The V-Model is Dead. Long Live the V-Model!

Dr. Royce Recommends an Iterative and Incremental Approach

Colin Hood, Colin Hood Systems Engineering

1 Aims

To explain that a V-model documents relationships between information, and does not restrict the sequence of creation of information.

To explain that a V-model shows that larger systems can be considered to be a number of sub-systems to aid specification, and that sub-systems can be further considered to consist of a number of components to aid specification. A V-model also shows that components can be integrated to create sub-systems, and that sub-systems can be integrated to create systems.

2 Introduction

This paper does not concern itself with the German Vorgehensmodell which is correctly, but sometimes confusingly, also known as the V-Model. This German Vorgehensmodell gives guidance and rules for realisation of German government projects.

This paper introduces a simple V-Model.

Based upon part of this a simple information model is presented in order to structure the main part of the discussion in this paper. Each level of the left side of a V-model is described.

A V-model is a static model that helps to structure a system into smaller parts, and does not introduce any sequence of creation of specifications or implementation.

3 A Simple Information Model

Purpose: to simplify the structuring of specifications of systems, their component parts, and the verification and validation thereof.

Consider the simplified information model shown in the form of a V.

The left side represents requirements (things to achieve), and the right side represents planned verification (checking that the thing is produced right), and validation (checking that the right thing is produced).

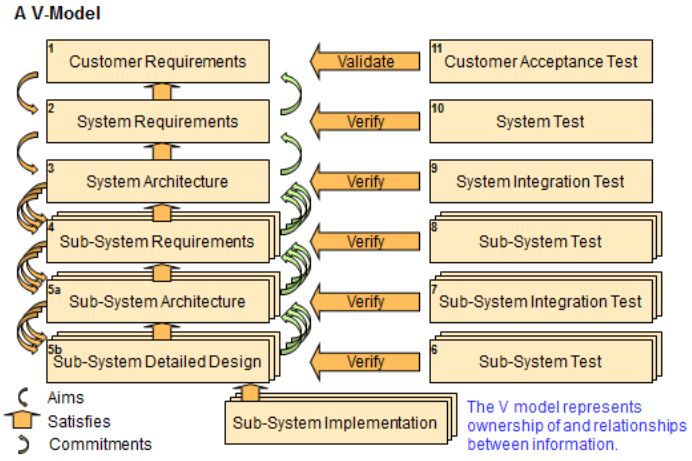

Figure 1: A V-Model

Let´s concentrate on the left side, so that the right-hand side is not considered here. The right-hand side is important but is outside the scope of discussion here. The right-hand side is where the plans to integrate parts are documented and intended verification and validation are specified.

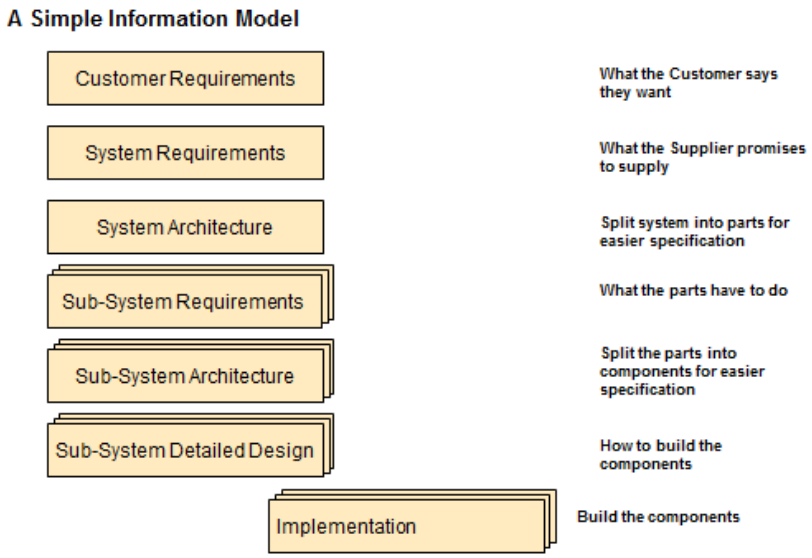

Figure 2: Simple Information Model

The boxes in the simple information model represent documents that are specifications, e.g. Customer Requirements Specification.

The numbers at each level are used as a short-hand reference for simplification of referring to a specific level on the information model, or to a specific type of specification e.g. the Customer Requirements Specification is level 1. The numbers do not represent a sequence or level of detail. Each level consist of the definition of responsibility e.g. the role of System Architect, the definition of a process to achieve the aim of each level e.g. Specify System Architecture, and the definition of a work-product e.g. System Architectural Specification. These 3 things are summarised by the use of a number to refer to the definition of responsibility, process, and work-product.

The specifications may even all be worked on at the same time.

4 Considering each Level

Purpose: to better understand each level of a V-model, and their effects on each other.

The sequence of these sub-chapters, and the numbering thereof, does not denote a sequence of creation of the specifications.

Level 1 Customer Requirements Specification

Purpose: to enable a common understanding of a Customer’s needs and wishes.

Customer wishes are documented. We shall call this type of information level 1. The Customer is responsible for this. The work might be done by other experts, but the customer retains responsibility. The customer may wish for anything. So quite naturally and correctly the Customer Requirements Specification might, from time-to-time, restrict the solution.

Level 2 System Requirements Specification

Purpose: to enable a common understanding of a commitment to supply a particular solution as required by the Customer Requirements Specification

A Supplier documents what they have committed to supply. We shall call this type of information level 2. At this point compliance or non-compliance with the customer wishes are documented. The System Analyst is responsible for this. The work might be done by other experts, but the System Analyst retains responsibility.

The System Architect will be involved to ensure that what is promised by the System Analyst is feasible within the resources available.

Level 3 System Architecture Specification

Purpose: to document the splitting of a system specification to enable better specification of the whole by the specification of sub-systems and how the sub-systems are to interact.

As part of a design process, requirements that have been, or will be, committed to be supplied (level 2) are allocated to be satisfied by parts of the system under development. We shall call this type of information level 3. Based upon the design decisions made at this level, the requirements are either re-written (split or further detailed) or left unchanged according to needs.

The System Architect is responsible for this definition of requirements at level 3.

The main requirements work at level 3 can be summarised as four parts:

Part A is the introduction of interfaces between the parts of the system as defined by the System Architect.

Part B is requirements are introduced e.g. to specify transforming information between the system interface and the hardware software interface.

Part C Requirements to be fulfilled by sub-systems need to be written at level 3 to use the information at the sub-system interface. At level 3 the content of requirements describing behaviour can normally have the same content as the requirements at level 2 but the interface used is different.

Part D specifies the collaboration between the sub-systems.

Level 4 Sub-system Requirements Specification

Purpose: to enable a common understanding of a commitment to supply a particular solution to satisfy the requirements as specified by the System Architecture.

We shall call this type of information level 4. The analysts for each sub-system are responsible. It is easy for the requirements at level 4 to be the same as the requirements at level 3.

The System Architect and Sub-System Analyst working together to ensure agreement can prevent the need to repeat requirements from level 3 to level 4. Extended across the whole information model this way of working has shown savings of between 40 and 80% of the effort needed to define and release requirements.

Level 5a Sub-system Architecture Design Specification

Purpose: to document the splitting of a sub-system specification to enable better specification of the whole by the specification of components and how the components are to interact.

As part of a design process, requirements that have been agreed to be satisfied by the creation of a sub-system (level 4) are allocated to be satisfied by components of the sub-system under development. We shall call this type of information level 5a. Based upon the design decisions made at this level, the requirements are either re-written (split or further detailed) or left unchanged according to needs.

The Sub-System Architect (e.g. SW Architect) is responsible for this.

Level 5b Sub-system Detailed Design Specification

Purpose: to document the detailed design of how a component is to be built to satisfy its requirements.

At level 5b requirements are transformed into design. That is, wishes for results to be achieved are transformed into concrete plans for implementation.

This level is strongly influenced by information and experience from lower levels. Experts in implementation will work with the architects to ensure that what is planned is realistic given the available resources.

5. Feature Owner and Chief Requirements Engineer

The V-model does not restrict a person from performing more than one role.

Using small teams to flexibly fulfill all roles on the left side of the V-model to create a specification for a feature has been shown to increase commitment because the people enjoy taking responsibility for ensuring building something with which they can identify. The teams are responsible that customer requirements are fulfilled.

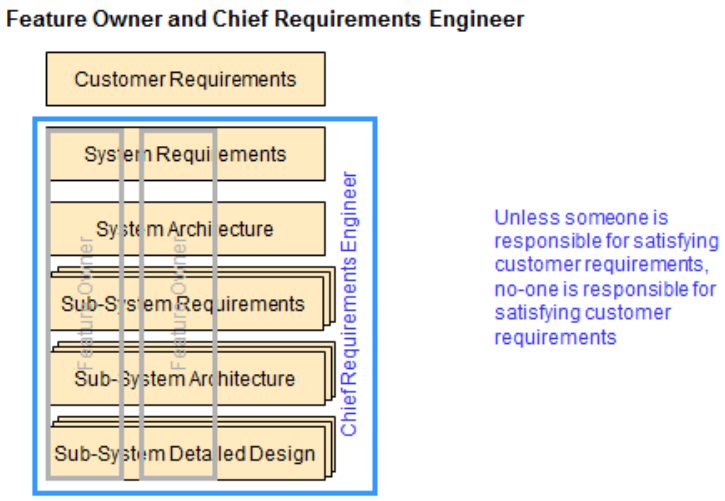

Figure 3: Feature Owner and Chief Requirements Engineer

6. Development of requirements across the levels of the information model.
A lot of people associate a V-model with the Waterfall Model.

A V-model is a static model that specifies roles, tasks, information, and relationships between information. The information may be produced in any order that makes sense to the authors.

Requirements from higher levels will guide the production of requirements at lower levels. Also requirements at lower levels guide production of requirements at higher levels as the possibility to satisfy higher level requirements is checked and compromises negotiated.

Requirements from higher levels set aims to be achieved. Requirements from lower levels document the constraints that design decisions and real-world situations introduce.

The Waterfall was first introduced in an article by Dr. Winston W. Royce in 1970 in his article entitled MANAGING THE DEVELOPMENT OF LARGE SOFTWARE SYSTEMS. Dr.Royce recommends an iterative incremental approach to software development. In building to this conclusion he built up a model from a uni-directional flow of tasks to a more complete and practical model. See Figure 4.

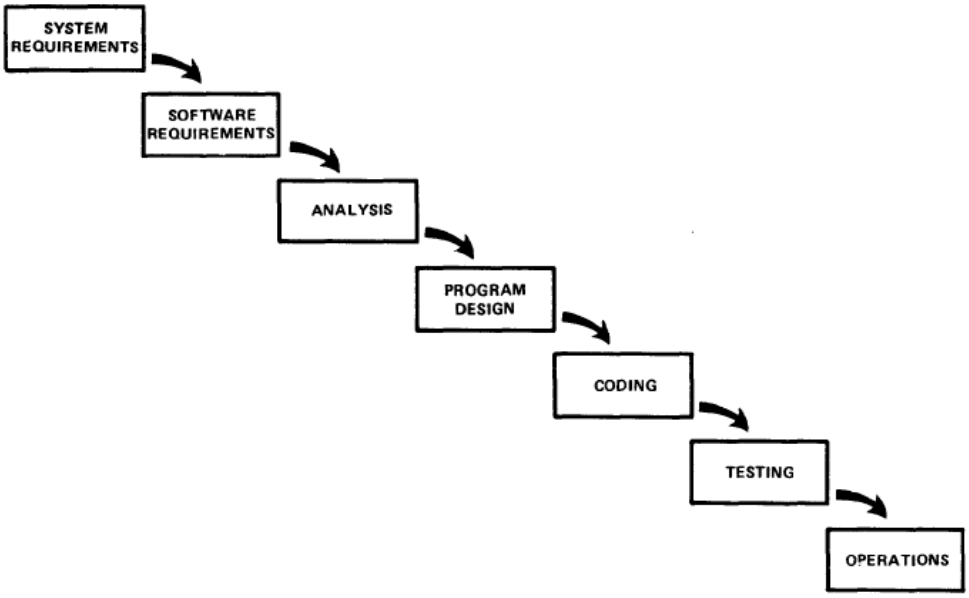

Figure 4: The First Waterfall Model (Royce 1970)
Source: https://www.cs.umd.edu/class/spring2003/cmsc838p/Process/waterfall.pdf

Dr. Royce used this to emphasise that this was not the recommended way to develop large software systems. Unfortunately over the years many have not read the paper but the diagram has entered legend, and some people have not taken the warning not to use the waterfall model. The waterfall model had the charm of being simple and memorable. Unfortunately the waterfall model did not have the charm of being very useful.

7 Agile

The basis for what today is called "Agile" is laid here by Dr. Royce. The iterative aspects will be very familiar to modern developers who know of the 12 principles behind the Agile Manifesto, with lessons learned from implementation being fed back directly into the development of requirements and the design process. The incremental aspects will be familiar to anyone, for instance, that has heard of Sprints from Scrum.

The V-model is an ideal static model to apply these iterative and incremental dynamic models to.

The V-model helps to organise the derivation of interfaces as larger challenges are divided into traceable, manageable, and achievable work packages.

The V-model also provides a plan for building components into sub-systems, and for combining sub-systems to create reliable working systems to fulfill the customer requirements.

7. Conclusion

The V-model represents graphically; ownership of and relationships between information.

The V-model is a static model and does not restrict sequence of creation of artefacts.

The V-model supports iterative creation of a feature or capability, and also the incremental introduction of features or capabilities to a system

The V-model supports any number of levels, with a system being decomposed as many or as few times as is deemed useful.

The V-model is state-of-the-art.

A1 Literaturverzeichnis

MANAGING THE DEVELOPMENT OF LARGE SOFTWARE SYSTEMS
Dr. Winston W.Royce
https://www.cs.umd.edu/class/spring2003/cmsc838p/Process/waterfall.pdf

Manifesto for Agile Software Development
http://agilemanifesto.org/

Autor

Colin Hood hat seit 1977 die Evolution der Steuerungssysteme von relaisgestützten Systemen über programmierbare logische Controller (PLCs) bis hin zu modernen softwaregesteuerten Safety Critical Systemen begleitet. In verschiedenen Tätigkeitsfeldern war Colin Hood verantwortlich für Analyse, Design, Implementierung, Test und Auslieferung komplexer Softwaresysteme. Der Einsatz von Requirements Engineering war von jeher die Methodik, die seinen Erfolg bei Firmen wie Alcatel, BMW, DaimlerChrysler, Hella oder Miele begründet hat. Seine Spezialität ist neben der Verbesserung der Prozesse die Unterstützung des Veränderungsprozesses bei der Einführung neuer Methoden und Werkzeuge.

Kontakt

Colin Hood Systems Engineering
Dorfstr. 12, D-85253 Erdweg
Email: colin.hood@colinhood-se.com