# DISEÑO DE UNA ARQUITECTURA PARA LA DEFINICIÓN DE UNIDADES DE COMBATE QUE APOYAN EL ENTRENAMIENTO TÁCTICO MILITAR EN LA ESCUELA NAVAL DE CADETES “ALMIRANTE PADILLA” –ENAP


INVESTIGADORESF

Raúl Alejandro Téllez Mora

Christian Gabriel Torres Martínez



UNIVERSIDAD DE CARTAGENA

FACULTAD DE INGENIERÍA

PROGRAMA DE INGENIERÍA DE SISTEMAS



CARTAGENA DE INDIAS, 2015

# DISEÑO DE UNA ARQUITECTURA PARA LA DEFINICIÓN DE UNIDADES DE COMBATE QUE APOYAN EL ENTRENAMIENTO TÁCTICO MILITAR EN LA ESCUELA NAVAL DE CADETES “ALMIRANTE PADILLA” –ENAP

TESIS DE GRADO


Grupo de investigación

PIEMOVIL

INVESTIGADORES

Raúl Alejandro Téllez Mora

Christian Gabriel Torres Martínez

Director: Yasmín Moya, Msc. (Universidad de Cartagena)



UNIVERSIDAD DE CARTAGENA

FACULTAD DE INGENIERÍA

PROGRAMA DE INGENIERÍA DE SISTEMAS


CARTAGENA DE INDIAS, 2015

UNIVERSIDAD DE CARTAGENA

**Tesis de Grado:** DISEÑO DE UNA ARQUITECTURA PARA LA DEFINICIÓN DE UNIDADES DE COMBATE QUE APOYAN EL ENTRENAMIENTO TÁCTICO MILITAR EN LA ESCUELA NAVAL DE CADETES "ALMIRANTE PADILLA" –ENAP

**Autores:** RAÚL ALEJANDRO TÉLLEZ MORA
CHRISTIAN GABRIEL TORRES MARTÍNEZ

**Director:** MSc. YASMÍN MOYA VILLA

**Nota de Aceptación**

_____

_____

_____

_____

**Presidente del Jurado**

_____

**Jurado**

_____

**Jurado**

Cartagena de Indias D.T. y C. ____ de _____ de 2015

## RESUMEN

La Escuela Naval de Cadetes Almirante Padilla, como institución distinguida de la Costa Caribe colombiana, se ve en la necesidad de estar a la vanguardia en lo que concierne a herramientas tecnológicas de enseñanza. Dentro de su formación, utiliza los simuladores para capacitar estratégicamente en situaciones de defensa, ataque, apoyo, entre otras. La Institución ha suplido la necesidad de entrenar a sus estudiantes con un software adquirido que simula una batalla naval, sin embargo La Escuela no posee un simulador a la medida que cuente con todos los requerimientos que consideran relevantes a la hora de entrenar al estudiantado, por lo que en ocasiones se torna complicado la forma de llevar a cabo el proceso.

Con la finalidad de suplir la necesidad descrita en el párrafo anterior se planteó el presente proyecto, cuyo objetivo radicó en diseñar una arquitectura de software donde se definan las unidades militares como componente de un sistema de entrenamiento táctico a largo plazo, con el fin de contribuir a un proyecto que sea competente, útil y didáctico para el buen aprendizaje en la formación. Simular el comportamiento de las unidades militares navales en un enfrentamiento, dando cumplimiento los objetivos de esta y convirtiéndose en un proyecto totalmente implementable para satisfacer las necesidades de aprendizaje de los cadetes de La Escuela.

Gracias al trabajo realizado se obtuvo un prototipo funcional que refleja la arquitectura diseñada, además de un completo análisis y revisión de literatura que pudo ser conjugada en el estado del arte del presente documento.

Por último, esta arquitectura de software permite, además de conocer los lineamientos técnicos básicos que deberá tener la aplicación en cuanto a Unidades Militares se refiere, fue apoyada mediante un software de simulación del mundo real (Vensim), para su mejor interpretación; La instancia de esta arquitectura también unifica criterios, encapsula comportamientos comunes a lo largo de toda la aplicación, reduce e identifica tempranamente los riesgos técnicos y ayuda a cumplir con los

requerimientos de calidad., además de un completo análisis y revisión de literatura que pudo ser conjugada en el estado del arte del presente documento.

**Palabras clave:** *Arquitectura De Software, Simulación.*

## ABSTRACT

The Naval Academy of Cadets Almirante Padilla as an institution distinguished from the Colombian Caribbean Coast, is the need to be at the forefront as far as technology is concerned teaching tools. Within its training simulators used to train strategically in situations of defense, attack, support, among others. The institution has supplied the need to train their students with an acquired software that simulates a naval battle, however the school does not have a simulator to measure receiving all the requirements considered relevant when it comes to training the students, by which sometimes it becomes complicated way to carry out the process.

In order to meet the need described in the preceding paragraph of this project, which aims lay in designing a software architecture where military units as a component of a system of tactical training in the long term be defined, in order he arose contribute a project that is competent, useful and educational for good learning in training. Simulate the behavior of naval military units in a confrontation, fulfilling the objectives of this and becoming a fully implementable project to meet the learning needs of cadets School.

Thanks to the work of a functional prototype that reflects the architecture designed addition to a thorough analysis and review of literature that could be conjugated in the state of art of this document was obtained.

Finally, this software architecture also allows to know the basic technical guidelines that should have application in terms of military units are concerned, was supported by a software simulation of the real world (Vensim), for better interpretation; The instance of this architecture also unifies criteria, encapsulate common behaviors throughout the entire application, reduce and early identifies technical and helps meet quality requirements risks., Plus a full analysis and review of literature that could be conjugate in the state of art of this document.



***Keywords***: software architecture, simulation.

## DEDICATORIA

Dedico esta tesis a Dios quien es el motor de nuestras vidas.

A nuestros amigos quienes fueron un gran apoyo emocional durante el tiempo en que escribía esta tesis.

A nuestros padres quienes me apoyaron todo el tiempo.

A nuestros maestros quienes nunca desistieron al enseñarme, en especial a mi tutora Yasmín Moya Villa, por su paciencia y amor, al director David Franco Borré por solucionar cada problema que le ponía en sus manos, y al Decano Miguel García Bolaños por su orientación y consejos a lo largo de la carrera, a ellos gracias por continuar depositando su esperanza en mí.

A todos los que me apoyaron para escribir y concluir esta tesis.

Para ellos es esta dedicatoria de tesis, pues es a ellos a quienes se las debo por su apoyo incondicional.

## TABLA DE CONTENIDO

## ÍNDICE DE ILUSTRACIONES

## ÍNDICE DE TABLAS

# 1. INTRODUCCIÓN

Dentro de los múltiples campos que abarca la Ingeniería De Sistemas se pueden encontrar soluciones útiles a la hora de resolver problemas en campos variados como las redes, telecomunicaciones, soporte y educación, lo que finalmente conlleva a la satisfacción de una o más necesidades. La línea de la ingeniería del software no es la excepción y, desde sus inicios, ha ido en un proceso continuo de mejoramiento buscando finalmente producir software de calidad.

La vertiente tradicional -partir de una necesidad y desembocar en su solución-, surgió como respuesta a los inconvenientes que se estaban presentando en el área debido a la crisis del software y fue tan alto su impacto que, hoy día, los productos de alta calidad se concretan en tiempos mínimos, con procesos más manejables y completamente tecnificados y estandarizados, de una forma muy diferente a como se llevaba unos años atrás. Es así que con ayuda de los múltiples patrones de software propuestos por los estudiosos del área (Gamma, Helm, Johnson, & Vlissides, 1995), además de las buenas prácticas de programación y cambios de paradigma -en primer momento estructurales, luego orientados a objetos-, la ingeniería del software se ha convertido actualmente en una de las áreas en donde más esfuerzo investigativo se emplea y por ende, una de las áreas que más innovación introduce y que más interés despierta (Canfora & Di Penta, 2007).

Como solución a una necesidad presentada por la Escuela Naval de Cadetes “Almirante Padilla (De ahora en adelante ENAP), referente a la manera de entrenar a sus cadetes en situaciones de combate y resumida en la pregunta que se realizó como planteamiento del problema, ¿Cómo se puede asegurar un buen desarrollo, codificación y estandarización de las unidades militares navales para el apoyo durante el entrenamiento táctico-militar de los estudiantes de la ENAP?, nace esta arquitectura de software (Enfocada única y exclusivamente a las unidades militares), la cual es una base que soporta y complementa el posterior desarrollo de un simulador de batalla naval. A su vez, el proyecto aplicó conocimientos adquiridos en temáticas electivas relacionadas con Arquitectura de Software

y Tópicos Avanzados de Ingeniería de Software del Programa de Ingeniería de Sistemas de la Universidad de Cartagena, proporcionándole además a la Universidad, aplicación de nuevos conceptos encontrados dentro de la investigación.

La seguridad del producto final desarrollado (Simulador de batalla naval) se vería potenciado puesto que gracias al análisis arquitectural de las Unidades Militares es más fácil encontrar fallas de seguridad a través de los cuales el software puede presentar mal funcionamiento.

Finalmente el campo académico de la Ingeniería De Sistemas también obtiene beneficios gracias al proyecto realizado, puesto que los resultados obtenidos pueden servir como punto de partida y guía para nuevas iniciativas que se encuentren enmarcadas dentro del área de la ingeniería del software y la simulación, aumentando la producción científica y el nivel de los productos construidos además de generar interés en la academia para continuar desarrollando investigaciones en el área.

Para llevar a cabo el presente proyecto se ha clasificado la investigación a partir de sus características como una *Investigación Cualitativa* y sus técnicas y postulados reafirmados gracias al *Análisis Documental* y revisión de la literatura realizada a fuentes académicas verificables y científicamente aceptadas que soportan el trabajo realizado.

Inicialmente se construyó un estado del arte que permitió conjugar los antecedentes, investigaciones realizadas y contexto actual de la temática abordada, cuyo foco de la investigación se dividió en 2 partes, Arquitectura de Software y Simuladores. Los trabajos relacionados a nuestra investigación sirven como punto de partida para iniciar la búsqueda de una solución TI a las necesidades planteadas por la ENAP. Posteriormente se realizó un análisis al estado del arte construido, que permitió obtener y clasificar una serie de necesidades que se tienen el campo del análisis arquitectura, se pudo observar que en América latina se han realizado proyectos enfocados al área militar, el entrenamiento de aprendices con escenarios simulados para evaluar el manejo de las unidades de guerra con las que cuenta el Estado y las acciones en diversas situaciones (Plúas Andrade, Rivera Cárdenas, & Cortez Alvarez). Se observó que las arquitecturas encontradas, cuyo fin es

satisfacer la problemática presentada, simplemente se contextualizan a un único entorno, al cual será aplicada, lo cual dificulta la versatilidad y variabilidad de los escenarios que influyen en las Unidades Militares, además de que los productos software que buscan satisfacer el problema planteado, fueron desarrollados con un enfoque hacia el entretenimiento en lugar de entrenamiento de cadetes militares. Uno de los productos software que más se asemeja al producto final de esta investigación es el SETAC, un sistema de entrenamiento de toma de decisiones militares, que simula la conducción táctica de unidades desde el nivel División hasta el de Sección o Pelotón, en el cual los comandantes subordinados conducen las acciones desde sus puestos de mando, este software tiene una interfaz de usuario poco amigable y de acuerdo a los requerimientos planteados por la ENAP, no sería un software desarrollado a la medida. partir de las necesidades encontradas se realizó una arquitectura de software fundamentada en un modelo orientado a objetos y eventos discretos debido a la necesidad de exactitud de las simulaciones, y finalmente se realizó la validación de la arquitectura propuesta, mediante la aplicación de la solución en un software de simulación de sistemas reales, lo que soporta y valida el escenario de prueba propuesto.

# 2. OBJETIVOS Y ALCANCE

## 2.1. OBJETIVO GENERAL

- Diseñar una arquitectura soportada en cloud computing, para la definición de unidades militares navales, que permita estudiar la composición y el papel que cumplen dichas unidades en el entrenamiento táctico militar que realiza la Escuela Naval de Cadetes “Almirante Padilla”.

## 2.2. OBJETIVOS ESPECÍFICOS

- Identificar los requerimientos funcionales para la definición de unidades que permitan saber las entradas, comportamientos y salidas de las mismas dentro de la interacción con los escenarios de combate.

- Diseñar una arquitectura para definir unidades militares navales, a partir de los requerimientos identificados.

- Desarrollar un escenario de prueba y simulación para validar la arquitectura diseñada.

- Realizar pruebas de caja blanca y caja negra para validar la arquitectura, a partir del prototipo implementado.

## 2.3. ALCANCE

El presente proyecto de investigación, está limitado desde el punto de vista espacial y conceptual al diseño de una arquitectura de software, dedicada a definir unidades militares. Mediante un prototipo funcional se valida y prueba la arquitectura con el fin de garantizar una buena implementación y buen funcionamiento de esta.

Desde el punto de vista temporal, el periodo en que se desarrolla el presente proyecto de investigación gira alrededor de 32 semanas (8 meses),

El producto final a entregar será la arquitectura de software que abarca solo los temas respectivos del enfoque de la investigación (Unidades Militares), tendrá un proceso de ciclo de vida que incluye análisis, diseño y pruebas, manejará tipos de datos exactos para el manejo de simuladores físicos y con el fin de aplicar modelos matemáticos, usará fuentes de datos y bases de datos proporcionadas por La Escuela bajo un acuerdo de confidencialidad y sólo se vincularán al desarrollo de la investigación las organizaciones mencionadas previamente (Universidad de Cartagena y La Escuela).

La presente investigación ha sido enmarcada en el campo del desarrollo de software, específicamente en las vertientes conocidas como diseño y arquitectura de software, cuyos contenidos y herramientas permitieron el cumplimiento de los objetivos de la investigación. Gracias a la realización del estado del arte referente a los métodos de análisis a arquitecturas de software en el ámbito de la simulación actualmente existentes se pudieron determinar problemáticas y necesidades que buscan ser resueltas mediante la arquitectura de software, que representa el entregable principal de esta iniciativa.

# 3. MARCO DE REFERENCIA

## 3.1. CONTEXTUALIZACIÓN: ANÁLISIS DE UNA ARQUITECTURA DE SOFTWARE

Luego de que la arquitectura de software está definida, nacen interrogantes sobre cómo estar seguro de que la arquitectura desarrollada es la correcta para mi software y cumple con los requisitos iniciales.

No es tarea fácil encontrar respuesta a esta pregunta, de lo que si se tiene certeza es que la arquitectura de software es el pilar fundamental de cualquier software.

Cuanto más temprano se encuentre un problema en un proyecto de software, mejor. Corregir un error en etapas tempranas del diseño representa un costo mucho menor al costo de corregir ese mismo error en la fase de verificación o implementación. Dado que la arquitectura es un producto temprano de la fase de diseño, esta tiene un profundo efecto en el sistema y en el proyecto.

Una mala arquitectura puede llevar a un proyecto al fracaso debido a que esta puede determinar la estructura del proyecto: configuración, agenda y presupuesto, alcance, entre otros aspectos. Puntualmente esas son las razones por las que se debe analizar o evaluar una arquitectura de software, para no asumir costes innecesarios luego de finalizado el ciclo de vida del software.

### 3.1.1. Métodos para analizar arquitecturas de software

Los métodos de diseño arquitectónico y los métodos de evaluación arquitectónica consideran la concepción de un sistema de software a un alto nivel de abstracción, con base en una visión arquitectónica. Algunos de estos métodos tienen como objetivo la generación de una arquitectura que responda a un conjunto de requisitos iniciales. Otros están dirigidos

a evaluar configuraciones arquitectónicas ya existentes para escoger la mejor de acuerdo a un conjunto de requisitos. (Francisca Losavio, 2010)

#### 3.1.1.1. ¿Que implica analizar una arquitectura de software?

El proceso de arquitectura de software toma los requisitos del cliente, los analiza y produce un diseño para obtener un software que solventará sus necesidades. Los diseños exitosos de software deben medir las complejidades infalibles que surgen debido a requisitos complicados; cumplir con los principios de diseño y las buenas técnicas de procedimiento que han evolucionado con el tiempo; y complementar el hardware moderno, las redes y los sistemas de administración. Una arquitectura incuestionable implica tener mucha experiencia en temas teóricos y prácticos, así como el enfoque necesario para convertir lo que al parecer son escenarios y requisitos comerciales aproximados en diseños de trabajo estables y prácticos.

Ahora, analizar la arquitectura de software implica especificar una solución ordenada que solucione todos los requisitos técnicos y operacionales y, a la vez, mejorar los atributos comunes de calidad como rendimiento, seguridad y capacidad de administración. Además, implica una serie de decisiones fundamentadas en una extensa gama de factores, y cada una de esas decisiones puede tener un impacto importante sobre la calidad, rendimiento, mantenimiento y éxito general de ese software.

El mayor beneficio que brinda la evaluación de una arquitectura, es que descubre los problemas que si se hubiesen dejado sin descubrir, habría sido mucho más costoso corregirlos luego. En breve, una evaluación produce una mejor arquitectura.

Algunos beneficios más que brinda la evaluación se presentan a continuación.

**Potencia una unión en las metas específicas de calidad**

El rol del stakeholder[1] es articular las metas de calidad que la arquitectura está obligada a conseguir para ser considerada exitosa. Estas metas no siempre son capturadas en algún documento de requerimientos.

**Fomenta una definición clara de la arquitectura**

El arquitecto convoca a un grupo de personas, no en privado, para explicarles la creación de la arquitectura, en detalle y sin ambigüedades. El proyecto se ve beneficiado cuanto antes se realice esta explicación.[1]

**Mejora la calidad de la documentación de la arquitectura**

Generalmente, la evaluación pide documentación que aún no está terminada, entonces se designa alguien del plantel a terminarla. Una vez más, el proyecto se ve beneficiado porque se ingresa al desarrollo mejor preparado al tener los documentos terminados.

**Muestra oportunidades de reuso**

Los stakeholders y el equipo de evaluación provienen de afuera del proyecto de desarrollo, pero trabajan o están familiarizados con otros proyectos. Por lo tanto, ambos están en una buena posición para detectar componentes que pueden ser reusados en otros proyectos, o conocer componentes que ya existen que pueden ser utilizados en el proyecto actual.

[1] Stakeholder: El término agrupa a trabajadores, organizaciones sociales, accionistas y proveedores, entre muchos otros actores clave que se ven afectados por las decisiones de una empresa.

**Resultan mejoras en las arquitecturas**

Las organizaciones que practican la evaluación arquitectónica como un estándar de su proceso de desarrollo, reportan una mejora en la calidad de las arquitecturas que son

evaluadas. Las arquitecturas evaluadas presentan mejorías no sólo al inicio del proyecto, también al culminar este.

### 3.1.1.2. ¿Qué técnicas de análisis existen y cómo compararlas?

Una mala arquitectura puede llevar a un proyecto al fracaso. Todos los requerimientos de calidad pueden quedar insatisfechos. La arquitectura determina la estructura del proyecto: configuración, agenda y presupuesto, alcance, entre otros aspectos. Es mejor cambiar la arquitectura antes que otros artefactos, que están basados en ella, se estabilicen.

Realizar una evaluación de la arquitectura es la manera más económica de evitar desastres. Se ha llevado a cabo una investigación exhaustiva de las técnicas o métodos enfocados en analizar la visión de una arquitectura. En este trabajo se han seleccionado los siguientes métodos: SAAM (Scenario-Based Analysis of Software Architecture) (Bass et al. 2003); ALPSM (Architecture Level Prediction of Software Maintenance) (Bengtsson & Bosch, 1999); AQA (Architecture Quality Assessment) (Hilliard II et al. 1997); SAE (Software Architecture Evaluation) (AT&T, 1993); FAAM (Family-Architecture Analysis Method) (Dolan, 2001); ASAAM (Scenario-Based Analysis of Software Architecture) (Tekinerdogan, 2003); ALMA (Architecture Level Modifiability Analysis) (Bengtsson et al. 2004); QAW (Quality Attribute Workshps) (Barbacci et al. 2003); ATAM (Architecture Tradeoff Analysis Method) (Clements et al. 2002); PASA (Performance Assessment of Software Architectures) (Connie & Williams, 2002); ARID (Active Reviews for Intermediate Designs) (Clements, 2000); CBSP (Grünbacher et al. 2003); PRESKRIPTOR

(Brandozzi & Perry, 2002); VAP (Visual Architecture Process) (Bredemeyer & Malan, 2005); CBAM-WIN WIN (Cost Benefit Analysis Method, combinado con el método de negociación de requisitos WIN WIN) (In et al. 2001); ABD (Architecture Based Design) (Bass et al. 2000); SACAM (Software Architecture Comparison Analysis Method) (Bachmann et al. 2003);SARM (Software Architecture Reengineering Method) (Bengtsson & Bosch, 1998); Bosch (2000); Proteus (Chung et al. 2002); Lamsweerde (2003); MECABIC (Método de Evaluación de Arquitecturas de Software Basadas en Componentes) (González et al. 2005); Losavio-Chirinos-Lévy-Ramdane Cherif (2003); CBAM (Cost Benefit Analysis Method) (Asundi & Kazman, 2001); QUADRAD (Quality-Driven Architecture Development) (Thiel, 2005); ADD (Attribute-Driven Design) (Bass et al. 2003); ASAA (Applied Software Architecture Approach) (Hofmeister et al. 2000.

Para comparar las técnicas anteriormente mencionadas se pueden utilizar los conceptos arquitectónicos estructurales comúnmente aceptados los cuales son: componentes, conectores, estilos arquitectónicos, patrones, puertos, interfaces y vistas. Estos conceptos son utilizados para ocultar detalles de implementación y de diseño detallado. En la tabla se pueden observar los conceptos asumidos por cada método estudiado (Losavio & Guillen-Drija, 2010).

| | Conceptos Arquitectónicos | | | | | | | | | | Otros Conceptos | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Componentes | Conectores | Puertos | Interfaces | Vistas | Estilos arquitectónicos | Patrones Arquitectónicos | Propiedades | Estrategias Arquitectónicas | Patrones de Diseño | Unidades de Software | Conexiones | Restricciones | Módulos | Relaciones | Topologías | N/A |
| SAAM | • | | | | | | | | | | | | | | | | |
| ATAM | • | | | | | | | | | | | | | | | | |
| AQA | • | | | | • | | | | | | | • | • | | | | |
| FAAM | • | | | | | | | • | | | | | | | • | | |
| ALMA | • | | | | | | | | | | | | | | | | |
| SAE | | | | | | | | | | | | | | | | | • |
| ALPSM | • | | | | | | | | | | | | | | | | |
| ASAAM | • | | | | | | | | | | | | | | | | |
| SACAM | • | • | | | | | | | | | | | | | | | |
| PASA | | | | | | | • | | | | • | | | | | | |
| CBAM-WIN WIN | | | | | | | | | • | | | | | | | | |
| CBAM | | | | | | | | | • | | | | | | | | |
| QAW | | | | | | | | | | | | | | | | | • |
| CBSP | • | • | | | | | | | | | | | | | | • | |
| ARID | | | | | | | | | | | | | | | | | • |
| PRESKRIPTOR | • | • | | | | | | • | | | | | | | | | |
| MECABIC | • | | • | | • | | | | | | | | | | | | |
| ASAA | • | • | | • | | | | | | | | | | | | | |
| Losavio | • | • | | | | | • | • | | | | | • | | | | |
| VAP | • | | | • | | | | | | | | • | | | | | |
| Proteus | | | | | | • | • | | | | | | | | | | |
| Bosch | • | • | | | | • | • | | | • | | | | | | | |
| Lamsweerde | • | • | | | | • | • | | | | | | | | | | |
| QUADRAD | • | • | | | • | | • | | | | | | | | | | |
| ADD | | | | • | | | | | | | | | | • | | | |
| ABD | • | | | | | | | | | | | | | | | | |
| SARM | • | | | | | | | | | | | | | | | | |

**Tabla 1. Conceptos arquitectónicos utilizados en los métodos estudiados (Losavio & Guillen-Drija, 2010)**

## HERRAMIENTAS DE ANÁLISIS, DISEÑO Y EVALUACIÓN DE ARQUITECTURAS DE SOFTWARE

Kazman (1996) plantea que una herramienta de análisis y diseño de arquitecturas de software debe cumplir con ciertos requerimientos, entre los que se encuentran:

• Debe ser capaz de describir cualquier arquitectura. En principio, debe permitir la especificación gráfica de arquitecturas, de forma similar a como se lleva a cabo en un equipo de desarrollo. Los esbozos gráficos de la arquitectura se realizan para facilitar el diseño, la exploración y la comunicación. Para lograr el mejor cumplimiento del objetivo, la herramienta deberá hacer uso de los componentes y los conectores que el equipo de desarrollo actualmente usa, más que el uso de un grupo preestablecido por la herramienta.

• Debe ser capaz de añadir elementos arquitectónicos de forma recursiva, y poder establecer asociaciones semánticamente significativas con elementos en otros niveles de abstracción. El cumplimiento de este requerimiento es crucial para soportar el refinamiento iterativo y el análisis de arquitecturas, donde el análisis es significativo a cualquier nivel de refinamiento. Por ejemplo, debería permitir esbozar un nodo en la arquitectura bajo el nombre “base de datos”, y permitir preguntas importantes de diseño, o ejecutar análisis de desempeño, sin la necesidad de mayor descomposición del elemento definido. Con el cumplimiento de este requerimiento, la herramienta debería permitir el diseño y análisis arquitectónico en cualquier grado de resolución. Por supuesto, a mayor grado de resolución, mejores deben ser las respuestas a la hora de comparar varias alternativas.

• Debe ser capaz de determinar la concordancia de interfaces entre componentes, tanto dentro de la arquitectura que se está diseñando, como con sistemas externos. De igual forma, debe estar en capacidad de determinar la correspondencia de la arquitectura implementada con la arquitectura diseñada.

• Debe contar con la capacidad de realizar ingeniería de reverso.

• Debe ser capaz de analizar arquitecturas con respecto a métricas.

• Debe asistir al desarrollador en el diseño, tanto como una actividad creativa, como una actividad de análisis. En específico, se propone que debe soportar distintos métodos de diseño, y los procesos que estos métodos implican.

• Debe funcionar como un repositorio que contenga diseños, trozos de diseño, justificaciones de diseño y escenarios. Como repositorio, debe permitir la búsqueda de una arquitectura, la extracción de subconjuntos significativos de ésta, y también permitir su actualización.

• Debe proveer la generación de plantillas de código, para simplificar la transición del diseño al código, contribuyendo así con la consistencia entre ambos. De igual forma, debe proveer herramientas de modelaje de control de datos y flujo, así como también modelaje de desempeño.

El planteamiento de Kazman (1996) no contempla aspectos de evaluación de las arquitecturas de software diseñadas con una herramienta que cumpla con los requerimientos mencionados. Por otro lado, considerando que la calidad de un sistema de software está determinada en gran parte por su arquitectura, resulta de particular interés extender el alcance de tal herramienta, agregando la capacidad de evaluación del diseño generado.

La intención es entonces resaltar algunos de los elementos que aún pueden añadírsele a una herramienta como la propuesta por Kazman (1996), que derive en un ambiente que tenga otras capacidades no consideradas aún. Entre ellas se encuentran:

• Inclusión de nuevos elementos de descripción, tales como los ADL, vistas arquitectónicas y distintas notaciones.

• Posibilidad de evaluar la calidad de la arquitectura diseñada en base a los elementos de diseño utilizados.

• Ofrecer la posibilidad de soporte a los distintos métodos y técnicas de evaluación de arquitecturas de software.

Para efectos de la evaluación, resulta interesante conocer los distintos tipos de decisión que pueden ser tomados a nivel de diseño, puesto que un ambiente de evaluación y análisis debe estar en capacidad de soportar estos procesos (Kazman, 1996). A nivel arquitectónico, Bredemeyer et al. (2002) establecen que los tópicos de decisión con frecuencia son:

• Estilos y patrones arquitectónicos

• Arquitecturas de referencia

• Responsabilidades asociadas a los componentes

• Identificación de interconexiones entre componentes

• Comportamiento dinámico del sistema

• Interfaces entre componentes y sus responsabilidades

• Manejo de múltiples configuraciones para sistemas distribuidos o concurrentes

Este proceso de análisis y diseño de arquitecturas de software presenta sobrecargas de recolección, manejo y presentación de la información relevante a ellos. Esto resulta un impedimento sustancial para las organizaciones que quieren adoptar una actitud más madura a su práctica en el diseño de arquitecturas de software. Proveer una herramienta que soporte estas prácticas es un primer paso de ayuda a extender su adopción en la industria (Kazman, 1996).

Es importante que el ambiente esté en capacidad de asistir en el proceso de diseño para efecto de la toma de decisiones, independientemente de la metodología y en el nivel en que

se encuentre el proceso de desarrollo (Bredemeyer et al., 2002). Al añadirle la capacidad de apoyo a la evaluación del diseño, con el manejo de las técnicas y métodos existentes hasta el momento, la herramienta proveerá información acerca de los puntos de riesgo en el diseño que se está evaluando. Esto resulta de gran ayuda al arquitecto al momento de tomar decisiones que harán posible la satisfacción de los requerimientos de calidad por parte del diseño en cuestión.

Si bien es cierto que la calidad del sistema depende en gran parte de la implementación, también es cierto que gran parte de ella depende de la arquitectura. De aquí la importancia de la correspondencia entre el diseño y la implementación. Es por ello que el ambiente de análisis, diseño y evaluación de arquitecturas de software se propone como un medio que permitiría la satisfacción de los requerimientos de calidad del sistema establecidos por los involucrados en el desarrollo del mismo.

#### *3.1.1.3. Pertinencia de la arquitectura con respecto a las unidades aeronavales*

Lograr desarrollar una arquitectura que garantice estabilidad y viabilidad al proyecto se considera un punto de suma importancia entre las dos fases que componen el ciclo de vida de un proyecto, la fase de ingeniería (definición del producto y su solución) y la fase de desarrollo (construcción, integración, evaluación y entrega del producto). En la primera se toman las decisiones más importantes mientras que en la segunda se realizan los mayores gastos y esfuerzos (Mollineda, Arquitectura y ocio, 2005).

La arquitectura de software en primera instancia se debe contextualizar al entorno que se aplicará, con una sola acepción: procesos iterativos. Debido a que no existe una teoría establecida sobre cómo proceder, el diseño, implementación y evaluación de la arquitectura de un sistema complejo como el de definición de unidades. En América latina se han realizado proyectos enfocados al área militar, el entrenamiento de aprendices con escenarios simulados para evaluar el manejo de las unidades de guerra con las que cuenta el

Estado y las acciones en diversas situaciones. En Ecuador se adelantó un proyecto con el título "*Integración de Sistemas Tácticos Navales"* (Plúas Andrade, Rivera Cárdenas, & Cortez Alvarez) , donde se diseñan y desarrollan interfaces, es decir, un medio que permita la comunicación entre dos sistemas, los cuales se encuentran en las unidades tipo Corbeta Misilera de la Armada de Ecuador. Los sistemas tácticos involucrados en el presente trabajo son los siguientes: el Sistema de Guerra Electrónica y el Sistema de Control de Tiro, que desarrollan las funciones de vigilancia y defensa respectivamente, y el Indicador Panorámico Naval, que recibe la información proveniente de los otros sistemas y ordena las acciones a seguir.

La integración se basó en el funcionamiento de los dos sistemas descritos en el párrafo anterior, especialmente en su parte de comunicación e información, y se desarrolló el hardware y software apropiados para establecerla. Es un proyecto ambicioso que servirá como soporte de la arquitectura planteada, a causa de que involucra el comportamiento de una unidad como las Corbetas Misileras y todas las características fundamentales de la misma para idear y ejecutar una estrategia en una "Guerra Electrónica", cómo la denominan Andrade y Cárdenas en su investigación. El aspecto limitado que aparece en este artículo, es que el sistema aborda un número limitado de unidades y además sólo de la nación del Ecuador, además de que el sistema sólo evalúa la interacción unidad-unidad ignorando las condiciones donde se presente la guerra.

Indagando un poco más en los procesos militares, se encuentra el concepto de "*Electronic War"* (Guerra Electrónica - EW). El concepto moderno, considera la EW como una parte importante de la estrategia militar global, concentrada en la neutralización del sistema de mando y control enemigo (C3), manteniendo simultáneamente la capacidad operativa del sistema C3 propio (dialnet.unirioja.es/descarga/articulo/4769588.pdf, Pedro Luis Aldea Gracia). El autor Aldea Gracia describe de manera muy precisa todos los posibles escenarios, unidades y sistemas que interactúan en una EW, es importante resaltar del artículo que con el tiempo los simuladores de EW se han convertido en una herramienta que permite evaluar y ejecutar una EW en tiempo real; con la evaluación de estos

simuladores se puede establecer de forma más precisa el rol dentro de una estrategia de las unidades aeronavales.

Sin embargo el documento redactado por Aldea Gracia, resulta limitado para su análisis debido a que este es un artículo netamente investigativo y sólo se hace mención de sistemas que tiene relación con la EW y que además se emplean en la actualidad. El nivel de abstracción hacia este proyecto se ve limitada por los términos militares empleados.

Por último se tomó en cuenta el desarrollo de un Sistema de Información, en el que se han integrado técnicas de Ingeniería de Software tradicional e Ingeniería del Conocimiento, para conseguir información objetiva y completa del estado de alistamiento de las unidades navales pertenecientes a una empresa que opera una flota de buques auxiliares (Rancan, 2004). Esta investigación ofrece ventajas notorias hacia este proyecto ya que en ambas se tiene como objetivo obtener información precisa y completa de las unidades navales y áreas que integran las operaciones militares.

Existen grandes ventajas a la hora de indagar en el artículo propuesto por Rancan, ya que aquella falencias que estén presentes se pueden complementar con la información suministrada por La Escuela en los *Jane's Information Group*, que son documentos organizados donde se presenta información sobre las unidades y artefactos utilizados en las operaciones militares y aeroespaciales.

## 3.2. ESTADO DEL ARTE

### 3.2.1 Reseña histórica de la Arquitectura de Software

Aún no se ha definido por completo el término de AS (ARQUITECTURA DE SOFTWARE). Mary Shaw o David Garlan parafrasearon los comienzos de la especialidad

a principios de los 90, los mismos párrafos han sido reutilizados a lo largo en la literatura, sin mayor énfasis en el aporte al estado del arte de este tópico y de las fuentes referidas en la reseña primaria y con afán por ir al grano, que usualmente no es de carácter histórico, sino técnico. En esta investigación se ha optado, más bien, por darle gran valor a la literatura e inspeccionar las fuentes más de cerca, con el propósito de señalar las supervivencias y las evoluciones que han experimentado las ideas fundadoras en la AS, comprender con mayor claridad el contexto y entender por qué algunas ideas que surgieron hace cuatro décadas demoraron un cuarto de siglo en materializarse.

La arquitectura de software remonta sus antecedentes al menos hasta la década de 1960, su historia no ha tenido tanta continuidad como la del campo más amplio en el que se escribe, la ingeniería de software (Pfleeger, 2002). Después de las tempranas inspiraciones de Edsger Dijkstra, acerca de una estructuración correcta de los sistemas de software, aunque no la llama arquitectura como tal, durante la década de los setenta, David Parnas cita en sus trabajos sobre técnicas de modularización en decisiones de diseño y familias de programas los aportes de Dijkstra, fueron, sin duda, aportaciones esenciales y permanentes. Las decisiones "tempranas" de diseño, argüía Parnas, probablemente permanecerían invariantes en el desarrollo de la solución; estas ideas se convierten luego en lo que hoy se conocen como decisiones arquitectónicas. La AS quedó en estado de vida latente durante unos cuantos años, impulsado a lo que es actualmente, en la década de los ochentas, aparece el paradigma de la programación orientada a objetos. En esta década, Mary Shaw contribuye a la construcción de la literatura con la entrega de dos trabajos importantes, que retoman las abstracciones de alto nivel: "Técnicas de abstracción en lenguajes modernos de programación" y "Los sistemas de gran escala requieren de abstracciones de alto nivel", hacia la década de los noventa, comienza a incubarse de manera más clara la idea de que las aplicaciones tienen una estructura. El punto de partida o comienzo de una expansión explosiva para lo que hoy conocemos como arquitectura de software, se da con los manifiestos de Dewayne Perry y Alexander Wolf (Perry & Wolf, 1992), son los primeros que proponen un modelo para la arquitectura de software; el cual contempla a la arquitectura compuesta por tres partes: elementos, forma y razón. Los elementos variaban

entre procesamiento, datos o conexión; la forma se define de acuerdo a las propiedades de, y a las relaciones entre los elementos; la razón se contempla en términos de restricciones del sistema, que se derivan de los requerimientos del sistema.

Puede decirse que estos genios fundaron la disciplina, y su llamamiento fue respondido en primera instancia en la década de 1990, que fue sin duda la etapa de la consolidación y diseminación de la AS en una escala sin precedentes. Las contribuciones más importantes surgieron de los miembros de lo que podría llamarse la escuela estructuralista de Carnegie Mellon: David Garlan, Mary Shaw, Paul Clements, Robert Allen.

Se trata entonces de una práctica joven, de apenas unos doce años de trabajo continuo, que poco a poco ha ido evolucionando hasta llegar a temas como el surgimiento de los patrones, cristalizada en dos textos fundamentales, el de la Banda de los Cuatro en 1995 (Gamma, Helm, Johnson, & Vlissides, 1995) y la serie POSA desde 1996. Christopher Alexander, quien incidentalmente fue arquitecto de edificios; Desarrolló en varios estudios de la década de los setenta, temas de análisis del sentido de los planos, las formas, la edificación y la construcción, en busca de un modelo constructivo y humano de arquitectura, diseñada de forma que contemple las necesidades de los habitantes (Alexander, 1977). El arquitecto debe ser un agente del usuario.

A lo largo de una cadena de cambios, las ideas llegaron por fin a la informática una década más tarde. Si bien la idea de arquitectura implícita en el trabajo actual con patrones está más cerca de la implementación y el código, y aunque la reutilización de patrones guarda estrecha relación con la tradición del diseño concreto orientado a objetos, algunos arquitectos emanados de la escuela de Carnegie Mellon formalizaron un acercamiento con esa estrategia. Tanto en los patrones como en la arquitectura, la idea dominante es la de re-utilización.

Como quiera que sea, la AS de este período tuvo la capacidad de estandarización de la terminología, desarrolló la codificación de los estilos arquitectónicos y elaboró lenguajes de descripción de arquitectura (ADLs). También se consolidó la concepción de las vistas arquitectónicas, como lo son 4+1, TOGAF, RM/ODP, IEEE, entre otras.

En el siglo XXI, la AS aparece dominada por las nuevas tendencias empresariales cuyo objetivo es encontrar un punto de equilibrio entre el negocio y las TI basados en estrategias orientadas a líneas de productos. Todo lo que se ha hecho en ingeniería debe formularse de nuevo, integrando la AS en el conjunto. La producción de estas nuevas metodologías ha ido incrementando de manera exponencial, y una vez más tiene como epicentro el trabajo del Software Engineering Institute en Carnegie Mellon. La aparición de las metodologías basadas en arquitectura, junto con la popularización de las metodologías ágiles han causado un reordenamiento del campo de los métodos, hasta entonces dominados por las estrategias de diseño "de peso pesado". Después de la AS y de las tácticas radicales, las metodologías nunca volverán a ser las mismas. (Reynoso, 2004)

### 3.2.2 Simuladores: Recorrido Histórico

La aparición de los simuladores computarizados se dio en la segunda mitad del siglo pasado. El motor intelectual de su uso se le atribuye a John Dewey en su obra "Education and Experience" en donde argumentaba en contra del exceso de teoría. La primera simulación gerencial fue auspiciada por la American Management Association en 1957. Bass estimó en 1964 que existían más de 100 simulaciones. Graham y Gray publican una descripción en 1969 de 180 simuladores computarizados. Fue en ése mismo año 1969 cuando se publica la primera colección anotada de simuladores. Diez años más tarde aparecía la cuarta edición describiendo tres veces más simulaciones. La cuarta parte de las simulaciones listadas en ésa 4ta edición fueron completamente nuevas. Otro estudio fechado en 1973 por Zuckerman catalogó 215 simuladores. Al año siguiente en 1974,

Schriesham localizó 400 simuladores. Parte de este gran crecimiento fue el estándar de acreditación impuesto por la American Association of Collegiate Schools of Business (AACSB) al exigir que el plan de estudios de los MBA's debía concluir con un curso integrador de Estrategia y Política, un curso ideal para el uso de simuladores y en donde se ha concentrado su uso. (LABSAG)

Durante los años 80 las simulaciones crecieron especialmente en complejidad. Sin duda la más compleja fue la simulación usada en el Ejercicio Ace de la Organización del Atlántico Norte en 1989 en la que participaron tomando decisiones 3,000 comandantes durante once días seguidos. Hacia 1996, una encuesta dirigida por Anthony J. Faria, encontró en los Estados Unidos a 11,386 instructores universitarios usando simuladores en las universidades americanas, y a 7,808 empresas usando simuladores en la capacitación de su personal.

#### *3.2.2.1 Ámbito Internacional*

Se realizó una investigación utilizando referentes bibliográficos en diferentes bases de datos e información brindada por la Escuela Naval Almirante Padilla donde se encontraron a nivel internacional proyectos enfocados a la emulación de situaciones de guerra naval cómo Naval War: Arctic Circle, Warship Combat: Navies at War, los cuales se presentan a continuación:

**Naval War: Arctic Circle:** Este juego está desarrollado por la compañía noruega **Turbo Tape Games** y distribuido por la distribuidora sueca **Paradox**, intentando involucrar al usuario en lo que es una mezcla perfecta entre la simulación y la estrategia, su principal función se describe a continuación:

"Naval War Arctic Circle es un software RTS (Real Time Strategy – 'Estrategia en tiempo real') ambientado en la época de la Guerra Fría en general, y en particular, cuenta las confrontaciones entre **Rusia** y la **NATO** (Organización del Tratado del Atlántico Norte), en forma de batallas que se deben liderar escogiendo uno de los dos bandos.

Durante el modo campaña, se podrá optar por jugar con uno de los dos bandos, los Rusos o la NATO, las batallas se libraran en un vastísimo mapa, se permite escoger la velocidad del juego y variarla durante el mismo, porque la recreación de la velocidad es simplemente sublime, y si un avión debe tardar X horas en llegar a su destino, las tarda.

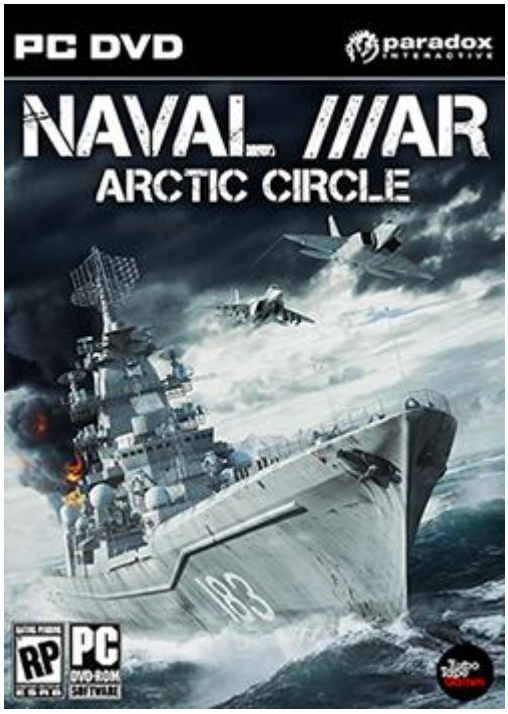


**Ilustración 1. Portada del software Naval War Artic Circle** (Turk, 2012)

Debido al difícil manejo de los controles internos del juego, se requiere obligatoriamente realizar un completo tutorial inicial que permite manejar las distintas flotas, desde buques y portaaviones hasta aviones y helicópteros, e incluso submarinos." (Hinajosa Devesa, 2012)


**Ilustración 2. Interfaz amigable del software Naval War Artic Circle** (Kempretan, 2013)

**Warship Combat : Navies at War:** Es un juego de guerra desarrollado por la compañía **Naval Warfare Simulations**, en el que se busca abarcar diferentes escenarios de combate dándole al usuario la habilidad de tomar mejores decisiones en el campo.

"Warships Combat: Navies at War (AKA WCNAW) es un juego de guerra muy detallado y preciso, muestra tácticas de superficie naval de combate, es basado en turnos. Permite a un equipo comandar una flota de hasta 32 buques de guerra frente a una IA (Máquina de Inteligencia Artificial) eficaz y desafiante. El motor es capaz de combatir cubriendo los eventos sucedidos entre 1890-1950. Hay más de 6 países y 400 clases de buques de guerra para elegir que van desde los destructores a los acorazados. Las características incluyen animaciones de combate, las diversas condiciones climáticas, controles de incendio, radares ( 2 ª Guerra Mundial ) , la selección de armas , las baterías principales , secundarias y terciarias de armas , torpedos, efectos completos de sonido estéreo , maniobras tácticas , interfaz intuitiva , gráficos detallados de la nave , cortinas de humo , bengalas , reflectores , daños en el sistema , el daño estructural y flotación , y algunas otras funciones más. Futuras

ampliaciones y mejoras en el motor agregarán más naves, armadas, la tierra, la capacidad de luchar las batallas más grandes, y otras características. El diseño de escenarios es muy simple y el sistema incluye una amplia gama de opciones de creación de escenario. Hay una interfaz de diseño de escenario integrado o usted puede editar fácilmente los archivos de texto del escenario.

Este juego de guerra naval sigue el concepto de " las apariencias engañan". Aunque WCNAW está diseñado para ser fácil de jugar, no es sólo una cuestión de la comparación de la artillería, la armadura, y otros factores. Las calificaciones y los factores indicados en las tablas de datos de buques son indicadores de lo que un buque de guerra es capaz de hacer en el combate, sin embargo la gran variedad de mecanismos detallados de combate, decisiones tácticas y cálculos son los que determinan la forma de un barco de guerra realmente en combate. En la sección de artillería y torpedos PK, existe una lista de los cálculos que dan una idea del grado de detalle de los sistemas mecánicos en cuanto a los términos de resolución de combate.

Como este juego de guerra está diseñado específicamente para jugar al solitario la IA es, obviamente, una característica crítica y la calidad y capacidad de la IA se toma muy en serio. La IA es muy competente, no mecánica, aportará una dura lucha, y utiliza una variedad de tácticas. La IA tiene un sistema incorporado en el análisis de amenazas y combate teniendo en cuenta las capacidades de los buques de guerra que participan en la batalla y decide sobre una variedad de tácticas a utilizar para las maniobras, la artillería y los ataques de torpedo. La IA intentará maximizar el efecto de sus ataques y maniobras y se analiza la situación de la batalla continua. En cada actualización publicada, sobre todo cuando se añaden nuevas características sobre el efecto de la mecánica de combate, la IA es re- calibrado para ayudar a asegurarse de que tiene las mismas opciones de táctica naval como el jugador por lo que ni el jugador o la IA, tienen una ventaja injusta sobre los otros. También hay una función de opción de juego que le permite establecer el nivel de agresividad de la IA.

El objetivo general de este diseño es para permitir la facilidad de juego a través de una interfaz intuitiva de pantalla única, con la utilización de un modelo detallado y preciso de combate naval que no sacrifica el realismo de la jugabilidad". (Dean & Miller, 2013)

### 3.2.3. EL DESARROLLO DE LOS JUEGOS DE GUERRA

Los Juegos de Guerra son de origen militar y se han utilizado para preparar a los líderes militares y hacer frente a circunstancias imprevistas en el combate. Los usaron los antiguos griegos y en 1811 los *prusianos* introdujeron los tableros de juego tridimensionales para añadirle realismo al juego**.**

En la Segunda Guerra Mundial el Almirante Nimitz, con los juegos de guerra, previó todas las batallas navales que se dieron en el Pacífico excepto la táctica japonesa de los kamikazes (Ayala Ruiz & Ramiro Arias, 2011). En los últimos años, con la ayuda de computadores y programas inteligentes, los juegos de guerra previeron la caída de la Unión Soviética, determinaron las opciones para el uso de la fuerza militar en la campaña "Tormenta del Desierto" en Irak y el desembarco en Haití. (Suarez Chacón, s.f)

El precipitado adelanto científico y tecnológico que el mundo entero vive y experimenta, obliga a sus Fuerzas Armadas a que informaticen sus actividades, para mejorar sus procesos, ganando tiempo y economizando recursos.

Para que esto ocurra es necesario mejorar y adaptar la infraestructura para la simulación de los juegos de guerra y así cumplir de una mejor manera con las necesidades actuales y futuras de las instituciones militares. Histórica y pragmáticamente hablando, hay razones sólidas para volver a enfocar y volver a definir el uso de esta herramienta de valor incalculable para poder planificar y ejecutar mejor la guerra.

No se debe argumentar si el juego de guerra simboliza la capacitación o el adiestramiento o si es operacional o analítico. Todas las doctrinas son útiles para promover temas que preparan a los soldados y a los planificadores a tomar buenas decisiones (ver Ilustración 3).

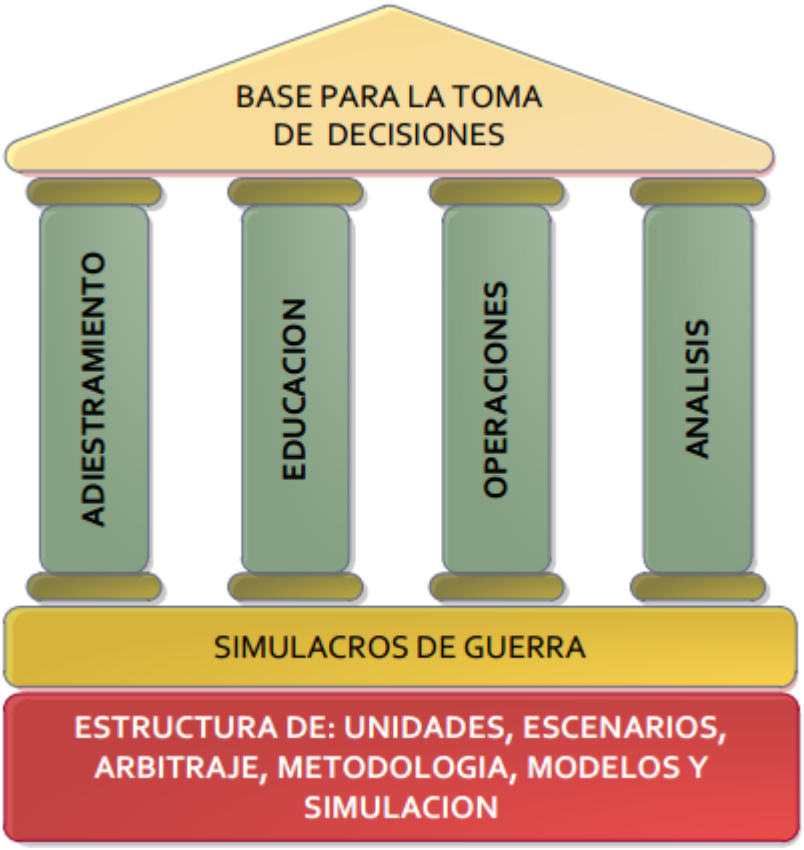


**Ilustración 3. Pilar para la toma de decisiones en los juegos de combate.** (Carrillo Andrade & Ortega Gutierrez, 2012)

Junto con sus herramientas de apoyo de diseño y simulación, el juego de guerra le enseña al personal a procesar los problemas de manera más eficaz al tomar buenas decisiones. En ese sentido, el uso profesional del juego de guerra puede abarcar un amplio espectro de tiempos, marcos hipotéticos y circunstancias.

Podemos encontrar muchos ejemplos históricos de los aportes de los juegos de guerra a las estrategias, operaciones o tácticas de éxito, y es tentador intentar comprobar el valor del

juego de guerra señalando la causalidad directa entre los juegos de guerra y el éxito en la guerra.

#### 3.2.3.1. Clasificación de los juegos de guerra

Actualmente se utilizan diversos tipos de simuladores para el entrenamiento de las unidades militares, algunos asistidos por computador, otros totalmente computarizados y aquellos que emplean sistemas de simulación.

1. *Los juegos de guerra asistidos por computador*, son juegos de mesa que simulan combates, batallas o enfrentamientos, ya sean: terrestres, navales, aéreos o de submarinos y se juegan sobre cartografía o sobre un tablero y los cálculos se los realiza mediante el computador en forma externa.

   Se crea una situación bélica con su respectiva cadena de mando para permitir a los comandantes manipular modelos a través de posibles escenarios en su planificación militar, simulando situaciones habituales de incertidumbre militar, unos mapas predeterminados para simular el escenario, se usan iconos para marcar las posiciones en la que se enfrentan dos equipos, gráficos con calcos para representar la operación y finalmente son dirigidos o arbitrados por un árbitro lo que genera confusiones a la hora de ejecutar las órdenes e imposibilidad de cumplirlas al cambiar las situaciones tácticas por la falta coherente de resultados.

2. *Los juegos de guerra computarizados* son aquellos que se juegan directamente sobre la pantalla del computador, mediante un modelo digital del terreno, modelos matemáticos y un software comercial diseñado y construido especialmente para el efecto.

   Es un sistema construido mediante una combinación de medios humanos (un equipo multidisciplinario), elementos técnicos (servidores, computadoras) un software

comercial para desarrollo de la interfaz del usuario, un software para la definición y tratamiento del escenario y una base de datos para almacenar la información, lo cual permite hacer viable el desarrollo de los juegos de guerra.

3. *Los juegos de guerra simulados* son los que tienen en esencia un motor de ejecución, contiene parámetros estructurados (Base de Datos Documental) fácil manipulación de ambientes 3D, desarrollado en base a los protocolos DIS y HLA, permite la reusabilidad de componentes de simulación.

   Son un campo de la tecnología muy relacionado con la inteligencia artificial, su arquitectura es dirigida por eventos, la información enviada está basada en los cambios de estado de las entidades y en su interacción, las aplicaciones se comunican a través de un mecanismo de distribución de datos llamado el RTI[2] y sus atributos, asociaciones, e interacciones son soportadas por una federación[3].

### 3.2.3.2. Los desarrollos actuales de software para juegos de guerra

Un juego de guerra es aquel que recrea un enfrentamiento armado de cualquier nivel (táctico, operacional, estratégico o global) con reglas que implementan cierta simulación de la tecnología, estrategia y organización militar usada en cualquier entorno histórico, hipotético o fantástico y que no implica en algún momento el uso de violencia física. Es una simulación de combate o acción bélica, ya sea como un juego de mesa, como un videojuego por consola o por computadora, o como una recreación real. (Informaciones, 2009)

---

[2] RTI:"Runtime Infrastructure" Proporciona un conjunto de servicios común a los federados, su función primaria es la distribución de datos, bajo HLA Los federados envían la información al RTI, el cuál transmite los datos a los federados apropiados.

[3] Federación: Colección de simulaciones de las que consta el ejercicio.

Los juegos de guerra se han desarrollado en muchos países para desempeñar un papel importante en el entrenamiento del personal militar, enfocándose en la capacitación y adiestramiento en el ámbito táctico de la guerra.

La tecnología de la Simulación en los centros de entrenamiento a nivel mundial, ha permitido el desarrollo de Simuladores de Juegos de Guerra cuya ventaja significativa es la disminución relativa de costos por entrenado, costos equipos y disminución de riesgos posibles que se presentan en una situación de entrenamiento real. A continuación veremos algunos ejemplos de países en Latinoamérica que han logrado desarrollo significativo en este campo:

**SETAC – Sistema de entrenamiento táctico computarizado**

**Ilustración 4. Logotipo del CEOTAC.** (Carrillo Andrade & Ortega Gutierrez, 2012)

Es un software computacional de tecnología abierta que permite la realización de Juegos de Guerra en la modalidad denominada doble acción, con dos elementos adversarios que se enfrentan, este fue diseñado el año 1993 en la Academia de Guerra del Ejército de Chile, a través del trabajo integrado de Oficiales de Estado Mayor, Ingenieros Politécnicos Militares e Ingenieros Civiles provenientes de las principales Universidades del país.

Este software desarrollado íntegramente en Chile permite que se enfrenten en tiempo real, dos hipotéticos adversarios sobre un mismo escenario geográfico. Los resultados del

enfrentamiento de las Unidades participantes son entregados por los modelos matemáticos en que se sustenta el sistema, fallos que están en directa relación, con las capacidades de los sistemas de armas que las unidades poseen y con las variables de terreno y tiempo atmosférico existentes al momento de producirse los enfrentamientos.

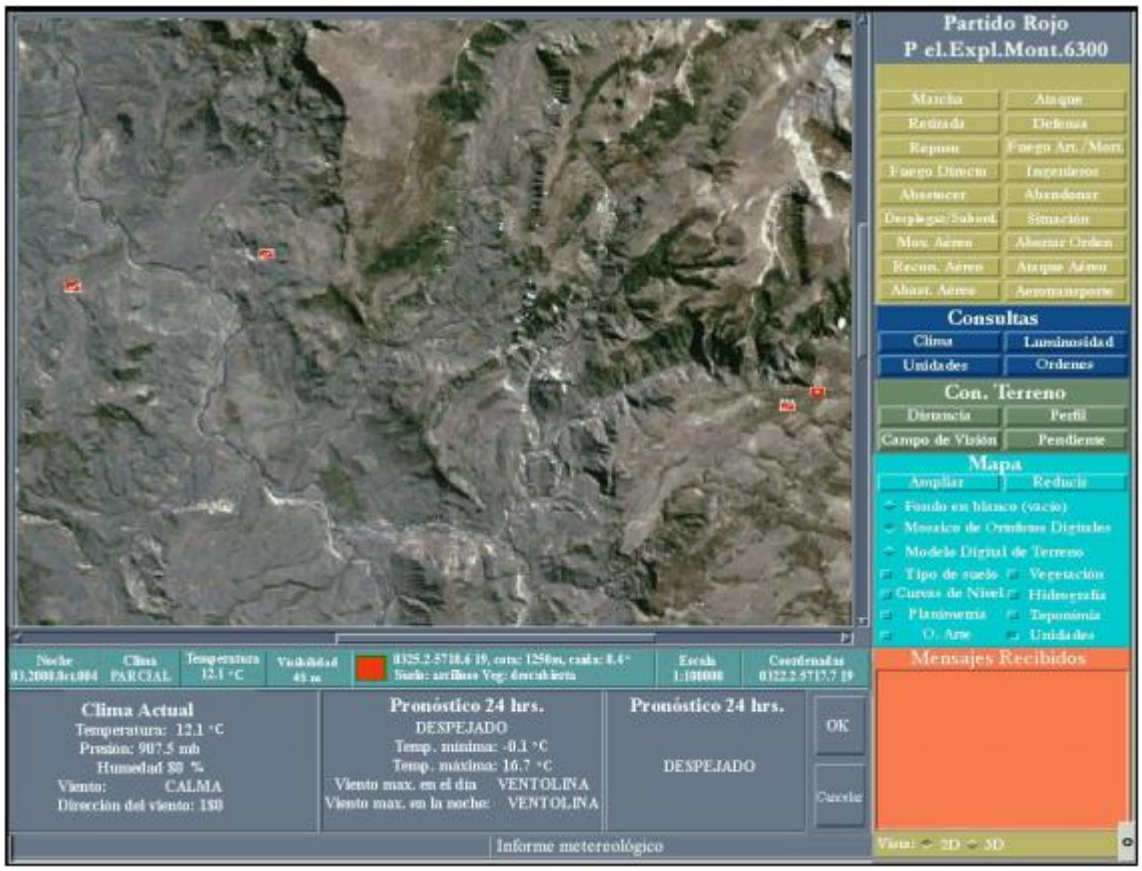


**Ilustración 5. Interfaz del Comandante software SETAC.** (Carrillo Andrade & Ortega Gutierrez, 2012)

El objetivo de este software, es entrenar a través de simulación a los Comandantes y Estados Mayores de nivel Brigada y División (en la versión utilizada por la Academia de Guerra y las Unidades Operativas del Ejército) y a los Comandantes de Batallón y sus Planas Mayores (en la versión en uso en las Escuelas de Infantería y de Caballería Blindada), en el proceso de toma de decisiones militares, planificación, ejecución y control de las acciones tácticas, bajo un ambiente de presión y apremio.

Posee un sistema de información geográfico propio, que permite manejar imágenes satelitales geo referenciadas en un mapa digital, aspecto que lo destaca dentro de otros sistemas de simulación de este tipo que existen en el mundo. Lo anterior, debido a que se realizó con éxito la incorporación de imágenes raster en su modalidad de ortofotos digitales e imágenes de satélites, permitiendo contar con un nivel adicional de información geográfica y actualizada del territorio nacional, posibilitando que los niveles de información vectorial, se aproximen a la realidad temporal, pudiendo incorporar los elementos más críticos del espacio geográfico.

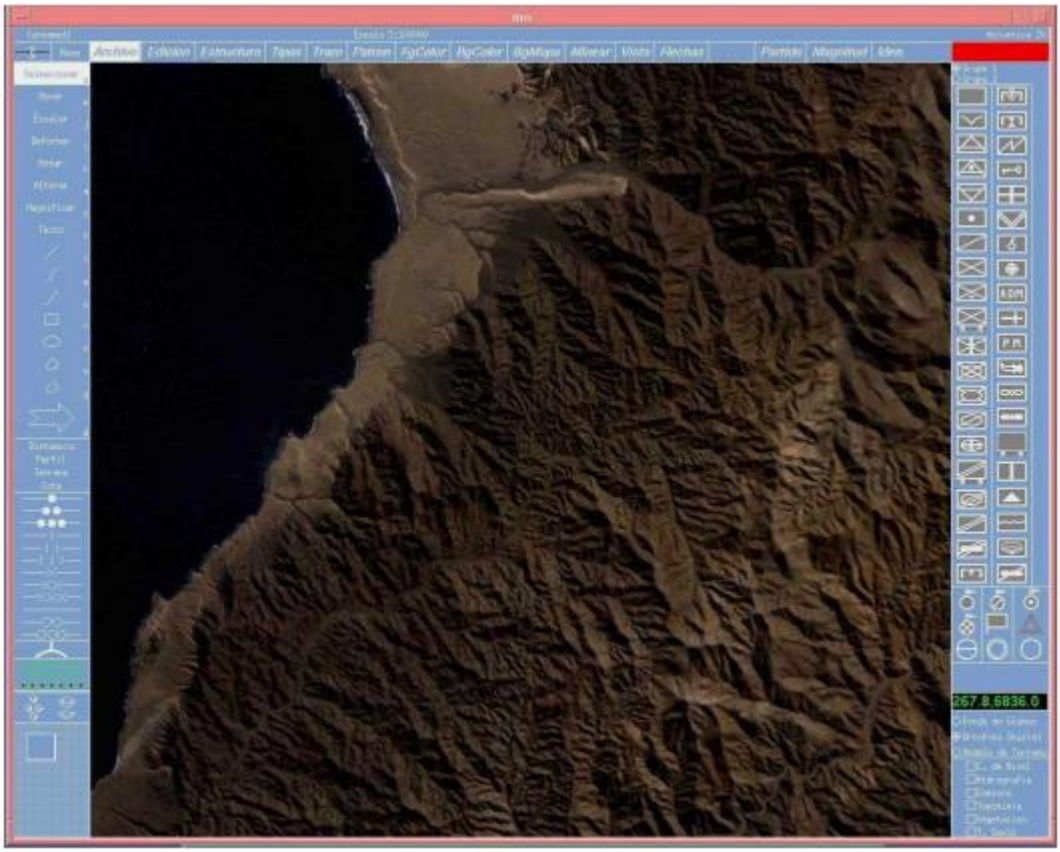

**Ilustración 6. Interfaz del usuario software SETAC.** (Carrillo Andrade & Ortega Gutierrez, 2012)

El impacto del uso de la simulación ha sido determinante para el entrenamiento de los Comandantes y sus Cuarteles Generales y para la utilización de los sistemas y procedimientos de mando y control, especialmente en lo relacionado con los procesos de

planificación e integración de las funciones de Personal, Inteligencia, Operaciones y Logística. (Carrillo Andrade & Ortega Gutierrez, 2012)

**Steel Beasts Professional**

Durante el año 2002, se desarrolló en el Regimiento Reforzado Nº 2 "Cazadores" un proyecto tendiente a incrementar la eficiencia de las tripulaciones de tanques Leopard 1- V, específicamente para los puestos de artillero y comandante, mediante la utilización de un juego computacional denominado "Steel Beasts", recurso didáctico que hasta la fecha se utiliza con buenos resultados.

Luego de algún tiempo se notó que este "simulador" era eficaz no solo con los tripulantes de vehículos blindados, sino que también podía ser utilizado en el ámbito "táctico", entrenando comandantes de unidades pequeñas, específicamente de los niveles de Pelotón y Escuadrón, ya que cada acción dentro del "mundo virtual" debía ser planificada y ejecutada tanto técnica, como tácticamente. (Carrillo Andrade & Ortega Gutierrez, 2012)

Las características más relevantes de este software son:

- Permite la ejecución de ejercicios demostrativos, capaces de ilustrar a los alumnos el desarrollo de acciones tácticas fundamentales.

- Tiene la capacidad para generar ejercicios de doble acción, fundamentales en el entrenamiento de comandantes, ya que se enfrentan ante adversarios con inteligencia y voluntad.

- Utiliza el sistema de graficación y la nomenclatura OTAN

- Incluye menús y textos en español, lo que facilita la comprensión por parte de todos los usuarios.

- Es posible importar terrenos reales y transformarlos en virtuales, los que son desplegados en dos y tres dimensiones.

- Tiene la capacidad para otorgar distintos niveles de operacionalidad a cada unidad, diferenciando calibres y cantidades y tipos de munición.

- Una característica destacable es el sistema de intercambio de información entre los usuarios registrados. Lo anterior, se materializa mediante un sistema de "foro en línea", lo que permite conocer avances y generar contactos con otros ejércitos y academias. Esta agrupación es denominada "comunidad".

- Por la condición de software "licenciado", no es posible acceder a los códigos de programación; sin embargo, es relevante el sistema de "actualizaciones colaborativas", en donde cada nuevo desarrollo del programa, financiado por un usuario específico, es puesto a disposición de la "comunidad".

**La función del software para los juegos de guerra**

Bajo la estructura de los diseños actuales de los juegos de guerra, los participantes a menudo invierten la mayor parte de su tiempo decidiendo cómo desplazar, emplear y sostener a las tropas, estos no son los objetivos principales del juego de guerra, pero definitivamente forman parte de su asignación de tareas.

Los juegos de guerra se llevan a cabo para desarrollar un mejor entendimiento de unos cuantos asuntos claves. Librar a los participantes en los Juegos de temas que tienen que ver con el desplazamiento y el empleo de las fuerzas lo que les permite concentrarse en temas de política decisivos.

La función principal del juego de guerra radica en la exploración del arte operacional para elaborar ideas en cuanto a cuestiones y conceptos de las operaciones actuales y futuras, *no radica en intentar predecir o probar resultados*, como para justificaciones presupuestarias o de estructura de fuerza.

La experiencia muestra que los juegos pueden revelar campos fructíferos para más tarde analizarlos a fondo. Los problemas que surgen de los juegos de guerra pueden servir como un mapa para enfocar las iniciativas de los laboratorios de batalla, los centros de guerra, los proyectos de experimentaciones conjuntas y las iniciativas de las fuerzas expedicionarias. Este mapa fomentaría una iniciativa de juego de guerra coherente, eficaz y económica. De la misma manera que los aviadores han aprendido las maneras más eficaces de aplicar el poder aeroespacial, nosotros también debemos explorar los usos más productivos del juego de guerra.

### 3.2.3.3. Ámbito Nacional

También se realizó una investigación a nivel nacional, de la cual se tomaron referentes bibliográficos de la ENAP de Colombia y eventos relacionados a los juegos de guerra en América. Los eventos relacionados a los juegos de guerra donde interviene la nación colombiana se muestran a continuación:

**Juego de Guerra Naval Interamericano 2012:** Colombia fue la anfitriona de la trigésima octava versión del Juego Naval Interamericano IAWG-2012, organizado por la Escuela Superior de Guerra para los 15 países integrantes de la Conferencia de las Escuelas Navales de las Américas.

Este encuentro académico, que se realiza desde 1972 para fomentar el trabajo de las fuerzas multinacionales en la neutralización de las amenazas potenciales en las fronteras marítimas, se desarrolló en tres fases proyectadas desde el 9 de abril hasta el 28 de septiembre de 2012.

En la primera etapa, los representantes invitados de Brasil, Chile, Ecuador, Estados Unidos, México, Perú y República Dominicana, serán partícipes presenciales del planeamiento adelantado por la Escuela Superior de Guerra de Colombia, a través del Departamento de Armada, como líder del ejercicio.

En la segunda fase, con las representaciones de las escuelas de guerra navales participantes, se afrontaron escenarios de amenazas simuladas y, en tiempo real, se coordinaron acciones conjuntas para contrarrestar los diferentes riesgos planteados para medir la capacidad de reacción y la calidad de las decisiones tomadas.

Durante esta fase de ejecución, el soporte tecnológico dispuesto por la unidad anfitriona, mediante su Centro de Simulación y Análisis de Crisis, desempeñaron un papel fundamental. Una plataforma virtual facilitó la comunicación entre los países participantes, desde sus respectivos territorios, e hizo posible recrear desastres naturales, por primera vez en la historia de este evento.

A lo largo del Juego Naval Interamericano se analizaron temas como la aplicabilidad de las leyes internacionales y propias al momento de actuar frente a la piratería, el terrorismo, el tráfico de drogas y el efecto de los desastres naturales en estos escenarios, además de las operaciones de mantenimiento de paz, asistencia humanitaria y contaminación por derrames de combustible.

La fase final del evento académico, tuvo lugar del 24 al 28 de septiembre en Cartagena de Indias, ciudad donde se dieron cita los directores de las escuelas navales participantes, para analizar las experiencias adquiridas en el desarrollo del ejercicio y revisar los puntos de acuerdo mutuo. **(Colombia, 2012)**

En la Escuela Naval Almirante Padilla actualmente se realizan las simulaciones de juegos de guerra para el entrenamiento de sus cadetes en el software Harpoon desarrollado por AGSI y distribuido por Matrix Games. A continuación se describirá más detalladamente el aplicativo:

**Harpoon: E**s una serie de simuladores no clasificados que se encuentra entre los más precisos del mundo de aire moderno y la guerra naval. Los simuladores permiten controlar todos los elementos de una fuerza  naval moderna y explorar las complejidades de combate

del siglo 21. La serie Harpoon, de hecho, se considera que está lo suficientemente cerca de la realidad por ello ha estado en uso durante años por diversas ramas militares de todo el mundo en su formación y como la herramienta de simulación. Se requiere pertenecer a un grupo de la marina de un país para tener una mejor experiencia de combate y sacar mejores conclusiones sobre ella.

**Harpoon Classic** fue el primer juego Harpoon para ordenador. Aunque carece de la sofisticación de los nuevos simuladores Harpoon II y Harpoon 3, era probablemente el mejor juego naval en el mundo desde que se dejó de usar Three-Sixty en diciembre de 1989. Alianza Interactiva licenció el código y los derechos de ACSI a mediados de 1994 y desarrolló Harpoon Classic para Windows 3.1. En 1996, el código licenciado de Harpoon Classic '97 paso a Interactive magic, que desarrolló el juego aún más a Windows 95.

**Harpoon II** fue el más complejo, más realista y más preciso simulador de estrategia de las operaciones aéreas y navales a disposición de los usuarios que no son militares en su tiempo.

**Harpoon 3** es la versión de Harpoon II de Windows mejorado enormemente (Windows 95, 98, ME, NT 4.0, Windows 2000 y Windows XP) y Mac (OS y OSX). Harpoon 3 para PC fue lanzado a mediados de febrero de 2002 cuando Harpoon 3 para Macintosh fue lanzado en 2001. Es el más complejo, más realista y más preciso simulador de estrategia en las operaciones aéreas y navales a disposición de los usuarios no militares en la actualidad. El simulador está siendo apoyado activamente y mantenido por el promotor y la comunidad, y las nuevas mejoras, características y escenarios se lanzan sobre una base regular. El simulador contiene una gran cantidad de nuevas características y correcciones de errores demasiado numerosos para mencionarlos todos. (AGSI, 2007)

## 3.3. MARCO TEÓRICO

Actualmente es cada vez más común escuchar términos asociados a las buenas prácticas del desarrollo de software y al desarrollo de software, tener una arquitectura de software es importante para el desarrollo exitoso de un sistema de software.

Los sistemas de software son construidos para satisfacer los objetivos de negocio de las organizaciones. La arquitectura es un puente entre esos (a menudo abstractos) objetivos del negocio y el sistema final resultante (artefacto concreto). Mientras la ruta desde los objetivos abstractos hasta los sistemas concretos puede ser compleja, la buena noticia es que las arquitecturas de software pueden ser diseñadas, analizadas, documentadas e implementadas usando técnicas conocidas que darán soporte al cumplimiento de los objetivos y la misión del negocio. (Len Bass, 2012)

### 3.3.1. Arquitectura de software

Hay muchas definiciones de arquitectura de software, fácilmente descubribles con una búsqueda en Internet, pero la que reúne gran cantidad de bondades de las demás definiciones, es la siguiente: La arquitectura de software de un sistema, es un conjunto de estructuras necesarias para razonar acerca del sistema, la cual comprende los elementos de software, relaciones entre ellos y las propiedades de ambos. (Len Bass, 2012) Basado en esto, el término de arquitectura de software hace referencia a la estructuración del sistema que, idealmente, se debe crear en etapas tempranas del desarrollo del software. Esta estructuración no es más que un diseño de alto nivel del sistema que busca satisfacer los atributos de calidad (escalabilidad, confiabilidad, mantenibilidad, etc.), y servir como guía en el desarrollo, así, enfocándolo aun proyecto de desarrollo de software, los programadores, diseñadores, ingenieros y analistas pueden trabajar bajo una línea común (Arquitectura de software) que les posibilite la compatibilidad necesaria para lograr el objetivo deseado. (Guglielmetti, 2012). Al igual que en cualquier proyecto donde se desee construir un producto final, las decisiones críticas relativas al diseño general deben hacerse

en la etapa más temprana del desarrollo. El no crear este diseño desde etapas tempranas del desarrollo puede comprometer severamente que el producto final satisfaga las necesidades del cliente. Además, el costo de las correcciones relacionadas con problemas en la arquitectura es muy elevado. Es así que la arquitectura de software juega un papel fundamental dentro del desarrollo.

**Patrones de arquitectura:** Los patrones de arquitectura están orientados a representar los diferentes elementos que componen una solución de software y las relaciones entre ellos. A diferencia de los patrones de diseño de software que están orientados a objetos y clases (patrones creacionales, estructurales, de comportamiento, de interacción, etc.), los patrones de arquitectura están a un mayor nivel de abstracción. (Wilson, 2011)

**Diseño de alto nivel:** En un diseño de alto nivel se describen los componentes principales del sistema y el modo en que interactúan entre sí para lograr los objetivos del diseño. En el desarrollo del diseño de alto nivel, están implicadas las actividades de la lista siguiente, aunque no necesariamente en un orden determinado. (Microsoft Developer Network)

### 3.3.2 Patrón modelo vista controlador

*MVC* (por sus siglas en inglés) es un patrón de diseño de arquitectura de software usado principalmente en aplicaciones que manejan gran cantidad de datos y transacciones complejas donde se requiere una mejor separación de conceptos para que el desarrollo esté estructurado de una mejor manera, facilitando la programación en diferentes capas de manera paralela e independiente (Rivera López, 2008). *MVC* sugiere la separación del software en 3 estratos: Modelo, Vista y Controlador, los cuales serán explicados en breve:

**Modelo:** Es la representación de la información que maneja la aplicación. El modelo en sí son los datos puros que puestos en contexto del sistema proveen de información al usuario o a la aplicación misma.

**Vista:** Es la representación del modelo en forma gráfica disponible para la interacción con el usuario.

**Controlador:** Es la capa encargada de manejar y responder las solicitudes del usuario, procesando la información necesaria y modificando el Modelo en caso de ser necesario.

#### 3.3.2.1 MÉTODO SAAM: SOFTWARE ARCHITECTURE ANALYSIS METHOD

Existen varias técnicas para evaluar arquitecturas de software, que se clasifican en cualitativas y cuantitativas. Dentro de las técnicas de evaluación cualitativas se pueden utilizar: escenarios, cuestionarios o listas de verificación. Por otro lado, en las técnicas de evaluación cuantitativas se pueden emplear: métricas, simulaciones, prototipos, experimentos o modelos matemáticos. La mayoría de los métodos de evaluación utilizan escenarios, que son secuencias específicas de pasos que involucran el uso o la modificación del sistema. Por lo regular, las 146 técnicas de evaluación cualitativas son usadas cuando la arquitectura se encuentra en construcción, mientras que las técnicas de evaluación cuantitativas, se usan cuando la arquitectura ya ha sido implantada. (Carrasco Puebla, 2014)

Luego de responder en el Marco de Referencia del presente trabajo de investigación a interrogantes como ¿Qué importancia tiene el análisis de la arquitectura?, ¿Qué implica analizar una arquitectura?, ¿Qué técnicas de análisis existen?, ¿Cómo compararlas?, ¿Qué herramientas para hacer análisis de arquitectura existen?, ¿Qué tipos de análisis se pueden hacer?, entre otras preguntas importantes, se evidenció claramente que el método utilizado para analizar una arquitectura de tipo software es el SAAM debido a que este al igual que el ATAM, y el ARID son de los métodos de evaluación de arquitectura de software, más usados pero se diferencia del método ARID debido a que combina prácticas de la ATAM y la SAAM y la ATAM es más utilizado a nivel de arquitecturas empresariales o arquitecturas con poco o ausencia de código.

Se sometió a este análisis la arquitectura que se propone con el fin de adquirir mayor validez científica.

El método fue originalmente creado para el análisis de la modificabilidad de una arquitectura, pero en la práctica ha demostrado ser muy útil para evaluar de forma rápida distintos atributos de calidad, tales como modificabilidad, portabilidad, escalabilidad e integralidad. El método de evaluación SAAM se enfoca en la enumeración de un conjunto de escenarios que representan los cambios probables a los que estará sometido el sistema en el futuro (Kazman R. C.-W.-0.). Como entrada principal, es necesaria alguna forma de descripción de la arquitectura a ser evaluada. Las salidas de la evaluación del método SAAM son las siguientes:

Una proyección sobre la arquitectura de los escenarios que representan los cambios posibles ante los que puede estar expuesto el sistema.

Entendimiento de la funcionalidad del sistema, e incluso una comparación de múltiples arquitecturas con respecto al nivel de funcionalidad que cada una soporta sin modificación.

Con la aplicación de este método, si el objetivo de la evaluación es una sola arquitectura, se obtienen los lugares en los que la misma puede fallar, en términos de los requerimientos de modificabilidad. Para el caso en el que se cuenta con varias arquitecturas candidatas, el método produce una escala relativa que permite observar qué opción satisface mejor los requerimientos de calidad con la menor cantidad de modificaciones. (Kazman R. C.-W.-0.)

| Pasos | Descripción |
|---|---|
| Desarrollo de escenarios. | Un escenario es una breve descripción de usos anticipados o deseados del sistema. De igual forma, estos pueden incluir cambios a los que puede estar expuesto el sistema en el futuro. |
| Descripción de la arquitectura. | La arquitectura (o las candidatas) debe ser descrita haciendo uso de alguna notación arquitectónica que sea común a todas las partes involucradas en el análisis. Deben incluirse los componentes de datos y conexiones relevantes, así como la descripción del comportamiento general del sistema. El desarrollo de escenarios y la descripción de la arquitectura son usualmente llevados a cabo de forma intercalada, o a través de varias iteraciones. |
| Clasificación y asignación de prioridad de los escenarios. | La clasificación de los escenarios puede hacerse en dos clases: *directos e indirectos*.<br>Un *escenario directo* es el que puede satisfacerse sin la necesidad de modificaciones en la arquitectura. Un escenario indirecto es aquel que requiere modificaciones en la arquitectura para poder satisfacerse. Los *escenarios indirectos* son de especial interés para SAAM, pues son los que permiten medir el grado en el que una arquitectura puede ajustarse a los cambios de evolución que son importantes para los involucrados en el desarrollo. |
| Evaluación individual de los escenarios indirectos. | Para cada *escenario indirecto*, se listan los cambios necesarios sobre la arquitectura, y se calcula su costo. Una modificación sobre la arquitectura significa que debe introducirse un nuevo componente o conector, o que alguno de los existentes requiere cambios en su especificación. |
| Evaluación de la interacción entre escenarios. | Cuando dos o más *escenarios indirectos* proponen cambios sobre un mismo componente, se dice que interactúan sobre ese componente. Es necesario evaluar este hecho, puesto que la interacción de componentes semánticamente no relacionados revela que los componentes de la arquitectura efectúan funciones semánticamente distintas. De forma similar, puede verificarse si la arquitectura se encuentra documentada a un nivel correcto de descomposición estructural. |
| Creación de la evaluación global. | Debe asignársele un peso a cada escenario, en términos de su importancia relativa al éxito del sistema. Esta asignación de peso suele hacerse con base en las metas del negocio que cada escenario soporta. En el caso de la evaluación de múltiples arquitecturas, la asignación de pesos puede ser utilizada para la determinación de una escala general. |

**Tabla 2. Pasos contemplados en el método SAAM (Adison-Wesley, 2005)**

El método SAAM tiene como característica principal la realización de un análisis que delimita la forma en que variarán los atributos de calidad, como resultado de algunas modificaciones futuras de la arquitectura. (Kazman R. C.-W.-0.) Este elemento es fundamental pues, le da una visión arquitectónica al equipo de desarrollo, que conocen hasta qué punto puede variar la arquitectura sin que afecte el nivel requerido de los atributos de calidad. Sin embargo, el comportamiento de un atributo de calidad puede afectar el desempeño de otros, por lo que no solamente se debe tener en cuenta la estructura de los componentes, sino también las relaciones que se establecen entre los mismos. Y es ahí donde este método, presenta su principal desventaja, ya que no valora la interrelación entre los distintos atributos. (Kazman R. C.-W.-0.)

#### 3.3.2.1.1 Analizando el diseño de arquitecturas propuesto para las unidades militares bajo el método SAAM

**Paso 1: Desarrollo de escenarios**

| Escenario | Definición |
|---|---|
| **1** | Cambiar el aspecto visual de las unidades de guerra. |
| **2** | Cambio de versión del algoritmo de simulación. |
| **3** | Agregar nueva funcionalidad. |
| **4** | Permitir diferentes visualizaciones al usuario del resultado de la simulación (enfrentamiento de unidades). |

**Tabla 3. Descripción de escenarios de la arquitectura definidos por los autores, según el método SAAM (Adison-Wesley, 2005)**

El prototipo que simula un enfrentamiento entre 2 unidades militares en contexto también debe ser eficiente en espacio. Sus componentes deben servir como entidades reutilizables. Debe ser capaz de apoyar los siguientes cambios en el futuro:

• Los cambios en el algoritmo de procesamiento: Por ejemplo, la manera en como la interfaz recibe información arrojada del escenario de simulación con el fin de mejorar rendimiento o algún otro atributo de calidad.

• Cambio en la representación de datos: por ejemplo, rendimiento de las armas, porcentaje de vida, precisión, etc. Se pueden almacenar en varias formas.

• Mejora de la función del sistema: Por ejemplo, modificar el sistema de vencimiento al enemigo, dándole más información al usuario acerca de toda la batalla librada y permitirle visualizarlo a manera de reporte.

**Paso 2: Descripción de la arquitectura candidata**

La presente es una arquitectura capaz de proveer una definición para las unidades militares navales, y que a partir de dicha definición se permita estudiar la constitución y la labor que cada una delas unidades cumplen en el entrenamiento táctico militar que realiza la Escuela Naval de Cadetes "Almirante Padilla". Para entrar en detalle sobre los las estructuras de uso, de importación y de flujo de datos y control se puede ir a la página distinguida por el nombre DESARROLLO DE LA ARQUITECTURA, donde se especifica cada característica relevante de esta arquitectura, y donde también se describen agrupaciones de componentes en subsistemas o capas.

**Paso 3: Clasificación de escenarios**

- Todos los escenarios son indirectos con respecto a la arquitectura candidata.

**Paso 4: Evaluación de escenarios**

| **Escenario** | | | **Modificación** | |
|---|---|---|---|---|
| **No** | **Descripción** | **Tipo** | **Componente** | **Cambio** |
| 1 | Cambiar forma de las Unidades | Indirecto | Interna | Modificar el aspecto visual de las Unidades Militares en combate. |
| 2 | Cambio / Mejora del algoritmo de | Indirecto | interna | Crear hilos para el procesamiento de los atributos de |

| | | | | |
|---|---|---|---|---|
| | procesamiento | | | clase de cada Unidad. |
| 3 | Nueva funcionalidad | Indirecto | Interna | Agregar código con el fin de cumplir un requerimiento funcional. |
| 4 | Diferente visualización de los resultados de la batalla | Indirecto | Interna | Modificar la implementación de acuerdo a la nueva representación de datos. |

**Tabla 4. Descripción de interacción de escenarios de la arquitectura definidos por los autores, según el método SAAM (Adison-Wesley, 2005)**

**Paso 5: Interacción de escenarios**

- Cambiar el aspecto visual de las unidades de guerra. Este escenario está ligado notoriamente al componente de la vista del aplicativo o lo que se pretende hacer visual en la arquitectura. Véase Diagrama de Componentes en el desarrollo de la arquitectura, a fin de cuentas lo que se pretende con este escenario es darle un equilibrio a la brecha que existe actualmente cuando se evalúa la satisfacción del usuario final con respecto a sus ambiciones para con el simulador resultante de esta arquitectura. En resumen se busca proporcionar el nivel de satisfacción entre los usuarios nuevos y los ya experimentados.

- Los escenarios para realizar cambios de versión del algoritmo de simulación y agregar nuevas funcionalidades al simulador que válida la arquitectura interactúan de una manera directa sobre a los componentes capaces de controlar las vistas del aplicativo y los controladores del mismo. Se debe tener en cuenta que estos escenarios están ligados a continuas mejoras del aplicativo por lo cual es impreciso determinar resultados veraces

respecto a su relación con el atributo de calidad que se evalúa, sin embargo es contundente decir que el modelo arquitectónico no carece de cohesión en sus componentes ya que los cambios en estos, no afectan directa o indirectamente a otros que interactúan en el sistema.

- El escenarios para visualizar resultados es una adición obligatoria al componente encargado de manejar los controladores del sistema, permitiéndole a este manejar resultados verídicos de los enfrentamientos que permitan aclarar las dudas y decisiones del usuario final.

**Paso 6: Interacción de escenarios**

En este ítem de la evaluación se asigna un peso a cada escenario en términos de su influencia para que el sistema sea exitoso. Este peso fue elegido de acuerdo a los objetivos del proyecto, costos, riesgos, etc.

En este proceso participaron todos los interesados (investigadores y director de proyecto).

| **Escenario** | | |
|---|---|---|
| **No.** | **Descripción** | **Peso** |
| **1** | Cambiar forma de las Unidades | **10%** |
| **2** | Cambio / Mejora del algoritmo de procesamiento | **35%** |
| **3** | Nueva funcionalidad | **30%** |
| **4** | Diferente | **25%** |

| | | |
|---|---|---|
| | visualización de los resultados de la batalla | |

**Tabla 5. Evaluación de la interacción de escenarios de la arquitectura definidos por los autores, según el método SAAM (Adison-Wesley, 2005)**

Conclusión: Con respecto a la modificabilidad, se puede decir que, los escenarios correspondientes a futuros cambios pueden ser evaluados en la arquitectura, logrando estudiar e identificar las áreas potencialmente complejas. El esfuerzo y costo de los cambios mencionados pueden ser estimados con anticipación.

### 3.3.3. Evolución de la simulación

La simulación ha evolucionado a través de siete categorías distintas, estas categorías nos proveen de una sólida experiencia para entender el dominio completo de la simulación.


**Ilustración 7. Categorías de la evolución de la simulación.** (Carrillo Andrade & Ortega Gutierrez, 2012)

**Ensayo en vivo:** Esto es, personas que se entrenan antes de formar parte del evento real. Se habla así de los "zafarranchos o simulacros". Un marino o soldado, se prepara a través de diferentes ejercicios de pensamiento, habilidad y destreza, para que llegado el día de un combate real, tenga la capacidad para enfrentar al enemigo, sin temor y con mayor decisión. (Uparella, 2007)

Un bombero puede y debe cumplir esta tarea. Pero, ¿será que podemos incluir a un ciclista, un patinador, un nadador o cualquier deportista, que cumple con esta norma para poder obtener una medalla?

Teniendo en cuenta el concepto militar de esta categoría, de ella se desprenden dos, las cuales hacen parte del Entrenamiento Asistido por Computador, como son los Juegos de Guerra y la Realidad Virtual.35

**Juegos de Guerra:** Evento en el cual la destreza en la toma de decisiones y el uso de recursos, manejo de personal y control de la logística, son lo indispensable para el entrenamiento. Sólo las entidades militares hacen uso de este entrenamiento. Está basado en la Investigación de Operaciones. (Carrillo Andrade & Ortega Gutierrez, 2012)

**Realidad Virtual:** Es la inmersión de un grupo de trabajo, personas, equipos o individuos, en un mundo generado por computador para estimular sus conductas reactivas o para estimular su aptitud para hacer o desarrollar algo. (Uparella, 2007)

**Análisis de Sistemas:** Se utiliza para predecir el futuro o para realizar pruebas sobre las capacidades de un sistema disponible.

**Simulación de Eventos Discretos:** Técnica para estructurar el mundo como una serie de eventos secuenciales que determinen el estado de ese mundo.

**Entretenimiento:** Es la aplicación de métodos científicos en juegos de simulación, video–juegos y otros productos de entretenimiento masivo. (Uparella, 2007)

**Interoperabilidad:** Tecnología para enlazar múltiples simulaciones con el fin de que se pueda interactuar en un mundo virtual compartido. Un juego de guerra se puede enlazar con

un simulador virtual local táctico proporcionando mayor realismo y capacidad de entrenamiento. (Uparella, 2007)

Las simulaciones han evolucionado en todos los frentes, como resultado, ha sido difícil establecer sus categorías o compararlas. Sin embargo, el Consejo para la Ciencia de la Defensa (Defense Science 36Board), presentó la mejor categorización de la simulación para la defensa, facilitando la comunicación con la comunidad. Ninguna simulación encaja perfectamente en una categoría, pero cada simulación es dominada por las características de una de ellas. El entrenamiento simulado típicamente viene en cuatro categorías:

**Simulación "En Vivo" - de "Vida" (Instrumentadas):** es cuando las personas reales usan equipo simulado en el mundo real, lo que permite que los alumnos practiquen las actividades físicas de la guerra con su equipo verdadero, el uso de ambientes de combate reales así como el empleo de fuerzas amigas y enemigas al mismo tiempo.

"El objetivo es realizar simulacros de combate en un medio no letal". Ejemplo de este nivel de simulación es el Sistema MILES americano en el cual los soldados usan un sistema láser en el fusil para simular los disparos y un receptor en el cinturón, casco y suspender para conectar el equipo con el láser a fin de registrar el impacto del láser en el soldado.

**Simulación "Virtual" (Tripuladas):** es cuando las personas reales usan equipo simulado en mundos virtuales. Son sistemas de simulación que proveen entrenamiento colectivo de personal, práctica de tiro y artillería, y entrenamiento para procedimientos especiales Se pueden reconfigurar para la simulación de varios sistemas y armamentos (T-72, T-62, T-55, BVP, BMP, M1, helicópteros, vehículos con ruedas, etc.). Son compatibles con los protocolos de comunicación DIS (simulación interactiva 37distribuida), y HLA (arquitectura de alto nivel) que se utilizan en simulaciones de combate y ejercicios combinados.

Son de alta fidelidad y de bajo costo lo que permite lograr las metas de entrenamiento ya fijadas, combinan el uso de equipos reales con escenarios proyectados en el computador o por medio de un casco. Se utilizan los gráficos de computador para estimular a los soldados que operan equipos de combate, sumergiéndolos en un mundo virtual donde las acciones

físicas tales como conducir o disparar un arma, tienen una directriz visible en el mundo sintético en el que se encuentra. “Su objetivo es el entrenamiento del soldado a través de su inmersión en un mundo virtual”. Ejemplo de este nivel es el simulador de movimiento completo, denominado full motion simulator flight, el cual duplica todos los aspectos de una aeronave y de su entorno, incluyendo movimientos básicos de la aeronave. Este tipo de simuladores pueden generar movimientos de modo que los ocupantes sientan un nivel de realismo tal como pasaría en una aeronave real, engañando a las tripulaciones y haciéndoles creer que estos se encuentran volando. Para poder realizar esto se combina una serie de aspectos tecnológicos que estimulan el sistema visual y vestibular de los pilotos. Lo que convierte a la simulación de vuelo en un área de conocimientos intensivos.

**Simulación "Constructiva",** es cuando personas simuladas, usan equipo simulado, en ambientes simulados, se conocen extensamente como “Juegos de Guerra” (Wargames). Las decisiones tácticas y estratégicas se reflejan en el movimiento de iconos militares (NTDS – Naval Tactical Display System) en un 38 mapa, probando la capacidad del comandante y del personal para utilizar sus fuerzas con eficacia. Se utiliza en los niveles de Batallón, Brigada y División, y permite que los comandantes y estados mayores “observen” lo planificado mediante modelos que simulan la conducta de unidades en el terreno. Opera con bases de datos de terrenos y armamentos que son específicos de cada país. Permite la integración de simuladores tripulados y de campos de entrenamiento instrumentados.

Se atiene a los estándares de la OTAN para simulación incluyendo los protocolos de comunicación DIS (simulación interactiva distribuida), y HLA (arquitectura de alto nivel). “Su objetivo es el entrenamiento para mejorar la habilidad en la toma de decisiones”. Ejemplo de este nivel de simulación es el Sistema de entrenamiento táctico SETAC del Ejército Chileno, el sistema táctico de adiestramiento tanto a nivel Brigada (SISTAB) como a nivel Batallón (SABRE) del Ejército de Brasil, el SETAC del Ejército de El Salvador.

**Simulación “Analítica”** se utiliza para estudiar problemas como la composición de la fuerza, la eficacia de las armas, y logística. Las simulaciones analíticas se diferencian generalmente en que no se centran en intercambios interactivos con la gente durante una simulación. Es capaz de producir simulaciones parecidas a la constructiva y permitir la

evaluación y análisis inmediato de los eventos. La ciencia permite que estos niveles sean aplicados a cualquier ambiente diferente al entorno militar como el entrenamiento en

Defensa Civil y en asistencia de desastres. El entrenamiento y toma de decisiones por parte de civiles ha logrado un estatus de importancia en el desarrollo de nuevos modelos orientados a la educación de cualquier persona. (ORTEGA GUTIERREZ & ANDRADE CARRILLO, 2012).

**M&S (Modeling and Simulation)** El modelado es una forma de representar un sistema del mundo real usando software, hardware o una combinación de ambos. Si los componentes de software de este modelo son impulsados por relaciones matemáticas, el modelo se puede simular bajo varios escenarios para verificar que el sistema se representa correctamente y que responde correctamente a las señales de entrada.

Modelado y simulación es una teoría valiosa para las condiciones de prueba que podrían ser difíciles de reproducir con prototipos de hardware por sí solos, especialmente en la fase temprana del proceso de diseño cuando el hardware puede no estar disponible para probar. (MathWorks)

**DEVS (Discrete Event Systems Simulation)** La simulación de sistemas de eventos discretos (DEVS) apoya la especificación orientada a objetos (OO) de modelos discretos de eventos de una manera jerárquica y modular.

### 3.3.4 Unidad de Combate

Organización militar de encuadramiento que reúne elementos y unidades de las armas y servicios, según una organización prevista pero flexible, capaz de prolongar y profundizar el combate y de ser empleada como un todo. (López, 2008).

### 3.3.5 Entrenamiento táctico-militar

Ejercicios prácticos militares, en el área técnica, táctica y Estratégica con el fin de acrecentar su capacidad de combate. (EJÉRCITO NACIONAL DE COLOMBIA, 2010).

# 4. METODOLOGÍA

## 4.1 DISEÑO DE LA INVESTIGACIÓN

En este apartado se presentan los aspectos referentes al diseño de la investigación. De esta manera se muestran los métodos de investigación, al igual que las técnicas e instrumentos utilizados en el estudio, su relevancia y la justificación de su utilización. Se describen pertinentemente las características de los métodos, técnicas, estrategias e instrumentos de recogida de datos.

### 4.1.1 Diseño de la metodología de la investigación

Se entiende por diseño de una investigación la manera de recolectar, analizar e interpretar los datos para luego proceder a realizar el escrito con la información obtenida.

**La Ilustración 8** muestra el esquema del diseño metodológico de este estudio que se ha organizado en las siguientes etapas:

**Etapa I. Preguntas de Investigación**: Las preguntas de investigación del estudio se planean para expresarse a través de los datos conseguidos por medios cualitativos.

**Etapa II. Recogida de datos**: Se ejecutó en base a las dos aproximaciones metodológicas:

**Cualitativa**: Las estrategias cualitativas de recogida de datos, como la observación cualitativa y la entrevista semiestructurada, se llevan a cabo con la muestra de tres oficiales de la armada seleccionados.

**Etapa III. Análisis**:

**Cualitativo**: Se realiza análisis cualitativo de la observación capturada por los dispositivos audiovisuales y de las entrevistas realizadas a los tres oficiales de la armada.

**Etapa IV. Interpretación**: La información obtenida se interpreta según cada método.

**Etapa V. Triangulación**: En esta etapa se triangulan la información dada por los acercamientos según el diseño de la investigación, con los datos cualitativos.

**Etapa VI. Conclusiones**: Se elaboran las conclusiones en base a la información obtenida de los análisis de los datos cualitativos y la triangulación.

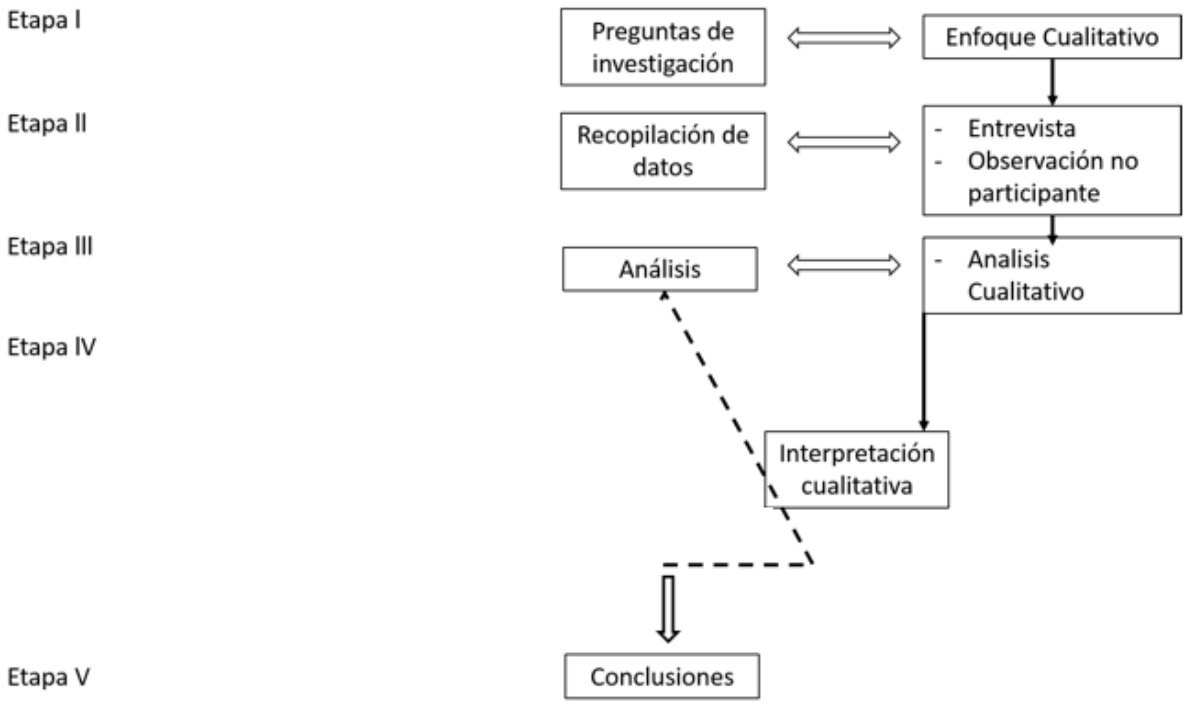


**Ilustración 8. Diseño Metodológico de la arquitectura**

## 4.2 ENFOQUE Y TIPO DE INVESTIGACIÓN

Para dar cumplimiento a los objetivos propuestos en el presente proyecto se llevó a cabo una investigación de tipo cualitativa, puesto que las temáticas que representan interés para el trabajo a realizar hacen parte del mundo real y requiere de simulaciones que generen resultados exactos, pero adicional también son basados en herramientas metodológicas propias de la investigación cualitativa, como la revisión documental, las entrevistas y las fuentes principales para la elaboración final del diseño, que es netamente basadas en un proceso sistemático de la creación de la arquitectura de software para elaboración del diseño de unidades de combate, además se pretende con la presente iniciativa sentar una base académica documentada para que sea utilizada en futuras investigaciones cuyas temáticas estén relacionadas con el área de estudio abordada en este trabajo.

A partir de las técnicas de recolección y análisis de la información, la investigación realizada se fundamenta en el tipo de investigación confirmatoria, con datos cuantitativos y análisis cualitativo, (Hernández, Fernández y Baptista 2003) señalan que los diseños mixtos: representan el más alto grado de integración o combinación entre los enfoques cualitativo y cuantitativo. Ambos se entremezclan o combinan en todo el proceso de investigación, agregando complejidad al diseño de estudio; pero contempla todas las ventajas de cada uno de los enfoques.

Debido a que los lineamientos de este tipo de técnica propician el uso de múltiples registros legibles que contengan diagramas, gráficos, tablas e imágenes que resultaron bastante útiles para la comprensión de las distintas temáticas abordadas cuyas teorías no fueron tan sencillas de entender pero cuya valía académica es tan alta que permiten soportar de una manera robusta y científicamente aceptada los distintos lineamientos propuestos a lo largo del presente documento.

El presente proyecto fue realizado en la ciudad de Cartagena De Indias, Bolívar durante los años 2014 y 2015 siguiendo los lineamientos estipulados por la Universidad De Cartagena

para presentar proyectos de investigación y por las directrices establecidas por el Programa De Ingeniería De Sistemas adscrito a la Facultad De Ingeniería para la elaboración de proyectos de grado. La investigación fue dirigida por la ingeniera Msc. Yasmín Moya Villa quien actualmente se desempeña como profesor e investigador del Programa De Ingeniería De Sistemas de la Universidad De Cartagena.

## 4.3 PROCEDIMIENTO DE TRABAJO

Para llevar a cabo el presente proyecto se siguió un procedimiento de trabajo por objetivos, consistente en la planificación y ejecución de actividades generales denominadas hitos determinadas a partir de cada uno de los objetivos propuestos en la investigación. Los hitos a su vez contienen actividades más específicas denominadas sub actividades, que son herramientas de la investigación que a su vez generaron resultados entregables y contribuyeron a la elaboración del informe final.

Para cada uno de los hitos y actividades realizadas se presenta un resumen a continuación:

- ***Estado del arte***

Se realizó una revisión minuciosa de la literatura mediante el uso de las bases de datos proporcionadas por la Universidad De Cartagena para construir un estado del arte completo y detallado que permitió contextualizar y concretar la actualidad de la temática abordada, con miras a contar con un soporte académico robusto que permitió sustentar los postulados propuestos que surgieron como resultado de la investigación. Este hito constituye uno de los objetivos del proyecto que más tiempo requirió debido a que a partir de los resultados de esta actividad se realizaría el resto del proyecto y fue fundamental revisarlo minuciosamente. En la ilustración 9 se aprecia la representación esquemática de este paso.

Sub actividades para este hito: -Revisión de la literatura.

-Recolección de información.

-Análisis de información.

-Elaboración de estado del arte.

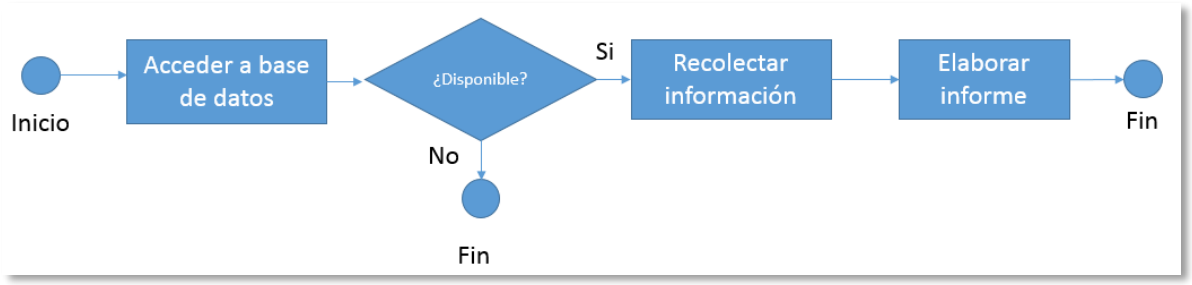


**Ilustración 9. Representación esquemática para la actividad Estado Del Arte. (Osorio Romero, 2014)**

- ***Identificación de simulaciones relacionadas con arquitecturas de software***

Tomando como base la documentación generada en la fase anterior, se procedió a identificar las necesidades que surgieron como resultado de las problemáticas y trabajos futuros encontrados en la literatura, propuestos por los autores que han abordado la temática con anterioridad. Posteriormente dichas necesidades se clasificaron siguiendo un orden prioritario de acuerdo a la recurrencia de las mismas en el estado del arte realizado. En la ilustración 10 se aprecia la representación esquemática de este paso.

Sub actividades para este hito:

-Análisis del informe del estado del arte.

-Definición de necesidades encontradas.

-Elaboración de informe.

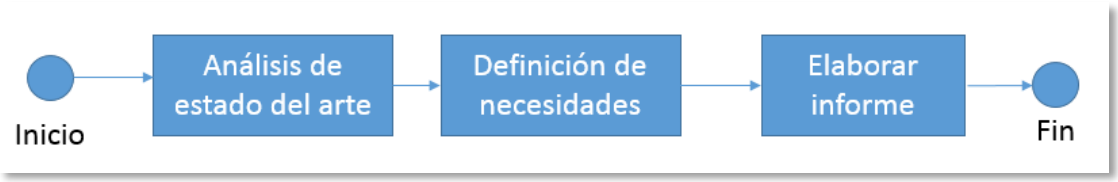


**Ilustración 10. Representación esquemática para la actividad de Identificación de Necesidades. (Osorio Romero, 2014)**

- ***Definición de la arquitecturas de software para el análisis de Unidades Militares***

En esta fase se empezó a indagar sobre trabajos que enfocaran la arquitectura de software al desarrollo de simuladores de guerra naval, los resultados obtenidos pudieron ser utilizados como base al desarrollo de nuestro trabajo de investigación. En esta fase se obtuvo como resultado una arquitectura basada en el patrón modelo – vista - controlador, que teniendo en cuenta el significado de este, se divide el sistema en tres capas donde, se da la encapsulación de los datos, la interfaz o vista y por último la lógica interna o controlador.

Sub actividades para este hito: -Definición de la arquitectura.
-Elaboración de informe.

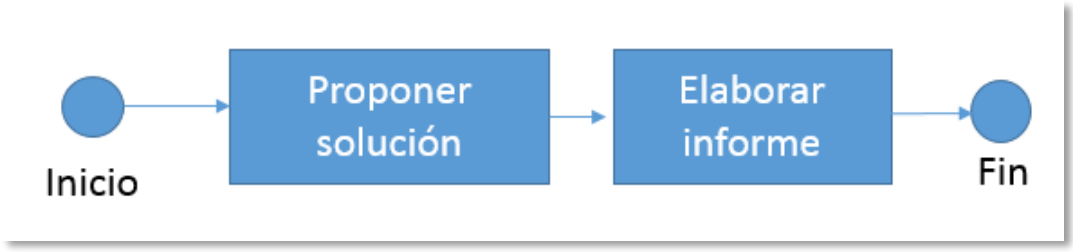


**Ilustración 11. Representación esquemática para actividad Definición de la arquitectura (Proponer solución). (Osorio Romero, 2014)**

## 4.4 TÉCNICAS DE RECOLECCIÓN Y ANÁLISIS DE INFORMACIÓN

Por la naturaleza de la investigación realizada anteriormente descrita en la sección *enfoque y tipo de investigación*, la recolección de información estuvo centrada exclusivamente en la revisión y análisis sistemático de la literatura. Las fuentes consultadas permitieron fundamentar la investigación realizada y los resultados obtenidos, debido a que en primer

lugar se restringió la búsqueda a bases de datos académicas que hacen parte de la comunidad científica y además, se aplicaron una serie de técnicas pertenecientes al proceso de revisión sistemática de literatura que permitieron discriminar los resultados obtenidos durante las búsquedas realizadas obteniendo información mucho más precisa y relacionada con las temáticas del proyecto.

Para dar solución a objetivo general del proyecto, se realizaron las siguientes actividades (Ver Anexo 1):

- Búsqueda y recolección de información referente a los procesos de desarrollo de software enfocado a la simulación de eventos discretos, teniendo como referente bibliográfico las bases de datos ofrecidas por la Universidad de Cartagena.

- Búsqueda y recolección de información referente a soluciones que se han dado en centros de entrenamiento militar, con el fin de definir el campo de acción de las tecnologías y prácticas involucradas, teniendo en cuenta que los referentes bibliográficos son los lugares que documentan las soluciones que han aplicado dicha tecnología.

- Búsqueda y recolección de información referente a arquitecturas plasmadas bajo Android que utilizan dispositivos externos para tener como punto de partida al momento de realizar las vistas arquitectónicas del sistema que se propone, cuyo referente bibliográfico son las bases de datos ofrecidas por la Universidad de Cartagena.

- A través de reuniones con el departamento de sistemas y personas a cargo de la iniciativa del proyecto en La Escuela, se recolectó la información de cómo se ha venido educando a los cadetes en el área de combate y cuáles son los requerimientos para el desarrollo. Con esta actividad se dio solución al primer objetivo.

- Una vez realizadas las actividades anteriores, se desarrolló el análisis de requerimientos que constó de 4 actividades: primera actividad, obtención de los requerimientos a través de entrevistas con el fin de descubrir, documentar y entender las necesidades y restricciones de La Escuela (Se grabaron archivos de audio); segunda actividad, se realizó el análisis de requerimientos, con el fin de

refinar las necesidades y restricciones de La Escuela; tercera actividad, se realizaron las especificaciones de los requerimientos, donde se documentaron de forma clara, las necesidades y restricciones de La Escuela; cuarta actividad, se hizo la verificación de los requerimientos con el fin de asegurar que estos estuvieran completos, correctos, consistentes y claros. Luego de esto, se diseñó la arquitectura del sistema, teniendo como base el patrón modelo- vista- controlador. Con esta actividad se dio solución al segundo objetivo.

- Luego de diseñar la arquitectura y los modelos, se realizaron las validaciones necesarias sobre un prototipo que aplica dicha arquitectura con un número mínimo de unidades para desarrollar una batalla. Con esta actividad se dio solución al tercer objetivo y por ende al objetivo general del proyecto.

# 5. DESARROLLO DE LA ARQUITECTURA

## 5.1. Objetivos significativos para la arquitectura

- Validar el diseño de la arquitectura, creando las estructuras que la componen y utilizando el patrón más adecuado para esta etapa.

- Comprender y mejorar la estructura de los elementos relacionados en la arquitectura, teniendo en cuenta que se trabaja sobre un ambiente militar enfocado a las unidades navales.

- Planificar la evolución del prototipo teniendo en cuenta la aplicación descrita, identificando las partes mutables e inmutables de la misma.

- Analizar el prototipo y su grado de cumplimiento respecto a los requerimientos iniciales para aplicar las pruebas pertinentes.

### 5.2. Restricciones de la arquitectura

Esta arquitectura estará enfocada única y exclusivamente a las unidades aero-navales que interactúan en una “Guerra Naval”, mostrando todas las características de las unidades registradas en la base de datos de la ENAP. De manera autodidactica y gracias a la experiencia de los oficiales de la ENAP, con esta arquitectura se podrá realizar una retroalimentación enfocada en las limitaciones del uso de una determinada unidad en un caso o situación dada, teniendo en cuenta las ventajas y desventajas que tiene su uso.

| Restricción | Requerimiento de la arquitectura |
|---|---|
| Tiempo | La arquitectura debe ser desarrollada y entregada antes de la culminación del año 2015, en conjunto a un prototipo funcional que valide dicha arquitectura |
| Desarrollo | El prototipo de la arquitectura propuesta está hecho en el lenguaje de simulación VENSIM y puesto en marcha en la web gracias a la tecnología JavaScript. |
| Negociación | La arquitectura de unidades se tomará como base para un macro proyecto de la ENAP |
| Coste | Los costes del desarrollo de la arquitectura correrán por cuenta de los investigadores, en colaboración con la ENAP y la Universidad de Cartagena |

**Tabla 6. Relación Restricción x Requerimiento con respecto al proyecto**

### 5.3. Definición de unidades militares navales a desplegar en el sistema

Las unidades militares sobre las cuales se trabajó esta investigación, están dadas de la siguiente manera:

**NOMBRE DE LA NAVE** (Otros nombres que haya tenido), (fecha de creación).

Desplazamiento normal, Peso en toneladas (Máximo peso en toneladas).

Longitud total, Longitud en pies. Beam (Medida de estribor a babor), Longitud en pies. Profundidad del buque Cargado, Longitud en pies. Profundidad máxima del buque Cargado, Longitud en pies. (Línea de flotación) Longitud en pies.

Armas: Dimensiones en pulgadas y pequeña descripción de funcionamiento.

Torpedos:

Armadura Vertical: Dimensiones en pulgadas y pequeña descripción de funcionamiento.

Armadura de cubierta interior: Dimensiones en pulgadas y pequeña descripción de funcionamiento y sus componentes.

**Maquinaria**: Descripción de la maquinaria de la nave.

**Carbón**: Uso del carbón para el funcionamiento de la nave.

**Notas de Artillería**: Anotaciones del armamento de la nave.

**Notas de Torpedo**: Anotaciones de los torpedos de la nave.

Cabe resaltar que no todas las naves tienen la misma información debido a que su funcionamiento y el fin para las que fueron creadas son diferentes en cada situación.

**1. AGINCOURT** (ex Sultán Osman I, ex Río de Janeiro), (enero de 1913).

Desplazamiento normal, 27.500 toneladas (30.250 toneladas de carga completa).

Longitud total, 671,5 pies. Beam, 89 pies. Proyecto, 27 pies de media. Max. proyecto, 30.5 ft. Longitud (pp) 632 pies, (línea de flotación) 668 pies.

Guns (Elswick): 14-12 pulgadas, 50 cal. Dir. Con .; 20-6 pulgadas, 50 cal. Dir. Con .; 8 - 3 pulgadas; 2-3 pulgadas (antiaéreo); 4-3 pdr .; 1-3 pdr. (Antiaéreo)

Tubos de torpedos (21 pulgadas): 2 sumergida (Elswick de carga lateral)

Armadura Vertical: 9 " cinta inferior;  6" correa superior; 6 "- 4"  fines; 6 "- 3" Los mamparos; 6 " Batería; 6" mamparos de la batería;  9" 1 barbetas; 12 "- 8" Gunhouses; 12 "C.T. (6" - 4 "Campanas); 8" Fore Com. Tubo; 9 "C.T. Torpedo; 6" Después de Com. Tube.

Armour 2½ "cubierta inferior protección Timón especial: 1 ½" - 1 "pantallas dobles (barbetas finales); 1½" pantallas individuales (midships barbetas).

**Maquinaria**: Parsons turbina 4-eje. Calderas: 22 Babcock. Diseñado H.P. 34000 = 22 nudos.

**Carbón**: Normal, 1.500 toneladas; máximo, aceite de 3200 toneladas + 620 toneladas.

**Notas de Artillería**.- Las barbetas son un diseño especial de Elswick, todas las operaciones de carga y de disparo son controladas por una sola palanca, que trabaja en un cuadrante muy parecido a la caja de cambio de velocidad de un automóvil.

**Notas de Torpedo**.- torpedos de 21 pulgadas con pequeñas cabezas de guerra. El agua de los tubos se bombea hacia fuera en plano, para recargar; cuando se requiere un rápido lanzamiento de torpedos, hay 3 pies de agua en plana.

**2. BELLEROPHON** (julio de 1907), Temeraire (agosto de 1907), MAGNÍFICO (noviembre de 1907).

Desplazamiento normal, 18.600 toneladas (alrededor de 22.000 toneladas de carga completa).

Longitud (línea de flotación), 520 pies. Beam, 82,5 pies. Max. proyecto, 30 '11 "calado medio de 27,25' pies. Eslora total, 526 pies (pp 490 pies).

**Armas de fuego**: 10 a 12 pulgadas 45 cal. Dir. Con .; 11-4 pulgadas (. Tem y Superb) Dir. Con .; 10-4 pulgada (de Bell); Dir. Con .; 2-4 pulgadas (antiaéreo B. & T.); 2-3 pulgadas (antiaéreo Sup.); 4-3 pdr. 5 M. G .; 1 (de aterrizaje).

Tubos de torpedos (18 pulgadas, M. '04): 2 sumergidos (del costado)

**Armas** (Vertical), (K.C.): 10 "- 9" cinturón inferior; 8 "cinturón superior; 7" - 6 "Baja y correas superiores (arco); 5" Belt (popa); 8 "Después del mamparo, 10" - 9 "barbetas; 11" - "Gunhouses; 11" Formulario CT (4 "tubo); 8" Después C.T. (4 "tubo)

**Armas** (Cubiertas), (K. N. C): 1.75 "- 0.75" Principal; 3 "- 1.75" Oriente; 4 "- 1.50" Inferior.

**Maquinaria**: Parsons turbina. 4 tornillos. Dreadnought. Calderas: 18 Babcock o Yarrow. Diseñado H.P. 23,000 = 20,75 nudos.

**Carbón**: normales, 900 toneladas; máximo, 2.648 toneladas + 842 toneladas de aceite + 170 toneladas de combustible de patentes.

**3. CANADÁ** (ex Almirante Latorre), (noviembre de 1913)

Desplazamiento, 28.000 toneladas (alrededor de 32.000 a plena carga).

Complemento (1176)

Longitud (P.P.), de 625 pies. Beam, 92,5 pies., Calado medio, 29 pies, Max. redactar 32 pies. Longitud (sobre todo). 661 pies.

**Armas** (Elswick): 10-14 pulgadas, 45 cal. Dir. Con. ; 12-6 pulgadas 50 cal. Dir. Con .; 2-3 pulgadas (antiaéreo); 4-3 pdr .; (2 aterrizaje); 4 M. G.

Tubos de torpedos (21 pulgadas): 4 sumergidos

**Armadura Vertical**: 9 "cinta inferior; 7" Middle Belt; 4½ "Cinturón superior; 6" - 4 "Cinturón (extremos); 4½", 4 "Los mamparos (f & a..); 6" Batería *; 10 "barbetas; 10" - "Gunhouses; 3" C.T. Base *; 11 "C.T. (6" - 3 "Hood); 6" Fore Com. Tubo; 6 ". Torp C.T.; 6" Después de Com. Tube. * No se muestra en los planos.

**Cubierta Armadura**: 1 "Shelter (más de casamatas); 1" Fo'c'sle (sobre batería); 1½ Superior (batería externa); 1½ "Principal (popa); 1" de protección; 2 "(adelante) inferior; 4" (después) Inferior.

**Maquinaria:** Turbina, 4 - Eje: (L.P.) Parsons; (H.P.) Brown-Curtis. Calderas: 21 Yarrow. Diseñado H. P. 37,000 = 22,75 nudos.

**Carbón:** Normal, 1.150 toneladas; máximo, 3.300 toneladas. Petróleo: 520 toneladas.

**4. COLOSO** (abril de 1910) y HERCULES (mayo de 1910).

Desplazamiento normal, 20.000 toneladas (alrededor de 22.250 a plena carga).

Complementar, 831 y 845.

Longitud (línea de flotación), 540 pies. Beam, 85 pies. Calado (normal), 27 pies. Calado (máx.), 31 pies. Eslora total, 546 pies. Longitud, (P.P.) 510 pies.

**Armas de fuego**: 10 a 12 pulgadas, 50 cal. Dir. Con .; 12-4 pulgadas Dir. Con .; 1-4 pulgadas (antiaéreo) (Col.); 1-3 pulgadas (antiaéreo) (. Herc)); 4-3 pdr .; 5 M.G .; (1 aterrizaje)

Tubos de torpedos (21 pulgadas): 2 sumergidos

**Armadura Vertical** (KC): 11 "correa inferior; 8" cinturón superior; 1,5 ", 2", 5 "y 8" Los mamparos; 7 "- 2.5" Belt (arco); 2.5 "Belt (popa); 10" - 7 "barbetas; 11" - "Gunhouses; 1" captaciones embudo; 11 "(" capuchas; 5 6 C.T. "y 3)" Fore com. Tubo; 3 "Torpedo C.T.; 3" Tube, T.C.T.

**Maquinaria:** Parsons turbina; Diseñado H.P. 25000 = 21 nudos. Calderas: Yarrow o Babcock.

**Carbón**: normales, 900 toneladas; máximo, aceite de 2900 toneladas + 800 toneladas.

**5. COMMONWEALTH** (mayo de 1903). (Re-construido 1918).

Desplazamiento normal, 16.350 toneladas. Carga completa; 17.050 toneladas.

Longitud: línea de flotación, 439 metros; sobre todo, 457 pies; Manga: en W.L., 78 pies. protuberancias exteriores, , 26.75, pies media. Max. 28 pies.

**Armas de fuego**: 4-12 en, IX 40 cal.. Dir. Con. 4-9,2 en, IX, 45 cal.. Dir. Con .; 4 - 6 pulgadas, VII; 8 - 12 pdr .; 2-3 pulgadas (antiaéreo); 6-3 pdr, 2- 3 pdr.. (Antiaéreo) (sic); 5 M. G., (1 de aterrizaje)

Tubos de torpedos (18 pulgadas): 4 sumergido (del costado)

**Armadura** (Krupp); 9 "Belt (el centro del buque); 7" - 3 "Cinturón (adelante) 2.5." Cinturón (después); 2 "- 1" Principal, medio cubiertas inferiores; 12 "- 6" barbetas (Carolina del Norte); 12 "- 8" Gunhouses (K.C.); 8 "lado Cubierta inferior; 7" de la batería (2 "atraviesa); 4" torretas secundarias; 12 "C.T. (6" tubo)

Protección especial: bombeos profundos.

**Maquinaria:** 2 conjuntos de 4 cilindros de triple expansión. 2 tornillos. Calderas: 16 Babcock. Diseñado H.P. 17500 (n.d.), 18.000 (F.D.) = 18,5 nudos. (Se añadieron 15 ya que la protección bulto).

**Carbón**: normales, 950 toneladas; máximas, 2.194 toneladas + 380 toneladas de combustible líquido.

**6. KING GEORGE V** (octubre de 1911), CENTURION (noviembre de 1911), AJAX (marzo de 1912).

Desplazamiento normal, 23.000 toneladas (alrededor de 25.500 a plena carga).

Eslora total, 597,66 metros. Beam, 89 pies. Proyecto, 27.5 ft media. Max. . 30 ft 10 in. Longitud (P.P.), 555 pies.

**Armas de fuego:** 10 a 13,5 pulgadas (M.V.) Dir. Con. 16-4 pulgadas Dir. Con. 2-4 pulgadas (Cent antiaéreo.); 2-3 pulgadas (antiaéreo Ajax, K.G.V.); 4-3 pdr .; 5 M.G .; (1 aterrizaje);

Tubos de torpedos (21 pulgadas) 2 sumergida (del costado)

**Armadura** (Vertical) (K.C.); 12 "correa inferior; 9" cinturón de Oriente; 8 "cinturón superior, 4", 6 "8" B'lkh'ds (primer plano); 2 "10" 8 "B'lkh'ds (después); 6" - 4 "Belt (arco); 2,5" Belt (popa); 10 "- 7" barbetas; 11 "- Gunhouses; 1.5" captaciones de embudo; 3 "- 1" de la batería (4 cañones pulgadas); l "base CT (5" tubo dentro) 11 "CT (4" campana); 6 "Torpedo C.T. (4" tubo)

**Armadura** (cubierta) (HT) I.75 ", 1,5 Superior; 1.5 Principal; 1" Oriente; 2.5 "'- 1" (arco); 3 "(popa) inferior; 4" (más del timón).

**Maquinaria:** Turbina (Parsons); Calderas: 18 Babcock o milenrama; Diseñado H.P. 27000 = 21 nudos.

**Carbón:** normal. 900 toneladas; máximo (A. & C.) 3150 toneladas, (KGV) 2.870 toneladas de petróleo + 850 toneladas en los tres.

**7. DREADNOUGHT** (febrero de 1906).

Desplazamiento normal, 17.900 toneladas. A plena carga, a unos 20 700 toneladas.

Longitud (w.l), 520 pies. Beam, 82 pies. Calado máximo, 31 pies. Proyecto 26.5 La media; Longitud, (O.A.) 526 pies (P.P. 490 pies).

**Maquinaria:** Parsons turbina. 4 tornillos; Calderas: 18 Babcock & Wilcox en 3 grupos; diseñado H.P. 23000 = 21 nudos.

**Carbón:** normales, 900 toneladas; máximo, 2.900 toneladas + 1.120 toneladas de aceite + 120 toneladas de combustible de patentes. Radio nominal: 6600 a los 10 nudos .; 5000 a los 19 nudos.

**8. ERIN** (ex Reshadieh), (septiembre de 1913).

Desplazamiento normal, 23.000 toneladas; 25.250 a plena carga.

Longitud (o.ä.,) 559.5 pies. Beam, 91 ft. 7.25 en. Max. proyecto,. 30 ft 11 in., calado medio de 28 pies. 5 en. Longitud (pp), de 525 pies.

**Armas** (Vickers): 10 a 13,5 pulgadas, 45 cal. Dir. con,; 16-6 pulgadas, 50 cal. Dir. con,; 2-3 pulgadas (antiaéreo)

Tubos lanzatorpedos. (21 pulgadas): 4 sumergido.

**Armadura Vertical** (K.C.): 12 "- 9" cinta inferior; 8 "Cinturón superior; 8", 5 ", 4" Los mamparos (. F & A.); 5 "Batería *; 1" Traverses batería; 10 "- 8" barbetas; 11 "Gunhouses; 1" captaciones de embudo; 12 "C.T. (6" tubo); 4 "(tubo 3) Torpedo C.T.".

**Armadura Cubierta** (HAT): 1½ "Fo'c'sle (sobre batería); 1½ superior (más allá de la batería); 1½" principal; 3 "(Final), Media; 1" (el centro del buque), Oriente.

**Maquinaria:** Parsons turbinas. 4 tornillos. diseñado H.P. 26500 = 21 nudos. Calderas: 15 Babcock.

**Carbón:** normal; 900 toneladas; máximo, el aceite de 2.120 toneladas + 710 toneladas.

**9. HOOD** (22 de agosto 1918).

Desplazamiento normal, 41.200 toneladas (hasta 45.000 toneladas a plena carga).

Longitud: P.P. 810 * ft .; O.A. 860 ft .; Haz W.L. ? fuera protuberancias 104 pies Calado:. significa 28.5 pies máx.. ? * Aproximado

**Armas de fuego:** 8-15 pulgadas, 42 cal. Dir. Con. ; 12 a 5,5 pulgadas, 50 cal. Dir. Con. ; 4-4 pulgadas (antiaéreo); 4-3 pdr. ; 5 M. G.; (Aterrizaje I).

Tubos de torpedos (21 pulgadas): Desconocido.

**Armadura Vertical:** (KC): 12 "correa inferior; 5", 6 "cinturón Baja; 3" (proa y popa) 7 "cinturón de Oriente; 5" cinturón Medio (arco); 5 "cinturón superior; __ mamparos; 12" barbetas; 15 "- 11"

**Armadura - Skate** (HT): 2 "Fo'xle (amids); 1½" Flat, principal; 5 "Cuestas, principal 1" Bow (inferior); 3 "Stern (Baja)

**Maquinaria:** Brown-Curtis (dirigidos) turbinas. 4 tornillos; Calderas. Yarrow; Diseñado S.H.P. 144000 = 31 nudos.

**Combustible:** sólo aceite.  4.000 toneladas máximo.

**10. INFLEXIBLES** (junio de 1907) e INDOMABLE (marzo de 1907).

Desplazamiento normal, 17.250 toneladas (alrededor de 20.000 a plena carga.)

Longitud (línea de flotación), 560 pies. Beam, 78 ft 10 en calado medio, 26 pies; Max. proyecto 29.75; los pies. Eslora total, 567 pies (530 P.P.).

**Armas de fuego:** 8-12 pulgadas XI, 45 cal. Dir. Con. ; 12-4 pulgadas Dir. Con. ; 1-4 pulgadas (antiaéreo); 1-3 (antiaéreo); 4-3 pdr. ; 5 M. G. (1 aterrizaje)

Tubos de torpedos (18 pulgadas): 4 sumergidos.

**Armadura Vertical** (KC): 6 "correa (en medio del barco); 4" Belt (arco); 7-6 "Los mamparos; 7" barbetas; 7 "

**Armadura - cubierta:** 1 "- ¾" principal; 1½ "Oriente; 2½" Inferior (popa), 2 "Cuestas, Baja (centro del buque), 1½ plana inferior (centro del buque).

**Maquinaria:** Parsons turbinas. 4 tornillo. Calderas: Yarrow o Babcock. diseñado H.P. 41000 = 25 nudos.

**Carbón:** Normal, 1.000 toneladas; máximas, 3.080 toneladas + 710-725 toneladas de combustible líquido.

**11. Benbow** (Nov., 1913); Emperador de la India (ex-Delhi, noviembre de 1913); Iron Duke (octubre de 1912); MARLBOROUGH (Nov., 1912).

Desplazamiento normal, 25.000 toneladas. A plena carga, 28.800 toneladas. Complementar, 995-1.022.

Longitud (O.A.), 622.75 pies. Beam, 89,5 pies. Proyecto, 28,5 pies media. Max. proyecto de 32.75 metros de longitud (P.P.), 580 pies

**Armas de fuego:** 10 -13.5 pulgadas (M.V.) Dir. Con .; 12-6 pulgadas, 50 cal. Dir. Con .; 2-3 pulgadas (antiaéreo); 4-3 pdr .; 5 M.G .; (1 aterrizaje)

Tubos de torpedos (21 pulgadas): sumergido (del costado)

**Armadura Vertical** (K.C.): 12 "-8" cinta inferior; 9 "Cinturón Medio; 8" Cinturón superior; 6 "- 4" Cinturón (extremos); 8 ", 6", 4 "Los mamparos (f & a..); 6" Batería *; 10 "- 7" barbetas; 11 "-"

**Armadura Cubierta** (HT): 1 "Fo'c'sle (sobre batería); 2" - 1¼ superior; 1 "(Centro del buque), Oriente; 2 ½ - 1 ½" (popa), Oriente; 2 ½ "- 1" Baja

**Maquinaria:** Turbina (Parsons). Calderas: (véanse Notas). Diseñado H.P. 29000 = 21 nudos.

**Carbón:** Normal, 1.000 toneladas; máximo, 3.250 toneladas. Petróleo: Normal, 1.050 toneladas; máx., 1600 toneladas.

**12. LEÓN** (agosto de 1910), Princesa Real (abril de 1911).

Princesa Real

Desplazamiento Normal, 26.350 toneladas. A plena carga, 29.700 toneladas.

Longitud (W. L.), 675 pies. ; Beam, 88,5 pies. Proyecto, 27.66 pies media. Maxx. proyecto, 31.66 pies. Eslora total, 700 pies. Longitud P.P., 660 pies ..

**Armas de fuego:** 8-13,5 pulgadas (M.V.) Dir. con .; 16-4 50 pulgadas cal. Dir. Con. ; 2-3 pulgadas (antiaéreo); 4-3 pdr. ; 5 M. G. (1 aterrizaje)

Tubos de torpedos (21 pulgadas): 2 sumergida (del costado)

**Armadura Vertical** (KC): 9 ", 6", 5 "cinturón inferior; 6", 5 ", 4" cinturón superior; 4 "Los mamparos; 9" - 8 "barbetas; 9"

**Armadura – cubierta**: 1 "superior; 1 ½" - 1 "en medio del barco (inferiores); 2½" Ends (inferior).

**Maquinaria:** Parsons turbina. 4 tornillos. Calderas: 42 Yarrow. diseñado H.P. 70000 = 28 nudos.

**Carbón:** normales 1.000 toneladas; máximo, 3.500 toneladas de aceite + 1.135 toneladas.

**13. NEPTUNO** (septiembre de 1909).

Desplazamiento normal, 19.900 toneladas (alrededor de 22.000 a plena carga).

Longitud (línea de flotación), 540 pies. Beam, 85 pies. Calado (normal), 27 pies, Calado (máx.) 30 pies; Eslora total, 546 pies (P.P. 510 pies).

**Armas de fuego:** 10 a 12 pulgadas, 50 cal. Dir. con .; 12-4 pulgadas Dir. con .; 2-3 pulgadas (antiaéreo); 4-3 pdr .; 5 M.G .; (1 aterrizaje)

Tubos de torpedos (18 pulgadas): 2 sumergidos

**Armadura vertical** - (KC): 10 "correa inferior; 8" cinturón superior; 5 "y 8" Los mamparos; 7 "- 2.5" Belt (arco); 2.5 "Belt (popa); 9" barbetas; 11 "-"

**Armadura - Cubiertas** - (Níquel): 1.25 "Principal; 1,75" Oriente; 1,50 (arco) - inferior; 3 "(popa) -inferior.

**Maquinaria:** Parsons turbina; diseñado H.P. 25000 = 21 nudos. ; Calderas: Yarrow.

**Carbón:** normales 900 toneladas; 2.710 toneladas máxima + 790 toneladas de combustible líquido.

**14. NUEVA ZELANDA** (julio de 1911).

Desplazamiento normal, 18.800 toneladas; cerca de 20.000 toneladas, a plena carga.

Tripulación, 853.

Longitud (línea de flotación), ?? pies; Beam, 80 pies; Calado medio, 26,5 pies, Max. 30 pies; Longitud sobre-todo 590 pies. (P.P., 555 pies).

**Armas de fuego:** 8-12 pulgadas 50 cal. Dir. Con. ; 10-4 pulgadas Dir. Con. ; 1-4 pulgadas (antiaéreo); 4-3 pdr. ; 2 - 2 pdr. pompón; 5 M. G. (1 aterrizaje)

Tubos de torpedos (18 pulgadas): 2 sumergida (del costado)

**Armadura Vertical** (KC): 6 "correa (en medio del barco); 5" - 4 "Cinturón (adelante); 4" Los mamparos; 7 ", 4", 3 "barbetas; 7"

**Armadura - cubierta**: 1 "Principal; 2" - 1 "Centro del buque (piso inferior); 2½" para ya popa, (piso inferior).

**Maquinaria:** Parsons turbina. 4 tornillos. Diseñado H.P. 44000 = 25 nudos. Calderas: Babcock.

**Carbón:** Normal, 1.000 toneladas; máximo, aceite de 3170 toneladas de carbón + 840 toneladas.

**15. ORION** (agosto, 1910). Tronador (enero de 1911), monarca (marzo de 1911), CONQUISTADOR (mayo de 1911).

Desplazamiento normal, 22.500 toneladas (alrededor de 25.000 a plena carga.)

Longitud (p p.), 545 pies. Beam, 88,5 pies. Calado medio, 26 pies y 10 pulgadas, Max. 30.75 pies; longitud sobre todo, 581 pies.

**Armas de fuego:** 10 a 13,5 pulgadas (M.V.), dir. Con. ; 16-4 pulgadas, Dir. Con. ; 1-4 pulgadas (antiaéreo); 1-3 pulgadas (antiaéreo); 4-3 pdr. ; 5 M.G .; (Aterrizaje I);

Tubos de torpedos (21 pulgadas): 2 sumergida (del costado)

**Armadura (Vertical)** (KC): 12 "correa inferior; 9" cinturón de Oriente; 8 "cinturón superior, 4", 6 ", 8" B'lkh'ds (primer plano); 6 "- 4" Cinturón (arco); 2.5 "Belt (popa); 2.5", 8 ", 10" B'lkh'ds (después); 10 "- 7" barbetas; 11 "-"

**Armadura (Níquel)**: 1.5 "Upper; 1.5" Principal; 1 "Oriente; 2.5" Baja; 4 "sobre el timón;

**Maquinaria:** Parsons turbina; Diseñado H.P. 27000 = 21 nudos. 4 tornillos; Calderas: 18, (véanse Notas).

**Carbón:** normales, 900 toneladas; 3.300 toneladas de carbón máximo de aceite + 800 toneladas.

**QUEEN ELIZABETH** (octubre de 1913); Warspite (Nov., 1913); VALIANT (04 de noviembre 1914); BARHAM (31 de diciembre 1914); Malaya (18 de marzo 1915)

Desplazamiento normal, 27.500 toneladas (alrededor de 31.000 - 33.000 a plena carga).

Longitud (P.P.), 600 pies. Beam, 90 1/2 pies. Proyecto, 30 2/3 pies media. Max. redactar 33 1/2 pies. Longitud (sobre todo) QE 643 3/4 ft. Descanse 639 3/4 pies.

**16. REPULSE** (08 de enero 1916), Renown (04 de marzo 1916).

Desplazamiento normal, 26.500 toneladas (alrededor de 32,000-32,700 plena carga).

Longitud (P.P.), 750 pies. O.A. 794 pies Repulse; O.A. 794 pies 1.5 en Renown.; Haz: Repulsión 90 pies * Renown 90 pies 2 en * Calado:. Repulse 26,25 pies (media), 30,5 pies (máx.); Renown 26,66 pies (medias), 30 pies (máx.) * Protuberancias externas.

**Armas de fuego:** 6-15 pulgadas, 42 cal. Dir. Con .; 17-4 pulgadas, 50 cal. Dir. Con .; 2-3 pulgadas (antiaéreo); 4-3 pdr .; 5 M. G .; (1 aterrizaje).

Tubos de torpedos (21 pulgadas): 2 sumergida * Ver Torpedo Notes.

**Armadura Vertical:** (KC): 6 "(el centro del buque), cinta inferior; 4" (dentro del arco), cinta inferior; 3 "(popa), cinta inferior; 4" Para mamparo, cinta inferior; 3 "Después b'lkh'd, cinta inferior; 1.5" cinturón superior; 7 "- 4" barbetas; 11 "- 7"

**Armadura - Skate** (H.T.): 1,5-0,5 Fo'xle; 16.7 "- 1.5" Alta; 3 "- 0.75" Principales (2 "pendientes; 2,5)" Bow (inferior); 3.5 "- 3" Stern (inferior);

**Armadura - Coronas:** - "barbetas; 3" C.T. y campana; 1.5 "Torpedo C.T.

**Maquinaria:** Brown-Curtis (accionamiento directo) turbinas. 4 tornillos. Calderas: 42 Babcock & Wilcox. Diseñado H.P. no especificado con exactitud, pero se espera que sea de 110.000 a 120.000 SHP de 30 nudos. En el servicio, S.H.P. 112 000 = aproximadamente 31,5 nudos.

**Combustible (sólo aceite):** 1.000 toneladas normales; Repulsa 4243 toneladas como máximo; Renown 4289 toneladas máximo.

**17. Royal Sovereign** (29 de abril 1915); ROYAL OAK (17 de noviembre 1914); RESOLUTION (14 de enero 1916); RAMILLIES (12 de septiembre 1916); VENGANZA (29 de mayo 1915).

Desplazamiento normal, 25.750 toneladas (alrededor de 31.250 - 33.500 toneladas a plena carga).

Longitud (sobre todo), 624¼ pies. Beam, 88½ pies. * La media de proyecto, 27 pies. Longitud (P.P.), 580 pies

**Armas de fuego**: 8-15 pulgadas, 42 cal. Dir. Con .; 14-6 pulgadas, 50 cal. Dir. Con. ; 2 -3 pulgadas (antiaéreo); 4-3 Pdr .; 5 M. G.

Tubos lanzatorpedos de 21 pulgadas): 4 sumergidos

**Armadura Vertical** (KC): 13 "Cinturón; 6" - 4 "Cinturón (extremos); 1" Belt (arco); 6 ", 4" Los mamparos (. F & A.); 6 "Batería; 10" - 7 "barbetas; 13"

**Armadura Cubierta** (HT): 1 "Fo'xle sobre la batería; 1¼" - 1½ superior; 2 ", 1 ½", 1 "principal; 2 ½", 1 "(forw'd) Bajo; 4", 3 ", 2 ½" (después) Inferior.

**Maquinaria:** Turbina, Parsons. Calderas: 4 tornillos. Diseñado H.P. 40.000 = 23 nudos. sin protuberancias, sobre 21,5-22 con protuberancias.

**Combustible:** Petróleo única, normal, 900 toneladas; máximo, alrededor de 3.400 toneladas. Carbón: 140 toneladas (sólo para uso "doméstico")

**Artillería Notas:** La Batería está dispuesta de manera diferente en estos barcos. Baterías de 6 pulgadas son húmedas en mar de proa, pero los muretes en batería retienen el agua y se escurre rápidamente lejos.

**18. ST. VICENTE** (septiembre de 1908); y COLLINGWOOD (noviembre de 1908).

Desplazamiento normal, 19.250 toneladas. A plena carga, a unos 22 900 toneladas.

Longitud (línea de flotación), 530 pies. Beam, 84 pies. Calado medio. 27 pies. Max. redactar 31.25 pies. Eslora total, 536 pies (P.P. 500 pies).

**Armas de fuego:** 10 a 12 pulgadas, - Cal. Dir. Con .; 12-4 pulgadas Dir. Con .; 2-3 pulgadas (antiaéreo); 4-3 pdr .; 5 M.G .; (1 aterrizaje)

Tubos de torpedos (18 pulgadas): 2 sumergida (del costado)

**Armour - vertical** - (KC): 10 "correa inferior; 8" cinturón superior; 7 "y 2" Cinturón (arco); 2 "Cinturón (popa); 8" - 5 "Los mamparos; 9" barbetas; 11"

**Armadura - Cubiertas** - (K.N.C.): 1,5 "- 0.75" Principal; "Oriente; 1.5" I.75 - 3 " Baja

**Maquinaria:** Parsons turbina (como Dreadnought). 4 tornillos; Calderas: Yarrow o Babcock. Diseñado H.P. 24500 = 21 nudos.

**Carbón:** normales, 900 toneladas; máximo, 2.800 toneladas + 940 toneladas de aceite + 190 toneladas sensación patente.

**5.4. Inicios de un trabajo orientado a objetos: herramientas y soluciones**

Durante las últimas dos décadas, se ha producido una migración constante hacia el modelado orientado a objetos y simulación de entornos (Singh & Sarjoughian, Software Architecture for Object-Oriented Simulation Modeling and Simulation Environments: Case Study and Approach, 2003). Existen muchos temas además de la utilización de un enfoque de modelado y simulación y una programación orientada a objetos, que son importantes para tener en cuenta en el desarrollo de entornos de modelado y simulación. En este trabajo, se aplica el modelado y simulación de entorno orientado a objetos llamado DEVSJAVA, debido a sus rasgos arquitectónicos es capaz de brindar apoyo a las funcionalidades esenciales de una simulación y métodos de la teoría del sistema.

Los conceptos y principios de la arquitectura de software se aplican a la modelación de sistemas de eventos discretos y marcos de simulación. Para aumentar el apoyo, DEVSJAVA brinda un enfoque específico para la realización de estudios de simulación.

### 5.4.1. Método de modelado de la Arquitectura (M&S)

Durante muchos años, los entornos de modelado y simulación han sido utilizados como parte de un repertorio herramientas de ingeniería de software. La ingeniería de sistemas basados en computadoras, en particular las que son complejas y de gran escala, cada vez requieren de más modelos de simulación.

Puesto que hay numerosos modelados y herramientas de simulación para una variedad de métodos de modelado y técnicas de simulación, es indispensable conocer las relaciones entre la construcción de modelos de simulación y ejecución en relación con la arquitectura de software, diseño y desarrollo, para escoger la mejor opción y evaluar numerosos diseños y conceptos operativos para un sistema del mundo real.

En general, existen varias alternativas de desarrollo de procesos de software, cascada, espiral, proceso ágil, entre otros. Si seguimos el proceso de cascada, el método más popular, el proceso se compone de la siguiente manera, desarrollo actividades: planificación, análisis de requerimientos, diseño, implementación, prueba, y

mantenimiento. La Ilustración 12 proporciona una comparación lado a lado entre un proceso general de desarrollo de software y el proceso de M & S para el desarrollo de software de simulación.

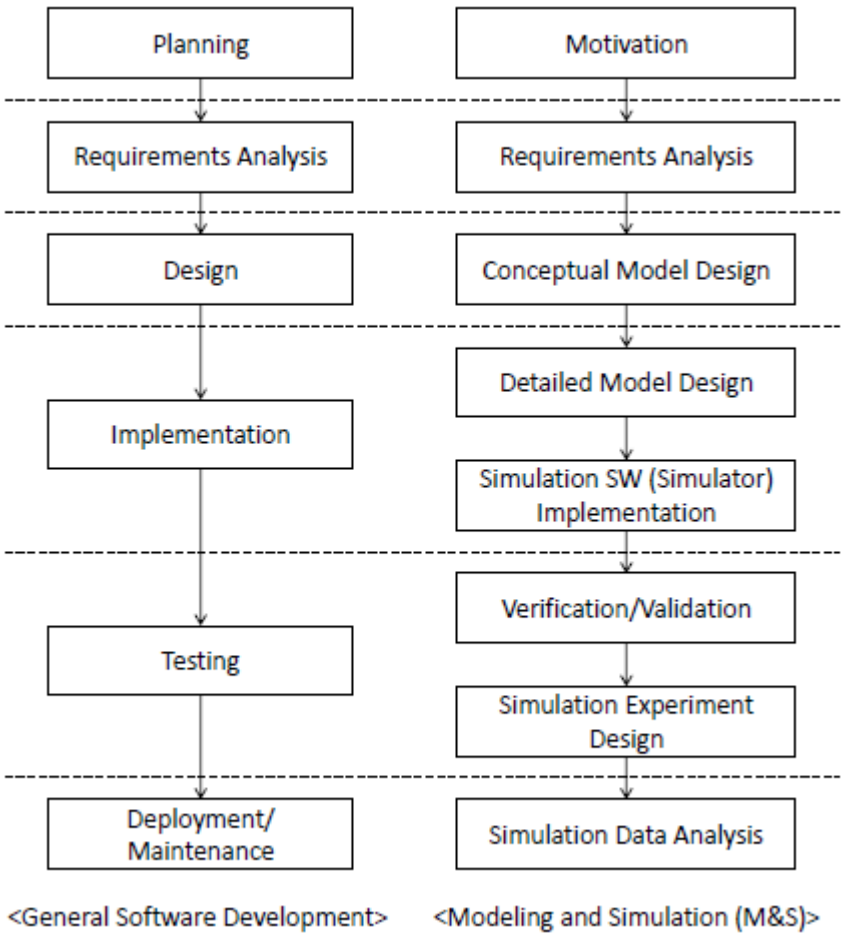


**Ilustración 12. Comparación de actividades entre el desarrollo general de software y el proceso M&S.**
(Sung, Moon, & Kim, 2010)

El proceso de M & S puede corresponder al ciclo de desarrollo de software, pero el Ciclo de M & S tiene distintas actividades en el modelo, como motivación, análisis de requerimientos, diseño del modelo conceptual, diseño detallado del modelo,

implementación de software de simulación, verificación y validación, diseño de experimentos de simulación, y el análisis de datos de simulación.

En cuanto a la amplia gama de usos de M & S, este es aplicable a conceptos fundamentales de Modelado y Simulación independiente del campo en particular. Una teoría general de modelado y simulación, basada en los fundamentos de la teoría de sistemas (por ejemplo, de descomposición y agregación) suministra las bases hacia la creación de software basado en entornos.

El punto de vista conceptual del enfoque propuesto para la arquitectura de software de las unidades de combate se ilustra en la siguiente Ilustración (13). (a) muestra un conjunto de modelado y artefactos de simulación y las relaciones entre ellos. (b) representa un estilo hacia la realización de un modelado y entorno de simulación. La idea de la arquitectura es la de tomar nociones dentro del dominio de M & S y asignándolos a las contrapartes construidas dentro del ámbito de software. Por ejemplo, el modelo en 1 (b) puede ser cualquiera de los diversos métodos de modelización y simulación alternativa (por ejemplo, la simulación paralela de modelos orientados a objetos).
Similarmente, la vista puede presentar textual y / o gráficamente el comportamiento de la simulación y el control puede ser una de las varias estrategias para mediar en la interacción entre la vista y el modelo.

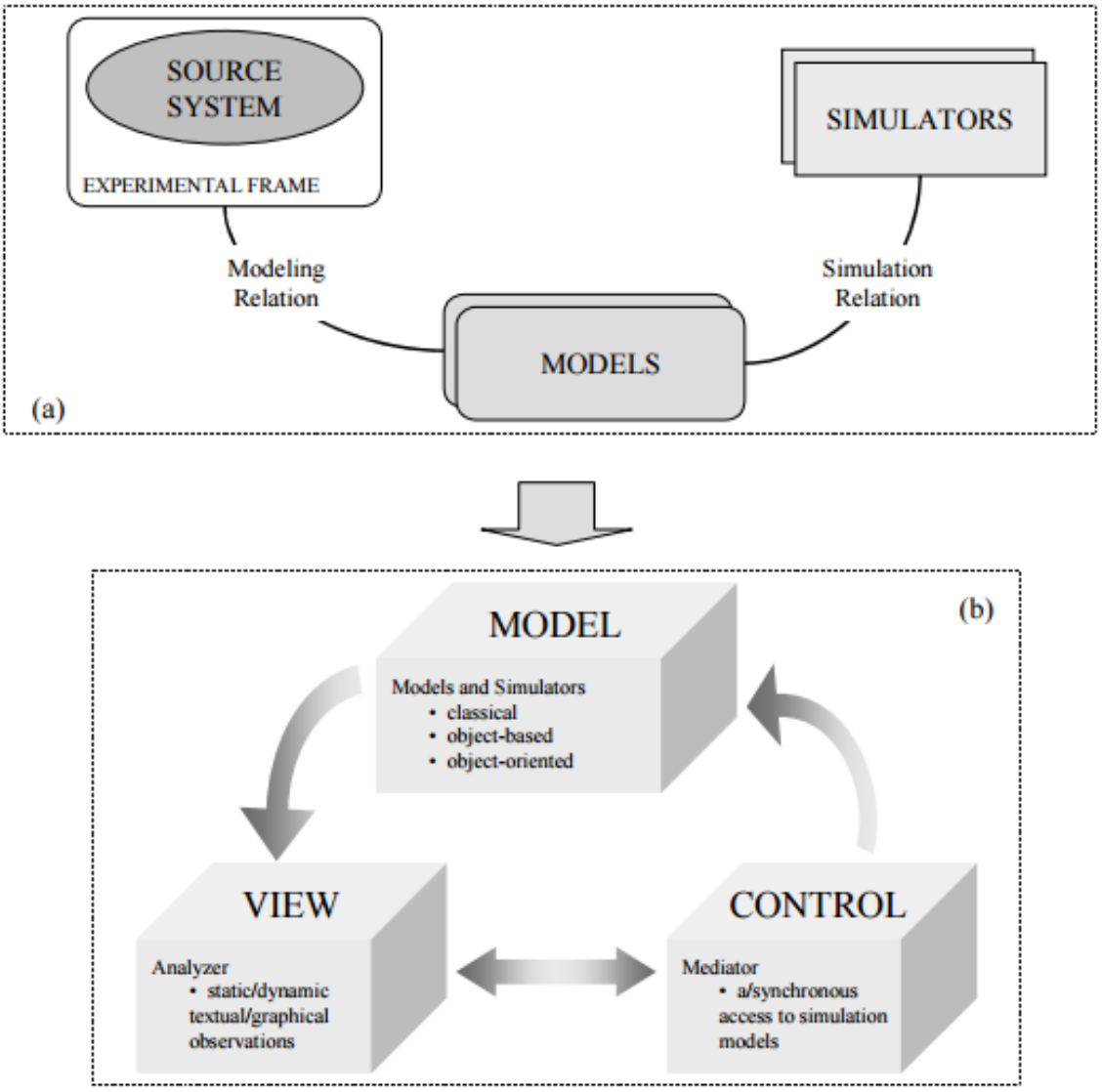


**Ilustración 13. Mapeo de un framework M&S aplicado a una arquitectura de software (a) Modelando y simulando entidades y relaciones; (b) concepto de Arquitectura de Software para ambientes M&S.**

(Singh & Sarjoughian, Software Architecture for Object-Oriented Simulation, 2003)

## 5.5. Análisis del patrón aplicado a la arquitectura

Después de un cuidadoso análisis de los objetivos del proyecto, se determinó que la mejor manera de estructurar el sistema era haciendo uso del "patrón de diseño[4]": *Model-View-Controller-MVC*.

#### *5.5.1.1. Uso del patrón*

Uno de los patrones de arquitectura que más se utilizan para el desarrollo de aplicaciones es el Modelo-Vista-Controlador (MVC) descrito anteriormente. MVC es un patrón que fue utilizado para construir las interfaces de usuario, siendo la aportación más importante de este patrón la separación de los componentes relacionados con los datos de la aplicación de los componentes de la interfaz de usuario. La separación de las capas permitirá tener, a nivel de desarrollo, un código más claro, flexible y reusable.

Al separar la presentación, los datos y la lógica de negocio se tiene una idea más clara de lo que necesita la aplicación, se desarrollan los componentes y se establecen las relaciones necesarias. La utilización de MVC permite tener menor acoplamiento, modificando solo las partes involucradas (se modifica solo lo que se necesita), siendo transparente para las demás. Por otra parte, al utilizar MVC se tiene la capacidad de representar la información de varias formas sin necesidad de modificar la fuente. (Mora, 2011)

## 5.6 Modelo de Vistas de la arquitectura

Esta arquitectura se diseñó en base a una visión general del modelo, teniendo en cuenta los requerimientos obtenidos de la recolección de información, estructurándose con el modelo de vistas 4+1. A través de estas vistas se presentarán los principales componentes y elementos relacionados que hacen parte de la arquitectura.

[4] Un *"patrón de diseño"* es una solución probada para un problema en un contexto

### 5.6.1 Vista de Escenarios

En la vista de escenarios se describirá en su totalidad la funcionalidad del aplicativo mediante el diagrama de casos de uso. El sistema basa su funcionalidad en 2 tipos de usuarios: el usuario o jugador y el administrador, cada uno de ellos puede consultar la información contenida en el aplicativo. El usuario, es cualquier usuario que desee ejecutar una nueva simulación, listar las unidades ya existentes. El administrador/desarrollador del aplicativo, puede realizar las funciones del visitante además de añadir o suprimir funcionalidades al sistema, es el encargado de revisar que todas las respuestas que da el sistema sean las adecuadas. En la ilustración 14 se muestra el diagrama de casos de uso del sistema.

## Diagrama de casos de uso aplicativo web NavalWarfare

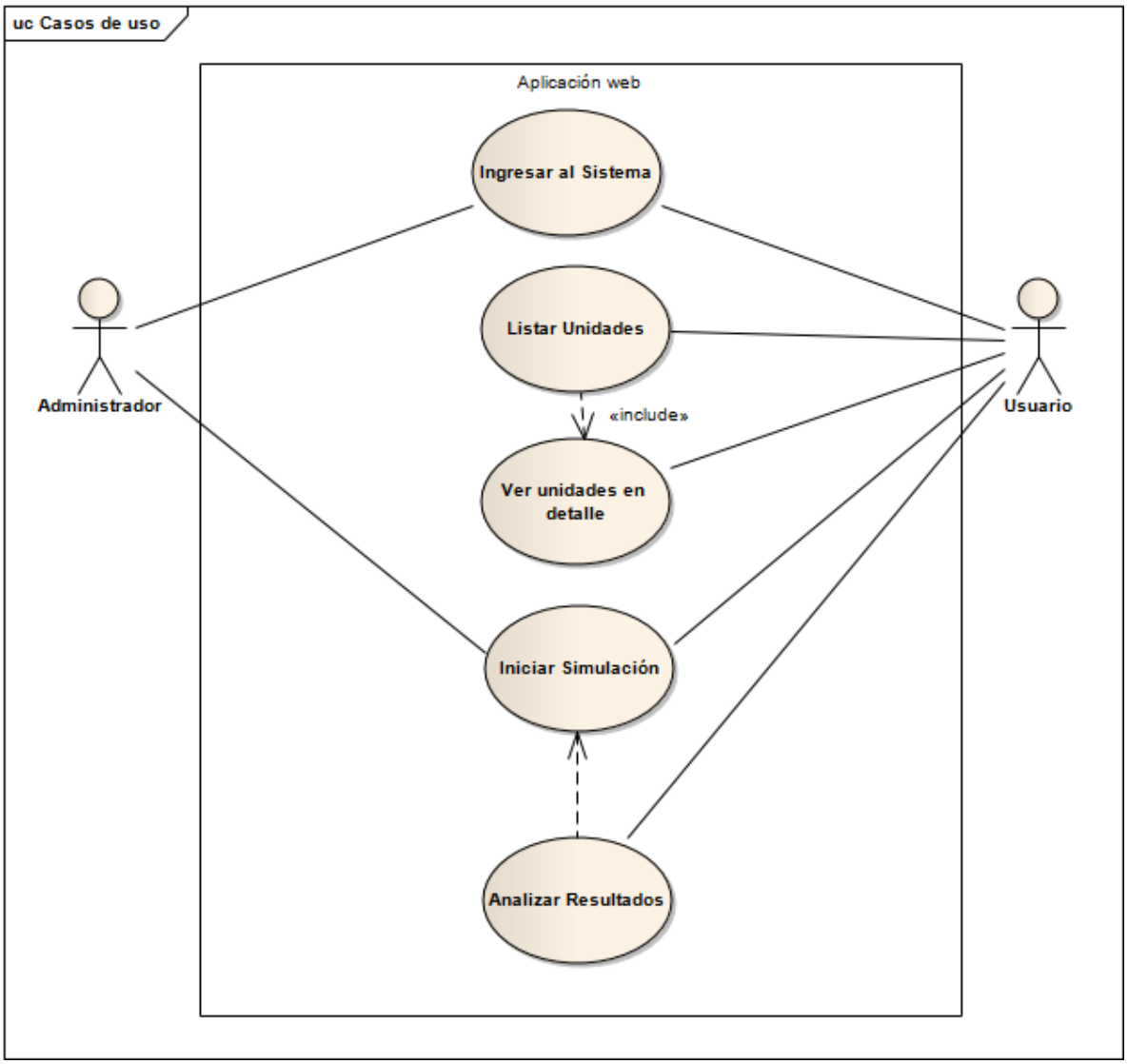


**Ilustración 14. Diagrama de casos de uso del sistema**

Especificación del caso de uso: CU1. Ingresar al Sistema

**Nombre:** Ingresar al sistema

**Descripción**: este caso de uso permite al administrador del sistema o al usuario entrar al sitio donde se encuentra alojada la aplicación web.

**Actor(es):** Administrador, Usuario

**Pre-condiciones:** Ninguna

**Post-condiciones:** Ver unidades o Iniciar simulación

**Flujo normal de eventos:** 1. El usuario digita en la barra de navegación del navegador la dirección web donde se encuentre alojada la aplicación

2. El sistema presenta la pantalla de ingreso al sistema.

3. El usuario selecciona la opción que desee se cargue en pantalla

4. El sistema redirecciona a la página correspondiente a la opción elegida por el usuario

5. Fin del flujo

**Manejo de situaciones excepcionales:** Este caso de uso no es funcional si la maquina no se tiene acceso a la web.

Especificación del caso de uso: CU2. Listar unidades

**Nombre:** Listar unidades

**Descripción**: este caso de uso permite al usuario mostrar una lista o catálogo de todas las unidades que se tienen registradas en el sistema.

**Actor(es):** Usuario

**Pre-condiciones:** Iniciar el sistema

**Post-condiciones:** Ninguna

**Flujo normal de eventos:** 1. El usuario indica al sistema que desea listar las unidades

2. El sistema redirecciona a la página donde se describen específicamente todas las unidades registradas.

3. El usuario tiene la opción de ver las unidades en detalle

4. Fin del flujo

Especificación de caso de uso: CU3. Ver unidades en detalle

**Nombre:** Ver unidades en detalle

**Descripción**: este caso de uso permite al usuario ver cada unidad aero-naval con todas sus especificaciones, además de una ilustración gráfica de cómo es dicha unidad.

**Actor(es):** Usuario

**Pre-condiciones:** Listar unidades

**Post-condiciones:** Ninguna

**Flujo normal de eventos:** 1. El usuario indica al sistema que desea listar las unidades

2. El sistema redirecciona a la página donde se describen específicamente todas las unidades registradas.

3. El usuario elige la unidad que desee ver en detalle

4. El sistema redirecciona a la página de la unidad y sus características

5. Fin del flujo

Especificación de caso de uso: CU4. Iniciar Simulación

**Nombre:** Iniciar Simulación

**Descripción**: este caso de uso permite al usuario simular un escenario aleatorio donde se expone el enfrentamiento entre dos unidades con condiciones climáticas específicas, y tomando en cuenta cada una de las características de las unidades en el enfrentamiento.

**Actor(es):** Administrador**,** Usuario

**Pre-condiciones:** Iniciar el sistema.

**Post-condiciones:** Ninguna.

**Flujo normal de eventos:** 1. El usuario selecciona la opción "New Game" que desee se carga en pantalla de inicio

2. El sistema genera un ambiente aleatorio con condiciones climáticas que afectan el enfrentamiento entre dos unidades de combate.

3. Resulta un vencedor y se visualizan los resultados del enfrentamiento

4. Fin del flujo.

Especificación de caso de uso: CU5. Visualizar Resultados

**Nombre:** Visualizar resultados

**Descripción**: este caso de uso permite al usuario tener en pantalla todos los resultados del enfrentamiento de las unidades.

**Actor(es):** Usuario, Administrador.

**Pre-condiciones:** Iniciar una simulación.

**Post-condiciones:** Ninguna

**Flujo normal de eventos:** 1. El usuario selecciona la opción “New Game” que desee se carga en pantalla de inicio

2. El sistema genera un ambiente aleatorio con condiciones climáticas que afectan el enfrentamiento entre dos unidades de combate

3. El sistema despliega en pantalla los resultados del combate entre las unidades

4. El sistema da la opción al usuario de recrear un nuevo combate (simulación)

5. Fin del flujo

### 5.6.2 Vista Lógica

Esta vista se encarga de responder a requerimientos funcionales del sistema, dividiéndolo en un conglomerado de 3 subsistemas (ver Ilustración 15):

- El primero, conocido como ***Modelo***, contiene las clases lógicas del sistema.
- El segundo subsistema es ***Procesos,*** que gestiona todos los procesos que se ejecutan para llevar a cabo la simulación. Este subsistema contiene en si subprocesos necesarios para la recreación del combate entre unidades con la ejecución de los archivos *game.js* y *gamesave.js*.
- El tercer subsistema ***Vistas***, contiene todas las vistas del sistema que se muestran al usuario y adicionalmente, se encuentra el subsistema ***Unknown Web Process***, que es el encargado de generar las vistas web que facilitan el ofrecimiento del simulador como servicio en la nube.
- Los subsistemas adicionales portan los archivos configurables generados para cargar y generar la simulación.

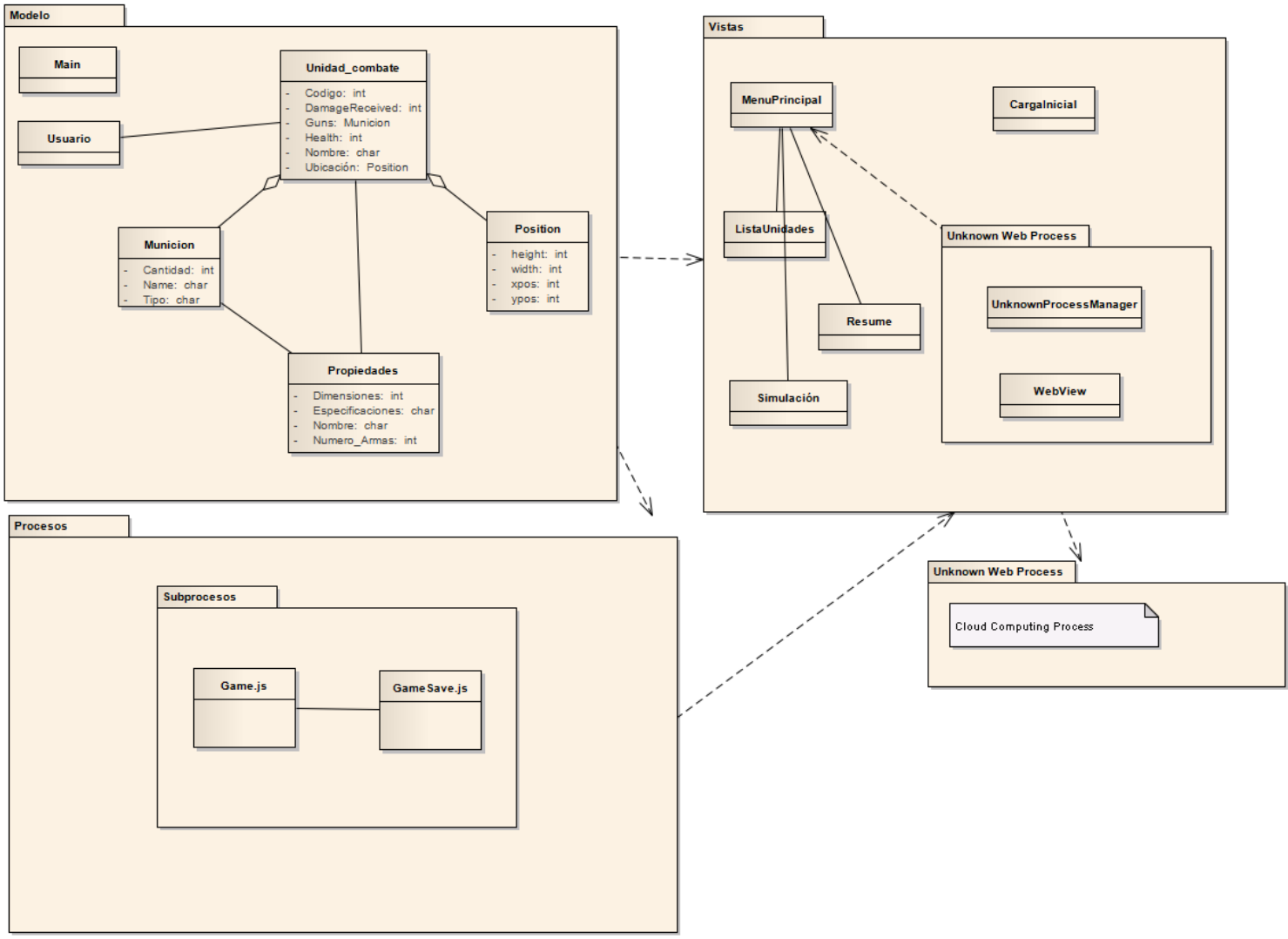


**Ilustración 15. Vista Lógica de la arquitectura del sistema**

A continuación se presentan los diagramas de secuencia pertinentes a los casos de uso presentados:

Diagrama de secuencia: CU Ingresar al sistema

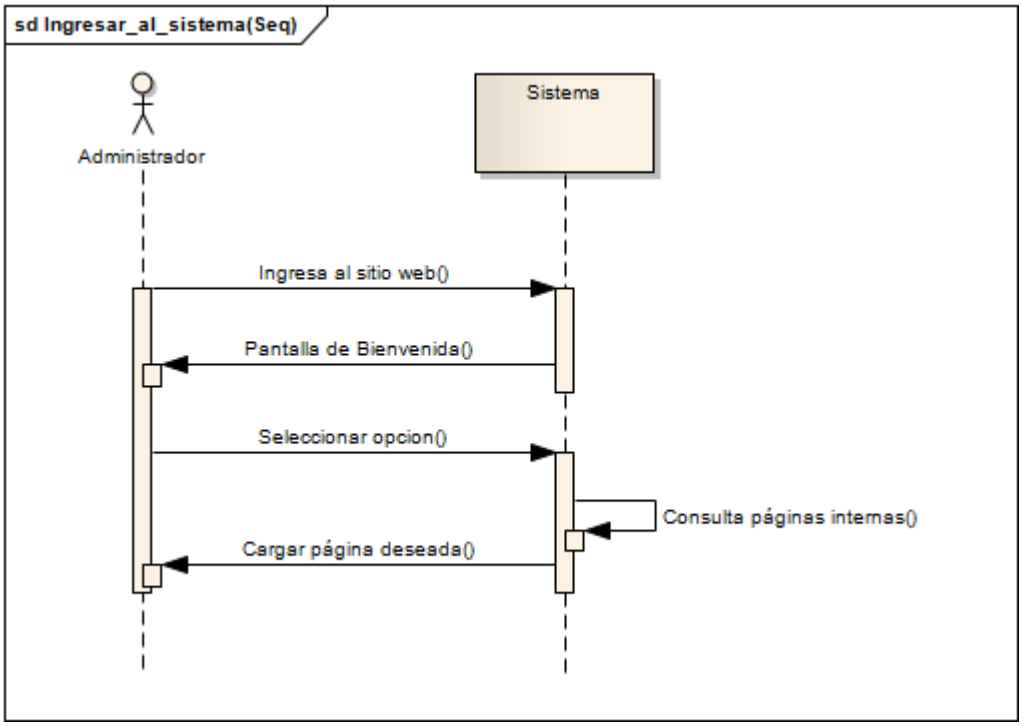


**Ilustración 16. Diagrama de Secuencia CU: Ingresar al Sistema**

Diagrama de secuencia: CU Listar unidades

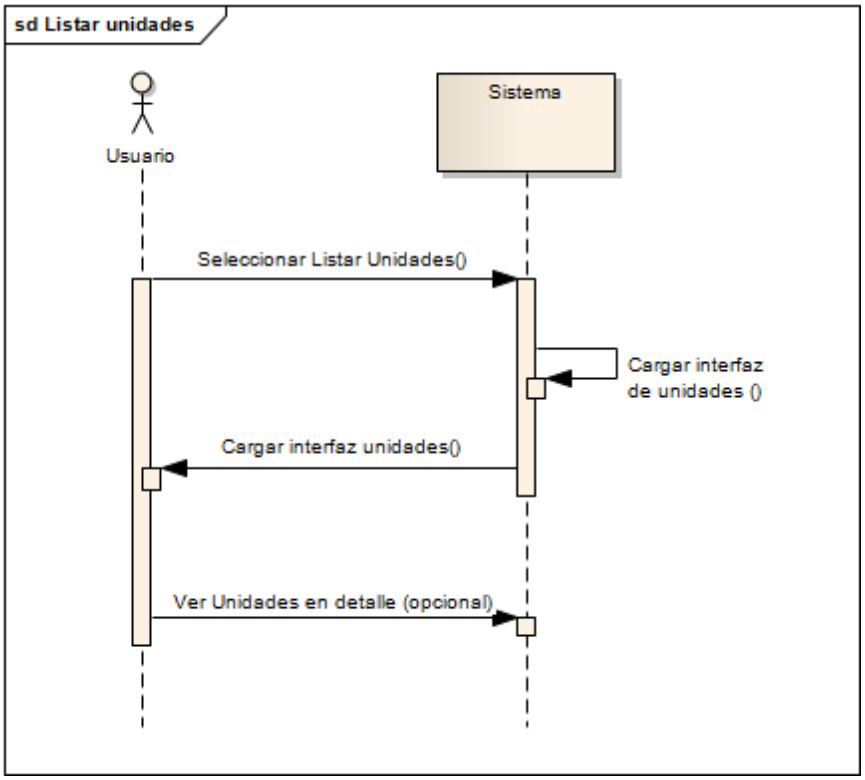


**Ilustración 17. Diagrama de Secuencia CU: Listar Unidades**

Diagrama de secuencia: CU Ver unidades en detalle

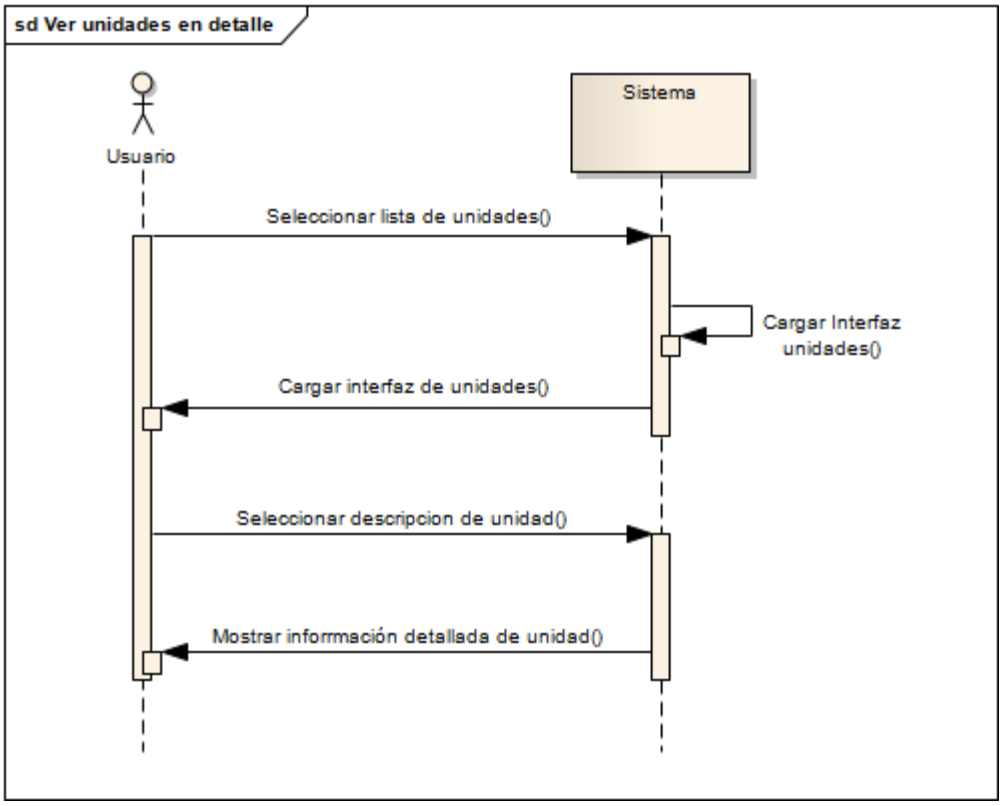


**Ilustración 18. Diagrama de Secuencia CU: Ver unidades en detalle**

Diagrama de secuencia: CU Iniciar Simulación

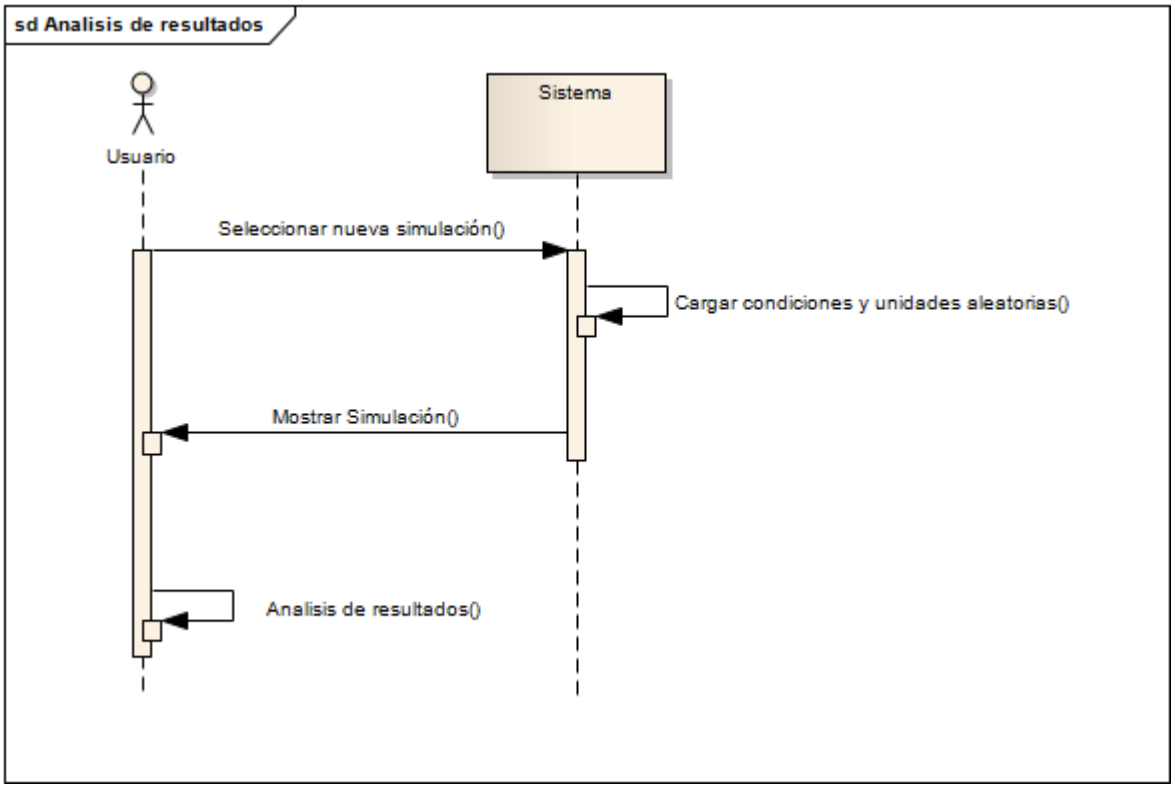


**Ilustración 19. Diagrama de Secuencia CU: Iniciar simulación**

Diagrama de secuencia: CU Analizar Resultados

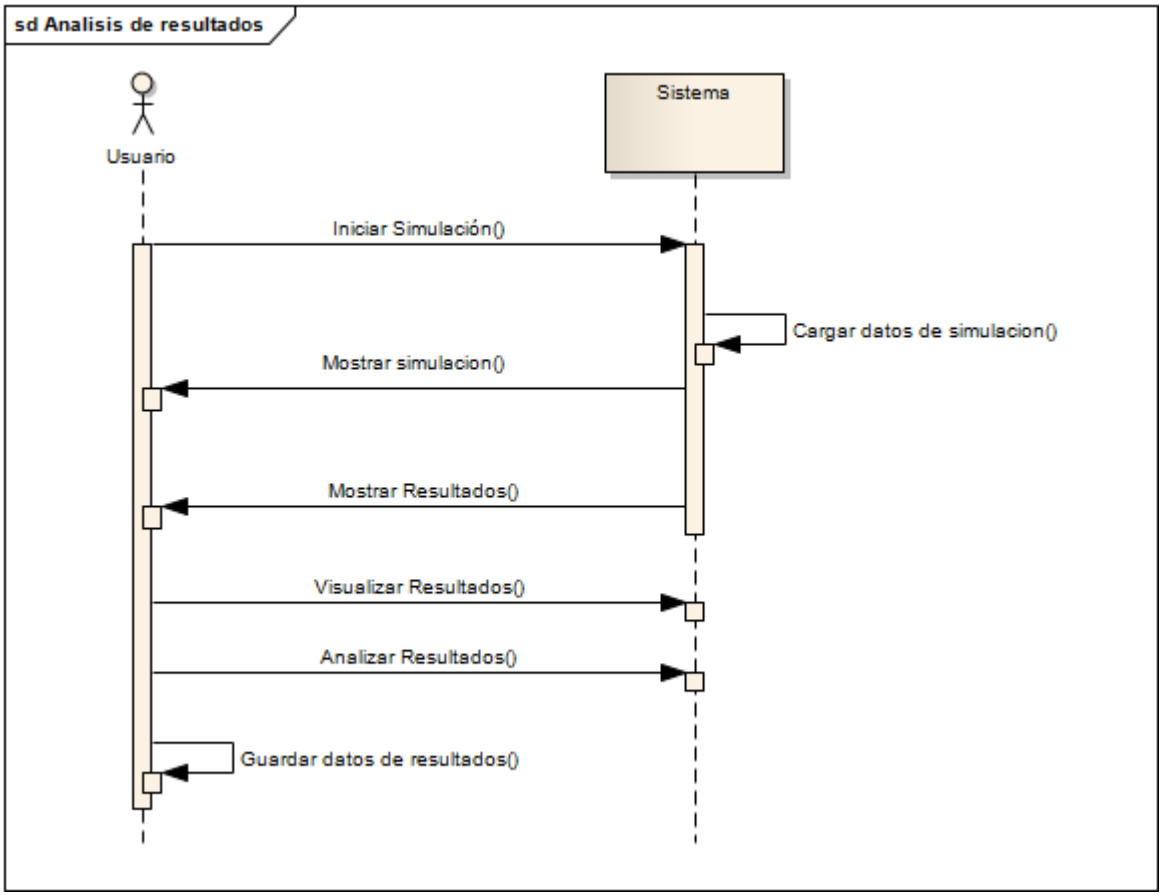


**Ilustración 20. Diagrama de Secuencia CU: Analizar Resultados**

### 5.6.3 Vista de Procesos

En esta vista se presentan los procesos que hay en el sistema y la forma en la que se comunican estos procesos; es decir, se representa desde la perspectiva de un integrador de sistemas, el flujo de trabajo paso a paso de negocio y operacionales de los componentes que conforman el sistema. Para completar la documentación de esta vista se muestra a continuación el diagrama de actividades del sistema

Diagrama de actividad: CU1. Ingresar al sistema

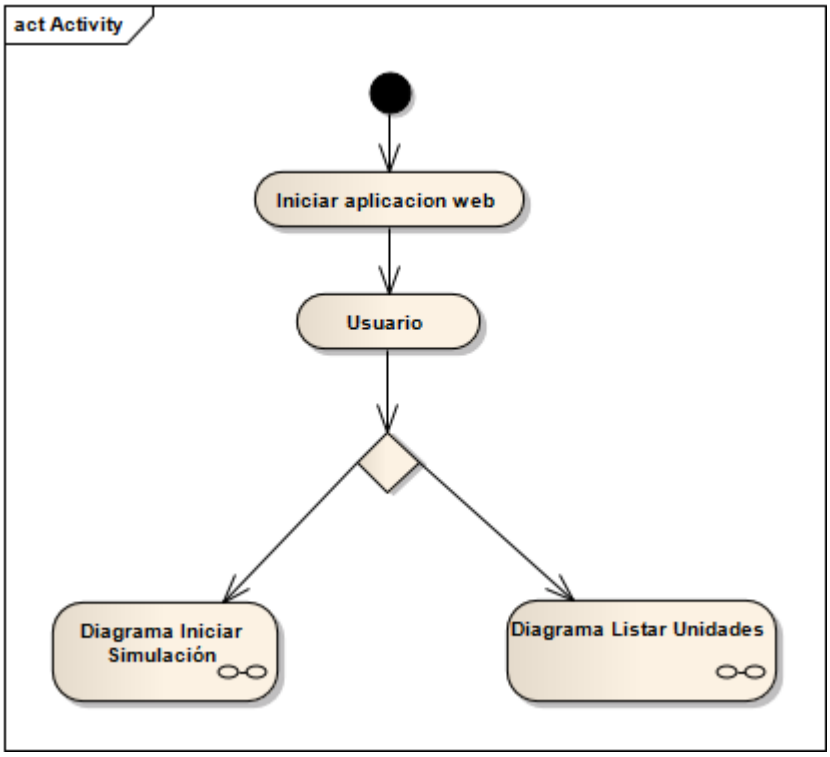


**Ilustración 21. Diagrama de Actividad. CU: Ingresar al sistema**

Diagrama de actividad: CU2. Listar Unidades

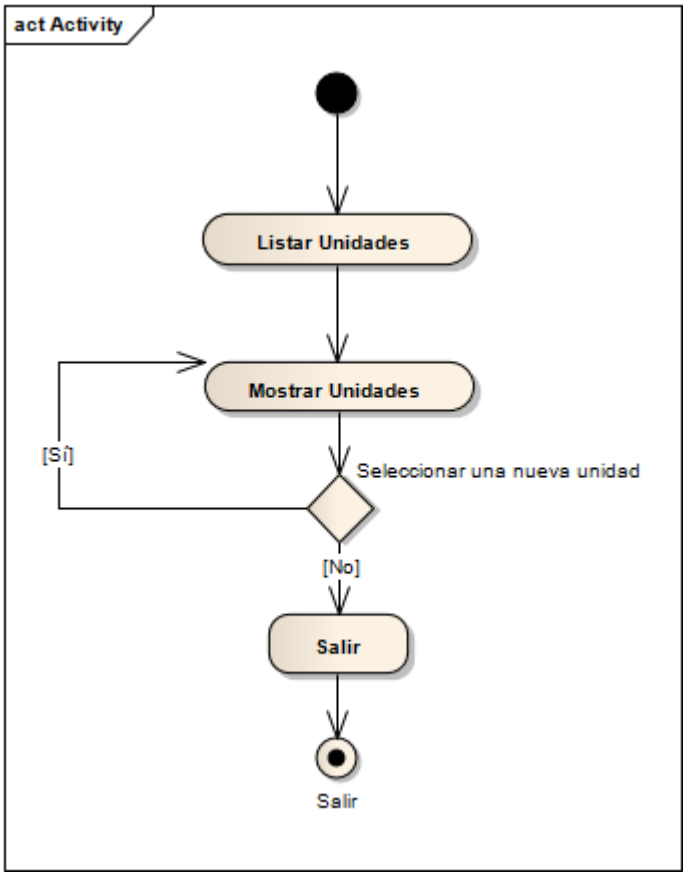


**Ilustración 22. Diagrama de Actividad. CU: Listar unidades**

Diagrama de actividad: CU2. Ver Unidades en detalle

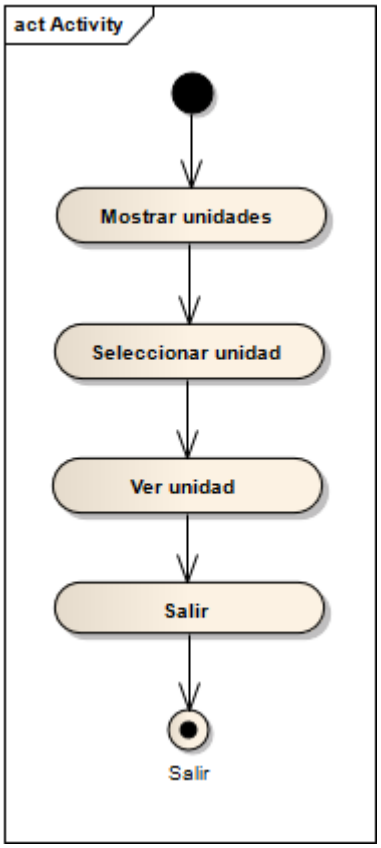


**Ilustración 23. Diagrama de Actividad. CU: Ver unidades en detalle**

Diagrama de actividad: CU4. Iniciar Simulación.

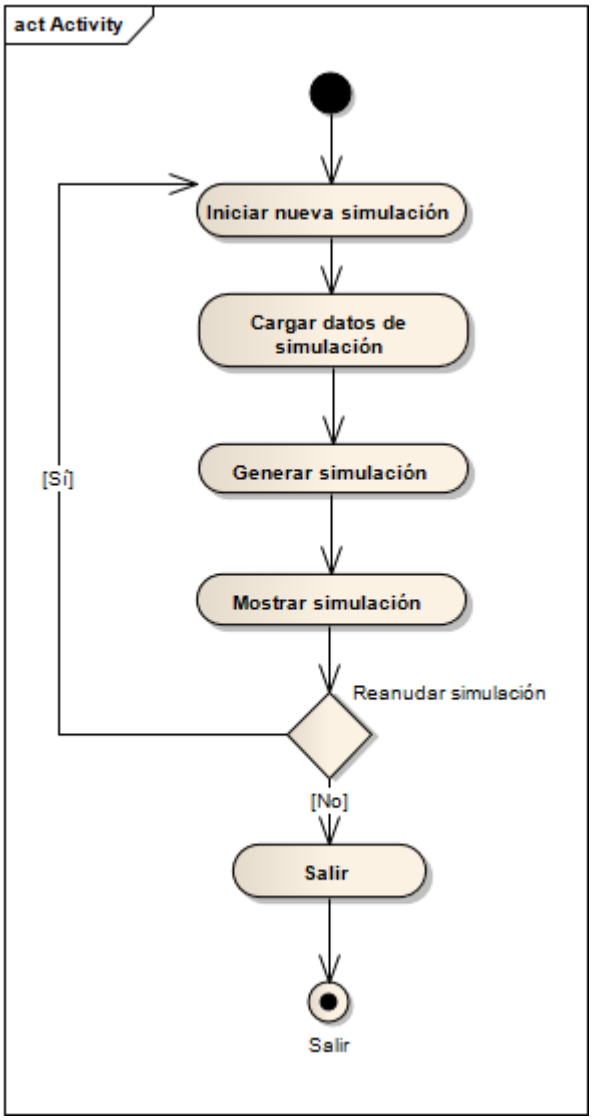


**Ilustración 24. Diagrama de Actividad. CU: Iniciar simulación**

Diagrama de actividad: CU5 Análisis de resultados.

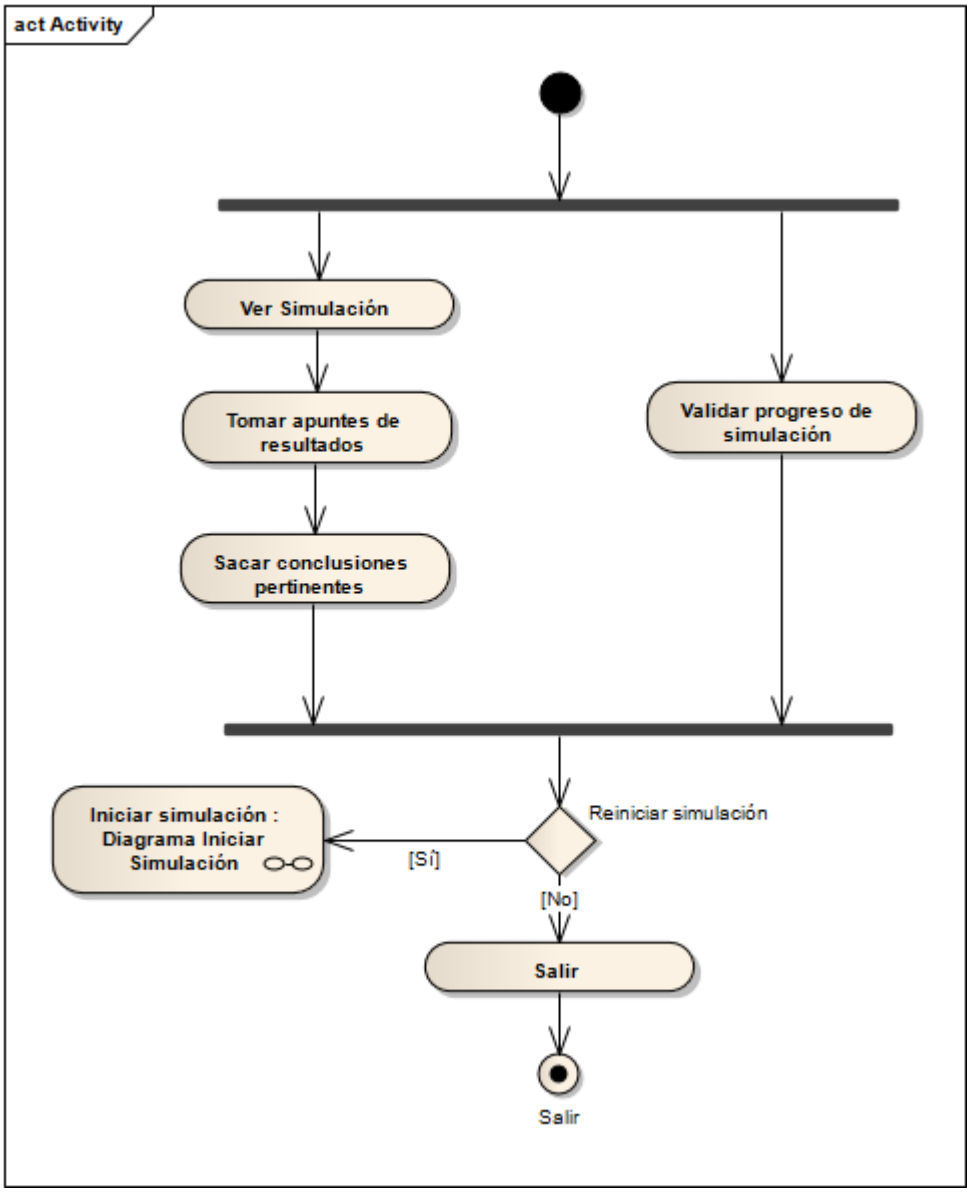


**Ilustración 25. Diagrama de Actividad. CU: Analizar resultados**

### 5.6.4 Vista de Desarrollo

Esta vista muestra los elementos o componentes que integran el sistema. En la Ilustración 26, se puede apreciar los dos 2 elementos principales que interactúan en el sistema base, que son la ***InternalStructure*** y ***Connection.***

- ***InternalStructure***, es el componente donde se ejecuta la aplicación, en ella se pueden apreciar los elementos de ***Game, gameSave, Index, Menus, Vista***; el primero es el encargado de manejar todo lo referente a la simulación y sus resultados; el segundo controla las archivos de guardado de cada simulación; el tercero maneja la página de inicio y todos los componentes que se inician al ejecutar la aplicación; por último la vista controla todos los componentes visuales que se genera en el prototipo.
- ***Connections***, en este componente se ejecuta todo lo referente a la conexión entre la página web y la plataforma que posteriormente ofrecerá el prototipo como servicio en la nube, este componente está enfocado en mostrar el aplicativo como servicio en cloud computing.

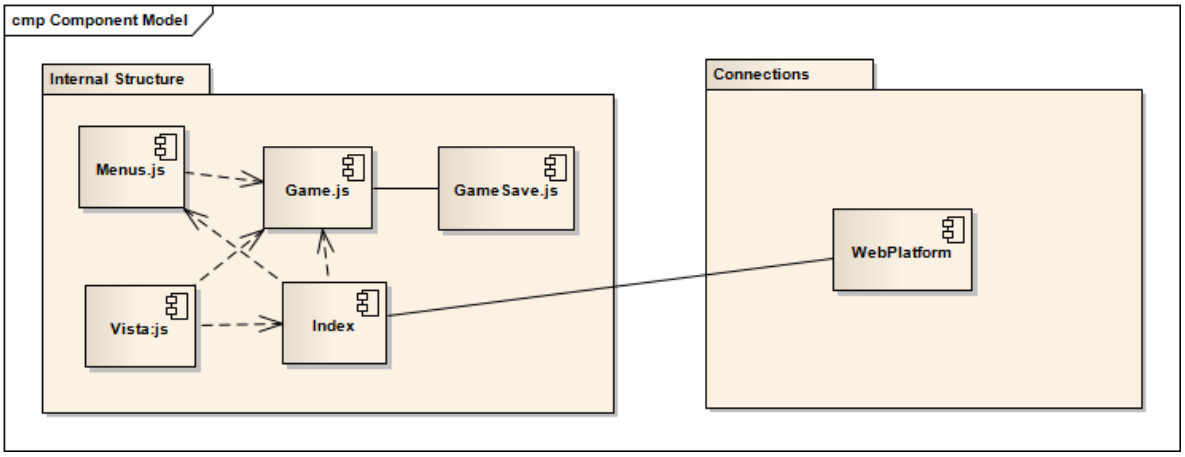


**Ilustración 26. Vista de Desarrollo de la arquitectura del sistema**

### 5.6.5 Vista Física

Esta vista se encarga de describir la configuración del sistema para su ejecución en un ambiente del mundo real. La vista está compuesta por dos subsistemas, el servidor principal donde está contenida la lógica del negocio (aplicativo web) y el servidor web; por otra

parte, se tiene la estación de trabajo que en pocas palabras es la máquina donde se ejecuta el navegador. La Ilustración 27 muestra la representación física del sistema y cómo está desplegado en un diagrama de despliegue.

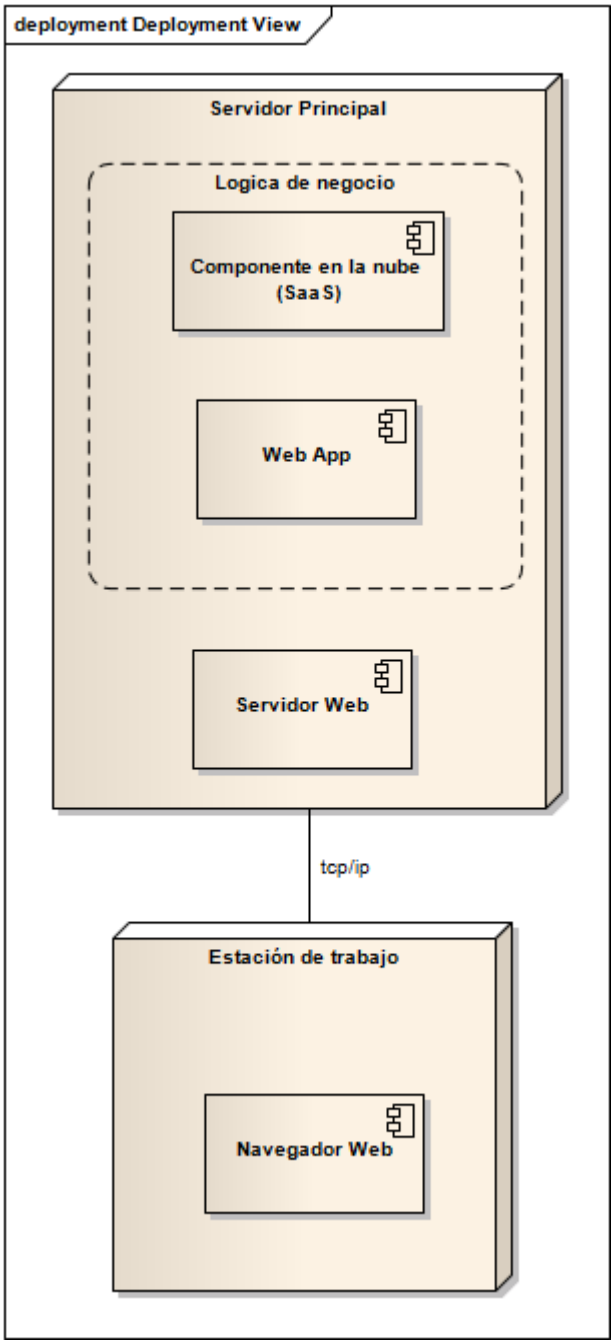


**Ilustración 27. Vista Física: Diagrama de despliegue de la arquitectura del sistema**

# 6. VERIFICACIÓN Y VALIDACICÓN DE LA ARQUITECTURA

Para la verificación y validación de la arquitectura planteada, se desarrolló un aplicativo web sobre el cual se creó un escenario que permitiera hacer realidad algunas características de dicha arquitectura. Este escenario, está desarrollado a partir de un código libre de simuladores realizado en el lenguaje JavaScript; siendo así, se adaptó todo para que cumpliera con los objetivos de la arquitectura, manteniendo el enfoque del proyecto.

El desarrollo del aplicativo web se realizó a través de la metodología RUP, basados en los siguientes diagramas:

## *6.1. Diagrama Jerárquico en la Simulación de Objetos De Clase*

En este diagrama (ver Ilustración 28) se pueden apreciar los subsistemas ***SimulationObject, Scenario, ElementCombat, Firing Transaction***; el subsistema ***SimulationObject***, es el referente de todas las clases contenidas en el sistema, ya que, sobre él se simulan todos los objetos que interactúan durante la simulación, además de la recreación de vistas para su posterior ejecución. Las clases dependientes de esta muestran los elementos de combate o unidades (objeto fundamental de la arquitectura) y la interacción de daños entre ellas está contenida bajo las clases provenientes de Firing Transaction. Por último las clases precedidas por el acrónimo WM_ , indican *Web Methods* utilizados para el ofrecimiento del aplicativo como un servicio en la nube.

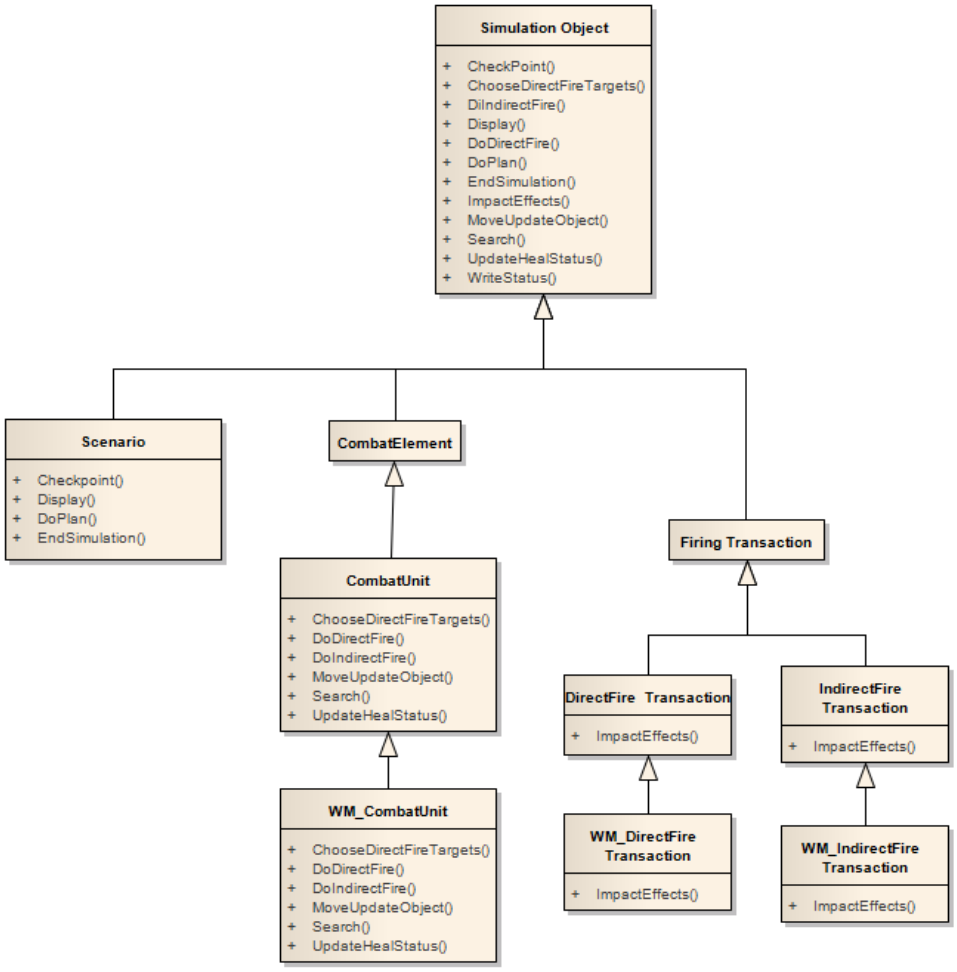


**Ilustración 28. Diagrama de clases con jerarquía en la simulación de objetos de clase**

### *6.2. Diagrama de Elementos principales en la Arquitectura*

El diagrama de elementos principales de alto nivel que interactúan en la arquitectura (Ilustración 29) muestra la interacción de cada uno de los componentes primordiales para generar la simulación. Descompone el sistema para mostrar las fuerzas (de las unidades de combate) que contienden en un escenario dado, teniendo en cuenta su armamento, el entorno que los afecta y demás factores esenciales. En los puntos *comando & control y Símbolos del mapa* se tomó en cuenta que dentro de cada unidad existe un lenguaje común para este tipo de elementos.

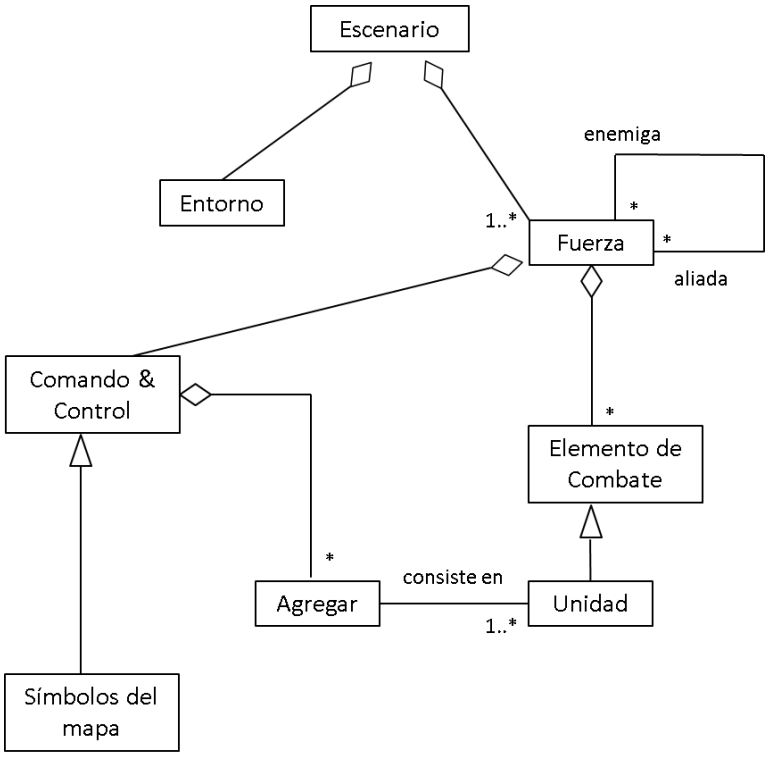


**Ilustración 29. Diagrama de elementos principales de alto nivel en la arquitectura**

### *6.3. Modelo de dominio y diagrama de capas del sistema*

A continuación en las ilustraciones 30 y 31 se presentan el modelo del dominio del sistema y la arquitectura basada en capas respectivamente. El primero indica de forma general las funcionalidades del sistema basada en las entidades de combate proporcionadas por la ENAP. La arquitectura basada en capas muestra de forma abstracta las 3 capas según las cuales fue desarrollada la arquitectura, teniendo así en el nivel de aplicaciones los paquetes de dominio donde se encierran los sistemas de combate, manejo de escenario (no se presenta en la vista ya que esto pertenece a otro tema de investigación), y la simulación de combate. Además, los servicios donde se almacena la información detallada de las unidades y por último la base de datos como capa de almacenamiento, junto con DIS/HLA (Distributed Interactive Simulation & High Level Architecture).

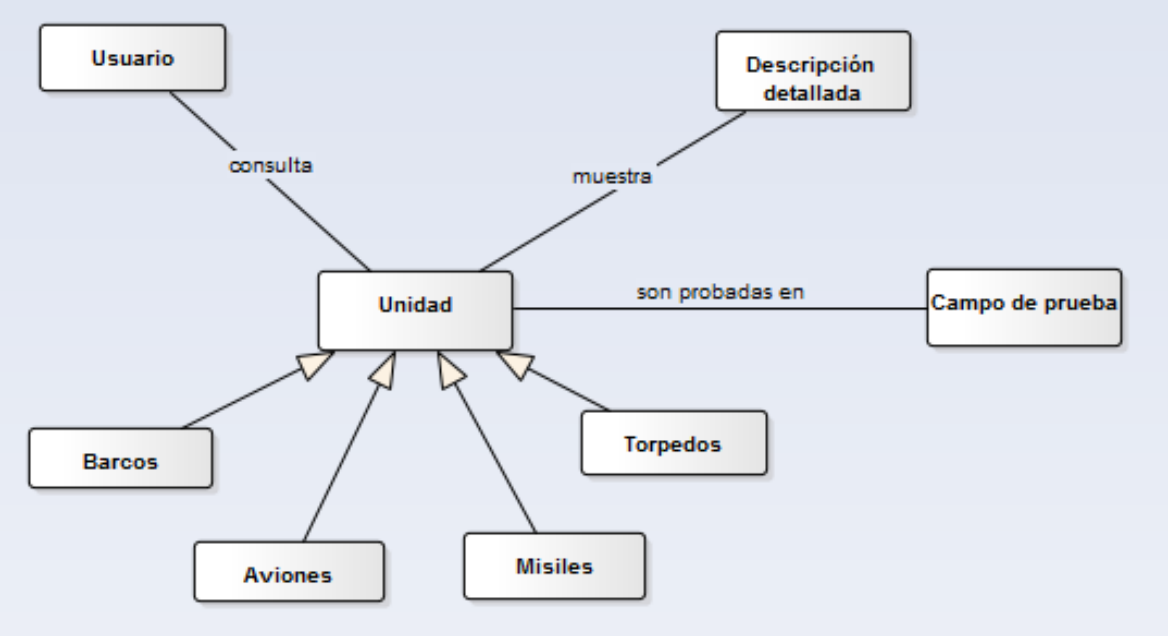


**Ilustración 30. Modelo de dominio de la arquitectura**

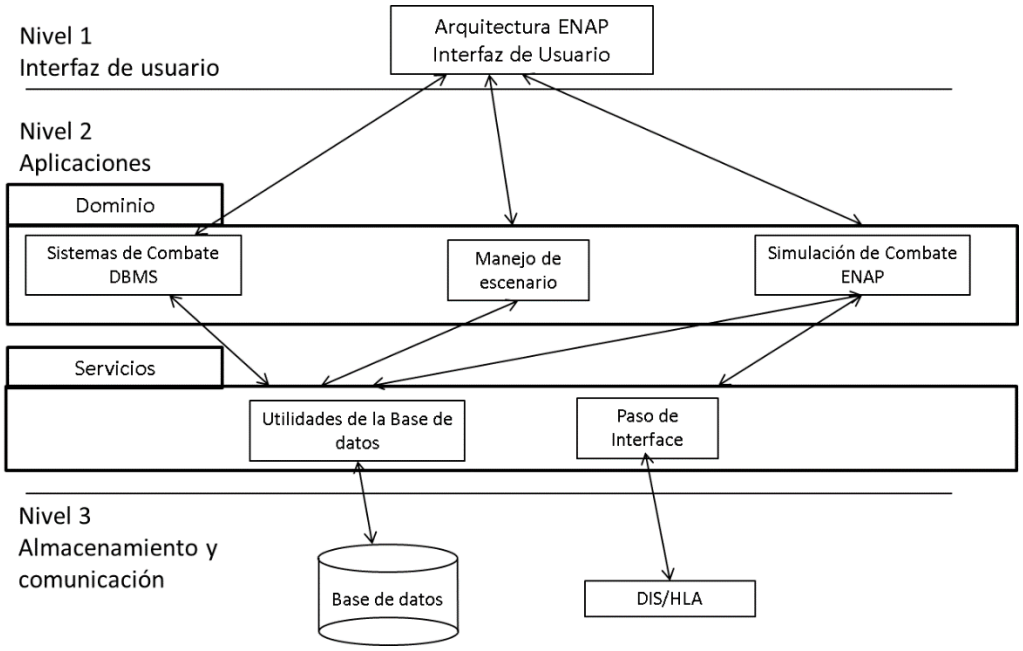


**Ilustración 31. Arquitectura mostrada en capas**

## *6.4. Ejecución del aplicativo web Naval Warfare Simulator*

Con base en la arquitectura planteada, se codificó una aplicación web, la cual fue denominada bajo el nombre de "Naval Warfare Simulator" como referencia para efectos de estética al momento de ser entregada al usuario final.

La página principal de la aplicación (index) (ver Ilustración 32) después de ingresar a ella, muestra el nombre dado más 3 opciones: ***New Game***, ***List Ships*** y ***Resume***


.

**Ilustración 32. Página principal del prototipo**

Al presionar ***New Game***, comienza la simulación de combate entre 2 unidades aleatorias (ver Ilustración 33), se debe tener en cuenta que cada circunferencia representa una unidad de combate, en la parte superior se visualizan las barras de "salud" o vida total de las unidades en cuestión, en lo que se refiere a la unidad propia se muestra la recarga de submerged, guns y torpedos utilizados que no son más que las municiones para el enfrentamiento, además existen weapon slots que son determinados dependiendo la unidad. En dichos weapons slots se muestran las características específicas de las armas que contiene la nave. En la interfaz Ship Options se visualiza el código reservado para la unidad en uso, y además existe un panel de simulación donde se determinan algunos factores que inciden dentro de la simulación tales como las barras de salud, viento, tiempo de lluvia, etcétera.

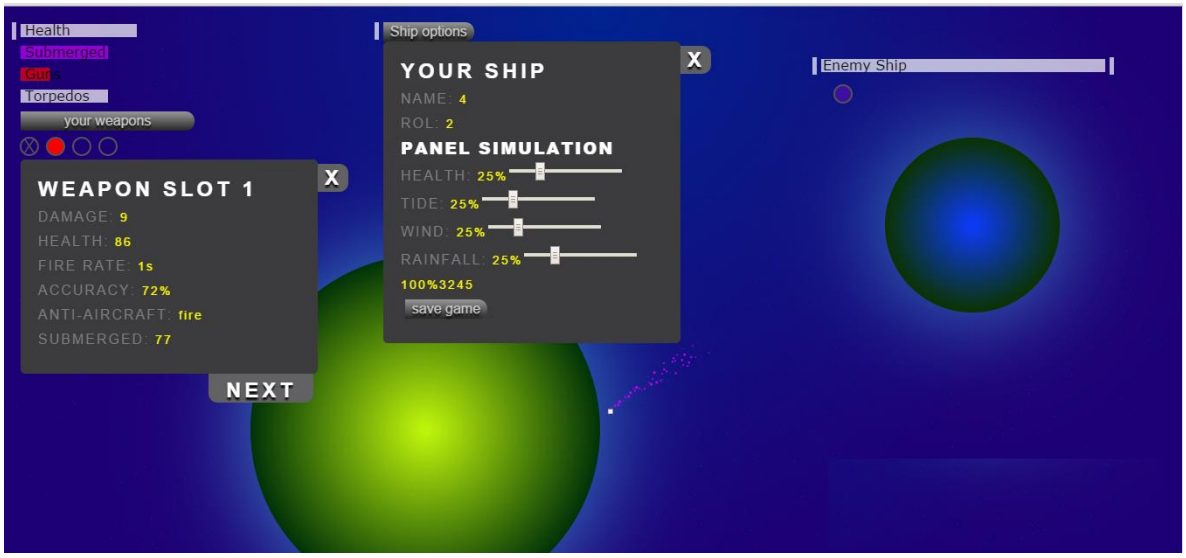


**Ilustración 33. Interfaz de ejecución de simulación en el prototipo**

Al finalizar la simulación el usuario puede resultar como ganador o perdedor de la contienda mostrando así un cuadro de dialogo diciendo que su unidad ha sido destruida y si desea reiniciar la simulación (ver Ilustración 34), o en el caso de conseguir la victoria en el enfrentamiento aparecerá una nueva interfaz con 3 opciones: ***Destroy, Peace y Takeover*** *(ver* Ilustración *35)*. Cuando se selecciona una de estas opciones existen cambios en la unidad propia o del enemigo, es decir, si se escoge la opción *destroy* automáticamente destruirá la unidad enemiga y se reiniciara la simulación con una nueva unidad contrincante; si se escoge la opción *peace* se oficializará una tregua con la unidad enemiga, dejandola escapar y dando lugar a una nueva contienda con un nuevo enemigo; por último, si escoge *takeover* se realizará un traslado hacia la unidad enemiga, invadiéndola y haciéndola propia para realizar una nueva simulación.

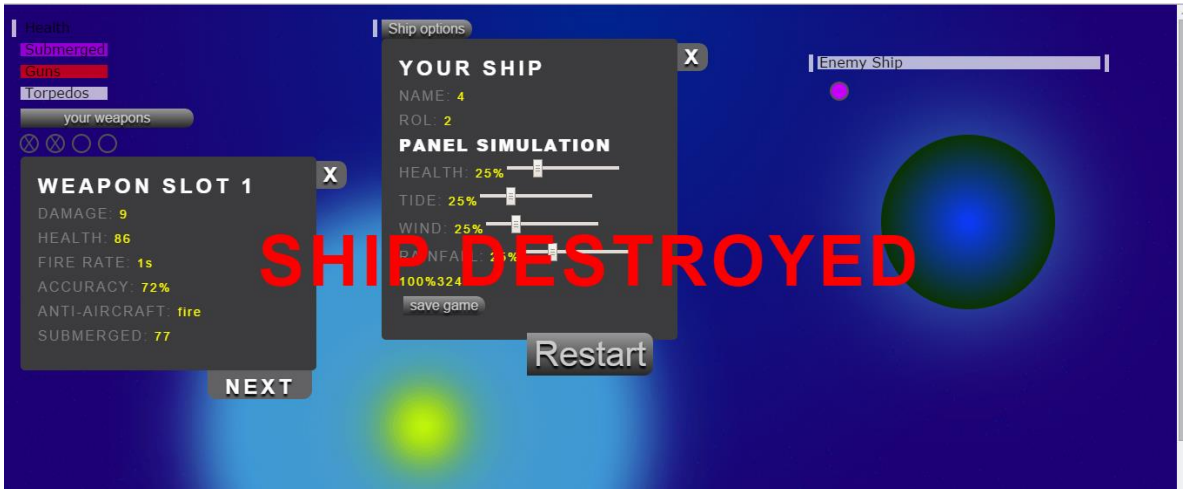


**Ilustración 34. Interfaz cuando se pierde la contienda al realizar la simulación.**

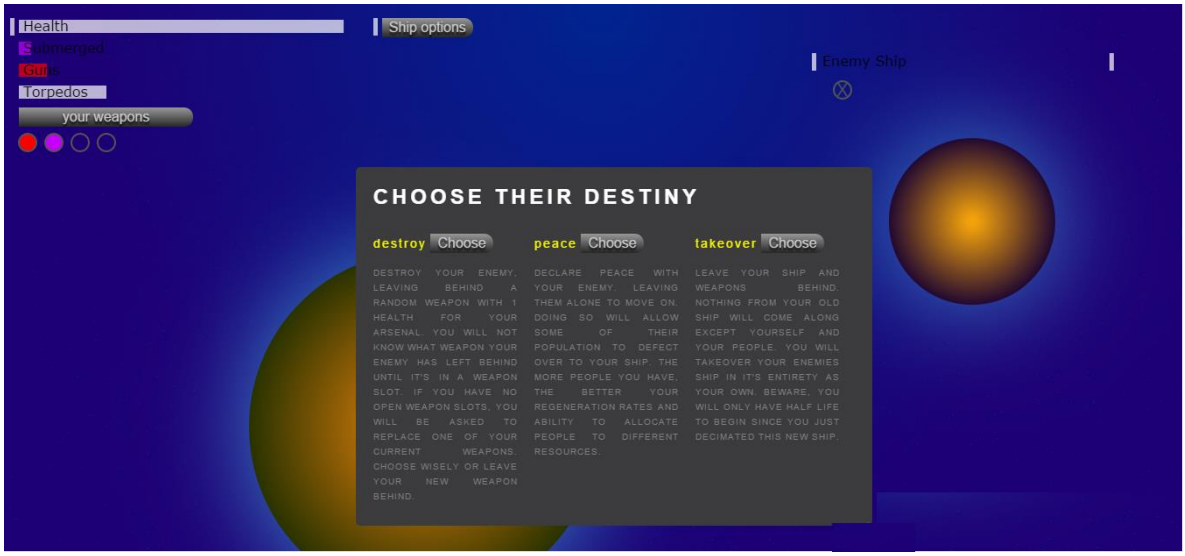


**Ilustración 35. Interfaz cuando se consigue la victoria sobre la unidad enemiga.**

### 6.4.1 Ejemplo práctico de uso del simulador

Este ejemplo está diseñado para orientar al cadete sobre el uso de la interfaz del Naval Warfare Simulator.

Este manual pretende que El cadete conozca la manera correcta de introducir las órdenes que el simulador ejecutará; que identifique los menús que se presentan en el simulador y que sepa qué tipo de datos espera la interfaz, en cada una de sus órdenes.

Se explican los tipos de objetos que puede encontrar en la Interfaz del Sistema, de tal manera que el usuario del Naval Warfare pueda identificarlos y manejarlos adecuadamente durante el desarrollo de un combate simulado entre dos unidades.

El usuario no requiere de conocimientos previos en el área computacional para poder manejar el Simulador.

Pantalla Inicial

La Pantalla Principal, véase la Ilustración 36, permite al usuario ver en su totalidad el funcionamiento del software. Es necesario ver esta pantalla antes de iniciar un Ejercicio o Juego,. en ella se encuentra todo lo que el comandante necesita para conducir a sus unidades durante la simulación de un Juego de Guerra.


**Ilustración 36. Interfaz de inicio del aplicativo.**

A continuación se explica detalladamente la funcionalidad de cada uno de los botones de la pantalla principal:

**1. New Game**

***Unidades de Combate***

El botón ***New Game*** señalado en la ilustración anterior con el número 1, da comienzo a la simulación del combate entre 2 unidades aleatorias (ver Ilustración 37), para comenzar se debe tener en cuenta que cada circunferencia (esfera) representa una unidad de combate que aparece al azar. A continuación se ve la pantalla de juego señalando las unidades.

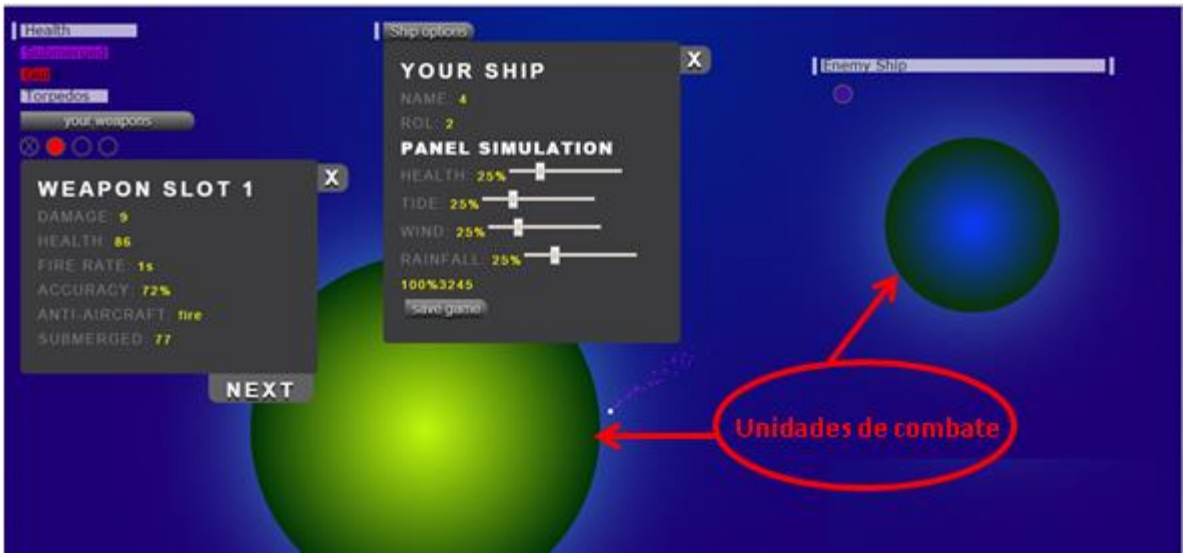


**Ilustración 37. Interfaz presentada al usuario cuando se inicia una nueva partida.**

**Barra de *salud, Inmersión, Armas y Torpedos***

En la Ilustración 37 se observan en la parte superior-izquierda y superior-derecha, las barras de "salud" y vida (Health) de cada una de las unidades. En lo que se refiere a la unidad asignada al usuario del software, en este caso la esfera de color verde, se muestran la barra de inmersión (submerged), armamento de la unidad (Guns) y torpedos (Torpedos), estos dos últimos utilizados para atacar a la unidad enemiga en enfrentamiento. A continuación se detalla en la Ilustración 38 lo descrito anteriormente

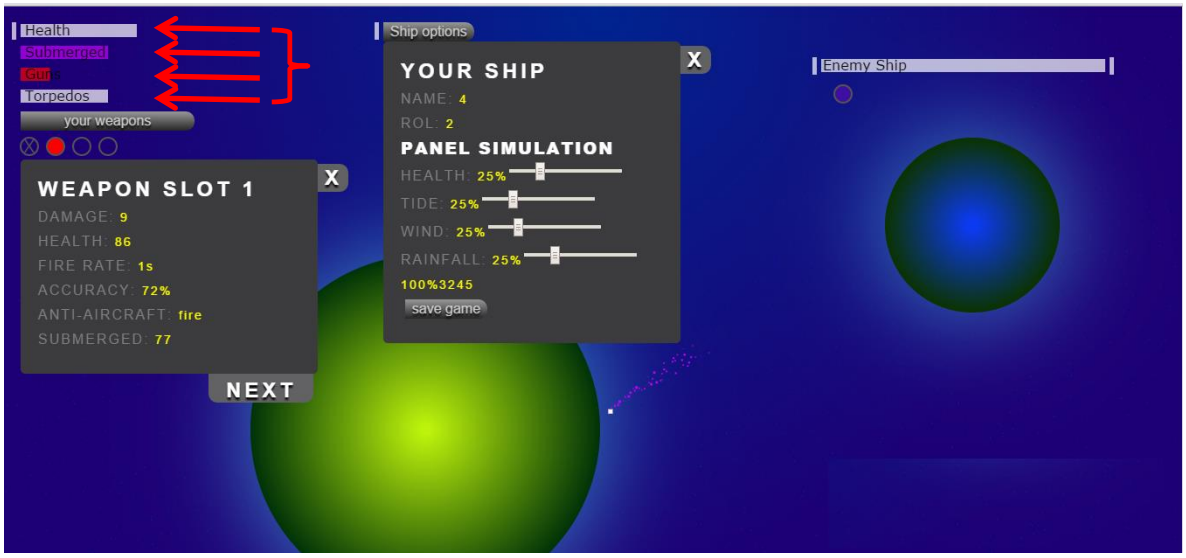


**Ilustración 38. Interfaz donde se indican las barras de salud, inmersión, torpedos y armas.**

### Panel *your weapons*

En la Ilustración 39 se resalta en la parte izquierda el panel de armas pertenecientes a la unidad escogida aleatoriamente. En dicho panel se muestran las características específicas de las armas (1) que contiene la nave, como por ejemplo el usuario podrá saber el daño (damage), salud o durabilidad del arma (health), frecuencia de disparo (fire rate), precisión (accuracy), anti-aereo (anti-aircraft) y anti-submarinos (submerged), cada uno de estos son aspectos relevantes para la formación de un cadete. Además los círculos coloreados debajo de "your weapons" (2) muestra las armas en cada uno de los slots de la nave, permitiendo navegar entre las características de estas con el botón PREV y NEXT.

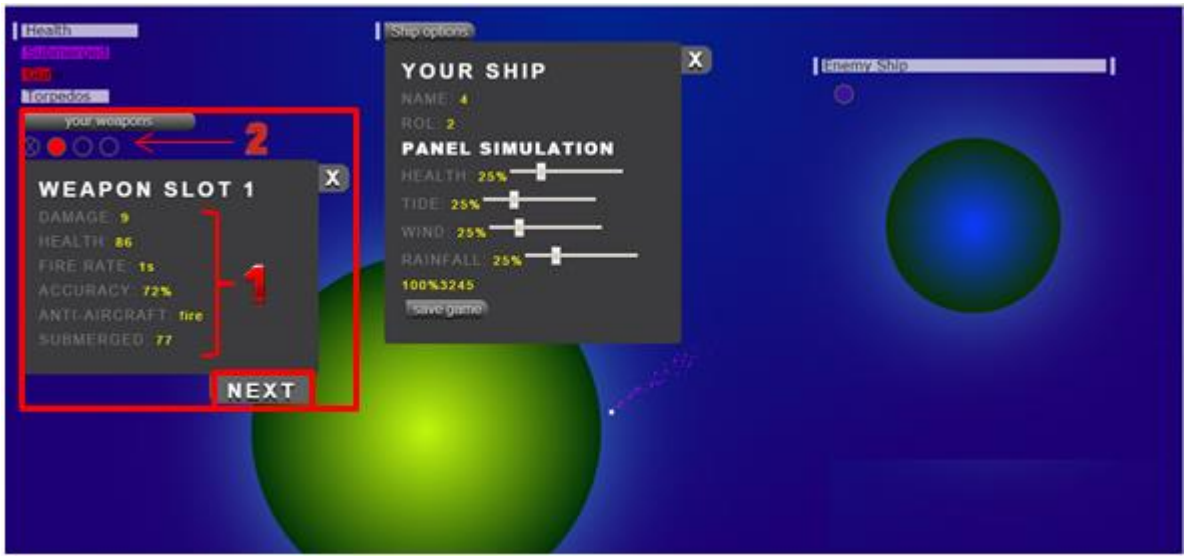


**Ilustración 39. Interfaz para indicar el panel de armas.**

### Panel *Opciones de la Unidad* o *Ship Options*

En la interfaz Ship Options, subrayada en la Ilustración 40 se visualiza el nombre código (1) y debajo de este el rol de dicha unidad en el combate (2) reservado para la unidad en uso. Debajo existe un panel de simulación donde se determinan las condiciones ambientales que inciden dentro de la simulación tales como las barras salud (Health), nivel del viento(Wind) , tiempo de lluvia(Rainfall) y Marea (Tide) Debajo de esta se encuentra el porcentaje de incidencia de estas condiciones en la simulación, en este ejemplo inciden equitativamente ya que todas están fijadas en un 25% y la suma de las mismas determina este nivel de incidencia. Por último este panel ofrece al usuario/cadete la posibilidad de guardar la partidas con todas las condiciones que se encuentren fijadas en ese momento (3), este evento activa el botón RESUME que carga la última partida guardada, esto se explica a fondo en el ítem correspondiente.

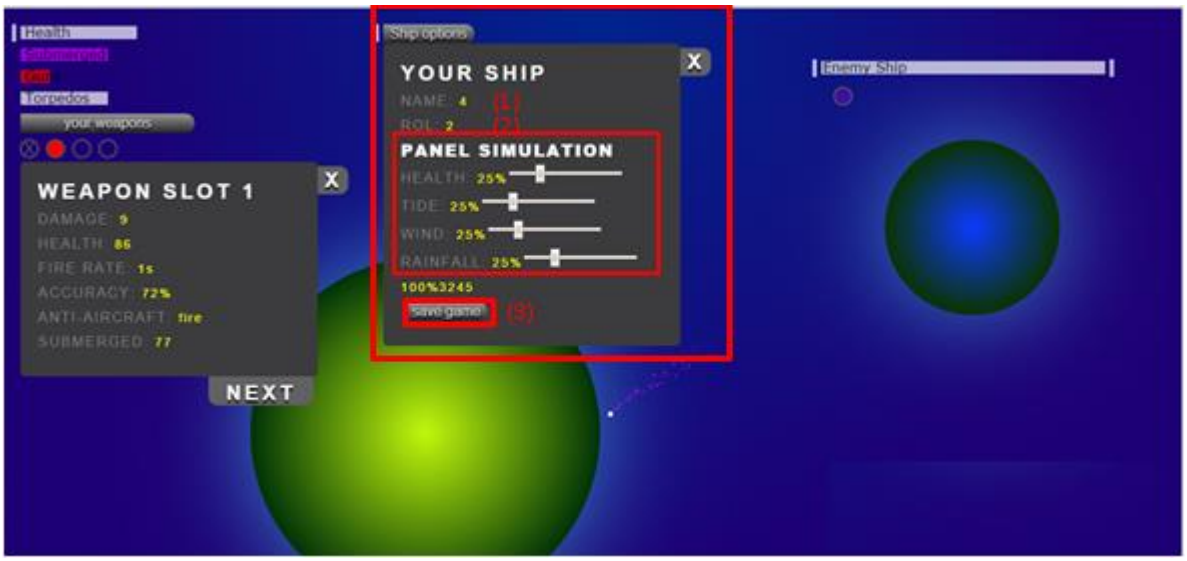


**Ilustración 40. Interfaz Ship Options.**

### *Final de la simulación*

Al finalizar la simulación el usuario puede resultar como ganador o perdedor del enfrentamiento  El sistema desplegará un cuadro de dialogo mostrando si su unidad ha sido destruida y si desea reiniciar la simulación (ver Ilustración 41), o en el caso de conseguir la victoria en el enfrentamiento aparecerá una nueva interfaz con 3 opciones: ***Destroy, Peace y Takeover*** *(ver* Ilustración *42)*. Cuando se selecciona una de estas opciones existen cambios en la unidad propia o del enemigo, es decir, si se escoge la opción *destroy (1)* automáticamente destruirá la unidad enemiga y se reiniciara la simulación con una nueva unidad contrincante; si se escoge la opción *peace(2)* se oficializará una tregua con la unidad enemiga, dando lugar a una nueva contienda con un nuevo enemigo; por último, si escoge *takeover(3)* se realizará un traslado hacia la unidad enemiga, invadiéndola y haciéndola propia para realizar una nueva simulación.

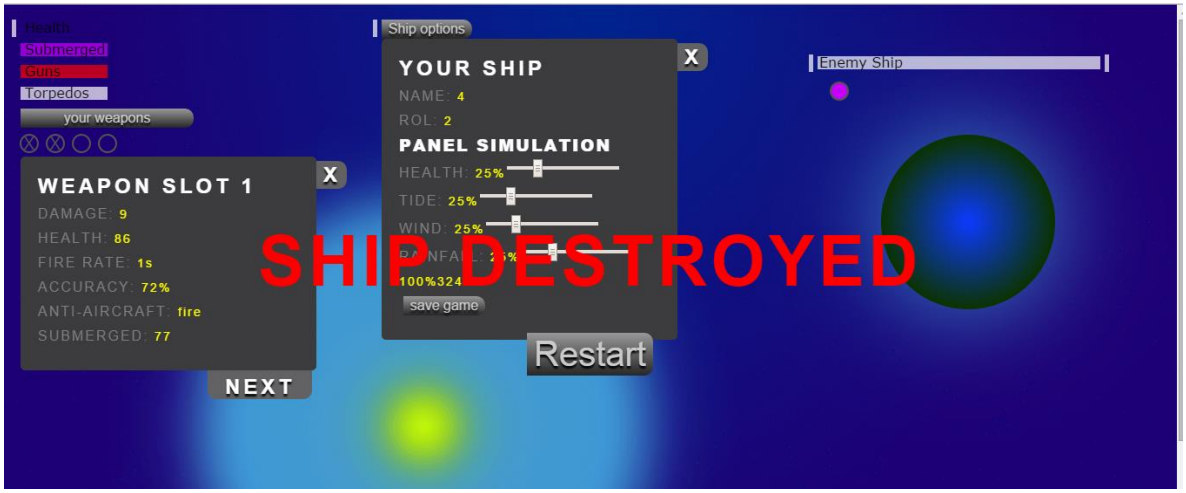


**Ilustración 41. Interfaz cuando se pierde la contienda al realizar la simulación.**


**Ilustración 42. Interfaz cuando se gana la contienda al realizar la simulación.**

## 2. List Ships

Al presionar el botón List Ships denotado en la pantalla principal con el número 2, cargará la interfaz de unidades (ver Ilustración 43 superior) donde se podrá observar con gran detalle cada una de las unidades que contempla el software, sólo basta con darle clic a la imagen en miniatura de la unidad para ver todas sus características en detalle. Para regresar a la interfaz de lista de unidades nuevamente presione el botón "x" subrayado en la Ilustración XXX Inferior, donde se detalla toda la información de la unidad.

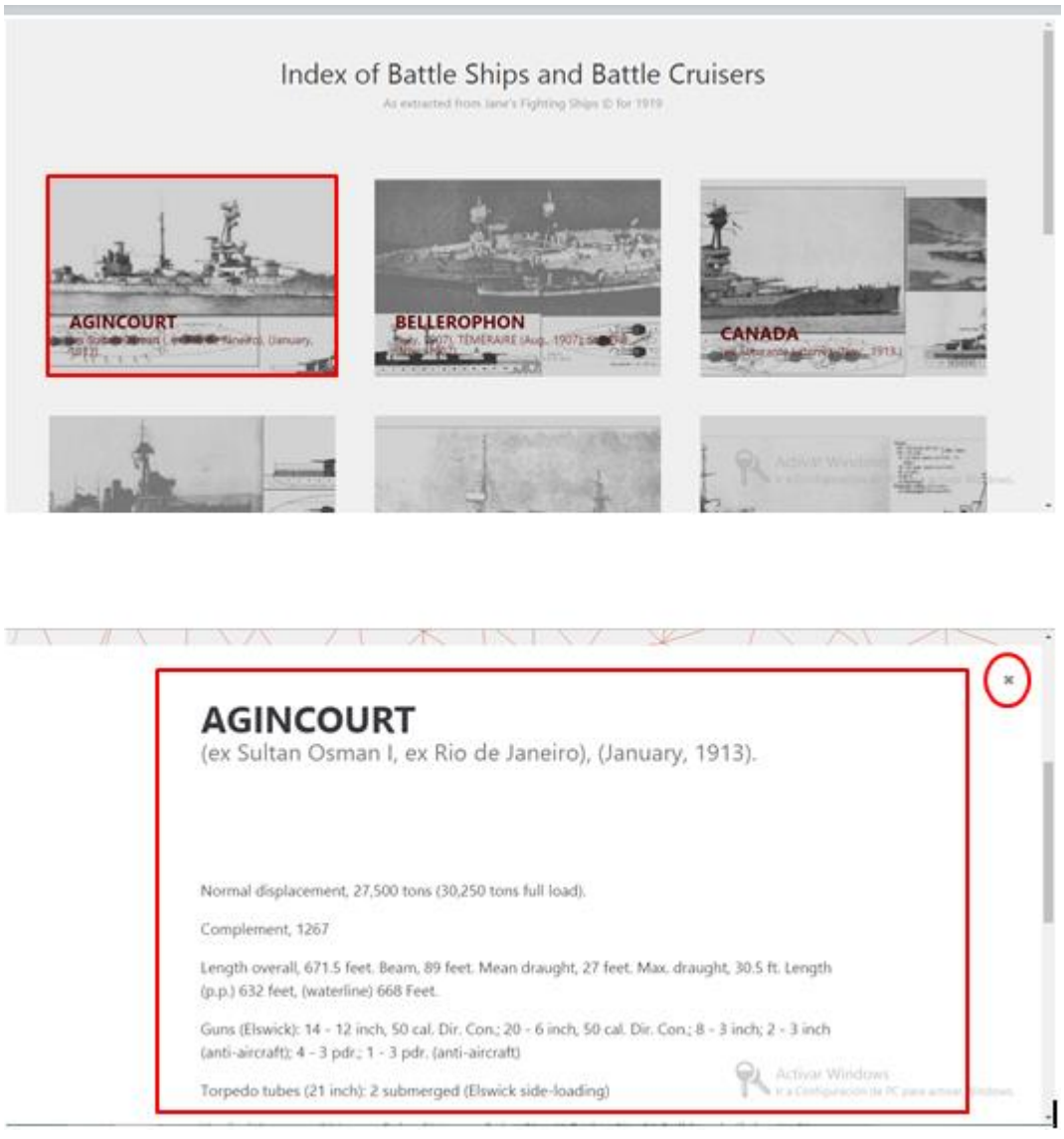


**Ilustración 43. Interfaz para mostrar las unidades militares.**

**2. RESUME**

En la pantalla inicial se carga el botón denominado RESUME denotado con el número 3, este botón sólo aparece si el usuario tiene una partida guardada previamente en el aplicativo, y por consiguiente al presionar este botón dicha partida se cargará en pantalla iniciando la simulación con las configuraciones guardadas en ese punto.

### 6.4.2 Ventajas del simulador NAVAL WARFARE frente a otros

Este simulador ofrece algunas ventajas para los cadetes de la ENAP, dichas ventajas se presentan a continuación:

- El cadete de la Escuela Naval Almirante Padilla puede sustentar su entrenamiento táctico militar con este software ya que le ofrece las herramientas necesarias para tomar una decisión con respecto a la usabilidad de las armas en situaciones específicas. El software ofrece la facilidad de alternar las armas de una unidad en cada enfrentamiento, teniendo en cuenta el desgaste de la munición de acuerdo al escenario planteado.
- El cadete tiene la facilidad de discernir la efectividad de una unidad frente a otra, permitiendo al cadete reconocer la efectividad de las unidades e ir construyendo su ranking en base a los resultados de la simulación. Se debe tener en cuenta el factor/error humano dentro de un enfrentamiento real, ya que esto no se toma en cuenta en la simulación.
- Gracias al panel de simulación, el cadete puede establecer los valores que el software tiene en cuenta para realizar la simulación y arrojar el resultado de un enfrentamiento. Los factores que el simulador toma en cuenta son: Salud(Health), Marea(Tide), Viento(Wind) y Humedad/Lluvia(Rainfall).

- De cierto modo el cadete afirma los conocimientos que imparte la Escuela Naval Almirante Padilla con este simulador, ya que en unos de sus módulos ofrece la información completa de las unidades investigadas, y además la facilidad de ver la efectividad y el rol de dicha unidad en un escenario predeterminado que contemple los factores mencionadas anteriormente. Cuando se habla de rol se refiere a la predisposición de dicha nave en un enfrentamiento normal de combate, como por ejemplo, las unidades que comúnmente se utilizan para asalto, las unidades que se ocupan de defender una base o pelotón(es), y las unidades que se utilizan generalmente para dar apoyo en situaciones de guerra.

Hoy en día, la simulación constituye una herramienta fundamental en la preparación para el combate de los miembros de las Fuerzas Armadas y en la planificación y conducción de las operaciones. A la Escuela Naval Almirante Padilla se le ofrece el aplicativo Naval Warfare Simulator como una herramienta capaz de simular el comportamiento de una unidad específica en un escenario predeterminado, esto no solo promueve la homogeneización y mejoramiento del nivel de preparación de los militares en Colombia, sino también la reducción del coste y el impacto medioambiental de sus actividades. De igual manera el uso de un simulador previene el riesgo al que es sometido un soldado dura la instrucción de estos ámbitos.

Aunque el objetivo real es poder integrar ésta arquitectura a una arquitectura mayor que abarque los escenarios y las unidades de guerra, validar la arquitectura planteada en este trabajo de investigación con un software que simula el trabajo de las unidades navales en entornos de combate, permitió denotar virtudes relevantes frente a la formación de cadetes. Dentro de estas virtudes están la facilidad del usuario para estipular condiciones ambientales favorables o desfavorables según sea el caso, este factor se tuvo en cuenta ya que no se ve muy seguido en los simuladores de combate naval convencionales, o por el hecho de probar la destreza del usuario obvian este aspecto de suma importancia en algunos casos como por ejemplo el simulador SETAC, expuesto anteriormente.

Otra de las bondades de este software es que prueba la capacidad de decisión de los cadetes, ya que muestra la efectividad de las unidades estudiadas en distintos escenarios permitiendo planificar mejores estrategias a la hora de estar en un enfrentamiento real.

### *6.5. Modelo de Simulación.*

Para la realización del modelo de combate entre unidades navales se tomó en cuenta la publicación del Ingeniero Donald R. Drew (Drew, 1995) donde se muestran los conceptos básicos a tener en cuenta a la hora de realizar simulación de combates navales. En base a los modelos matemáticos propuestos por Drew y los datos recolectados de los Janes proporcionados por la ENAP se llegó a un modelado final que es el implementado en la aplicación donde se aplica efectos probabilísticos tanto de impacto como de desacierto.

En la Ilustración 44 se aprecia el modelado desarrollado en el programa de simulación VENSIM PLE, donde se manejan dos equipos AZUL y ROJO, y diversas variables que determinan que conllevan a la victoria y derrota de algún bando. Para ambos se manejan variables de flujo de eventos tales como daño, probabilidad de impacto, zonas de impacto y número de disparos según el tiempo.

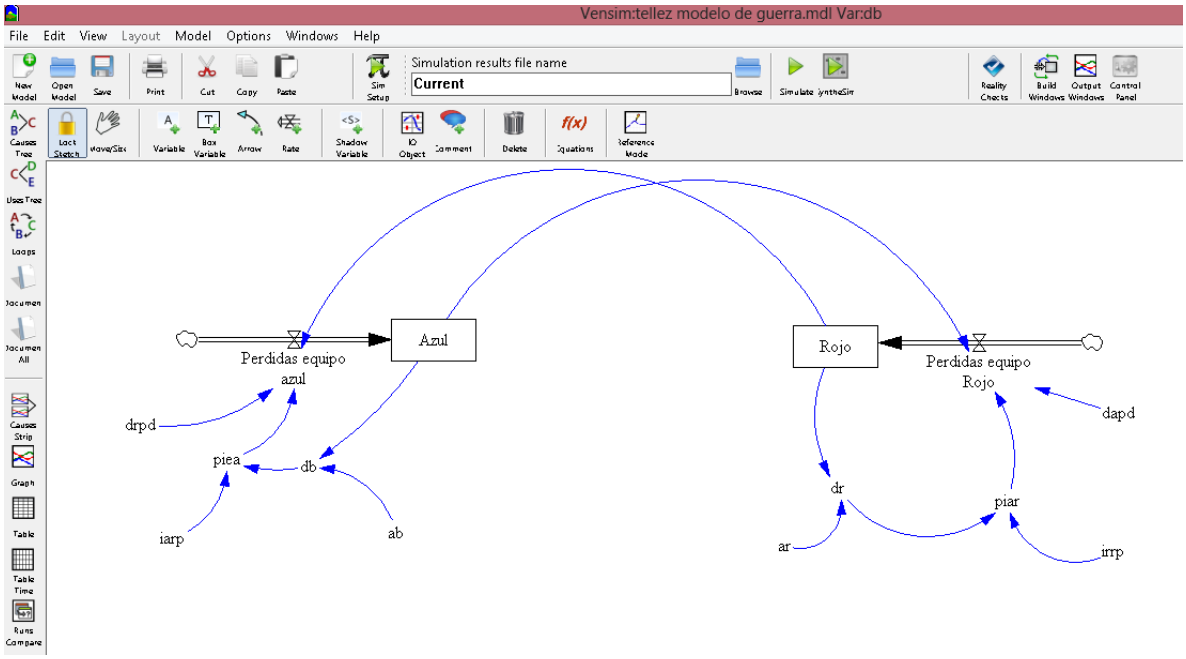


**Ilustración 44. Modelo de simulación de combate realizado en VENSIM.**

El trabajo de simulación representado en la presente investigación es basado en el libro Dinámica de Sistemas Aplicadas, del ingeniero Donald R. Drew, donde evalúa los conceptos necesarios para comprender la dinámica de combates, dando como resultado la definición de una Dinámica de Sistemas Militares explicado brevemente a continuación.

Las batallas se libran tanto en el tiempo como en el espacio. El proceso de desarrollo de modelos, además de tener en cuenta la representación espacial de las fuerzas, tiene básicamente que dividir el tiempo en intervalo o periodos discretos. Las condiciones iniciales al principio de un periodo se utilizan para predecir lo que va a pasar durante dicho período y los resultados obtenidos se transforman en las condiciones iniciales para el período siguiente. (Drew, 1995).

Con el fin de que un modelo matemático pueda ser desarrollado de manera concisa, es necesario que definamos los conceptos fundamentales. Una fuerza es un cuerpo o conjunto de unidades organizadas de combate. Estas unidades de combate pueden constar de

hombres, barcos, aviones, tanques o cualquier otro componente capaz de contribuir a la victoria de su bando. Hay muchos factores que contribuyen a esta capacidad. Estos pueden incluir la eficacia del armamento, su velocidad, maniobrabilidad, vulnerabilidad, etc. (Drew, 1995).

Supongamos que $ y X son los dos oponentes, $\$_0$ y $X_0$ , sus fuerzas respectivas, y $\$_1$ y $X_1$ sus fuerzas en el tiempo t después del comienzo del conflicto. La velocidad de atrición de $\$_t$ y $X_t$ está gobernada por ecuaciones diferenciales del siguiente tipo:

$$\frac{d\$_t}{dt} = -Q(\$_t) - B(X_t) - U(X_t)(\$_t) \qquad (4.3.1)$$

y

$$\frac{dX_t}{dt} = -P(S_t) - C(\$_t) - V(X_t)(\$_t) \qquad (4.3.2)$$

**Ilustración 45. Ecuaciones utilizadas para la simulación en el aplicativo.**

Los primeros términos de las ecuaciones corresponden a la atrición durante la interdicción de los objetivos, el segundo término corresponde a la atrición debida al fuego dirigido, y el tercer término a la atrición debido al fuego indiscriminado. Los parámetros Q y P son factores de atrición de las fuerzas amigas $ y las fuerzas enemigas X respectivamente; los parámetros B y C son parámetros de efectividad de las unidades de las fuerzas enemigas y amigas, respectivamente; y los parámetro U y V son factores de efectividad de la interacción para el fuego del área. (Drew, 1995).

# 7. RESULTADOS Y DISCUSIÓN

Principalmente, se obtuvo como resultado del proyecto, una arquitectura de software que simula la interacción entre unidades de combate naval y el resultado de la misma; en este caso, la entidad sobre la cual se propuso dicha arquitectura, fue la Escuela Naval Almirante Padilla. Sobre ella, se tomó como punto de partida, los programas con los que cuentan actualmente para entrenar a sus cadetes y oficiales, lo cual implicó dirigir la solución hacia una tecnología innovadora y más eficaz que la utilizada. Por tal razón, la arquitectura final representa la simulación de combates navales, utilizando las características de distintas unidades e incluyéndolo también en un entorno web.

Por otra parte, se cumplieron a cabalidad cada uno de los objetivos propuestos desde el principio del diseño de la arquitectura; con el aplicativo en un entorno web se cumple con el propósito de llegar a convertirlo en un servicio en la nube. Además dentro de este se listan las unidades más utilizadas por la ENAP, cumpliendo así con lo planteado en el objetivo general, respecto al estudio de la composición de dichas unidades navales. Durante el proceso de diseño de la arquitectura se fue cumpliendo con cada uno de los objetivos específicos planteados en el proyecto.

Muchos de los problemas de la ingeniería de software, en especial los cambios de funcionalidades en etapas tardías del proyecto, provienen de la imprecisión en la especificación de requerimientos. Por tal razón, la identificación de los requerimientos de este trabajo, fue una etapa que exigió mucho análisis, interpretación y tiempo, dando como resultado el logro del objetivo específico número 1.

Después de algunas reuniones con representantes de la ENAP, se fueron plasmando posibles soluciones para resolver la pregunta de investigación, esto permitió llegar hasta el documento fuente más relevante dentro de este proyecto (Jane’s) pudiendo conocer cómo estaban compuestas las unidades navales, identificando las variables que intervenían y como afectaban el comportamiento de estas según su valor. Cabe resaltar que nuestro trabajo de investigación se vio limitado por 2 puntos ajenos a la voluntad de los

investigadores: Primero, este proyecto fue pensado paralelamente a un trabajo sobre los escenarios de guerra, el cual no se llevó a cabo por motivos desconocidos, esto implicó recrear un escenario simple, basado en variables representativas que se mencionan más adelante, que afectaban una simulación para la validación de la arquitectura propuesta; por otro lado, La Escuela no tuvo presencia en la etapa de diseño e implementación del proyecto también por motivos desconocidos.

Con los requerimientos funcionales propuestos por La Escuela y plenamente identificados por los investigadores, se diseñó una arquitectura de software que permite conocer la composición de una unidad, dando así la solución al segundo objetivo específico.

Luego de desarrollar la arquitectura, se inició un proceso de verificación y validación, es decir, si se estaba diseñando correctamente la arquitectura y si estaba diseñando la arquitectura correcta, en la verificación la arquitectura debe estar conforme a sus requerimientos y en la validación debe permitir lo que el usuario realmente quiere. Para el proceso de verificación se contó con el apoyo del Ingeniero Martín Monroy, experto en el área de ingeniería de software. Finalmente para el proceso de validación, se desarrolló un prototipo funcional bajo las indicaciones de la arquitectura diseñada con el fin de garantizar su utilidad; dicho prototipo permite manipular variables (explicadas más adelante) que afectan el comportamiento de la simulación de enfrentamiento entre 2 unidades.

Finalmente con la arquitectura diseñada y el prototipo construido, se generaron pruebas de caja blanca y caja negra, con el fin de probar la funcionalidad y la estructura del proyecto respectivamente. Las pruebas de caja negra se basaron solo en el comportamiento de entrada y salida, para el proyecto los investigadores usaron la interfaz que compone el sistema en un entorno web, con la intención de buscar errores o en el mismo, dando como resultado errores en la comunicación del archivo de funciones, lo cual fue corregido de inmediato. Inicialmente se planteó una prueba de caja negra, donde se invitaba al cliente y a los usuarios finales a interactuar con el sistema pero debido a los problemas de comunicación entre La Escuela y La Universidad, esto no pudo realizarse. Se realizaron las pruebas por 3 días, a continuación se presenta el resumen en la siguiente tabla.

| Caja Negra | Falla | Correcciones |
|---|---|---|
| Pruebas de entradas de datos (Configuración de variables de factores externos) | No se presentaron | - |
| Pruebas de consulta (Listado de las unidades registradas en el Jane's) | Problemas con la obtención de los datos. | Configurar la invocación de la función que muestra las unidades en el WebService establecer conexión a los datos. |
| Prueba de ingreso al sistema | No se presentaron | - |
| Pruebas de salidas de información | Las salidas del módulo de la simulación no fueron las esperadas. | Rediseñar el sistema dinámico que alimentaba el servicio de simulación. |

**Tabla 7. Pruebas de caja negra realizadas por los investigadores**

Para las pruebas de caja blanca se buscó garantizar que:

• Se ejecutan al menos una vez todos los caminos independientes de cada función o servicio

• Se utilizan las decisiones en su parte verdadera y en su parte falsa

• Se ejecuten todos los bucles en sus límites

• Se utilizan todas las estructuras de datos internas

El tipo de prueba de caja blanca que se aplicó a este trabajo fue el de Bucles, debido a que estos son el mayor inconveniente de la mayoría de los algoritmos implementados en software, por lo que se debe prestar atención especial a la hora de realizar la prueba del software. De esta manera se evidencia el cumplimiento de los objetivos específicos 3 y 4.

Con el fin de lograr resolver la pregunta de investigación del presente trabajo, como punto de arranque, se realizó un plan de trabajo donde se especificó las actividades que componen un desarrollo de software de un proyecto; determinando el esfuerzo que tendrá cada una de las actividades, el seguimiento periódico y la duración de acuerdo al tamaño del proyecto, después se definieron los objetivos y el alcance del proyecto (una tarea de mucho análisis para no incurrir en cambios de requerimientos luego de iniciado el proyecto), se debatió con los integrantes que ejecutarán el proyecto, los tutores involucrados que estarán apoyando la iniciativa de desarrollo y se realizó un análisis de los hitos importantes de cada fase del proyecto. Una de las etapas más importantes en el desarrollo de software es la de Análisis de requerimientos, desde un inicio el proyecto debe ser entendido tal y como lo desea el cliente, en esta etapa la ingeniería del software es crítica para reconocer requerimientos incompletos, ambiguos o contradictorios. Usualmente el cliente/usuario tiene una visión incompleta/inexacta de lo que necesita y es necesario ayudarle para obtener la visión completa de los requerimientos. El contenido de comunicación en esta etapa es muy intenso ya que el objetivo es eliminar la ambigüedad en la medida de lo posible para no desarrollar el producto/servicio que el cliente no desea. Luego de realizar entrevistas con integrantes de La Escuela, se especificó de manera clara lo revisado y se debatieron las especificaciones de requerimientos de la arquitectura con los tutores del proyecto, como resultado se definió la Presentación del producto (Propósito de la arquitectura, sus restricciones y su alcance), la Descripción general (Funcionalidad del sistema, Diagramas de caso de uso, Listado de actores, etc.), el Detalle del requerimiento (Requerimientos funcionales y no funcionales, detalles de los casos de uso, prototipo de interfaz de usuario, reglas y las funciones del negocio o sistemas), los Requerimientos de Interfaz (Interfaces de usuario, Interfaces de Comunicación, etc.) y las especificaciones suplementarias (Restricciones de diseño, observaciones y criterios de aceptación de la arquitectura y el prototipo), de esta manera se pudo resolver la pregunta de investigación *¿Cómo se puede asegurar un buen desarrollo, codificación y estandarización de las unidades militares navales para el apoyo durante el entrenamiento táctico-militar de los estudiantes de la*

*ENAP?,* cabe resaltar que fue muy necesario llevar una administración total del proyecto, la planeación inicial de los detalles técnicos de la arquitectura y los planes de comunicación entre los investigadores y tutores del proyecto, el monitoreo del tiempo, esfuerzo, costo, actividades, entre otras, donde frecuentemente se compartía el avance real vs lo planeado, dándole seguimiento a las actividades hasta su cierre y por último, el aseguramiento de la calidad, revisar que todas las acciones del proyecto se lleven de acuerdo a los estándares de calidad en cuanto a la arquitectura y la codificación del prototipo.

Atendiendo el requerimiento de basarse en la tecnología Cloud Computing o Servicio en la nube, se diseñó la arquitectura como un servicio (SaaS: Software as a Service), que funciona como un modelo de distribución de software, en donde la empresa proveedora de dicho servicios (en este caso La Universidad) proporciona el mantenimiento, soporte y operación que usará el cliente (La Escuela) durante el tiempo que haya contratado el servicio. La arquitectura fue diseñada para que La Escuela usará el sistema alojado en la web , la cual mantendrá la información de La Escuela en sus sistemas y proveerá los recursos necesarios para explotar esa información (TechnoReeze, 2011).

Software como servicio o SaaS, es probablemente el tipo de servicio más común de la nube. Este trabajo fue diseñado para ser llamado desde cualquier navegador cliente y para estar alojado en un servidor web, esta arquitectura contempla la creación de un servicio desconocido que funcionaría como puente de conexión para procesar la solicitud que se entrega a los usuarios desde los servidores del proveedor.

### *7.1. Resultados del prototipo que valida la arquitectura*

Durante el proceso de diseño de la arquitectura de software, el objetivo de la fase de validación consiste en aumentar la confianza con respecto a que la arquitectura es adecuada para cumplir con los requerimientos del sistema. Como técnica de validación se escogió el desarrollo de un prototipo, cuyo objetivo es identificar posibles deficiencias y debilidades en el diseño para que puedan ser mejoradas antes de la implementación. Básicamente, esta técnica involucró la construcción de un prototipo que crea un arquetipo de la aplicación deseada, de esta forma su capacidad para satisfacer las necesidades se pueden evaluar con

más detalle, esta técnica es la mejor manera de validar la arquitectura diseñada debido a que nos da solución directa a dos interrogantes claves para el proceso de implementación del proyecto: ¿Puede la arquitectura como está diseñada, construirse en una forma que pueda satisfacer los requerimientos? Y ¿La tecnología (Lenguajes de programación, aplicaciones integradas, librerías, etc.) seleccionadas para implementar la aplicación se comportan como se espera?. Además de esto, el prototipo permite proporcionar a los usuarios la demostración, lo que les permite experimentar con el modelo operativo mientras que se avanza en el proceso de implementación de la aplicación real. A medida que obtienen experiencia con el prototipo, su curva de aprendizaje se reduce y colaboran mejor a la hora de perfeccionar y ajustar la aplicación. También se logra una posición mejor para conseguir una mayor productividad y satisfacción en la etapa de la implementación final porque ya comprenden la estructura básica de la aplicación.

Dada una situación hipotética de conflicto entre dos naciones, que obligue el enfrentamiento naval entre ellas, se deben tener en cuenta varios factores que incidirán en la victoria , los factores externos que el prototipo tiene en cuenta al momento de simular la batalla naval son Viento, Marea, Lluvia, Población, Probabilidad de impacto y Probabilidad de fallo, se definieron estas variables debido a que los modelos de sistemas dinámicos utilizados para la simulación, tenían en común dichos factores. Dependiendo de la composición robusta del barco se definirá su valor de salud, y con los factores de marea, viento y lluvia se completará el círculo de factores externos que inciden sobre el resultado de un combate naval, esto según las necesidades encontradas. En el prototipo se manejó un porcentaje que determina la concentración o la incidencia de dicho factor sobre la simulación.

Para la realización de las pruebas se establecieron los valores de las variables o factores externos en un punto específico del 25% cada uno, este valor fue un dato aleatorio para fines de pruebas, en total se realizaron 5 pruebas con las entradas que se ven en la Tabla 8, arrojando una simulación de una unidad propia frente a la máquina:

| Parámetros de entrada/salida | Porcentaje de Resultados |
|---|---|
| Viento | 25% |
| Marea | 25% |
| Lluvia | 25% |
| Población | 380 tripulantes |
| Probabilidad de impacto | 40% |

**Tabla 8. Porcentaje de los factores externos para las pruebas realizadas**

Al utilizar el prototipo NavalWarfare Simulator con el doble de los valores especificados anteriormente se observó que la relación entre la probabilidad de impacto y la probabilidad de fallo de los proyectiles era un poco más amplia a la mostrada por los resultados esperados, lo que permitía tiempo de recuperación para las unidades. Los resultados obtenidos de las 5 pruebas realizadas se presentan en la Tabla 9

| Parámetros de salida | Porcentaje de resultados |
|---|---|
| Victorias | 65% |
| Derrotas | 35% |
| Resistencia | 35% |
| Población | 320 tripulantes |
| Probabilidad de impacto | 58% |

**Tabla 9. Porcentaje de resultados obtenidos al realizar las pruebas en el simulador con diferentes factores externos**

Con estos resultados se evidencia que el simulador toma rigurosamente el aumento o disminución referente a factores externos, además se debe tener en cuenta que los tripulantes dentro de cada nave proveen reparaciones que aumentan el aguante de esta con el pasar del tiempo, también la potencia de los distintos tipos de municiones que se tenga ya que unas provocan un daño mucho mayor a otras. Por otra parte una de las ventajas que ofrece realizar pruebas de simulación con relación a los métodos convencionales de análisis, como los modelos estadísticos matemáticos, es que estos últimos no pueden dirigir eficientemente las variaciones pues los cálculos se derivan de valores constantes. Mediante un sistema que incorpora interdependencia, la simulación tiene en cuenta las variaciones, así como la interacción entre los componentes y el tiempo.

Es importante destacar que durante la etapa del anteproyecto se definió que la arquitectura estaría orientada a apoyar el entrenamiento del personal de la Escuela Naval Almirante Padilla, desafortunadamente y como se mencionó anteriormente, en la Escuela hubo cambio de administración en el área con la cual se establecieron los contactos para la realización de este proyecto (Facultad de Ciencias Navales). El personal de dicha área fue trasladado y no fue posible establecer las comunicaciones entre el nuevo personal y la Universidad de Cartagena. Por lo cual todas las pruebas fueron realizadas sin la intervención de la Escuela Naval por ende no se cuenta con ninguna carta de aceptación por parte de ella. Sin embargo, el desarrollo de la arquitectura se realizó siguiendo los lineamientos y requerimientos de la Escuela, ya que la parte de análisis y levantamiento de requerimientos si se alcanzó a llevar a cabo.

La necesidad del manejo de la arquitectura de un sistema de software nace con los sistemas de alta complejidad y envergadura, que se proponen como solución a un problema determinado. En la medida que los sistemas de software crecen en complejidad, bien sea por número de requerimientos o por el impacto de los mismos, se hace necesario establecer medios para el manejo de esta complejidad (Hofmeister et al., 1996). En general, la técnica es descomponer el sistema en piezas que agrupan aspectos específicos del mismo, producto de un proceso de abstracción (Bass et al., 1998) y que al organizarse de cierta manera constituyen la base de la solución de un problema en particular.

Concluyendo la etapa de validación del proyecto, según Kazman et al. (2001), el primer paso para la evaluación de una arquitectura es conocer qué es lo que se quiere evaluar, en este caso, el objetivo de evaluación del proyecto es comparar la arquitectura de software con respecto a los requerimientos del sistemas. Para esto se desarrolló un prototipo funcional bajo las indicaciones de la arquitectura, se sometió a pruebas de verificación y validación dando resultados positivos de acuerdo a los requerimientos iniciales y de esta manera cumplir satisfactoriamente el objetivo general del proyecto.

# 8. CONCLUSIONES Y RECOMENDACIONES

## 8.1. CONCLUSIONES

El análisis arquitectural ha emergido como una de las principales temáticas a futuro que concentrará los esfuerzos científicos aplicados en la ingeniería de software, los múltiples resultados obtenidos durante toda la investigación realizada fundamentados en referentes bibliográficos aceptados por la comunidad científica así lo demuestran.

El enfoque matemático puede proporcionar una salida a la necesidad de análisis arquitectural en el campo de las arquitecturas de software pero antes de llegar a una solución definitiva, múltiples propuestas y enfoques serán abordados lo que enriquecerá el resultado final una vez se consume.

Para la investigación realizada se cumplieron los objetivos planteados. En primer lugar se obtuvo como resultado un estado del arte debidamente construido y organizado que sin duda servirá a futuros investigadores del área como punto de partida para el desarrollo de sus proyectos. En aplicaciones software orientadas para el sector naval y/o de entrenamiento militar se pueden encontrar, a nivel internacional, una gran variedad de aplicaciones orientadas para computadores de escritorio como Naval War Artic Circle, Warship Combat , SETAC, entre otros. A nivel nacional, no existen soluciones en el campo. Estos trabajos proporcionan una opción de apoyo al Cadete a través de

computadores de escritorio y personales. Sin embargo, las aplicaciones previamente mencionadas se enfocan más en el entretenimiento del usuario que en el entrenamiento, quien más se asemeja a esto es el SETAC, cuyo objetivo es ser una herramienta para el entrenamiento para los oficiales y suboficiales de la Fuerza Armada Chilena, su interfaz no es muy amigable con el usuario y la usabilidad del software es limitada. El desarrollo de soluciones software a la medida con enfoques educativos en el ámbito de entrenamiento naval que contemplen el uso de computadores de escritorio y la intervención de estudiantado y profesorado, es todavía limitado. Con el fin de cumplir los objetivos propuestos, nos plantemos la siguiente pregunta de investigación, ¿Cómo se puede asegurar un buen desarrollo, codificación y estandarización de las unidades militares navales para el apoyo durante el entrenamiento táctico-militar de los estudiantes de la ENAP?, a lo que Finalmente llevó al logro de la definición de una arquitectura de software de unidades militares, el cual fue validado mediante un aplicativo web en un escenario de prueba pertinente dando cumplimiento a los objetivos tres y cuatro respectivamente de los planteados para la investigación.

La razones por la que la ENAP se debe inclinar por el desarrollo de esta arquitectura es debido a la tecnología que se implementó, los requerimientos que atendimos y la alta posibilidad de escalabilidad que tiene debido a que hace parte de un proyecto más grande previamente conversado con representantes de la ENAP y profesores de la Universidad de Cartagena.

Cabe mencionar que los resultados obtenidos presentan bastante afinidad con el trabajo científico de análisis de modelos de combate que habían realizado algunos autores, el método definido proporciona información proveniente del análisis arquitectural realizado que se complementa fácilmente con resultados encontrados en la revisión literaria coincidiendo de esta manera con estudios que existían previamente en el área de la arquitectura de software y modelos de combate naval.

Es importante destacar que la investigación se vio limitada por varios factores. En primer lugar el factor económico restringió el acceso únicamente a bases de datos gratuitas

proporcionadas por la Universidad De Cartagena, que si bien fueron muy útiles, seguramente no contienen trabajos más específicos y mejor realizados que pudieron haber sido incluidos en la investigación. En segundo lugar el factor tiempo fue otro limitante puesto que había un cronograma que cumplir y no se podían seguir prolongando las fechas de entrega. Por otro lado, el recurso humano fue otro de los limitantes de la investigación que solo recayó en el director del proyecto y los investigadores principales, y en este tipo de proyectos se necesita de un equipo totalmente capacitado para conseguir un material con la calidad óptima.

Finalmente es relevante recalcar que todas las pruebas fueron realizadas sin la intervención de la Escuela Naval, por diversos cambios no previstos en su administración, por ende no se cuenta con ninguna carta de aceptación por parte de ella. Sin embargo, el desarrollo de la arquitectura se realizó siguiendo los lineamientos y requerimientos de la Escuela, ya que la parte de análisis y levantamiento de requerimientos, al igual que el diseño, diagramado y ejecución de la arquitectura se llevaron a cabo a la perfección.

## 8.2. RECOMENDACIONES

Para trabajos futuros se recomienda tener énfasis en dos puntos que pueden complementar y aumentar la calidad del resultado obtenido en el proyecto realizado.

En primer lugar, conseguir herramientas para desarrollar un prototipo con mayor calidad, incursionando en novedosos lenguajes de programación que brinden las herramientas necesarias para cumplir con mayor precisión en todos los objetivos del proyecto, lo que sin dudas robustecería los resultados obtenidos.

Por otra parte es de mucha importancia mantener vinculado al usuario final del producto/servicio durante todas las fases del software, mantener la relación entre instituciones y generar cooperación de ambas partes permite trabajar en la dirección

correcta, bajo las indicaciones del cliente y con su participación, se obtiene un resultado oportuno y de calidad.

A partir de los objetivos específicos se tienen varias recomendaciones. En primer lugar, construir un protocolo debidamente estructurado para mapeo sistemático. Por otra parte, una recomendación fruto de la experiencia de los autores es que en la medida de lo posible, la temáticas de tipo didáctico con entidades de gran magnitud como la ENAP, debería poder contener el trabajo de varios desarrolladores e investigadores. Sería beneficioso que al menos dos o tres personas estén disponibles para desarrollar este tipo de temáticas para obtener un producto de mejor calidad. De esta forma se enriquecerían los resultados obtenidos.

Finalmente el diseño y desarrollo de una arquitectura de software enfocada en las unidades militares, resultado de la investigación realizada, constituiría un excelente valor agregado al proyecto final de Simulador de Combate para el entrenamiento táctico-militar de cadetes de La Escuela, puesto que implicaría la estandarización de muchos de los componentes para definir una unidad y los elemento que interoperan con ella.

## 9. ANEXOS

Entrevistas realizadas en la Escuela Naval por parte de los autores. La evidencia se encuentra en la carpeta Anexos, los archivos llevan por nombre Voz002 y Voz004 con extensión de sólo audio 3GA.

## 10. BIBLIOGRAFÍA


Real Academia De La Lengua Española - RAE. (s.f.). *Diccionario De La Lengua Española*. Recuperado el 22 de Julio de 2013, de http://lema.rae.es/drae/

Booch, G., Rumbaugh, J., & Jacobson, I. (1999). *The Unified Modeling Language User Guide.* Reading, MA: Addison-Wesley.

Canfora, G., & Di Penta, M. (2007). New Frontiers of Reverse Engineering. *Future of Software Engineering - IEEE Computer Society* .

Ducasse, S., & Pollet, D. (2009). Software Architecture Reconstruction: A Process Oriented Taxonomy. *IEEE Transactions On Software Engineering* , 573-590.

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1995). *Design Patterns: Elements of Reusable Object-Oriented Software.* Reading, MA: Addison-Wesley.

Gorton, I. (2006). Essential Software Architecture. *Springer-Verlag* .

Barbacci, M., Ellison, R., Lattanze, A., Stafford, J., Weinstock, C., & Wood, W. (2003). *Quality Attribute Workshops (QAWs), Third Edition.* Pittsburg, Pennsylvania: Carnegie Mellon University.

Canfora, G., Di Penta, M., & Cerulo, L. (2011). Achievements and Challenges in Software Reverse Engineering. *Communications of the ACM* , 142-151.

Chifosky, E., & Cross, J. (1990). Reverse Engineering and Design Recovery: A Taxonomy. *IEEE Software* , 13-17.

Hassan, A. E., & Holt, R. C. (2002). Architecture recovery of web applications. *ICSE '02* , 19-25.

Kazman, R., Clements, P., & Klain, M. (2005). *Evaluating Software Architectures, Methods and Case Studies.* Adison-Wesley Professional.

Kazman, R., & Bass, L. (2002). Making architecture reviews work in the real world. *IEEE Software* , 67-73.

Kazman, R., Bass, L., Abowd, G., & Webb, M. (2007). SAAM: A Method for Analyzing the Properties of Software Architectures. *Software Engineering Institute Papers* .

Larman, C. (2003). *UML y Patrones: una introducción al análisis y diseño orientado a objetos y al proceso unificado.* Madrid: Pearson Educación, S.A.

Lijun, L., Changyun, L., Gexin, M., & Zhibing, W. (2012). Layered Semantic Analysis of Software Architecture. *IEEE Computer Society* .

López Cano, J. L. (1984). *Métodos e Hipótesis Científicas.* Ciudad De México: Pearson Education.

Monroy, M., Arciniegas, J., & Rodríguez, J. (2012). Caracterización de Herramientas de Ingeniería Inversa. *Información Tecnológica* , 31-42.

Moreland, K., King, B., Maynard, R., & Ma, K.-L. (2013). Flexible Analysis Software for Emerging Architectures. *IEEE Computer Society* .

Podgurski, A. C. (2006). A formal model of program dependences and its implications for software testing, debugging, and maintenance. *IEEE Transaction on Software Engineering* , 965-979.

Pressman, R. (2009). *Ingeniería de software: un enfoque práctico.* Prentice-Hall.

Ritchey, T. (1991). Analysis and Sythesis on Scientific Method, Based on a Study by Bernhard Riemann. *Systems Research* , 21-41.

Sampieri Hernández, R., Fernández, C., & Baptista, P. (1996). *Metodología De La Investigación.* Bogotá: Mc Graw-Hill.

Störmer, C. (2007). *Software Quality Attribute Analysis by Architecture Reconstruction (Doctoral Thesis).* Vrije Universiteit.

Kazman, R., Klein, M., & Clements, P. (2000). *ATAM: Method for Architecture Evaluation.* Carnegie Mellon University.

Van Deursen, A. (2001). Software Architecture Recovery and Modelling. *Working Conference on Reverse Engineering*, (págs. 4-7).

Wu, J., & Winter, A. (2002). Towards a Common Query Language for Reverse Engineering. *IEEE Computer Society* .

Acher, M., Cleve, A., Collet, P., Merle, P., Duchien, L., & Lahire, P. (2011). Reverse Engineering Architectural Feature Models. *IEEE Computer Society* .

Panas, T., Löwe, W., & Assmann, U. (2003). Towards the Unified Recovery Architecture for Reverse Engineering. *IEEE Computer Society* .

Jarzabek, S., & Woon, I. (2007). Towards a Precise Description of Reverse Engineering Methods and Tools. *IEEE Computer Society* .

Beck, F., & Diehl, S. (2010). Visual Comparison of Software Architectures. *ACM - International symposium on Software visualization SOFTVIS* , 183-192.

Li, Z., Gittens, M., Shariyar Murtaza, S., & Madhavji, N. (2010). Towards an Extended Relational Algebra for Software Architecture. *ACM - Software Engineering Notes SIGSOFT volume 35 Issue 3* , 1-4.

SEI. (2013). *Software Engineering Institute - Carnegie Mellon*. Recuperado el 26 de Mayo de 2013, de http://www.sei.cmu.edu/architecture/tools/evaluate/atam.cfm

Rojas, R. (2005). *Guía para realizar investigaciones sociales.* Ciudad de México: Plaza y Valdéz editores.

Maturro, G., & Saavedra, J. (2012). *Gestión Del Conocimiento y La Experiencia en Ingeniería Del Software.* Montevideo, Uruguay: Lambert Academic Publishing GmbH & Co.

Universidad De Sevilla. (Noviembre de 2012). *Introducción al Álgebra Relacional.* Recuperado el 26 de Mayo de 2013, de http://www.lsi.us.es/docencia/get.php?id=6364

Sicilia, M. (2009). *Técnicas de Mantenimiento del Software*. Recuperado el 25 de Agosto de 2013, de http://cnx.org/content/m17431/1.4/

Heijstek, W., Kühne, T., & Chaudron, M. (2011). Experimental Analysis of Textual and Graphical Representations for Software Architecture Design. *ACM - International Symposium on Empirical Software Engineering and Measurement* .

Date, C. (2013). *Relational Theory for Computer Professionals.* United States: ACM - O'Reilly Media Inc.

Muller, H., Jahnke, J., Smith, D., Storey, M., Tilley, S., & Wong, K. (2000). Reverse Engineering: A roadmap. *ACM - Future of Software Engineering Track* , 47-60.

Achinstein, P. (2004). General Introduction to Science Rules: A Hitorical Introduction to Scientific Methods. Science Direct - Johns Hopkins University Press.

Tahuiton Mora, J. (Agosto de 2011). *Arquitectura de Software para aplicaciones móviles.* Obtenido de Arquitectura de Software para aplicaciones móviles: http://delta.cs.cinvestav.mx/~pmalvarez/tesis-tahuiton.pdf

Mollineda, R. (Febrero de 2005). *Arquitectura y ocio.* Obtenido de Arquitectura y ocio: http://www.iti.es/media/about/docs/tic/06/2005-02-arquitectura.pdf

Cervantes, H. (Abril de 2010). *Arquitectura de Software.* Obtenido de Arquitectura de Software: http://sg.com.mx/revista/27/arquitectura-software#.VPXHMpBZjIU

Limon Cordero, R. (s.f.). *Las vistas arquitectónicas de software y sus correspondencias mediante la gestión de modelos.* Obtenido de Las vistas arquitectónicas de software y sus

correspondencias mediante la gestión de modelos: http://www.dsic.upv.es/docs/bib-dig/tesis/etd-10132009-094823/borrador-tesis-rogelio-2.pdf

combate, M. d. (Junio de 2012). *Máquina de Combate.* Obtenido de Máquina de Combate: http://maquina-de-combate.com/blog/?p=25248

Osorio Romero, J. (2014). *Definición de un método basado en álgebra relacional para realizar análisis a arquitecturas de software obtenidas a partir de ingeniería inversa.* Cartagena.

Mollineda, R. (Febrero de 2005). *Arquitectura y ocio.* Obtenido de Arquitectura y ocio: http://www.iti.es/media/about/docs/tic/06/2005-02-arquitectura.pdf

Plúas Andrade, L., Rivera Cárdenas, L., & Cortez Alvarez, D. (s.f.). *Integración de Sistemas Tácticos Navales.* Obtenido de Integración de Sistemas Tácticos Navales: https://www.dspace.espol.edu.ec/bitstream/123456789/1400/1/2663.pdf

Rancan, C. (2004). *Arquitectura de Sistema Híbrido de Evaluación del Alistamiento de Unidades Navales Auxiliares.* Obtenido de Arquitectura de Sistema Híbrido de Evaluación del Alistamiento de Unidades Navales Auxiliares: http://www.researchgate.net/publication/26520175_Arquitectura_de_Sistema_Hbrido_de_Evaluacin_del_Alistamiento_de_Unidades_Navales_Auxiliares

Rivera López, A. (1 de Enero de 2008). *Sistema asistente para la generación de horarios.* Obtenido de Sistema asistente para la generación de horarios: (http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/rivera_l_a/capitulo2.pdf

*MathWorks*. (s.f.). Obtenido de http://www.mathworks.com/discovery/modeling-and-simulation.html

Turk, M. (13 de Febrero de 2012). *Naval War Artic Circle cover*. Recuperado el 10 de Junio de 2015, de Naval War Artic Circle cover: http://en.wikipedia.org/wiki/Naval_War:_Arctic_Circle#/media/File:Naval_War_Arctic_Circle_cover_art.jpg

Kempretan. (23 de Enero de 2013). *Kempretan blog*. Recuperado el Mayo de 2015, de Kempretan blog: http://kempretan.blogspot.com/2013/12/free-download-pc-games-nwac-naval-war.html

Reynoso, C. (2004). *Introducción a la arquitectura de software.* Obtenido de http://carlosreynoso.com.ar/archivos/arquitectura/Arquitectura-software.pdf

Carrillo Andrade, P. J., & Ortega Gutierrez, L. E. (2012). *METODOLOGIA DE DISEÑO, DESARROLLO Y EVALUACIÓN DE SOFTWARE PARA JUEGOS DE GUERRA.* Recuperado el Mayo de 2015, de Escuela Politécnica del ejercito, Ecuador: http://repositorio.espe.edu.ec/bitstream/21000/6372/1/T-ESPEL-CDT-1009.pdf

Informaciones. (2009). *Juegos para entrenar: Uso militar de los juegos de guerra.* Obtenido de https://haddensecurity.wordpress.com/2009/08/30/juegos-para-entrenar-uso-militar-de-los-juegos-de-guerra/

Uparella. (2007). *Simulacion Virtual Interactiva.* Obtenido de http://es.scribd.com/doc/44167909/Simulacion-Virtual-Interactiva#scribd

Singh, R., & Sarjoughian, H. S. (2003). *Software Architecture for Object-Oriented Simulation Modeling and Simulation Environments: Case Study and Approach.* Tempe: Dept. of Computer Science & Engineering Ira A. Fulton School of Engineering Arizona State University.

Sung, C. H., Moon, C., & Kim, T. G. (2010). *Collaborative Work in Domain-Specific Discrete Event Simulation SoftwareDevelopment: Fleet Anti-air Defense Simulation Software.* Recuperado el Mayo de 2015, de 2010 Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises: http://www.academia.edu/2830721/Collaborative_Work_in_Domain-Specific_Discrete_Event_Simulation_Software_Development_Fleet_Anti-air_Defense_Simulation_Software

Singh, R., & Sarjoughian, H. S. (Septiembre de 2003). *Software Architecture for Object-Oriented Simulation.* Recuperado el Mayo de 2015, de Software Architecture for Object-Oriented Simulation: http://acims.asu.edu/wp-content/uploads/2012/02/TechRep-03-CSE-ASU.pdf

Mora, J. T. (Agosto de 2011). *Arquitectura de software para aplicaciones web.* Obtenido de http://delta.cs.cinvestav.mx/~pmalvarez/tesis-tahuiton.pdf

Drew, D. R. (1995). Dinamica de Sistemas Aplicada. En D. R. Drew, *Dinamica de Sistemas Aplicada* (pág. 130). Madrid: Gráficas Marte.