

SÍLABO

Construcción de Software

Código	ASUC00947		Carácter	Obligatorio
Prerrequisito	Diseño de Software / Programación Orientada a Objetos			
Créditos	5			
Horas	Teóricas	4	Prácticas	2
Año académico	2024			

I. Introducción

Construcción de Software es una asignatura obligatoria de especialidad que se ubica en el séptimo periodo de la Escuela Académico Profesional de Ingeniería de Sistemas e Informática. Tiene como requisito haber aprobado las asignaturas de Diseño de Software y Programación Orientada a Objetos. Desarrolla en un nivel intermedio las competencias transversales Gestión de Proyectos y Experimentación; y las competencias específicas de Diseño y Desarrollo de Soluciones, y Uso de Herramientas Modernas. La relevancia de la asignatura reside en utilizar los métodos, herramientas y procedimientos para la construcción eficiente de software que satisfaga las necesidades del cliente.

Los contenidos generales que la asignatura desarrolla son los siguientes: fundamentos de la construcción de software, gestión de la construcción de software, prácticas de código completo y código limpio, tecnologías para la construcción de software, herramientas para la construcción de software.

II. Resultado de aprendizaje de la asignatura

Al finalizar la asignatura, el estudiante será capaz de aplicar los principios fundamentales de la construcción de software empleando metodologías y herramientas de construcción de software.

III. Organización de los aprendizajes

Unidad 1 Fundamentos de la construcción de software		Duración en horas	24
Resultado de aprendizaje de la unidad:	Al finalizar la unidad, el estudiante será capaz de aplicar los fundamentos de la construcción de software.		
Ejes temáticos:	1. Minimización de la complejidad 2. Anticipación a los cambios 3. Construcción para verificación 4. Reutilización 5. Estándares en la construcción		

Unidad 2 Gestión de la construcción de software		Duración en horas	24
Resultado de aprendizaje de la unidad:	Al finalizar la unidad, el estudiante será capaz de aplicar los procesos de gestión de la construcción del software.		
Ejes temáticos:	1. Modelos de construcción 2. Plan de construcción 3. Métricas de la construcción		

Unidad 3 Prácticas de código completo y código limpio		Duración en horas	24
Resultado de aprendizaje de la unidad:	Al finalizar la unidad, el estudiante será capaz de aplicar buenas prácticas de código completo y código limpio en la construcción del software.		
Ejes temáticos:	1. Diseño de la construcción 2. Lenguajes de construcción 3. Codificación 4. Pruebas de construcción 5. Construcción para reutilización 6. Construcción con reutilización 7. Calidad de construcción 8. Integración		

Unidad 4 Tecnologías y Herramientas para la construcción de software		Duración en horas	24
Resultado de aprendizaje de la unidad:	Al finalizar la unidad, el estudiante será capaz de aplicar las tecnologías y herramientas orientadas a la construcción del software.		
Ejes temáticos:	<ol style="list-style-type: none"> 1. Diseño y uso API 2. Problemas de tiempo de ejecución orientado a objetos 3. Parametrización y genéricos 4. Afirmaciones, diseño por contrato y programación defensiva 5. Control de errores, control de excepciones, y la tolerancia a fallos 6. Modelos ejecutables 7. Técnicas de construcción basadas en estados y tablas 8. Configuración de tiempo de ejecución y la internacionalización 9. Procesamiento de entrada basada en gramática 10. Middleware 11. Métodos de construcción para software distribuido 12. Construcción de sistemas heterogéneos 13. Entornos de desarrollo 14. Constructores GUI 15. Herramientas de prueba de unidad 		

IV. Metodología

Modalidad Presencial:

Método expositivo/lección magistral, estudio de casos, resolución de problemas y ejercicios, aprendizaje orientado a proyectos y aprendizaje colaborativo. Enseñanza programada, enseñanza modular, aprendizaje autodirigido, investigación y tutoría académica. Técnicas para identificar necesidades, lista de chequeo, preguntas por resolver, listado de expectativas, prueba de entrada, exposición, simulaciones, aprendizaje basado en casos reales o propuestos, y diálogo/preguntas.

Modalidad Semipresencial Blended, A Distancia

Método expositivo, estudio de casos, resolución de problemas y ejercicios, aprendizaje orientado a proyectos y aprendizaje colaborativo. Enseñanza programada, enseñanza modular, aprendizaje autodirigido, investigación y tutoría académica. Técnicas para identificar necesidades, lista de chequeo, preguntas por resolver, listado de expectativas, prueba de entrada, exposición, simulaciones, aprendizaje basado en casos reales o propuestos, gamificación y diálogo/preguntas.

**V. Evaluación
Modalidad Presencial**

Rubros	Unidad por evaluar	Fecha	Entregable/Instrumento	Peso parcial	Peso Total
Evaluación de entrada	Prerrequisito	Primera sesión	- Evaluación individual teórica / Prueba objetiva	0%	
Consolidado 1 C1	1	Semana 1 - 4	- Evaluación individual teórico-práctica / Prueba de desarrollo	40%	20%
	2	Semana 5 - 7	- Ejercicios grupales de análisis de casos desarrollados en clase / Rúbrica de evaluación	60%	
Evaluación parcial EP	1 y 2	Semana 8	- Evaluación individual teórico-práctica / Prueba de desarrollo	20%	
Consolidado 2 C2	3	Semana 9 - 12	- Evaluación individual teórico-práctica / Prueba de desarrollo	40%	20%
	4	Semana 13 - 15	- Ejercicios grupales de análisis de casos desarrollados en clase / Rúbrica de evaluación	60%	
Evaluación final EF	Todas las unidades	Semana 16	- Desarrollo individual de análisis de casos en plataforma virtual / Rúbrica de evaluación	40%	
Evaluación sustitutoria *	Todas las unidades	Fecha posterior a la evaluación final	- Aplica		

* Reemplaza la nota más baja obtenida en los rubros anteriores.

Modalidad Semipresencial Blended

Rubros	Unidad por evaluar	Fecha	Entregable/Instrumento	Peso parcial	Peso Total
Evaluación de entrada	Prerrequisito	Primera sesión	- Evaluación individual teórica / Prueba objetiva	0 %	
Consolidado 1 C1	1	Semana 1 - 3	Evaluación individual teórico-práctica / Prueba de desarrollo Ejercicios grupales de análisis de casos desarrollados en clase / Rúbrica de evaluación	85 %	20 %
			Actividades virtuales	15 %	
Evaluación parcial EP	1 y 2	Semana 4	- Evaluación individual teórico-práctica / Prueba de desarrollo	20 %	
Consolidado 2 C2	3	Semana 5 - 7	- Evaluación individual teórico-práctica / Prueba de desarrollo - Ejercicios grupales de análisis de casos desarrollados en clase / Rúbrica de evaluación	85 %	20 %
			- Actividades virtuales	15 %	

Evaluación final EF	Todas las unidades	Semana 8	- Desarrollo individual de análisis de casos en plataforma virtual / Rúbrica de evaluación.	40 %
Evaluación sustitutoria *	Todas las unidades	Fecha posterior a la evaluación final	- Aplica	

* Reemplaza la nota más baja obtenida en los rubros anteriores.

Modalidad A Distancia

Rubros	Unidad por evaluar	Fecha	Entregable/Instrumento	Peso
Evaluación de entrada	Prerrequisito	Primera sesión	- Evaluación individual teórica / Prueba objetiva	0 %
Consolidado 1 C1	1	Semana 2	- Evaluación individual teórico-práctica / Prueba de desarrollo	20 %
Evaluación parcial EP	1 y 2	Semana 4	- Desarrollo individual de análisis de casos en plataforma virtual / Rúbrica de evaluación	20 %
Consolidado 2 C2	3	Semana 6	- Evaluación individual teórico-práctica / Prueba de desarrollo	20 %
Evaluación final EF	Todas las unidades	Semana 8	- Desarrollo individual de análisis de casos en plataforma virtual / Rúbrica de evaluación	40 %
Evaluación sustitutoria	Todas las unidades	Fecha posterior a la evaluación final	- Aplica	

* Reemplaza la nota más baja obtenida en los rubros anteriores.

Fórmula para obtener el promedio:

$$PF = C1 (20 \%) + EP (20 \%) + C2 (20 \%) + EF (40 \%)$$

VI. Bibliografía

Básica

Mcconnell, S. (2004). *Code complete: a practical handbook of software construction* (2ª ed.). Microsoft Press. <http://bit.ly/3r5hPiQ>

Complementaria:

Martin, C. (2009). *Clean code: a handbook of agile software craftsmanship*. Upper Saddle River, NJ, Prentice Hall.

Fowler, M., & Beck, K. (2019). *Refactoring: Improving the Design of Existing Code* (Second edition ed.). Boston: Addison-Wesley.

- Martin, C. (2011). *The Clean Coder: A Code of Conduct for Professional Programmers*. EE. UU.: Pearson Educación.
- Sommerville, I. (2011). *Software Engineering* (9.ª ed.). EEUU: Pearson Educación.

VII. Recursos digitales:

- Beohar, H., & Mousavi, M. R. (2016). *Input-output conformance testing for software product lines*. Journal of Logical and Algebraic Methods in Programming, 85(6), 1131–1153. doi:10.1016/j.jlamp.2016.09.007
- Castañeda, P., & Mauricio, D. (2020a). *New factors affecting productivity of the software factory*. International Journal of Information Technologies and Systems Approach, 13(1), 1–26. <https://doi.org/10.4018/IJITSA.2020010101>
- Castañeda, P., & Mauricio, D. (2020b). *A model based on data envelopment analysis for the measurement of productivity in the software factory*. International Journal of Information Technologies and Systems Approach, 13(2), 1–26. <https://doi.org/10.4018/IJITSA.2020070101>
- IEEE Computer Society (2014). *Software Engineering Body of Knowledge v3.0*. <https://www.computer.org/education/bodies-of-knowledge/software-engineering> [Consulta: 06 de setiembre de 2020]
- Foucault, M., Blanc, X., Jean-Rémy Falleri, & Storey, M. (2019). *Fostering good coding practices through individual feedback and gamification: An industrial case study*. Empirical Software Engineering, 24(6), 3731-3754. doi: <http://dx.doi.org/10.1007/s10664-019-09719-4>
- Lee, E. T. (1996). *Context-free parallel grammars and their applications to generating context-sensitive languages*. Kybernetes, 25(4), 131–140. doi:10.1108/03684929610118390
- Vargas, P. S. C., & Mauricio, D. (2018). *A review of literature about models and factors of productivity in the software factory*. In International Journal of Information Technologies and Systems Approach (Vol. 11, Issue 1, pp. 48–71). IGI Global. <https://doi.org/10.4018/IJITSA.2018010103>