The V Model

by Morton Hirschberg
Formerly of the Army Research Laboratory

The author is the technical project officer for the Data Exchange Agreement for Software Technology between the United States and Germany. It was in this capacity that he became aware of the German software standards, known as the V Model, for the German Federal Armed Forces. The standards are published in three volumes and can be tailored to fit officially sponsored work. In this Web Addition, the author introduces these standards to give readers a flavor for them and to encourage learning more about software standards used by a political and military ally.

It is not the author's intention to contrast the V Model with U.S. standards and directives, nor to comment about their use by the German Federal Armed Forces. The characterizations, such as strengths, are loosely quoted from summaries written by Herr Fritz Haertel, one of the architects of the V Model. They do not necessarily reflect the author's viewpoint.

The V Model

The V Model is a series of General Directives (250, 251, and 252) that prescribe or describe the procedures, methods to be applied, and the functional requirements for tools to be used in developing software systems for the German Federal Armed Forces.

General Directive 250. August 1992. Software Lifecycle Process Model.

The objective of this directive is to regulate the software development process by a uniform and binding set of activities and products that are required during software development and its accompanying activities. Use of the V Model helps to achieve:

1) Improvement and warranty of software.
2) Reduction of software costs for the entire life cycle.
3) Improvement of communications among the different parties as well as the reduction of the dependence of the customer upon the contractor.

The V Model deals with procedure, method, and tool requirements. Its main advantage is that it can be generally applied. It fits into the international picture by fulfilling requirements of NATO standards, ISO 9000 technical standards, and the structure of the EURO-METHOD. None of these is discussed here, but they could be featured in subsequent articles.

The V Model organizes all activities and products and describes activities and products at different levels of detail. In the V Model, products take on one of four states:

1. Planned: the initial state of all products
2. Processed: either in private development or under the control of the developer
3) Submitted: completed and now ready for quality assessment. It can be returnedto the processing stage if rejected or advance to accepted for release.
4) Accepted.

While seemingly prescriptive, the V Model allows for tailoring throughout the product life cycle. That is one of its strengths. The V Model is composed of four submodels: software development, quality assurance, configuration management, and project management.

The submodels are closely interconnected and mutually influence one another by exchange of products and results. The software development submodel develops the system or software. The quality assurance submodel submits requirements to the other submodels and test cases and criteria to assure the products and the compliance of standards. The configuration management submodel administers the generated products. The project management model plans, monitors, and informs the other submodels. Each can be further decomposed. For instance, the software development submodel can be broken down as follows:

- System requirements analysis and design.
- Data processing requirements analysis and design.
- Software requirements analysis.
- Preliminary design.
- Detailed design.
- Implementation.
- Software integration.
- Data processing integration.
- System integration.

The directive contains very detailed information (rules) on the activities of each submodel showing product flow, handling, and, if warranted, recommendations. For example, in the planning stage the product flow is from an external specification to a submitted state. Handling consists of organization planning, cost and scheduling, resource planning, and risk considerations.

There may also be peculiarities that need to be addressed. Recommendations that might be considered are:

- Use of matured resources and an experienced staff.
- Correct membership participation in cost and planning
- Scheduling.
- Consideration of alternative problem solutions.
- Knowing how to handle unexpected events
- Considering costs for management activities and coordination activities.

Occasionally, the directive also includes further explanations.

It should be noted that prototyping can be used to verify and detail requirements. Prototyping allows for early completion, as an aid in refining requirements, feasibility, and testing.

Each directive has a set of appendices containing definitions, a list of abbreviations, a list of illustrations, a bibliography, a characterization of the roles of the products, a list of activities to be performed, a list of products, an index and annexes. Directive 250 has two annexes.

The purpose of Annex 1 is to provide explanations of the application of the V Model. It is to support the user and is not of a binding nature. The objective of the V Model is to submit the products created during software development, maintenance, and modification to certain standards. This is to guarantee a minimum quality of the results and to make it easier to control the product stages from requirements definition to thefinal product itself. The V Model offers support as a guide, as a checklist, and for the quality goal definition. The V Model allows for tailoring, defines required products, and establishes criteria for assessment.

Two kinds of applications have been intended for the V Model—as a basis for contracts and as a development guide. The V Model makes provisions for the use of commercial, nondeveloped items, and commercial off-the-shelf software. It also provides for information technology projects.

Annex 1 also provides the elements that may be deleted from the model.

Annex 2 is an explanation of the products. It deals with reports and software to be produced. This includes requirements, architectures, and design. It covers user, diagnostic, and operator manuals. It also is broken down by the same four submodels.

General Directive 251. September 1993. Methods Standard.

The objective of this standard is to set down all the tasks and results ofthe software development process. Standardization is done on three levels: procedure, methods to be applied, and functional requirements on tools to be used. While the V Model answers what is to be done, the Methods Standard answers *how* it is to be done.

More than 30 basic methods or categories of methods are listed in the standard. These are, for example, bar charts, tree diagram, decision table techniques, E(ntity)/R(elationship) modeling, normalization, object design technique, simulation models, and structured design.

The Methods Standard includes allocation tables listing those basic methods that are best suited to realize certain activities or to develop products according to the latest state of the art and by observing the criteria of quality improvement, economy, and maintainability. For each method referenced in the allocation tables, the standard describes the features that an applied method must include to reach the standard. In many instances a complex method may be required. This may represent a well-defined combination of several basic methods. Basic methods really refer to procedures that describe a special, limited aspect of a system such as, functionality, data orientation, analysis, preliminary design, or one of the activities—quality assurance, configuration management, or program management. Complex methods usually cover several aspects of a system.

Basic methods must be applied unless, by limiting conditions, they make applying the method impractical, or there are arguments against the method or for an alternative method in a special case. Each method listed includes identification/definition, characteristics, limits, specification, interfaces, and a list of references. The Methods Standard is not meant to be a methods manual. Regarding tools, a method may be applied in different versions depending upon the manufacturers. For this reason, tool-independent definitions are set up.

Allocation tables exist for software development, quality assurance, configuration management, and project management.

The Methods Standard can be made modified by a Change Control Board that meets annually and is made up of industry and government. Besides the main part of the Method Standard, there are two annexes.

Annex 1 provides an explanation of the methods; the method interfaces including a characterization of the interface, an example of the interface, tool support, relevant literature, and a description of the methods. The methods explanation contains information about technical and operational software inspection and walkthroughs.

Annex 1 addresses object design technique and configuration management. It further contains a section on estimation models (function point method, constructive cost model), simulation models (continuous and discrete—time-controlled, event-driven, activity-oriented, process-oriented, or transaction-oriented), system behavior models (Petri networks, state charts, specification and description language),and reliability models (statistic [Okumoto, execution time, Logarithmic Poisson, Jelinski-Moranda, and Schick and Wolverton], and error seeding).

Annex 2 helps when applying complex methods in connection with the software development standard. This annex describes the most important methods for application in German national projects. The methods are:

1) Graphical Engineering System (GRAPES)
2) Information Engineering Method (IEM)
3) Integrated Software Technology (ISOTEC)
4) The quality management system of the CAP Gemini Group (PERFORM)
5) Structured Analysis and SA with Real Time Extensions (SA & SA/RT)
6) Specification and Design Language (SDL)
7) Software Engineering Technology (SEtec)
8) Structured Systems Analysis and Design Method(SSADM)

For each of the above, a brief description, tabular comparison with the basic methods, specification of the allocation, and relevant literature is given.

General Directive 252. September 1993. Functional Tool Requirements (Standardized Criteria Catalogue).

The goal of this standard is to constrain the variety of applied methods and tools that can be employed during the software life cycle. While theV Model answers what is to be done and the Methods Standard answers how it is to be done, the Functional Tool Requirements answers *with what* it is to be done.

The standard increases the guarantee for software quality (higher quality products, lower risk), minimizes software cost for the entire life cycle, imposes communication among the different parties, and reduces dependence of the customer on the contractor. The latter is accomplished through its recommendations, focused approach, and required prescriptions.

The standard introduces the software development environment (SDE) reference model where SDE is defined as “the totality of all technical resources utilizedfor the professional software

development." A tool is defined as "a software product supporting the generation or maintenance or modification of software."

The structure of the SDE reference model is:

- User interface.
- Work flow management.
- Security and integrity requirements.
- Software development.
- Quality assurance.
- Configuration management.
- Project management.
- Object management.

The description of the fundamental units or criteria—or service units, as they are referred to in the standard—are laid out as allocation to the V Model and Methods Standards, brief characteristics, and finally, requirements.

The reference model puts all the technical data processing services offered into a basic schema. Fifty-eight service units are defined and explained. A service unit can cover exactly one method or several methods. It should be noted that a method can be covered by one or more service units or not covered at all. In addition, there may be other requirements that are not based on a method. Finally, the Methods Standard may not suggest a method for the item under consideration. Some examples of service units are:

- From the user interface—help functions.
- From software development—generating databases, compiling, and debugging.
- From quality assurance—static assessment of the code.
- From project management—cost estimation.

An example of a service unit schema for cost estimation is allocation—planning, detailed planning, and estimation models; brief characteristics—requirements on tools to support the cost estimation realized by basis of already available empirical values from earlier projects, project specific marginal conditions, and by assumptions of future developments. For requirements—granularity, input and output interfaces to other service units, estimation models for fixed and variable costs, and other requirements such as an experience database.

The standard has an appendix and two annexes. An important part of the appendix is the relationship between the V Model and the European Computer Manufacturers Association (ECMA) Reference Model. The services in the ECMA Reference Model are:

- Object management.
- User interface.
- Process management.
- Policy enforcement.
- Communication.

Tools per se are not further specified in the ECMA Reference Model. There is no strict one-to-one correspondence between the V Model and the ECMA Reference Model.

Finally, Annex 1 supports the user in his work with functional tool requirements by means of tabular overviews and applications scenarios. The latter covers the definition of the functional requirements on a tool, the selection of tools for setting up a SDE, tool evaluation, and the tool environment in a customer/contractor relationship.

Annex 2 is a used as an introduction into the basics of SDE data management and to offer an overview of standards for the integration of tools with regard to data, control information, and user interface. Data management is handled through the use of data models. The real world is first portrayed in a conceptual design from which a logical design of relevant features is developed. Annex 2 provides definitions of a data dictionary, repository, and development data base.

Finally, the appendix deals with standards. Not all requirements can be met by a single tool, so a SDE is only possible if tools can be integrated into a uniform environment. Such integration has three aspects: data integration, control integration, and uniform user interface. The concentration is on data integration.

Several standardization efforts in the DP industry are discussed, including those of the Object Management Group.

Model Summary

The V Model, Methods Standard, and Tool Standard present complete coverage of the functional areas sSoftware development, quality assurance, configuration management, and project management), provide concrete support, is sophisticated, yet flexible and balanced, has a wide spectrum, and is publically controlled under the supervision of a Change Control Board. Improvements as well as corrective changes are handled through the Control Board.

The advantages are improved communications among project members, uniform procedures, guarantee of a better product, productivity increases, better choice of methods, adaptability, reduced risk, lowered training costs, anddecreased maintenance.

Conclusion

It is my hope that the models presented can serve as a catalyst and framework for discussion of standards methodologies for the Department of Defense. It should be noted that the German Ministry of Defense, while similar to the Department of Defense, is much more homogeneous. Perhaps this is a major contribution to the use of their V Model.

Contact Information

Morton Hirschberg
207 Briarcliff Lane
Bel Air, Md. 21014-5524
E-mail: mortfran@aol.com