



Universidad Nacional Mayor de San Marcos

Universidad del Perú. Decana de América

Facultad de Ingeniería de Sistemas e Informática

Escuela Profesional de Ingeniería de Software

Essboard: una herramienta colaborativa para el uso de esencia en el monitoreo de desarrollo de software

TESIS

Para optar el Título Profesional de Ingeniero de Software

AUTORES

Diana Mary QUINTANILLA PEREZ

Arthur MAURICIO DELGADILLO

ASESOR

Dr. David Santos MAURICIO SANCHEZ

Lima, Perú

2019



Reconocimiento - No Comercial - Compartir Igual - Sin restricciones adicionales

<https://creativecommons.org/licenses/by-nc-sa/4.0/>

Usted puede distribuir, remezclar, retocar, y crear a partir del documento original de modo no comercial, siempre y cuando se dé crédito al autor del documento y se licencien las nuevas creaciones bajo las mismas condiciones. No se permite aplicar términos legales o medidas tecnológicas que restrinjan legalmente a otros a hacer cualquier cosa que permita esta licencia.

Referencia bibliográfica

Mauricio, A. & Quintanilla, D. (2019). *Essboard: una herramienta colaborativa para el uso de esencia en el monitoreo de desarrollo de software*. Tesis para optar el Título Profesional de Ingeniero de Software. Escuela Profesional de Ingeniería de Software, Facultad de Ingeniería de Sistemas e Informática, Universidad Nacional Mayor de San Marcos, Lima, Perú.

Hoja de metadatos complementarios

- **Código ORCID del autor:**
- **Código ORCID del asesor:** 0000-0001-9262-626X
- **DNI o pasaporte del autor:**
73646447
48448479
- **Grupo de investigación:**
- **Institución que financia la investigación:**
- **Ubicación geográfica donde se desarrolló la investigación:**
Calle Ramón Cerdeira N° 175, dpto. 301, Urb. Las Begonias, San Borja.
(12.105, -76.994)
- **Año o rango de años que la investigación abarcó:**
2018 - 2019



UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS
Universidad del Perú, DECANA DE AMÉRICA
FACULTAD DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
ESCUELA PROFESIONAL DE INGENIERÍA DE SOFTWARE

Acta de Sustentación de Tesis

Siendo las ^{13^h} del día ²³ de abril del año 2019, se reunieron los docentes designados como miembros de Jurado de la Tesis, presidido por el Dr. Javier Arturo Gamboa Cruzado, Ing. Fany Yexenia Sobero Rodríguez (Miembro), y el Dr. David Santos Mauricio Sánchez (Miembro Asesor) para la sustentación de la Tesis intitulada: **"ESSBOARD: UNA HERRAMIENTA COLABORATIVA PARA EL USO DE ESENCIA EN EL MONITOREO DE DESARROLLO DE SOFTWARE"**; por la Bach. Diana Mary Quintanilla Perez, para optar el Título Profesional de Ingeniero de Software.

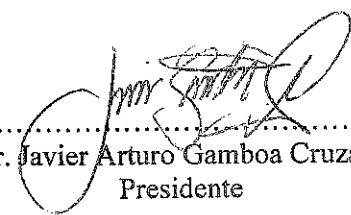
Acto seguido de la exposición de la Tesis, el Presidente invitó a la Bachiller a dar respuesta a las preguntas establecidas por los Miembros de Jurado.

La Bachiller en el curso de sus intervenciones demostró pleno dominio del tema, al responder con acierto y fluidez a las observaciones y preguntas formuladas por los señores miembros del Jurado.

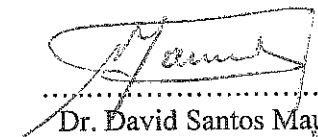
Finalmente habiéndose efectuado la calificación correspondiente por los miembros de Jurado, la bachiller obtuvo la nota de ¹³.... (En letras)..... ^{Diecitrés}.....

A continuación el Presidente del Jurado, Dr. Javier Arturo Gamboa Cruzado declara a la Bachiller Ingeniero de Software.

Siendo las ^{14²⁵} horas, se levantó la sesión.


.....
Dr. Javier Arturo Gamboa Cruzado
Presidente


.....
Ing. Fany Yexenia Sobero Rodríguez
Miembro


.....
Dr. David Santos Mauricio Sánchez
Miembro Asesor



UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS
Universidad del Perú, DECANA DE AMÉRICA
FACULTAD DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
ESCUELA PROFESIONAL DE INGENIERÍA DE SOFTWARE

Acta de Sustentación de Tesis

Siendo las ¹³ del día ²³ de abril del año 2019, se reunieron los docentes designados como miembros de Jurado de la Tesis, presidido por el Dr. Javier Arturo Gamboa Cruzado, Ing. Fany Yexenia Sobero Rodríguez (Miembro), y el Dr. David Santos Mauricio Sánchez (Miembro Asesor) para la sustentación de la Tesis intitulada: **"ESSBOARD: UNA HERRAMIENTA COLABORATIVA PARA EL USO DE ESENCIA EN EL MONITOREO DE DESARROLLO DE SOFTWARE"**; por el Bach. Arthur Mauricio Delgadillo, para optar el Título Profesional de Ingeniero de Software.

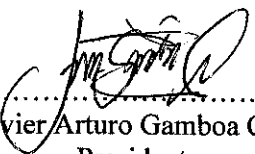
Acto seguido de la exposición de la Tesis, el Presidente invitó al Bachiller a dar respuesta a las preguntas establecidas por los Miembros de Jurado.

El Bachiller en el curso de sus intervenciones demostró pleno dominio del tema, al responder con acierto y fluidez a las observaciones y preguntas formuladas por los señores miembros del Jurado.

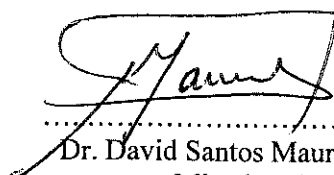
Finalmente habiéndose efectuado la calificación correspondiente por los miembros de Jurado, el bachiller obtuvo la nota de ¹⁸.... (En letras)... **DIECDOCHO**.....

A continuación el Presidente del Jurado, Dr. Javier Arturo Gamboa Cruzado declara al Bachiller **Ingeniero de Software**.

Siendo las ¹⁴ horas, se levantó la sesión.


.....
Dr. Javier Arturo Gamboa Cruzado
Presidente


.....
Ing. Fany Yexenia Sobero Rodríguez
Miembro


.....
Dr. David Santos Mauricio Sánchez
Miembro Asesor

A nuestros padres y hermanos.

A toda nuestra familia.

A nuestros amigos.

Por su apoyo, comprensión, consejos y compañía.

AGRADECIMIENTOS

A nuestro asesor, Dr. David Mauricio Sánchez, por el apoyo y el tiempo brindado para el desarrollo de esta tesis.

Al jurado, por prestar su evaluación crítica y desinteresada en el proyecto realizado.

A la Universidad Nacional Mayor de San Marcos y la Facultad de Ingeniería de Sistemas e Informática, por haber invertido en nuestra educación.

A nuestros padres, por su apoyo incondicional y por el empuje que admiramos en ellos.

A nuestra familia, por darnos su tiempo, cariño y consejos.

A todas aquellas personas que indirectamente nos ayudaron para culminar este trabajo y que muchas veces constituyen un invalorable apoyo.

Y, por encima de todo, damos gracias a Dios por ser muy, muy bueno con nosotros.

ÍNDICE

Lista de tablas	ix
Lista de figuras	x
Resumen	xi
CAPÍTULO 1: INTRODUCCIÓN	1
1.1 Antecedentes del Problema	1
1.2 Planteamiento del Problema	2
1.3 Importancia del Problema	3
1.4 Motivación	4
1.5 Objetivo	4
1.6 Objetivos Específicos	5
1.7 Propuesta	5
1.8 Estructura de la tesis	6
CAPÍTULO 2: UNA VISTA GENERAL A ESSENCE	8
2.1 Concepto	8
2.2 Las cosas con las que siempre trabajamos: Alphas, Estados y Checklists	9
2.3 Essence en la Dirección y Monitoreo de Desarrollos de Software	11
CAPÍTULO 3: ESTADO DEL ARTE	14
3.1 Criterios de búsqueda	14
3.2 Herramientas que soportan Essence	14
3.2.1 Semat Accelerator	16
3.2.2 SEMAT Kernel Tool	17
3.2.3 Alpha State Explorer App	19
3.2.4 OCMS Tool	20

3.3 Colaboración en las herramientas para dirección y monitoreo	21
3.3.1 Método de evaluación	22
3.3.2 Evaluación	23
CAPÍTULO 4: ESSBOARD	30
4.1 Una vista general	30
4.2 Interactuando con la herramienta	31
4.3 Essboard y su apoyo a la dirección y monitoreo	35
4.4 Proceso de desarrollo	36
4.5 Diseño e implementación	38
4.5.1 Template	39
4.5.2 Project	42
4.5.3 Cliente	43
4.5.4 Tecnologías	45
4.6 Limitaciones técnicas	46
CAPÍTULO 5: VALIDACIÓN	47
5.1 Caso de estudio	47
5.1.2 Dirección y monitoreo usando Essboard	49
5.2 Colaboración en Essboard	53
5.3 Discusión	57
5.3.1 Soporte a la dirección y monitoreo	57
5.3.2 Comparando Essboard	58
5.3.3 Limitaciones de la validación	59
CAPÍTULO 6: CONCLUSIONES	60
6.1 Conclusiones	60
6.2 Trabajos futuros	61

REFERENCIAS	62
ANEXOS	66
ANEXO A: Elementos de diseño del método de calidad de conciencia	66
ANEXO B: Matriz o tabla de correlación entre elementos de diseño y tipos de conciencia	69

Lista de tablas

Tabla 1. Herramientas que soportan Essence	15
Tabla 2. Elementos de diseño relevantes para la evaluación en el contexto de dirección y monitoreo	25
Tabla 3. Puntuaciones sobre los elementos de diseño	26
Tabla 4. Puntajes para las herramientas que soportan Essence	27
Tabla 5. Puntajes positivos y negativos de las herramientas por categorías de conciencia	28
Tabla 6. Información general del proyecto RIS	48
Tabla 7. Puntajes para Essboard en los 3 escenarios de evaluación	54
Tabla 8. Puntajes positivos y negativos de las herramientas por categorías de conciencia	55

Lista de figuras

<i>Figura 1.</i> Essboard como un medio para el monitoreo y dirección basados en los estados de Essence	6
<i>Figura 2.</i> Los alphas (demarcados por símbolos alfa) y las relaciones entre ellas	9
<i>Figura 3.</i> Estados del alpha Sistema de Software con sus respectivos checklist	10
Figura 4 Ventana principal de Semat Accelerator	16
Figura 5. Ventana principal de Semat Kernel Tool	18
<i>Figura 6</i> Ventana principal de Alpha State Explorer	19
Figura 7. Dashboard de OCMS y progreso de los alphas	21
Figura 8. Puntaje obtenido por (a) Semat Kernel Tool, (b) OCMS	29
Figura 9. Essboard: una visión general	31
Figura 10. Página de inicio de Essboard	32
Figura 11. Vista de un proyecto en Essboard	33
Figura 12. Espacio de trabajo de una sesión en Essboard	34
Figura 13. Arquitectura de Essboard	38
Figura 14. Componente Template	39
Figura 15. Essence como un grafo	41
Figura 16. Vista de los eventos de un proyecto a través del tiempo	42
Figura 17. Estructura del módulo cliente	44
Figura 18. Reuniones usando tarjetas de Essence	49
Figura 19. El proceso del meeting de Essence en Essboard en el alpha Oportunidad	51
Figura 20. Tonalidades de colores en una tarjeta de estado en Essboard	52
Figura 21. Puntaje obtenido por los escenarios,	56
Figura 22. Evaluación de Semat Kernel Tool (a) y Essboard en el escenario pesimista (b)	59

Essboard: Una herramienta colaborativa para el uso de Essence en el monitoreo y dirección del desarrollo de software

Resumen

Essence, un estándar de Object Management Group, define un conjunto de elementos que se presentan en todo desarrollo de software. Una de sus aplicaciones se orienta en apoyar a equipos a establecer el estado actual de su proyecto (monitoreo) y definir dónde dirigirse (dirección). Actualmente se han desarrollado pocas herramientas que contribuyan a esta aplicación, más aún ellas dejan de lado la colaboración que promueve la comunicación e interacción del equipo en el desarrollo de software y que en sí es inherente a ella. Por lo que en este trabajo desarrollamos Essboard, una herramienta que toma un enfoque colaborativo en la dirección y monitoreo de proyectos de software, al brindar un entorno de trabajo a equipos de desarrollo que promueve la colaboración en la definición del estado y las metas de un proyecto, como también promueve una visión común del estado y el progreso del desarrollo a través de radares de información. Mostramos, a través de un caso práctico, que Essboard puede soportar reuniones comunes de dirección y monitoreo de proyectos; y evidenciamos mediante un test de generación de conciencia que Essboard promueve mejor el trabajo colaborativo al obtener resultados superiores en relación a las herramientas para Essence actuales.

Palabras clave: SEMAT Essence, dirección, monitoreo, proyectos, herramienta, software colaborativo

Essboard: A collaborative tool for Essence support in the monitoring and steering of software development

Abstract

Essence, an Object Management Group standard, defines a set of elements that are part of any software development. One of its applications is focused at supporting development teams, in establishing the current status of their project (monitoring) and defining where to go (steering). Currently, few tools have been developed for this purpose, moreover, they leave aside the collaboration, aspect inherent of software development that promotes communication and team interaction. Therefore, in this work, we present Essboard, a tool that takes a collaborative approach in the steering and monitoring of software projects by providing a team workspace that promotes the collaboration in establishing the current status and goals of a project, as well as promoting a shared vision of the state and the progress of development through information radars. We show in a case study, that Essboard can support common meetings of steering and monitoring; and we demonstrate through a test of awareness that Essboard has better support for collaborative work than current Essence tools.

Keywords: SEMAT Essence, steering, monitoring, software project, tool, collaborative software

CAPÍTULO 1: INTRODUCCIÓN

1.1 Antecedentes del Problema

En el aspecto personal, mantener un balance en cada aspecto esencial de la vida (trabajo, familia, amigos, salud, educación) es una meta que muchos buscan; de igual manera esto sucede en el desarrollo de software (Booch, 2018). Sin embargo, muchos proyectos de desarrollo de software se enfocan en un solo aspecto del desarrollo dejando de lado otros (p. ej., la participación de los stakeholders, la búsqueda de una forma organizada de trabajar o la validación de las necesidades del cliente), causando así el fracaso de muchos de los proyectos de desarrollo (Shane *et al.*, 2015).

SEMAT (Jacobson, *et al.*, 2009), una iniciativa para redefinir la ingeniería de software, plasmó estos aspectos esenciales que se presentan en todo desarrollo de software en el estándar Essence (OMG, 2015), con ello hace explícito qué elementos debe considerar un equipo en el desarrollo de un proyecto, sirviendo, así como un recordatorio para no olvidar los aspectos básicos y trabajar continuamente para el progreso de estos. Por ello, Essence juega un rol importante en la dirección y monitoreo (Péraire & Sedano, 2014).

Por otro lado, la comunidad de Essence busca promover su uso y aceptación tanto a nivel académico como práctico. Para lograr esta meta, a nivel académico se han realizado diversos talleres y conferencias (Jacobson, *et al.*, 2013), cambios de currícula en planes académicos de ingeniería de software basados en Essence (Zapata & Jacobson, 2014) y casos de estudio sobre reuniones de reflexión promovidas por Essence (Péraire & Sedano, 2014). Mientras que a nivel práctico (es decir, orientado al desarrollo de software) se han desarrollado guías de usuario, talleres y charlas para empresas

importantes como Google¹ y Red Hat². Además, SEMAT, consciente de que la mayoría de los métodos y prácticas de desarrollo de software necesitan de herramientas que faciliten su uso, realizó un llamado al desarrollo de estas:

“Necesitamos iniciar un proyecto de código abierto donde se desarrolle al menos una herramienta que soporte Essence similar a las herramientas de código abierto de Agile / Scrum / Kanban disponibles hoy en día (por ejemplo, Trello, JIRA, SeeNowDo, etc.)”. (SEMAT, 2018)

A la fecha distintas herramientas se han desarrollado para diversos usos de Essence, entre ellas en el desarrollo, adaptación de métodos y prácticas (Evensen, *et al.*, 2018), en la educación (Pieper, 2015) y en modos de visualización (Brandt, *et al.*, 2017). Sin embargo, pocas herramientas se enfocan en el monitoreo y dirección para proyectos reales (Senzaki, *et al.*, 2017), más aún el enfoque colaborativo, inherente en todo desarrollo de software (Whitehead, 2007), no es puesto en primer plano (a diferencia de las herramientas para otros frameworks y métodos de trabajo).

1.2 Planteamiento del Problema

Las herramientas computarizadas existentes para Essence en el monitoreo y dirección del desarrollo de software actualmente no proporcionan un soporte adecuado para la colaboración en las sesiones de discusión de Essence, donde todo el equipo interactúa en la determinación del estado actual y los objetivos del desarrollo de software.

¹ The Essence of Software Engineering: the SEMAT Approach, <https://www.ivarjacobson.com/videos/semat-presentation-google>, último acceso el 19/12/18

² Essence in practice @RedHat, <https://www.ivarjacobson.com/publications/presentations/presentation-redhat-essence-practice>, último acceso el 19/12/18

1.3 Importancia del Problema

Como hemos descrito en la sección 1.1, Essence está siendo adoptado en la comunidad de ingeniería de software, tanto a nivel académico (Zapata & Jacobson, 2014), (Sedano, 2016), (Pieper, 2015), donde es usado en la mejora de planes curriculares (e.g. Carnegie Mellon) y en el desarrollo de juegos para su enseñanza; como a nivel práctico (Seymour, E. 2015) (Cunningham, D. 2013) (McDonough, 2013) donde es usado en la planeación de proyectos como en el desarrollo y estandarización de métodos de servicios en empresas como Red Hat, Fujitsu UK y Munich Re. Además, en el uso de Essence para el monitoreo y dirección de proyectos de software, diversos estudios se han realizado sobre su efectividad (Péaire & Sedano, 2014) (Morales-Trujillo, et al., 2016) (Pan, 2014), concluyendo que este es útil para este fin al abordar diversas dimensiones de un desarrollo de software, promover reuniones de reflexión comparables y complementarias a las retrospectivas ágiles, permitir a equipos explorar áreas de mejora en una manera autodirigida y probar cómo funcionan las prácticas que usan.

Sin embargo, Essence en este contexto, implica usar tarjetas físicas y checklist, lo cual dificulta el rastreo de información a lo largo del tiempo –un aspecto principal en el monitoreo (Beck, & Fowler 2001), por ello el uso de herramientas computarizadas juega un rol importante, de tal forma que apoyen en gestionar el inmenso volumen de información que se produce en un proyecto de software (Sommerville, 2011).

Por otro lado, es conocido que la colaboración es inherente en el desarrollo de software, incluso es puesta en primer plano en los bien conocidos métodos ágiles (Highsmith & Cockburn, 2001). Promoverla contribuye en facilitar la comunicación e interacción en este contexto. (Whitehead, 2007). En el monitoreo y dirección del

desarrollo de software, la colaboración juega un rol importante en la evaluación del estado y progreso de este, ya que una evaluación sin colaboración implica una evaluación desde sola una perspectiva, pudiendo ser sesgada a los aspectos que sola una persona cree importante, llegando muchas veces a no reflejar la situación real. Por ello, es importante que las herramientas tengan en consideración la colaboración, ya que un sistema colaborativo ayuda a los grupos a comunicarse, colaborar y coordinar sus actividades (Ellis, *et al.*, 1991), además, de generar conciencia en el grupo (Neale, *et al.*, 2004), esto es, lograr una visión compartida del estado y progreso del proyecto, la cual es la base para la comunicación eficaz e interacción del equipo durante la toma de decisiones sobre las acciones en un proyecto.

1.4 Motivación

Este trabajo ha sido motivado por dos razones: la razón principal, proveer una herramienta que se ajuste en mayor medida a las necesidades de un equipo en manejar la planificación y monitoreo de un proyecto usando Essence, y la segunda, el llamado de SEMAT a la creación de herramientas de código libre, con el fin de incentivar el uso de Essence en el desarrollo de proyectos de software.

1.5 Objetivo

Desarrollar una herramienta web que soporte el trabajo colaborativo para el uso de Essence en el monitoreo y dirección del desarrollo de software.

1.6 Objetivos Específicos

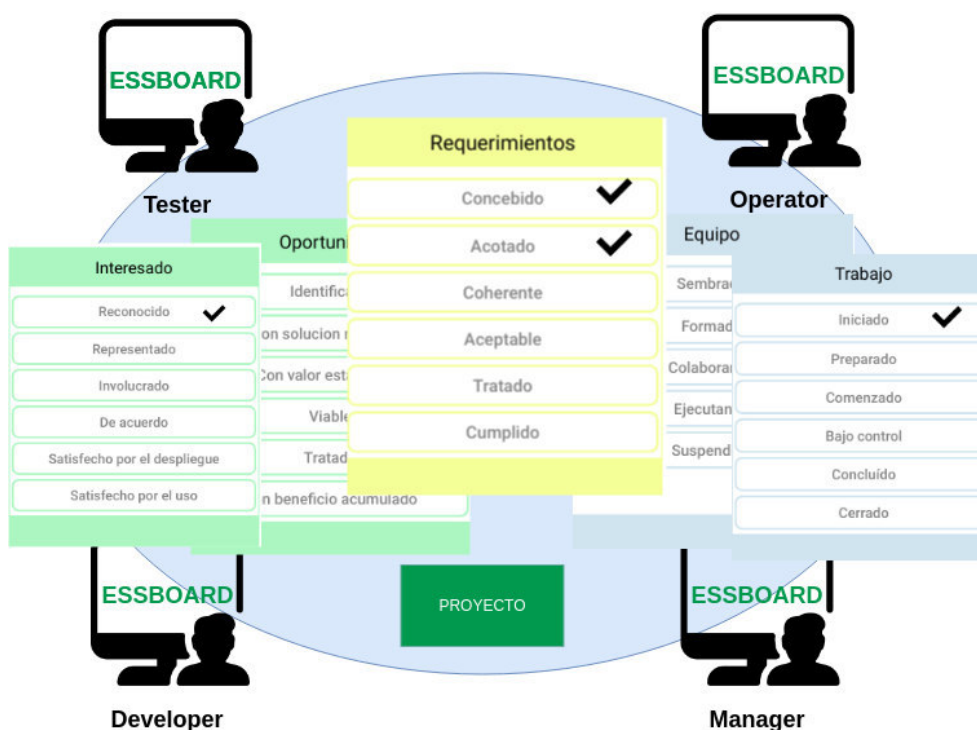
Los objetivos específicos de esta tesis son los siguientes:

- (a) Identificar las bondades y limitaciones de las herramientas existentes que soportan Essence, enfocándose en las características que apoyan la colaboración.
- (b) Diseñar e implementar una herramienta web colaborativa que apoye el monitoreo y dirección del desarrollo de software basado en Essence.
- (c) Validar que la herramienta posea un grado aceptable de colaboración.

1.7 Propuesta

Proponemos Essboard: una herramienta web para direccionar y monitorear proyectos de desarrollo de software colaborativamente basada en los estados de Essence. Essboard toma un enfoque colaborativo al poseer características que facilitan la colaboración (p. ej., comunicación por chat, conectividad de miembros) en los meetings de evaluación del estado y los objetivos de un proyecto, a través de un espacio de trabajo donde se discuten los aspectos que son importantes en el proyecto (ver figura 1).

Figura 1. Essboard como un medio para el monitoreo y dirección basados en los estados de Essence



Fuente. Elaboración propia

1.8 Estructura de la tesis

La presente tesis está estructurada en 5 capítulos que a continuación se describen brevemente.

En el capítulo 2, se describe el marco teórico, en el que se describe concisamente Essence, sus elementos principales y los principios que ayudan al monitoreo y dirección del desarrollo de software.

En el capítulo 3, revisamos el estado actual de colaboración en las herramientas que soportan Essence en el monitoreo y dirección del desarrollo de software; describimos las herramientas existentes que están apoyando el uso de Essence y analizamos sus grados de colaboración con base a uno de sus componentes principales, la generación de conciencia.

En el capítulo 4, se introduce una herramienta que denominamos Essboard, que permite a equipos monitorear y direccionar en forma colaborativa el desarrollo de su proyecto de software. Se describe la herramienta, su diseño e implementación y como este apoya a la dirección y monitoreo.

En el capítulo 5, se valida que la herramienta propuesta ayuda en la dirección y monitoreo de proyectos de software colaborativamente mediante la aplicación de Essboard en un caso que usaba tarjetas físicas de Essence (Kajko-Mattsson, M., 2014) y evaluamos el grado de colaboración de Essboard usando el método de (Antunes, *et al.*, 2014).

Finalmente, en el capítulo 6, se describe las conclusiones del presente trabajo, sus limitaciones y los trabajos futuros que se pueden desarrollar como consecuencia de la presente investigación.

CAPÍTULO 2: UNA VISTA GENERAL A ESSENCE

En este capítulo describimos Essence, enfocándonos en especial en cómo este da apoyo en la dirección y monitoreo. Para ello describimos brevemente Essence, sus elementos más importantes para la dirección y monitoreo de proyectos de software, finalmente, describimos cómo Essence soporta a este fin.

2.1 Concepto

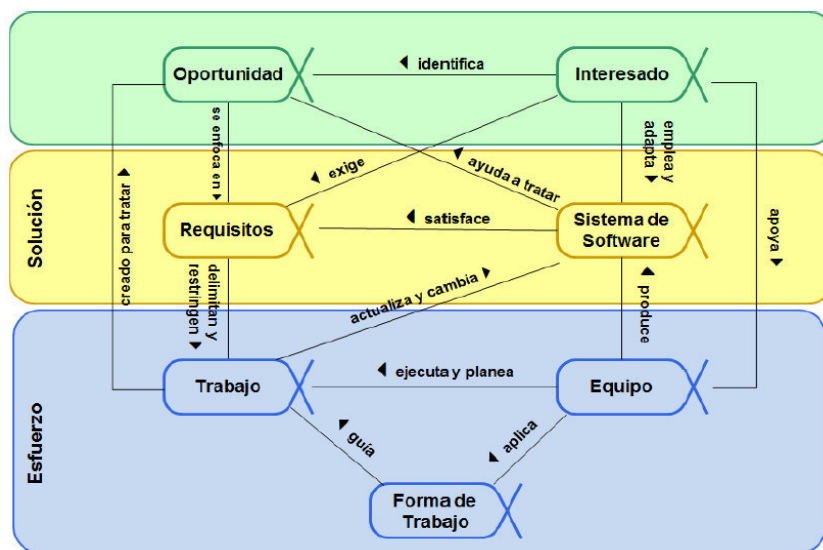
Ivar Jacobson, Bertrand Meyer y Richard Soley con el objetivo de "refundar la ingeniería de software como una disciplina rigurosa" iniciaron SEMAT - Software Engineering Methods And Theory (Jacobson, *et al.*, 2009). De esta iniciativa surgió el Essence, que en 2013 se convirtió en un estándar de Object Management Group para describir y desarrollar métodos de desarrollo de software basado en un núcleo y un lenguaje (OMG, 2015). El núcleo contiene un conjunto de elementos denominados “Cosas con la que siempre trabajamos”, “Cosas que siempre hacemos” y “Habilidades que siempre necesitamos tener” cuando desarrollamos sistemas de software (Jacobson, Spence, & Ng, 2013). El lenguaje permite especificar estos elementos y sus relaciones de forma visual e intuitiva, similar a como UML permite describir distintos aspectos de un sistema de software.

En la presente tesis nos orientaremos a las “Cosas con las que siempre trabajamos”, dado que estos son los elementos esenciales en la dirección y monitoreo del desarrollo de software.

2.2 Las cosas con las que siempre trabajamos: Alphas, Estados y Checklists

Essence brinda 7 “cosas con las que siempre trabajamos” o también denominados alphas (ver figura 2), que son en sí los aspectos que se deben tomar en cuenta en todo desarrollo de software, sea en un desarrollo de un aplicativo web o un sistema embebido. Un ejemplo de alpha es el alpha Interesado, el cual representa las personas, grupos u organizaciones que afectan o son afectados por el sistema de software. Los alphas poseen una vital importancia en el desarrollo de software, dado que si no son considerados a lo largo del desarrollo pueden perjudicar el proyecto (p. ej., no tomar en cuenta a los stakeholders a la hora de definir sus requisitos puede perjudicar a todas las actividades posteriores de desarrollo).

Figura 2. Los alphas (demarcados por símbolos alfa) y las relaciones entre ellas



Fuente. Jacobson, Spence, & Ng (2013).

El grado de progreso del alpha es definido a través de una serie de estados, de forma conjunta los estados pueden indicar el grado de progreso (también conocido como estado global) de un proyecto. Para que un alpha logre un estado, este debe cumplir con el checklist de su estado, por ejemplo, en la figura 3, el alpha “Sistema de software”

podrá atravesar los siguientes estados: Con arquitectura seleccionada, Demostrable, Usable, Operacional, Listo, y Retirado; para lograr el estado “Con arquitectura seleccionada” (el 1.^{er} estado de los 6 que posee), debe cumplir con todo su checklist, en el ejemplo, aún se debe seleccionar las plataformas, tecnologías y lenguajes que se usarán, y se deben tomar decisiones de compra, construcción y reuso de componentes de software. De la misma forma, se deberá hacer lo mismo para cada uno de los siguientes estados.

Figura 3. Estados del alpha Sistema de Software con sus respectivos checklist

α Sistema de software Con arquitectura seleccionada ✓ Se seleccionó la arquitectura que trata los riesgos técnicos clave ✓ Se acordaron los criterios para seleccionar la arquitectura ✗ Se seleccionaron las plataformas, tecnologías y lenguajes ✗ Se tomaron las decisiones de compra, construcción y reuso 1/6	α Sistema de software Demostrable • Se demostraron las características clave de la arquitectura • Los interesados relevantes acordaron que la arquitectura es apropiada • Se ejercieron la interfaz crítica y las configuraciones del sistema 2/6	α Sistema de software Usable • El sistema es usable y tiene las características de calidad deseadas • Los usuarios pueden operar el sistema • Se aceptaron los niveles de defectos • Se conoció el contenido de liberación 3/6
α Sistema de software Operacional • El sistema se usó en un ambiente operacional • El sistema está disponible para los usuarios previstos • Al menos un ejemplo del sistema es completamente operacional • El sistema es compatible con los niveles de servicio acordados 5/6	α Sistema de software Listo • Se puso a disposición la documentación de usuario • Los representantes de los interesados aceptaron el sistema • Los representantes de los interesados quieren que se haga operacional el sistema 4/6	α Sistema de software Retirado • No se da más soporte al sistema • No se producirán más actualizaciones al sistema • Se reemplazó o se discontinuó el sistema 6/6

Fuente. Elaboración propia.

Los checklist de Essence se basan en resultados que el equipo desea alcanzar, ello permite que Essence sea agnóstico de métodos (Jacobson, *et al.*, 2012), ya que los

checklist definen resultados independientes de la forma como estas se realicen, esto es de los métodos y prácticas que el equipo use.

Cabe resaltar que el kernel es extensible: los alphas, los estados, los checklist y en sí todo elemento del kernel puede ser extendido y personalizado para apoyar distintos tipos de desarrollo. Por ejemplo, alphas como Sprint, Product Backlog pueden ser vistos en el kernel extendido para Scrum (Park, *et al.*, 2016).

2.3 Essence en la Dirección y Monitoreo de Desarrollos de Software

Aunque Essence fue creado principalmente para la descripción y el desarrollo de métodos de desarrollo de software, puede ser usado en la dirección y monitoreo, ya que proporciona un enfoque de monitoreo basado en estados y dirección dirigido por metas, tal como lo presenta el trabajo de (Péraire & Sedano, 2014). Monitoreo basado en estados ya que a través del estado de cada alpha se puede medir el progreso de un proyecto y dirección basado en metas, puesto que el equipo define metas (con base en los checklist) para lograr un estado objetivo.

Este trabajo nos muestra, además, los pasos para usar Essence en la dirección y monitoreo, los cuales se describen a continuación:

- Ver el proyecto holísticamente: En cualquier tiempo, el equipo puede ver el estado de su proyecto (al ver cada alpha) y reflexionar sobre el grado de progreso de cada uno.
- Monitorear progreso: El equipo registra el progreso del proyecto, evaluando cada ítem del checklist que los estados de Essence definen, y comparando el progreso a través del tiempo.

- Establecer metas: El equipo decide poner metas para alcanzar cada estado objetivo, estas metas son extraídas de cada checklist del estado a alcanzar.
- Decide cómo lograr las metas: El equipo decide qué actividades realizar para alcanzar las metas asociadas a cada estado objetivo.
- Realizar las actividades: En el día a día, el equipo realiza las actividades planeadas.

Además, otro beneficio de usar Essence en la dirección y monitoreo es que propicia las discusiones al evaluar la situación de los alphas definidos por Essence. (p. ej., ¿Será que el Sistema de Software puede ser Demostrable para nuestro Cliente?, ¿hemos cumplido todos los puntos?), por lo que también es visto como un framework de pensamiento o reflexión del desarrollo (McMahon, P., 2015).

El kernel de Essence principalmente permite la dirección y monitoreo debido a los principios en los que se basa, en especial el ser práctico y accionable (Jacobson, *et al.*, 2013). El ser práctico nos permite usar los alphas y estados en el día a día, en formas de tarjetas (como se aprecia en la figura 3); y el ser accionable guía el comportamiento del equipo a tomar acciones frente a problemas u objetivos.

En este último principio, “Las cosas que siempre hacemos” (también conocido como, espacios de actividad) juegan un rol importante, ya que permiten guiar al equipo en la realización de las actividades de una forma genérica, para así lograr los estados objetivos. De forma genérica, dado que Essence no desea ser intrusivo, no desea imponer prácticas ni métodos, solo brinda una guía sobre las actividades mínimas, el equipo es quien escoge las prácticas y métodos que crea conveniente (p. ej., el espacio de actividad “Comprender Requisitos” contiene un conjunto de actividades que pueden provenir de

prácticas como Casos de Uso, Historias de Usuario, FDD - Feature Driven Development). De igual forma, “Las habilidades que siempre necesitamos tener” (también conocido como competencias) juegan un rol importante, al ser un conjunto de capacidades, conocimientos y habilidades necesarias para realizar ciertas tareas relacionada al desarrollo de software.

CAPÍTULO 3: ESTADO DEL ARTE

En este capítulo revisamos el estado actual de colaboración en las herramientas que soportan Essence en el monitoreo y dirección del desarrollo de software. Para ello describimos las herramientas existentes que están apoyando el uso de Essence y presentamos el grado de colaboratividad de estas herramientas basado en uno de sus componentes principales: la generación de conciencia.

3.1 Criterios de búsqueda

Para la revisión de la literatura sobre el estado actual de Essence, se ha realizado una búsqueda de artículos en revistas y conferencias en los bancos IEEE Xplore, ACM DL y Springer usando las siguientes palabras clave: Essence, SEMAT, Kernel Essence y Tool.

3.2 Herramientas que soportan Essence

Essence se encuentra en proceso de crecimiento, iniciando en la comunidad académica y recientemente en la comunidad profesional, para lo cual la iniciativa SEMAT hizo un llamado a la acción para el desarrollo de herramientas (SEMAT, 2018). Actualmente se han desarrollado distintas herramientas que apoyan el uso de Essence para distintos objetivos, tales como en el mapeo, desarrollo y adaptación de métodos y prácticas Essenciary (Evensen, *et al.*, 2018) y Practice Workbench (IJI, 2018), en la educación, Sim4 SEEd: Essence Kernel Puzzler (Pieper, 2015), SEMAT Game (Zapata, *et al.*, 2010) y UNI-DUE Essence Model Viewer (Brandt, *et al.*, 2017). De igual forma, en el área de monitoreo y dirección de desarrollo de software, existen herramientas como las siguientes: Semat Accelerator (Graziotin, 2013), Semat Kernel Tool (Sedano &

Péaire, 2015), Alpha State Explorer (Ivar Jacobson International, 2014) y OCMS (Senzaki & Kobayashi, 2017). Una descripción breve de estas herramientas puede ser vista en la tabla 1.

Tabla 1. Herramientas que soportan Essence

Herramienta	Descripción
Essenciary	Una herramienta para “esencializar” prácticas y métodos de ingeniería de software usando el lenguaje gráfico de Essence. Busca facilitar la adopción de Essence.
Practice Workbench	Permite descargar, instalar y usar el Kernel de Essence como resultado de la iniciativa de estandarización de SEMAT.
SIM4Sedd: Essence Kernel Puzzler	Un juego virtual que busca enseñar de forma eficiente y entretenida los elementos que deben ser tomados en cuenta en un desarrollo de software y enseñar guías para poder usar métodos y prácticas de Ingeniería de software.
SEMAT Game	Es un juego para enseñar los elementos del núcleo de Semat y sus relaciones con las prácticas de gestión de proyectos.
UNI-DUE Essence Model Viewer	Una aplicación que refina, extiende e implementa las ideas de los cuadros de Kanban y las tarjetas físicas de Essence con el fin de ser usada a nivel industrial. Usada para demostrar que con un alto grado de apoyo a la gestión de proyectos puede ser logrado con un pequeño conjunto de medios de visualización.
Semat Accelerator	Es una herramienta web de código libre que permite registrar estados para cada alpha y mostrar el estado del proyecto mediante gráficos de radares y de barras.
SEMAT Kernel Tool	Es una herramienta web open source útil en los meetings de reflexión que Essence.
Alpha State Explorer App	Herramienta móvil desarrollada por Ivar Jacobson International (IJI) que permite a los equipos de desarrollo de software realizar un seguimiento del estado de su proyecto y establecer objetivos.
OCMS Tool	Es una herramienta en línea para la gestión de checklists (por sus siglas en inglés, Online Checklist Management Tool) de Essence enfocada en usos prácticos de desarrollo.

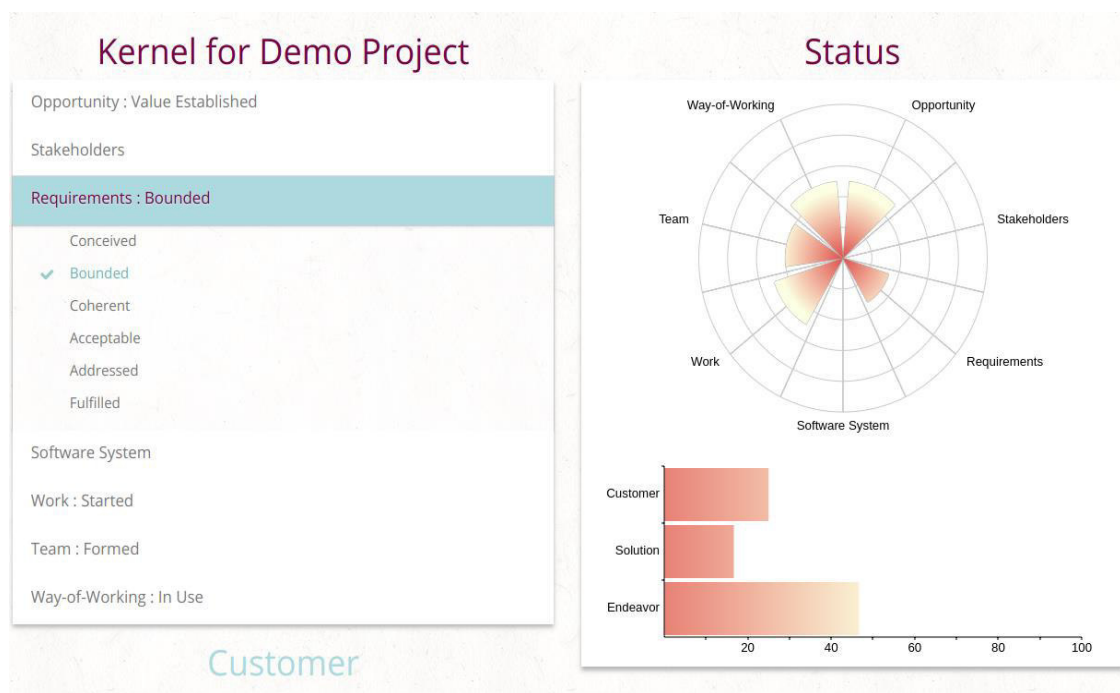
Fuente. Elaboración propia

Dado que nuestro foco de investigación es en el monitoreo y dirección de desarrollo de software, a continuación, describimos las cuatro últimas herramientas de la tabla 1.

3.2.1 Semat Accelerator

(Graziotin, 2013) presentó Semat Accelerator, una herramienta web de código libre que permite registrar estados para cada alpha y mostrar el estado del proyecto mediante gráficos de radares y de barras, como se muestra en la figura 4. Esta herramienta fue desarrollada para registrar datos empíricos en evaluaciones (como casos de estudio) sobre la efectividad de aplicar o adoptar Essence.

Figura 4 Ventana principal de Semat Accelerator



Fuente. Graziotin, (2013)

Sin embargo, esta herramienta no permite marcar cada punto del checklist de un estado, el checklist solo es mostrado de forma figurativa, tampoco permite consultar el estado a lo largo del tiempo (por lo que no soporta el concepto de iteraciones de un

proyecto). Además, su uso es limitado a un solo usuario, en vez del concepto de un equipo que trabaja en la definición de los estados de Essence.

3.2.2 SEMAT Kernel Tool

Es una herramienta web open source usada en una evaluación empírica de la reflexión que Essence provee al equipo de desarrollo a través de meetings (Péraire & Sedano, 2015). Esta herramienta permite que un equipo registre los checkpoints de cada estado de un alpha, anote acciones para lograr las metas que el equipo se ha trazado (ver sección 2.3), además de notas, en caso sea necesario (ver figura 5). La herramienta toma snapshots diariamente de estos registros con el fin de brindar al equipo el historial del estado de un proyecto. Una de sus principales características es el concepto de versiones de Essence (también conocido como plantillas), que permite a un equipo escoger los alphas y estados que crea conveniente para su proyecto, como bien dijimos en la sección 2.2, el kernel de Essence es extensible de acuerdo a la necesidad del proyecto.

Figura 5. Ventana principal de Semat Kernel Tool

SEMAT About Alphas Teams Versions Log out

Requirements⁹

1. Conceived

- ☐ List of high-level requirements captures system purpose
- ☐ User types are identified

2. Bounded

- ☐ Purpose and extent of system are agreed
- ☐ Success criteria are clear
- ☐ Processes and tools for handling requirements are in place
- ☐ Scope constraints are identified
- ☐ Assumptions made while defining requirements are captured

3. Coherent

- ☐ Requirements are captured and shared
- ☐ Important usage scenarios are explained
- ☐ Requirements are prioritized
- ☐ There are no conflicting requirements
- ☐ Rationale is understood
- ☐ Impact is understood

4. Acceptable

- ☐ Requirements are acceptable to stakeholders
- ☐ Rate of change is acceptable
- ☐ Value is clear
- ☐ Requirements are testable

5. Addressed

- ☐ Core requirements are implemented
- ☐ Stakeholders agree implemented requirements are worth making operational

6. Fulfilled

- ☐ There are no outstanding requirements to be implemented
- ☐ Implemented requirements satisfy the need for a new system
- ☐ Implemented requirements are fully accepted

Notes

Enter notes about your current situation.

Possible Actions

Enter possible actions your team can perform to achieve the next state. Add

Fuente. Péraire & Sedano, (2015)

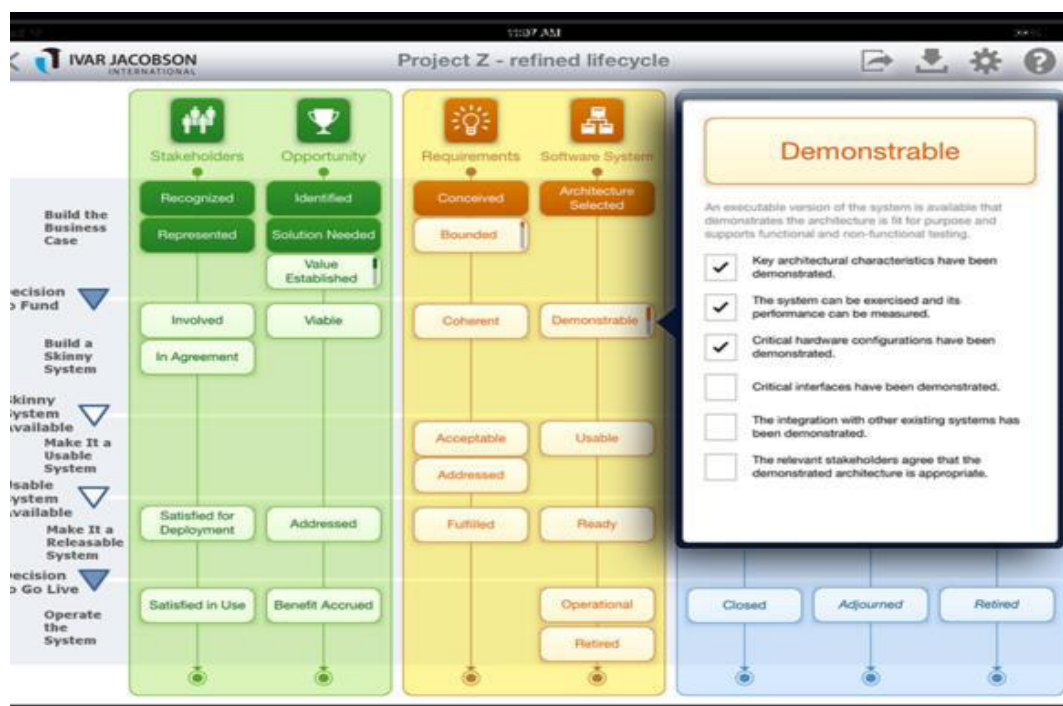
Aunque esta herramienta introduce muchos conceptos para el uso de Essence, trae consigo aspectos por mejorar. Por ejemplo, su concepto de equipo está acoplado a un proyecto, esto es, un equipo solo puede manejar un proyecto, además al permitir que un equipo acceda a un mismo proyecto, resulta fundamental conocer la autoría de las acciones de cada miembro, quizá en tiempo real. De igual manera, en los snapshots diarios, aunque útiles, no permiten mostrar de forma explícita la correspondencia de tiempo con las iteraciones de un proyecto (p. ej., alcanzamos el estado Acotado en el

alpha Requerimiento el 3 de marzo, pero ¿en qué Sprint?). Además, las acciones no están enlazadas directamente con las metas (checkpoints), sino a nivel de alphas.

3.2.3 Alpha State Explorer App

(Ivar Jacobson International, 2014) presenta Alpha State Explorer App, una versión electrónica de las tarjetas de estados de los alphas. Esta aplicación es usada por equipos para comprender dónde están y qué necesitan lograr para asegurar un proyecto exitoso y mejorar su forma de trabajo. Su desventaja se debe a que es una aplicación móvil solo disponible a dispositivos Apple, además de no mostrar la autoría de la lista de chequeo ni mostrar cambios en tiempo real. En la figura 6, se muestra la ventana principal de la aplicación Alpha State Explorer, en donde se observa el tablero de estados del proyecto y una lista de chequeo para un determinado estado.

Figura 6 Ventana principal de Alpha State Explorer



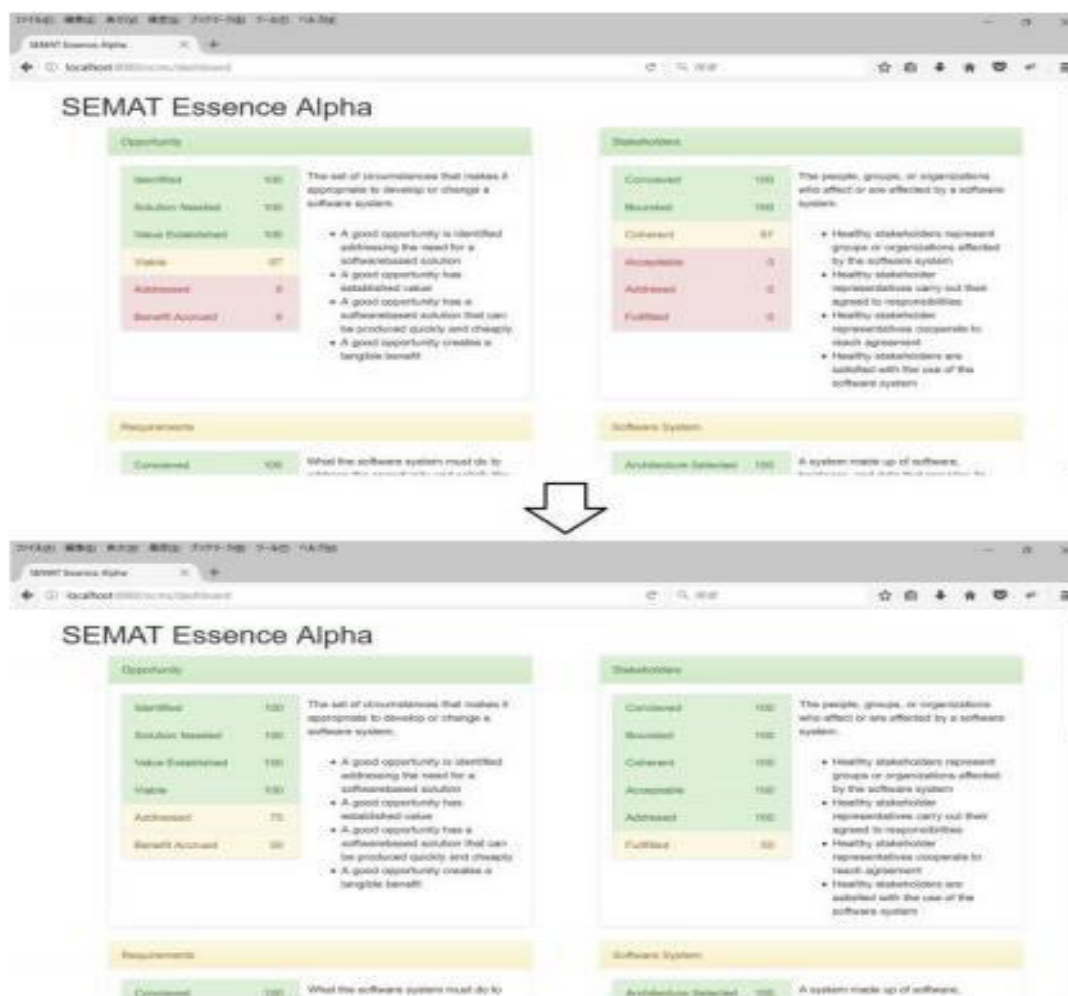
Fuente. Ivar Jacobson International, (2014)

3.2.4 OCMS Tool

(Senzaki & Kobayashi, 2017), argumentando que las actuales herramientas son problemáticas y no satisfacen suficientemente los requisitos para desarrollos prácticos, presenta OCMS, un sistema web de gestión de checklist que permite rastrear el estado de desarrollo del proyecto basados en los estados propuestos por Essence. Este sistema muestra el progreso de los alphas en un vistazo (ver figura 7), indicando una breve descripción por cada alpha y mostrando el progreso de sus estados (a través de ratios). Además, el sistema permite al equipo registrar comentarios mientras va evaluando los checklist.

Sin embargo, esta herramienta no permite que múltiples usuarios colaboren en el registro del checklist, lo cual es muy importante en entornos de trabajo remotos. Además, por el momento no permite la personalización de alphas y estados.

Figura 7. Dashboard de OCMS y progreso de los alphas



Fuente. Senzaki & Kobayashi, (2017)

3.3 Colaboración en las herramientas para dirección y monitoreo

Con el fin de ver el grado que las herramientas descritas anteriormente soportan la colaboración dentro del monitoreo y dirección, en esta sección usamos el método de evaluación de (Antunes, *et al.*, 2014), lo describimos brevemente y procedemos a realizar la evaluación.

3.3.1 Método de evaluación

Evalúamos a las herramientas descritas considerándolas como aplicaciones colaborativas o también llamadas aplicaciones CSCW (por sus siglas en inglés, Computer-Supported Cooperative Work), aunque en realidad estas sean aplicaciones de propósito general enfocadas a un usuario, ya que esto nos permite usar métodos y técnicas de evaluación conocidas en esta área. Para nuestro fin, escogimos el test de evaluación de (Antunes, *et al.*, 2014), principalmente porque brinda un enfoque práctico y estructurado de evaluación y se centra en un factor fundamental de éxito y quizá considerado el desafío principal en una aplicación colaborativa, que es la generación de conciencia, ya que estas aplicaciones deben promover que sus usuarios, los cuales no están en el mismo lugar o tiempo, permanezcan conscientes de la presencia de sus contrapartes, sus herramientas, sus conocimientos y sus metas actuales. (Neale, *et al.*, 2004).

Dicho test mide la generación de distintos tipos de conciencia (colaboración, ubicación, contexto, social, espacio de trabajo y situación) a través de un checklist (ver Anexo A). El tipo de conciencia colaboración se refiere a la percepción de los miembros sobre la disponibilidad del grupo y su comunicación, en general, cómo el equipo siente que está colaborando; la ubicación, a las relaciones geográficas entre colaboradores; contexto, relacionado con el concepto de espacios virtuales o espacios interactivos soportados por computadora; social, a la importancia de comprender la práctica social (roles y actividades de los demás) o qué y cómo contribuyen los miembros del grupo a una tarea; espacio de trabajo, asociado con el concepto de un contenedor de lugares con actividades en curso; y situación, que corresponde a tener dinámicas de percepción y

acción sobre un espacio de trabajo (p. ej., percibir eventos, interpretar lo sucedido o construir escenarios futuros). El checklist representa un conjunto de 54 elementos de diseño que un sistema colaborativo debe implementar. La evaluación de cada elemento se relaciona en diferente medida a la implementación de las tipos o categorías de conciencia con base en una tabla de correlaciones normalizadas (ver Anexo B). Por ejemplo, consideremos un caso en el que un revisor responde +1 a la pregunta #1 del checklist (“¿El sistema informa quiénes son los usuarios disponibles para colaborar?”), de acuerdo con la tabla de correlaciones, el resultado será el siguiente: conciencia de colaboración: 32.5 puntos, conocimiento de la ubicación: 6.4 puntos, conciencia del contexto: 0 puntos, conciencia social: 4 puntos, conciencia del espacio de trabajo: 4.5 puntos, conciencia de la situación: 0 puntos; ahora considere que el revisor responde -1, entonces, paralelamente, en otro resultado se acumula esta puntuación; así al final se podrá comparar los resultados positivos y negativos totales por categoría. Cabe recalcar que, dependiendo del tipo de aplicación, algunos de los elementos de diseño deben ser obviados, ya que es innecesario e incluso dañino implementar todos los elementos.

3.3.2 Evaluación

Para la evaluación de las herramientas que soportan Essence en el monitoreo y dirección, identificamos los elementos de diseño relevantes de todo el checklist (ver figura 9), enfocándonos principalmente en los tipos de conciencia: situación y colaboración, ya que estos fortalecen la noción de espacio de trabajo, la percepción de lo que está sucediendo y la disponibilidad de grupo; elementos útiles para soportar una dirección y monitoreo. Por ejemplo, se considera el elemento de diseño “disponibilidad” (relacionado con el estar consciente de los usuarios disponibles para colaborar), dado que

el elemento “disponibilidad” es útil en la colaboración al realizar la determinación del estado de un alpha, mientras que el elemento “lugares físicos” (que es una metáfora de "lugares físicos", por ejemplo, salas de reuniones y cafetería) es descartado debido a que no influye en el uso de Essence en el contexto de dirección y monitoreo. De esta forma, se han ido filtrando los elementos de diseño a un total de 20, los cuales se muestran en la tabla 2.

Tabla 2. Elementos de diseño relevantes para la evaluación en el contexto de dirección y monitoreo

Elemento de diseño	Descripción
Disponibilidad	El sistema informa quiénes son los usuarios disponibles a colaborar.
Modo de comunicación	El sistema informa si los usuarios están <i>online</i> u <i>offline</i> .
Conectividad a red	El sistema informa cuando se pierde la conectividad.
Entrega del mensaje	El sistema informa a los usuarios cuando sus mensajes son recibidos.
Lugares virtuales	El sistema soporta el concepto de lugares virtuales.
Objetos del grupo	El sistema permite a los usuarios compartir recursos.
Objetos públicos	El sistema identifica objetos públicos.
Roles	El sistema muestra los roles de usuario.
Privilegios	El sistema informa acerca de privilegios.
Historial del grupo	El sistema resalta las convenciones acordadas.
Como	El sistema indica cómo una tarea está siendo llevada.
Historial de tareas	El sistema muestra la secuencia de tarea realizadas en el tiempo.
Eventos	El sistema muestra los eventos pasados.
Acciones	El sistema muestra a los usuarios acciones en el tiempo
Recursos	El sistema muestra los objetos que cambiaron en el tiempo.
Elementos críticos	El sistema resalta la presencia de conflictos en el ambiente de trabajo (por ejemplo, eventos o situaciones).
Significados	El sistema nos da una vista estratégica sobre lo que estamos haciendo en el espacio de trabajo.
Escenarios futuros	El sistema muestra pistas acerca de futuras situaciones que puedan ocurrir en el espacio de trabajo.
Individual	El sistema provee información que ayuda a los usuarios redireccionar su curso de acción.
Colaborativo	El sistema proporciona información que ayuda a los usuarios a mantener un sentido compartido de sus objetivos y logros.

Fuente. Elaboración propia

Con base en los elementos de diseño seleccionados, calificamos la implementación de cada elemento por parte de cada una de las herramientas siguiendo el esquema descrito en la sección anterior, si el elemento de diseño es correctamente implementado por la herramienta, obtiene un puntaje positivo, en caso de que no se implementara, se obtiene un puntaje negativo, en caso de que los evaluadores no estuvieran seguros sobre la implementación, este obtiene puntaje neutro. Este sistema de puntajes se resume en la tabla 3.

Tabla 3. Puntuaciones sobre los elementos de diseño

Relevance	Implementation		Score
Relevant	Acceptable ("good")		+1
Relevant	Unacceptable ("bad")		-1
Relevant	Neutral ("do not know")		0
Irrelevant	("do not care")		0

Fuente. Elaboración propia

Los puntajes obtenidos por cada herramienta sobre los elementos de diseño seleccionados se pueden ver en la tabla 4.

Tabla 4. Puntajes para las herramientas que soportan Essence

Elementos de diseño	HERRAMIENTAS			
	Semat Accelerator	Semat Kernel Tool	Alpha State Explorer	OCMS Tool
Disponibilidad	-1	-1	-1	-1
Modo de comunicación	-1	-1	-1	-1
Conectividad a red	-1	-1	-1	-1
Entrega del mensaje	-1	-1	-1	-1
Lugares virtuales	-1	-1	-1	0
Objetos del grupo	-1	+1	+1	+1
Objetos públicos	-1	-1	-1	-1
Roles	-1	-1	-1	-1
Privilegios	-1	-1	-1	-1
Historial del grupo	-1	-1	-1	-1
Como	-1	-1	-1	-1
Historial de tareas	-1	+1	-1	+1
Eventos	-1	0	-1	0
Acciones	-1	+1	-1	0
Recursos	-1	+1	-1	+1
Elementos críticos	-1	-1	-1	-1
Significados	+1	+1	+1	+1
Escenarios futuros	-1	0	-1	0
Individual	-1	0	-1	0
Colaborativo	+1	+1	+1	+1

Fuente. Elaboración propia

Tomando en cuenta estos votos y la tabla de correlaciones (ver Anexo B), se calcularon los puntajes finales (dentro de un rango de 0 a 100) que pertenecen a cada tipo de conciencia para cada una de las herramientas (ver tabla 5).

Tabla 5. Puntajes positivos y negativos de las herramientas por categorías de conciencia

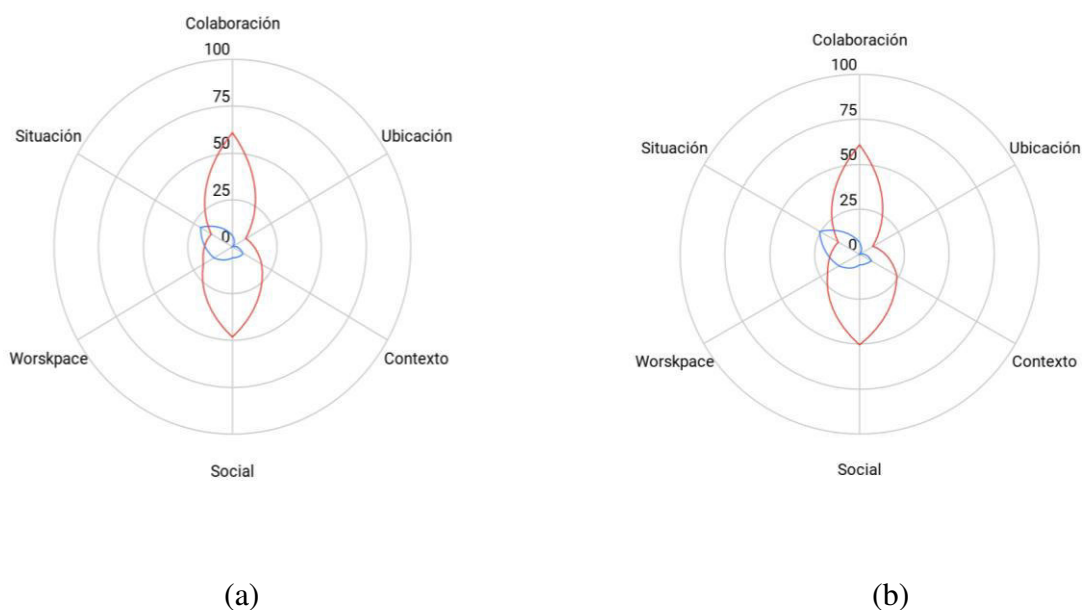
Herramientas	Categorías					
	Colaboración	Ubicación	Contexto	Social	Espacio de trabajo	Situación
Semat Accelerator	+3.7 -65.8	0 -8.2	+1.9 -32.1	+3.9 -52.6	+3.3 -34	+15 -46.6
Alpha State Explorer	+4.5 -65	0 -8.2	+6.1 -27,9	+5.4 -51.1	+6.4 -30.9	+15 -46.6
Essence Kernel Tool	+6.5 -60.8	+0 -8.2	7,8 -23,8	+6 -50.5	+13.1 -20.5	+25.9 -13.7
OCMS Tool	+5.7 -60.8	+0 -8.2	+7.2 -18.5	+6 -48.3	+11.8 -19.1	+20.9 -13.7

Fuente. Elaboración propia

En una mirada rápida podemos ver que la herramienta Semat Accelerator es la que posee menos elementos de diseño implementados, probablemente es debido a que ella fue una de las primeras herramientas en soportar Essence; fue lanzada en el 2013, y contempló muchas menos características que las demás, de similar manera la herramienta Alpha State Explorer (2014) consigue puntuaciones bajas. Por otro lado, SEMAT Kernel Tool consigue un grado positivo en la conciencia sobre la situación (+25. 9), esto es debido principalmente a que ella fue diseñada para equipos, permitiendo así que sus usuarios vean el estado a proyecto a través de snapshots y gráficos, además de enviar información sobre este a través de correo electrónico. En el caso de OCMS, una de las herramientas más recientes, solo nos basaremos en la descripción presentada en su artículo, ya que no accedimos a usarlo, como sí sucedió con SEMAT Kernel Tool, bajo esta premisa ella posee casi igual generación de conciencia que SEMAT Kernel Tool e incluso reduce en un grado menor los puntajes negativos en el contexto, mas debido a que

se desconoce si tiene algún registro de actividades, el grado positivo a la situación es menor (ver figura 8).

Figura 8. Puntaje obtenido por (a) Semat Kernel Tool, (b) OCMS, línea azul: indica puntajes positivos, línea roja: indica puntajes negativos



Fuente. Elaboración propia

Como hemos visto, las herramientas actuales no ponen en primer plano el trabajo colaborativo dentro de la dirección y monitoreo del desarrollo de software, esto se debe, como antes mencionamos, a que no fueron diseñadas para tal fin. Sin embargo, rescatamos que la mayoría de las herramientas tienen en consideración la generación de conciencia sobre la “situación”, a nivel del estado y progreso del proyecto, ello es debido a las característica de practicidad de Essence; el usar los alphas en el día a día permite que el equipo sea consciente de cómo va su situación.

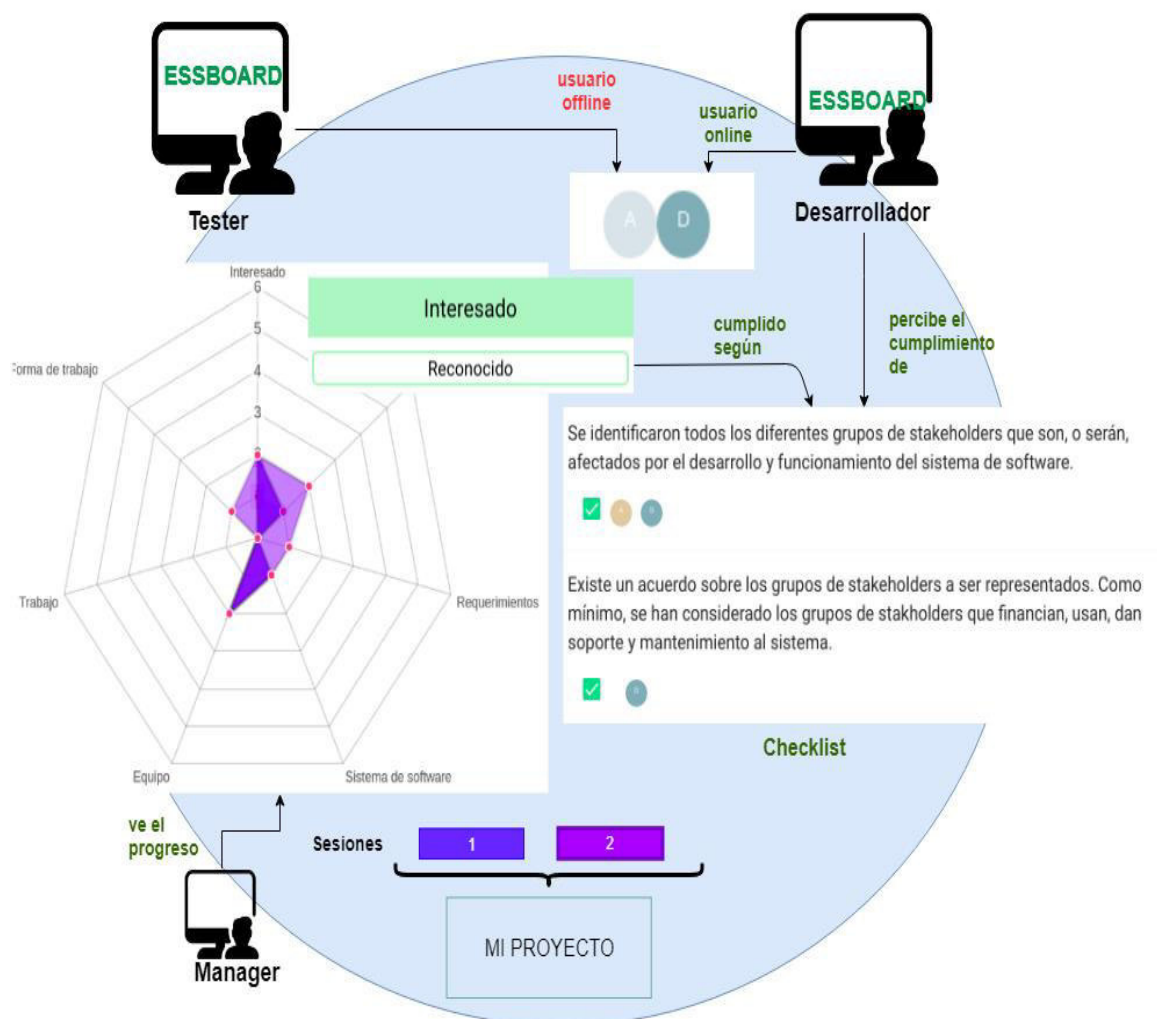
CAPÍTULO 4: ESSBOARD

En este capítulo se introduce Essboard, una herramienta web colaborativa que apoya la dirección y monitoreo de proyectos de software usando el marco Essence. Presentamos una vista general de Essboard y describimos cómo las funcionalidades apoyan a la dirección y monitoreo, finalmente, describimos su diseño e implementación.

4.1 Una vista general

Essboard es una herramienta web para direccionar y monitorear proyectos de desarrollo de software colaborativamente basado en los alphas y estados de Essence. Toma un enfoque colaborativo al brindarle a los equipos de desarrollo un entorno de trabajo que promueve la colaboración en la determinación del estado actual de un proyecto y las metas que desea alcanzar, facilitando que cada miembro del equipo comparta su percepción de los aspectos claves del proyecto en tiempo real usando los checklist de Essence (ver figura 9). Essboard busca que los equipos logren una visión común del estado y progreso de un proyecto, al brindarles una visión holística del proyecto y de sus sesiones de trabajo. Además, las sesiones de trabajo permiten que Essboard pueda ser usado en cualquier momento, sea en una iteración (o Sprint), un meeting en el inicio de un proyecto, o incluso una auditoría por parte de un stakeholder externo (p. ej., un consultor); ya que en ellas el equipo define el estado del proyecto, las metas y acciones para lograrlas con base en las percepciones que cada miembro del equipo tiene sobre la realización de los estados de cada alpha (Péaire & Sedano, 2014).

Figura 9. Essboard: una visión general



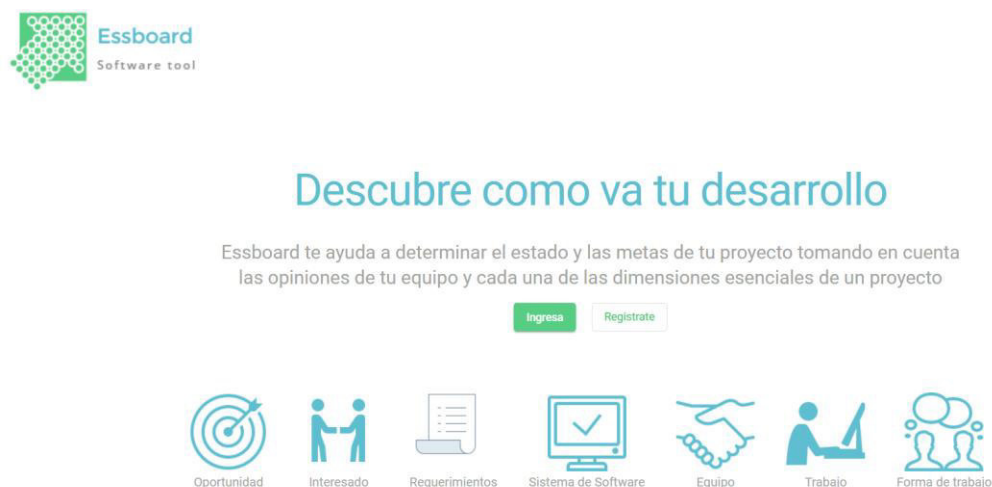
Fuente. Elaboración propia

4.2 Interactuando con la herramienta

Para direccionar y monitorear un proyecto de software con Essboard, los usuarios interactúan con las funcionalidades que ofrece el sistema, esta interacción comienza al acceder a la página de inicio (ver figura 10), aquí un usuario podrá autenticarse (o registrarse si es que fuese la primera vez que entran a Essboard), ya dentro del sistema, verá un listado de proyectos propios o compartidos por otros usuarios, y podrá crear un

nuevo proyecto si así lo desea, indicando el nombre y descripción del proyecto. Una acción importante es acceder a un proyecto, ya que esto da inicio a la dirección y monitoreo del mismo.

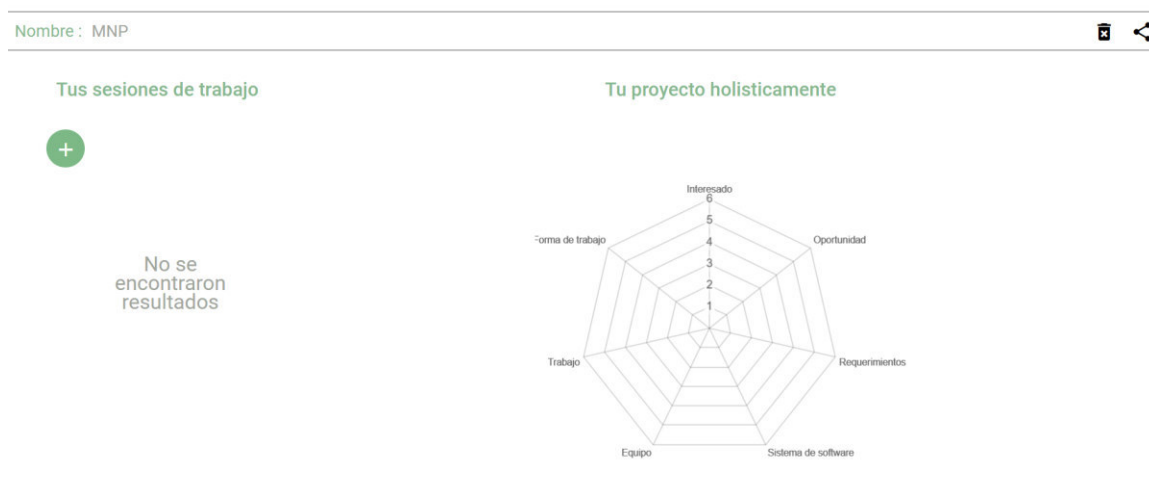
Figura 10. Página de inicio de Essboard



Fuente. Elaboración propia

Cuando el usuario accede a un proyecto, Essboard muestra las sesiones pertenecientes al proyecto (en caso de un proyecto nuevo, ninguna sesión) y un gráfico de radar que resume su estado (ver figura 11).

Figura 11. Vista de un proyecto en Essboard



Fuente. Elaboración propia

En esta pantalla, además, se aprecia, en el lado derecho de la zona superior, un icono referente a “compartir”, el cual permite seleccionar qué usuarios de Essboard pueden recibir una invitación a colaborar en el proyecto. Para compartir un proyecto, se debe buscar por un nombre de usuario y seleccionar alguno, realizada la invitación, este usuario recibirá una notificación y podrá observar el proyecto compartido en su lista de proyectos.

Ya creado un proyecto con todos los miembros disponibles a colaborar, en cualquier momento se puede crear sesiones de trabajo, la única restricción existente es que se debe concluir una sesión para empezar otra. Al crear una sesión de trabajo, Essboard brinda un espacio de trabajo (ver figura 12), el cual muestra a los usuarios elementos importantes para promover la colaboración en la determinación del estado actual y los objetivos de un proyecto en las dimensiones o alphas que Essence provee.

Figura 12. Espacio de trabajo de una sesión en Essboard

Sesión 18 de marzo Dar por concluido

D

A

Ir a mi proyecto

1 Estado actual

2 Determina tu estado

Determina el estado actual o objetivo de los siguientes aspectos de tu proyecto

Interesado

Oportunidad

Requerimientos

Sistema de software

Equipo

Trabajo

Forma de trabajo

Iniciar evaluación

Equipo

Sembrado

Formado

Colaborando

Ejecutando

Suspendido

Equipo

Sembrado

- La misión del equipo es clara
- El equipo conoce como lograr una misión
- Las competencias requeridas han sido identificadas.
- Se ha determinado el tamaño del equipo.

1/5

✓

🚩

☒

La misión del equipo se ha definido en términos de oportunidades y resultados.

👤

→

!

✓

🚩

☒

Están definidas las reglas de gobierno.

👤

→

!

✓

🚩

☒

El modelo de liderazgo ha sido seleccionado.

👤

→

☒

✓

🚩

☒

👤

→

☒

Actividades

A

arthur opinó que el check 51-10 ha si...

18 mar. 2019

A

arthur opinó que el check 51-1 ha sid...

18 mar. 2019

A

arthur opinó que el check 31-4 ha sid...

7 mar. 2019

A

arthur opinó que el check 31-3 ha sid...

7 mar. 2019

Conversación

-----Inicio de la conversación-----

D

di...

7

A

Empezaremos por el alpha Equipo

D

di...

7

D

di...

7

Escriba aquí su mensaje

Fuente. Elaboración propia

Por ejemplo, un chat (ubicado en la esquina inferior derecha), los usuarios participantes de la sesión (ubicado en la parte superior central) y la capacidad de ver votos en tiempo real, es aquí donde el equipo podrá determinar el estado y las metas de su proyecto usando los elementos del kernel de Essence (alphas, estados y checklists). Adicionalmente, en la sección 5.1, se muestra cómo las funcionalidades de Essboard son usadas en un escenario común.

4.3 Essboard y su apoyo a la dirección y monitoreo

Essboard tiene como objetivo apoyar el monitoreo basado en estados y dirección dirigido por metas que ofrece Essence, el cual se plasma en pasos, descritos en la sección 2.3. A continuación describiremos cómo las funcionalidades de Essboard brindan soporte a cada uno de los pasos de dicho enfoque:

1. Ver el proyecto holísticamente

En Essboard, los miembros de un proyecto pueden ver el proyecto y cada una de las sesiones de manera holística. En un proyecto, el equipo podrá observar las sesiones de trabajo y un gráfico de radar que representa el progreso de las 5 últimas sesiones de trabajo, en cada sesión (dibujada por una línea del radar), se pueden ver los estados que se logró por cada alpha. En una sesión de trabajo, podrá observar de igual forma un gráfico de radar con los estados logrados en cada alpha para dicha sesión. En especial, el gráfico de radar puede servir para ver el progreso de un proyecto en cada una de las dimensiones, sirviendo además al punto siguiente.

2. Monitorear progreso

Essboard brinda un espacio de trabajo donde el equipo puede definir el estado de su proyecto, al evaluar el estado en que se encuentra cada alpha a través de checklist, y votar según su perspectiva si se ha logrado o no un determinado checkpoint. El progreso de un alpha se presenta a través de una tarjeta del alpha con estados de diferentes tonos de color: normal, si todo el equipo marcó que todos los check se cumplieron, color opaco, si es que hay algunos check cumplidos, y no tienen color de fondo, si es que ningún check se cumplió.

3. Establecer metas

Essboard indica los estados siguientes a alcanzar, el equipo, con base en ello, escoge el estado objetivo, de ello Essboard extrae las metas tomando como base el checklist del estado objetivo y los pone como metas para la siguiente sesión de trabajo.

Cabe resaltar que Essboard no apoya al paso “como lograr las metas”. Essboard, al basarse en Essence, no indica formas de cómo lograr metas, por ser independiente de prácticas y métodos, el equipo será el que decida qué actividades realizar para alcanzar las metas asociadas a cada estado objetivo. De igual forma, esto sucede con el paso “realización de actividades”.

4.4 Proceso de desarrollo

Essboard fue desarrollado tomando en cuenta varias prácticas y artefactos pertenecientes a eXtreme Programming (XP). Por ejemplo, los autores realizaron varias sesiones de Pair Programming al desarrollar una funcionalidad compleja. Cabe aclarar, que el desarrollo de Essboard no fue riguroso en seguir el método XP, de hecho, combinó muchas

prácticas de otros métodos, por ejemplo, en vez de usar Historias de Usuario para describir los requisitos a implementar, se optó por mostrarlas como Características (Features en Feature Driven Development).

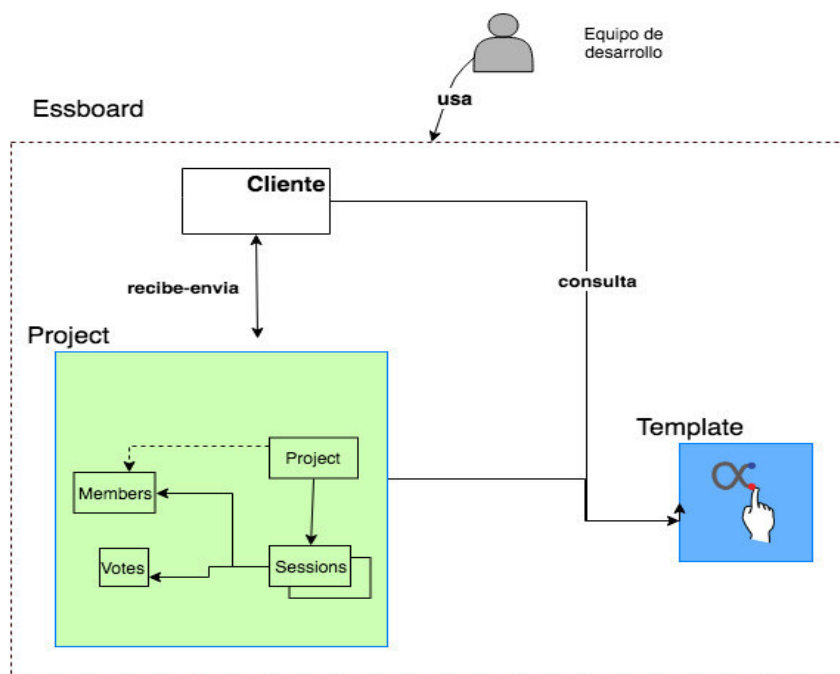
En general, el proceso de desarrollo inició teniendo algunas funcionalidades a desarrollar, obtenidas de las herramientas existentes para Essence y herramientas colaborativas conocidas y altamente usadas (como Trello o Google Drive, Hangouts, C9 Cloud IDE), además se agregaron nuevas funcionalidades a medida que el desarrollo avanzaba³. En cada iteración (de 2 semanas), se priorizo y se seleccionaron las funcionalidades a desarrollar. Además, se discutía las interacciones que el usuario tendría con el sistema para llevar a cabo una funcionalidad, con el fin de poder desarrollar una UI que simulará el flujo de interacción de un usuario, esta parte era importante, dado que la UI definía la forma que iba a consumir y solicitar servicios (siguiendo el enfoque Consumer Driven Contracts) a través de primero trabajar con solicitudes al servidor con respuestas falsas (Mock data) y posteriormente con la implementación real. Cabe resaltar, que muchas veces el diseño de algún servicio implicó iterar en la arquitectura y diseño del sistema (Architectural Spike), especialmente en el módulo Project (tal como se plasma en la sección 4.5.2) para ello los autores estuvieron realizando distintos prototipos y pruebas para ver que diseño cumpliera con los requisitos del sistema, para ello se realizaron muchas sesiones de Pair Programming (como se vio en el párrafo anterior). Finalmente, cada funcionalidad era desplegada (usando un ciclo de Delivery Continuo) en caso que se detectara bugs en alguna funcionalidad ya desplegada, éstas se registraban para su posterior corrección.

³ Tablero de características de Essboard, <https://trello.com/b/B3btv7bj/essboard>, último acceso el 19/03/19

4.5 Diseño e implementación

Essboard fue diseñado para cumplir con la colaboración en tiempo real y el monitoreo de proyectos de desarrollo de software, su arquitectura (ver figura 13) busca cumplir dichos objetivos a través de 3 componentes, siguiendo a (Fowler, 2003) componentes que creemos que son los más importantes y la base del sistema, estos son: Template, Cliente y Project. El componente Template provee la información de los elementos de Essence, el componente Project se encarga de procesar toda información relacionada a un proyecto (sesiones, miembros, votos o percepciones sobre cada elemento del checklist), y el componente Client es la interfaz web al equipo de desarrollo.

Figura 13. Arquitectura de Essboard



Fuente. Elaboración propia

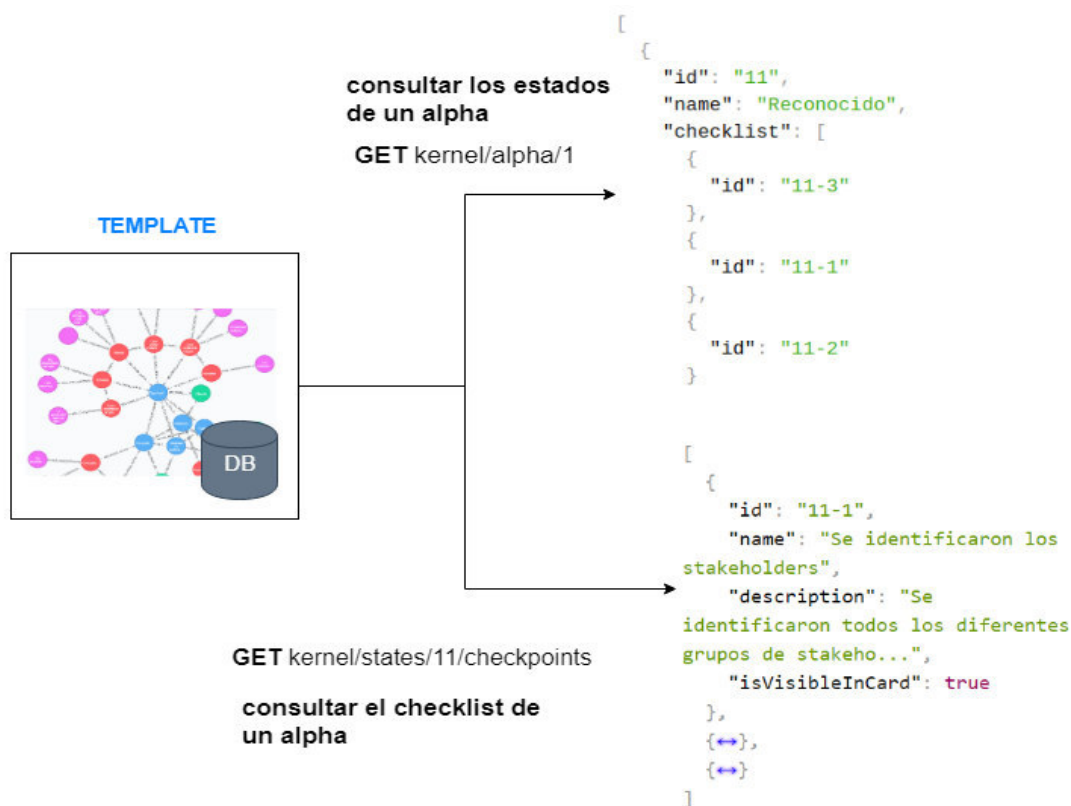
En general, los componentes trabajan juntos todo el tiempo para lograr una función del sistema, por ejemplo, para establecer el estado de un alpha, el Cliente pide al componente Template la información de un alpha determinado, el componente Cliente

recibe la información y los muestra en pantalla a través de una tarjeta; si un usuario va a emitir un voto, entonces el Cliente comunica al componente Project sobre este evento, Project procesa el voto (para ello a su vez lee información del Template) y le muestra al componente Cliente el resultado. Internamente, cada componente posee cierto grado de complejidad, por lo cual a continuación describiremos cada una con más detalle.

4.5.1 Template

El componente Template es un sistema en sí mismo que se centra en proveer información sobre el núcleo de Essence a través de servicios REST (ver figura 14), independientemente de cualquier proyecto (o sesiones), además, permite sin mucho esfuerzo que a futuro se trabaje con alphas, estados y checkpoints personalizados.

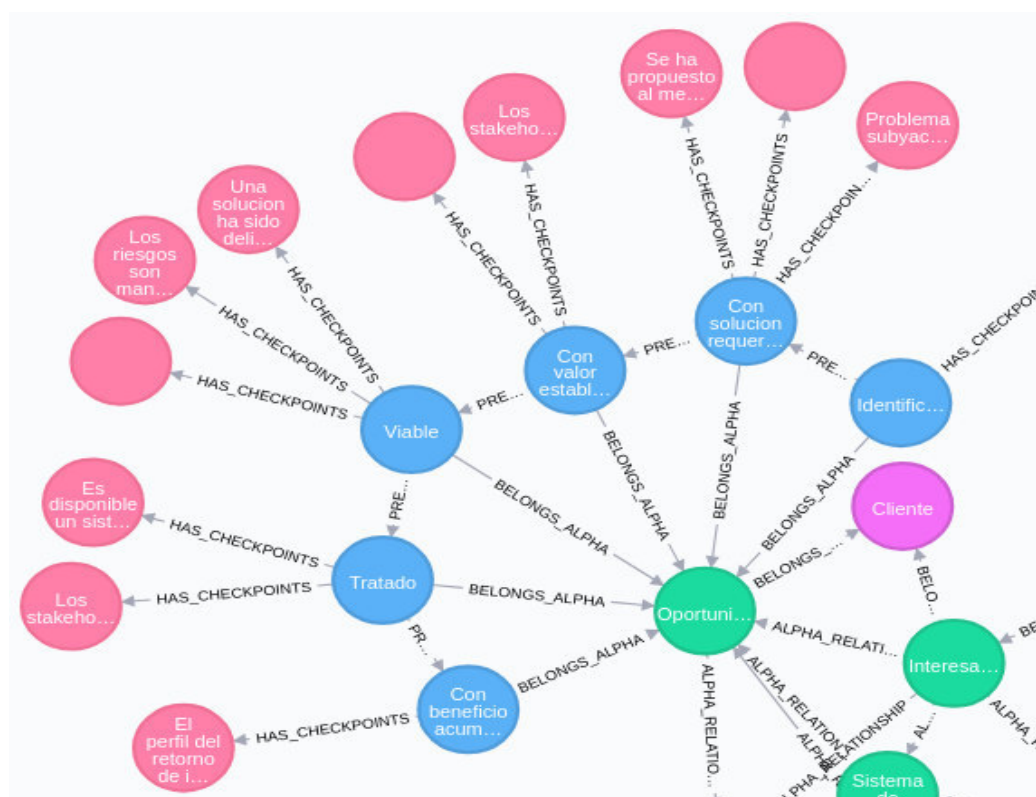
Figura 14. Componente Template



Fuente. Elaboración propia

Almacenar estos datos representó un desafío, ya que si bien se podía guardar todo en memoria, las extensas relaciones entre los elementos hacía muy tedioso este camino, puesto que el núcleo de Essence representa una ontología de elementos intensamente conectados (Exman, I, 2014); dado este escenario, se decidió almacenar estas relaciones y elementos a través de una base de datos orientado a grafos, en nuestro caso Neo4j, ya que este enfoque facilita las consultas sobre elementos con muchas relaciones, permite registrar los elementos como nodos independientes y luego conectarlos a través de las relaciones que Essence presenta. Este enfoque nos permitió conocer que Essence contiene 257 nodos y 302 relaciones, como ejemplo, en la figura 15, podemos visualizar un conjunto de nodos de color verde que representa a los alphas, los cuales se conectan a otros alphas (relación entre alphas), además, nodos de color azul que representan a los estados y vemos cómo estos se relacionan con los nodos de color verde (relacion alpha y estados) y, por último, los nodos de color rojo que se conectan con los nodos de color azul (relación estado y checkpoints).

Figura 15. Essence como un grafo

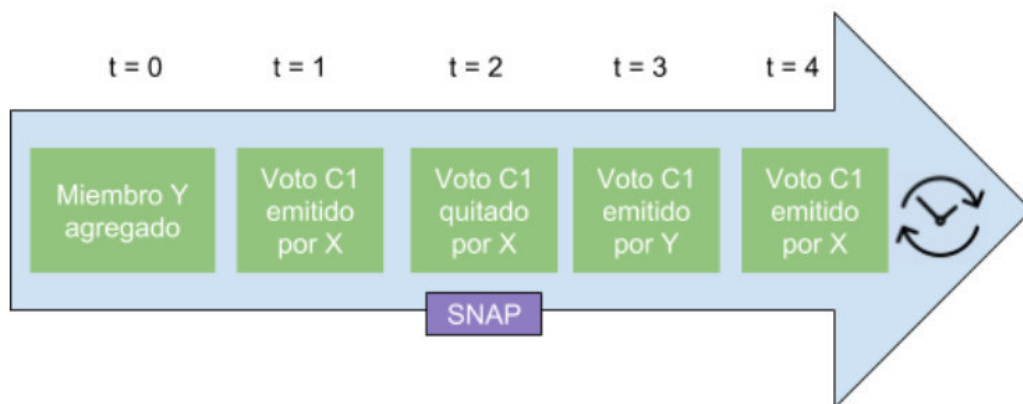


Fuente. Elaboración propia

4.5.2 Project

El componente Project posee la lógica de creación de proyectos, sesiones y votaciones a checkpoints. Dado que Essboard tiene como objetivo la colaboración, este componente se encarga de brindar características con soporte al tiempo real, principalmente en los votos (las percepciones de cada miembro sobre el estado del proyecto). Además, como Essboard busca servir en el monitoreo, toda información debe ser registrada en lo posible, por ello Essboard usa Event Sourcing (Young, 2012) para poder registrar cualquier cambio, en forma de eventos, en este caso todo cambio al estado de un proyecto. Por ejemplo, en la figura 16, se observa unos posibles eventos (descritos por cuadros verdes) que pudieron ocurrir en un proyecto (se agregó un miembro Y al proyecto, un miembro X realizó un voto a un checkpoint C1, entre otros) en distintos tiempos de manera secuencial.

Figura 16. Vista de los eventos de un proyecto a través del tiempo



Fuente. Elaboración propia

Cabe recalcar que estos eventos están estrechamente relacionados a los eventos en los websockets, es decir, cada evento se emite en tiempo real a los usuarios correspondientes; en caso sea un voto emitido, sus usuarios serán los participantes de la misma sesión de trabajo, en caso sea un miembro agregado, sus usuarios serán los

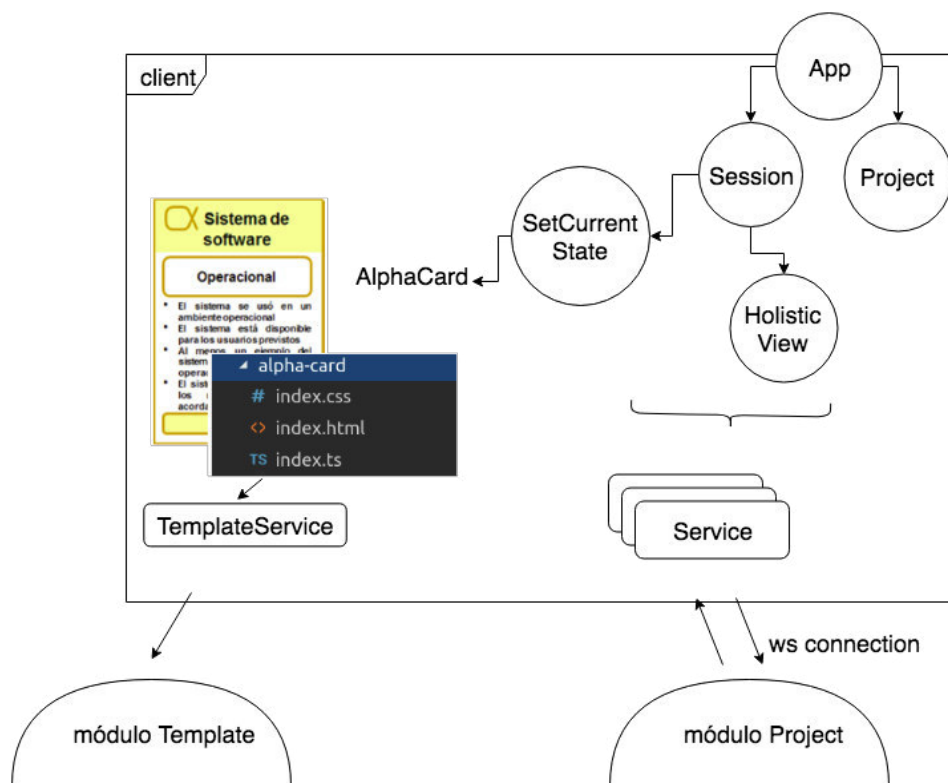
miembros del mismo proyecto. No solo ello, a través de Event Sourcing, Essboard puede consultar el estado de un proyecto en cualquier tiempo (en especial en las sesiones de trabajo), pudiendo servir a distintos ámbitos (p. ej., la auditoría de un proyecto). Por ejemplo, tomando en cuenta los eventos de la figura 16, si se evaluara cada evento hasta el tiempo $t=2$, el número de votos sería 0, dado que el usuario X votó y quitó su voto, mientras que el número de votos en el tiempo $t=4$ son 2. Sin embargo, en un proyecto con un gran número de eventos, el rendimiento puede disminuir, por lo que, en vez de evaluar todos los eventos, podemos tomar un snapshot (una foto del momento) en un determinado tiempo y evaluar desde el snapshot hacia los eventos posteriores a él. Por ejemplo, bajo el mismo caso, si tomáramos un snapshot (SNAP en la figura) en el $t=2$, tendríamos 0 votos, en caso de que quisiéramos obtener el número de votos actuales, simplemente evaluamos los eventos en $t=3$ y $t=4$ y los sumamos al número de votos del snapshot (0), con el cual obtendremos 2 votos. El uso de snapshots es, en este caso, fundamental, dado que es una aplicación en tiempo real, de otra forma, el tiempo de latencia crecería a lo largo del tiempo.

4.5.3 Cliente

El componente Cliente se encarga de mostrar la información de un proyecto, como sus sesiones, y procesar las interacciones de los usuarios, como en la determinación de estados y metas, es en sí la interfaz web de la aplicación. Este componente se comunica con el componente Project principalmente a través de una conexión websocket, permitiendo así escuchar cualquier evento que Project quiera comunicar. También se comunica con el componente Template, para consultar la información de los alphas, estados y checkpoints para mostrar al usuario a través de tarjetas. En general, este

componente está estructurado como un árbol de componentes web (ver figura 17), por ejemplo, al ingresar a una sesión de trabajo, podremos ver un gráfico de radar (definido por el componente HolisticView) o determinar el estado actual del proyecto (definido por el componente SetCurrentState), donde, a su vez, en este último podremos ver una lista de alphas (definido por el componente AlphaCard que se encarga de mostrar tarjetas alpha de Essence). Además, cada uno de estos componentes, para mostrar su información correspondiente o encargarse de la interacción de un usuario, se comunica a través de servicios a los módulos correspondientes.

Figura 17. Estructura del módulo cliente



Fuente. Elaboración propia

Cabe resaltar que este componente emplea los colores que usa Essence para mostrar las tarjetas de estados y checklist, así, busca asemejarse en lo posible a las

tarjetas físicas originales de Essence, además, en sí usa estos colores como piel de toda la aplicación.

4.5.4 Tecnologías

Essboard fue implementado principalmente usando tecnologías open source, todos los componentes siguieron un formato estándar, buscamos reforzar ello a través del uso de herramientas de formato y linting de código; en nuestro caso, usamos TSLint y ESLint como herramientas de linting y Prettier como formateador de código. Cada uno de las tecnologías usadas en los componentes fue escogida por su aporte al diseño y novedad. A continuación, detallaremos las tecnologías usadas por cada componente.

- Cliente: Desarrollado con el framework Angular, para el diseño de interfaces web usamos la librería Angular Material, que nos apoyó en desarrollar rápidamente las interfaces y con ello tener un diseño consistente en toda la aplicación.
- Template: Expone servicios web desarrollado con el framework NestJS, internamente se conecta al driver de Neo4j, al cual le envía consultas en el lenguaje Cypher al motor de base de datos de Neo4j alojado en un servidor gratuito de base de datos llamado GrapheneDB.
- Project: Dado la necesidad de comunicación en tiempo real, este componente fue desarrollado con FeathersJS, el cual expone servicios REST a través de http y websockets, además, escogimos como base de datos a MongoDB por su fácil integración (es la más común en el ecosistema de Nodejs), ella fue alojada en el servicio gratuito MLab. El soporte a event sourcing fue desarrollado bajo nuestros propios medios.

Cada uno de los componentes fue desplegado en servicios en la nube, los componentes Template y Project están alojados en Heroku, mientras que el componente Client, en Netlify. Sin embargo, estos servicios están siendo usados bajo el nivel de servicio más bajo, ya que por el momento Essboard no es de uso comercial.

Finalmente, Essboard puede ser accedido mediante el siguiente link: <https://www.our-academy.org/posts/essboard>.

4.6 Limitaciones técnicas

Debido a las tecnologías y las decisiones de diseño tomadas, Essboard posee las siguientes limitaciones:

- Soporte a navegadores:
 - Internet Explorer 9, 10, 11
 - Edge, Safari y iOS (2 últimas versiones)
 - Android (desde v5), aunque actualmente no posee un diseño responsivo
 - Chrome, Firefox (últimas 5 versiones)
- El tiempo de respuesta al mostrar los gráficos de radar varía entre 2 a 5 segundos, principalmente debido al nivel de servicio de los servicios en la nube.
- Número de proyectos por usuario debe ser menor a 20
- Número de usuarios concurrentes debe ser menor a 1000

CAPÍTULO 5: VALIDACIÓN

En este capítulo se valida que la herramienta propuesta ayuda en la dirección y monitoreo de proyectos de software colaborativamente. Para cumplir este propósito, mostramos cómo Essboard soporta un escenario de dirección y monitoreo en un proyecto de retail basándonos en el escenario de (Kajko-Mattsson, M., 2014), que originalmente usaba tarjetas físicas de Essence, y evaluamos el grado de colaboración de Essboard usando el método descrito en la sección 3.3. Finalmente, analizamos las ventajas y limitaciones de Essboard comparándolo con las tarjetas físicas y otras herramientas que soportan Essence.

5.1 Caso de estudio

En esta sección presentamos cómo Essboard soporta el escenario de uso descrito en (Kajko-Mattsson, M., 2014), en el que se usa tarjetas físicas de Essence para direccionar y monitorear proyectos de desarrollo en las reuniones de reflexión de Essence en un proyecto de retail.

El escenario describe una serie de eventos en el inicio de un proyecto de mejora de un sistema legado de venta al por menor denominado RIS, aplicando Essence por parte de un equipo de 3 personas: Cynthia, Fred y Sam; una información más detallada sobre el proyecto se presenta en la tabla 6.

Tabla 6. Información general del proyecto RIS

¿Para quién?	HomeWare Retailer, una cadena de tiendas que ofrece productos de calidad para la construcción.
El problema	Su sistema informático, el Retailer Information System (RIS), busca atender a 200 minoristas adicionales, pero este no puede soportarlo, ya que es una aplicación heredada que fue desarrollada hace unos cuarenta años.
Objetivo	Un nuevo proyecto con la misión de modernizar RIS
Equipo de proyecto	Formado por un equipo muy experimentado: Cynthia, Fred y Sam; ellos tienen diferentes conocimientos tecnológicos (p. ej., lenguajes de programación) y nunca antes han trabajado juntos. Todos tienen buen conocimiento de Essence a través de la universidad, por haber leído papers o libros.

Fuente. Elaboración propia

5.1.1 Dirección y monitoreo usando tarjetas físicas

El escenario inicia en el desarrollo de la primera reunión del proyecto, el cual tiene como objetivo definir el estado del proyecto, por lo que el equipo decide realizar un “poker planning”. Cynthia reparte a cada miembro un mazo de tarjetas de alphas y estados, empiezan a evaluar el alpha Oportunidad; Sam explica que cada miembro debe determinar el estado logrado a través del cumplimiento de los ítems de su checklist (un ejemplo de estos diálogos puede ser visto en la figura 18). Luego de unos minutos, todos

completan su evaluación y muestran sus tarjetas. Todos han concordado que el estado Valor Establecido es el que ha sido logrado. El equipo acuerda en llegar al estado Viable para la siguiente reunión y anotan una tarea: Identificar y evaluar riesgos.

Figura 18. Reuniones usando tarjetas de Essence



Fuente. Sedano, T., & Péraire, C. (2015)

El equipo repite este proceso para cada alpha de Essence, en caso de que el equipo no tenga un acuerdo sobre qué estado de un alpha se logró, el equipo buscará alguna forma de acordar, ya sea exponiendo las ideas por las que estén en desacuerdo o seleccionando el estado que la mayoría acuerda; además, en este proceso no es mandatorio establecer la meta para cada alpha, ya que así el equipo puede focalizarse en lograr solo aspectos del proyecto importantes para el momento (p. ej., la iteración).

Al finalizar la reunión, el equipo tiene el registro de las tareas a realizar, el estado del proyecto y los estados meta por alpha.

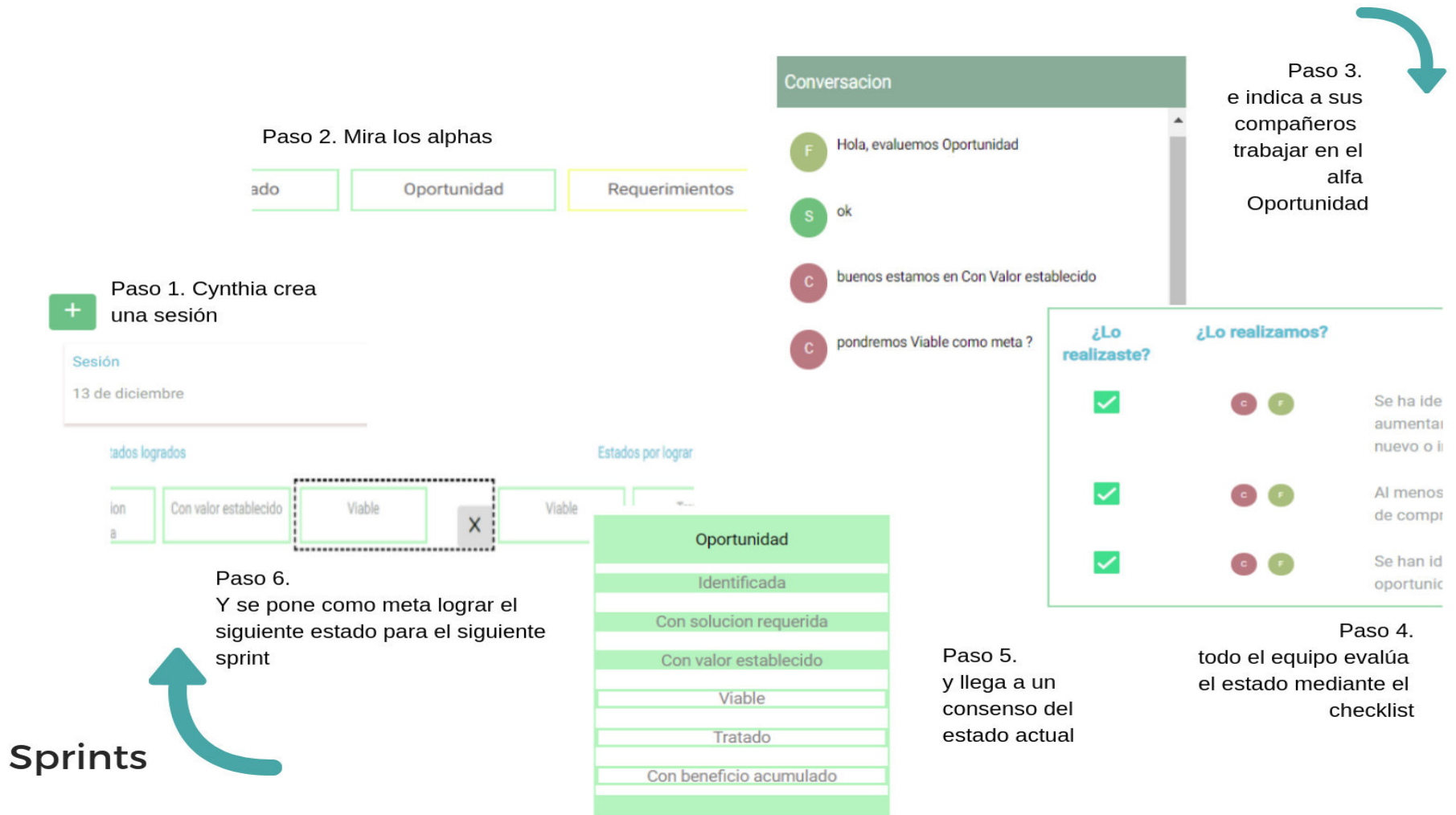
5.1.2 Dirección y monitoreo usando Essboard

Para iniciar la reunión, Cynthia crea una sesión de trabajo en el proyecto RIS (ver paso 1 de la figura 19), proyecto previamente registrado en Essboard junto con las invitaciones a colaborar a cada miembro del equipo. Sin embargo, Cynthia no tendrá la necesidad de repartir a cada miembro un mazo de tarjetas alphas, ya que cada miembro

cuando ingrese a Essboard, específicamente al espacio de trabajo de la sesión, podrá ver tarjetas virtuales de Essence (ver paso 2 de la figura 19), además, Cynthia podrá enterarse si los demás miembros ingresaron al espacio de trabajo observando los usuarios activos en la parte superior de la pantalla (similar como lo muestra Google Docs), de tal manera que esté segura de la participación de todo el equipo al realizar las evaluaciones del estado del proyecto.

Siguiendo con el escenario, el equipo decide empezar a evaluar el alpha Oportunidad, lo cual es comunicado por Cynthia a través del chat del espacio de trabajo, característica indispensable cuando existe equipos distribuidos; Sam, de igual forma, comunica las pautas para poder lograr un estado a través del cumplimiento de los ítems de su checklist (ver paso 3 de la figura 19). Cada miembro empieza a evaluar el alpha Oportunidad, recorre estado por estado y evalúa su checklist, los resultados son mostrados al costado de cada checkpoint al finalizar (ver paso 4 de la figura 19) los minutos planeados para la evaluación (aunque Essboard brinda la capacidad de poder ver las evaluaciones de cada checklist en tiempo real, si fuera necesario).

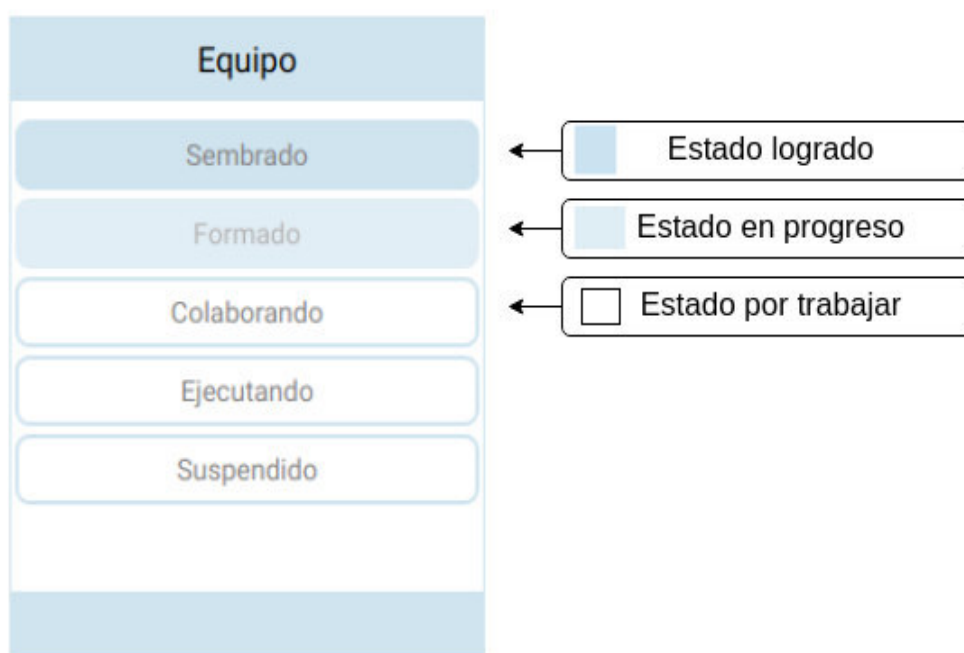
Figura 19. El proceso del meeting de Essence en Essboard en el alpha Oportunidad



Fuente. Elaboración propia

El equipo empieza la evaluación del alpha Oportunidad, al finalizar el tiempo de evaluación, Essboard indicará el cumplimiento de cada uno de los estados evaluados a través de diferentes tonalidades (ver figura 20), donde un tono natural corresponde a un estado logrado, tono opaco indica un estado en progreso (algunos checkpoints logrados) y un fondo blanco indica que el estado no ha sido evaluado, es decir, que no se han realizado votos. En este caso, el equipo acordó que el alpha Oportunidad logró el estado “Valor Establecido”, por lo tanto, Essboard mostrará un tono natural, como se muestra en el paso 5 de la figura 19.

Figura 20. Tonalidades de colores en una tarjeta de estado en Essboard



Fuente. Elaboración propia

El equipo, además, acuerda en llegar al estado Viable para la siguiente reunión (ver paso 6 de figura 19) y anotan una tarea en el chat: Identificar y evaluar riesgos. El equipo repite el mismo proceso de evaluación de estado para cada alpha de Essence. Al

finalizar la reunión, el equipo tiene el registro en el chat de las tareas a realizar, el estado del proyecto y los estados meta por alpha.

5.2 Colaboración en Essboard

Para evidenciar que Essboard posee características colaborativas, evaluamos el grado de colaboración de Essboard usando el mismo test descrito en la sección 3.3.1, considerando 3 escenarios: optimista, medio y pesimista. El escenario pesimista describe aquella evaluación que solo considera con votos positivo a aquellos elementos de diseño que se evidencian a simple vista en las funcionalidades que ofrece la herramienta (p. ej., el elemento “disponibilidad de los usuarios” se muestra explícitamente en el espacio de trabajo); el escenario medio considera con puntaje positivo a aquellos elementos de diseño que han sido implementados con una funcionalidad o una combinación de funcionalidades usual (p. ej., usualmente los equipos usan el chat como un registro de los acuerdos, lo cual hace cumplir el elemento “historial de grupo”); y el escenario optimista considera, además, alternativas, al combinar varios funcionalidades que afectan de manera indirecta la realización de los elemento (p. ej., el elemento “historial de tareas” se puede lograr al mostrar las tareas en cada una de las sesiones).

Los puntajes obtenidos por cada escenario sobre los elementos de diseño se pueden ver en la tabla 7.

Tabla 7. Puntajes para Essboard en los 3 escenarios de evaluación

Elementos de diseño	Escenarios		
	Pesimista	Medio	Optimista
Disponibilidad	+1	+1	+1
Modo de comunicación	+1	+1	+1
Conectividad a red	+1	+1	+1
Entrega del mensaje	-1	-1	-1
Lugares virtuales	+1	+1	+1
Objetos del grupo	+1	+1	+1
Objetos públicos	-1	-1	-1
Roles	-1	-1	+1
Privilegios	-1	-1	-1
Historial del grupo	-1	+1	+1
Como	-1	-1	+1
Historial de tareas	-1	-1	+1
Eventos	+1	+1	+1
Acciones	+1	+1	+1
Recursos	+1	+1	+1
Elementos críticos	-1	-1	-1
Significados	+1	+1	+1
Escenarios futuros	-1	-1	-1
Individual	+1	+1	+1
Colaborativo	+1	+1	+1

Fuente. Elaboración propia

Tomando en cuenta estos votos, se calcularon los puntajes finales que pertenecen a cada tipo de conciencia para cada una de los escenarios (ver tabla 8).

Tabla 8. Puntajes positivos y negativos de las herramientas por categorías de conciencia

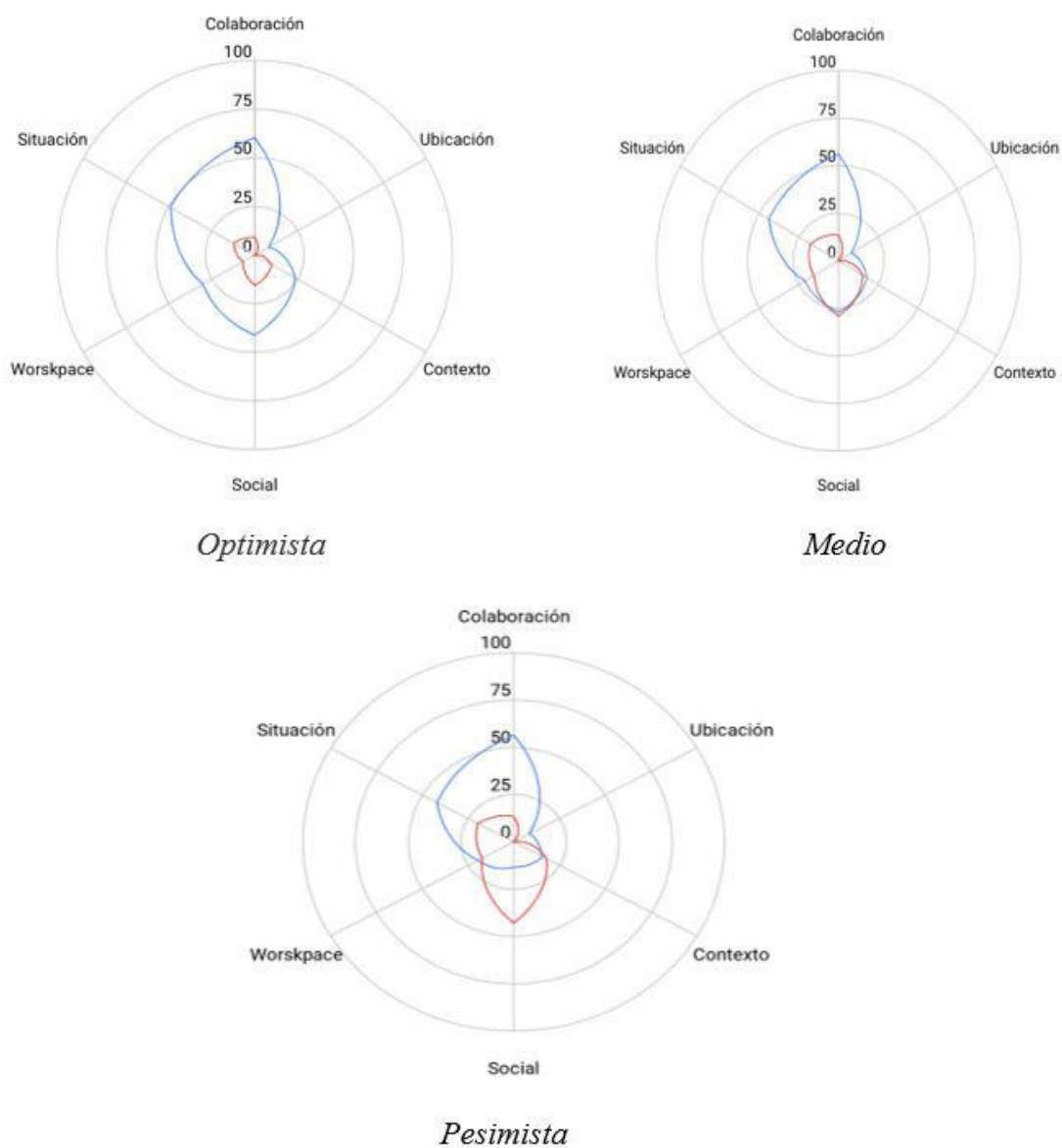
Escenarios	Categorías					
	Colaboración	Ubicación	Contexto	Social	Espacio de trabajo	Situación
Pesimista	+56,2/-13,3	+8,2/0	+15,7/-18,3	+13,7/-42,8	+20/-17,3	+42/-19,6
Medio	+56,2/-13,3	+8,2/0	+18,1/-15,9	+27,1/-29,4	+21,4/-15,9	+43,9/-17,7
Optimista	+60,6/-8,9	+8,2/0	+23,8/-10,2	+41,1/-15,4	+30,5/-6,8	+49/-12,6

Fuente. Elaboración propia

A primera vista, en la figura 21, se observa un contraste entre los tres escenarios, sobre todo en las categorías situación, colaboración y espacio de trabajo, considerando, además, que estos siempre obtuvieron un puntaje positivo en los tres escenarios. Y al igual como sucedió con las herramientas examinadas en el capítulo 3, existe un mayor soporte sobre conciencia de la situación, debido a que implícitamente adquieren la propiedad practicidad de Essence que influye en la generación de conciencia sobre la situación. Además, podemos observar que no hay variación en la conciencia sobre la ubicación (se mantiene en 8.2 puntos), al ser una categoría poco relevante en la dirección y monitoreo usando Essence, ya que conceptos como espacios físicos no son aplicables. Quizá la mayor limitación de Essboard, al observar los gráficos de radar, es el aspecto social, ello es debido a que ciertos elementos de diseño que promueven esta categoría (como el historial de grupo, elementos públicos y privilegios) no se han implementado.

Figura 21. Puntaje obtenido por los escenarios,

línea azul: indica puntajes positivos, línea roja: indica puntajes negativos



Fuente. Elaboración propia

5.3 Discusión

5.3.1 Soporte a la dirección y monitoreo

En la sección 4.3, se describió que Essboard apoya actividades de dirección y monitoreo de esfuerzos de software; en el caso de estudio se evidenció que Essboard apoyó la realización de estas actividades tan igual a las tarjetas físicas. Mas existen ventajas de usar Essboard frente a las tarjetas físicas, principalmente, cada actividad en las sesiones de trabajo queda registrada. Todo diálogo, voto sobre un elemento del checklist en la sesión de trabajo e incluso las invitaciones a un proyecto son registradas y pueden ser consultadas fácilmente para distintos fines, como auditorías. En especial, Essboard puede mostrar gráficos de radares del progreso del proyecto en las últimas sesiones de trabajo. Además, el uso de tarjetas virtuales hace innecesario que cada miembro tenga un mazo de tarjetas alphas, estados y checklist, lo cual resulta bastante tedioso cuando el número de miembros de un equipo es mediano y existen varios proyectos dentro de la organización en la que se trabaja.

Cabe recalcar que, aunque la colaboración natural que se da en las reuniones de equipo es insuperable, Essboard brinda funcionalidades básicas para la colaboración (p.ej., chat en la sesión de trabajo o evaluaciones en tiempo real) en las actividades que suceden en una reunión de dirección o monitoreo, como se vio en el caso de estudio.

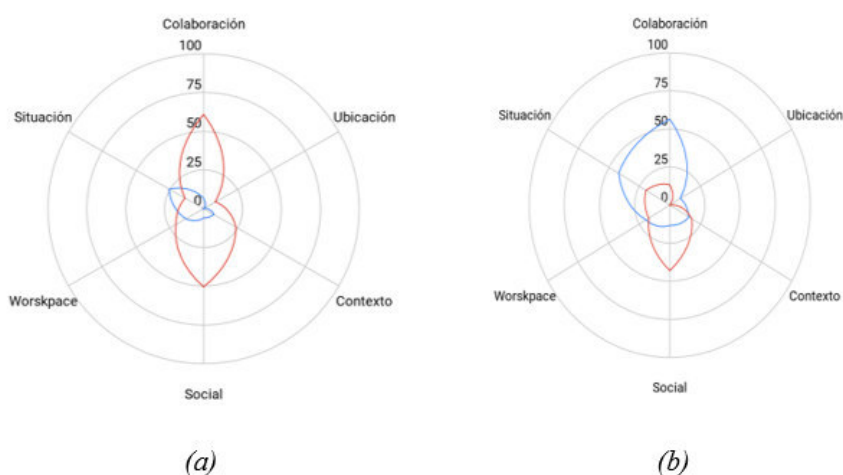
Sin embargo, siguiendo el primer valor del manifiesto ágil (Beck, *et al.*, 2001): Individuos e interacciones sobre procesos y herramientas, el equipo debe usar Essboard solo cuando lo crea conveniente, por ejemplo, cuando se requiera reuniones presenciales, en especial en organizaciones donde las reuniones virtuales aún no son maduras.

5.3.2 Comparando Essboard

Como vimos en la sección 3.3.2, las herramientas OCMS y Semat kernel tool son las que poseen mejor grado de colaboración en el soporte a Essence en el contexto de dirección y monitoreo, entre estas, se escogió a Semat Kernel Tool como un punto de comparación, debido a que posee prácticamente el mismo grado de colaboración que OCMS.

Al comparar el resultado pesimista de Essboard con el resultado de Semat Kernel Tool, se observa que existe un notorio mayor soporte colaborativo por parte de Essboard, en especial en las categorías que teníamos como objetivo: situación del proyecto y colaboración (ver figura 22). Como también en la categoría de espacio de trabajo, este último es debido principalmente a que Essboard brinda facilidades para ayudar a la coordinación de la determinación de estados (p.ej., ver la disponibilidad de los otros miembros o un chat para la coordinación). En el caso de Essence Kernel tool, vimos que, gracias a su capacidad de consultar los estados sobre fechas pasadas, este pudo tener votos positivos en la categoría de situación, sin embargo, posee en la mayoría resultados negativos, si bien posee funcionalidades importantes, estas no soportan la colaboración, al ser esta una herramienta de propósito general. De igual forma, se podría hacer una comparación de Semat Kernel Tool frente a los otros escenarios de evaluación (medio y optimista), más las mismas conclusiones se obtendrán, e incluso el margen será aún mayor a favor de Essboard.

Figura 22. Evaluación de Semat Kernel Tool (a) y Essboard en el escenario pesimista (b)



Fuente. Elaboración propia

Cabe indicar que, al evaluar las herramientas, notamos que todas ellas poseen un grado positivo en conciencia sobre la situación, ello demuestra que Essence, en general, apoya a la generación de conciencia sobre los aspectos del proyecto, creemos que ello es debido al principio de practicidad del cual está basado; al proveer los alphas y estados en el día a día al equipo, genera de cierta manera conciencia sobre la situación del estado de cada alpha y, por ende, del proyecto en sí.

5.3.3 Limitaciones de la validación

La validación de este trabajo se ha limitado a la validación del soporte a la dirección y monitoreo a través del soporte a un escenario de (Kajko-Mattsson, M., 2014) y a la validación del soporte de la colaboratividad a través del método de (Antunes, *et al.*, 2014), además de su comparativa, usando los resultados del mismo método, frente a la herramienta Semat Kernel Tool, evidenciando que la herramienta propuesta consigue un mejor resultado a las herramientas actuales.

CAPÍTULO 6: CONCLUSIONES

6.1 Conclusiones

En este trabajo se presentó Essboard, una herramienta web colaborativa que apoya la dirección y monitoreo de desarrollos de software basado en los alphas y estados de Essence. Identificamos 9 herramientas que soportaban Essence, y, en el contexto de dirección y monitoreo, 4. Se realizó una evaluación a estas últimas herramientas con un test que mide el grado de generación de conciencia, una característica que todo sistema colaborativo busca lograr, así, vimos que estas no ponían en primer plano la colaboración.

Se desarrolló Essboard teniendo en primer plano el soporte al trabajo colaborativo a través de la implementación de una serie de funcionalidades, como chat, votos sobre elementos de un checklist en tiempo real, gráficos de radar, disponibilidad de otros usuarios, entre otras. Estas funcionalidades se implementaron gracias al manejo de websockets para la comunicación en tiempo real, event sourcing para el registro de cualquier actividad y base de datos orientada a grafos para organizar los elementos y sus relaciones entre estas que presenta Essence.

El soporte a la colaboración por parte de Essboard se ve reflejado en los resultados obtenidos en el test de generación de conciencia sobre los escenarios pesimista, medio y optimista, donde se obtuvo puntajes superiores al compararlas con la mejor herramienta para la dirección y monitoreo actual, consiguiendo en el escenario pesimista una mejora en las dimensiones colaboración, contexto, social, espacio de trabajo y situación del 669.86, 93.83, 128.33, 57.48, 62.16 %, respectivamente. Además,

se mostró, a través de un caso práctico, que Essboard puede soportar reuniones comunes de dirección y monitoreo de proyectos, donde tarjetas físicas de Essence eran usadas.

6.2 Trabajos futuros

Aún existen muchas mejoras que pueden realizarse a Essboard. Al realizar el caso práctico, pudimos notar la posibilidad de integrar Essboard con herramientas de planeación de tareas (p. ej., Trello), dado que la evaluación basada en estados sirve de entrada para la planeación de tareas. De igual forma, al evaluar el grado de generación de conciencia, pudimos notar que aún existen mejoras que se deben realizar en la categoría social de la herramienta, tales como poder crear proyectos públicos; además de la personalización de elementos de Essence, a través de plantillas, para que así un proyecto se pueda evaluar con base a un conjunto de alphas y estados definidos por el mismo equipo.

REFERENCIAS

- Antunes, P., Herskovic, V., Ochoa, S. F., & Pino, J. A. (2014). Reviewing the quality of awareness support in collaborative applications. *Journal of Systems and Software*, 89, 146-169.
- Beck, K., & Fowler, M. (2001). *Planning extreme programming*. Addison-Wesley Professional.
- Brandt, S., Striewe, M., Beck, F., & Goedicke, M. (2017). A Dashboard for Visualizing Software Engineering Processes Based on ESSENCE. In: *Working Conference on Software Visualization (VISOFT)*, pp. 134-138. IEEE, Shanghai.
- Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M. & Kern, J. (2001). *Manifesto for agile software development*.
- Booch, G. (2018). The History of Software Engineering. *IEEE Software*, 35(5), 108-114.
- Cockburn, A., & Highsmith, J. (2001). Agile software development, the people factor. *Computer*, 34(11), 131-133.
- Cunningham, D. (2013). Enabling Fujitsu's industrialized delivery of application services. In *Essence Information Day Berlin (2013)*. OMG Technical Meeting. Recuperado el 20 de Marzo de 2019 de <http://www.omg.org/news/meetings/tc/berlin-13/special-events/essence-pdfs/S7-Cunningham.pdf>.
- Ellis, C. A., Gibbs, S. J., & Rein, G. (1991). Groupware: some issues and experiences. *Communications of the ACM*, 34(1), 39-58.
- Evensen, A., Kemell, K. K., Wang, X., Risku, J., & Abrahamsson, P. (2018). Essencery – A Tool for Essentializing Software Engineering Practices. *CoRR abs/1808.02723*
- Exman, I. (2014). A bootstrap theory: the SEMAT kernel itself as runnable software. In *Proceedings of the 3rd SEMAT Workshop on General Theories of Software Engineering* (pp. 9-12). ACM.
- Fowler, M. (2003). Who Needs an Architect? *IEEE Software*.
- Graziotin, D., and Abrahamsson, P. (2013). A Web-based modeling tool for the SEMAT Essence theory of Software Engineering. *Journal of Open Research Software*, 1.
- Kajko-Mattsson, M. (2014). A Scenario on Kick-Starting a Project. Recuperado el 21 de 05 de 2016, de SEMAT: www.semat.org/introduccion

- IJI, Ivar Jacobson International. (2018), Trial Practice Workbench | Ivar Jacobson International, <https://www.ivarjacobson.com/publications/tool/esswork-practice-workbench>, último acceso 2018/12/14.
- Ivar Jacobson International (2014), Alpha State Explorer App | Agile Services | Ivar Jacobson International, <https://www.ivarjacobson.com/alpha-state-explorer-app>, último acceso 2018/10/10
- Jacobson, I., Spence, I., & Ng, P. W. (2013). Agile and SEMAT- Perfect Partners. *Communications of the ACM*, 53-58.
- Jacobson, I., Meyer, B. and Soley, R. (2009) Call for action: The SEMAT initiative. *Dr. Dobb's Journal*.
- Jacobson, I., Ng, P. W., McMahon, P. E., Spence, I., & Lidman, S. (2013). *The essence of software Engineering: applying the SEMAT kernel*. Addison-Wesley.
- Jacobson, I., Ng, P. W., McMahon, P. E., Spence, I., & Lidman, S. (2012). The essence of software engineering: the SEMAT kernel. *Communications of the ACM*, 55(12), 42-49.
- McMahon, P. (2015). A “Thinking Framework” to Power Software Development Team Performance. *CrossTalk*, 22-27.
- McDonough, A (2013). *Munich Re and ESSENCE – Kernel and Language for Software Engineering Methods: A Case Study*. Recuperado el 20 de Marzo de 2019 de https://www.bpmi.org/news/whitepapers/Munich_Re_Essence_Case_Study-2014-12-01_JP.pdf
- Morales-Trujillo, M. E., Oktaba, H., & Orozco, M. J. (2016). Using ESSENCE ALPHAs in a CMMI level 5 software development organization. In *2016 Federated Conference on Computer Science and Information Systems (FedCSIS)* (pp. 1531-1538). IEEE.
- Neale, D. C., Carroll, J. M., & Rosson, M. B. (2004, November). Evaluating computer-supported cooperative work: models and frameworks. In *Proceedings of the 2004 ACM conference on Computer supported cooperative work* (pp. 112-121). ACM.
- OMG, Object Management Group. (2015). *Essence Kernel and Language Software Engineering Method 1.1*. Recuperado el 28 de 06 de 2016, de [www.omg.org: http://www.omg.org/spec/Essence/1.1](http://www.omg.org/spec/Essence/1.1)
- Pan-Wei Ng, (2014) *Software Process Improvement and Gaming using Essence: An Industrial Experience*. *Journal of Industrial and Intelligent Information*

- Park, S., McMahon, & Myboght. (2016). Scrum Powered by Essence. ACM SIGSOFT Software Engineering Notes, 1-8.
- Péraire, C., & Sedano, T. (2014, May). State-based monitoring and goal-driven project steering: Field study of the semat essence framework. In Companion Proceedings of the 36th International Conference on Software Engineering (pp. 325-334).
- Péraire, C., & Sedano, T. (2014). Essence Reflection Meetings Field Study. Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering.
- Pieper, J. (2015). Discovering the Essence of Software Engineering - An Integrated Game-Based Approach based on the SEMAT Essence Specification. In: Global Engineering Education Conference (EDUCON), pp. 939-947. IEEE, Tallinn.
- Sedano, T., & Péraire, C. (2015). Enhancing Student Experience in Team-Based Project Courses Using Essence Reflection Meetings. In 2015 IEEE 28th Conference on Software Engineering Education and Training (CSEET) (pp. 10-12). IEEE.
- Sedano, T., Rengasamy, A., & Péraire, C. (2016, April). Green-lighting proposals for software engineering team-based project courses. In 2016 IEEE 29th International Conference on Software Engineering Education and Training (CSEET) (pp. 175-183). IEEE.
- SEMAT (2018). SEMAT Practice Area, Tools. Recuperado el 23 de 01 de 18, de: http://old.semat.org/indexfcf7.html?page_id=1262
- Senzaki, Y., Liu, S., Washizaki, H., Fukazawa, Y., Kobayashi, H., & Adachi, M. (2017). Web Application to Manage and Improve Software Development Projects by SEMAT Essence. In: Proceedings of the 5th International Workshop on Quantitative Approaches to Software Quality co-located with 24th Asia-Pacific Software Engineering Conference APSEC, pp. 38–42. CEUR-WS.org, Nanjing.
- Seymour, E. (2015). We all have different ways to do things and that's OK. In Essence-in-Practice Conference, Berlin (2015). OMG Technical Meeting. Recuperado el 20 de Marzo de 2019 de <http://www.omg.org/news/meetings/tc/berlin-15/special-events/essence-presentations/seymour.pdf>
- Sommerville, I. (2011). Software engineering (9th ed., pp.35-36). Boston, Pearson, Addison Wesley.
- Shane Hastie, S. W. (2015). Standish Group 2015 Chaos Report - Q&A with Jennifer Lynch. Recuperado el 20 de 04 de 2016, de infoq: <http://www.infoq.com/articles/standish-chaos-2015>

- Whitehead, J. (2007). Collaboration in software engineering: A roadmap. In 2007 Future of Software Engineering (pp. 214-225). IEEE Computer Society.
- Young, G. (2012). Event centric: Finding simplicity in complex systems. Addison-Wesley.
- Zapata, C., & Jacobson, I. (2014). A first course in software engineering methods and theory. *Dyna*, 81(183), 231-241.
- Zapata-Jaramillo, C. M., Lopez, M. D. R., Sanchez, R. E. A., & Pinzon, L. D. J. (2010). SEMAT GAME: applying a project management practice. In *Developments in Business Simulation and Experiential Learning: Proceedings of the Annual ABSEL conference*, 133–143.

ANEXOS

ANEXO A: Elementos de diseño del método de calidad de conciencia

Tomado de (Antunes, *et al.*, 2014)

Elementos de diseño	Preguntas
Disponibilidad	El sistema informa que usuarios están disponibles a colaborar.
Modo de comunicación	El sistema informa si otros usuarios están trabajando en línea, offline o ambos.
Conectividad de red	El sistema informa cuando la conectividad a la red se ha perdido o se ha recuperado.
Entrega de mensajes	El sistema informa al usuario cuando su mensaje fue recibido por el receptor.
Retrasos de mensajes	El sistema informa a los usuarios del tiempo que se gastó en enviar un mensaje.
Ubicaciones cartesianas	El sistema indica la ubicación de posibles colaboradores.
Ubicaciones topológicas	El sistema informa si otros usuarios están cerca.
Distancias	El sistema indica las distancias físicas a otros usuarios.
Orientaciones	El sistema indica la orientación de otros usuarios (por ejemplo, si se están moviendo recto).
Rango de atención	El sistema da pistas del centro de actividad de otros usuarios (por ejemplo, que están mirando).
Modalidad de ubicación	El sistema indica el tipo de movilidad de los usuarios (por ejemplo, si están deambulando visitando o viajando).
Nivel de movilidad	El sistema reconoce el tipo de movilidad del dispositivo (por ejemplo, si el dispositivo está parado o en movimiento por un usuario o de manera autónoma).
Relación con otros dispositivos	El sistema reconoce si el dispositivo es independiente de otros dispositivos integrados en otro dispositivo (por ejemplo, un automóvil) o si se extiende por todo el entorno.
Restricciones físicas	El sistema aborda las restricciones impuestas por el entorno físico donde se utiliza.
Lugares físicos	El sistema da una metáfora de lugares físicos.
Topología física	El sistema da indicaciones sobre la complejidad del entorno físico donde se utiliza.

Atributos físicos	El sistema da indicaciones sobre las condiciones ambientales del lugar donde se usa (por ejemplo, condiciones climáticas).
Lugares virtuales	El sistema apoya el concepto de "lugares virtuales" diferentes lugares para la colaboración).
Topología virtual	El sistema representa la topología del entorno virtual (es decir, el desplazamiento entre lugares virtuales).
Mapas de vista	El sistema brinda una vista general del entorno virtual.
Miradores	El sistema permite a los usuarios mirar otras actividades.
Objetos grupales	El sistema permite a los usuarios compartir proyectos.
Objetos públicos	El sistema identifica objetos públicos.
Atributos virtuales	El sistema muestra los atributos de objetos/recursos en el espacio de trabajo.
Relaciones virtuales	El sistema muestra las relaciones entre objetos/recursos en el área de trabajo.
Restricciones virtuales	El sistema proporciona indicaciones sobre las restricciones de objeto/recurso (como ubicación o propiedad).
Roles	El sistema muestra los roles de los usuarios.
Actividades	El sistema da pistas sobre las actividades actuales de los usuarios.
Privilegios	El sistema informa al usuario sobre los privilegios de los demás en el sistema.
Historial de grupo	El sistema destaca las convenciones/protocolos acordados por los usuarios para colaborar (por ejemplo, quién dirige la discusión).
Quien	El sistema indica quién está haciendo una determinada tarea.
Qué	El sistema muestra la actividad siendo realizada por una persona determinada.
Donde	El sistema indica el lugar donde un usuario está trabajando actualmente.
Cuando	El sistema informa cuando una tarea ha sido realizada.
Como	El sistema da indicaciones sobre cómo está (o se realizó) una tarea.
Historial de tareas	El sistema muestra la secuencia de tarea realizadas en el tiempo.
Retroalimentación	El sistema provee retroalimentación sobre las acciones realizadas.
Alimentación	El sistema notifica acerca de las actuales actividades.
Retroalimentación del canal	El sistema notifica al usuario si los demás están siguiendo lo que está haciendo.
Señales por mirada	El sistema da pistas sobre dónde están mirando los usuarios.
Pistas de voz	El sistema proporciona comentarios sobre quién está hablando con quién.

Actividades paralelas	El sistema indica si los usuarios están realizando actividades paralelas.
Actividades coordinadas	El sistema indica si los usuarios están realizando actividades coordinadas (por ejemplo, a través de un flujo de trabajo).
Actividades mutuamente ajustadas	El sistema informa si los usuarios están realizando actividades ajustadas mutuamente (es decir, modificando su propio trabajo de acuerdo con otras actividades).
Control de acceso	El sistema notifica quién tiene el control de un objeto/recurso compartido.
Eventos	El sistema muestra los eventos pasados.
Acciones	El sistema muestra a los usuarios acciones en el tiempo.
Recursos	El sistema muestra los objetos que cambiaron en el tiempo.
Elementos críticos	El sistema resalta la presencia de conflictos en el ambiente de trabajo (por ejemplo, eventos o situaciones).
Significados	El sistema nos da una vista estratégica sobre lo que estamos haciendo en el espacio de trabajo.
Escenarios futuros	El sistema muestra pistas acerca de futuras situaciones que puedan ocurrir en el espacio de trabajo.
Individual	El sistema provee información que ayuda a los usuarios redireccionar sobre su curso de acción.
Distribuido	El sistema da pistas sobre los cambios en el sistema que pueden ser relevantes para la toma de decisiones
Colaborativo	El sistema proporciona información que ayuda a los usuarios a mantener un sentido compartido de sus objetivos y logros.

ANEXO B: Matriz o tabla de correlación entre elementos de diseño y tipos de conciencia

Tomado de (Antunes, *et al.*, 2014)

Elementos de diseño	Tipos de conciencia					
	Colaboración	Ubicación	Contexto	Social	Espacio de trabajo	Situación
Disponibilidad	32,50	6,40		4,00	4,50	
Modo de comunicación	8,10	1,80		1,30	1,00	
Conectividad de red	8,10		0,80	0,80	0,70	
Entrega de mensajes	8,10		1,10	0,80	0,70	
Retrasos de mensajes	8,10					
Ubicaciones cartesianas		5,70	0,60	0,70		
Ubicaciones topológicas		5,70	0,70			0,90
Distancias		5,70	1,10	1,10		
Orientaciones		5,70	0,60	0,70		
Range de atención		5,70	1,10	1,20		0,90
Modalidad de ubicación		9,20	1,10	2,20	1,10	2,00
Nivel de movilidad	1,60	9,20	1,20			1,30
Relación con otros dispositivos	2,40	9,50	1,40			
Restricciones físicas		7,10	1,70			2,30
Lugares físicos		7,1	1,5			
Topología física		7,1	1,6			
Atributos físicos		7,10	1,6			0,90
Lugares virtuales			5,3	2,2	1,4	
Topología virtual		1,1	5,3	2	1,4	
Mapas de vista		1,6	5,3	1,3	2,6	1,70

Miradores		1,2	5,3	1	2,5	1,70
Objetos grupales	0,8		4,2	1,5	3,1	
Objetos públicos			4,2	1,2	2,9	
Atributos virtuales			4,2		0,7	
Relaciones virtuales		1,1	4,2		0,9	
Restricciones virtuales			4,2		1,5	
Roles	3,20		4,2	13,4	1,7	2,60
Actividades	2,80		3,4	13,4	2,2	3,40
Privilegios			4,2	13,4	1,7	2,60
Historial de grupo			2,4	13,4	1,4	1,90
Quién	1,10		1,2	1,2	3,7	0,90
Qué	0,80		1,5		3,7	1,40
Dónde	0,90	2	1,1		3,7	1,40
Cuándo	1,10		1,2		3,7	1,40
Cómo			0,9		3,7	1,60
Historial de tareas	1,2		0,6	0,6	3,7	0,90
Retroalimentación	1		0,5	1,5	4,5	1,20
Alimentación	1		0,5	1,6	4,5	1,50
Retroalimentación del canal			0,7		4,5	0,90
Señales por mirada			1,1	5,4		
Pistas de voz			1,1	5,4		
Actividades paralelas	1,2		0,9	0,8	5,6	2,10
Actividades coordinadas	2,2		1,2	1,2	5,6	2,30
Actividades mutuamente ajustadas	2,2		1,5		5,6	2,10
Control de acceso	1,2		5,3		1,8	
Eventos	0,8		0,8		1,3	5,00

Acciones	0,8		0,6		1,3	5,00
Recursos			0,50		1,70	5,00
Elementos críticos	0,80		0,70		0,80	5,00
Significados	0,70		0,70	0,60	0,70	5,00
Escenarios futuros					0,70	5,00
Individual	1,40		1,60		1,70	12,00
Distribuido	2,70		1,60	1,60	3,00	10,00
Colaborativo	3		1,2	3,3	2,6	10,00