What is the software development life cycle (SDLC)?

3 November 2026 · 4 min read · blogs.iictp.org/what-is-the-software-development-life-cycle

The software development life cycle is the sequence of activities a team goes through to take software from an idea to a running, maintained product: understand what is needed, design it, build it, test it, release it and look after it. Different teams arrange those activities differently, but nearly all of them cover the same ground.

TL;DR The short version

- The SDLC is the sequence that takes software from idea to running, maintained product: requirements, design, build, test, release and maintenance.
- Waterfall finishes each phase before the next, so user feedback arrives late; iterative models repeat the cycle in small steps.
- Testing runs throughout, the cycle does not end at release, and the right model depends on how well requirements are known and how costly mistakes are.

Read online, listen to it, or change the text size: https://blogs.iictp.org/what-is-the-software-development-life-cycle

What are the usual phases?

1. **Requirements:** find out what users and the business need.
2. **Design:** decide how the system will be structured.
3. **Implementation:** write the code.
4. **Testing:** check that it works and meets the need.
5. **Release:** put it in front of users.
6. **Maintenance:** fix, improve and eventually retire it.

What is the difference between waterfall and iterative models?

In a waterfall approach each phase is finished before the next begins, so feedback from users arrives late. In an iterative or agile approach the team repeats the whole cycle in short rounds, delivering something usable each time and changing course as it learns. Royce's 1970 paper, often cited for waterfall, itself warned that a single sequential pass is risky.

Where does testing fit?

Throughout, not only after coding. Requirements can be reviewed, designs can be challenged, and code can be tested as it is written. The earlier a mistake is found, the cheaper it is to fix, which is why modern teams run automated checks on every change.

Does the life cycle end at release?

No. Most of a product's life is spent in operation, where the team monitors it, fixes defects, answers support questions and adds features. Planning for maintenance from the start, with logs, tests and documentation, makes that long stretch easier.

Which model should a team choose?

Choose by how well the requirements are known and how costly a mistake is. Work with fixed, well-understood requirements and heavy regulation may suit more up-front planning. Work where needs are uncertain suits short iterations with frequent feedback. Many teams blend the two.

Related credential: Associate Technical Practitioner. Explore the exam facts and syllabus.

Sources and further reading

1. Royce, W. W. (1970). "Managing the Development of Large Software Systems." Proceedings of IEEE WESCON.
2. ISO/IEC/IEEE 12207. Systems and software engineering: software life cycle processes.
3. Sommerville, I. Software Engineering. Pearson.

Published by IICTP, an independent technical credential body, on 3 November 2026. The current version of this post is at https://blogs.iictp.org/what-is-the-software-development-life-cycle. More guides: blogs.iictp.org.