas.

A todos aquellos que he conocido durante todo este tiempo.

A los profesores que me han ayudado a vencer los obstáculos que se me han presentado.

A la profesora Yirka que sin ella no estaría haciendo este trabajo hoy.

A la profesora Yadira por siempre estar pendiente de mí.

A todos los que de una manera u otra me han ayudado a salir adelante y han podido hacer mi sueño realidad.

RESUMEN

El proyecto AsiXMec del centro Vertex (Entornos Interactivos 3D) de la Universidad de las Ciencias Informáticas (UCI) no soporta la creación de arcos de elipse ni las funcionalidades asociadas a los mismos en el módulo 2D. Dicha funcionalidad es necesaria en el diseño de engranes no circulares de tipo elíptico como son los mecanismos balanceadores, los medidores de caudales, mecanismos de pistón-biela-manivela, mecanismos de dirección de automóviles, el tren planetario de algunas bicicletas de alto rendimiento, entre otros.

Para dar respuesta al problema anterior planteado se implementaron la entidad arco de elipse en el módulo Entity y las funcionalidades *Copy*, *Move*, *Delete.*, *Scale*, *Rotate*, *Offset*, *Break* y *Trim*.

Palabras claves: arcos de elipse, CAD, opencascade.

ÍNDICE DE CONTENIDO

RESUMEN.....	5
ÍNDICE DE CONTENIDO	6
ÍNDICE DE TABLAS.	8
INTRODUCCIÓN.....	10
CAPÍTULO 1: MARCO TEÓRICO DE LA INVESTIGACIÓN.....	12
1.1 Introducción.....	12
1.2 CAD/CAE/CAM	12
1.3 AsiXMec.....	13
1.4 Otros sistemas CAD	13
1.5 Módulo.....	13
1.6 Elipse	14
1.7 Arco	15
1.8 Arco de Elipse.....	15
1.9 Herramientas, lenguajes y metodologías.	16
1.9.1 QT.....	16
1.9.2 QtCreator	16
1.9.3 Lenguaje de programación.....	17
1.9.4 Open CASCADE.....	17
1.9.5 Metodología de desarrollo	17
1.10 Conclusiones parciales.....	19
CAPÍTULO 2: PLANIFICACIÓN Y DISEÑO.	20
2.1 Introducción.....	20
2.2 Requisitos Funcionales	20
2.3 Fase de Planificación	20
2.3.1 Descripción de las HU.....	20
2.3.2 Estimación del esfuerzo por HU.....	25
2.3.3 Plan de iteraciones	26
2.3.4 Plan de entregas	27
2.4 Fase de diseño.....	28
2.4.1 Estructura de los módulos	28
2.4.2 Patrones de software	30

2.4.3 Tarjetas clase-responsabilidad-colaboración	31
Conclusiones parciales.....	32
CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBAS.	33
3.1 Introducción.....	33
3.2 Fase de desarrollo.....	33
3.2.1 Tareas de ingeniería o desarrollo.....	33
3.3 Fase de pruebas	38
3.3.1 Pruebas de aceptación.....	38
3.3.2 Pruebas de aceptación de la primera iteración.....	39
3.3.3 Pruebas de aceptación de la segunda iteración	42
3.3.4 Pruebas de aceptación de la tercera iteración.....	45
3.4 Conclusiones parciales.	47
CONCLUSIONES	49
RECOMENDACIONES.....	50
BIBLIOGRAFÍA.....	51
Referencias	51

ÍNDICE DE TABLAS.

Tabla 1 HU1 Desarrollar el Arco de Elipse	21
Tabla 2 HU2 Desarrollar Move	22
Tabla 3 HU3 Desarrollar Copy-Paste	22
Tabla 4 HU4 Desarrollar Delete	23
Tabla 5 HU5 Desarrollar Traslata	23
Tabla 6 HU6 Desarrollar Scale	23
Tabla 7 HU7 Desarrollar Rotate.....	24
Tabla 8 HU8 Desarrollar Offset	24
Tabla 9 HU9 Desarrollar Break.....	25
Tabla 10 HU10 Desarrollar Trim	25
Tabla 11 Estimación de esfuerzo por HU	26
Tabla 12 Plan de iteraciones	27
Tabla 13 Plan de entregas	28
Tabla 14 Tarjeta CRC create-ellipse-arc-controller	31
Tabla 15 Tarjeta CRC ellipse-arc-command.....	32
Tabla 14 Tareas de Ingeniería	34
Tabla 15 Tareas de Programación de HU1	35
Tabla 16 Tareas de Programación de HU2.....	35
Tabla 17 Tareas de Programación de HU3.....	35
Tabla 18 Tareas de Programación de HU4.....	35
Tabla 19 Tareas de Programación de HU5.....	36
Tabla 20 Tareas de Programación de HU6.....	36
Tabla 21 Tareas de Programación de HU7.....	36
Tabla 22 Tareas de Programación de HU8.....	37
Tabla 23 Tareas de Programación de HU9.....	37
Tabla 24 Tareas de Programación de HU10.....	37
Tabla 25 UH1_P1: Desarrollar el Arco de Elipse	39
Tabla 26 UH2_P1: Desarrollar Move	40
Tabla 27 UH3_P1: Desarrollar Copy	40
Tabla 28 UH4_P1: Desarrollar Delete	41
Tabla 29 UH5_P1: Desarrollar Traslata.....	42

Tabla 30 UH6_P1: Desarrollar Scale.....	43
Tabla 31 UH7_P1: Desarrollar Rotate.....	44
Tabla 32 UH8_P1: Desarrollar Offset.....	45
Tabla 33 UH9_P1: Desarrollar Break.....	46
Tabla 34 UH10_P1: Desarrollar Trim.....	47

INTRODUCCIÓN

Las herramientas **CAD/CAE/CAM** (*Computer Aided Design, Computer Aided Engineering y Computer Aided Manufacturing* respectivamente), son herramientas que posibilitan el diseño, ingeniería y fabricación asistido por computadoras de productos industriales, revisten una gran importancia a nivel mundial por el potencial que brindan. En la actualidad la mayoría del proceso creativo en esta rama de la industria puede ser realizado utilizando software de este tipo, utilizando equivalentes digitales a las tradicionales reglas, compases y plantillas.

En los softwares de diseño asistido por ordenador el modelador geométrico es una herramienta poderosa que permite crear de una manera muy sencilla los objetos a partir de figuras básicas tales como líneas, círculos, rectángulos y polígonos. Con el avance de las tecnologías los artículos de la vida diaria e incluso los componentes industriales, poseen cada vez diseños más sofisticados para conseguir mejores atractivos visuales y fundamentalmente mejor funcionalidad.

El software AsiXMec del centro Vertex de la UCI no soporta la creación de arcos de elipse ni las funcionalidades asociadas a los mismos en el módulo 2D, funcionalidad casi imprescindible para diseñar engranes no circulares de tipo elíptico, que son la base de importantes diseños como son los mecanismos balanceadores, los medidores de caudales, mecanismos de pistón-biela-manivela, mecanismos de dirección de automóviles el tren planetario de algunas bicicletas de alto rendimiento, entre otros.

A partir de la situación problemática antes expresada se define como **problema de investigación**: ¿Cómo incorporar a los módulos del modelador geométrico la entidad arco de elipse y extender las funcionalidades que actúan sobre la misma?, a partir del cual se define como **objeto de estudio**: las entidades básicas en los modeladores geométricos de los softwares CAD. Teniendo en cuenta lo planteado anteriormente se plantea el siguiente **objetivo general**: desarrollar la entidad arco de elipse en los módulos del modelador geométrico de AsiXMec 2.0. Todo esto enmarcado en el **campo de acción**: la entidad arco de elipse en el modelador geométrico de AsiXMec 2.0.

Para dar cumplimiento al objetivo planteado se definen las siguientes **tareas de investigación**:

1. Desarrollo de la creación de arco de elipse en el módulo AsiXMec.
2. Análisis y diseño de las funcionalidades move, copy y delete.
3. Desarrollo de la funcionalidad **Move** para la entidad arco de elipse.
4. Desarrollo de la funcionalidad **Copy** para la entidad arco de elipse.

5. Desarrollo de la funcionalidad **Delete** para la entidad arco de elipse.
6. Análisis y diseño de las funcionalidades translate, scale y rotate.
7. Desarrollo de la funcionalidad **Translate** para la entidad arco de elipse.
8. Desarrollo de la funcionalidad **Scale** para la entidad arco de elipse.
9. Desarrollo de la funcionalidad **Rotate** para la entidad arco de elipse.
10. Análisis y diseño de las funcionalidades offset, break y trim.
11. Desarrollo de la funcionalidad **Offset** para la entidad arco de elipse.
12. Desarrollo de la funcionalidad **Break** para la entidad arco de elipse.
13. Desarrollo de la funcionalidad **Trim** para la entidad arco de elipse.
14. Validación de la propuesta de solución.

Para dar cumplimiento a las distintas tareas, se pusieron en práctica los siguientes métodos de investigación:

Métodos Empíricos:

Consulta de sitios web, artículos científicos, medios electrónicos, documentos oficiales, entre otros tipos de fuentes: sirvió para encontrar la manera más factible de desarrollar el arco de elipse.

Métodos Teóricos:

- Analítico-Sintético: sirvió para realizar el procesamiento de toda la información, sintetizándola y diferenciándola para de esta forma enfocarla hacia el desarrollo del arco de elipse.
- Sistémico: facilitó la implementación de cada uno de los elementos desarrollados, en la conformación de la entidad del arco de elipse (1).

CAPÍTULO 1: MARCO TEÓRICO DE LA INVESTIGACIÓN.

1.1 Introducción

En este capítulo se presentan las diferentes herramientas a utilizar en el desarrollo de este trabajo de diploma, así como definiciones importantes para la construcción del arco de elipse, el lenguaje de programación a utilizar así como los diferentes módulos empleados.

1.2 CAD/CAE/CAM

Las herramientas **CAD**, **CAE**, **CAM** conforman los conocimientos y técnicas del estado del arte. La aplicación de este conjunto de conocimientos y técnicas consigue obtener una mejora del ciclo de desarrollo y fabricación de un producto o servicio. Por tanto, debe considerarse de sumo interés a incorporación de metodologías de trabajo y herramientas informáticas de ingeniería que sirvan de soporte a dicha estrategia (2).

La utilización conjunta de las herramientas CAD/CAM/CAE, cuando estos sistemas se hayan incorporado correctamente en la organización de la empresa, permite conseguir que el tiempo de desarrollo y fabricación del producto, *Lead Time* y el tiempo de poner el producto en el mercado, *Time to Market*, sea mínimo (3).

Las tecnologías CAD/CAM/CAE se hallan ya en una fase de madurez. Su utilidad es incuestionable y han abierto posibilidades para el rediseño y producción impensables sin estas herramientas. La falta de procedimientos de diseño va asociada a rediseños que se realizan sobre la marcha, con la consiguiente pérdida de tiempo y dinero. El factor tiempo también repercute de forma prioritaria en el desarrollo de prototipos (2).

Los fabricantes de maquinaria informática que permiten soportar programas de CAD, van a proporcionar en los próximos años ordenadores más veloces, con más memoria y mayor potencia gráfica. Como tendencia de futuro, se confirmará la desaparición de la ya tenue frontera entre el mundo de los PC's y el de las Estaciones de Trabajo CAD. En el campo de los periféricos CAD sucederá algo parecido: los *plotters*¹, consolidada la tecnología de inyección de tinta, van a ser cada vez más rápidos y de mejor resolución (2).

Otra tendencia de futuro en el campo de los periféricos es la popularización de los dispositivos de impresión 3D. Hasta el presente, las tecnologías de *Rapid Prototyping*², aunque consolidadas, no se han utilizado intensivamente dado su elevado coste. Los aparatos de

¹ Máquina que imprime en forma lineal que se utiliza junto con el ordenador.

² Proceso utilizado para fabricar artículos de plástico, metal o cerámica.

reproducción tridimensionales de diseños compartirán un lugar con el *plotter* en la oficina técnica del mañana (3).

El futuro se muestra ambicioso tecnológicamente hablando, por la introducción de las células de fabricación flexible y el gran avance de los computadores y de los robots. Todo ello lleva a pensar que en un futuro próximo la "Fábrica Automática" será una realidad. Tener un conjunto grande de entidades básicas en un modelador geométrico robusto y confiable será la clave para un inicio eficiente del proceso, porque de cuan correcto esté lo que se modele dependerán los resultados finales de las pruebas y la fabricación (3).

1.3 AsiXMec

El software en su versión 1.0 es desarrollado por el proyecto DISEM del centro Vertex, este es un modelador geométrico y paramétrico basado en *Feature* e historial de operaciones, permite a los usuarios manipular, crear, editar, importar y exportar archivos *.step*, *.iges*, *.stl*, compatibles con otras aplicaciones para Linux como *Salome-Meca*, *DraftSight*, *LibreCAD* y para Microsoft Windows como *AutoCAD*, *SolidWorks* e *Inventor*, entre otros de los más conocidos (4) (5).

AsiXMec se ha desarrollado con lenguaje C++ para el desarrollo de la línea CAD-CAE, utiliza el framework³ *Open CASCADE Community Edition (OCE)*, *framework Qt-5*, soporte para *plugins*, cuenta con menú en forma de *ribbon*⁴. Cuenta con un *solver*⁵ de restricciones que utiliza la biblioteca *Eigen*, permite el diseño asistido por *snap* e identificación automática de caras. Su interfaz gráfica tiene soporte multilinguaje (inglés y español) (4) (5).

1.4 Otros sistemas CAD

Para la confección de la entidad arco de elipse se hicieron comparaciones con otros programas de modelación CAD tanto del sistema operativo Linux como de Windows. Por el lado de Windows se analizó el software *Inventor* (año 2012) arrojando que este carecía de la entidad arco de elipse. En el caso de Linux se comparó con *LibreCAD* (versión 2.1.0) dando como resultado que dicho modelador presenta la forma de crear un arco de elipse mediante su ejes de coordenadas. Debido a estos resultados se decidió hacer la entidad deseada de manera similar ya que es de fácil construcción por parte del usuario.

1.5 Módulo

En programación un módulo es una porción de un programa de ordenador. De las varias tareas que debe realizar un programa para cumplir con su función u objetivos, un módulo realizará,

³ Marco de trabajo para el desarrollo de aplicaciones.

⁴ Interfaz gráfica de usuario

⁵ Ajusta los valores cambiantes que se especifiquen

comúnmente, una de dichas tareas o varias, en algún caso. En general, un módulo recibe como entrada la salida que haya proporcionado otro módulo o los datos de entrada al sistema si se trata del módulo principal de éste; y proporcionará una salida que, a su vez, podrá ser utilizada como entrada de otro un módulo o bien contribuirá directamente a la salida final del sistema, si se retorna al módulo principal (6).

El proyecto AsiXMec está estructurado en varios módulos entre los que se encuentran:

- **cad-core**: están presentes las funcionalidades generales y básicas que componen al resto de los módulos.
- **entity**: encargado de crear y gestionar las entidades.
- **modify**: contiene las funcionalidades que modifican a las entidades.
- **transformation**: presenta las funcionalidades que aplican transformaciones lineales a las entidades, sin dejar de ser la misma entidad.
- **qt-ribbon**: gestiona la interfaz *ribbon*.
- **solver-2d-plugin**: *plugins* que implementan un *solver* de restricciones geométricas.
- AsiXMec: integra las funcionalidades especificadas de este producto, utiliza todo aquello que necesita del resto de los módulos.

1.6 Elipse

Es el lugar geométrico de todos los puntos de un plano, tales que la suma de las distancias a otros dos puntos fijos llamados focos es constante. Una elipse es la curva cerrada con dos ejes de simetría que resulta al cortar la superficie de un cono por un plano oblicuo al eje de simetría – con ángulo mayor que el de la generatriz respecto del eje de revolución. La elipse es también la imagen afín de una circunferencia (7) (8) (9).

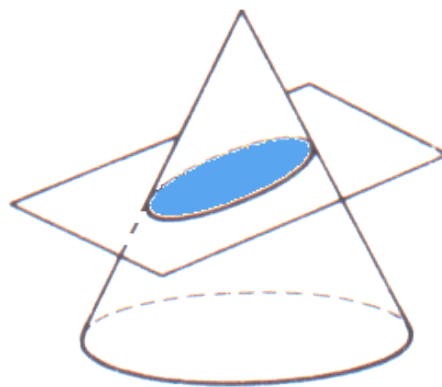


Ilustración 1 Elipse (tomado de <http://matematicasmerce.blogspot.com>)

1.7 Arco

En geometría, un arco es cualquier curva continua que une dos puntos. También se denomina arco a un segmento de circunferencia; un arco de circunferencia queda definido por tres puntos, o dos puntos extremos y el radio, o por la longitud de una cuerda y el radio (10).



Ilustración 2 Arco(tomado de <http://fagorama.com/que-es-linea-curva-abierta.html>)

1.8 Arco de Elipse

Para obtener un arco de elipse primero se debe ver sus orígenes en la matemáticas además que el framework Opencascade soporta el sistema cartesiano. Obteniendo un sistema (u,v) se obtiene:

$$\begin{aligned}u &= \alpha \cos n(t) \\v &= \alpha \sin n(t)\end{aligned}$$

donde

$$\begin{aligned}n(t) &= n1 + \Delta n(t + 0.5) \\ \Delta n &= n2 - n1\end{aligned}$$

Con $t \in [-0.5, 0.5]$. Utilizando el sistema de coordenadas cartesiano (x,y) se obtiene:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} \cos \alpha & -\sin \alpha \\ \sin \alpha & \cos \alpha \end{pmatrix} * \begin{pmatrix} u \\ v \end{pmatrix} \text{ dando finalmente}$$

$$R^2 = a^2[\cos n(t) - \cos n(t')]^2 + b^2[\sin n(t) - \sin n(t')]^2$$

Llegando a la conclusión que un arco de elipse es una curva continua que une dos puntos de una elipse y coincide en su totalidad con un tramo de esta. En la imagen siguiente se muestra

un arco de elipse mostrando los puntos referentes a las fórmulas anteriores (11).

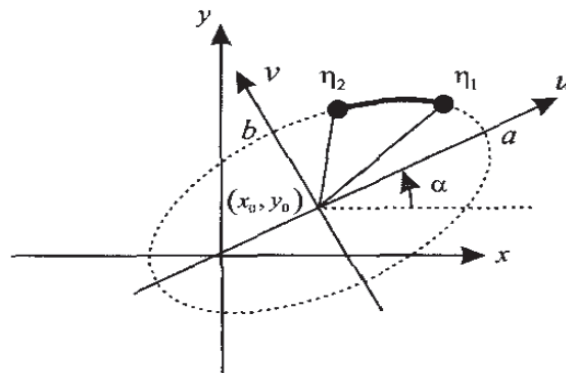


Ilustración 3 Arco de elipse (tomado de (11))

1.9 Herramientas, lenguajes y metodologías.

Para la confección del arco de elipse se utilizó un conjunto de herramientas tales como un IDE (entorno de desarrollo integrado), un lenguaje de programación empleado para desarrollar la entidad de arco de elipse así como una metodología para el correcto desarrollo del producto. Las herramientas y lenguajes que se muestra a continuación fueron elegidas debido a que son las utilizadas para el desarrollo del producto AsiXMec.

1.9.1 QT

Es un *framework* multiplataforma que se utiliza ampliamente para el desarrollo de software que se pueden ejecutar en varios sistemas operativos y plataformas de hardware con poco o ningún cambio en el código de base, sin dejar de ser una aplicación nativa con las capacidades y la velocidad de los mismos. Qt usa C++ estándar con extensiones como las señales y ranuras que facilitan el manejo de los acontecimientos lo que ayuda en el desarrollo de ambas aplicaciones GUI (*Graphic User Interface*) y servidores que reciben su propio conjunto de información del evento y debe procesar en consecuencia. Qt es compatible con muchos compiladores, incluyendo el compilador GCC, C ++ y la suite Visual Studio (12).

1.9.2 QtCreator

Es un IDE multiplataforma que se ajusta a las necesidades de los desarrolladores Qt. QtCreator se centra en proporcionar características que ayudan a los nuevos usuarios de Qt a aprender y comenzar a desarrollar rápidamente, también aumenta la productividad de los desarrolladores (12).

Ventajas

- Editor de código con soporte para C++, QML y ECMAScript.
- Herramientas para la rápida navegación del código.
- Resaltado de sintaxis y auto-completado de código.
- Control estático de código y estilo a medida que se escribe
- Soporte para *refactoring* de código.
- Ayuda sensitiva al contexto.
- Plegado de código (*code folding*).
- Paréntesis coincidentes y modos de selección (12).

1.9.3 Lenguaje de programación

Los lenguajes de programación son herramientas que permiten el desarrollo de softwares. Entre ellos tenemos Delphi, Visual Basic, Pascal, C++, entre otros, en particular se conoce como código de máquinas o lenguaje de máquinas.

C++

Lenguaje de programación diseñado a mediados de los años 1980 por Bjarne Stroustrup. La intención de su creación fue el extender al lenguaje de programación C mecanismos que permitieran la manipulación de objetos. En ese sentido, desde el punto de vista de los lenguajes orientados a objetos, C++ es un lenguaje híbrido (13).

Sus principales características son:

- Tiene un conjunto completo de instrucciones de control.
- Permite la agrupación de instrucciones.
- Incluye el concepto de puntero (variable que contiene la dirección de otra variable).
- Los argumentos de las funciones se transfieren por su valor.
- Permite la separación de un programa en módulos que admiten compilación independiente.
- Programación de bajo nivel (nivel bit) (13).

1.9.4 Open CASCADE

Es una plataforma de desarrollo de software que proporciona servicios de superficie 3D y modelado de sólidos, el intercambio de datos CAD y visualización. La mayor parte de la funcionalidad se encuentra disponible en forma de bibliotecas de C++. Es usado principalmente en el desarrollo de softwares CAD/CAE/CAM (14).

1.9.5 Metodología de desarrollo

Debido a la alta necesidad de que los proyectos lleguen al éxito y obtener un producto de gran valor para los clientes, es necesario elegir una metodología de desarrollo que permita cumplir

los objetivos planteados, puesto que unas se adaptan mejor que otras al contexto del proyecto brindando un mayor número de ventajas. Es por eso la necesidad de seleccionar una metodología robusta que ajustada al equipo cumpla con sus metas y satisfaga más allá de las necesidades definidas al inicio del proyecto (15).

El éxito del producto depende en gran parte de la metodología escogida por el equipo ya sea tradicional o ágil, donde los equipos maximicen su potencial, aumenten la calidad del producto con los recursos y tiempos establecidos (15) (16).

Programación Extrema (XP)

Es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes y donde existe un alto riesgo técnico (15).

Características principales de XP:

- Metodología basada en prueba y error
- Fundamentada en Valores y Prácticas
- Está orientada hacia quien produce y usa el software
- Reduce el costo del cambio en todas las etapas del ciclo de vida del sistema.
- Combina las que han demostrado ser las mejores prácticas para desarrollar software y las lleva al extremo (15).

Ventajas

- Programación organizada.
- Menor tasa de errores.
- Satisfacción del programador (15).

Desventajas

- Es recomendable emplearlo solo en proyectos a corto plazo.
- Altas comisiones en caso de fallar (15).

1.10 Conclusiones parciales.

Al finalizar este capítulo se ha podido apreciar los diferentes conceptos necesarios para la construcción del arco de elipse, empezando por lo más elemental que parte desde que es un módulo, siguiendo a que es una elipse, que es un arco y terminando en que es un arco de elipse, también se evidencia el uso de las diferentes herramientas los lenguajes de programación y bibliotecas necesarias para el desarrollo de la nueva entidad deseada.

CAPÍTULO 2: PLANIFICACIÓN Y DISEÑO.

2.1 Introducción

En este capítulo se realizará la planificación y diseño siguiendo la metodología XP, dando a lugar una fase de planificación donde se tendrá las historias de usuarios, la estimación por esfuerzo así como un plan de entrega. También se dispondrá de una fase de diseño donde se podrá ver la estructura de los módulos usados en el desarrollo de la entidad de arco de elipse dando como resultado el agregó en los módulos de AsiXMec el arco de elipse, respetando la forma en interactúan con el resto de las entidades y dando tratamiento al comportamiento y las características específicas del arco de elipse.

2.2 Requisitos Funcionales

Para distinguir las características que debía cumplir el arco de elipse se hizo una entrevista al jefe de proyecto y desarrolladores de los módulos dando como respuesta:

- Se debe poder construir el arco de elipse.
- Se debe poder realizar las funcionalidades **Copy y Paste** en el arco de elipse.
- Se debe poder realizar la funcionalidad **Rotation** en el arco de elipse.
- Se debe poder realizar la funcionalidad **Move** en el arco de elipse.
- Se debe poder realizar la funcionalidad **Scale** en el arco de elipse.
- Se debe poder realizar la funcionalidad **Traslate** en el arco de elipse.
- Se debe poder realizar la funcionalidad **Offset** en el arco de elipse.
- Se debe poder realizar la funcionalidad **Break** en el arco de elipse.
- Se debe poder realizar la funcionalidad **Trim** en el arco de elipse.

2.3 Fase de Planificación

En esta fase, los clientes plantean a grandes rasgos las Historias de Usuario (HU) que son de interés para la primera entrega del producto. Al mismo tiempo el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto (18).

2.3.1 Descripción de las HU

Las HU son escritas por el cliente, en su propio lenguaje, como descripciones cortas de lo que el sistema debe realizar. Las principales características de las historias de usuario

independientes unas de otras, negociables, valoradas por los clientes o usuarios, estimables, pequeñas y verificables (19).

Para la confección de las HU se realizaron diferentes entrevistas a desarrolladores del proyecto AsiXMec. Si bien el jefe de proyecto no fue quien escribió personalmente las HU, fue él quien diseñó su contenido, asignó la prioridad de cada una de ellas y dirigió la redacción de las mismas.

El jefe de proyecto se encargó de asignarle una prioridad a cada HU. El equipo de desarrollo por su parte revisó esta prioridad analizando la dependencia entre HU y asignó el costo de cada una de ellas, este se traduce en las semanas para su desarrollo. También es importante destacar, que las HU nuevas pueden describirse en cualquier momento, con esto se comprueba la flexibilidad de la metodología (19).

Las HU se representan mediante tablas las cuales contienen las secciones siguientes:

- **Código:** siglas de HU más un número consecutivo.
- **Nombre:** nombre que identifica la HU.
- **Referencia:** conjunto de códigos de las diferentes HU de las cuales depende actualmente la que se encuentra en desarrollo.
- **Prioridad:** esta característica es dada por el cliente con los valores: alta, media o baja en dependencia de la importancia en que desean ser implementadas.
- **Iteración asignada:** número de la iteración en la cual se desarrollará la HU.
- **Puntos estimados:** tiempo estimado en semanas que se le asignara.
- **Descripción:** breve descripción del proceso que define la HU.
- **Observaciones:** alguna acotación importante de señalar acerca de la HU.

Se confeccionaron un total de 10 HU, tomando para los puntos estimados la unidad como una semana de trabajo. En las tablas de la 1 a la 10 se presentan dichas HU. La HU 1 es la construcción del arco de elipse, de la 2,3 y 4 son las funcionalidades básicas, 5,6 y 7 de transformación, 8,9 y 10 de modificación. Todas las funcionalidades son usadas con frecuencia además de tener una alta complejidad por lo que tienen una prioridad alta o media

Tabla 1 HU1 Desarrollar el Arco de Elipse

HU	
Código: HU1	Prioridad: Alta
Puntos estimados:0.1	Iteración asignada: 1

Responsable: Henry González Olivera	Referencias: _
Nombre: Crear el Arco de Elipse	
Descripción: Primero se pondrá el punto central, seguido por dos puntos que serán los radios, seguido se pondrán un cuarto y quinto punto para formar el arco en contra de las manecillas del reloj.	
Observaciones: El arco se forma a partir del cuarto y quinto punto, creándose la entidad de arco de elipse desde el punto cuatro hacia el punto quinto uniéndolos de manera contra reloj	

Tabla 2 HU2 Desarrollar Move

HU	
Código: HU2	Prioridad: Alta
Puntos estimados:0.5	Iteración asignada: 1
Responsable: Henry González Olivera	Referencias: _
Nombre: Desarrollar Move	
Descripción: Trasladar la entidad hasta el punto deseado arrastrándola por la pantalla.	
Observaciones: Se debe seleccionar el punto central para poder mover la entidad.	

Tabla 3 HU3 Desarrollar Copy-Paste

HU	
Código: HU3	Prioridad: Alta
Puntos estimados:0.3	Iteración asignada: 1
Responsable: Henry González Olivera	Referencias: _
Nombre: Desarrollar Copy	
Descripción: A través del teclado se copia, seleccionada ya la figura se presionan las teclas ctrl+c para copiar y ctrl+v para pegar.	
Observaciones:	

La nueva figura aparecerá con una pequeña traslación hacia arriba y hacia la derecha.

Tabla 4 HU4 Desarrollar Delete

HU	
Código: HU4	Prioridad: Alta
Puntos estimados:0.1	Iteración asignada: 1
Responsable: Henry González Olivera	Referencias: _
Nombre: Desarrollar Delete	
Descripción: Se selecciona la figura y se presiona la tecla delete o backspace del teclado.	
Observaciones:	

Tabla 5 HU5 Desarrollar Traslata

HU	
Código: HU5	Prioridad: Alta
Puntos estimados:0.5	Iteración asignada: 2
Responsable: Henry González Olivera	Referencias: _
Nombre: Desarrollar Traslata	
Descripción: Se selecciona la figura, se selecciona la opción de traslata, abierto el dialogo se selecciona el cursor para trasladar la entidad en un vector dándole sentido y distancia.	
Observaciones: Se debe dar aceptar para ver el cambio y soportar la funcionalidad <i>Copy</i>	

Tabla 6 HU6 Desarrollar Scale

HU	
Código: HU6	Prioridad: Alta

Puntos estimados:0.5	Iteración asignada: 2
Responsable: Henry González Olivera	Referencias: _
Nombre: Desarrollar Scale	
Descripción: Abierto el dialogo se selecciona la entidad, se pone el punto base hacia dónde va a dirigirse el scale, se pone la cantidad deseada a escalar.	
Observaciones: Se debe dar aceptar para ver el cambio y debe soportar la funcionalidad <i>Copy</i>	

Tabla 7 HU7 Desarrollar Rotate

HU	
Código: HU7	Prioridad: Alta
Puntos estimados:0.4	Iteración asignada: 2
Responsable: Henry González Olivera	Referencias: _
Nombre: Desarrollar Rotate	
Descripción: Abierto el dialogo se selecciona la figura, se pone el punto base de donde va a rotar la entidad, se ponen la cantidad de grados a rotar.	
Observaciones: Se debe dar aceptar para ver el cambio y debe soportar la funcionalidad <i>Copy</i>	

Tabla 8 HU8 Desarrollar Offset

HU	
Código: HU8	Prioridad: Alta
Puntos estimados:1	Iteración asignada: 3
Responsable: Henry González Olivera	Referencias: _
Nombre: Desarrollar Offset	
Descripción: Seleccionada la figura se selecciona la opción ya seleccionada se procede a crear otra figura igual a al anterior con el mismo punto central pero de otro tamaño.	

Observaciones:

Tabla 9 HU9 Desarrollar Break

HU	
Código: HU9	Prioridad: Alta
Puntos estimados:1	Iteración asignada: 3
Responsable: Henry González Olivera	Referencias: _
Nombre: Desarrollar Break	
Descripción: Al seleccionar esta funcionalidad y seleccionar la entidad el arco de elipse se divide en dos por el punto de intercepción.	
Observaciones: Solo es posible cuando la entidad es cortada con otra entidad.	

Tabla 10 HU10 Desarrollar Trim

HU	
Código: HU10	Prioridad: Alta
Puntos estimados:1	Iteración asignada: 3
Responsable: Henry González Olivera	Referencias: _
Nombre: Desarrollar Trim	
Descripción: Al seleccionar esta funcionalidad y seleccionar la entidad el arco de elipse se divide en dos por el punto de intercepción y se borra la parte seleccionada por el usuario.	
Observaciones: Solo es posible cuando la entidad es cortada con otra entidad.	

2.3.2 Estimación del esfuerzo por HU

En la fase de planificación se priorizan las HU y se acuerda el alcance de cada una de las entregas. Los desarrolladores estiman cuánto esfuerzo requiere cada historia y a partir de allí se define el plan de iteraciones. La primera iteración crea un sistema con la arquitectura del

sistema completo. Esto es alcanzado seleccionando las historias que harán cumplir la construcción de la estructura para el sistema completo. El cliente decide las historias que se seleccionan para cada iteración. Las pruebas funcionales creadas por el cliente se ejecutan al final de cada iteración (18).

A continuación, se presenta una tabla donde se resume la estimación del esfuerzo realizado por parte del desarrollador y la duración total que tendrá el desarrollo del arco de elipse y las funcionalidades. Se toma como referencia la siguiente escala:

0.1 representa un día laborable, 0.2 representa 2 días laborables, 0.3 representa 3 días laborables, 0.4 representa 4 días laborables, 0.5 representa 5 días laborables y 1.0 representa una semana.

Tabla 11 Estimación de esfuerzo por HU

HU	Puntos de estimación (semanas)
UH1 Crear el Arco de Elipse	0.4
UH2 Desarrollar Move	1
UH3 Desarrollar Copy	1
UH4 Desarrollar Delete	0.1
UH5 Desarrollar Traslate	1
UH6 Desarrollar Scale	1
UH7 Desarrollar Rotate	1
UH8 Desarrollar Offset	2
UH9 Desarrollar Break	2
UH10 Desarrollar Trim	2
Total estimado	11.5

2.3.3 Plan de iteraciones

Luego de identificar y redactar cada una de las HU y de la estimación del esfuerzo necesario para realizarlas, se debe conformar el plan de iteraciones. Las HU seleccionadas para cada iteración son desarrolladas y probadas de acuerdo al orden preestablecido.

El desarrollo del mecanismo fue dividido en 3 iteraciones teniendo en cuenta la dificultad de cada funcionalidad las cuales son agrupadas de tres formas:

- básicas (Move, Copy y Delete)

- transformación (Translate, Scale y Rotate)
- modificación (Offset, Break y Trim)

Al finalizar la iteración se realiza una entrega funcional para darle paso al proceso de revisión y análisis de la misma.

Iteración 1: se desarrollan las HU relacionadas con las funcionalidades básicas del sistema.

Iteración 2: se implementan las HU concernientes a las funcionalidades de transformación.

Iteración 3: se desarrollan las HU respectivas a las funcionalidades de modificación.

Para aproximar el tiempo de ejecución de cada iteración, se tomó como medida en cada una de las iteraciones que la semana constaba de 5 días en los que se trabajaban 6 horas sin distracciones.

Tabla 12 Plan de iteraciones

Iteración	HU	Duración total
1	Crear el Arco de Elipse	2,5 semanas
	Desarrollar Move	
	Desarrollar Copy	
	Desarrollar Delete	
2	Desarrollar Traslate	3 semanas
	Desarrollar Scale	
	Desarrollar Rotate	
3	Desarrollar Offset	6 semanas
	Desarrollar Break	
	Desarrollar Trim	

2.3.4 Plan de entregas

El plan (o cronograma) de entregas establece qué HU son agrupadas para conformar una entrega y el orden de las mismas. Este cronograma es el resultado de una reunión entre todos los actores del proyecto (cliente, desarrolladores, gerentes, etc.). El mismo se realiza en base a las estimaciones de tiempos de desarrollo realizadas por los desarrolladores (18).

A partir del plan de iteraciones analizado en el acápite anterior y en correspondencia con el mismo se realiza el plan de entregas en el cual se proponen 3 versiones funcionales y una

última entrega de iteraciones del producto el 16 de mayo, para dar paso a la fase de pruebas.

Tabla 13 Plan de entregas

HU	1 iteración 25 de febrero	2 iteración 28 de marzo	3 iteración 16 de mayo
UH1 Crear el Arco de Elipse	Versión 1.0		
UH2 Desarrollar Move			
UH3 Desarrollar Copy			
UH4 Desarrollar Delete			
UH5 Desarrollar Traslate		Versión 2.0	
UH6 Desarrollar Scale			
UH7 Desarrollar Rotate			
UH8 Desarrollar Offset			Versión 3.0
UH9 Desarrollar Break			
UH10 Desarrollar Trim			

2.4 Fase de diseño

La metodología XP sugiere que hay que conseguir diseños simples y sencillos, procurando hacerlo todo lo menos complicado posible para conseguir un diseño fácilmente entendible que a la larga costará menos tiempo y esfuerzo desarrollar.

2.4.1 Estructura de los módulos

El sistema AsiXMec está compuesto por varios módulos. En la creación del arco de elipse y en las funcionalidades que deben ser soportadas intervienen los módulos: Entity, Transformation y Modify.

El módulo Entity crea la entidad arco de elipse y contiene el controlador de la entidad. La estructura del módulo se presenta en la ilustración siguiente.

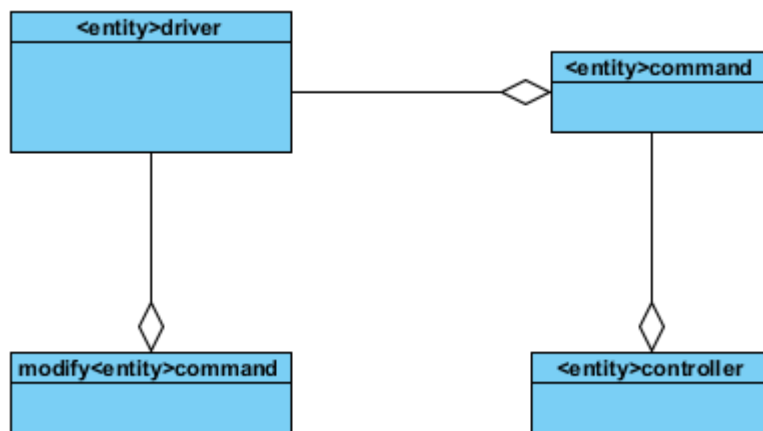


Ilustración 4 Estructura del módulo Entity

En el módulo Transformation se encuentran las funcionalidades que permiten transformar la entidad creada. Permite trasladar la entidad, aumentar su tamaño y hacerla rotar. La estructura del módulo se presenta a continuación:

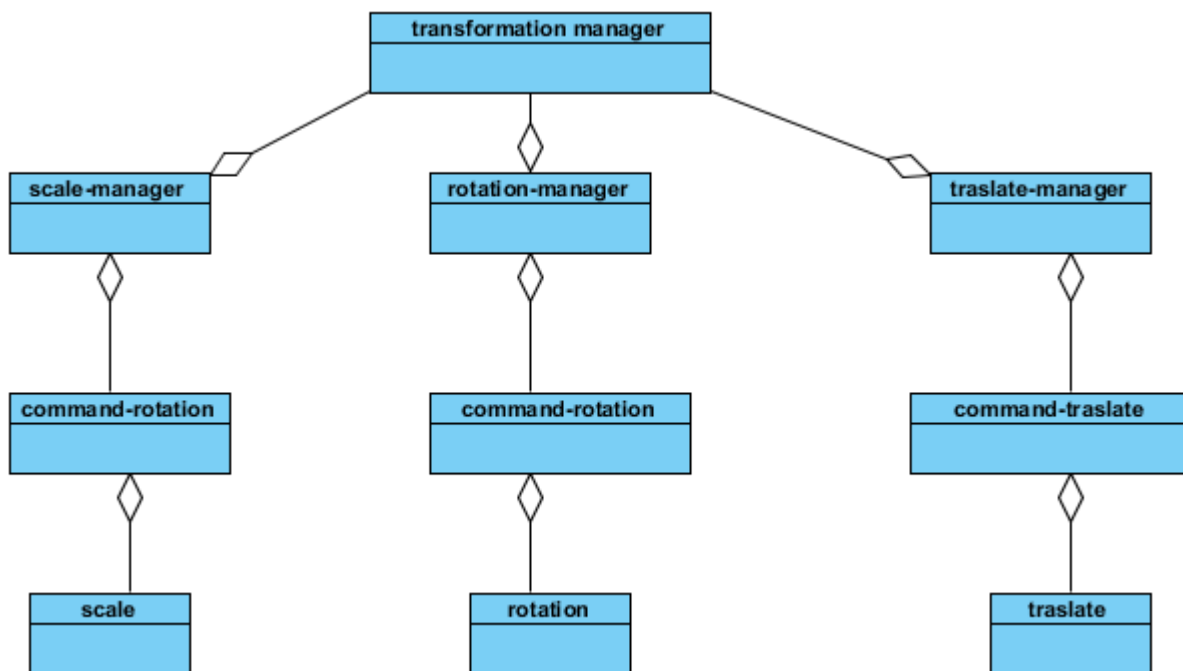


Ilustración 5 Estructura del módulo Transformation

El módulo Modify se encarga de modificar la entidad, permitiendo crear una nueva de diferente tamaño, dividirla en dos o borrar una parte de esta.

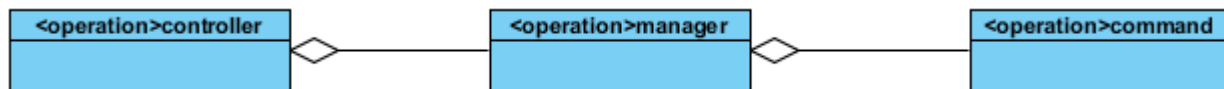


Ilustración 6 Estructura del módulo Modify

2.4.2 Patrones de software

Los Patrones de Software para la Asignación General de Responsabilidad (GRASP, por sus siglas en inglés) describen los principios fundamentales de la asignación de responsabilidades a objetos expresados en formas de patrones (20). En la implementación de la aplicación se utilizaron los siguientes:

- **Experto:** Es un principio básico que suele utilizarse en el diseño orientado a objetos (20). La idea es determinar sobre qué clase es mejor delegar las responsabilidades de acuerdo a su información; durante la creación de las entidades puede apreciarse que el “Controlador” es quien dirige el proceso de creación y decide el momento en que la información obtenida es suficiente para luego delegar la construcción a los “Comandos”.
- **Alta cohesión:** Es la meta principal que ha de tenerse en cuenta en cada momento en todas las decisiones de diseño. Es un patrón evaluativo que el desarrollador aplica al valorar sus decisiones de diseño. Se simplifican el mantenimiento y las mejoras en funcionalidad. A menudo genera un bajo acoplamiento (20). Este patrón fue utilizado en el diseño de la aplicación de manera general, siguiendo la premisa de que cada clase debe contener operaciones que resuelvan necesidades afines con ellas.
- **Bajo acoplamiento:** Es un principio que deben recordar durante las decisiones de diseño: es la meta principal que es preciso tener presente siempre. Estimula asignar una responsabilidad de modo que su colaboración no incremente el acoplamiento tanto que produzca los resultados negativos propios de un alto acoplamiento. Soporta el diseño de clases más independientes, que reducen el impacto de los cambios (20). Este patrón fue utilizado para el diseño de las clases utilizadas en la construcción de las entidades.

Teniendo en cuenta lo expuesto en el libro Design Patterns (21) y la arquitectura del sistema AsiXMec se hizo uso de los siguientes patrones GoF (Gang of Four):

- **Command:** encapsula una petición como un objeto, permitiendo que de este modo se parametriza con los clientes las diferentes solicitudes, la cola o las solicitudes de registro

(21). Este patrón se utiliza prácticamente en todas las funcionalidades y su diseño puede apreciarse en las Ilustraciones 4, 5 y 6.

- **State:** permite que un objeto modifique su comportamiento cuando cambia su estado interno (21). Por las características del software asixmec se utiliza a lo largo de todas las funcionalidades ya que este funciona como una gran máquina de estados, por ejemplo para la creación de un arco de elipse se debe entrar en el estado “Crear arco de elipse”, para realizar el offset sobre cualquier entidad se debe entrar en el estado “Offset”, de manera que ante las mismas acciones en diferentes estados se obtienen diferentes respuestas.
- **Builder:** separa la construcción de un objeto complejo de su representación de modo que el mismo proceso de construcción puede crear diferentes representaciones (21). De esta forma se puede previsualizar cualquier objeto y posteriormente construirlo sin duplicar código, el ejemplo claro de esto es la creación de las entidades.

2.4.3 Tarjetas clase-responsabilidad-colaboración

Las tarjetas clase-responsabilidad-colaboración (CRC) permiten desprenderse del método basado en procedimientos y trabajar con una metodología basada en objetos, así el programador se concentra y empieza a apreciar el desarrollo orientado a objetos. Las tarjetas CRC representan objetos y se describen partir de los siguientes elementos.

- **Clase:** nombre de la clase a la cual pertenece la tarjeta.
- **Responsabilidad:** describe cuales son las funcionalidades que deben ser implementadas por la clase.
- **Colaboración:** enumera las diferentes clases con las cuales tiene relación la clase a la cual pertenece la tarjeta CRC.

Tabla 14 Tarjeta CRC create-ellipse-arc-controller

TARJETA CRC
Clase: create-ellipse-arc-controller

Responsabilidad: mouseRelease(Se llama cuando el botón de ratón es liberado) mouseRelease(Se llama cuando el cursor del ratón se mueve) mouseMove(Termina la creación del arco de elipse)	Colaboración: create-entity-controller.h prs-length-dimension.h prs-angle-dimension.h prs-ellipse-dimension.h point-node.h
---	--

Tabla 15 Tarjeta CRC ellipse-arc-command

TARJETA CRC	
Clase: ellipse-arc-command	
Responsabilidad: EllipseArcCommand(Es el constructor de la clase)	Colaboración: command.h point-node.h ellipse-arc-node.h

Conclusiones parciales

Al finalizar este capítulo se logró obtener las características de la entidad arco de elipse, una descripción para cada HU, un plan de iteraciones y un plan de entregas real. Se pudo analizar la estructura de los módulos empleados además de estudiar los diferentes patrones utilizados en la implementación de la entidad de arco de elipse.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBAS.

3.1 Introducción

En este capítulo se podrá entrar en la fase de desarrollo de la entidad, además de realizar las diferentes pruebas para verificar que la aplicación desarrollada tenga un correcto funcionamiento.

3.2 Fase de desarrollo

Dentro de la fase de desarrollo se encuentra la implementación del código el cual se sustenta en buenas prácticas planteadas por el ciclo de vida de XP para esta etapa, entre ella se encuentran:

- disponibilidad del cliente.
- uso de estándares.
- programación dirigida por las pruebas (*Test-driven programming*)
- integraciones permanentes y ritmo sostenido (19).

3.2.1 Tareas de ingeniería o desarrollo

Asociado a cada iteración se encuentra la planificación de las tareas de ingeniería o programación, cada HU se transforma en estas tareas que son desarrolladas por programadores, dentro del equipo de desarrollo. Para cada iteración se realizó la distribución de tareas en correspondencia con las HU que se desarrollaron (18).

A continuación se relacionan las tablas correspondientes a las tareas de programación realizadas, cuyos campos responden a las siguientes descripciones:

- **Número de HU:** número de la HU a la que corresponde. Índice de la HU a la que se corresponde esta tarea de programación.
- **Número tarea:** índice de la tarea de programación. Es un número único que se le asigna a cada tarea de programación que pertenece a una HU determinada con el fin de lograr una mejor organización de éstas.
- **Nombre tarea:** nombre de la tarea de programación. Debe ser descriptivo, en la medida de las posibilidades, de lo que se realizará y no muy extenso.
- **Tipo de tarea:** informa el tipo de tarea a realizar. Las tareas pueden ser:
 - **Desarrollo:** tarea que se realizará por primera vez.

- **Corrección:** tarea que se realiza a partir de una anterior que no se realizó correctamente, es decir no pasó correctamente todos los casos de prueba que le corresponden.
- **Mejora:** tarea que se realiza a partir de una anterior que se realizó correctamente pero se incorporan nuevos requerimientos para la misma.
- **Puntos estimados:** representación en porcentaje de la cantidad de tiempo estimada de una semana, que se utilizara para su realización.
- **Fecha inicio:** fecha estimada de inicio de realización.
- **Fecha fin:** fecha estimada de fin de realización.
- **Descripción:** describe en qué consiste la tarea y qué elementos deben cumplirse para declararla terminada.

A continuación se relacionan las tareas de programación a realizar para cada HU.

Tabla 16 Tareas de Ingeniería

TAREAS DE PROGRAMACIÓN	
HU	Tarea de programación
UH1 Desarrollar el Arco de Elipse	1. Implementar la funcionalidad Arco de Elipse
UH2 Desarrollar Move	1. Implementar la funcionalidad Move
UH3 Desarrollar Copy	1. Implementar la funcionalidad Copy
UH4 Desarrollar Delete	1. Implementar la funcionalidad Delete
UH5 Desarrollar Traslata	1. Implementar la funcionalidad Traslata
UH6 Desarrollar Scale	1. Implementar la funcionalidad Scale
UH7 Desarrollar Rotate	1. Implementar la funcionalidad Rotate
UH8 Desarrollar Offset	1. Implementar la funcionalidad Offset
UH9 Desarrollar Break	1. Implementar la funcionalidad Break
UH10 Desarrollar Trim	1. Implementar la funcionalidad Trim

A continuación se presentan las tareas de programación correspondiente a las historias de usuarios de la creación de la elipse.

Tabla 17 Tareas de Programación de HU1

TAREA DE PROGRAMACIÓN	
No. de tarea: 1	No.de HU: 1
Nombre de la tarea: Implementar la funcionalidad Arco de Elipse	
Tipo de tarea: Desarrollo	Responsable: Henry González Olivera
Fecha inicio:	Fecha fin:
Descripción: Hacer el arco de elipse para fabricar modelos de curvas de difícil construcción.	

Tabla 18 Tareas de Programación de HU2

TAREA DE PROGRAMACIÓN	
No. de tarea: 1	No.de HU: 2
Nombre de la tarea: Implementar la funcionalidad Move	
Tipo de tarea: Desarrollo	Responsable: Henry González Olivera
Fecha inicio:	Fecha fin:
Descripción: Mover el arco de elipse para cualquier punto de la pantalla deseado.	

Tabla 19 Tareas de Programación de HU3

TAREA DE PROGRAMACIÓN	
No. de tarea: 1	No.de HU: 3
Nombre de la tarea: Implementar la funcionalidad Copy	
Tipo de tarea: Desarrollo	Responsable: Henry González Olivera
Fecha inicio:	Fecha fin:
Descripción: La nueva entidad aparecerá con una pequeña traslación hacia arriba y hacia la derecha	

Tabla 20 Tareas de Programación de HU4

TAREA DE PROGRAMACIÓN	
------------------------------	--

No. de tarea: 1	No.de HU: 4
Nombre de la tarea: Implementar la funcionalidad Delete	
Tipo de tarea: Desarrollo	Responsable: Henry González Olivera
Fecha inicio:	Fecha fin:
Descripción: Borrar la entidad creada.	

Tabla 21 Tareas de Programación de HU5

TAREA DE PROGRAMACIÓN	
No. de tarea: 1	No.de HU: 5
Nombre de la tarea: Implementar la funcionalidad Traslata	
Tipo de tarea: Desarrollo	Responsable: Henry González Olivera
Fecha inicio:	Fecha fin:
Descripción: Trasladar la figura en la dirección y distancia puesta por el usuario.	

Tabla 22 Tareas de Programación de HU6

TAREA DE PROGRAMACIÓN	
No. de tarea: 1	No.de HU: 6
Nombre de la tarea Implementar la funcionalidad Scale	
Tipo de tarea: Desarrollo	Responsable: Henry González Olivera
Fecha inicio:	Fecha fin:
Descripción: Proyectar la entidad hacia una dirección y tamaño especificado.	

Tabla 23 Tareas de Programación de HU7

TAREA DE PROGRAMACIÓN	
No. de tarea: 1	No.de HU: 7
Nombre de la tarea: Implementar la funcionalidad Rotate	
Tipo de tarea: Desarrollo	Responsable: Henry González Olivera
Fecha inicio:	Fecha fin:

Descripción: Rotar la entidad una cierta cantidad de grados dados por el usuario alrededor de un punto especificado también por el usuario.

Tabla 24 Tareas de Programación de HU8

TAREA DE PROGRAMACIÓN	
No. de tarea: 1	No.de HU: 8
Nombre de la tarea: Implementar la funcionalidad Offset	
Tipo de tarea: Desarrollo	Responsable: Henry González Olivera
Fecha inicio:	Fecha fin:
Descripción: Crear una nueva entidad a partir de la ya creada dando a lugar una nueva entidad de con nuevas dimensiones especificadas por el usuario	

Tabla 25 Tareas de Programación de HU9

TAREA DE PROGRAMACIÓN	
No. de tarea: 1	No.de HU: 9
Nombre de la tarea: Implementar la funcionalidad Break	
Tipo de tarea: Desarrollo	Responsable: Henry González Olivera
Fecha inicio:	Fecha fin:
Descripción: Dividir la figura en dos cuando la entidad arco de elipse es cortada por otra entidad cualquiera.	

Tabla 26 Tareas de Programación de HU10

TAREA DE PROGRAMACIÓN	
No. de tarea: 1	No.de HU: 10
Nombre de la tarea: Implementar la funcionalidad Trim	
Tipo de tarea: Desarrollo	Responsable: Henry González Olivera
Fecha inicio:	Fecha fin:

Descripción: Borrar la entidad seleccionada que previamente ha sido cortada el arco de elipse con otra entidad cualquiera.

3.3 Fase de pruebas

En concordancia con el autor Kent Beck, se puede afirmar que uno de los pilares de la Programación Extrema es el proceso de pruebas. La metodología XP anima a probar constantemente tanto como sea posible. Esto permite aumentar la calidad de los sistemas reduciendo el número de errores no detectados y disminuyendo el tiempo transcurrido entre la aparición de un error y su detección. También permite aumentar la seguridad de evitar efectos colaterales no deseados a la hora de realizar modificaciones y refactorizaciones. En XP las pruebas del sistema se dividen en dos grupos: pruebas unitarias y pruebas de aceptación (18).

3.3.1 Pruebas de aceptación

Las pruebas de aceptación son creadas en base a las HU, en cada ciclo de la iteración del desarrollo. El cliente debe especificar uno o diversos escenarios para comprobar que cada HU ha sido correctamente implementada. Las pruebas de aceptación son consideradas como “pruebas de caja negra”. Los clientes son responsables de verificar que los resultados de estas pruebas sean correctos (18).

Como criterio de aprobación de cada iteración se tomó que el 100% de los casos de prueba sean exitosos para pasar de iteración. El objetivo de estas pruebas no es tener un conjunto de casos escritos que cubran el 100% del código, sino poder realizarle pruebas al sistema desde el punto de vista del usuario. Para la realización de cada una de las pruebas de aceptación se siguieron una serie de pasos que se muestran a continuación:

1. Identificar todas las acciones en la HU.
2. Para cada acción escribir al menos una prueba.

Para representar las pruebas de aceptación se definieron los siguientes elementos:

- **Código:** representa al caso de prueba, incluye el número de HU, de la prueba y si posee diferentes escenarios.
- **HU:** número de la HU a la cual pertenece.
- **Nombre:** conforma el identificador del caso de prueba junto al código.
- **Descripción:** acción que debe realizar el sistema.

- **Condiciones de ejecución:** describe las características y elementos que debe contener el sistema para realizar el caso de prueba.
- **Pasos de ejecución:** pasos para realizar el caso de prueba.
- **Resultados Esperados:** descripción de la respuesta del sistema ante el caso de prueba.
- **Resultado Obtenido:** respuesta visual del sistema después de realizar el caso de prueba.
- **Evaluación de la prueba:** clasificación de la prueba en satisfactoria o insatisfactoria.

3.3.2 Pruebas de aceptación de la primera iteración

Tabla 27 UH1_P1: Desarrollar el Arco de Elipse

PRUEBA DE ACEPTACIÓN	
Código :UH1_P1	HU: 1
Nombre : Desarrollar el Arco de Elipse	
Descripción: Al inicializar el sistema se escogerá la opción de nuevo documento. Se seleccionará la opción de arco de elipse donde se procederá a pintarla mediante los puntos dados por el usuario.	
Condiciones de ejecución :	
Pasos de ejecución: <ol style="list-style-type: none"> 1. Seleccionar la opción de crear nuevo documento. 2. Seleccionar la opción de crear arco de elipse. 3. Poner el primer punto que será el punto central. 4. Poner los otros dos puntos para construir la elipse a conveniencia 5. Poner los otros dos puntos que delimitaran de dónde y hacia dónde irá el arco de elipse. 	
Resultados esperados: Se creará el arco de elipse.	
Resultado obtenido:	

Se creó el arco de elipse.

Evaluación de la prueba : Satisfactoria

Tabla 28 UH2_P1: Desarrollar Move

PRUEBA DE ACEPTACIÓN	
Código :UH2_P1	HU: 2
Nombre : Desarrollar Move	
Descripción: Ya creado el arco de elipse, para poder mover dicha entidad se seleccionará el punto central y se moverá el mouse hacia la dirección deseada, otra manera de mover el arco es seleccionando todos los puntos y arrastrar la entidad hacia el lugar deseado	
Condiciones de ejecución : El arco de elipse debe de estar creado.	
Pasos de ejecución: <ol style="list-style-type: none">1. Crear el arco de elipse.2. Seleccionar el punto central.3. Mover la entidad hacia el lugar deseado.	
Resultados esperados: Se moverá el arco de elipse al lugar deseado por el usuario.	
Resultado obtenido: Se movió el arco de elipse al lugar indicado por el usuario.	
Evaluación de la prueba : Satisfactoria	

Tabla 29 UH3_P1: Desarrollar Copy

PRUEBA DE ACEPTACIÓN

Código :UH3_P1	HU: 3
Nombre : Desarrollar Copy	
Descripción: Creado el arco este se deberá poder copiar pegar.	
Condiciones de ejecución: El arco de elipse debe de estar creado.	
Pasos de ejecución: <ol style="list-style-type: none"> 1. Crear el arco de elipse. 2. Seleccionar el arco de elipse. 3. Apretar las teclas ctrl+c para copiar. 4. Apretar las teclas ctrl+v para pegar. 	
Resultados esperados: Se copiará la entidad del arco de elipse.	
Resultado obtenido: Se copió la entidad del arco de elipse	
Evaluación de la prueba : Satisfactoria	

Tabla 30 UH4_P1: Desarrollar Delete

PRUEBA DE ACEPTACIÓN	
Código :UH4_P1	HU: 4
Nombre : Desarrollar Delete	
Descripción: Al seleccionar la entidad de arco de elipse se podrá borrar.	
Condiciones de ejecución: El arco de elipse debe de estar creado.	

<p>Pasos de ejecución:</p> <ol style="list-style-type: none"> 1. Crear el arco de elipse. 2. Seleccionar la entidad creada. 3. Presionar la tecla delete.
<p>Resultados esperados:</p> <p>Se borrará la entidad arco de elipse.</p>
<p>Resultado obtenido:</p> <p>Se borró la entidad arco de elipse.</p>
<p>Evaluación de la prueba : Satisfactoria</p>


En la primera iteración se desarrollaron 2 etapas de pruebas. En total se ejecutaron un total de 7 casos de pruebas. En la primera etapa se detectaron un total de 3 no conformidades representando 46% de resultados insatisfactorios, una vez corregidos, se desarrolló la segunda etapa de pruebas donde no se encontraron errores representando el 100% de resultados satisfactorios.


3.3.3 Pruebas de aceptación de la segunda iteración

Tabla 31 UH5_P1: Desarrollar Traslate

PRUEBA DE ACEPTACIÓN	
Código : UH5_P1	HU: 5
Nombre : Desarrollar Traslate	
<p>Descripción:</p> <p>Al seleccionar esta funcionalidad se le mostrará un diálogo al usuario, con el cual deberá de interactuar y podrá trasladar la entidad creada en una dirección y distancia especificada por el usuario.</p>	
<p>Condiciones de ejecución:</p> <p>El arco de elipse debe de estar creado.</p>	
<p>Pasos de ejecución:</p> <ol style="list-style-type: none"> 1. Crear el arco de elipse. 	

2. Seleccionar la funcionalidad Traslata

3. Seleccionar la entidad creada con el botón de la izquierda ()

4. Seleccionar con el botón de la derecha () la distancia y dirección en cual se quiere trasladar la figura.

Resultados esperados:


Se trasladará la entidad arco de elipse.


Resultado obtenido:

Se trasladó la entidad arco de elipse.

Evaluación de la prueba : Satisfactoria

Tabla 32 UH6_P1: Desarrollar Scale

PRUEBA DE ACEPTACIÓN	
Código : UH6_P1	HU: 6
Nombre : Desarrollar Scale	
Descripción: Al seleccionar esta funcionalidad se mostrará un diálogo al usuario con el cual deberá de interactuar y podrá escalar la entidad en la cantidad y la dirección deseada.	
Condiciones de ejecución: El arco de elipse debe de estar creado.	
Pasos de ejecución: <ol style="list-style-type: none">1. Crear el arco de elipse.2. Seleccionar la funcionalidad Scale3. Seleccionar la entidad con el botón de la izquierda ()4. Poner en la cantidad que se quiere escalar dicha entidad.	

5. Seleccionar con el botón de la derecha () el punto al cual se quiere direccionar el escalado.
6. Dar a Done para ver el resultado.

Resultados esperados:



Se escalará en la dirección deseada la entidad arco de elipse.

Resultado obtenido:

Se escaló en la dirección deseada la entidad arco de elipse.

Evaluación de la prueba : Satisfactoria

Tabla 33 UH7_P1: Desarrollar Rotate

PRUEBA DE ACEPTACIÓN	
Código : UH7_P1	HU: 7
Nombre : Desarrollar Rotate	
Descripción: Al seleccionar esta funcionalidad se mostrará un diálogo con el cual se deberá de interactuar y se podrá rotar la entidad hacia donde se desee la cantidad de ángulos requeridos.	
Condiciones de ejecución: El arco de elipse debe de estar creado.	
Pasos de ejecución: <ol style="list-style-type: none"> 1. Crear el arco de elipse. 2. Seleccionar la funcionalidad Rotate. 3. Seleccionar la entidad creada con el botón de la izquierda () 4. Poner la cantidad de ángulos deseados para rotar. 5. Con el botón de la derecha () seleccionar el punto en el que alrededor de este se va a rotar la entidad. 	
Resultados esperados:	

Se rotará la entidad arco de elipse alrededor del punto puesto.

Resultado obtenido:

No se rotó la entidad arco de elipse alrededor del punto puesto.

Evaluación de la prueba : Satisfactoria

En la segunda iteración de pruebas se desarrollaron 2 etapas. En total se ejecutaron un total de 10 casos de pruebas. En la primera etapa se detectaron un total de 4 no conformidades representando 40% de resultados insatisfactorios, una vez corregidos, se desarrolló la segunda etapa de pruebas donde no se encontraron errores representando el 100% de resultados satisfactorios.

3.3.4 Pruebas de aceptación de la tercera iteración

Tabla 34 UH8_P1: Desarrollar Offset

PRUEBA DE ACEPTACIÓN	
Código : UH8_P1	HU: 8
Nombre : Desarrollar Offset	
Descripción : Al Seleccionar esta funcionalidad se puede crear otra entidad de diferentes dimensiones pero con el mismo punto central.	
Condiciones de ejecución : El arco de elipse debe de estar creado.	
Pasos de ejecución: <ol style="list-style-type: none">1. Crear el arco de elipse.2. Seleccionar la funcionalidad Offset.3. Seleccionar la entidad creada.4. Mover el mouse para ver el tamaño deseado de la nueva entidad.	
Resultados esperados: Se creará una nueva entidad arco de elipse con diferente tamaño.	

Resultado obtenido:

Se crea la entidad deseada.

Evaluación de la prueba : Satisfactoria

Tabla 35 UH9_P1: Desarrollar Break

PRUEBA DE ACEPTACIÓN	
Código : UH9_P1	HU: 9
Nombre : Desarrollar Break	
Descripción : Al seleccionar esta funcionalidad se podrán crear dos nueva entidades donde antes solo había una compuesta.	
Condiciones de ejecución : El arco de elipse debe de estar creado, este debe de estar interceptado por cualquier otra entidad.	
Pasos de ejecución: <ol style="list-style-type: none">1. Crear el arco de elipse.2. Crear cualquier otra entidad de manera que corte al arco de elipse.3. Seleccionar la funcionalidad Break.4. Dar clic en la entidad nueva creada y esta mostrará por donde se cortarán para crear dos entidades nuevas.	
Resultados esperados: Se dividirá en dos nuevas entidades al arco de la elipse.	
Resultado obtenido: No se cortó el arco por las intersecciones.	
Evaluación de la prueba : Satisfactoria	

Tabla 36 UH10_P1: Desarrollar Trim

PRUEBA DE ACEPTACIÓN	
Código : UH10_P1	HU: 10
Nombre : Desarrollar Trim	
Descripción : Al seleccionar esta funcionalidad se dividirá donde antes había una entidad compuesta y borrará la seleccionada.	
Condiciones de ejecución : El arco de elipse debe de estar creado, debe estar cortado por cualquier otra entidad.	
Pasos de ejecución: <ol style="list-style-type: none">1. Crear el arco de elipse.2. Crear cualquier otra entidad de manera que corte al arco de elipse.3. Seleccionar la funcionalidad Trim.4. Seleccionar la parte que quieres borrar cuando corte la entidad creada.	
Resultados esperados: Después de cortar se borrará la parte seleccionada.	
Resultado obtenido: Se borró toda la figura.	
Evaluación de la prueba : Satisfactoria	

En la tercera iteración se desarrollaron 2 etapas. En total se ejecutaron un total de 8 casos de pruebas. En la primera etapa se detectaron un total de 4 no conformidades representando 50% de resultados insatisfactorios, una vez corregidos, se desarrolló la segunda etapa de pruebas donde no se encontraron errores representando el 100% de resultados satisfactorios.

3.4 Conclusiones parciales.

Al concluir este capítulo se puede apreciar las diferentes pruebas realizadas a la aplicación dando como resultado que la entidad arco de elipse está completa incluyendo una correcta implementación dando lugar un orden cronológico para la implementación de las funcionalidades y la entidad.

CONCLUSIONES

Con la culminación de este trabajo se incorporó al modelador geométrico de AsiXMec la entidad arco de elipse y las funcionalidades básicas que actúan sobre esta entidad. Por lo tanto se concluye que:

- El desarrollo de este trabajo permite crear un arco de elipse de forma directa.
- El arco de elipse y las funcionalidades que operan sobre este son una extensión de la entidad elipse.

RECOMENDACIONES

Para futuras versiones del producto AsiXMec se recomienda:

- Incorporar otras opciones de creación del arco de elipse.

BIBLIOGRAFÍA

Referencias

1. **METODO SISTEMICO PARA LA RESOLUCION DE PROBLEMAS.**
2. 3D CAD Portal. [En línea] <http://www.3dcadportal.com/3d-software/cad/>.
3. Scrid. [En línea] <https://es.scribd.com/doc/17754860/Sistema-CAD>.
4. *AsiXMec aplicación cubana para el diseño e ingeniería asistida por computadoras.* 1, s.l. : Copextel, 2015, GIGA, pág. 68. 1028-270x.
5. Ochoa, Andrés. Humanos. [En línea] 2014. <https://humanos.uci.cu/2014/11/asixmec-aplicacion-cubana-para-el-diseno-e-ingenieria-asistida-por-computadora/>.
6. Algesa. [En línea] <http://www.alegsa.com.ar/Dic/modulo.php>.
7. **DEFINICIÓN DE ELIPSE.**
8. Ditor. [En línea] http://www.ditor.com/geometria_analitica/elipses.html.
9. Pagina oficial de la Universidad Nacional Autónoma de México. [En línea] 2009. <http://www.prepa5.unam.mx/wwwP5/profesor/publicacionMate/14X.pdf>.
10. Vitutor. [En línea] <http://www.vitutor.net/2/1/22.html>.
11. Santiago Cogollos, Stephan Marini, Pablo Soto, Hector Esteban, Jose V. Mono, Vicente E. Bona and Benito Gimeno. *Modal Computation of Arbitrary Waveguides Composed of Linear, Circular and Elliptical Arcs.*
12. Build a better connected world. [En línea] <https://www.qt.io/>.
13. Employee Monitoring & Threat Prevention. [En línea] <https://www.teramind.co/>.
14. Open CASCADE Technology. [En línea] <http://dev.opencascade.org/doc/refman/html/index.html>.
15. García, María N. Moreno. Gestión de recursos Informáticos del Departamento de Informatica y Automatica. [En línea] <http://avellano.usal.es/~mmoreno/ASTema2.pdf>.
16. Modelos De Procesos De Software. [En línea] <https://www.clubensayos.com/Tecnolog%C3%ADa/Modelos-De-Procesos-De-Software/36779.html>.
17. [En línea] http://ingenieriadesoftware.mex.tl/52753_XP---Extreme-Programing.html.
18. Joskowicz, Jose. *Reglas y prácticas en Extreme Programing.* Universidad de Vigo.España : s.n., 2008.
19. Beck, Kent. *Extreme Programming Explained.* 1999.
20. *Diseño Dirigido por Responsabilidades con los patrones GRASP.*
21. Erich Gamma, Richal Helm, Ralf Jonhson, John Vlissides. *Design Patterns.*
22. Portal colaborativo de Open CASCADE Technology. [En línea] <http://dev.opencascade.org/doc/refman/html/index.html>.
23. Siemens. [En línea] https://www.plm.automation.siemens.com/es_sa/plm/cad.shtml.