Cátedra Ingeniería de Software
Asignatura: Análisis y Diseño de Sistemas
Código 03301
Segundo Cuatrimestre 2025

Temas e instrumentos de evaluación por semana

Semana	Tema	Instrumento
1	Requerimientos de software y su especificación	
2	Análisis de software	
3	Análisis de software	
4	Análisis de software	Proyecto 1
5	Diseño de software	
6	Diseño de software	
7	Diseño de software	Proyecto 2
8	Arquitectura de software	
9	Arquitectura de software	
10	Arquitectura de software	Proyecto 3
11	Administración de la configuración	Evaluación 1
12	Administración de la configuración	

Lecturas obligatorias por semana

Se recomienda que el estudiante siga las lecturas en orden temático según el cuadro de arriba con el fin de que pueda ir dándose una idea de cómo entender el proceso de análisis y diseño de software. Sin embargo, se recomienda que el estudiante haya finalizado las lecturas en las semanas expuestas a continuación lo cual es fundamental para desarrollar los instrumentos expuestos en el cuadro.

Semanas obligatorias de lecturas: 1 y 2.

1. Libro Ingeniería del Software (1a. ed.) de Guillermo Pantaleo y Ludmila Rinaudo, capítulo 10 Relevamiento de requerimientos (páginas 144-166).

2. Libro Ingeniería del Software (1a. ed.) de Guillermo Pantaleo y Ludmila Rinaudo, capítulo 11 Análisis de requerimientos (páginas 168-177).

3. Libro Ingeniería del Software (1a. ed.) de Guillermo Pantaleo y Ludmila Rinaudo, capítulo 12 Pruebas a los requerimientos (páginas 180-197).

Semanas obligatorias de lecturas: 3 y 4.

1. Libro Ingeniería del Software (1a. ed.) de Guillermo Pantaleo y Ludmila Rinaudo, capítulo 7 El Lenguaje UML y sus modos de utilización (páginas 110-119).

2. Libro Ingeniería del Software (1a. ed.) de Guillermo Pantaleo y Ludmila Rinaudo, capítulo 8 El Lenguaje UML – Diagramas estáticos (páginas 122-131).

3. Libro Ingeniería del Software (1a. ed.) de Guillermo Pantaleo y Ludmila Rinaudo, capítulo 9 El Lenguaje UML – Diagramas dinámicos (páginas 134-141).

Semanas obligatorias de lecturas: 5, 6 y 7.

1. Libro Ingeniería del Software (1a. ed.) de Guillermo Pantaleo y Ludmila Rinaudo, capítulo 14 Diseño de Software (páginas 222-237).

2. Libro Ingeniería del Software (1a. ed.) de Guillermo Pantaleo y Ludmila Rinaudo, capítulo 15 Métricas de software (páginas 240-258).

3. Libro Ingeniería del Software (1a. ed.) de Guillermo Pantaleo y Ludmila Rinaudo, capítulo 16 Pruebas de software (páginas 260-284).

4. Libro Ingeniería del Software (1a. ed.) de Guillermo Pantaleo y Ludmila Rinaudo, capítulo 17 Proceso de pruebas de software (páginas 286-301).

Semanas obligatorias de lecturas: 8.

1. Libro Ingeniería del Software (1a. ed.) de Guillermo Pantaleo y Ludmila Rinaudo, capítulo 13 Arquitectura de software (páginas 200-219).

Semanas obligatorias de lecturas: 9 y 10.

1. Libro Ingeniería del Software (1a. ed.) de Guillermo Pantaleo y Ludmila Rinaudo, capítulo 4 Metodologías de Desarrollo (páginas 54-70).

2. Libro Ingeniería del Software (1a. ed.) de Guillermo Pantaleo y Ludmila Rinaudo, capítulo 5 Metodologías conducidas por los planes (páginas 72-90).

3. Libro Ingeniería del Software (1a. ed.) de Guillermo Pantaleo y Ludmila Rinaudo, capítulo 6 Metodologías ágiles (páginas 92-107).

Semanas obligatorias de lecturas: 11 y 12.

1. Libro Ingeniería del Software (1a. ed.) de Guillermo Pantaleo y Ludmila Rinaudo, capítulo 25 Estándares de Calidad de software (páginas 440-450).

2. Alcolea, C. D. (2019, diciembre 19). Qué es SOLID. Recuperado de https://openwebinars.net/blog/que-es-solid/

3. Martín, C. M. (2019, abril 4). Principios SOLID con ejemplos. Recuperado de https://enmilocalfunciona.io/principios-solid/

4. Leiva, A. (2016, enero 1). Principios SOLID. Recuperado 10 de abril de 2020, de https://architectcoders.com/wp-content/uploads/2019/08/principios-solid-devexperto.pdf

5. Clean Architecture. (2018, marzo 6). Recuperado de https://www2.deloitte.com/es/es/pages/technology/articles/clean-architecture.html

6. R. (2018, enero 31). Principios de una arquitectura limpia: mantenible y testeable. Recuperado de https://www.genbeta.com/desarrollo/principios-de-una-arquitectura-limpia-mantenible-y-testeable

7. El camino hacia una arquitectura software limpia - Hexagonal, Onion y Clean Architecture. (2016, diciembre 7). Recuperado de http://aitorrm.github.io/t%C3%A9cnicas%20y%20metodolog%C3%ADas/arquitectura_software_limpia/

8. Clean Coder Blog. (2012, agosto 13). Recuperado de https://blog.cleancoder.com/uncle-bob/2012/08/13/the-clean-architecture.html

9. Leiva, A. (2017, marzo 27). 12 ideas de la filosofía Clean que no pueden faltar en tu código. Recuperado 3 de mayo de 2020, de hHYPERLINK. https://www.genbeta.com/desarrollo/12-ideas-de-la-filosofia-clean-que-no-pueden-faltar-en-tu-codigo

Bibliografía

1. Libro Ingeniería del Software (1a. ed.) de Guillermo Pantaleo y Ludmila Rinaudo.

2. Ágil Es - Por Cris Rúa. (2018b, septiembre 12). User Story Mapping. Recuperado 3 de mayo de 2020, de https://www.youtube.com/watch?v=txOLx58sT5g

3. Ágil Es - Por Cris Rúa. (2018a, abril 15). Como empezar un proyecto - Agile Inception - Proyectos bien definidos. Recuperado 3 de mayo de 2020, de https://www.youtube.com/watch?v=Q-mLfDXqSKo

4. Alonso, A. (2019, enero 10). Técnicas Ágiles: Impact Mapping. Recuperado 3 de mayo de 2020, de https://adrianalonso.es/project-management/tecnicas-agiles-impact-mapping/

5. Big Learning. (2015, febrero 18). Descripción de Casos de Uso. Recuperado 2 de mayo de 2020, de https://www.youtube.com/watch?v=0NxXtTMvMfE

6. Campderrich Falgueras, B. (2013). Ingeniería del software. Editorial UOC. https://issuu.com/dariodelrincon/docs/ingenier__a_del_software_-_benet_ca

7. Cohn, M. (s. f.). User Stories and User Story Examples by Mike Cohn. Recuperado 13 de abril de 2020, recuperado de https://www.mountaingoatsoftware.com/agile/user-stories

8. Comsysto Reply GmbH. (2015, noviembre 17). User Story Mapping with Jeff Patton. Recuperado 3 de mayo de 2020, de https://www.youtube.com/watch?v=AorAgSrHjKM

9. Copyright IBM Corp. 1987, 2006. . (1987, enero 1). Artefacto: Visión. Recuperado 2 de mayo de 2020, de https://cgrw01.cgr.go.cr/rup/RUP.es/LargeProjects/core.base_rup/workproducts/rup_vision_2D6D6F1.html

10. Copyright IBM Corp. 1987, 2006. . (1987, enero 1). Plantilla Vision. Recuperado 2 de mayo de 2020, de https://cgrw01.cgr.go.cr/rup/RUP.es/LargeProjects/extend.formal_resources/guidances/templates/vision_3B89EF72.html

11. Dutoit, A. H. y H. Dutoit, A. (2002). Ingeniería de software orientado a objetos. Pearson Educación.). https://n9.cl/3adfo

12. Extending Impact Mapping to Gain Better Product Insights. (s. f.). Recuperado 3 de mayo de 2020, de https://www.scrum.org/resources/blog/extending-impact-mapping-gain-better-product-insights

13. Figuerola, N. (2016, agosto 3). Definiendo Requerimientos: Tradicional vs. Caso de Uso vs. Historias de Usuario. Recuperado 3 de mayo de 2020, de https://articulospm.wordpress.com/2012/04/16/definiendo-requerimientos-tradicional-vs-caso-de-uso-vs-historias-de-usuario/

14. Grau, J. L. V. (2020, febrero 27). Cómo generar un Backlog de Producto: el mapa de historias de usuario. Recuperado 3 de mayo de 2020, de https://proagilist.es/blog/agilidad-y-gestion-agil/generar-backlog-producto-mapa-historias-usuario/

15. Ingjulian Jota. (2015, abril 27). Documentación casos de Uso. Recuperado 2 de mayo de 2020, de https://www.youtube.com/watch?v=Hi_Nv04tw6g

16. Historias de usuario –. (2018, enero 29). Recuperado de https://julibetancur.blog/category/historias-de-usuario/

17. Impact Mapping | The Agile Coach. (2019, enero 23). Recuperado 3 de mayo de 2020, de https://theagilecoach.co.nz/impact-mapping/

18. Ingjulian Jota. (2015, abril 27). Documentacion casos de Uso. Recuperado 2 de mayo de 2020, de https://www.youtube.com/watch?v=Hi_Nv04tw6g

19. Larman, C. (2003). UML y patrones: una introducción al análisis y diseño orientado a objetos y al proceso unificado (2a. ed.). Pearson Educación. https://n9.cl/n8tx0

20. Mago, E., & Harvey, G. (s. f.). El Papel de la Arquitectura de Software en Scrum. Recuperado 3 de mayo de 2020, de https://sg.com.mx/revista/30/el-papel-la-arquitectura-software-scrum

21. Mapeo de Historias de usuario (USM) e Inception Directo al punto. (2019, febrero 22). Recuperado 3 de mayo de 2020, de https://www.caroli.org/es/mapeo-de-historias-de-usuario-usm-y-inception-directo-al-punto/

22. Menzinsky, A. (s. f.). ¿Las historias de usuario son casos de uso? Recuperado 3 de mayo de 2020, de

http://scrum.menzinsky.com/2016/08/las-historias-de-usuario-son-casos-de.html

23. Monte Galiano, J. (2016). Implantar scrum con éxito. Editorial UOC. https://elibro-net.cidreb.uned.ac.cr/es/ereader/uned/58575?page=1

24. ¿Quién escribe las Historias de Usuario? (2019, marzo 17). Recuperado de https://comunidad.iebschool.com/guadahidalgo/2019/03/17/quien-escribe-las-historias-de-usuario/

25. Rivadeneira, S., Vilanova, G., Miranda, M., & Cruz, D. (2013). *El modelado de requerimientos en las metodologías ágiles*. Recuperado de http://sedici.unlp.edu.ar/bitstream/handle/10915/27196/Documento_completo.pdf?sequence=1&isAllowed=y

26. Rodríguez, M. A. (2019, junio 19). Arquitectura Software y Metodologías Ágiles. ¿Realmente son compatibles? Recuperado 3 de mayo de 2020, de https://www.adictosaltrabajo.com/2019/06/06/arquitectura-software-y-metodologias-agiles-realmente-son-compatibles/

27. Sabadí, P. (s. f.). Historias de Usuario, trucos y consejos (lean+scrum) - Scrumízate. Recuperado 3 de mayo de 2020, de http://scrumizate.com/post/69/historias-de-usuario-trucos-y-consejos

28. Sanders, M. (2019, diciembre 4). Top 5 takeaways from User Story Mapping - Mal Sanders. Recuperado 3 de mayo de 2020, de https://medium.com/@mal.sanders/top-5-takeaways-from-user-story-mapping-by-jeff-patton-f8c80cf73750

29. Senn, J. A. (1992). Análisis y diseño de sistemas de información. McGraw-Hill Interamericana. https://n9.cl/8phv5

30. S. (2018, abril 23). Resumen User Stories Applied de Mike Cohn |. Recuperado de https://samuelcasanova.com/2017/06/resumen-user-stories-applied/

31. Sobczak, H. (2020, marzo 4). Historias de Usuarios - Henryk Sobczak. Recuperado de https://medium.com/@henryksobczak/historias-de-usuarios-fcbd6dba29e8

32. Sommerville, I. (2011). Ingeniería de software (9a. ed.). Pearson Educación. https://n9.cl/p44t1

33. Valdez, V. (2019, agosto 6). Técnicas efectivas para la toma de requerimientos. Recuperado 2 de mayo de 2020, de https://www.northware.mx/2012/01/15/tecnicas_efectivas_toma_requerimientos/

34. Martín, M. J. (2019, septiembre 6). SOLID: los 5 principios que te ayudarán a desarrollar software de calidad | Consultoría y Servicios IT para empresas. Recuperado de https://profile.es/blog/principios-solid-desarrollo-software-calidad/

35. Alcolea, C. D. (2019, diciembre 19). Qué es SOLID. Recuperado de https://openwebinars.net/blog/que-es-solid/

36. MartÍN, C. M. (2019, abril 4). Principios SOLID con ejemplos. Recuperado de https://enmilocalfunciona.io/principios-solid/

37. Leiva, A. (2016, enero 1). Principios SOLID. Recuperado 10 de abril de 2020, de https://architectcoders.com/wp-content/uploads/2019/08/principios-solid-devexperto.pdf

38. Clean Architecture. (2018, marzo 6). Recuperado de https://www2.deloitte.com/es/es/pages/technology/articles/clean-architecture.html

39. R. (2018, enero 31). Principios de una arquitectura limpia: mantenible y testeable. Recuperado de https://www.genbeta.com/desarrollo/principios-de-una-arquitectura-limpia-mantenible-y-testeable

40. El camino hacia una arquitectura software limpia - Hexagonal, Onion y Clean Architecture. (2016, diciembre 7). Recuperado de http://aitorrm.github.io/t%C3%A9cnicas%20y%20metodolog%C3%ADas/arquitectura_software_limpia/

41. Clean Coder Blog. (2012, agosto 13). Recuperado de https://blog.cleancoder.com/uncle-bob/2012/08/13/the-clean-architecture.html

42. Leiva, A. (2017, marzo 27). 12 ideas de la filosofía Clean que no pueden faltar en tu código. Recuperado 3 de mayo de 2020, de https://www.genbeta.com/desarrollo/12-ideas-de-la-filosofia-clean-que-no-pueden-faltar-en-tu-codigo