U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Software Engineering

ESOF – 3MIEIC05

T2 – Software Processes

Duarte Nuno Esteves André Lima de Carvalho– up201503661@fe.up.pt

João Álvaro Cardoso Soares Ferreira – up201605592@fe.up.pt

João Augusto dos Santos Lima – up201605314@fe.up.pt

Index

Table of Contents

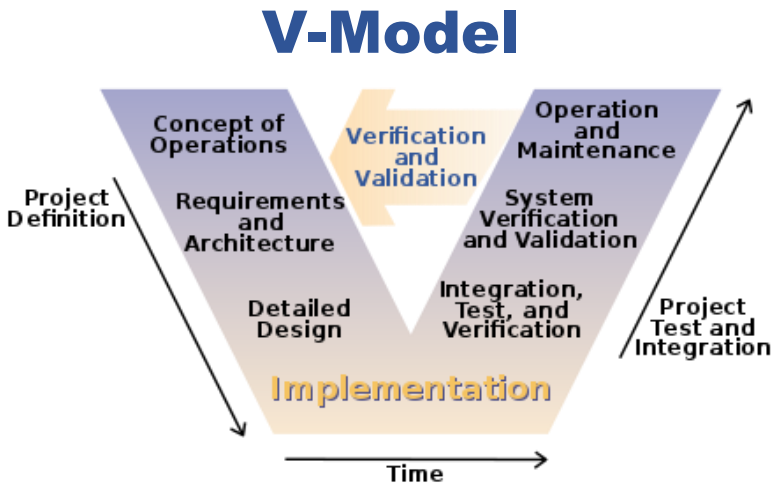


Figure 1 - A basic representation of the V-Model software development cycle

A Brief Summary

The V-Model is a strict, linear and unique model for a software development cycle that originated in Germany as an extension of the Waterfall model, being the official project management methodology used by the government in the early 1990s.

Despite its origin as an equivalent to the PRINCE2 method for project management, it's shown itself to be very relevant to software development along with systems engineering in the military and medical industries, and is still used by governments such as the USA's or the UK's. Its main asset is its reliability and guarantee of product quality, though at a loss for flexibility.

The system itself can be summed up in its shape: a V, with its two sides representing the following:

- the left side entails the verification stages, namely the requirements analysis, system design, architecture design and module design;
- the right side entails the validation stages, namely the unit, integration, system and acceptance testing.

At the bottom of the V we have the implementation, where the code is developed. The stages themselves aren't set in stone as there isn't one universal V-Model, but these are the four stages that, in one way or another (even maybe just named differently), a V-Model for software development goes through. It is also valuable to note is that while the original V-Model used verification via analysis, demonstration, investigation and testing, the model for software development uses tests.

It's important to note that the tests that are developed for the right side of the V are defined simultaneously to the concepts that they are intended to test, having a corresponding stage on the opposite side – respectively, the requirement analysis and acceptance testing, the system design and system testing, the architectural design and integration testing and the module design and unit testing. These details are crucial, as it is what differentiates the V-Model from other models for software development. While others might also be similarly strict and develop the testing simultaneously to the conceptualization, the V-Model is the only one that accompanies the project's development by cycling through the stages in this manner.

The Stages

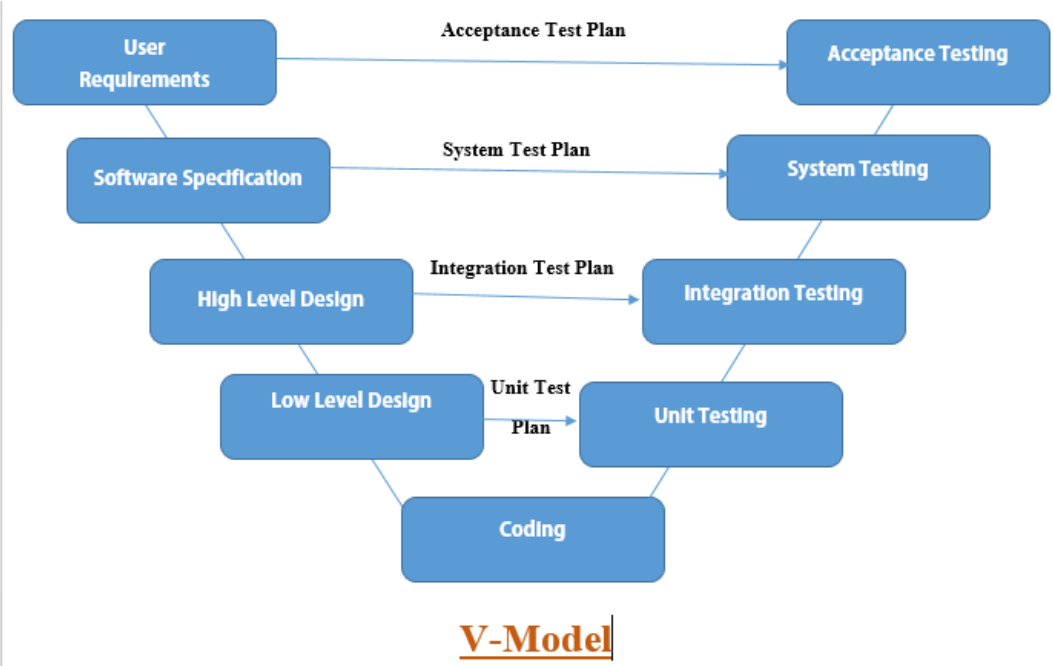


Figure 2 - Extended Software Development V-Model

User Requirements

This is the first stage of development after communication with the client has been established. Here, the user's needs are determined, as are the solutions that will be required to meet them.

System Design

In this stage, the goal is to create a document to determine the techniques, frameworks, materials and methods that will be optimal to develop the solutions outlined in the previous stage. Communication with the client is continued and their feedback is used for this stage as to what might or might not be feasible to develop.

Architecture Design

Also known as high-level design, in this stage the various concepts designed in the last two stages are connected in a functional manner, to determine how the application will link them all together. It's in this stage that class diagrams are often drawn up, for example.

Module Design

Commonly represented as "low level design", this final stage before the implementation is supposed to define all of the functions and break down the concepts in the previous stages so that the coders, in the following stage, can just start implementing the functions directly. A good tool to do this would be Doxygen documentation, for example.

Implementation

Usually the most time-consuming stage, the implementation is where all of the concepts and specifications done in the previous stages is written into code and made from theoretical to practical. All of it must be developed before the testing can begin.

Unit Testing

Developed during the module design by the development team, this phase should be able to rid the program of all bugs at a code level, being also the lengthiest of the testing stages.

Integration Testing

Developed during the architecture design, to check if all components are working together as intended.

System Testing

Often developed in contact with the consumer, these tests verify if the entirety of the application is performing as expected. This is developed during the system design.

Acceptance Testing

The last overall stage, acceptance testing is done by monitoring the contact the user has with the application, verifying that all needs are met. As mentioned before, these tests are developed during the user requirements stage.

Positives

- Due to a highly disciplined strict approach, there's an overwhelming amount of testing, documentation and stability, leading to an overall more secure product;
- Simple to use and understand, especially for small projects;
- Prevents extensive debugging, as all errors are discovered at an appropriate stage;

Negatives

- It's not an Agile model, lacking flexibility and a good response to changes during or after the development process;
- It promotes strictness and streamlining, sometimes overlooking creative solutions;
- Issues with time constraints;
- Ill-suited for overwhelming support after launch such as constant patches;
- Easy to apply and understand at first, but also prone to lead inexperienced developers to stick to its guidelines too much.

When to Use V-Model

It is ideal for projects where reliability and stability are a must-have, where costs and time-constraints aren't an issue (as both might be unexpectedly high) and where patching and updates aren't expected. It is thus very appropriate for matters of

developing software for science, military and healthcare, as was mentioned beforehand.

Example of Success

As was just mentioned, the medical industry has adopted the V-Model for its software needs, as an there are quite a few difficulties with adopting software developed with an Agile method (source) and the V-Model's reliability has shown it to be ideal for their needs. In this case, the V-Model's success has shown itself in stability in its adoption and in the lack of any major failures in the medical industry that can be attributed to it.

Sources

A Software Process Development, Assessment and Improvement Framework,for the Medical Device Industry - Dundalk Institute of Technology

What is V-Model? - airbrake.io

V-Model vs Scrum: Who Wins? - reqtest.com

Good Design Practice for Medical Device and Equipment - Cambridge University

Using V-Model for Testing - Carnegie Mellon University

introducing the V-Model with a Case Study

Course Design and Orienting Students with the "V Model" - MIT

Joint Application Development

MANAGEMENT

Joint Application Development

Introduction

Joint Application Development, JAD, it's software development methodology created at IBM, by Chuck Morris, in

1977, with the purpose of gathering the requirements for geographically distributed systems. In 1984, IBM published the JAD Overview pamphlet. By the late 1980s, many companies adopted this methodology.

Before the existence of JAD, project requirements were pointed out by stakeholders individually. This process was not effective, because it relied in individual input rather than group consensus, so JAD appeared.

Principles and values

JAD focus on team work, emphasizing consensus based problem-solving. JAD allow system requirements to be documented more quicly, by emphasizing a spirit of partnership, and combines technology and business needs in a process that is consistent, repeatable, and effective.

This process accelerates the design of projects, by using customer involvement to more easily realize customer needs and develop a solution as a group.

Project types for a successful use of JAD

- New systems
- Enhancements to existing systems
- System conversions
- Purchase of a system

Important project characteristics for JAD

- Involves many groups of users whose responsibilities cross traditional department or division boundaries
- Is considered critical to the future success of the organization
- Involves willing users
- Is a first-time project for the organization
- Has a troubled project history or relationship between the systems and user organizations

Participants

This process has the following participants:

- Executive Sponsor
- Facilitator
- User
- IT Representative
- Scribe
- Observer

Executive Sponsor

The executive sponsor is the person from the customer's organization who has the ultimate authority to make decisions, and also who facilitator works with. This element has the following responsibilities:

- Accept ultimate authority and responsibility for the functional area to be addressed by the system.
- Resolve business policy conflicts
- Set the project objectives.
- Honor the results of the JAD process.
- Ensure the project team has access to all resources and support it needs.
- Communicate customer support and cooperation.

This element communicates with the whole team to express support for a cooperative effort, confirms that the JAD process has the corporation's support and demonstrates confidence in the facilitator during the orientation session. The executive sponsor doesn't need to go to the JAD sessions, he only needs to appear occasionally to show continued interest in and commitment to the process.

Facilitator

The facilitator is one of the most important elements in this process. This role demands high skills of communication and big knowledge of the tools and techniques to be used for capturing requirements in the JAD sessions. The facilitator must also be able to communicate effectively with the different personality types present on a JAD team. the success of this process is linked to how well the facilitator handles the session.

The facilitator is responsible for:

- Organize and schedule JAD activities.
- Guide the JAD sessions.
- Mediate disputes.
- Encourage participation.
- Maintain focus.
- Enable the decision making process by summarizing the discussions.
- Have no vested interest in the outcome of the session.

It's important that this element has no particular interest in the outcome of the session. The facilitator must ask good questions in order to identify when something is not good for the project.

User

Users must:

- Serve as the main focus of JAD (65 %to 75 % of the total group).
- Provide business expertise.
- Represent all major user groups affected by the project.
- Represent the strategic, tactical, or operational direction of the business.
- Represent multiple levels of the organization.

IT Representative

When required, this element should give technical advice, help develop specifications and models. IT representatives are typically some of the key developers of the project. The IT Representative shouldn't try to force the decision making process.

This element has the following roles:

- Help customer turn ideas into models of business requirements.
- Ensure all technological constraints are represented.
- Develop an understanding of user business goals, priorities, and strategies.
- Represent other job management functions.
- Ensure a solution that is realistic for the budget, can be delivered when needed, and takes advantage of available technology most effectively.

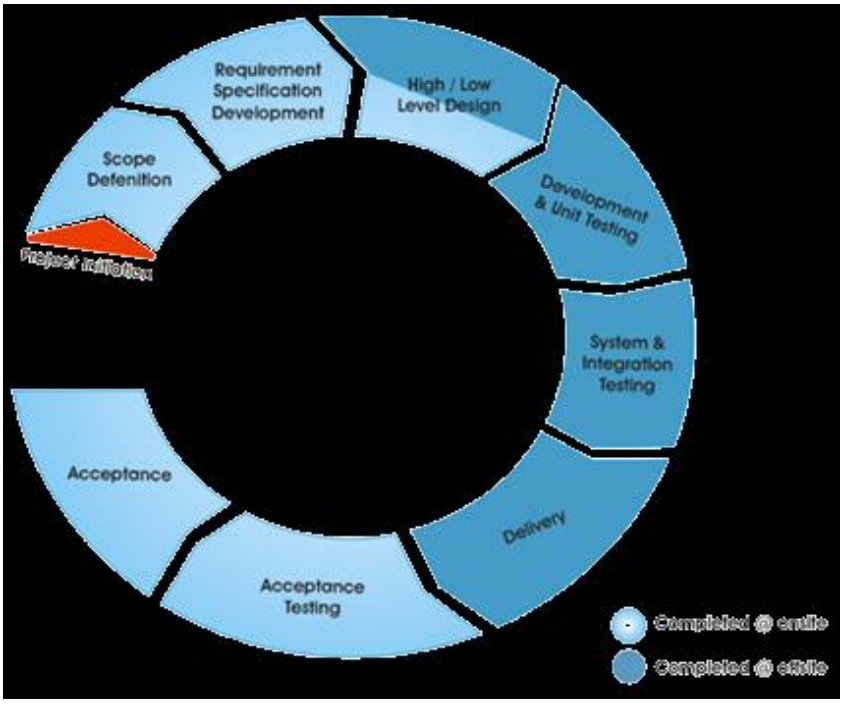
Scribe

The scribe must ensure that the results of JAD sessions are documented and delivered as planned. Serve as a partner to the facilitator before, during, and after the workshop. Provide reference and review information for the facilitator.

Observer

This element must learn about user needs and interact with the participants and facilitator only during breaks or before and after sessions.

Life cycle phases



The JAD is made up of theses stages:

- Planning
- Preparation
- Design Sessions
- Finalization

Planning

In this phase the executive sponsor shall be designated, the need for the system shall be established and the scope of the session shall be defined. During this phase, the facilitator works with the executive sponsor to provide an orientation to the JAD process. The executive sponsor's full commitment is critical.

Preparation

In the preparation stage you schedule the design sessions, conduct orientation and training for design session participants, customize the design session and conduct the kick-off meeting and prepare the materials, room, and software aids.

Design Sessions

In this phase the project scope , the objectives and the definition document are reviewed. The data, the system requirements and the system interface are identified. A prototype is developed and, after that, the decisions, issues, assumptions and definitions of terms are documented. Finally, someone is assigned to take care of all the issues.

Finalization

In this final stage the design documents are completed and signed off. after that, in order to obtain the executive sponsor approval to proceed, a presentation must be made and the prototype must be demonstrated. Finally, the process must be evaluated.

JAD: Pros and Cons

Pros

- Allows you to overcome challenges more simply and produce better, error-free software.
- Company and clients collaboration lowers risks.
- JAD is more economical in terms of cost and time.
- Well-defined requirements improve system quality.
- Faster progress.
- Teamwork leads to work faster and respect deadlines.

Cons

- Different ideas and prespectives make it harder to focus and acomplish objectives.
- JAD may require long sessions depensing on project size

Conclusion

JAD is a technique which focuses on teamwork between developers, company or executives and users, in order to achieve consensus based system requirements. This is obtained by using trained JAD facilitators and customized, planned agendas to assist the participant in arriving at complete, high quality requirements.

References

https://activecollab.com/blog/project-management/joint-application-development

https://study.com/academy/lesson/joint-application-development-definition-phases-methodology.html

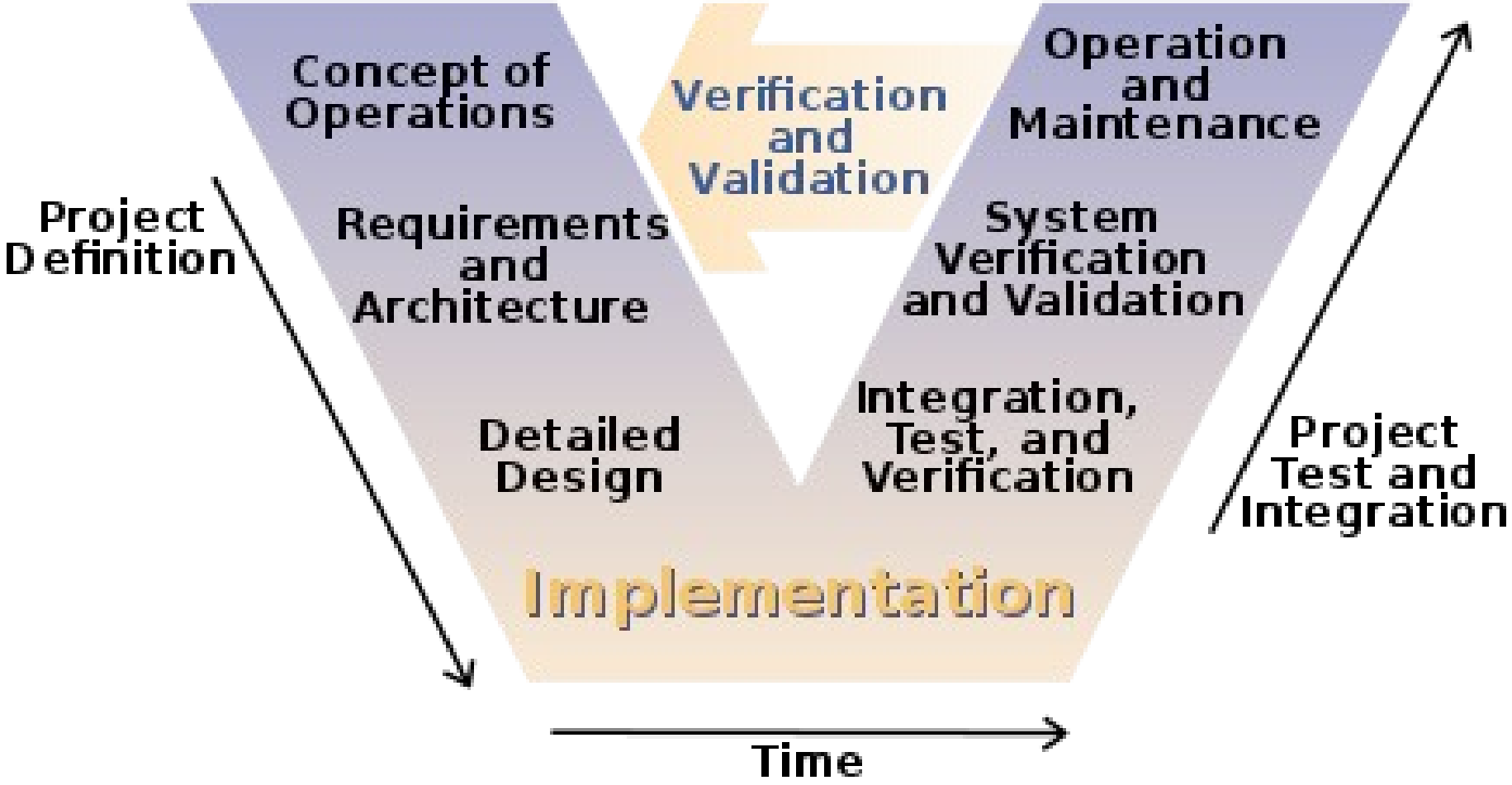
V-Model Presentation

Presentation for the class of ESOF - 3MIEIC05

T2 - Software Processes

Duarte Nuno Esteves André Lima de Carvalho– up201503661@fe.up.pt
João Álvaro Cardoso Soares Ferreira – up201605592@fe.up.pt
João Augusto dos Santos Lima – up201605314@fe.up.pt

What Is The V-Model?



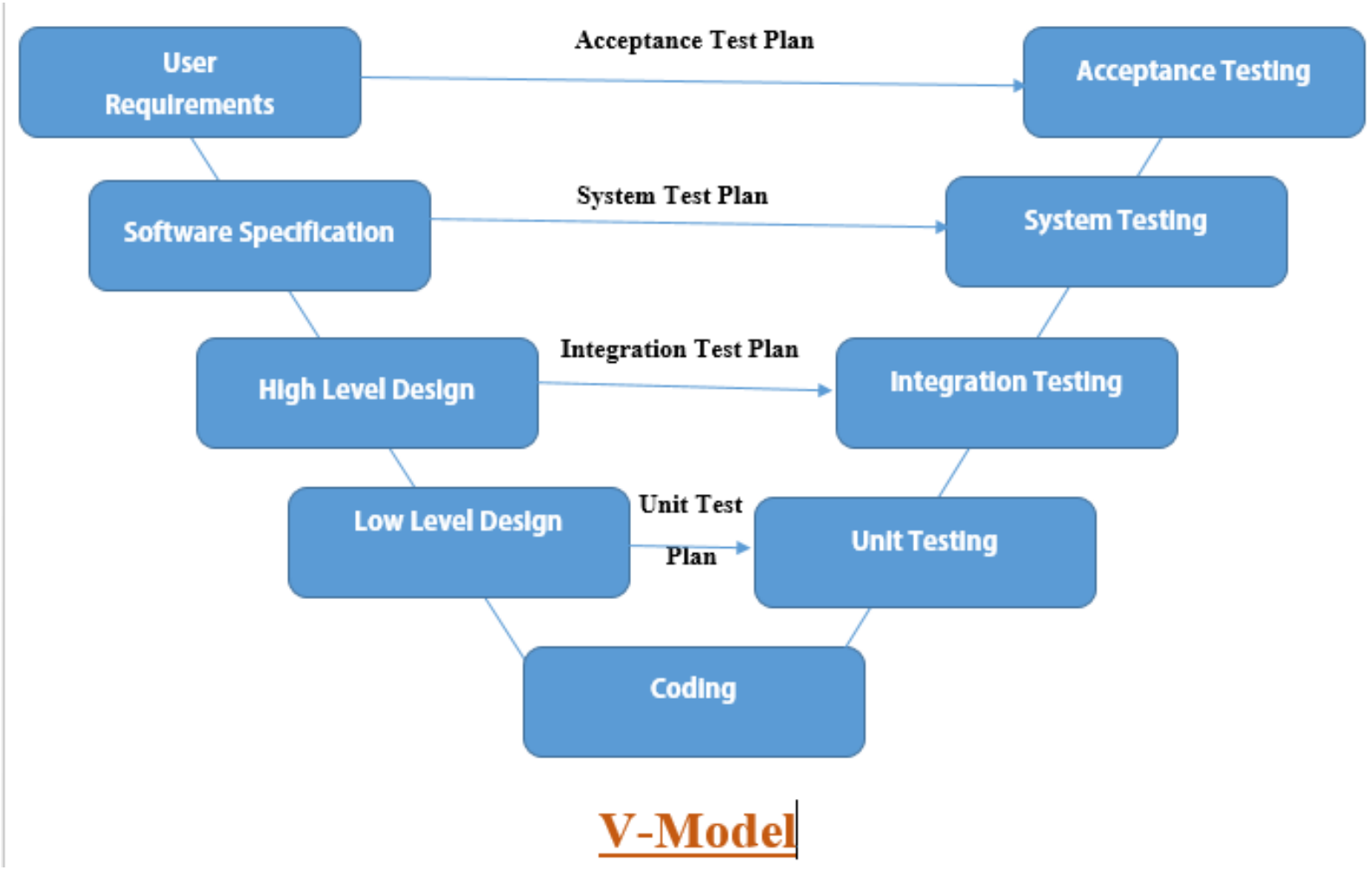
What Is The V-Model?

- **V-Model is a strict model for software development developed as an extension from the Waterfall model.**
- **It can be summarized by its shape, a V:**
  - The left side has the verification stages.
  - The right side is comprised of the validation stages.
  - At the point where both sides meet, at the bottom of the V, code is developed and implemented.
  - The tests on the right side are developed at the same time as the respective concepts on the left side.

Origin of the V-Model

- The V-Model, or Vee Model, was created by the German government in the 1990s to manage project development in their defence department.
- It was developed as a counter-part to the PRINCE2 model, and is to this day still used by the UK and USA's governments.

The Stages



V-Model

The Stages – Left Side (Verification Stages)

- User Requirements – Where the user's needs and respective solutions are determined.
- System Design – Where the techniques and methods that will be needed to accomplish the solutions are decided.
- Architecture Design – Where the high-level concepts for the code are designed, making use of tools such as class diagrams.
- Module Design – Where the functions to be developed in the implementation stage are defined.

Implementation

The middle-point of the V and usually the most time consuming stage, this is where the code is developed.

The Stages – Right Side (Validation Stages)

- Unit Testing – the goal of this phase is to get rid of bugs at a code level, being the lengthiest of the testing stages.
- Integration Testing – checks if all components are working together properly.
- System Testing – often developed while in contact with the consumer, these tests verify if the entirety of the system is working as expected and as was designed.
- Acceptance Testing – the last overall stage, it checks if all the needs of the client are met.

Pros and Cons

Pros

- Simple to use and understand, especially for smaller projects.
- Prevents extensive debugging, as all issues are discovered at an appropriate stage.
- There’s an overwhelming amount of testing, documentation and stability due to the highly strict approach, leading to a more stable product.

Cons

- Lacks flexibility and ways to respond to unexpected problems during development.
- Promotes strictness and streamlining, sometimes overlooking creative solutions.
- Issues with time constraints.
- Ill-suited for continued support after launch (for example, with patches).
- Due to how easy it is to understand at first, inexperienced developers might stick to its guidelines too much.

When to use the V-Model

The V-Model is ideal for projects where stability is a priority, at the cost of a higher budget and lengthier development time, along with a product that won't need frequent patches.

Examples of V-Model Usage

The V-Model is frequently used in software for:

- Healthcare
- Military
- Government

Sources

A Software Process Development, Assessment and Improvement Framework,for the Medical D evice Industry - Dundalk Institute of Technology

What is V-Model? - airbrake.io

V-Model vs Scrum: Who Wins? - reqtest.com

Good Design Practice for Medical Device and Equipment - Cambridge University

Using V-Model for Testing - Carnegie Mellon University