ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS

INNOVACIÓN PARA LA EXCELENCIA

Desarrollo de Software Aplicado a la Interculturalidad - 23398

Autores:

Luis Burbano

David Cantuña

Mateo Román

Sebastián Torres

Departamento de Ciencias de la Computación

Universidad de las Fuerzas Armadas - ESPE

Fecha de Entrega: 2025/06/05

Índice de tablas

Índice de Figuras

1. Introducción general sobre la importancia de la documentación y la trazabilidad de requisitos en proyectos de DSG.

La documentación y la trazabilidad de requisitos son fundamentales en el Desarrollo de Software Global (DSG), donde equipos distribuidos geográficamente enfrentan desafíos como diferencias culturales, horarias y de comunicación. Una documentación clara y estructurada garantiza que todos los involucrados comprendan los objetivos del proyecto, reduciendo malentendidos y errores. La trazabilidad, por su parte, permite establecer el ciclo de vida de los requisitos, desde su definición hasta su implementación, asegurando que cada funcionalidad cumpla con las expectativas del cliente y los estándares de calidad. Estas prácticas son esenciales para mantener la coherencia en entornos distribuidos, mejorar la colaboración y garantizar el cumplimiento de los plazos y presupuestos (Sommerville2016).

2. Desarrollo del contenido, abordando los siguientes subtemas:

a) Factores de estimación: Defina qué son y explique cómo afectan la planificación en entornos globales.

Los factores de estimación son elementos que influyen en la planificación y ejecución de proyectos de DSG, como el tamaño del software, la complejidad del proyecto, la experiencia del equipo y las restricciones de tiempo y recursos. En entornos globales, estos factores se ven afectados por desafíos adicionales, como la coordinación entre equipos en diferentes zonas horarias, barreras lingüísticas y diferencias en metodologías de trabajo. Por ejemplo, la estimación del esfuerzo puede variar debido a la necesidad de reuniones virtuales frecuentes o la adaptación a regulaciones locales. Modelos como COCOMO II ayudan a cuantificar estos factores, permitiendo una planificación más precisa. Herramientas como JIRA o Trello facilitan la trazabilidad y el seguimiento de requisitos, mejorando la gestión en contextos distribuidos (Boehm2000). Una estimación adecuada reduce riesgos y optimiza la calidad del software entregado.

b) Modelos de estimación: Compare al menos dos modelos conocidos (por ejemplo, COCOMO, Puntos de función, Use Case Points).

Los modelos de estimación permiten convertir las características técnicas y funcionales de un sistema en métricas que pueden ser cuantificables, esto para poder planificar tiempo, esfuerzo, personal y costo en los proyectos. A continuación se comparan dos modelos que permiten esto:

COCOMO (Constructive Cost Model)

COCOMO es un modelo que se basa en una estimación del tamaño del proyecto en las miles de líneas de código que tiene (KLOC). Para esto se usan fórmulas matemáticas dependiendo del proyecto (orgánico, semi ensamblado o empotrado)

Puntos de función

Los Puntos de Función (PF) miden el tamaño funcional del software, es decir, lo que el proyecto o sistema realiza para el usuario, evaluando la cantidad y complejidad de las entradas externas, salidas externas, consultas, archivos lógicos internos e interfaces externas.

c) Herramientas de estimación: Investigue e identifique al menos dos herramientas de software utilizadas para la estimación de requisitos en DSG (por ejemplo: JIRA, CostPerform, Estimaster, etc.).

Las herramientas de estimación en el desarrollo de software nos permiten automatizar, visualizar y centralizar el proceso de planificación, asignando recursos, esfuerzos y monitoreando el avance que tiene el proyecto, esto es muy útil, especialmente en entornos de desarrollo de software global, donde los equipos de trabajo necesitan una mejor colaboración, precisión en la asignación de tareas y trazabilidad en cada etapa del proyecto. Dos de las herramientas más utilizadas son:

Jira: Plataforma de gestión de proyectos, útil cuando se aplican metodologías ágiles, con la utilización de plugins como Advanced Roadmaps o Big Picture se pueden realizar estimación con tareas y cronogramas además de tener una buena integración con herramientas de desarrollo como Git, siendo muy fácil de usar aunque no incluye modelos formales como COCOMO por defecto

CostPerfom: Herramienta especializada para estimar costos y el esfuerzo en proyectos de software, se basa en modelos formales como COCOMO y Puntos de Función, siendo una gran ventaja para estimaciones detalladas y análisis de costos para equipos ubicados en diferentes regiones aunque se debe tener un mayor conocimiento técnico para poder iniciar.

3. Aplicación práctica

Definición de la ERS

1. El sistema debe analizar las calificaciones actuales y las estadísticas de uso, proporcionando retroalimentación personalizada al estudiante y estadísticas detalladas al profesor.
2. El sistema debe responder a las consultas de los estudiantes con información precisa y relevante, basada en el contenido del curso en cualquier contexto dentro de Moodle.

3. El sistema debe analizar las respuestas de los estudiantes en los cuestionarios y ofrecer retroalimentación personalizada, explicando errores y proporcionando orientación para mejorar si la calificación es baja, o un mensaje de motivación si la calificación es alta.
4. El sistema debe proporcionar recomendaciones sobre cómo realizar las tareas correctamente y ofrecer retroalimentación adicional según el estado y las calificaciones de las tareas.
5. El sistema debe permitir al docente cargar un sílabo estructurado en formato JSON o XML, validando automáticamente el formato y la estructura del archivo antes de almacenarlo.
6. El sistema debe comparar automáticamente los temas, objetivos y actividades planificadas en el sílabo con las actividades registradas en el curso de Moodle
7. El sistema debe enviar notificaciones automáticas al docente cuando se detecten incumplimientos en el sílabo

Tabla 1

Tabla de Puntos de Función (PF)

Tipo de Componente	Descripción	Cantidad	Complejidad Estimada	PF Unidad	Total PF
Entradas Externas (EE)	Carga de sílabo (JSON/XML), ingreso de consultas de estudiantes	2	Media	4	8
Salidas Externas (SE)	Retroalimentación a estudiantes (cuestionarios, tareas), estadísticas a profesores, notificaciones de incumplimiento, resultados de cumplimiento	4	Media	5	20
Consultas Externas (CE)	Consulta de contenido del curso por estudiantes, uso de APIs de Moodle para notas, cuestionarios e info del curso	2	Media	4	8
Archivos Lógicos Internos (ALI)	Base de datos: usuarios, mensajes (interacciones del tutor), sílabos, actividades/curso	4	Media	10	40

Interfaces Externas (IE)	Integración con API para base de datos (usuarios, mensajes), servicios internos de Moodle	2	Media	7	14
Total Puntos de Función (PF)					**90**

Nota. Tamaño funcional del proyecto utilizando Puntos de Función.

Cálculo de KLOC

El desarrollo será en JavaScript con un promedio de 47 LOC promedio por punto de función (Software Project Estimation, 2022).

$$KLOC \ = \ (90 \ * \ 47) \ / \ 1000 \ = 4,23 \ KLOC$$

Clasificación de Proyectos y Coeficientes

El proyecto se clasifica como orgánico porque opera en un entorno estable dentro de una plataforma conocida, sus requisitos están claramente definidos y se asume que el equipo es experimentado y cohesivo. Por ello, aplican los coeficientes COCOMO orgánicos:

$$a = 2.4,\ b = 1.05,\ c = 2.5,\ d = 0.38$$

Fórmulas del Modelo COCOMO

Esfuerzo (E)

- **Fórmula**: E = a × (KLOC)^b
    - **Descripción**: E = esfuerzo total estimado en persona-mes
    - **KLOC**: 4.23 (miles de líneas de código estimadas del proyecto)
    - **a, b**: Coeficientes para proyecto orgánico (a=2.4, b=1.05)
- **Cálculo**:
    - E = 2.4 × (4.23)^1.05
    - E ≈ 2.4 × 4.543 ≈ 10.903 persona-mes
- **Significado**: Esto indica que una sola persona tardaría aproximadamente 11 meses en desarrollar el sistema ERS.

Tiempo de Desarrollo (T)

- **Fórmula**: T = c × (E)^d
    - **Descripción**: T = tiempo estimado en meses calendario

    - **E**: 10.903 (esfuerzo en persona-mes, calculado anteriormente)
    - **c, d**: Coeficientes para proyecto orgánico (c=2.5, d=0.38)
- **Cálculo**:
    - T = 2.5 × (10.903)^0.38
    - T ≈ 2.5 × 2.151 ≈ 5.378 meses
- **Significado**: El proyecto tomaría aproximadamente 5 meses calendario con un equipo trabajando en paralelo.

Personal Necesario (P)

- **Fórmula**: P = E / T
    - **Descripción**: P = número de personas necesarias
    - **E**: 10.903 (esfuerzo en persona-mes)
    - **T**: 5.378 (tiempo en meses)
- **Cálculo**:
    - P = 10.903 / 5.378 ≈ 2.028 personas
- **Significado**: Se necesitan aproximadamente 2 personas trabajando a tiempo completo para completar el proyecto en 5 meses.

Productividad (PROD)

- **Fórmula**: PROD = KLOC / E
    - **Descripción**: PROD = productividad en KLOC/persona-mes
    - **KLOC**: 4.23
    - **E**: 10.903 (esfuerzo total)
- **Cálculo**:
    - PROD = 4.23 / 10.903 ≈ 0.388 KLOC/persona-mes
- **Significado**: El equipo produciría en promedio 0.39 KLOC (388 líneas de código) por persona al mes, reflejando una eficiencia típica para un proyecto orgánico.

Costo Total (CT)

- **Fórmula**: CT = E × salario promedio mensual
    - **Descripción**: CT = costo total del proyecto en dólares
    - **E**: 10.903 (esfuerzo en persona-mes)
    - **salario_promedio**: $150 USD/mes (asumido) (pasante)
- **Cálculo**:
    - CT = 10.903 × 150 ≈ $1635.45 USD

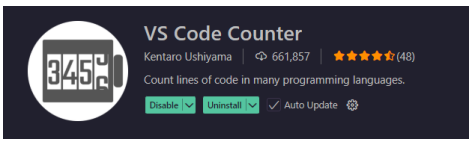
- **Significado**: El costo total estimado del proyecto sería aproximadamente $1635.45, basado en el esfuerzo y el salario promedio mensual.

CÁLCULO CON HERRAMIENTA DIGITAL:

Para el cálculo de KLOC del proyecto usamos la extensión VS Code Counter:

Figura 1.

Extensión de VS Code Counter



Nota. Captura de pantalla tomada por el autor desde Visual Studio Code en el marketplace (2025).

La herramienta en cuestión permite contar las líneas de código en archivos y directorios, ofreciendo soporte para múltiples lenguajes de programación y generando resúmenes detallados de las líneas contadas, incluyendo código, comentarios y líneas en blanco.

Figura 2.

Tabla comparativa de cada punto de fusión de 7 lenguajes.

Total : 26 files, 1900 codes, 165 comments, 275 blanks, all 2340 lines

Summary / Details / Diff Summary / Diff Details

Languages

language	files	code	comment	blank	total
JavaScript	6	1,172	96	153	1,421
Python	5	269	14	46	329
PostCSS	3	241	22	33	296
PHP	9	165	21	34	220
MS SQL	1	50	12	8	70
Markdown	1	2	0	1	3
pip requirements	1	1	0	0	1

Nota. Captura de pantalla tomada por el autor desde Github (2025).

Basado en el resumen obtenido, se analizaron varios lenguajes como JavaScript (1,172 líneas de código), Python (269), PostCSS (241), PHP (165), MS SQL (50), Markdown (2) y Requirements (1), con un total de 26 archivos y 1900 líneas de código en conjunto (KLOC). Este informe proporciona una visión clara de la distribución del código, útil para evaluar el tamaño y la estructura de un proyecto.

Conclusiones

- La documentación y trazabilidad de requisitos en DSG aseguran una comunicación efectiva entre equipos distribuidos, minimizando errores y garantizando que el software cumpla con los objetivos establecidos.
- La integración de modelos de estimación y herramientas de gestión en DSG mejora la coordinación entre equipos globales, asegurando una entrega eficiente y alineada con los requisitos del proyecto.
- Usar modelos de estimación como COCOMO o Puntos de Fusión permite una mejor estimación del esfuerzo que tendrá un proyecto, reduciendo el riesgo que el proyecto fracase al tener una mejor asignación de recursos, sobretodo en equipos distribuidos, entregando un mejor producto
- Usar herramientas como Jira y CostPerfom permite una mejor gestión de los proyectos, teniendo un mayor control sobre los recursos, tareas y tiempos, teniendo mayor facilidad para realizar seguimientos y tomar mejores decisiones.

Bibliografías:

Software Project Estimation. (2022, 6 diciembre). *Function Point Languages Table*. QSM Software Project Estimation. https://www.qsm.com/resources/function-point-languages-table

GeeksforGeeks. (2025, abril 11). *COCOMO Model – Software Engineering*. https://www.geeksforgeeks.org/software-engineering-cocomo-model/

GeeksforGeeks. (2025, abril 11). *Functional Point (FP) Analysis – Software Engineering*. https://www.geeksforgeeks.org/software-engineering-functional-point-fp-analysis/

International Function Point Users Group (IFPUG). (s.f.). *Function Point Analysis (FPA)*. https://ifpug.org/ifpug-standards/fpa

Atlassian. (2025). *Estimate a work item | Jira Cloud*.

https://support.atlassian.com/jira-software-cloud/docs/estimate-an-issue/

CostPerform. (2025). *Cost estimation tools*.

https://www.costperform.com/cost-estimation-tools/

Sommerville, I. (2016). Software Engineering (10th ed.). Pearson.
https://www.pearson.com/store/p/software-engineering/P100000260013

Boehm, B. W., Abts, C., Brown, A. W., Chulani, S., Clark, B. K., Horowitz, E., Madachy, R., Reifer, D. J., & Steece, B. (2000). Software Cost Estimation with COCOMO II. Prentice Hall.
https://www.pearson.com/store/p/software-costestimation-with-cocomo-ii/P100000259987