2. Diseño

2.1 Diseño de procesos propuestos.
**2.1.1 Herramientas CASE para diseño.**
2.2 Diseño arquitectónico.
2.3 Diseño de datos.
2.4 Diseño de interfaz de usuario.

Las herramientas CASE (Computer-Aided Software Engineering) son aplicaciones diseñadas para apoyar y facilitar el desarrollo de la ingeniería automatizada. Estas herramientas abarcan una amplia gama de funcionalidades que incluyen modelado, diseño, análisis, documentación, pruebas y gestión de proyectos.

Proporcionando a los equipos de trabajo las capacidades necesarias para planificar, diseñar, gestionar y mantener sistemas con calidad.

Nota: La elección de la herramienta adecuada dependerá de las necesidades y requerimientos específicos del proyecto.

Desafíos en el uso de herramientas CASE:

- **Curva de aprendizaje**: Algunas herramientas pueden tener una curva de aprendizaje que impacta el costo/tiempo.
- **Integración con otros sistemas**: La integración de herramientas CASE con otras tecnologías.

Evolución de las herramientas CASE:

- Las herramientas CASE han evolucionado para adaptarse a las cambiantes metodologías para la generación de procesos automatizados.
- La tendencia es hacia herramientas colaborativas, basadas en la nube para facilitar el trabajo en equipo y la accesibilidad en cualquier lugar.

**Tipos de herramientas CASE:**

- **Modelado**: Permiten la creación de diagramas y modelos que representan visualmente el diseño de ingeniería de procesos, como diagramas de flujo, diagramas de clases, diagramas de casos de uso, entre otros diagramas. **Ejemplos Microsoft Visio, Lucidchart y Enterprise Architect.**
- **Análisis y diseño**: Ayudan en la planificación y el diseño de ingeniería de procesos, permitiendo definir la arquitectura, estructura y componentes utilizados en el proyecto. **Ejemplos Rational Rose, Sparx Systems, y Visual Paradigm.**
- **Gestión de requisitos**: Ayudan a capturar, rastrear y gestionar los requisitos a lo largo del ciclo de vida del proyecto. **Ejemplos IBM Engineering Requirements Management DOORS y Jama Connect.**
- **Generación de código**: Permiten generar automáticamente código a partir de modelos y diseños previamente definidos. E**jemplos Eclipse Modeling Framework (EMF) y CodeCharge Studio.**
- **Generación de pruebas**: Facilitan la planificación, ejecución y seguimiento de pruebas. **Ejemplos HP ALM (Application Lifecycle Management) y TestRail.**
- **Gestión de proyectos**: Ayudan en la planificación, seguimiento y gestión de proyectos. **Ejemplos Jira, Trello, ASANA y Microsoft Project.**

Ventajas de las herramientas CASE:

- **Productividad**: Automatizan tareas manuales.
- **Calidad**: Facilitan la detección temprana de errores y eventualidades orientadas al diseño.
- **Colaboración**: Permiten a los equipos trabajar de manera colaborativa y compartir información de manera más efectiva.
- **Documentación**: Facilitan la generación automática de documentación técnica y de usuario.
- **Gestión de cambios**: Ayudan a gestionar y mantener un registro de versiones.

Referencias


- BARDUS, I. & PRYVEZENTSEV, O.. (2023). ANALYSIS OF THE PROFESSIONAL ACTIVITY OF A SPECIALIST IN THE DEVELOPMENT OF USER INTERFACES. Scientific papers of Berdiansk State Pedagogical University Series Pedagogical sciences. 1. 199-209. 10.31494/2412-9208-2023-1-1-199-209.
- Lin, Liannan & Zheng, Zitao & Li, Ziqi. (2023). AI Interaction Design Driven Software Engineering: An Exploratory Experimental Teaching Method. 10.1007/978-981-99-2446-2_24.
- Erazo, Lenin & Suquisupa, Steveen & Bermeo, Alexandra & Cedillo, Priscila. (2023). Model-Driven Engineering Applied to User Interfaces. A Systematic Literature Review. 10.1007/978-3-031-24985-3_42.
- Khaddam, Iyad & Barakat, Hanaa & Vanderdonckt, Jean. (2016). Enactment of User Interface Development Methods in Software Life Cycles.
- Imam, Ayad & Alnsour, Ayman & Alhroob, Aysh. (2015). The Definition of Intelligent Computer Aided Software Engineering (I-CASE) Tools. Journal of Information Engineering and Applications. 5. 47-56.
- Arman, Nabil. (2013). Towards E-CASE Tools for Software Engineering. International Journal of Advanced Corporate Learning (iJAC). 6. 16-19. 10.3991/ijac.v6i1.2309.
- Kosavinta, Satakhun & Kanongchaiyos, Pizzanu & Jinuntuya, Pinyo. (2007). Integration of CAD Software with DSS for Engineering and Architectural Project Design. Computer-Aided Design & Applications. 4. 467-476. 10.1080/16864360.2007.10738566.
- Leventhal, Laura & Mynatt, Barbee. (2006). A scarce resource in undergraduate software engineering courses: User interface design materials. 10.1007/BFb0043599.


Referencias (continuación).


- Thompson, J. & Goh, Angela. (1993). CASE Tools in Software Engineering Education. 319-321. 10.1016/B978-0-444-81597-2.50039-5.
- Bode, Stephan & Fischer, Anja & Kühnhauser, Winfried & Riebisch, Matthias. (2009). Software Architectural Design Meets Security Engineering. Proceedings of the International Symposium and Workshop on Engineering of Computer Based Systems. 109-118. 10.1109/ECBS.2009.17.
- Capilla, Rafael & Ali Babar, Muhammad. (2008). On the Role of Architectural Design Decisions in Software Product Line Engineering. 5292. 10.1007/978-3-540-88030-1_18.
- Kinnula, Atte. (1999). Software process engineering in a multi-site environment: an architectural design of a software process engineering system.
- Six, H.-W & Voss, J.. (1992). A software engineering perspective to the design of a user interface framework. 128 - 134. 10.1109/CMPSAC.1992.217591.
- Bhowmick, Twinkle & Koner, Suraj & Saha, Biraj & Ghosh, Debosree & Pramanik, Bablu. (2023). Software Engineering: New Methodologies, Tools, and Best Practices in Software Development. International Journal of Innovative Research in Science,Engineering and Technology. 12. 10.15680/IJIRSET.2023.1208077.