



**UNIVERSIDAD  
DE ANTIOQUIA**

**EXPLORACIÓN DE UNA HERRAMIENTA PARA LA  
REPRESENTACIÓN Y VALIDACIÓN DE PRÁCTICAS DE  
DESARROLLO DE SOFTWARE BASADA EN EL ESTÁNDAR  
ESSENCE DE OMG**

Autor

Orvie Nadir Salgado Espitia

Universidad de Antioquia

Facultad de Ingeniería, Departamento Ingeniería de  
Sistemas

Medellín, Colombia

2021



EXPLORACIÓN DE UNA HERRAMIENTA PARA LA REPRESENTACIÓN Y VALIDACIÓN DE PRÁCTICAS DE  
DESARROLLO DE SOFTWARE BASADA EN EL ESTÁNDAR ESSENCE DE OMG

**Orvie Nadir Salgado Espitia**

Trabajo de grado presentado como requisito parcial para optar al título de:

**Ingeniero de Sistemas**

Asesores (a):

Juan Ricardo Cogollo Oyola

Ingeniero de Sistemas y Magíster en Ingeniería

Universidad de Antioquia

Facultad de Ingeniería, Departamento Ingeniería de Sistemas

Medellín, Colombia

2021

# EXPLORACIÓN DE UNA HERRAMIENTA DE SOFTWARE QUE PERMITA LA VALIDACIÓN DE PRÁCTICAS DE DESARROLLO DE SOFTWARE BASADA EN EL ESTÁNDAR DE ESSENCE DE OMG

## RESUMEN

La ingeniería de software cuenta desde sus inicios con diversas metodologías y prácticas de desarrollo de software que buscaban obtener el mejor resultado en el menor tiempo y con la mayor calidad, esto ha permitido que existan diferentes variantes de metodologías con sus respectivas prácticas, lo anterior evidencia la disponibilidad de un gran catálogo de prácticas de desarrollo de software para seleccionar al momento de ejecutar un proyecto de desarrollo de software, lo cual se convierte en un hito importante del proyecto , ya que ello influye en el éxito o fracaso del resultado final. En éste trabajo se propone y desarrolla el prototipo funcional de una herramienta que permite documentar y representar de manera textual prácticas de desarrollo de software usando los elementos definidos en el estándar ESSENCE[1] de OMG. Dicha herramienta permite además la evaluación de la práctica que se representa, mediante la definición de criterios de validación parametrizados. Como resultado de este trabajo se logra la puesta en funcionamiento del prototipo web llamado SEMOVA (*SEMAT Modeler Validator*) que permitió representar prácticas de desarrollo de software, alineado a las restricciones y posibles relaciones entre elementos que se definen en el estándar ESSENCE. Las prácticas usadas para probar el prototipo fueron tomadas de representaciones hechas en artículos de investigación, adicionalmente se crearon los respectivos criterios de validación para cada práctica, ofreciendo la posibilidad de ser evaluadas por el público objetivo mediante un acceso web. Éste trabajo siembra las bases para la construcción de una plataforma que facilite el entendimiento y promueva el uso del estándar Essence en la industria y en la academia.

# Tabla de Contenido

INTRODUCCIÓN.....	8
OBJETIVO GENERAL.....	9
OBJETIVOS ESPECIFICOS.....	9
MARCO TEÓRICO.....	9
SEMAT.....	10
ESSENCE.....	10
Áreas del conocimiento.....	10
Alphas.....	11
Espacio de actividades.....	11
competencias.....	12
Mongo DB.....	13
JSON.....	13
Vue.js.....	14
HTML.....	14
CSS.....	14
JavaScript.....	15
Node.js.....	15
Express.js.....	15
GitHub y Git.....	16
SCRUM.....	16
METODOLOGÍA.....	16
Búsqueda de información.....	17
Entendimiento de la iniciativa SEMAT y la especificación del estándar ESSENCE de OMG....	17
Elementos del lenguaje gráficos.....	19
Identificación de la necesidad para enfocar el alcance del prototipo.....	20
Definición e implementación del prototipo.....	20
Requisitos funcionales accesos.....	20
Requisitos funcionales maestros.....	20
Requisitos funcionales modelado.....	20
Requisitos funcionales validación.....	20
Requisitos no funcionales generales.....	21
Arquitectura del prototipo.....	21
Lenguaje de programación, tecnologías y frameworks.....	22
Tecnologías.....	22
Frameworks.....	22
Lenguajes de programación.....	23

Otros lenguajes.....	23
Modelo del domino del prototipo.....	23
Elementos básicos del lenguaje ESSENCE.....	23
Modelo validación de prácticas.....	25
Modelo cuentas de usuarios.....	26
Documentación técnica y de configuración del prototipo.....	27
ANÁLISIS Y RESULTADOS.....	27
Menú de navegación y área de trabajo.....	27
Ingreso a la aplicación prototipo.....	27
Registro de nuevo usuario.....	28
Actualización de datos de usuario.....	28
Creación de elementos del núcleo del lenguaje de ESSENCE.....	29
Modelado textual de prácticas.....	29
Configuración de criterios de validación.....	32
Representación de la práctica.....	33
Validación de la práctica.....	34
Pruebas reales del prototipo con representaciones graficas de público objetivo.....	35
Prácticas modeladas.....	35
CONCLUSIONES.....	42
TRABAJO FUTURO.....	43
REFERENCIAS BIBLIOGRAFICAS.....	43
ANEXOS.....	45

## Listado de figuras

Figura 1. Áreas de conocimiento en las que se divide el núcleo[1].....	11
Figura 2. Representación gráfica de relación dentro de un Alpha[1].....	11
Figura 3. Espacios de actividades de Essence[1].....	11
Figura 4. Competencias definidas en el núcleo[1].....	12
Figura 5. La clave del lenguaje de ESSENCE [11].....	13
Figura 6. Reactividad de Vuejs [16].....	14
Figura 7. SCRUM Task board ejemplo[23].....	16
Figura 8. Diagrama causa efecto.....	20
Figura 9. Arquitectura prototipo SEMOVA.....	22
Figura 10. Diagrama de paquete general de los elementos del lenguaje. [1, p71].....	23
Figura 11. Elementos genéricos del lenguaje. [1, p73].....	24
Figura 12. Elementos del lenguaje tipo contenedores [1. p73].....	24
Figura 13. Elementos relacionados con los ALPHAS. [1, p87].....	25
Figura 14. Elemento para representar las competencias. [1, p102].....	25
Figura 15. Diagrama estructura de una validación.....	26
Figura 15.1 Diagrama de clases para la validación de prácticas.....	26
Figura 16. Diagrama de clases para una cuenta de usuario.....	27
Figura 17. Sección del panel de navegación y área de trabajo.....	27
Figura 18. Área de ingreso a la aplicación.....	28
Figura 19. Ventana de registro de nuevos usuarios.....	28
Figura 20. Ventada de actualización de información de cuenta de usuario.....	28
Figura 21. Administración de áreas del conocimiento.....	29
Figura 22. Administración de ALPHAS kernel.....	29
Figura 23. Administración de espacios de actividades del núcleo del lenguaje.....	29
Figura 24. Administración de competencias del núcleo del lenguaje.....	29
Figura 25.1. Modelado textual de prácticas con sus elementos básicos del lenguaje ESSENCE. 30	
Figura 25.2 Modelado textual de prácticas con sus elementos básicos del lenguaje ESSENCE.. 30	
Figura 26. Administración de ALPHAS para la práctica en modelada.....	31
Figura 27. Administración de productos de trabajo para la práctica modelada.....	31
Figura 28. Administración de espacios de actividades y actividades de la práctica modelada... 32	
Figura 29. Administración de criterios de validación para las prácticas modeladas.....	33
Figura 30. Representación de la práctica modelada.....	34
Figura 31. Validación de la práctica bajo los criterios definidos.....	35
Figura 32. Fase de análisis de la práctica gestión de requisitos.....	36
Figura 33. Fase de diseño de la práctica gestión de requisitos.....	36
Figura 34. Representación textual en SEMOVA.....	37

Figura 35. Representación área de proceso de CMMI.....	38
Figura 36. Representación textual en SEMOVA.....	39
Figura 37. Representación de un área de proceso de CMMI.....	39
Figura 38. Representación textual en SEMOVA.....	40
Figura 39. Representación pair programming en ESSENCERY.....	40
Figura 40. Representación textual en SEMOVA.....	41
Figura 41. Ingreso de valores de criterios en SEMOVA.....	41
Figura 42. Visualización de resultados de evaluación del criterio en SEMOVA.....	42

## INTRODUCCIÓN

Para este proyecto es fundamental tener una idea clara sobre ESSENCE el cual es un estándar mantenido por la OMG<sup>1</sup> (Object Management Group), derivado de la iniciativa SEMAT propuesto por Jacobson et Al [1], que permite representar prácticas de ingeniería de software usando un lenguaje y un conjunto de elementos que representan todo lo relacionado con la teoría y el método en la ingeniería de software. Además, en [2] se discute sobre la necesidad de una “teoría general de la ingeniería de software”, del mismo modo, se debate acerca de la necesidad de definir apropiadamente las prácticas de desarrollo de software, de manera que se asegure su comprensión y se garantice su implementación [3]. Para modelar las prácticas de ingeniería de software usando el estándar ESSENCE existen plantillas en herramientas como Microsoft Visio, PowerPoint y herramientas comerciales de uso privativo como ESSENCE Enterprise 365<sup>2</sup>. Adicionalmente, desde la academia se han derivado iniciativas menos robustas como: SematAcc [4, p.] (SEMAT Accelerator), donde se logra representar los estados de los ALPHA (Abstract-Level Progress Health Attribute) individuales y el estado del proyecto en términos de los ALPHA (Abstract-Level Progress Health Attribute). La herramienta Essencery [5], la cual permite graficar elementos usando la sintaxis gráfica del estándar ESSENCE para alcanzar como salida un gráfico con los elementos que conforman la práctica.

En relación con esto, se debe disponer de una herramienta que permita representar estas prácticas de desarrollo de software, configurar uno o más criterios de validación, para someterlas a validación por parte de personas externas bajo los criterios previamente configurados, y así obtener retroalimentación por parte de los usuarios sobre la representación de una práctica de desarrollo de software, pero lo más importante es su utilidad y los resultados obtenidos en su implementación, lo cual será útil para la toma de decisiones que ayuden a depurar las prácticas mismas o las herramientas/elementos disponibles para su representación. Lo anterior se puede considerar un aporte al proceso de ingeniería de software para promover la construcción de software de calidad de manera eficiente y bajo las prácticas que mejor resultado aporten a un proyecto de software en específico [6]. El problema que se pretende mitigar es el modelado, representación textual y configuración de criterios de validación de cualquier práctica existente y futura de desarrollo de software, lo que hace necesario tener una herramienta que ayude a fundamentar el proceso de la ingeniería de software, mediante prácticas de desarrollo de software que tengan una base teórica sólida y sean validadas por expertos bien sea en la academia o el mundo empresarial, apoyando así en cierto modo el objetivo principal de la iniciativa SEMAT

- 1 Consorcio de estándares tecnológicos internacional de membresía abierta, que tiene como objetivo desarrollar y revisar estándares conforme la tecnología avanza a través de los años y de la cual hacen parte usuarios finales, compañías privadas, agencias de gobiernos, universidades y centros de investigación.
- 2 Solución de Ivar Jacobson International que permite documentar y tener acceso a diferentes prácticas de desarrollo de software en forma de librerías.



(Software Engineering Method and Theory)[7], de manera que no usemos el ensayo y error como principal fuente de experiencia para determinar la efectividad de dicha metodología sobre un proyecto de software en un escenario específico[8], sino que usemos bases sólidas y probadas que promuevan la rigurosidad de la ingeniería de software como disciplina.

Para lograr el objetivo propuesto con este proyecto, se pretende abordar la base teórica propuesta en el estándar ESSENCE, con el fin de comprender el modelo del dominio, de esta manera lograr realizar un análisis del prototipo a construir, definir sus funcionalidades mínimas e ir implementando de manera iterativa, usando en lo posible algunos elementos propuestos en SCRUM[9] como lo son, BACKLOG[9] para rastrear las historias de usuarios o requerimientos funcionales y no funcionales, tablero con columnas de estado para revisar el estado de las historias de usuario o tareas, SPRINTS[9] para agrupar las historias de usuarios en bloques de entregas e incrementos de versiones para hacer efectiva la separación de la iteración tanto en el repositorio de código como el despliegue de la versión del SPRINT del prototipo. En este informe se muestra toda la documentación respectiva al proyecto realizado, como sus fundamentos teóricos, estado del arte, metodología, resultados y conclusiones.

## **OBJETIVO GENERAL**

Facilitar el proceso de modelado, entendimiento y validación de prácticas de desarrollo de software, mediante su representación en una herramienta de software web que se base en el estándar ESSENCE de OMG.

## **OBJETIVOS ESPECIFICOS**

- Revisar literatura científica relacionada con SEMAT y el estándar ESSENCE de OMG para identificar formas de representación, modelado y uso del lenguaje.
- Identificar antecedentes de iniciativas y herramientas de software que permitan realizar modelado en SEMAT basado en el estándar ESSENCE de OMG.
- Definir e implementar un prototipo funcional para la representación de prácticas de desarrollo de software basado en el estándar ESSENCE de OMG.
- Representar prácticas de desarrollo de software en dicha herramienta y realizar validación con público objetivo.

## **MARCO TEÓRICO**

A continuación se presentan los conceptos teóricos que sustentan el desarrollo de éste trabajo, comenzando por algunas definiciones referentes al movimiento Semat, el estándar Essence y los elementos del estándar comúnmente usados para modelar.

Posteriormente, se definen algunos conceptos técnicos referentes a la tecnologías usadas para el desarrollo del prototipo.

### **SEMAT**

SEMAT es una iniciativa para re-modelar la ingeniería del software de modo que la ingeniería de Software califique como una disciplina rigurosa[8]. La iniciativa se concibió como un esfuerzo de varios años para cerrar la brecha entre la comunidad de desarrolladores y la comunidad académica, y también, para crear una comunidad que de valor a toda la comunidad del software.

### **ESSENCE**

Es un estándar de OMG (Object Management Group) propuesto por Jacobson et .al [1] derivado de la iniciativa SEMAT, que permite representar prácticas de ingeniería de software usando un lenguaje definido para tales fines. También en otras fuentes se define como: “Kernel y lenguaje para métodos de ingeniería de software”. El kernel o núcleo proporciona el terreno común para definir las prácticas de desarrollo de software. Incluye los elementos esenciales que siempre prevalecen en cada esfuerzo de ingeniería de software, como los requisitos, el sistema de software, el equipo y el trabajo. Estos elementos tienen estados que representan el progreso y la salud, por lo que a medida que avanza el esfuerzo, los estados asociados con estos elementos progresan [10].

El núcleo se describe usando un pequeño subconjunto de elementos del lenguaje, el cual está organizado en tres áreas del conocimiento y estas a su vez contienen un pequeño número de los siguientes elementos:

#### **Áreas del conocimiento**

El núcleo es organizado en tres áreas del conocimiento, cada una enfocada en un aspecto específico de la ingeniería de software [1] como se muestra en la figura 1. Abarcando así todo lo que tiene que ver con el uso y explotación del sistema de software que será producido, con la especificación y desarrollo del sistema de software y con el equipo y la forma en que ejecuta el trabajo por hacer.

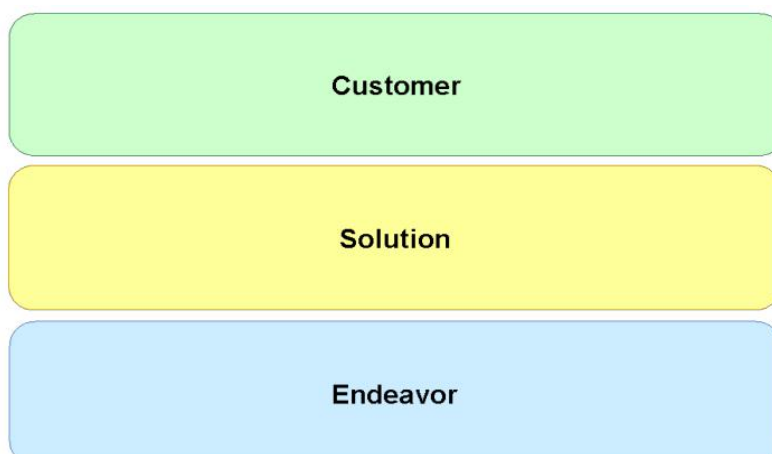


Figura 1. Áreas de conocimiento en las que se divide el núcleo[1]

### Alphas

Representan las cosas con las que siempre se trabaja y en el núcleo se definen siete y se clasifican y relacionan como se muestra en la siguiente figura 2.

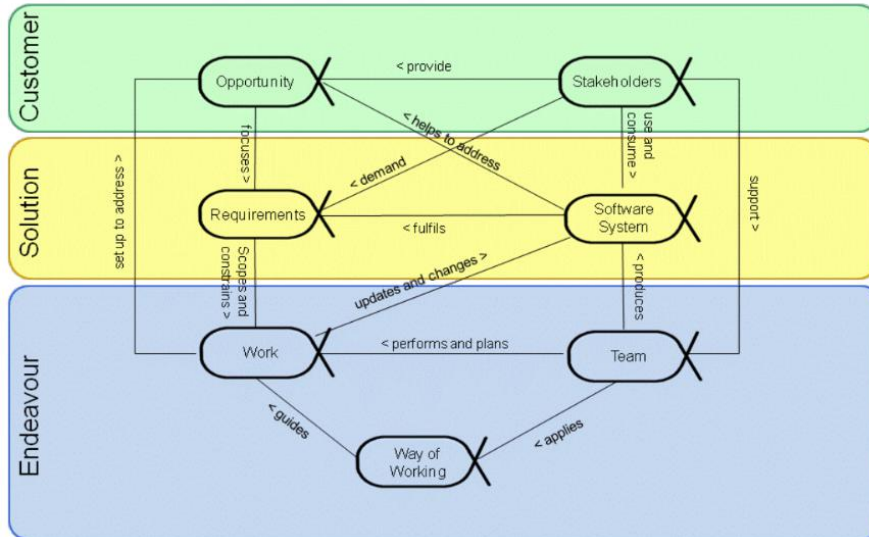


Figura 2. Representación gráfica de relación dentro de un Alpha[1]

### Espacio de actividades

El núcleo también proporciona un conjunto de espacios de actividades que complementan los Alphas para así poder proporcionar una vista basada en actividades de Ingeniería de software, los espacios de actividad son los siguientes como se muestra en la figura 3.

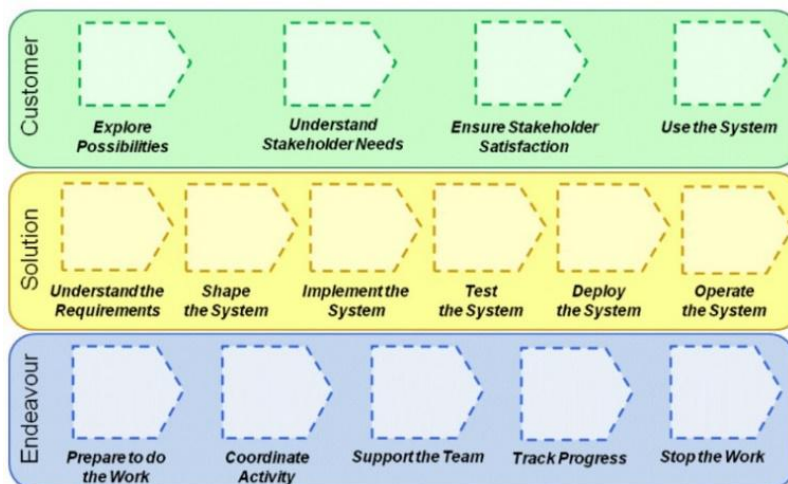


Figura 3. Espacios de actividades de Essence[1]

## competencias

Acá se describen las competencias necesarias para desarrollar o ejecutar ciertas actividades, el núcleo define para la ingeniería del software 6 competencias básicas, las cuáles se pueden apreciar en la siguiente figura 4.



Figura 4. Competencias definidas en el núcleo[1]

En síntesis, la especificación del estándar ESSENCE se representa con la figura 5, la cual muestra a muy alto nivel todos los elementos del lenguaje y las interacciones entre ellos, lo cual provee un mejor entendimiento de las situaciones enfrentadas en todo proyecto de ingeniería de software.

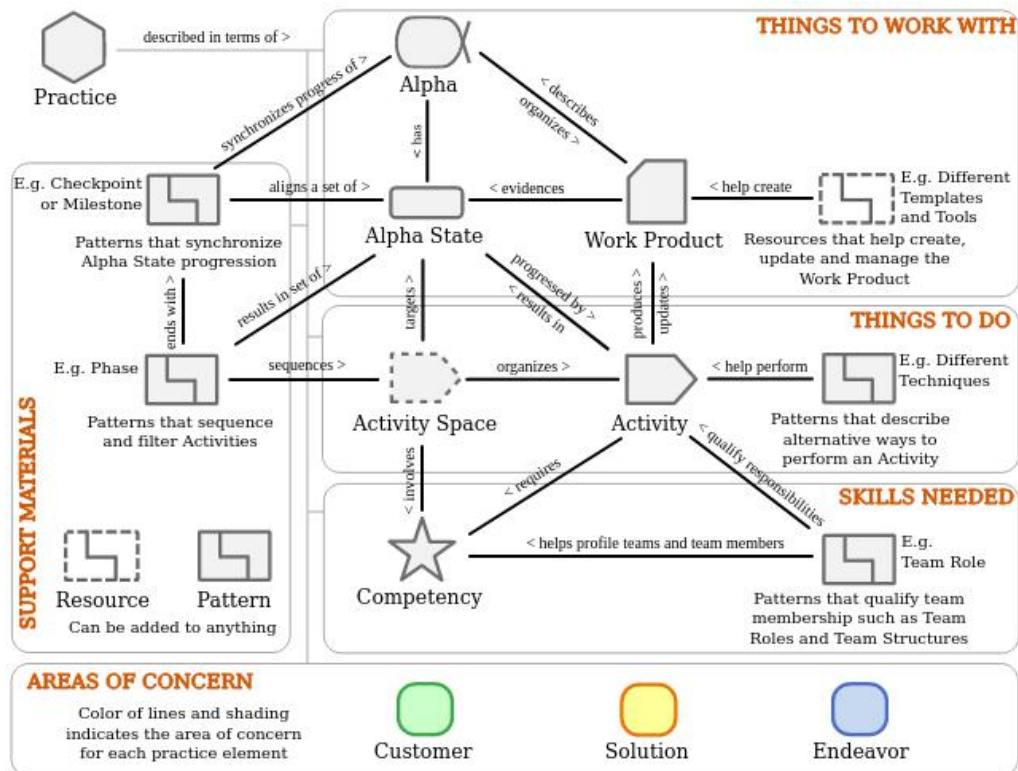


Figura 5. La clave del lenguaje de ESSENCE [11]

## Mongo DB

Es un tipo de base de datos basada en documentos, la cual ofrece grandes beneficios tales como escalabilidad y una gran flexibilidad, además de contar con un modelo para consultas e indexación.

Se puede trabajar gratis en la nube MongoDB Atlas y también se puede descargar un servidor local de MongoDB. Se escoge MongoDB porque su modelo de documentos es muy fácil de aprender y usar, y proporciona a los desarrolladores todas las funcionalidades necesarias para cumplir con los requisitos más complejos actuales de la conexión y alojamiento de una base de datos a un software. Además, existen drivers para más de 10 lenguajes, y entre la comunidad se desarrollan muchos más [12].

Cabe recalcar que MongoDB es una base de datos documental, lo que significa que ella almacena datos en documentos de tipo JSON. Es un tipo de base de datos que se diferencia de las tradicionales por no conformarse de relaciones ni tablas, por lo que este tipo es mucho más expresivo y potente [13].

## JSON

Es un formato de texto suave y liviano para el intercambio de datos entre partes de una arquitectura de software. Es un subconjunto de notación de objetos en JavaScript [14], y debido a que es una fuerte opción frente a XML, se consideró en el 2019 un formato independiente del lenguaje JavaScript.

Una de sus grandes ventajas es que es muy sencillo escribir analizadores sintácticos. JSON se emplea habitualmente en entornos donde el tamaño del flujo de datos entre el cliente y el servidor es muy importante.

## Vue.js

Es un framework progresivo, el cuál fue diseñado para construir interfaces de usuario. A diferencia de otros frameworks, este fue diseñado desde cero, para que se use incrementalmente. La librería está enfocada solo en la capa de la visualización y es muy fácil de utilizar y de integrar con otras librerías o proyectos que se deseen [15].

¿Por qué Vue? Primeramente, por la curva de aprendizaje y ahorro de tiempo al desarrollar, además de estar en línea con las herramientas usadas actualmente en el desarrollo de software web.

Vue cuenta con una característica importante, llamada: Reactividad, ver figura 6, la cual se le denomina a la capacidad de actualización automática a medida que se vayan generando modificaciones en los componentes, lo cual la hace perfecta para la construcción de una SPA (Single Page Application).

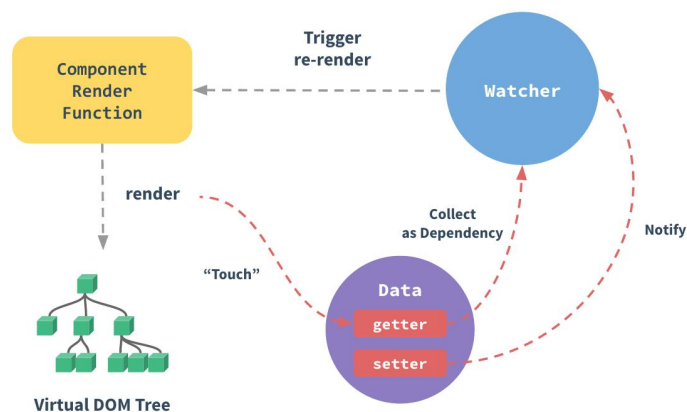


Figura 6. Reactividad de Vuejs [16]

## HTML

HTML (HyperText Markup Language) como sus siglas lo describen; es un lenguaje de marcado de hipertexto, y le permite al usuario crear y estructuras secciones visuales de una página web y aplicaciones. HTML no es un lenguaje de programación, lo que significa que no tiene la capacidad de crear dinamismos dentro de una página web.

## CSS

Se denomina CSS al lenguaje de hojas de estilo en cascada, las cuales se usan para dar estilo a elementos escritos en HTML. CSS realiza la separación entre el contenido de la

página web su representación visual. Las hojas de estilo en cascada (cuya abreviatura, CSS, no debe confundirse con XSS) controlan el diseño de un sitio web para varios medios. Se puede cambiar el tamaño de una página web o modificarla en función de si se procesa en un navegador, un teléfono móvil o se envía a una impresora [17, p. 2].

## **JavaScript**

Es un lenguaje de programación o de secuencias de comandos que permiten implementar funciones en páginas web. Permite crear contenido de actualización dinámica, controlar multimedia, animar imágenes y prácticamente todo lo que se pueda realizar en una página web [18]. También se define como un robusto lenguaje de programación que se puede aplicar a un documento HTML y usarse para crear interactividad dinámica en los sitios web. Fue inventado por Brendan Erich, cofundador del proyecto Mozilla, Mozilla Foundation y la Corporación Mozilla [19].

## **Node.js**

Es un entorno en tiempo de ejecución multiplataforma, basado en el código abierto, para la capa del servidor basado en el lenguaje de programación JavaScript, en una arquitectura orientada a eventos y está basado en el motor de Google V8. Se creó con el objetivo de ayudar a crear programas de red con ventajas de escalabilidad y durabilidad. Su evolución está apadrinada por la empresa Joyent. Unos de sus beneficios son: Su compilación se realiza en el tiempo de ejecución, gran escalabilidad, fácil expansión debido al NPM (Node Package Manager), alto rendimiento en proyectos donde se necesita ejecución en tiempo real, en startups se puede realizar un desarrollo Full-Stack y hasta aplicativos móviles con el mismo lenguaje [20].

## **Express.js**

Es un framework para Node.js que es utilizado para crear aplicaciones web en un menor tiempo, ya que proporciona funcionalidades como el enrutamiento y middlewares<sup>3</sup>, opciones para gestionar sesiones y cookies, entre muchas otras. Express está basado en connect, que a su vez es un framework basado en HTTP para Node.js [21].

3 Son funciones que se ejecutan durante el ciclo de vida de una petición http, esta función tiene acceso al objeto REQUEST (req), RESPONSE (res) y NEXT (next) y en dicha función se pueden manipular y controlar dichos objetos para diferentes fines de acuerdo con la necesidad.

## GitHub y Git

Git es un software de control de versiones para desarrolladores, el cual se instala en el ordenador, y desde una consola se pueden desplegar todas sus funcionalidades. Por otra parte, GitHub es una plataforma de repositorio de código abierto basados en la nube para que varios desarrolladores puedan trabajar en un solo proyecto y ver las ediciones de cada autor en tiempo real. En resumen, GitHub es una plataforma basada en la web, la cual incorpora características de control de versiones de Git, para que estas, se puedan utilizar de forma colaborativa.

## SCRUM

Actualmente, en el desarrollo de software existen diversos procesos para llevar a cabo esta función, uno de ellos es SCRUM, y en este proyecto se usaron algunas prácticas de esta metodología para llevar a cabo el desarrollo del prototipo.

Los elementos que se usaron en el desarrollo del prototipo fueron el BACKLOG, y el tablero con columnas típicas de estado (to do, in progress, done) de las pequeñas actividades en las que se divide todo el proceso para construir el prototipo. BACKLOG: El producto BACKLOG de un proyecto que sigue un proceso SCRUM se basa en una lista donde están los requerimientos iniciales del producto que se va a desarrollar. Se diferencia en que ésta es una lista dinámica que a medida que vaya avanzando el desarrollo, irá evolucionando de acuerdo a las necesidades y el entorno del proyecto. Su principal objetivo es ver las necesidades del producto para sí, llegar a obtener la máxima utilidad. También contiene las tareas y sub tareas de cada requerimiento, las cuales se organizan de acuerdo a su prioridad, y ayudan a estimar los tiempos de desarrollo de cada requisito [22]. SCRUM TASKBOARD: Este tablero se encarga de gestionar la lista de objetivos a realizar(BACKLOG), el cuál actúa como un espejo del proceso y generador de información y se divide en cuatro columnas: NOT PLANNED (No planeado), NOT STARTED (Sin comenzar), IN PROGRESS (En ejecución) y DONE(Hecho), donde en cada columna se van pegando o escribiendo las tareas desplegadas de los requisitos [23].

	NOT STARTED	IN PROGRESS	DONE
NOT PLANNED			
CONTINUOUS IMPLEMENT			

Figura 7. SCRUM Task board ejemplo[23]

## METODOLOGÍA



## **Búsqueda de información**

En el ámbito de la investigación, fue necesaria la revisión y análisis profundo de fuentes confiables en la bibliografía publicada. Identificar la literatura relevante, pertinente y confiable de manera eficiente es posible, a pesar de la gran cantidad de información disponible, algunas fuentes secundarias confiables y especializadas son los libros, revistas, tesis, ensayos, artículos científicos. Para el desarrollo de los dos primeros objetivos específicos, se revisó material publicado referente a SEMAT y ESSENCE y todo lo relacionado con el proyecto, la búsqueda también se centró mayormente en los manuales oficiales de los creadores y desarrolladores de ESSENCE y SEMAT, por lo que la muestra está limitada en este pequeño grupo, y la búsqueda de antecedentes de iniciativas y herramientas de software que permitan realizar modelado en SEMAT basado en el estándar ESSENCE, se tomó como punto de partida los manuales de ESSENCE y SEMAT, a partir de ahí, para dar desarrollo al primer objetivo, se comenzaron a encontrar artículos relacionados con estos temas, en donde se encontraron fundamentaciones teóricas, representaciones gráficas y proyectos de estudios e investigación similares al tema.

Para la búsqueda de antecedentes de iniciativas y herramientas de software que permitan realizar modelado en SEMAT basado en el estándar ESSENCE, se comenzó la búsqueda en las bases de datos de IEEE, y la revista SCIELO, apoyado también en motores de búsqueda como Google Scholar, de ahí partieron los artículos principales, y en base a las referencias de los mismos se encontraron los demás resultados de búsqueda. Los criterios para seleccionar los artículos partieron del análisis de su contenido, priorizando los que particularmente hacían uso del estándar Essence, abordaban la iniciativa Semat y como factor importante, aquellos que tuviesen representaciones gráficas que se pudieran usar para probar los resultados de éste trabajo.

## **Entendimiento de la iniciativa SEMAT y la especificación del estándar ESSENCE de OMG**

Para abordar técnicamente la especificación fue necesario estudiar el modelo del domino que se definió en el estándar ESSENCE [1] de OMG, información que se desglosa a continuación.

Todos los elementos del lenguaje contienen la siguiente información relevante:

- Título  
Nombre formal del elemento del lenguaje
- Descripción  
Descripción corta del elemento

- Generalizaciones
  - Lista de cada padre o superclase del elemento.
- Atributos
  - Lista de cada atributo definido para el elemento, cada atributo se especifica con su nombre formal, tipo de dato, y multiplicidad.
  - Literal
  - Booleano
  - Numérico
  - Elemento gráfico
  - Asociaciones: Identifica todas las asociaciones que son propiedad del elemento.
  - Invariante
    - Describe las reglas bien formadas para el elemento en formato de texto libre con expresiones OCL (Object Constraints Language) usado en UML.
  - Operaciones adicionales
    - Describe alguna operación adicional necesaria en formato de texto y OCL.
  - Semántica
    - Provee una descripción detallada del elemento en lenguaje natural.
- Los elementos del lenguaje son agrupados en seis paquetes, de la siguiente manera:
  - Fundamentales
    - La intención de este elemento es proporcionar todos los elementos base, incluidas las superclases abstractas, las cuales son necesarias para formar una base de referencia en el lenguaje.
  - Alphas y productos de trabajo.
    - Contiene los elementos básicos para formar prácticas mínimas. Un modelo de dominio para los esfuerzos de la ingeniería de software puede ser creado. Ninguna actividad puede ser expresada usando esta capa, pero productos de trabajo pueden ser relacionados a elementos abstractos del dominio.
  - Espacios de actividades y actividades

Contiene los elementos para enriquecer las prácticas expresado en forma de actividades.

- Competencias

Contiene los elementos para soportar la especificación de las competencias.




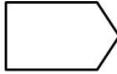


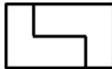
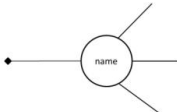
- Tipos definidos por el usuario

Contiene elementos para enriquecer elementos simples del grupo o paquete fundamental con información de tipos. Esto no hará parte del alcance de este trabajo.

- Vista

Contiene elementos para soportar la especificación del contenido de la vista. Esto no hará parte del alcance de este trabajo.

### Elementos del lenguaje gráficos

PRÁCTICA	
ASOCIACIÓN	
ALPHA	
PRODUCTO DE TRABAJO	
ACTIVIDAD	
ESPACIO DE ACTIVIDADES	
COMPETENCIA	
PATRÓN	
ASOCIACIÓN DE PATRÓN	

## Identificación de la necesidad para enfocar el alcance del prototipo

Para lograr definir un alcance y acotar la solución se usó un diagrama de causa efecto figura 8. El cual ayudó a identificar parte de los requerimientos funcionales claves que son soportados por el prototipo.

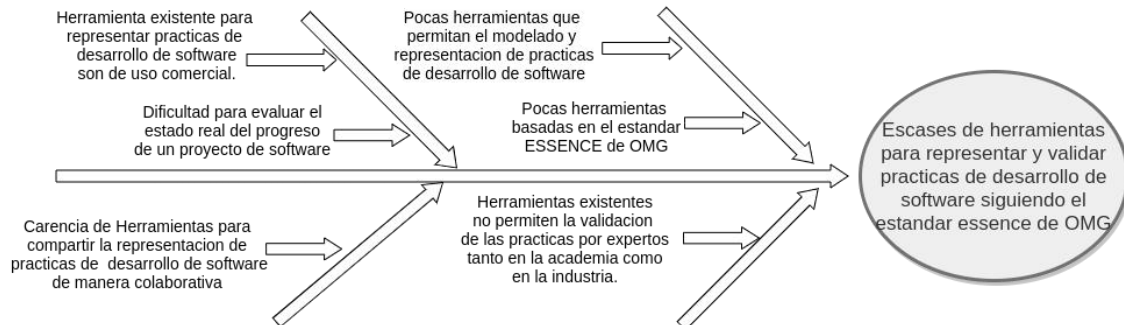


Figura 8. Diagrama causa efecto

## Definición e implementación del prototipo

El prototipo se dividió en cinco partes o módulos importantes con sus respectivos requerimientos funcionales: Accesos, Maestros, Modelado, Configuración de criterios y Validación.

### Requisitos funcionales accesos

- Permitir el registro de un usuario ingresando su correo electrónico, nombre y una contraseña.
- Permitir al usuario registrado ingresar usando su correo y contraseña registrada.
- Permitir al usuario cambiar sus datos.

### Requisitos funcionales maestros

- Permitir al usuario crear los elementos básicos del lenguaje definidos en el estándar ESSENCE como: Areas of Concern, ALPHAS, Activity Spaces y Competencias.

### Requisitos funcionales modelado

- Permitir al usuario modelar de manera textual y guiada prácticas de desarrollo de software utilizando los elementos básicos del núcleo del estándar ESSENCE.
- Permitir al usuario crear nuevos elementos propios de cada práctica de desarrollo de software como productos de trabajo, actividades y patrones.

### Requisitos funcionales validación

- Permitir al usuario definir por cada práctica de desarrollo de software previamente creada criterios de validación.

- Permitir al usuario por cada criterio de validación definir unas variables con su respectivo símbolo y significado, una expresión matemática que contenga dichas variables para luego ser computadas y arrojar un resultado numérico.
- Permitir al usuario crear solicitudes de validación por cada práctica creada, para luego ser enviadas en forma de URL (Uniform Resource Locator) a público objetivo.
- Permitir al usuario que validara la práctica bajo los criterios predefinidos, acceder a la visualización e ingresar los valores de los criterios a evaluar, a través de la URL suministrada sin necesidad de autenticarse en la aplicación prototipo.
- Permitir al usuario autenticado descargar los resultados de evaluación de la práctica en formato texto fácil de importar en herramientas de hojas de cálculo para la finalidad futura deseada.

### **Requisitos no funcionales generales**

- Permitir acceder a la aplicación prototipo a través de un navegador web.
- Permitir almacenar la información generada en la aplicación en un motor de bases de datos.
- Permitir solo a los usuarios autenticados acceder a las opciones de maestros, modelado y configuración de criterios.
- Permitir a usuarios no autenticados acceso a la opción pública de validación de prácticas, y la opción de registro para crear una nueva cuenta de acceso.
- Guiar el proceso de creación de prácticas de desarrollo de software usando los conceptos y restricciones definidos en el estándar ESSENCE de OMG.

### **Arquitectura del prototipo**

El diseño de la aplicación se dividió en tres partes como se muestra en la figura 9, y continuación se describe cada componente.

1. Una aplicación REST (Representational State Transfer) API (Application Programming Interface) backend que provee acceso y modificación a los recursos del modelo del dominio a través de direcciones URI (Uniform Resource Identifier).
2. Una aplicación SPA (Single Page Application) frontend que se conecta al backend y permite el acceso y modificación de los datos del modelo del dominio a través de una interfaz gráfica accesible desde un navegador web.

- Y una persistencia usando un modelo de base de datos no relacional NoSQL que almacena la información en documentos con formato JSON (JavaScript Object Notation).

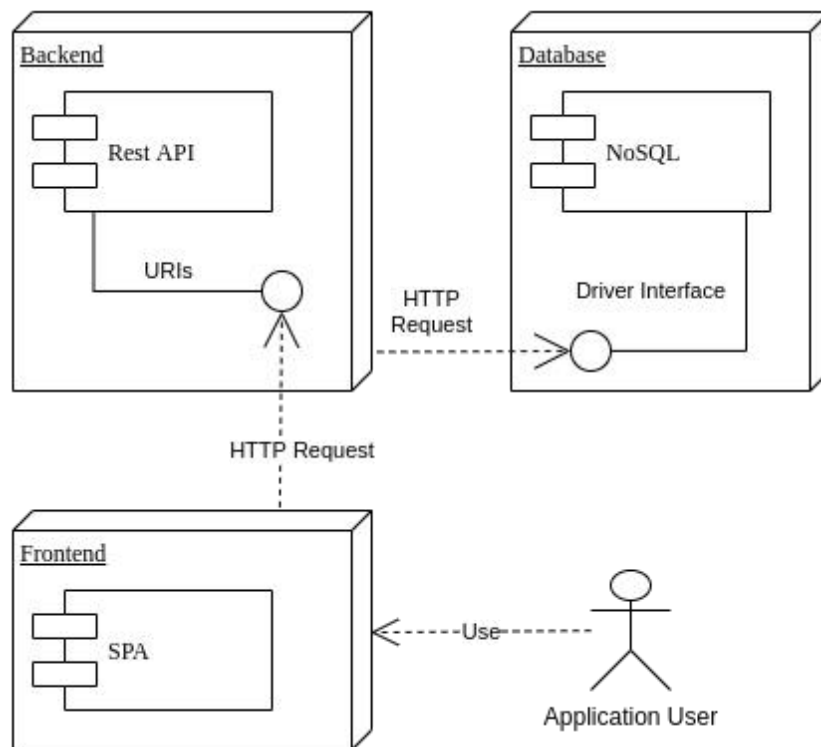


Figura 9. Arquitectura prototipo SEMOVA

## Lenguaje de programación, tecnologías y frameworks

En la implementación del prototipo se usó lo siguiente:

### Tecnologías

- MONGODB, herramienta usada para la persistencia de los datos generados por el prototipo.
- NODEJS, como herramienta que soportó el diseño de la API REST que expone las URIs para el acceso y modificación de los recursos definidos en el modelo del dominio del prototipo.
- GITHUB, como una herramienta que permitió la persistencia y versionado del código fuente generado para la construcción del prototipo de la aplicación.

### Frameworks

- EXPRESS el cual permitió de manera sencilla crear y configurar APIs REST.
- VUEJS permitió estructurar y crear las componentes en forma de SPA (Single Page Application) de manera sencilla y rápida usando componentes reutilizables y altamente configurables.

- BOOTSTRAP-VUE, permitió darle forma a los componentes gráficos para que el prototipo permitiera la interacción con los usuarios de manera fácil.

#### Lenguajes de programación

- JAVASCRIPT permitió programar todas las interacciones del usuario con la interfaz gráfica, las interacciones entre la interfaz gráfica SPA y la API REST y la interacción entre la API REST y el motor de bases de datos no relacional MongoDB.

#### Otros lenguajes

- HTML (HyperText Markup Language), Permitted definir la parte estructural de la SPA, distribuir los elementos en la pantalla y organizar los componentes de manera jerárquica.
- CSS (Cascade Style Sheet), permitió darle forma, color y algunos comportamientos visuales del prototipo para que la interacción con el usuario fuera fácil y sencilla.

### Modelo del domino del prototipo

El modelo del dominio representa los elementos básicos definidos en el estándar ESSENCE de OMG y la información del módulo de acceso y validación de las prácticas de desarrollo de software.

#### Elementos básicos del lenguaje ESSENCE

Los elementos definidos en el estándar ESSENCE están agrupados como lo muestra la figura 10. los cuales representan los espacios de actividades, actividades, ALPHAS, productos de trabajo, competencias y elementos genéricos.

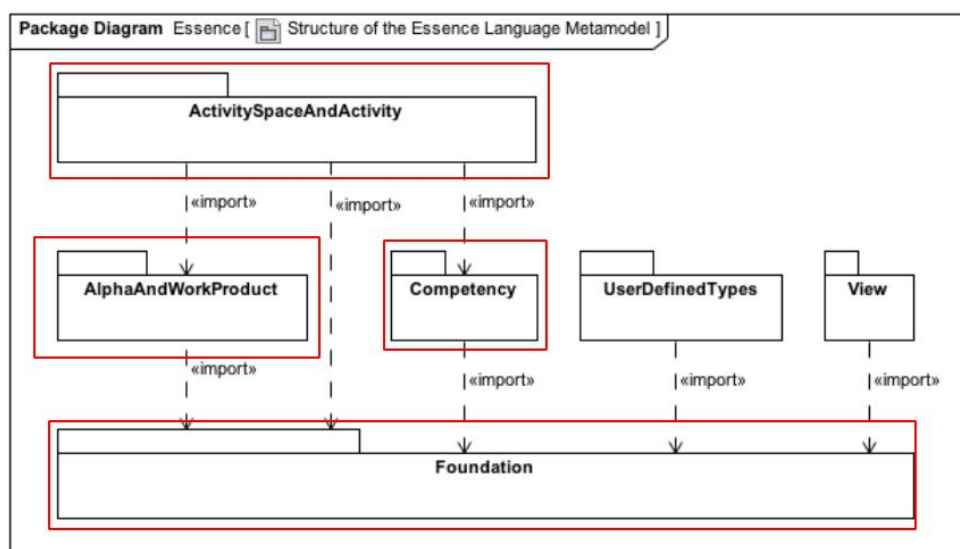


Figura 10. Diagrama de paquete general de los elementos del lenguaje. [1, p71]

El estándar define los siguientes elementos como genéricos, ver figura 11. Los cuales heredan sus atributos a elementos más especializados, como prácticas, patrones, alpha, espacio de actividades, productos de trabajo y competencias.

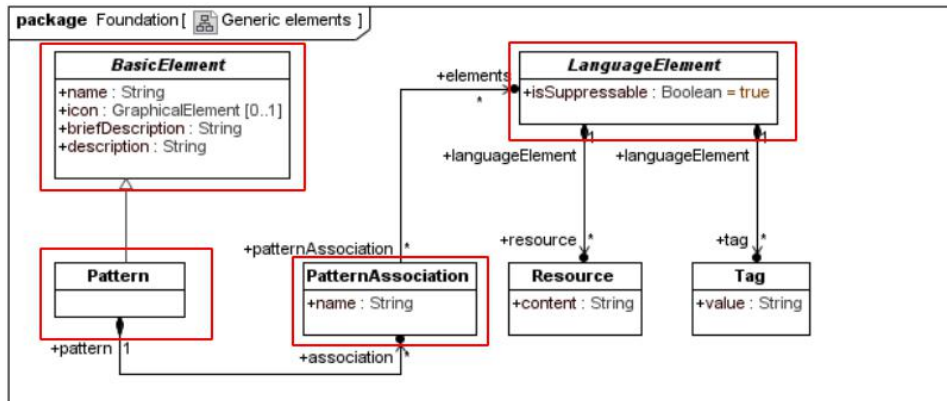


Figura 11. Elementos genéricos del lenguaje. [1, p73]

El elemento que representa y modela una práctica de desarrollo de software está definida por el lenguaje como se muestra en la figura 12.

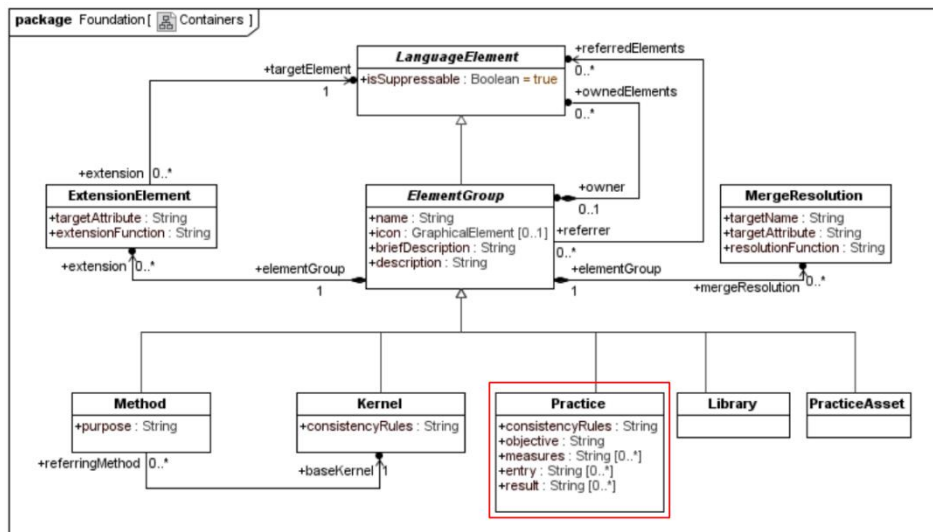


Figura 12. Elementos del lenguaje tipo contenedores [1. p73]

Los elementos que definen y modelan los elementos relacionados con los ALPHAS, como lo son productos de trabajo y manifiestos de productos de trabajos se encuentran en la figura 13.



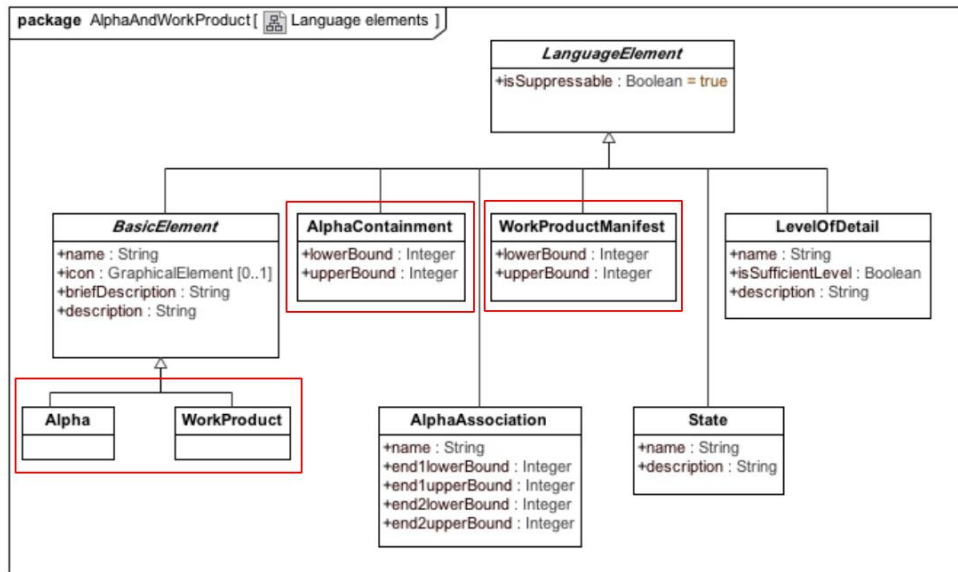


Figura 13. Elementos relacionados con los ALPHAS. [1, p87]

Finalmente, el modelo que representa una competencia se muestra en la figura 14.

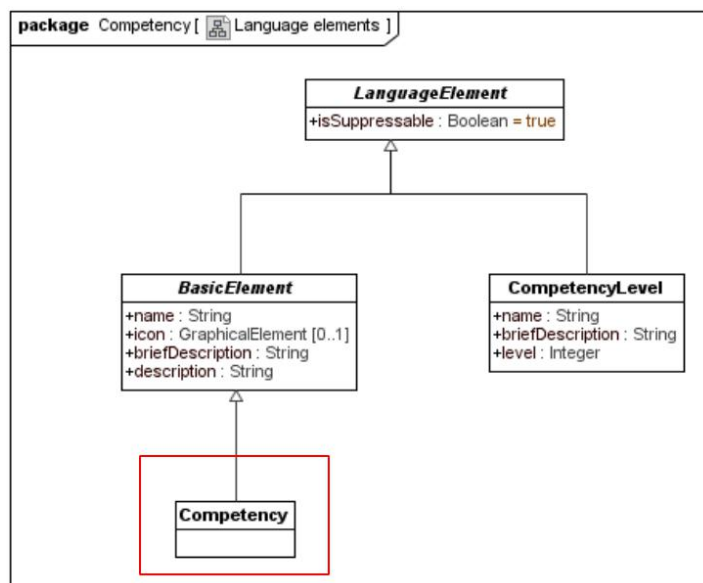


Figura 14. Elemento para representar las competencias. [1, p102]

Con los anteriores diagramas de paquete y clase se definieron los modelos del dominio que representan los elementos básicos del lenguaje del estándar ESSENCE.

### Modelo validación de prácticas

En la figura 15 y 15.1, se muestra el modelo del dominio y sus relaciones que tienen para la evaluación de prácticas de desarrollo de software.



Figura 15. Diagrama estructura de una validación.

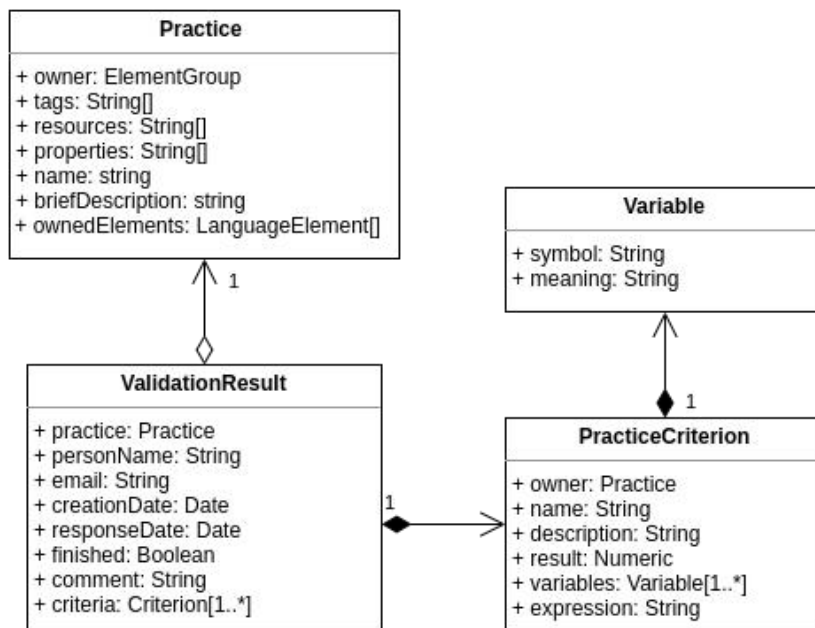


Figura 15.1 Diagrama de clases para la validación de prácticas.

### Modelo cuentas de usuarios

Finalmente, para modelar los accesos al sistema en la figura 16 se muestra el diagrama de clase que representa una cuenta de usuario del prototipo.

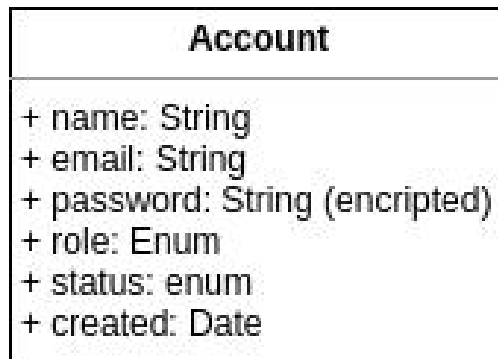


Figura 16. Diagrama de clases para una cuenta de usuario

### Documentación técnica y de configuración del prototipo

Ver anexos.

### ANÁLISIS Y RESULTADOS

El prototipo funcional construido cuenta con los siguientes elementos principales.

#### Menú de navegación y área de trabajo

Este componente del prototipo permite al usuario navegar fácilmente por las opciones que tiene disponible tanto a usuario no autenticado como usuarios autenticados, ver figura 17.

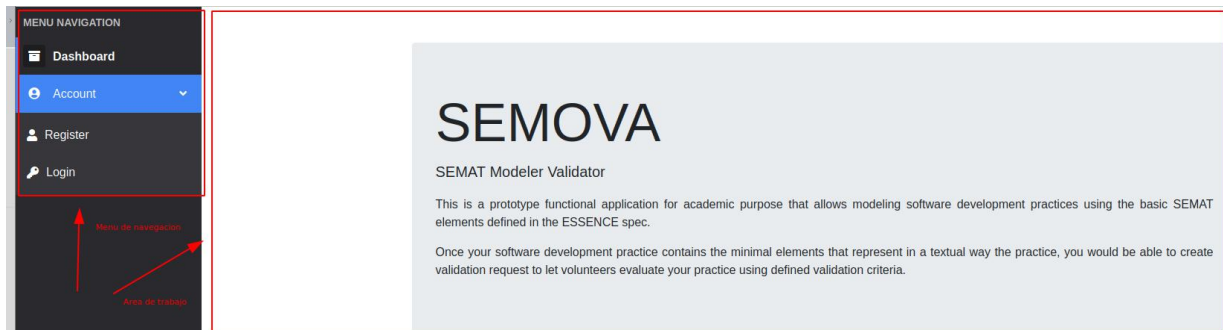
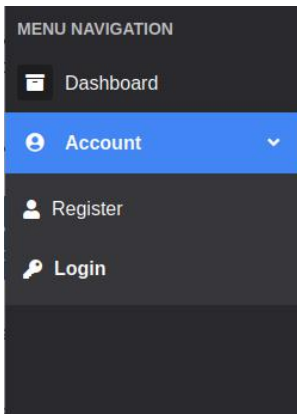


Figura 17. Sección del panel de navegación y área de trabajo.

#### Ingreso a la aplicación prototipo

Para ingresar a la aplicación es necesario ingresar las credenciales de acceso en la ventana de acceso, ver figura 18, previamente creadas cuando se registró un nuevo usuario ver figura 19.



Email  
john.doe@mail.com

Password

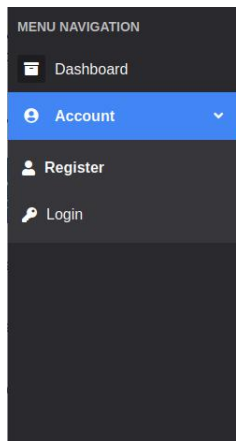
Sign In

Pantalla de inicio de sesión

Figura 18. Área de ingreso a la aplicación.

### Registro de nuevo usuario

Para crear el acceso a la aplicación se dispone de una opción de registro como se muestra en la figura 19.



Registration Form

what's your name?  
John Doe

what's your email?  
john.doe@mail.com

what's your password?

Register

Pantalla de registro de nuevo usuario

Figura 19. Ventana de registro de nuevos usuarios.

### Actualización de datos de usuario

Edit your profile information

Your name!  
George Wyman

Your email  
orvie.se@gmail.com

Save

Figura 20. Ventana de actualización de información de cuenta de usuario.

## Creación de elementos del núcleo del lenguaje de ESSENCE

Las figuras 21, 22, 23, 24 muestran la administración de los elementos básicos del lenguaje que se definieron en el alcance de la solución, estos elementos estarán disponibles para el modelado de todas las prácticas ya que hacen parte del núcleo del lenguaje de ESSENCE y son comunes a todas las prácticas de desarrollo de software.

Areas of concern

Name

Description

**Color Convention**

Index to order colors

Name	Description	Color Convention
Customer	This area of concern contains everything to do with the actual use and exploitation of the software system to be produced.	#a7fb8d
Solution	This area of concern contains everything to do the specification and development of the software system.	#fff062
Endeavor	This area of concern contains everything to do with the team, and the way that they approach their work.	#64b7f4

Figura 21. Administración de áreas del conocimiento

Kernel Alphas

Alpha Name

Alpha brief description

More complete description

Is kernel?

Area or concern

Listado de elementos creados

Name	Brief Description	Super Alpha	Is Kernel	Area Of Concern
Oportunity	The set of circumstances that makes it appropriate to develop or change a software system ssss		true	Customer
stakeholders	The people, groups, or organizations who affect or are affected by a software system		true	Customer
Epic	Epic	Oportunity	false	Customer

Figura 22. Administración de ALPHAS kernel

Activity Spaces

Activity Space Name

Description

Is kernel?

Area or concern

Listado de elementos creados

Activity	Description	Is Kernel	Area of Concern
Explore possibilities	Explore possibilities	true	Customer
Understand Stakeholder Needs	Understand Stakeholder Needs	true	Customer
Ensure Stakeholder Satisfaction	Ensure Stakeholder Satisfaction	true	Customer
Use the System	Use the System	true	Customer
Understand the requirements	Understand the requirements	true	Solution
Implement the System	Implement the System	true	Solution

Figura 23. Administración de espacios de actividades del núcleo del lenguaje.

Competencias

Competency Name

Description

Area or concern

Listado de elementos creados

Competency	Description	Is Kernel	Area of Concern
Stakeholder representation	Stakeholder representation	true	Customer
Analysis	Analysis	true	Solution
Development	Development	true	Solution
Testing	Testing	true	Solution
Leadership	Leadership	true	Endeavor
Management	Management	true	Endeavor

Figura 24. Administración de competencias del núcleo del lenguaje.

## Modelado textual de prácticas

En las figuras 25.1 25.2 podemos observar la ventana donde se ejecutan uno de los objetivos más importantes, y es el modelado de una práctica de desarrollo de software

usando los elementos del lenguaje ESSENCE, las figuras 26, 27, 28, muestran los diferentes pasos seguidos en el proceso de modelado, el proceso se subdivide en diferentes etapas donde el usuario va agregando los elementos del núcleo del lenguaje ESSENCE y los elementos propios de la práctica a modelar, como lo son: Sub-ALPHAS, productos de trabajo, manifiesto de productos de trabajo, actividades y patrones.

Editing practice: **Gestion de requisitos SIG**

Editing practice: <b>Gestion de requisitos SIG</b>						
Available Practices	Practice definition	Alphas	Work products	Work product manifest	Activities	Patterns
Name	Objective	Tags				
Pair Programming	Write code in pairs to share ideas, review the code and learn together	[ "Essencery" ]				
Gestion de requisitos SIG	Gestion de requisitos SIG	[ "Claudia Elena Durango Vanegas" ]				
Transformar las necesidades de las partes interesadas en requisitos de cliente	Transformar las necesidades de las partes interesadas en requisitos de cliente	[ "C. M. Zapata", "J. Valderrama and L. D. Jiménez" ]				
Asignar los requisitos de componente de producto	asignar los requisitos de componente de producto	[ "C. M. Zapata", "J. Valderrama and L. D. Jiménez" ]				

Figura 25.1. Modelado textual de prácticas con sus elementos básicos del lenguaje ESSENCE.

Available Practices	Practice definition	Alphas	Work products	Work product manifest	Activities	Patterns
---------------------	---------------------	--------	---------------	-----------------------	------------	----------

**Practice Name**

**Objective**

Gestion de requisitos SIG

The objective of this Practice, expressed as a concise and isolated phrase. The content of this attribute should be an explicit and short statement that describes the goal that the practice pursues.

**Tags**

Claudia Elena Durango Vanegas ✕

Enter desired tags separated by comma ','

**Resources**

Definición de buenas prácticas de desarrollo de sistemas de información geográfica utilizando el núcleo de Semat ✕

Figura 25.2 Modelado textual de prácticas con sus elementos básicos del lenguaje ESSENCE.

### Editing practice: **Gestion de requisitos**

Available Practices Practice definition **Alphas** Work products Work product manifest

Activities Patterns

**Alphas**

-- Please select an alpha -- Add New

Name	Description	Area Of Concern	
Requirements	Requirements	Solution	
Team	Team	Endeavor	

Things to work with

Alphas agregados a la practica actual en edicion.

Figura 26. Administración de ALPHAS para la práctica en modelada.

### Editing practice: **Gestion de requisitos**

Available Practices Practice definition Alphas **Work products** Work product manifest

Activities Patterns

Work Product Name

Description

Save Clear

Listado de productos de trabajo definidos para la práctica seleccionada y creada

Work Product	Description	
Contexto del proyecto SIG	Contexto del proyecto SIG	
Acta de aprobación de requisitos	Acta de aprobación de requisitos	
Viabilidad del proyecto	Viabilidad del proyecto	
NA	NA	
Diagramas UML	Diagramas UML	
Arquitectura del sistema	Arquitectura del sistema	

Figura 27. Administración de productos de trabajo para la práctica modelada.

## Editing practice: **Gestion de requisitos**

Available Practices Practice definition Alphas Work products Work product manifest

**Activities** Patterns

Activity spaces definidas por el estandar

**Activity Spaces to represent this practice**

-- Please select an activity space -- Add

Listado de espacios de actividades agregadas a la practica y que podran ser utilizadas para organizar las actividades propias de la practica

Available activity spaces for this practice to organize activities

Activity Space	Area of concern	
Understand the requirements	Solution	
Shape the system	Solution	
Use the System	Customer	

Available activity spaces for this practice

**Activities**

\*\* Select Activity Space \*\*

Select an activity space to group activities

Activity Name

Representative name for the activity

Stakeholder representation  Analysis  Development  Testing  
 Leadership  Management

Competency required to accomplish the activity (Optional)

Save Clear

Listado de actividades creadas en la practica.

Activity Space	Activity	Competencies (Level)	
Understand the requirements	Describir el proyecto SIG	Management 5	

Figura 28. Administración de espacios de actividades y actividades de la práctica modelada.

## Configuración de criterios de validación
















Otra parte muy importante como resultado de las opciones del prototipo es la de configurar los criterios de validación para las prácticas los cuales serán utilizados por los validadores al momento de validar la práctica, ingresando los valores para las variables definidas en cada criterio, la figura 29 describe las opciones para crear criterios, visualizar los criterios actuales de una práctica específica y la descarga de los resultados de todas las validaciones finalizadas. El archivo en formato de texto es de fácil importación a una hoja de cálculo, los registros están separador por el símbolo pipe (|)



## Current practice: **Gestion de requisitos**

Here you can create new practice criteria, new validation request and download a simple report with finished validation requests

Opciones para configurar criterios y ver las solicitudes de validaciones creadas.

Name	Objective	Tags	Create Validation Request / Visualize / Download
Extreme Programing	Extreme Programing sss	[ "dede" ]	  
Analisis de geousuarios basado en competencias	Analisis de geousuarios basado en competencias	[]	  
Gestion de requisitos	Gestion de requisitos	[]	  
Practica nueva con JR	Practica nueva con JR	[ "test", "agile", "scrum" ]	  
Test Manifest	Test Manifest	[]	  

Listado de practicas disponibles para configurarles lo criterios de validacion

Opciones para crear, vizualizar y descargar informe de validacion

Figura 29. Administración de criterios de validación para las prácticas modeladas.

### **Representación de la práctica**

La representación textual de la práctica previamente modelada se puede notar en la figura 30, allí se encontrará los elementos agrupados acorde a la especificación del estándar ESSENCE, se encontrarán todos los elementos agregados y configurados en la práctica de desarrollo de software modelada.

Practice Representation    Practice Validation

Get information ↓

Representacion textual de la practica con los elementos definidos en ella.

Practice: Gestion de requisitos

Alphas		Activity Space / Activities / Competencies			Patterns	
Alphas	Work Product	Activity Spaces	Activity	Required Competencies	Pattern	Association
Requirements	Contexto del proyecto SIG	Understand the requirements	Describir el proyecto SIG	Management	Analisis <FASE>	Contiene
				Analysis		
Requirements	Diagramas UML	Understand the requirements	Realizar diagramas de modelado de requisitos	Management	Analista sistemas <ROL>	Trabaja en
				Analysis		
				Management		
		Understand the requirements	Validar la especificación de requisitos	Management		
				Management		
		Understand the requirements	Determinar los recursos del proyecto	Management		
Analysis						
Leadership						
Shape the system	Describir la arquitectura del sistema	Analysis				
Use the System	Actividad de prueba		Stakeholder representation			
			Management			

Figura 30. Representación de la práctica modelada.

### Validación de la práctica

Finalmente, el prototipo permite realizar la validación de la práctica representada bajo criterios previamente definidos, donde se ingresan los valores para cada variable definida en cada criterio, el usuario con acceso a la URL de Internet ingresa los valores y envía los datos que serán almacenados para su futuro análisis, la figura 31 muestra brevemente esta opción importante, una vez enviado los datos la solicitud de validación quedará cerrada y no podrá editarse.

Public Validation: 60fe00e611af4d7f3cd412f5

Practice Representation Practice Validation

Identificador unico de la solicitud de validacion de la practica

Evaluator Name

Email

Additional Comments

Metadata para recolectar la informacion del evaluador de la practica

Status: **Open**

Listado de criterios disponibles para validar la practica

**Criteria List**

**Clarity**

**Objective:** This practice represents good elements according to the reality

**C**

Practice Clarity value between 1-5

**Result**

Formula expression: 1/5

Envia y cierra la validacion de la practica

Figura 31. Validación de la práctica bajo los criterios definidos.

### Pruebas reales del prototipo con representaciones graficas de público objetivo

Para las pruebas se tomaron representaciones de prácticas de desarrollo de fuentes confiables [24][25], luego de analizar su representación gráfica se realizó la representación textual usando el prototipo desarrollado, permitiendo además el ingreso de criterios de validación para dichas prácticas. La definición de los criterios de validación no eran parte del alcance de este trabajo pero se usaron datos de prueba que permitieran probar la funcionalidad de ingreso de datos de la práctica como la configuración de los criterios ejemplificados y la validación de la expresión de matemática definida.

### Prácticas modeladas

A continuación, se muestra una práctica representada usando los elementos gráficos y luego la misma práctica en SEMOVA.

- Práctica gestión de requisitos de un sistema de información geográfica, con sus fases de análisis y diseño [24], figuras (32,33, 34).

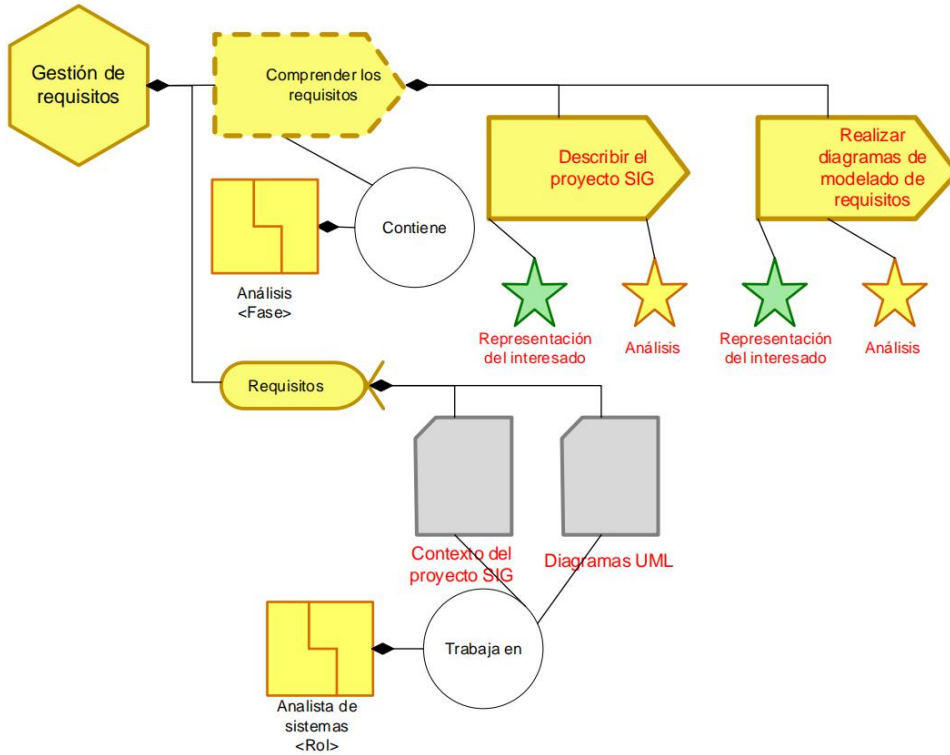


Figura 32. Fase de análisis de la práctica gestión de requisitos.

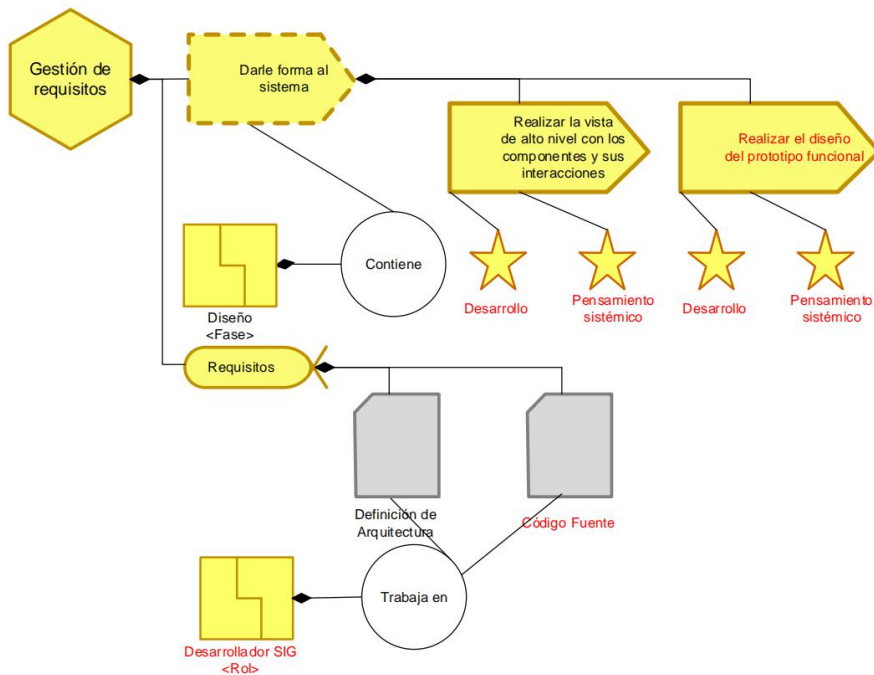


Figura 33. Fase de diseño de la práctica gestión de requisitos.

Alphas		Activity Space / Actividades / Competencies			Patterns				
Alphas	Work Product	Activity Spaces	Activity	Required Competencies	Pattern	Association	Element type	Element	
Requirements	Contexto del proyecto SIG	Understand Requirements	Describir el proyecto SIG	Stakeholder Representation	Diseño <FASE>	Contiene	activitySpace	Shape the System	
				Analysis	Analysis <FASE>	Contiene	activitySpace	Understand Requirements	
Requirements	Diagramas UML	Understand Requirements	Realizar diagramas de modelado de requisitos	Stakeholder Representation	Analista de Sistemas <ROL>	Trabaja en	workProduct	Contexto del proyecto SIG	
Requirements	Definición de arquitectura			Analysis					
Requirements	Código fuente			Development					
		Understand Requirements	Realizar la vista de alto nivel con los componentes y sus interacciones	Systemic Thinking	Desarrollador SIG <ROL>	Trabaja en	workProduct	Código fuente	
				Development					Definición de arquitectura
		Understand Requirements	Realizar el diseño del prototipo funcional	Systemic Thinking					

Figura 34. Representación textual en SEMOVA.

- Dos prácticas de un área de proceso de CMMI [26], figuras (35, 36, 37, 38)

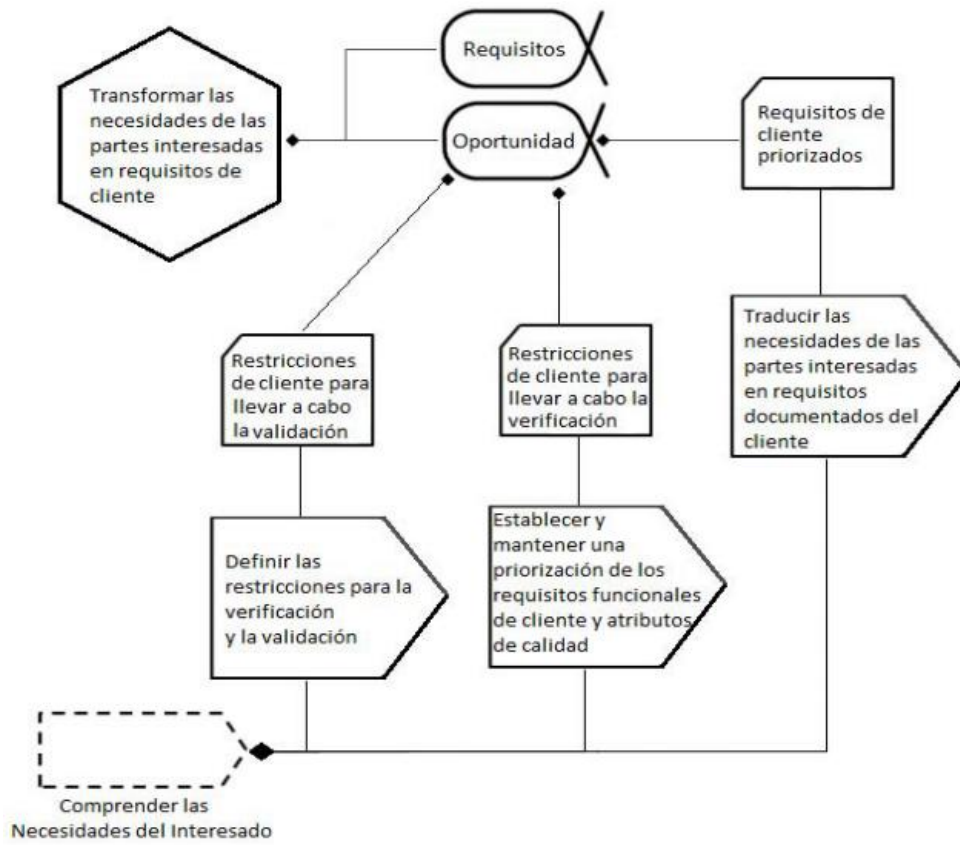


Figura 35. Representación área de proceso de CMMI.

Practice: Transformar las necesidades de las partes interesadas en requisitos de cliente

Alphas		Activity Space / Activities / Competencies			Patterns	
Alphas	Work Product	Activity Spaces	Activity	Required Competencies	Pattern Association type	Element Element
Oportunity	Restricciones de clientes para llevar a cabo la validación	Understand Stakeholder Needs	Definir las restricciones para la verificación y la validación			
Oportunity	Restricciones de clientes para llevar a cabo la verificación	Understand Stakeholder Needs	Establecer y mantener una priorización de los requisitos funcionales de cliente y atributos de calidad			
Oportunity	Requisitos de clientes priorizados	Understand Stakeholder Needs	Traducir las necesidades de las partes interesadas en requisitos documentados del cliente			

Figura 36. Representación textual en SEMOVA.

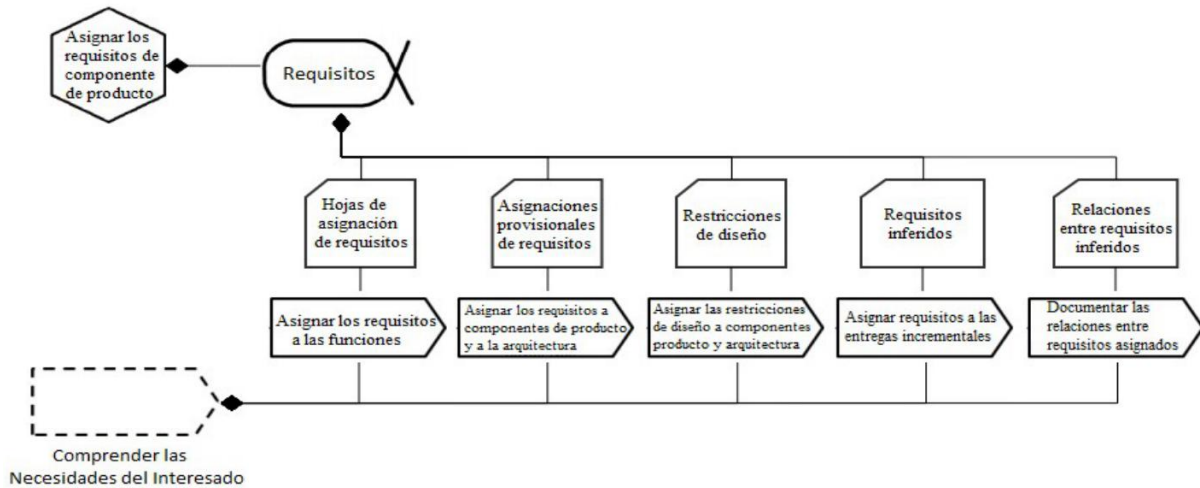


Figura 37. Representación de un área de proceso de CMMI.

Alphas		Activity Space / Activities / Competencies			Patterns			
Alphas	Work Product	Activity Spaces	Activity	Required Competencies	Pattern	Association	Element type	Element
Requirements	Hojas de asignación de requisitos	Understand Stakeholder Needs	Asignar los requisitos a las funciones					
Requirements	Asignaciones provisionales de requisitos	Understand Stakeholder Needs	Asignar los requisitos a componentes de producto y a la arquitectura					
Requirements	Restricciones de diseño	Understand Stakeholder Needs	Asignar las restricciones de diseño a componentes producto y arquitectura					
Requirements	Requisitos inferidos	Understand Stakeholder Needs	Asignar requisitos a las entregas incrementales					
Requirements	Relaciones entre requisitos inferidos	Understand Stakeholder Needs	Documentar las relaciones entre requisitos asignados					

Figura 38. Representación textual en SEMOVA.

- Práctica pair programming adaptada de la herramienta essencery [5]. Figuras (39, 40)

### Pair Programming

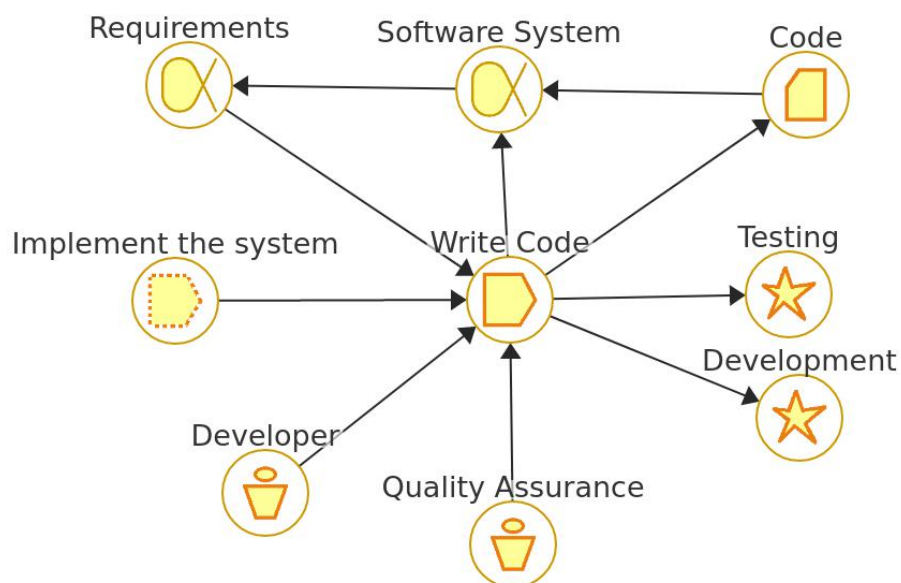


Figura 39. Representación pair programming en ESSENCERY.



Activity Space / Activities / Competencies								
Alphas		Activity Space / Activities / Competencies			Patterns			
Alphas	Work Product	Activity Spaces	Activity	Required Competencies	Pattern	Association	Element type	Element
Requirements	Code	Implement the System	Write Code	Development	Developer <ROL>	Write	workProduct	Code
				Testing	Quality Assurance <ROL>	Test	workProduct	Code

Figura 40. Representación textual en SEMOVA.

Ejemplo de ingreso de valores de criterios de una solicitud de validación y como se ven los resultados en el prototipo. Figuras (41, 42)

Practice Representation Practice Validation

---

Orvie Salgado Espitia

orvie.salgado@udea.edu.co

The practice represents the reality when implemented on teams that add new member to the team.

Status: **Closed**

**Criteria List**

**Representation against reality**

**Objective:** Does this representation reflect the real work activity and results when implemented?

**A** 4  
Representation 1-5

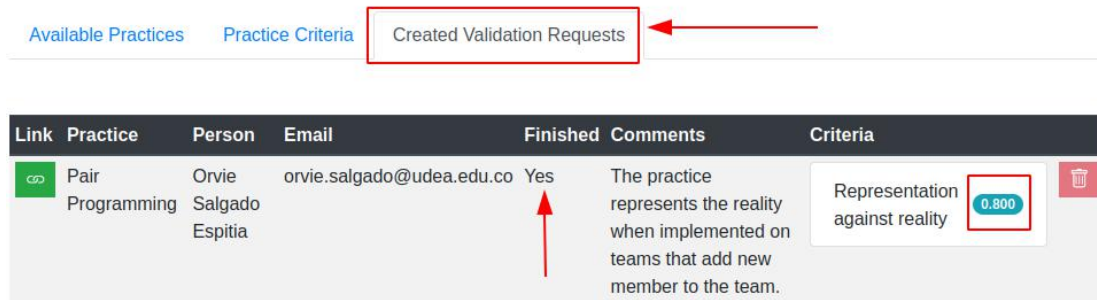
**Result** 0.8  
Formula expression: 4/5

Save Clear

Figura 41. Ingreso de valores de criterios en SEMOVA.

## Current practice: **Pair Programming**

Here you can create new practice criteria, new validation request and download a simple report with finished validation requests




Link	Practice	Person	Email	Finished	Comments	Criteria
	Pair Programming	Orvie Salgado Espitia	orvie.salgado@udea.edu.co	Yes	The practice represents the reality when implemented on teams that add new member to the team.	Representation against reality <span>0.800</span>

Figura 42. Visualización de resultados de evaluación del criterio en SEMOVA.

## CONCLUSIONES

Fue posible revisar la especificación formal ESSENCE versión 1.2 de la OMG (Object Management Group) el cual define y estandariza los fundamentos del núcleo y el lenguaje para los métodos de la ingeniería de software, entendiendo los aspectos técnicos del lenguaje y su utilización en el campo de la ingeniería de software. Por otra parte encontramos diferentes herramientas que buscan representar de manera gráfica los elementos del lenguaje, y la más relevante que es patrocinada directamente por Ivar Jacobson Internacional SA, que permite documentar y tener acceso a prácticas ya creadas a manera de biblioteca de prácticas ofreciendo un producto bajo licencia de pago llamado ESSENCE 365 Enterprise, y otras herramientas no tan robustas que permiten de alguna manera representar gráficamente prácticas usando los elementos del lenguaje pero sin la opción de tener una validación de dicha práctica usando unos criterios y que se puedan publicar para recibir retroalimentación de la práctica documentada y representada.

En consecuencia se logra definir e implementar un prototipo de software funcional, con acceso web, que permite suplir la necesidad a baja escala de representar textualmente una práctica de desarrollo de software usando los elementos básicos del estándar ESSENCE de OMG, además que permite definir un conjunto de criterios de validación, para luego ser compartidos o publicados bajo el esquema de solicitud de validación, finalmente permitiendo a un público objetivo externo revisar la representación textual de la práctica de desarrollo de software resumida, a su vez que permite realizar la validación y retornar el resultado de las validaciones a los autores de la representación y modelado de la práctica.

Éste trabajo constituyó una base para la continuación de trabajos futuros relacionados con el entendimiento y también se demostró la apropiación del estándar Essence en la comunidad académica e industria. El Código quedará abierto en forma pública.

Se demostró también la capacidad de enfrentarse al desarrollo de un producto de software a partir de la definición de un problema, donde se destacó el uso de herramientas propias de la ingeniería, el área de desempeño y vigentes en la industria, tales como: MongoDB, Vue.js, HTML, Node.js, control de versiones mediante github, etc.

## **TRABAJO FUTURO**

Como resultado del presente trabajo que constituye una base fundamental en la representación de practicas de desarrollo de software para su validación, el trabajo a realizar es evolucionar el prototipo para que en futuras versiones sea más robusto en el modelado, utilizando elementos más complejos del estándar ESSENCE, adición de los estados de los ALPHAS para tener trazabilidad de la implementación de la práctica de desarrollo de software en un contexto académico o empresarial, generación de una representación gráfica de la práctica modelada, definición de criterios más complejos con expresiones matemáticas más avanzadas, gráficos analíticos de los resultados de las validaciones, implementar perfiles de usuarios con roles específicos como: administrador, modelador y validador. Finalmente se espera que la iniciativa sea un referente a nivel académico y cuando este lo suficientemente madura una herramienta pionera de libre uso en el ámbito empresarial.

## **REFERENCIAS BIBLIOGRAFICAS**

- [1] Essence - Kernel and Language for Software Engineering Methods, OMG Std., Rev. 1.2, Oct 2018. [Online]. Available: <http://www.omg.org/spec/Essence/1.2>. [Accedido en: jun. 11, 2021]
- [2] P. Johnson, M. Ekstedt, y I. Jacobson, "Where's the Theory for Software Engineering?", IEEE Softw., vol. 29, n.o 5, pp. 96-96, sep. 2012, doi: 10.1109/MS.2012.127.
- [3] T. Päivärinta y K. Smolander, "Theorizing about software development practices", Sci. Comput. Program., vol. 101, pp. 124-135, abr. 2015, doi: 10.1016/j.scico.2014.11.012.
- [4] D. Graziotin y P. Abrahamsson, «A Web-based modeling tool for the SEMAT Essence theory of software engineering», J. Open Res. Softw., vol. 1, p. e4, sep. 2013, doi: 10.5334/jors.ad.
- [5] "Essencery tool", Official application site. [en línea]. Disponible en: <http://www.essencery.com>. [Accedido en: jun. 11, 2021]
- [6] "About the Essence Specification Version 1.2". [En línea]. Disponible en: <https://www.omg.org/spec/Essence/1.2>. [Accedido en: jun. 04, 2021]
- [7] M. Kajko-Mattsson et al., "Refounding software engineering: The Semat initiative (Invited presentation)", en 2012 34th International Conference on Software Engineering (ICSE), jun. 2012, pp. 1649-1650. doi: 10.1109/ICSE.2012.6227214.

- [8] I. Jacobson, H. “Bud” Lawson, P.-W. Ng, P. E. McMahon, y M. Goedicke, “The Essentials of Modern Software Engineering: Free the Practices from the Method Prisons!” Association for Computing Machinery and Morgan & Claypool, 2019.
- [9] RedAgile, “The Scrum Guide”, RedAgile, ago. 19, 2020. <https://www.redagile.com/post/the-scrum-guide> [Accedido en: jun. 04, 2021]
- [10] “What is OMG Essence | IGI Global”. <https://www.igi-global.com/dictionary/connecting-the-dots-between-human-factors-and-software-engineering/95427>. [Accedido en: jun. 11, 2021]
- [11] Essence Language Key, Ivar Jacobson International SA.[En línea]. Disponible en: <https://practicelibrary.ivarjacobson.com/content/essence-language-key>. [Accedido: jun. 11, 2021]
- [12] “¿Qué es MongoDB?”, MongoDB. [En línea]. Disponible en: <https://www.mongodb.com/es/what-is-mongodb>. [Accedido en: jun. 17, 2021]
- [13] “La base de datos líder del mercado para aplicaciones modernas”, MongoDB. [En línea]. Disponible en: <https://www.mongodb.com/es>. [Accedido: jun. 17, 2021]
- [14] “Introducing JSON”, ECMA-404 The JSON Data Interchange Standard. [En línea]. Disponible en: <https://www.json.org/json-en.html>. [Accedido en: jun. 17, 2021]
- [15] “Introducción — Vue.js”. [En línea]. Disponible en: <https://es.vuejs.org/v2/guide>. [Accedido jun. 17, 2021]
- [16] “Reactividad en profundidad — Vue.js”. [En línea]. Disponible en: <https://es.vuejs.org/v2/guide/reactivity.html>. [Accedido en: jun. 17, 2021]
- [17] M. Shema, “Chapter 2 - HTML Injection & Cross-Site Scripting (XSS)”, en Hacking Web Apps, M. Shema, Ed. Boston: Syngress, 2012, pp. 23-78. doi: 10.1016/B978-1-59-749951-4.00002-3.
- [18] “¿Qué es JavaScript? - Aprende sobre desarrollo web | MDN”. [En línea]: Disponible en: [https://developer.mozilla.org/es/docs/Learn/JavaScript/First\\_steps/What\\_is\\_JavaScript](https://developer.mozilla.org/es/docs/Learn/JavaScript/First_steps/What_is_JavaScript). [Accedido en: jun. 17, 2021]
- [19] “Fundamentos de JavaScript - Aprende sobre desarrollo web | MDN”. [En línea]. Disponible en: [https://developer.mozilla.org/es/docs/Learn/Getting\\_started\\_with\\_the\\_web/JavaScript\\_basics](https://developer.mozilla.org/es/docs/Learn/Getting_started_with_the_web/JavaScript_basics). [Accedido en: jun. 17, 2021]
- [20] Node.js, “About”, Node.js. [En línea]. Disponible en: <https://nodejs.org/en/about>. [Accedido en: jul. 23, 2021]
- [21] “Express - Infraestructura de aplicaciones web Node.js”. [En línea]. Disponible en: <https://expressjs.com/es>. [Accedido en: jul. 23, 2021]

- [22] EALDE, “Que es el Product Backlog y el Sprint Backlog en Scrum”, EALDE Business School, ago. 27, 2019. [En línea]. Disponible en: <https://www.ealde.es/product-backlog-sprint-backlog/> [Accedido en: jul. 05, 2021]
- [23] X. Albaladejo, “Ejemplo de uso del tablero o pizarra de tareas (Scrum Taskboard)”, Proyectos Ágiles, sep. 26, 2010. [En línea]. Disponible en: <https://proyectosagiles.org/2010/09/26/ejemplo-tablero-pizarra-tareas-scrum-taskboard>. [Accedido en: jul. 05, 2021]
- [24] D. Claudia, “Definición de buenas prácticas de desarrollo de sistemas de información geográfica utilizando el núcleo de Semat” Dep. Ciencias de la computación y decisión, Univ. Nacional, Medellín, 2019
- [25] P. Carlos, “Representación de las mejores prácticas de verificación de sistemas de software en el núcleo de la esencia de Semat” Dep. Ciencias de la computación y decisión, Univ. Nacional, Medellín, 2019
- [26] C. M. Zapata, J. Valderrama and L. D. Jiménez, “Representation Of CMMI-DEV Practices In The Semat Kernel”, Dep. Ciencias de la computación y decisión, Univ. Nacional, Medellín, 2019

## ANEXOS

Como complemento del resultado obtenidos con el desarrollo de este trabajo, se adjuntan manual técnico de la solución y código fuente de los proyectos BACKEND que contiene la API REST en Node.js y FRONTEND que contiene la aplicación cliente en Vue.js. Los documentos anexados son:

- manual-técnico-semova-v1.docx
- semat-modeler-backend-api-main.zip
- semat-modeler-front-main.zip