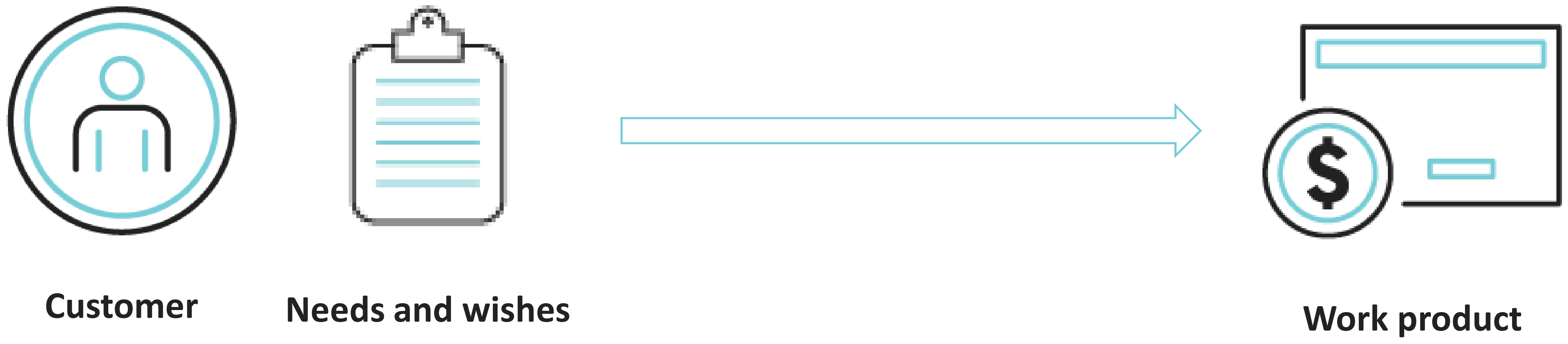
Software Development Life Cycle

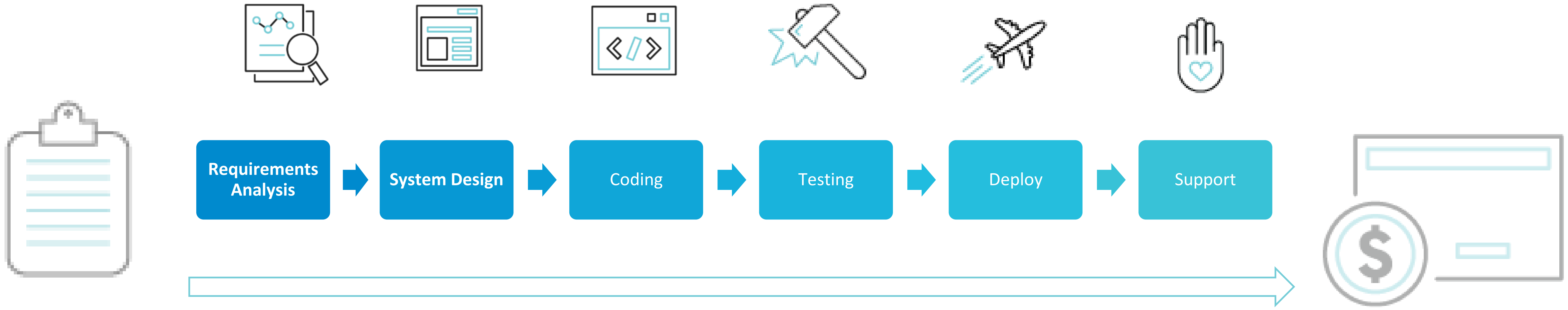
For External Courses

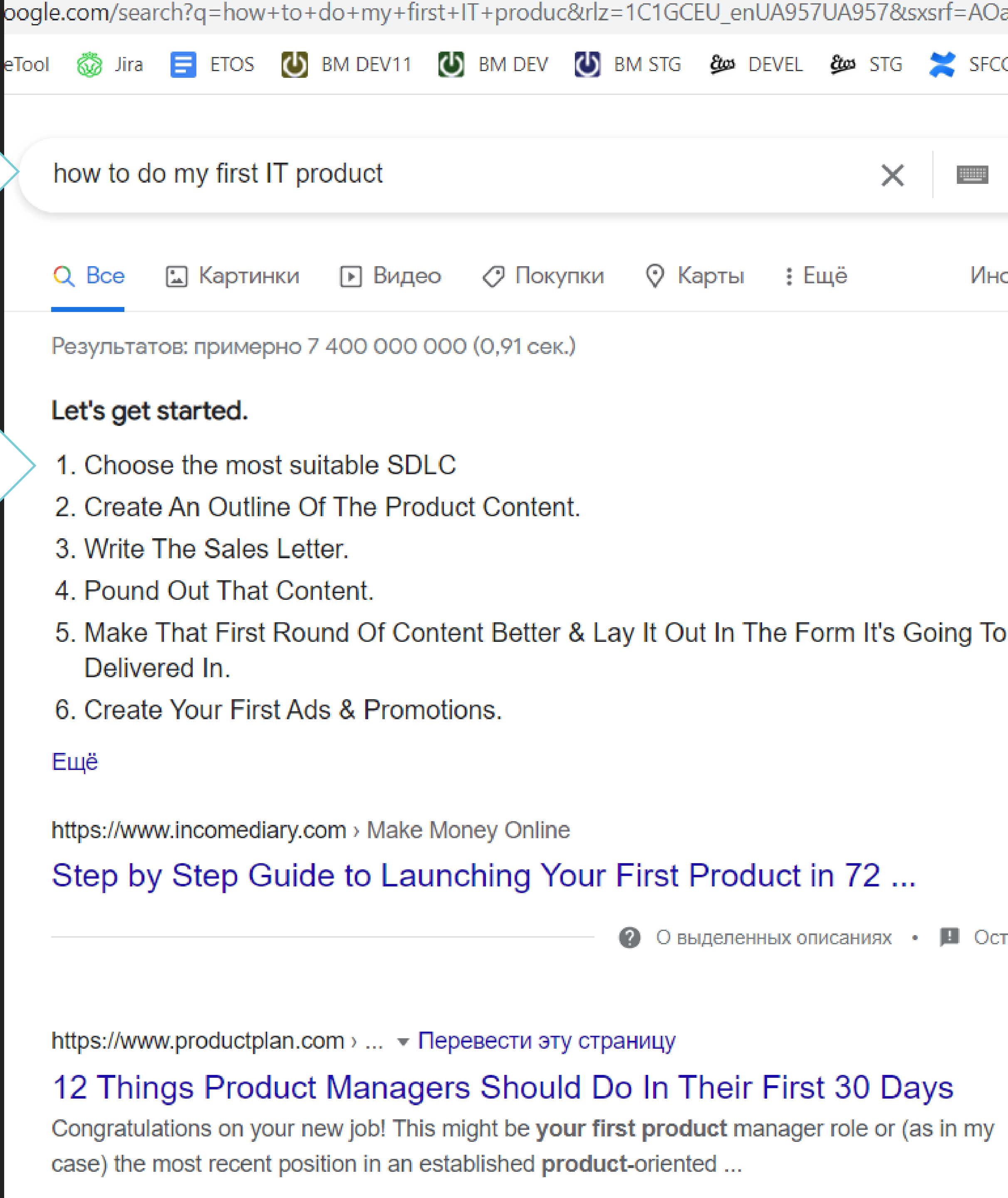
2022

Agenda

Name


Customer
Needs and wishes
Work product


Requirements Analysis
System Design
Coding
Testing
Deploy
Support


oogle.com/search?q=how+to+do+my+first+IT+produc&rlz=1C1GCEU_enUA957UA957&sxsrf=AOa

how to do my first IT product

Результатов: примерно 7 400 000 000 (0,91 сек.)

Let's get started.

1. Choose the most suitable SDLC
2. Create An Outline Of The Product Content.
3. Write The Sales Letter.
4. Pound Out That Content.
5. Make That First Round Of Content Better & Lay It Out In The Form It's Going To Delivered In.
6. Create Your First Ads & Promotions.

Ещё

https://www.incomediary.com › Make Money Online

Step by Step Guide to Launching Your First Product in 72 ...

https://www.productplan.com › ... ▾ Перевести эту страницу

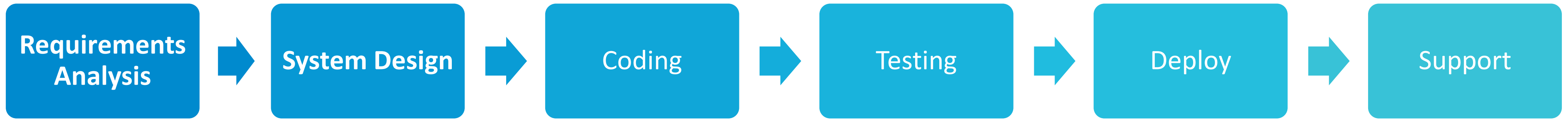
12 Things Product Managers Should Do In Their First 30 Days

Congratulations on your new job! This might be **your first product** manager role or (as in my case) the most recent position in an established **product**-oriented ...

Software Development Life Cycle

SDLC - the period of time that begins when a software product is conceived and ends when the software is no longer available for use. The software lifecycle typically includes a concept phase, requirements phase, design phase, implementation phase, test phase, installation and checkout phase, operation and maintenance phase, and sometimes, retirement phase. Note these phases may overlap or be performed iteratively.

Software Development Life Cycle



Why do we need to learn SDLC?

SDLC Models

Waterfall

V model

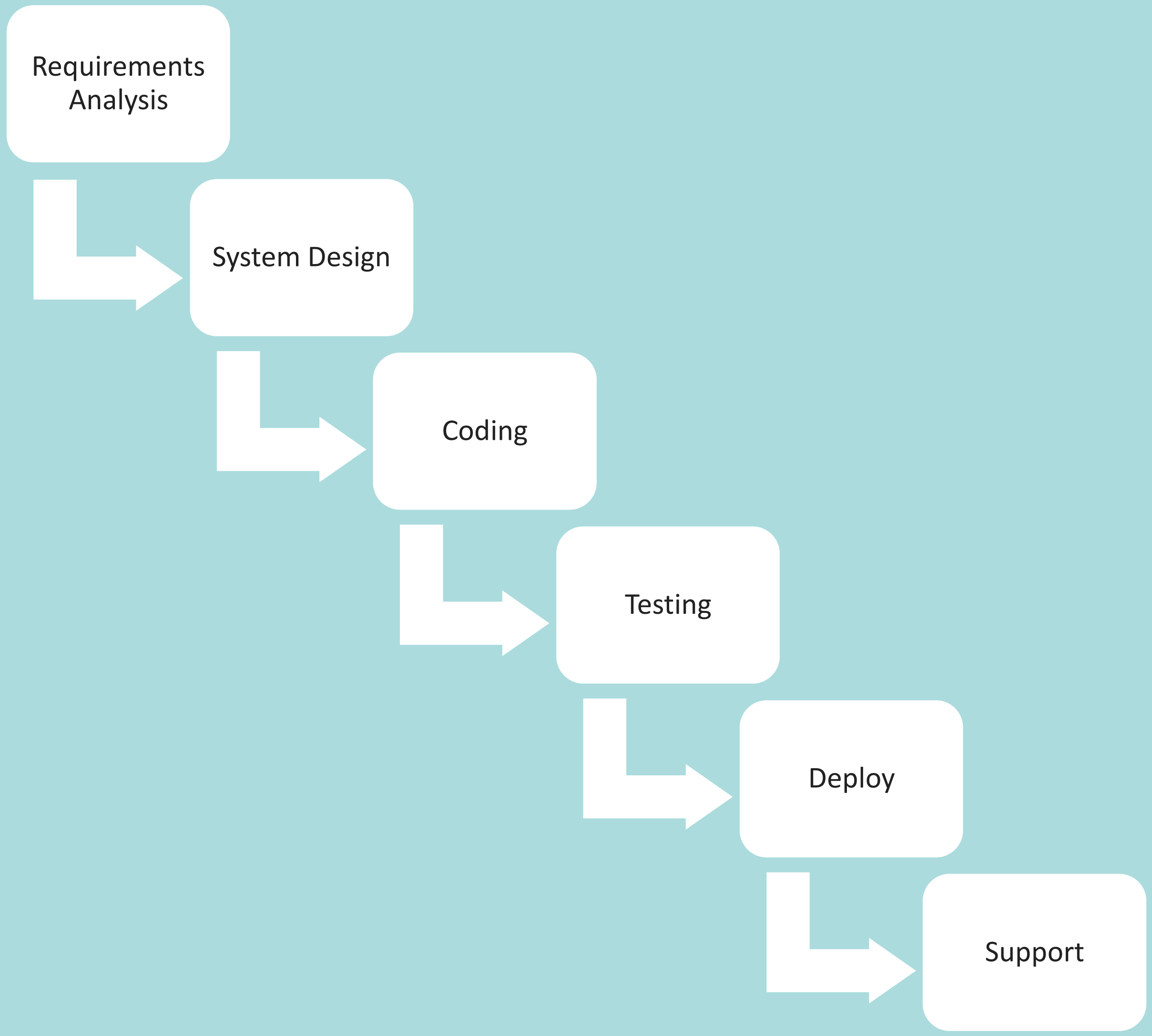
Iterative and Incremental

Agile

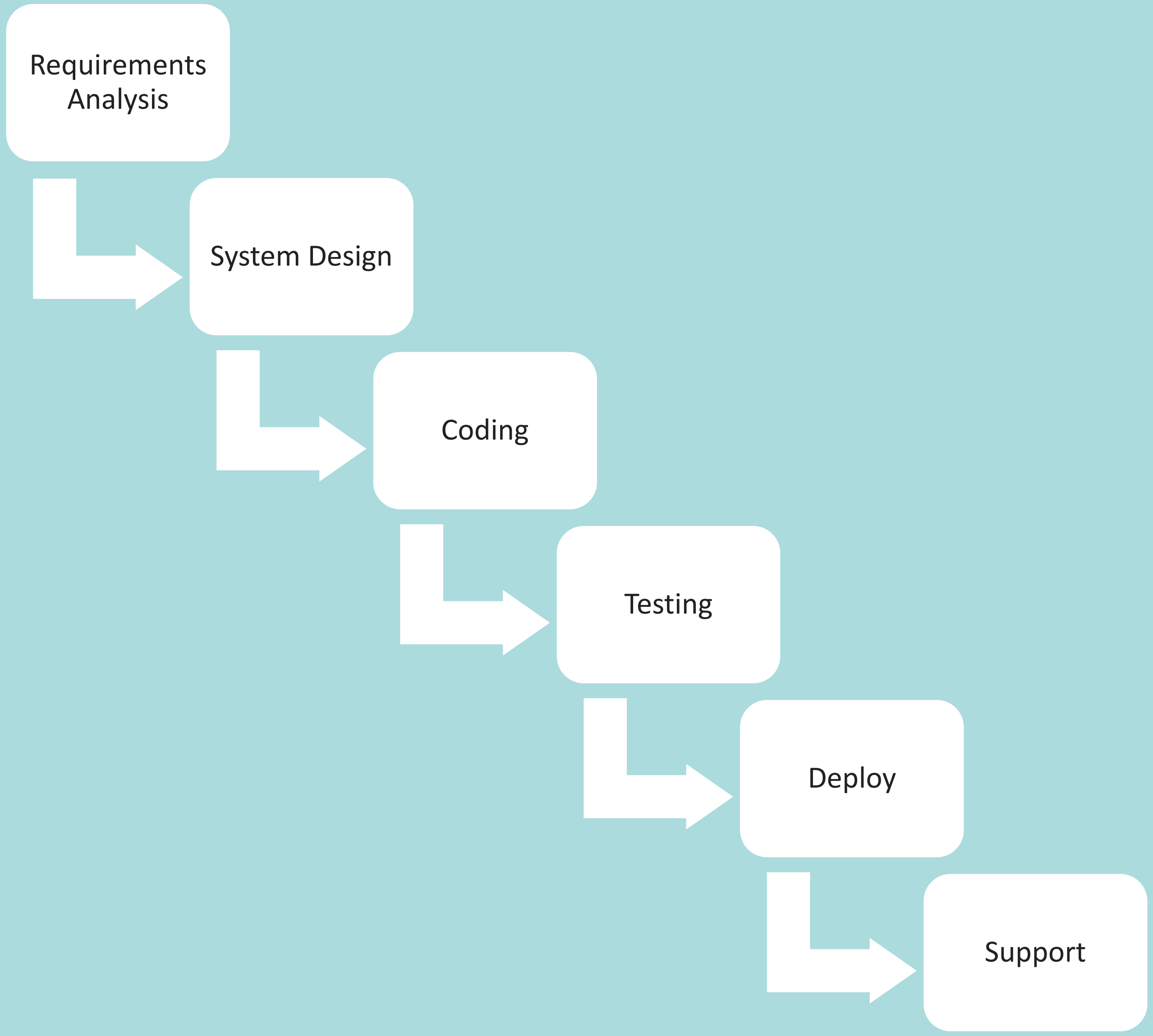
Waterfall

The Waterfall Model is a linear sequential flow. In which progress is seen as flowing steadily downwards (like a waterfall) through the phases of software implementation.

In a waterfall model, each phase must be completed fully before the next phase can begin. This type of model is basically used for the project which is small and there are no uncertain requirements. At the end of each phase, a review takes place to determine if the project is on the right path and whether or not to continue or discard the project.



Waterfall



All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification doc.

The requirement specifications from first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture.

The development stage is the part where developers actually write code and build the application according to the earlier design documents and outlined specifications.

Building software is not the end. Now it must be tested to make sure that there aren't any bugs and that the end-user experience will not negatively be affected at any point.

Once the functional and non functional testing is done, the product is deployed in the customer environment or released into the market.

There are some issues which come up in the client environment. To fix those issues patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

Waterfall

Model Application

Every software developed is different and requires a suitable SDLC approach to be followed based on the internal and external factors. Some situations where the use of Waterfall model is most appropriate are:

- Requirements are very well documented, clear and fixed
- Product definition is stable
- Technology is understood and is not dynamic
- There are no ambiguous requirements
- The project is short

Model Pros

- Simple and easy to understand and use.
- Easy to manage due to the rigidity of the model – each phase has specific deliverables and a review process.
- Phases are processed and completed one at a time.
- Works well for smaller projects where requirements are very well understood.
- Clearly defined stages.
- Easy to arrange tasks.
- Process and results are well documented.

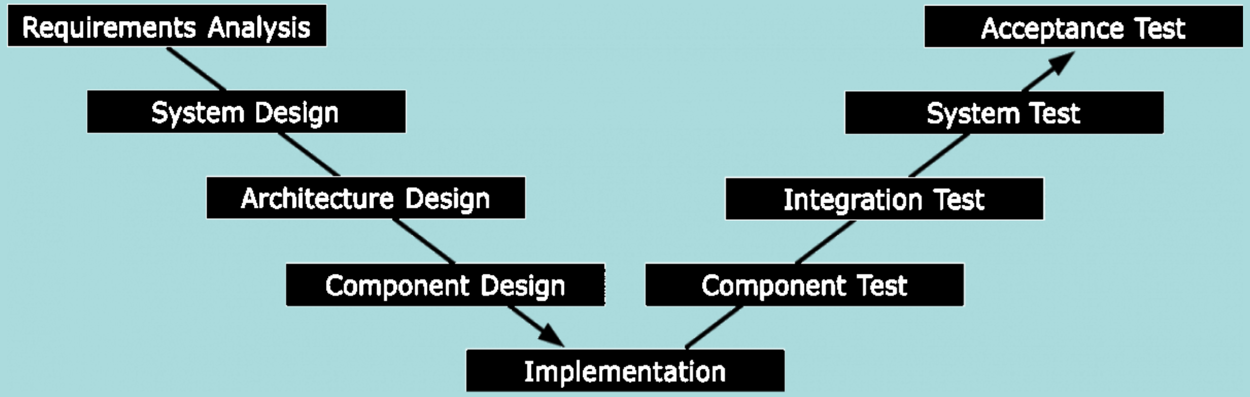
Model Cons

- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Poor model for long and ongoing projects.
- Not suitable for the projects where requirements are at a moderate to high risk of changing. So risk and uncertainty is high with this process model.
- It is difficult to measure progress within stages.
- Cannot accommodate changing requirements.

V- model

The V- model is SDLC model where execution of processes happens in a sequential manner in V-shape.

V- model is a framework to describe the software development lifecycle activities from requirements specification to maintenance. The V-model illustrates how testing activities can be integrated into each phase of the software development lifecycle



V- model

Model Application

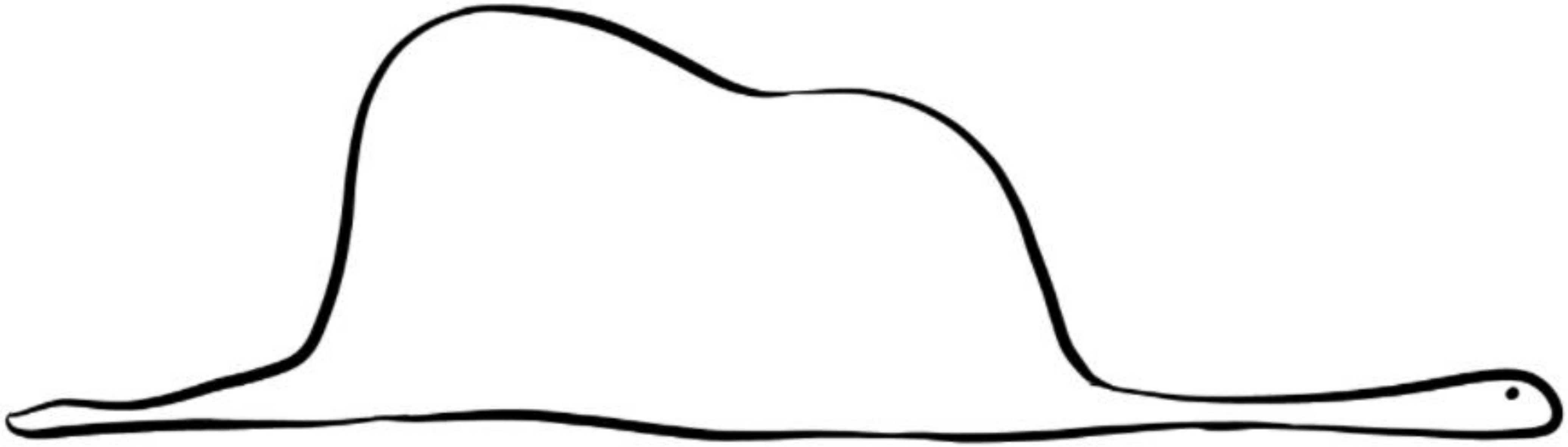
V- Model application is almost same as waterfall model, as both the models are of sequential type. Requirements have to be very clear before the project starts, because it is usually expensive to go back and make changes. This model (as and Waterfall) is used in the medical development field, as it is strictly disciplined domain.

Model Pros

- Simple and easy to understand and use.
- Easy to manage due to the rigidity of the model – each phase has specific deliverables and a review process.
- Phases are processed and completed one at a time.
- Works well for smaller projects where requirements are very well understood.
- Clearly defined stages.
- Well understood milestones.
- Easy to arrange tasks.
- Process and results are well documented.

Model Cons

- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Poor model for long and ongoing projects.
- Not suitable for the projects where requirements are at a moderate to high risk of changing. So risk and uncertainty is high with this process model.
- It is difficult to measure progress within stages.
- Cannot accommodate changing requirements.
- No working software is produced until late in the life cycle.

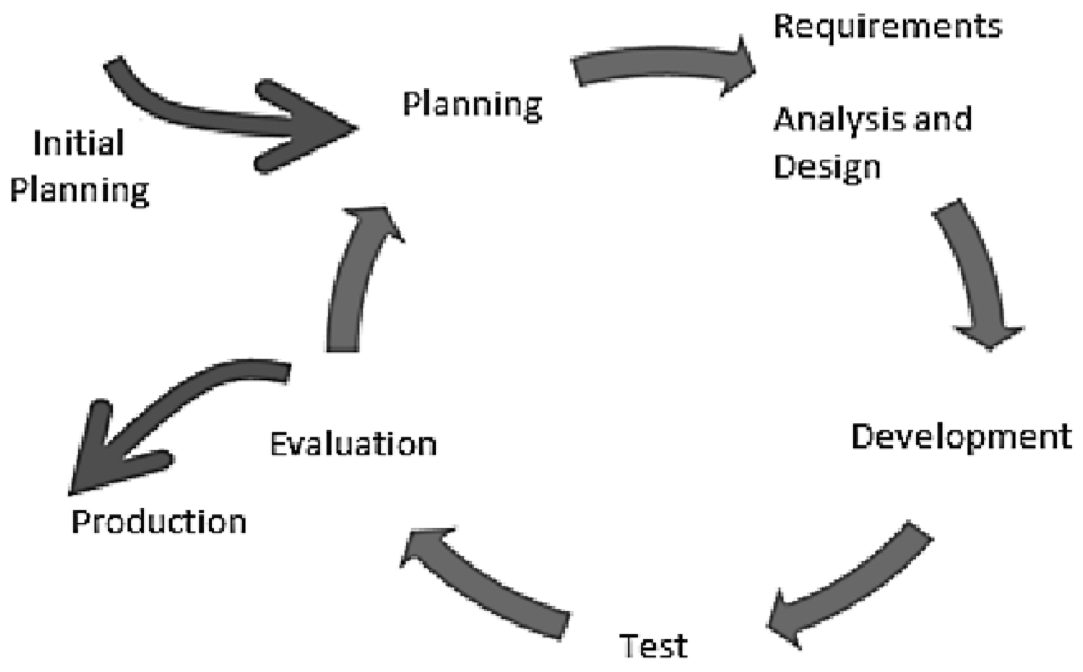


Incremental and Iterative model

Iterative model - a type of software development lifecycle model in which the component or system is developed through a series of repeated cycles.

Incremental model - a type of software development lifecycle model in which the component or system is developed through a series of increments.

It is developed to overcome the weaknesses of the waterfall model. It starts with initial planning and ends with deployment with the cyclic interactions in between. The basic idea behind this method is to develop a system through repeated cycles (iterative) and in smaller portions at a time (incremental), allowing software developers to take advantage of what was learned during the development of earlier parts or versions of the system. It can consist of mini waterfalls or mini V-Shaped model



Iterative and Incremental model

Model Application

Like other SDLC models, Iterative and incremental development has some specific applications in the software industry. This model is most often used in the following scenarios:

- Major requirements must be defined; however, some functionalities or requested enhancements may evolve with time.
- There is a time to the market constraint.
- There are some high risk features and goals which may change in the future.

Model Pros

- Produces business value early in the development lifecycle.
- Better use of scarce resources through proper increment definition.
- Can accommodate some change requests between increments.
- More focused on customer value than the linear approaches.
- We can detect project issues and changes earlier.

Model Cons

- Requires heavy documentation.
- Follows a defined set of processes.
- Defines increments based on function and feature dependencies.
- Requires more customer involvement than the linear approaches.
- Partitioning the functions and features might be problematic.
- Integration between the iterations can be an issue if it is not considered during the development and project planning.

Still doesn't work!

Agile

Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product.

It can be used with any type of project, but it needs more engagement from the customer and to be interactive. Also, we can use it when the customer needs to have some functional requirement ready in less than three weeks and the requirements are not clear enough. This will enable more valuable and workable piece for software early which also increase customer satisfaction.

Agile Manifesto

Individuals and interactions over processes and tools

Working software over comprehensive documentation

Customer collaboration over contract negotiation

Responding to change over following a plan

Principles behind the Agile Manifesto

Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

Business people and developers must work together daily throughout the project.

Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

Working software is the primary measure of progress.

Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

Continuous attention to technical excellence and good design enhances agility.

Simplicity--the art of maximizing the amount of work not done--is essential.

The best architectures, requirements, and designs emerge from self-organizing teams.

At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Agile Manifesto

Individuals and interactions over processes and tools

Agile development is very people-centered. Teams of people build software, and it is through continuous communication and interaction, rather than a reliance on tools or processes, that teams can work most effectively.

Working software over comprehensive documentation

From a customer perspective, working software is much more useful and valuable than overly detailed documentation and it provides an opportunity to give the development team rapid feedback. In addition, because working software, albeit with reduced functionality, is available much earlier in the development lifecycle, Agile development can confer significant time-to-market advantage. Agile development is, therefore, especially useful in rapidly changing business environments where the problems and/or solutions are unclear or where the business wishes to innovate in new problem domains.

Customer collaboration over contract negotiation

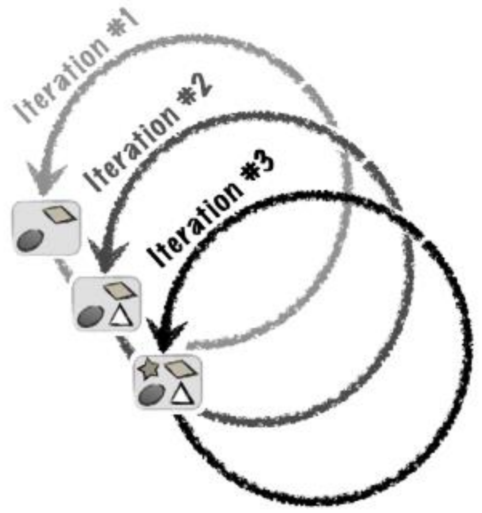
Customers often find great difficulty in specifying the system that they require. Collaborating directly with the customer improves the likelihood of understanding exactly what the customer requires. While having contracts with customers may be important, working in regular and close collaboration with them is likely to bring more success to the project.

Responding to change over following a plan

Change is inevitable in software projects. The environment in which the business operates, legislation, competitor activity, technology advances, and other factors can have major influences on the project and its objectives. These factors must be accommodated by the development process. As such, having flexibility in work practices to embrace change is more important than simply adhering rigidly to a plan.

So why has agile gotten so popular?

- When teams go agile, they find that it’s a lot easier to meet their deadlines.
- They also find that they can really cut down on bugs in their software.
- Their code is a lot easier to maintain—adding, extending, or changing their codebase is no longer a headache.
- The users are a lot happier, which always makes everyone’s lives easier.
- And best of all, when agile teams are effective, the team members’ lives are better, because they can go home at a reasonable hour and rarely have to work weekends (which, for a lot of developers, is a first!).



So what is agile, anyway?

Agile is a set of methods and methodologies that are optimized to help with specific problems that software teams run into, and kept simple so they're relatively straightforward to implement.

These methods and methodologies address all of the areas of traditional software engineering, including project management, software design and architecture, and process improvement. Each of those methods and methodologies consists of practices that are streamlined and optimized to make them as easy as possible to adopt.

Agile is also a mindset, and that's a new idea for a lot of people who haven't worked with agile before. It turns out that each team member's attitude toward the practices they use can make a huge difference in how effective those practices are.

The agile mindset is focused on helping people share information with each other, which makes it much easier for them to make important project decisions (rather than just relying on a boss or project manager to make those decisions).

It's about opening up planning, design, and process improvement to the entire team. To help everyone get into an effective mindset, each agile methodology has its own set of values that team members can use as a guide.

Scrum

An iterative incremental framework for managing projects commonly used with Agile software development. Scrum is the most common approach to agile.

There are many ways that teams can be agile, and there’s a long list of methods and methodologies that agile teams use. But there have been many surveys done over the years that have found that the most common approach to agile is Scrum, a software development framework focused on project management and product development. When a team uses Scrum, every project follows the same basic pattern.

Scrum

- Скрам — це фреймворк управління, згідно з яким одна чи декілька кросфункціональних команд створюють продукт інкрементами, тобто, поетапно
- В команді може бути близько семи людей
- У скрамі є система ролей, подій, правил і артефактів. У цій моделі за створення й адаптацію робочих процесів відповідають команди
- У скрамі використовуються ітерації фіксованої тривалості, які називаються **спринтами**. Зазвичай вони займають 2-5 тижні
- Спринт є основою Скраму, адже саме в них прості ідеї перетровюються на цінність
- Скрам-команди прагнуть створювати готовий до релізу інкремент продукту в кожній ітерації

Scrum Rollers

Product Owner

Захистник продукту, який повністю розуміє його цінність для бізнесу. Ця людина доносить потреби замовника і стейкхолдерів до команди розробки, але не відповідає за технічний бік процесу.

Scrum Master

Виступає фасилітатором роботи скрам-команди. Скрам-майстер допомагає власнику продукту й команді розробки виконувати роботу без перешкод і відволікаючих факторів.

Development team

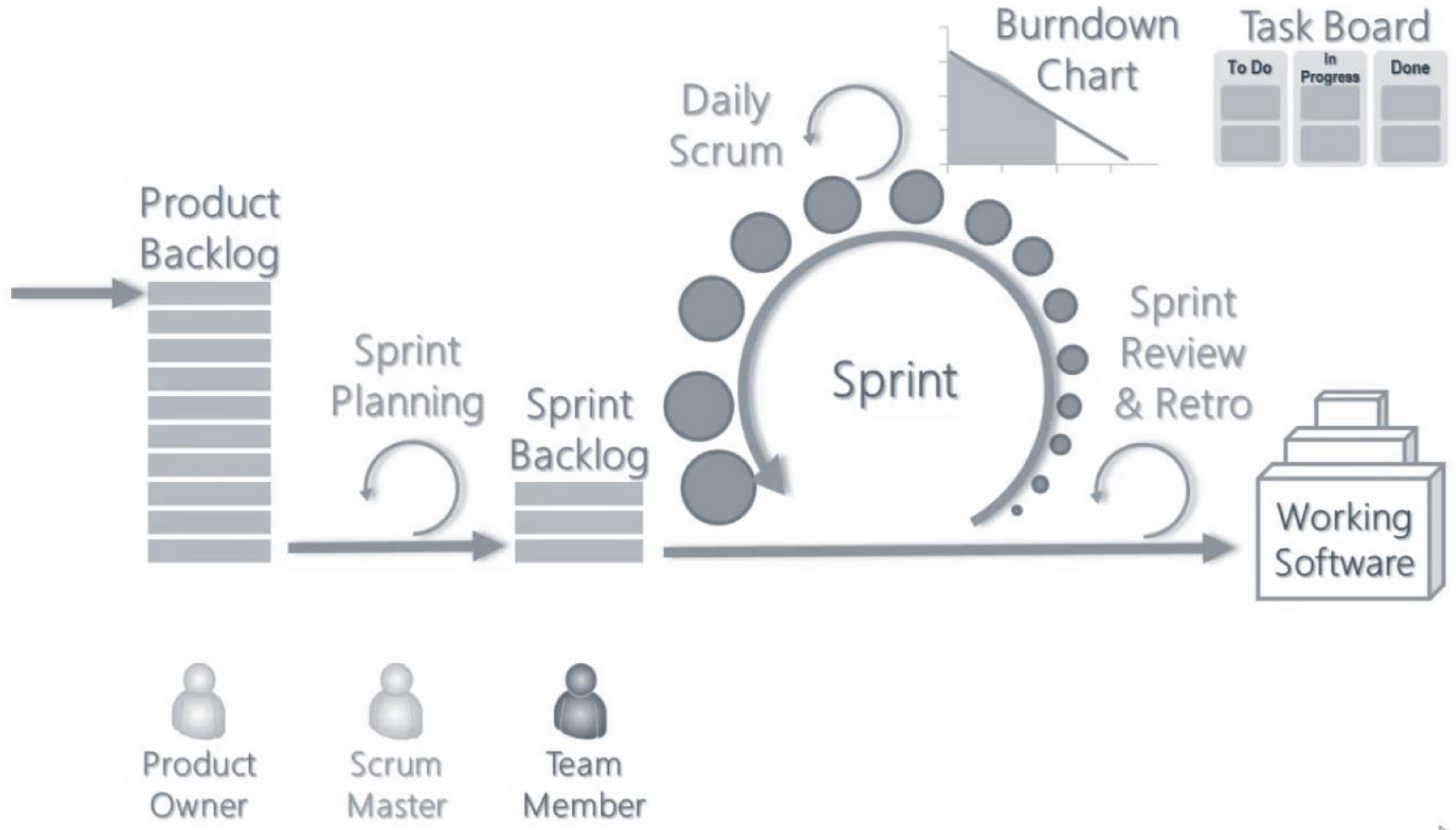
Виконує всі технічні завдання з розробки. Команда є кросфункціональною й відповідає за аналіз, дизайн, програмування, тестування, технічну комунікацію тощо

Scrum Meetings

Sprint Planning	Sprint Planning initiates the Sprint by laying out the work to be performed for the Sprint. This resulting plan is created by the collaborative work of the entire Scrum Team.
Daily Scrum	The purpose of the Daily Scrum is to inspect progress toward the Sprint Goal and adapt the Sprint Backlog as necessary, adjusting the upcoming planned work.
Sprint Review (Demo)	The purpose of the Sprint Review is to inspect the outcome of the Sprint and determine future adaptations. The Scrum Team presents the results of their work to key stakeholders and progress toward the Product Goal is discussed.
Sprint Retrospective	The purpose of the Sprint Retrospective is to plan ways to increase quality and effectiveness.
Backlog Refinement	In the Backlog Refinement Meeting, the team estimates the amount of effort they would expend to complete items in the Product Backlog and provides other technical information to help the Product Owner prioritize them.

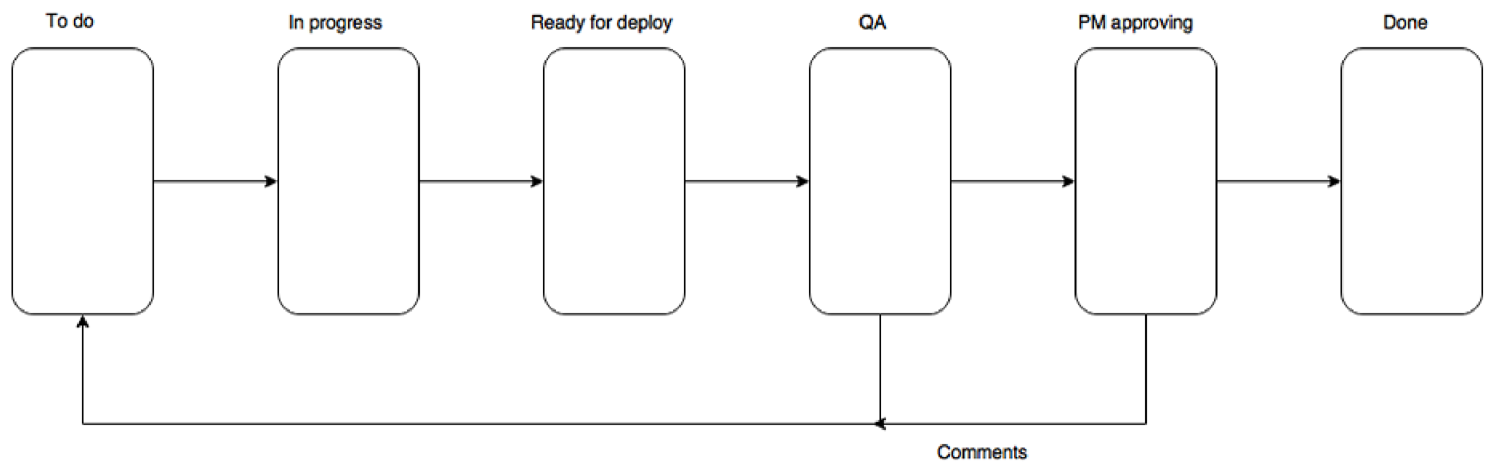
Scrum Artifacts

Product Backlog	The Product Backlog is an emergent, ordered list of what is needed to improve the product. It is the single source of work undertaken by the Scrum Team
Sprint Backlog	The Sprint Backlog is composed of the Sprint Goal (why), the set of Product Backlog items selected for the Sprint (what), as well as an actionable plan for delivering the Increment (how).
Increment	An Increment is a concrete stepping stone toward the Product Goal.
Task Board	Board which consists of rows and columns - each row is a task and columns is a status: “To Do”, “Work In Process”, “To Verify”, “Done”..
Burndown chart	is a graphical representation of work left to do versus time.


Product Backlog
Sprint Planning
Sprint Backlog
Daily Scrum
Burndown Chart
Task Board
To Do
In Progress
Done
Sprint
Sprint Review & Retro
Working Software
Product Owner
Scrum Master
Team Member

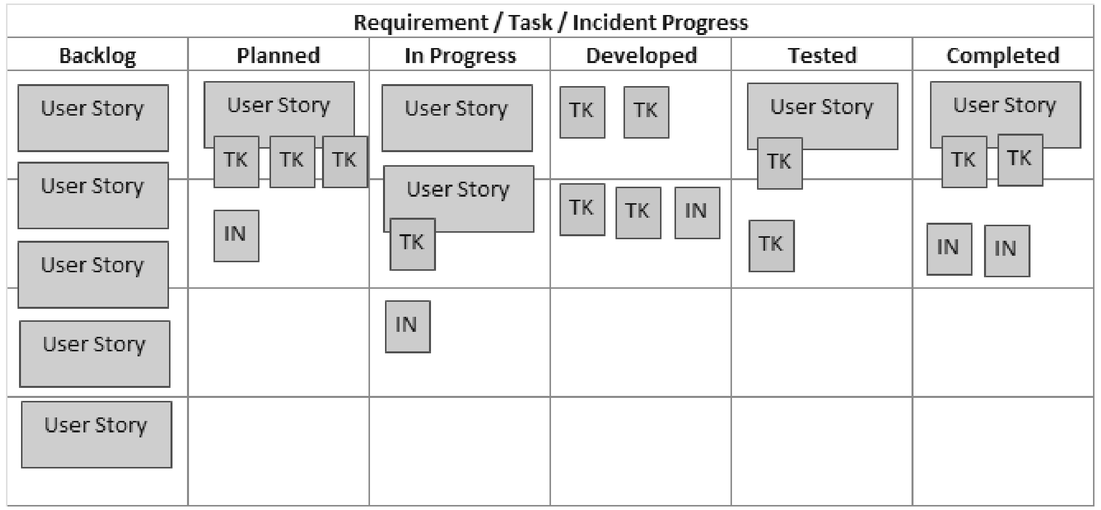
Kanban

Kanban is a workflow management method for defining, managing and improving services that deliver knowledge work. It aims to help you visualize your work, maximize efficiency, and improve continuously.



Kanban

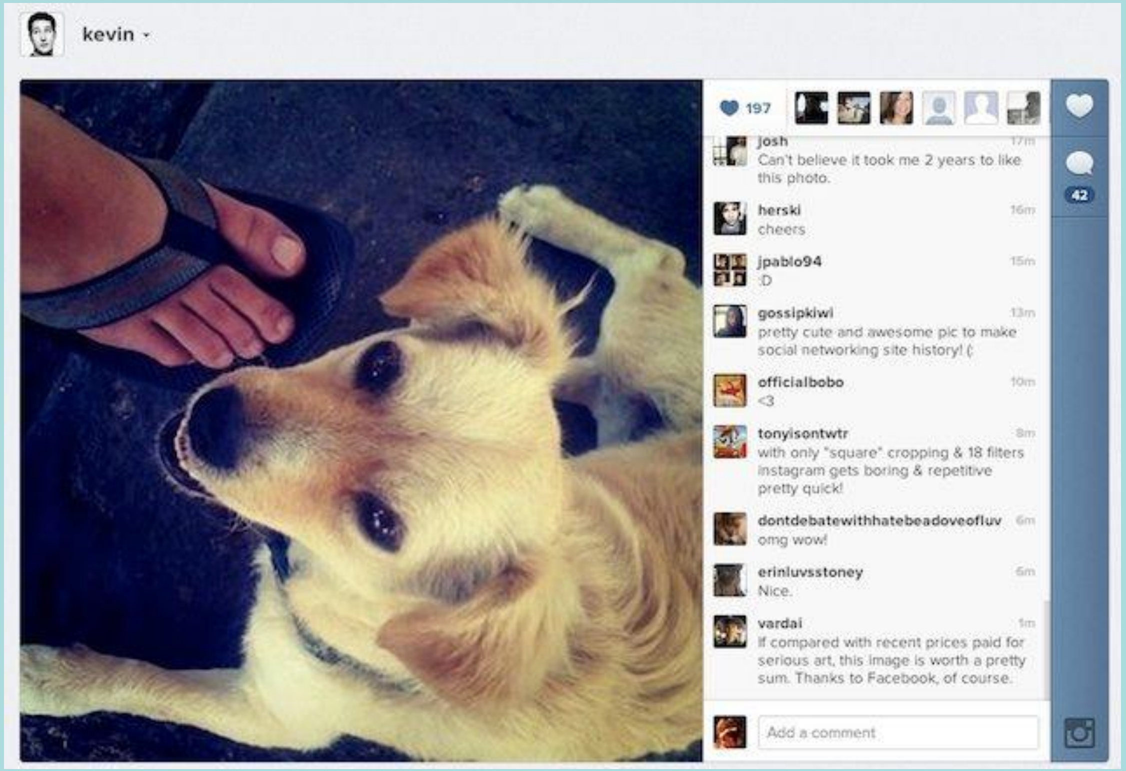
- Допомагає налагодити поточні процеси і не перевантажити команду.
- Незавершені завдання не простоюють і рухаються потоком по ланцюжку створення продукту або його підтримки.
- Систему канбан вигадали на заводі Toyota.
- Kanban – це насамперед візуалізація. Для візуалізації використовують дошку та набір карток.
- Канбан використовує три інструмети:
    1. Канбан дошка
    2. Ліміт незавершеної роботи
    3. Час виконання



History of Instagram

Timeline from 2010 to 2021

October 6, 2010	Was founded (Instagram was originally only available for iPhone until 2012)
December 12, 2010	Instagram hits 1 million users
January 2011	Instagram introduces hashtags on the platform
April 3, 2012	Instagram launches on Android
April 9, 2012	Facebook announces they are buying Instagram for 1 billion dollars
May 18, 2012	Facebook goes public at $42 a share
February 2013	Instagram hits 100 million monthly active users
September 2015	Instagram ads launch globally
March 15, 2016	Instagram announces the feed is changing from chronological to algorithmic
May 31, 2016	Instagram launches IG business pages
August 2, 2016	Instagram announces 'Stories'
February 22, 2017	Instagram launches carousel posts allowing multiple photos or videos in a single post
Jun 20, 2018	Instagram announces it has 1 billion users
June 20, 2018	Instagram launches IGTV and its standalone app
May 19, 2020	Instagram announces 'shops' on IG
August 5, 2020	Instagram Reels launch in over 50 countries worldwide
July 27, 2021	Instagram officially announces Instagram Reels can be up to 60 seconds long (to further compete with TikTok)



josh 17m
Can't believe it took me 2 years to like this photo.

herski 16m
cheers

jpablo94 15m
:D

gossipkiwi 13m
pretty cute and awesome pic to make social networking site history! (:

officialbobo 10m
<3

tonyisontwtr 8m
with only "square" cropping & 18 filters instagram gets boring & repetitive pretty quick!

dontdebatewithhatebeadoveofluv 6m
omg wow!

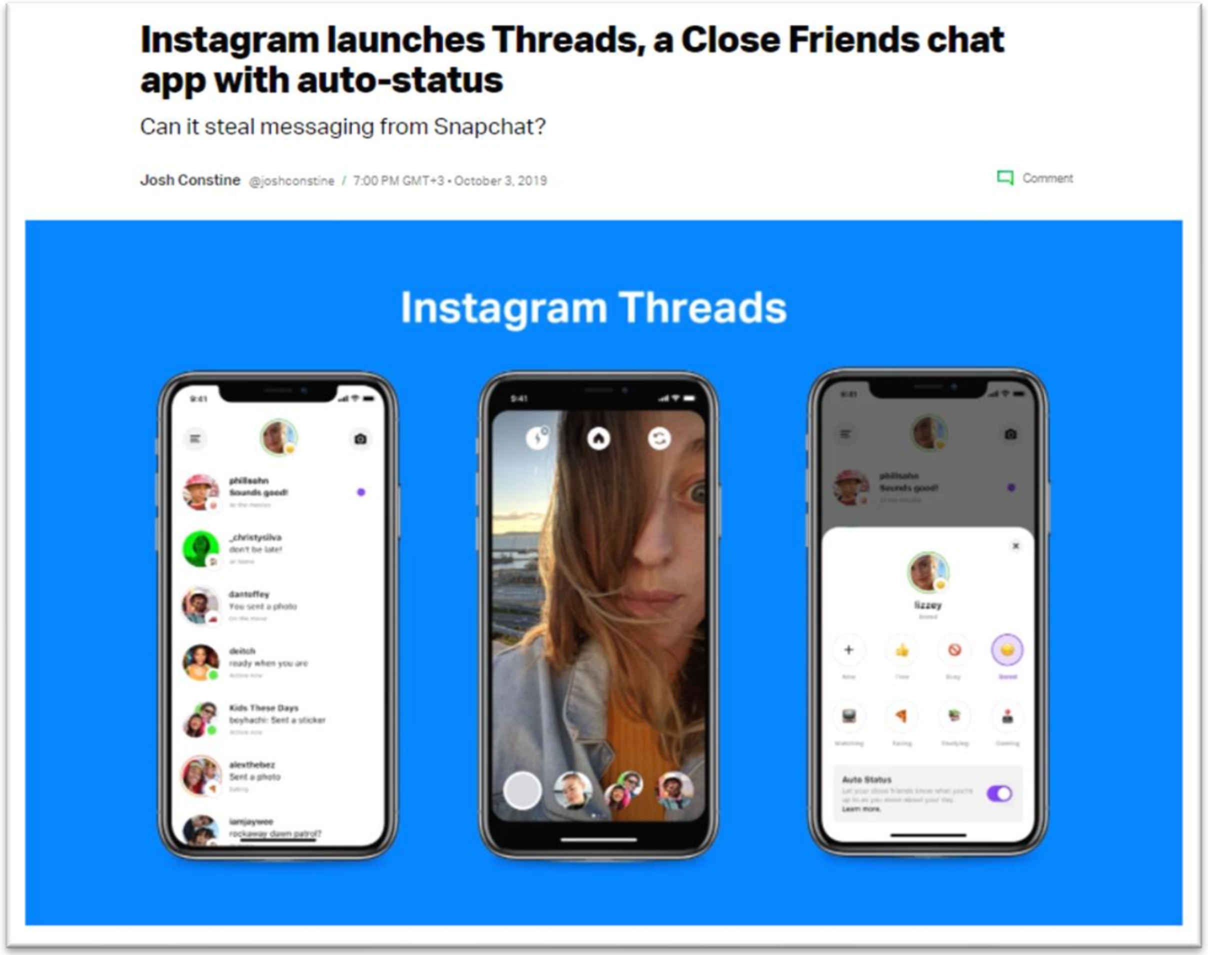
erinluvsstoney 6m
Nice.

vardai 1m
If compared with recent prices paid for serious art, this image is worth a pretty sum. Thanks to Facebook, of course.

Instagram Threads

October 2019

November 2021

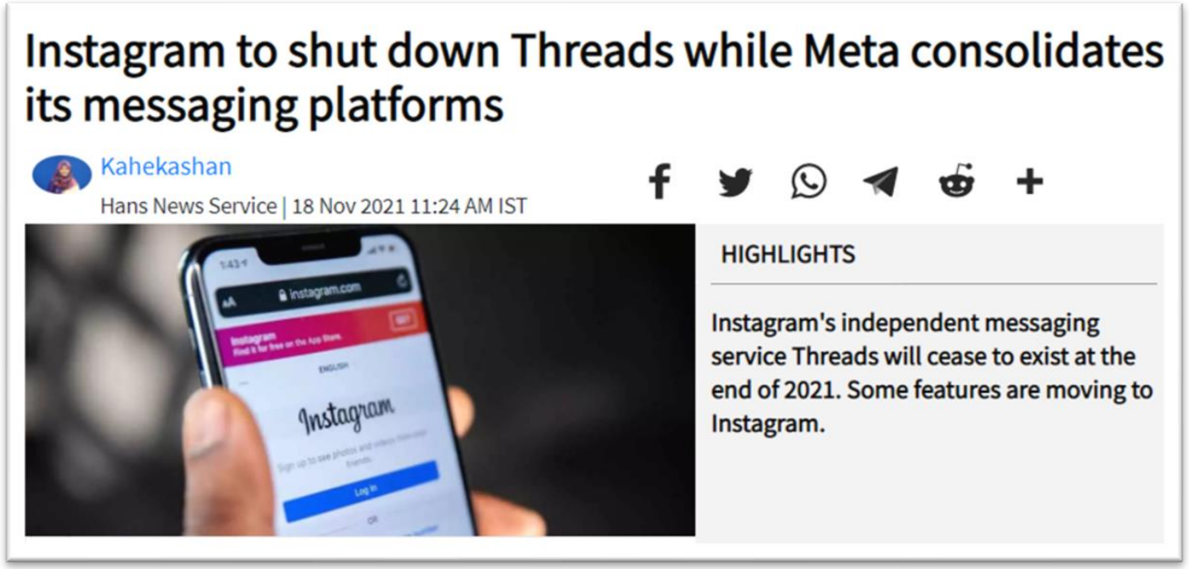


Instagram launches Threads, a Close Friends chat app with auto-status

Can it steal messaging from Snapchat?

Josh Constine @joshconstine / 7:00 PM GMT+3 • October 3, 2019

Comment



Instagram to shut down Threads while Meta consolidates its messaging platforms

Kahekashan

Hans News Service | 18 Nov 2021 11:24 AM IST

HIGHLIGHTS

Instagram's independent messaging service Threads will cease to exist at the end of 2021. Some features are moving to Instagram.

Tester in Agile

A tester on an Agile project will work differently than one working on a traditional project. Testers must understand the values and principles that underpin Agile projects, and how testers are an integral part of a whole-team approach together with developers and business representatives. The members in an Agile project communicate with each other early and frequently, which helps with removing defects early and developing a quality product

IT specialists

Business Analytics | **Project Manager** | **Product Manager** | **Designer** | **Developer** | **QA**

Testing Activities

Testing Activities

Planning	Планування складається з активностей, які визначають цілі тестування та підхід до досягнення цілей тестування з обмеженнями, що накладаються контекстом
Monitoring and Control	Моніторинг передбачає безперервне порівняння фактичного ходу роботи із планом тестування. Контроль тестування передбачає вжиття заходів, необхідних для досягнення цілей плану тестування
Analysis	У процесі аналізу тестування аналізують базис тестування для визначення тестованих функцій і встановлення відповідних тестових умов. Інакше кажучи - “що будемо тестувати?”
Design	Відбувається проектування тестових сценаріїв. Відповідає питанням «як будемо тестувати?».
Implementation	Під час реалізації тестів створюється та/або готується необхідне тестове забезпечення для виконання тестів.
Execution	Запуск тестів відповідно до розкладу виконання
Completion	Збір даних з виконаних активностей тестування

Home Task

to read:

1. This presentation
2. Scrum book
3. Software Testing (Куліков) 18 – 27 pp

to learn:

1. Vocabulary

to do:

1. Quiz

Questions

Thank you

<epam>