Documentation Driven Software Development for Embedded Systems

Beate Muranko Rolf Drechsler
Institute of Computer Science
University of Bremen
28359 Bremen, Germany
Email: {bmuranko, drechsle}@informatik.uni-bremen.de

Abstract

The system architecture of embedded systems includes both, i.e. software and hardware components. Embedded systems are integrated in e.g. dvd-players, television sets, telephones, cars, airplanes, etc. They are a part of our everyday life. Embedded systems are developed using traditional software and hardware development models, which often leads to a deficiency in documentation, since this is not explicitly addressed in these models. However, the complexity of such systems and the fact that their components are often reused good documentation is not an option, it is a necessity.

In this paper we present an approach regarding the integration of the documentation workflow into a software development process, i.e. the V-model. This model is studied, since it also includes validation and verification aspects. By this, we provide an integrated development model that ensures high quality of the software development process for embedded systems.

1. Introduction

Embedded systems (system-on-a-chip) are used in nearly all technical systems e.g. dvd-player, televisions, telephones, cars, airplanes etc. Already in the year 1996 it was ascertained that an American person mingle with approximately 60 microprocessors in one day [1]. It is obvious that embedded systems play an important role in our everyday life. This type of systems can be characterized by containing software as well as hardware components. The design of such systems presents an enormous challenge, because

- modern circuits already consist of several hundred million transistors and
- in accordance to Moore's Law grow.

With the hardware also the software complexity grows. In the meantime – due to intensive use of caches and memory – the effort for software development for embedded systems is at least in the same range as for hardware. E.g. in [2] it is

reported that “software can account for 80% of the embedded-systems development cost”.

To keep these costs under control, it is important to make use of software development models. Several models have been proposed for “pure” software development. But in the context of embedded systems the focus is different. In embedded systems, which are often applied in safety critical applications, software bugs might cause serious harm. Consider e.g. the fatal consequences in areas like airbag design, brake-by-wire, etc. In order to guarantee the correct functionality in embedded systems and to prevent errors validation and verification are very important processes. But, “software is intrinsically harder to verify – it has more complex, dynamic data and an enormous state space” [2]. Furthermore, due to the complexity, reuse of components is very important. For such reusable parts a good thorough documentation is mandatory.

In summary, although verification, validation and technical documentation are very important for embedded systems, no software development model including all these aspects has been proposed so far. This is why, particularly for complex systems, finding a new development process is necessary, one that fully includes validation, verification and an elaborate documentation workflow.

Recently, in [3] the status quo of present technical documentation in soft- and hardware development models has been analyzed. The study delivered as a result that the importance of technical documentation has increased significantly over years but it is treated insufficiently. With the integration of the documentation workflow into the Waterfall model a possible approach was presented in [3]. Even though the Waterfall model is well known and easy to understand, it is not considered state-of-the-art in software development, since aspects of validation and verification are not integrated. Especially for the application domains discussed above this is mandatory.

On the other hand, there exist modern software development models, like e.g. the V-model [4] (see Figure 1) that also take validation and verification into account. But in this case, the aspect of technical documentation is not discussed.

In this paper we study the V-model which we analyze with respect to an integration of technical documentation. The result of this paper shows our approach for such a software development model for embedded systems.

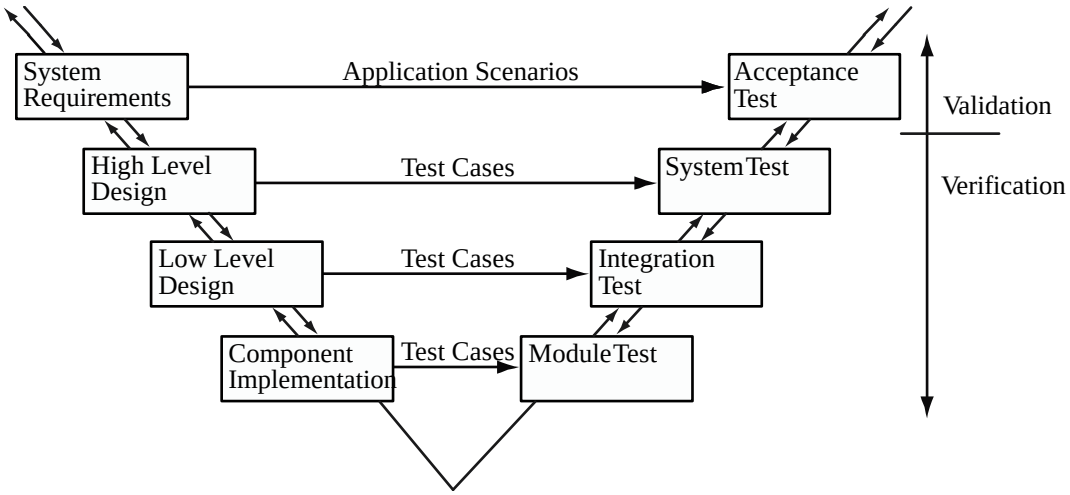

Fig. 1: V-model

The paper is structured as follows: Section 2 reviews previous work. To make the paper self-contained, we also provide a brief introduction to technical documentation and discuss the relevant aspects of the V-model. In Section 3 a workflow of a documentation process is presented. Section 4 describes the approach for the integration of the documentation workflow into the V-model. Finally, a summary of the results is given and future work is discussed.

2. Related Work and Preliminaries

Recently, in [3] technical documentation has been discussed in the context of embedded systems. In the following we summarize the key aspects presented there: Documentation is composed of technical documentation (all necessary information which are important for the product and its use), internal documentation (all information and instructions which are essential for internal usage), and external documentation (all instructions and user information about the product which are delivered to the customer). Beyond this, there is a further subdivision in documentation areas. The areas are: project documentation, development documentation, product documentation and user documentation. Only the technical documentation differs; it lacks the project documentation. This short framework provides an understanding of the term technical documentation.

Furthermore an analysis of the integration of technical documentation into currently established soft- and hardware development models in [3] showed that documentation is treated insufficiently and superficially. As a first step the integration of a documentation flow into the Waterfall model was discussed and presented. But the Waterfall model does not include a verification phase, which is very important for the development of embedded systems (see Section 1).

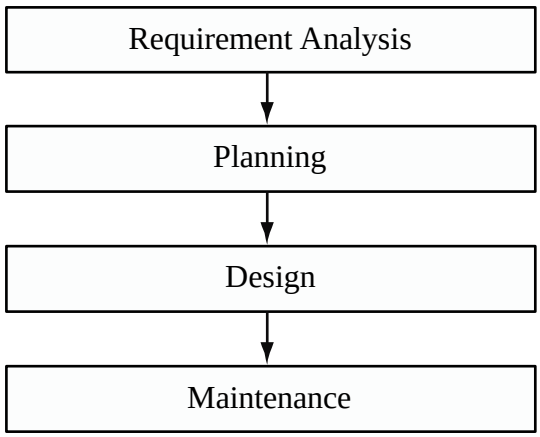

Fig. 2: Documentation workflow

Therefore in our approach we evaluate an integration of the documentation flow in the context of the V-model, where validation and verification is explicitly included.

For completeness, in the following the V-model is briefly discussed (for more details see [4]): The V-model includes aspects of quality assurance. In detail, it includes validation and verification concepts. The term verification means checking the implementation of the system against its specification. If they match, the system is correct. In this context, validation means the testing for correctness of the system. The system should cope in an adequate manner with the issue it was intended to solve.

3. Documentation Workflow

In this section we give a brief overview of the documentation workflow which is derived from practical experience [5], [6], [7] (see Figure 2). It is structured into four phases which are described below:

- **Requirement Analysis**

  Here, the system requirements have to be analyzed. In order to decide the level of detail for the instruction material the target group (e.g. beginner or expert) has to be determined. On the basis of the target group the technical depth of the documentation is determined. In this phase, the authors of the documentation familiarize themselves with the product. Furthermore the resulting draft has to be coordinated with all persons or departments involved and it has to be subsequently approved. This way, misunderstandings are eliminated early on.

- **Planning**

  The planning phase is supposed to yield a structured version of the draft from the previous phase. The responsibilities are divided among the documentation team: the layout, the level of detail and the rough structure are specified. Additionally the product is checked for compliance with the specific legal standards. This is also the phase where the rest of the documentation workflow is planned.

- **Design**

  In this phase the actual documentation is compiled in three consecutive versions: the alpha, the beta and the final version. These versions are created concurrent to the corresponding implementation phase. Any revisions by the staff members are transferred to the next version.

- **Maintenance**

  Once the documentation is in use, insufficient or incorrect information is reported which must then be revised. The care and update of the documentation must be done on a continuous basis.

In the following we study how this documentation flow can be integrated in the software development model.

4. Integrated Approach

In order to integrate documentation into the development model we first compare the documentation workflow presented in the previous section with the V-model. Then the V-model is extended by establishing connections between the respective documentation phases (see Figure 3). This shows explicitly where documentation needs to be integrated into the V-model.

On the left hand side of Figure 3 the documentation workflow is illustrated. It has been extended by feedback connections between its phases. Thus, connections allow that interaction can take place. Each individual phase is connected by dashed arrows with its predecessor. An example for such a feedback action is to go back from “Design phase” to “Planning phase”.

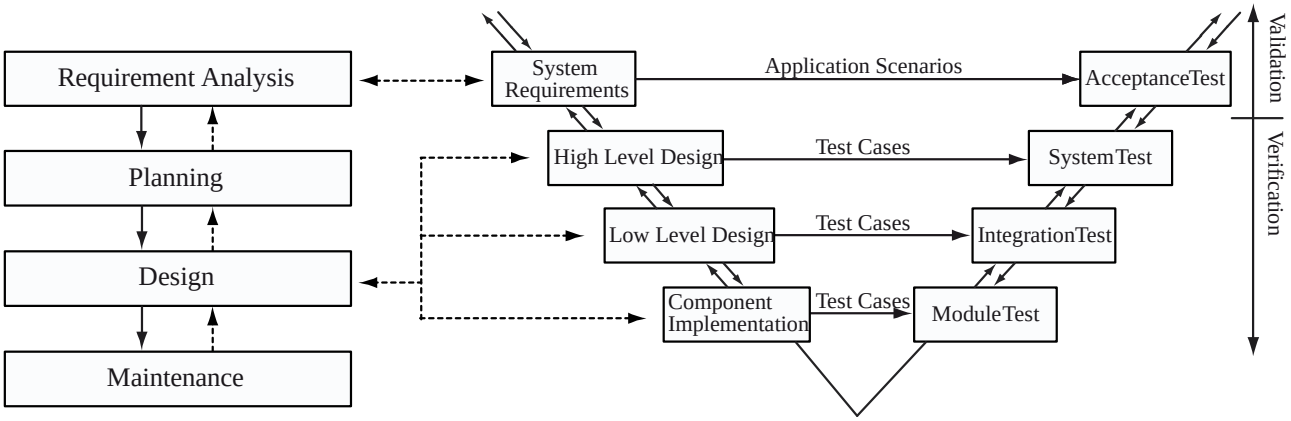

Fig. 3: Initial model

- **System Requirement**

  In the system requirement phase the system is specified and application scenarios for the final acceptance test are developed. It is important to record documentation requirements at the same time one decides on the specification of the system. At this point it is also necessary to establish the target group of the manual, thereby also determining the level of detail for the technical descriptions. Furthermore, at this stage the publishing venues for the documentation are also defined (print, web, etc.).

- **High Level Design and Low Level Design**

  The phases “High Level Design” and “Low Level Design” deal with an architectural view of the system. Here, the specification of subsystems is accomplished and the corresponding system architecture is constructed. Testing is also culminating in a final system test at the end of the phase.

  In this phase it is imperative that documenting is even more tightly coordinated with the actual implementation, since the individual documentation versions (Alpha-, Beta- and Final Version) need to match the corresponding stages of the product as perfectly as possible. This way updates in the level design can be incorporated directly into the documentation, thereby correctly reflecting the status quo.

- **Component Implementation**

  In the Component Implementation phase the individual components are specified, implemented and tested (Module Test). If changes take place in this phase, they also should be transferred directly in the documentation. Especially

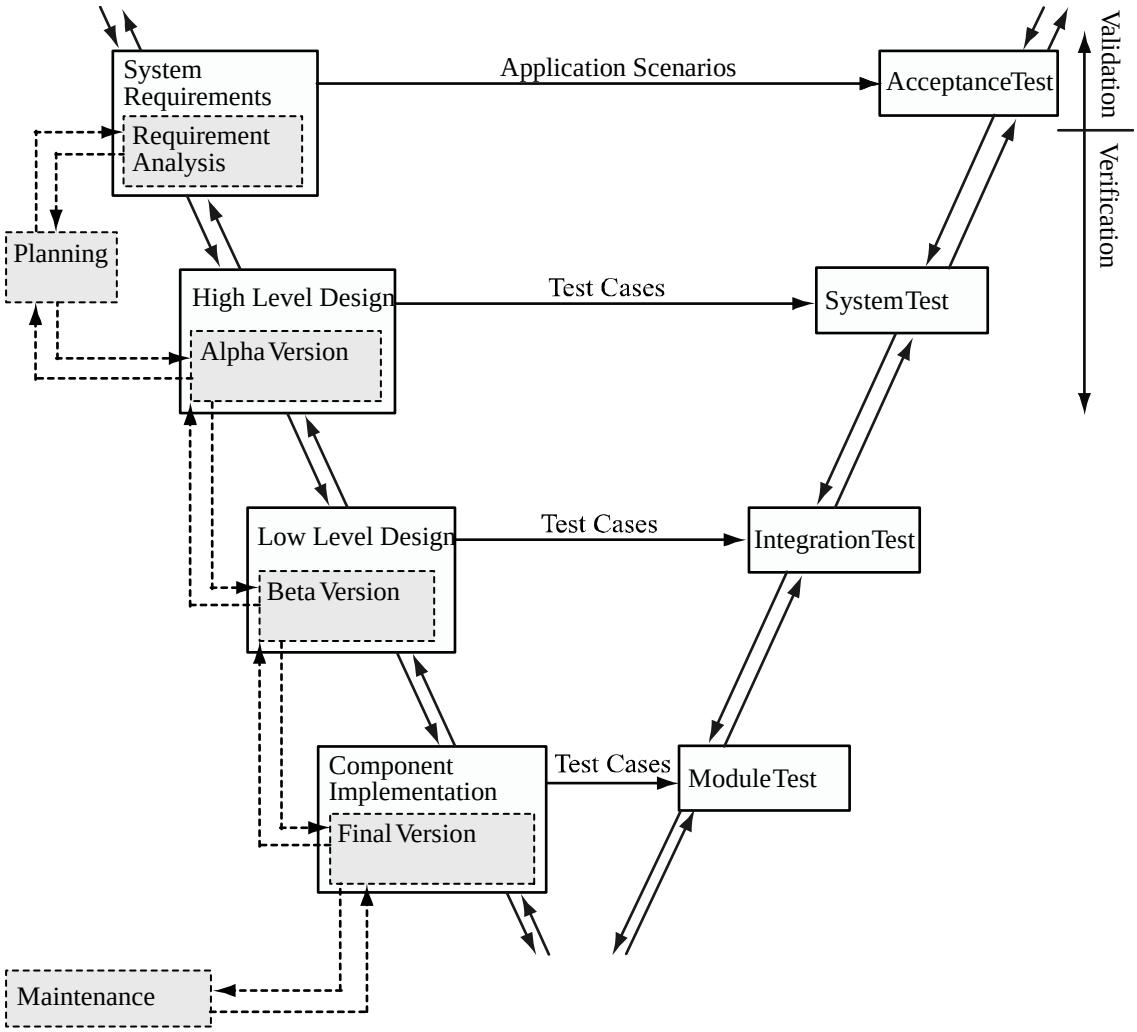

Fig. 4: Merged model

in this phase it is more important than ever that the documentation reflects precisely the state of the product.

Figure 4 displays the merged result as a new model. Obviously not all phases of the resulting compound model contain documentation. Figure 5 displays only those phases which contain documentation, validation and verification.

High Level Design and Low Level Design both deal with design and are therefore merged into a single phase which we label Design (see Figure 6).

Integrating the documentation demands for an extension of the development model by a phase which otherwise would not have been considered. The operation phase is not part of the actual development process. Therefore we add this to the model (see Figure 7). We include the important maintenance phase of documentation design into the model. This extension complicates the model. It also emphasizes the fact that the

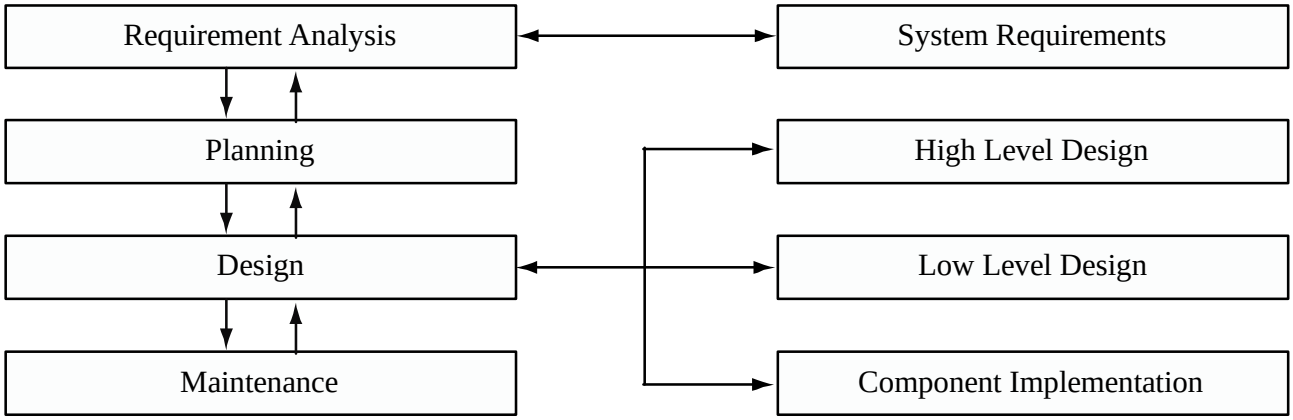

Fig. 5: Universal model

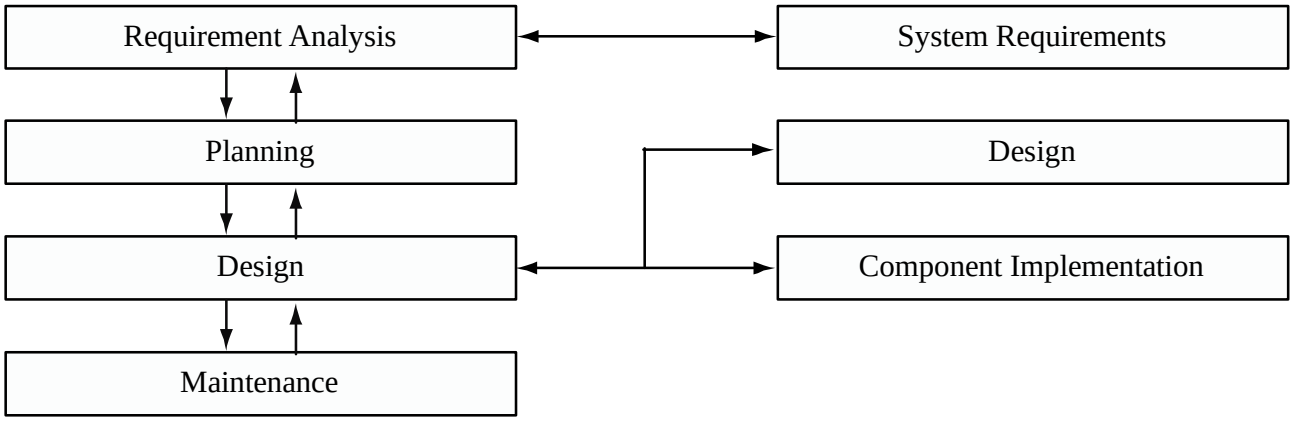

Fig. 6: Universal model with single design phase

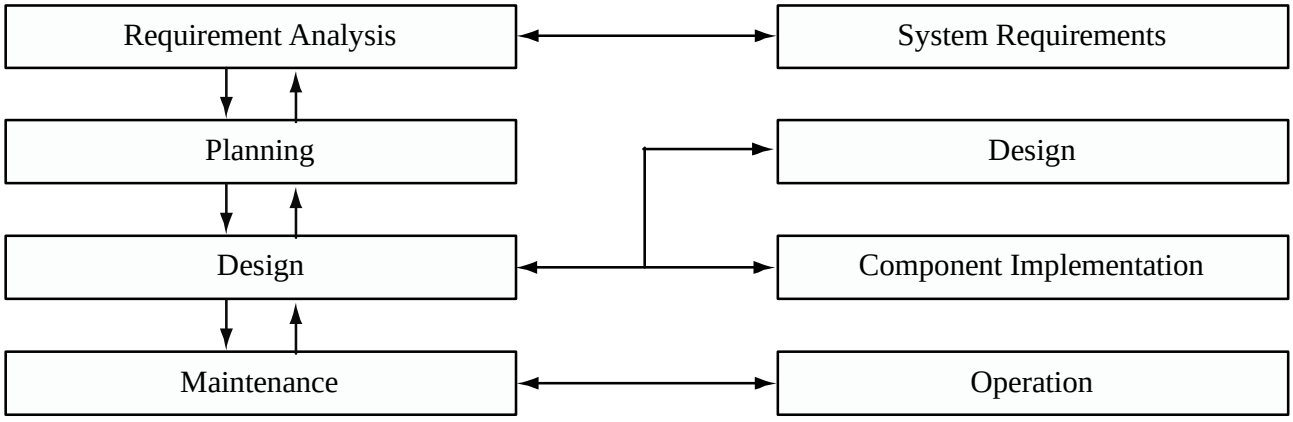

Fig. 7: Universal model with additional operation phase

developers' involvement with a product does not end with its release into the market but rather continues for the whole of its life cycle.

In a direct comparison of the universal model (see Figure 7) to the universal model of the Waterfall model in [3] a significant similarity is identifiable. Consequently both models have identical documentation development workflows. But as a major advantage in contrast to the Waterfall model, the software development model considered in our approach also includes validation and verification phases.

5. Conclusion

In this paper we have presented a framework for including technical documentation in a software development model. This framework ranged from a documentation workflow which is derived from practical experience to the V-model. Both workflows/models are particularly useful for embedded systems with their high demands for reusability of components and quality ensured by validation and verification approaches. A complete integration of the documentation into the V-model has been presented. It is focus of current work to discuss the concepts developed in this paper in the context of the newly developed V-model XT [8].

References

[1] P. Marwedel, Embedded System Design. Kluwer, 2003.

[2] The International Technology Roadmap for Semiconductors, Edition design. http://www.itrs.net/Links/2005ITRS/Design2005.pdf, 2005.

[3] B. Muranko and R. Drechsler, Technical Documentation of Software and Hardware in Embedded Systems. IFIP VLSI-SOC, 2006, page 261-266.

[4] IABG Information Technology, V-Model: Lifecycle Process Model. www.v-modell.iabg.de/kurzb/vm/k_vm_e.doc, 1993.

[5] T. Barker, Writing Software Documentation: A Task-Oriented Approach. Longman, 2003.

[6] A. Sikora and R. Drechsler, Software-Engineering und Hardware-Design: Eine systematische Einführung. Hanser, 2002.

[7] J. Price and H. Korman, How to Communicate Technical Information: A Handbook of Software and Hardware Documentation. Addison-Wesley, 1993.

[8] Bundesrepublik Deutschland, V-Model XT: Part 1: Fundamentals of the V-Modell. http://v-modell.iabg.de/index.php?option=com_docman&task=doc_download&gid=32, 2004.