Development models

R. Kuiper and E.J. Luit

1 Introduction

We reconsider the classical development models: the Waterfall Model [Bo76], the V-Model [Ro86], the Spiral Model [Bo88], together with the further assessment of the Spiral Model in [Bo96], and the Rational Unified Process [JBR99, Kr00]. All these models concern grouping of development activities and the flows of information and time.

Two observations are made. Firstly, time and information flow in these models are not always well separated. This is unfortunate, as these flows sometimes do and sometimes do not coincide and it is often important to know explicitly whether time or information flow is intended. Secondly, development activities are grouped together in various ways that are not always explicitly motivated. This is unfortunate, because it means the aim of certain groupings is not clear.

This analysis results in two proposals. Firstly, we separate the information flow and the time ordering. Secondly, we explicitly and systematically argue the grouping of activities. Although many activities and information flows are common to different development processes, there are also many differences, depending on the context. For example, hardware-software co-development requires additional activities over pure software development. Therefore, we define a development framework and give an example of a possible choice of activities and information flows. Likewise, we do not suggest a particular process, as this again depends on the context. Certain projects can be done best in a waterfall-like manner, whereas an iterative approach is more suited to others. For example, for a small project in a well-known domain, a waterfall-like process may be most appropriate, whereas an iterative process may be more appropriate for a large project.

We order activities in a V, following the traditional V models. This ordering represents only information dependencies between the activities. The left leg of the V corresponds to decomposition activities, whereas the right leg corresponds to composition activities. The V shape is convenient because this way, the flow of information between the legs of the V can be depicted concisely. We introduce several Vs to accommodate different roles so that information flows and dependencies between different roles can be explicitly visualized. The timing is treated separately, as are various kinds of process, e.g., top-down or bottom-up. Sequencing of activities is more conveniently visualized with, e.g., Gantt-charts.

To use this framework of Vs, the appropriate activities and their groupings must be selected first. Then, it must be considered which information flows and artifacts are needed. Finally, the time sequencing of different activities must be selected, i.e., the process must be defined.

2 Analyzing development models

Considering the classical development models, the most established are the Waterfall Model, the V-Model, the Spiral Model and the RUP.

2.1 The Waterfall Model

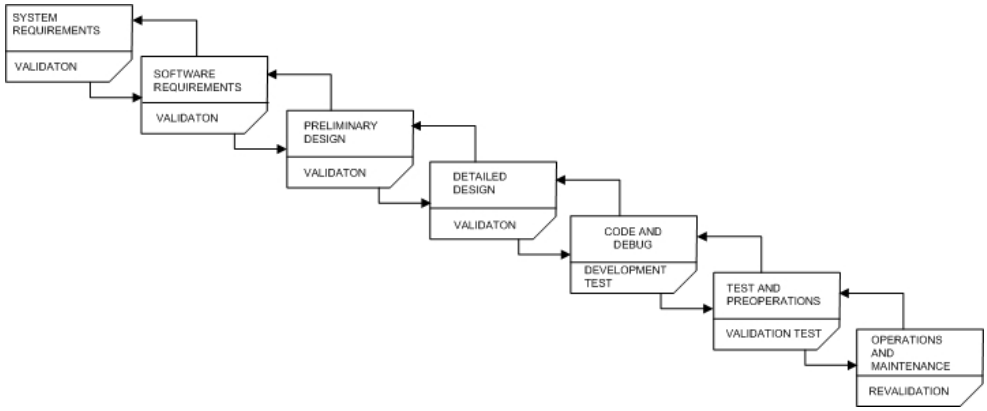


Figure 1 Boehm's waterfall model

The Waterfall Model, [Bo76], provides the following information.

1. The development phases in the software life cycle are ordered in time: a linear ordering of, roughly, requirements analysis, design, coding, testing and maintenance; these are depicted as boxes. Boehm actually uses slightly different terminology. We use verbs to name these phases, which indicates that activities are associated with the boxes. Transitions from one phase to another are drawn as arrows; the arrows express information flow and time ordering.

2. Validation occurs in each phase: the output of a development phase is validated against the input of that phase. In the Waterfall Model, validation is included as a box at the bottom of the phase-box. This suggests that validation occurs at the end of the activities in a box.

3. As later phases might lead to reconsidering earlier ones, there are feedback arrows.

Boxes identify a development phase and arrows indicate the time order. Implicitly, there is also information flow between phases: activities in a phase use information from the previous phase. A problem is that because of the feedback arrows, the time order and the information flow are quite complex. Especially for validation activities, neither the timing nor the information flow is very detailed or explicit. This suggests a first idea, namely to separate time and information flow.

2.2 The V Model

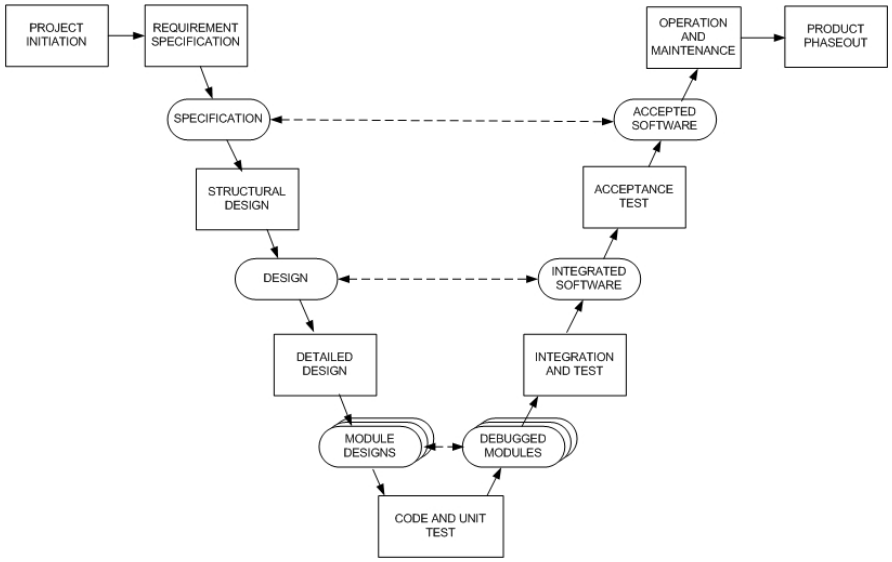


Figure 2 Rook's V-Model

The V-Model, [Ro86], again uses boxes to depict development phases. In addition, the main deliverables of the characteristic activities of the phases are depicted by ovals in between the boxes. Rook emphasizes that no strict time flow is intended by the arrows and that, in fact, iterations between the activities are common practice.

The V-Model recognizes that testing consists of various kinds of testing, like module and integration testing, and that information used stems not only from the coding phase, but also from earlier development phases. To indicate this information flow, the V-model folds at the coding phase, thus bringing the corresponding phases opposite one another. Arrows indicate the relations between the deliverables on both sides of the V. In fact, this is the information flow from the development activities to the corresponding test activities. In later versions of the V-Model [], the deliverables are often left out as shown in figure 3. Again, the arrows show the (time and?) information flow from the development activities to the corresponding test activities.

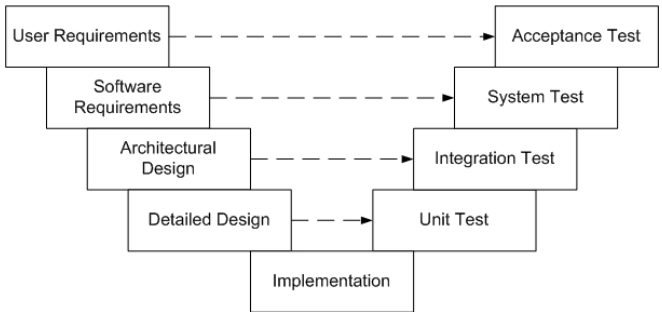


Figure 3 Other representation of the V-Model

The additional detail that the V-Model provides shows a problem. Even though Rook depicts the main deliverables, most other V-Models do not. It is not clear what the artifacts used and produced in the various phases are; are these test specifications, test implementations, executions of tests against implementations, test outcomes? In one V, it is difficult to accommodate all the artifacts. A solution to this problem is indicated by various attempts to split the V into several Vs. Activities that have a similar nature are grouped together in a separate V. Also V-Models with sub-models have been developed, an example is [IABG95]. Cockburn [Co95, Co97] groups management activities into a separate V. This leads to the second idea: group activities not only into Vs but also split these Vs into multiple Vs.

Other problems are that the V-Model suggests that the activities are executed sequentially and that no testing activities take place in early stages of a project. Even though a sequential execution of the activities was not intended by the original V-Model, it is often interpreted this way. In fact, Rook presents a figure with estimated effort for different development teams that is very similar to figures used in RUP publications. This leads to another idea, namely to provide an explicit and different representation of the sequencing of activities. Also, with a separate Test-V, the possibility for early testing activities can be made explicit.

2.3 The Spiral Model

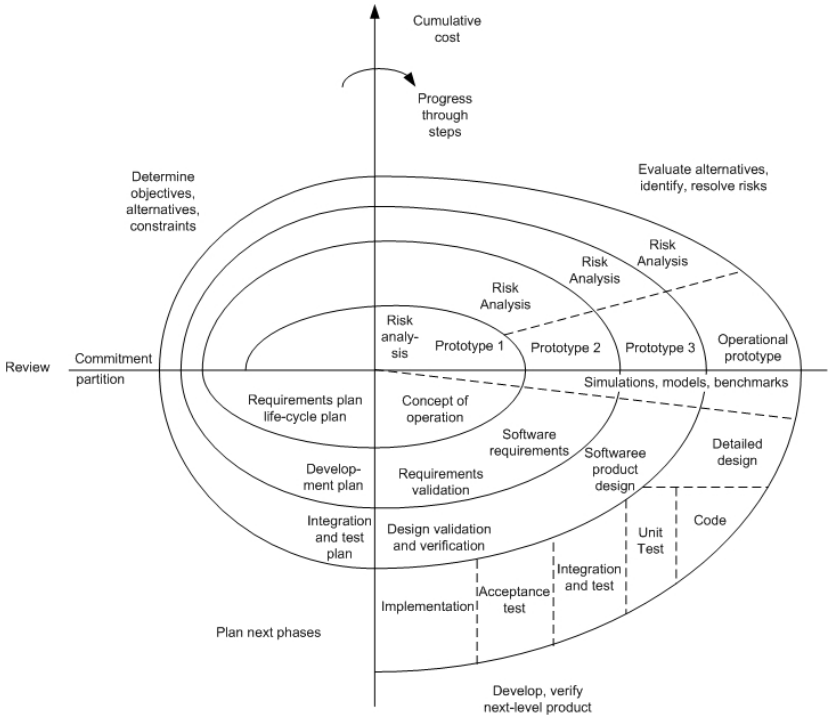


Figure 4 The Spiral Model

The Spiral Model [Bo88], addresses the problem that both Waterfall and V-Model suffer from an unrealistic modeling of timing of phases. It recognizes that development is often iterative and graphically depicts this by providing multiple instances of the phases from these models and ordering these as a spiral. Thus, a more realistic representation of timing is provided. The spiral model can then be viewed as one of the possible timing models for the Waterfall Model.

The drawback of detailing time is that now the complexity of the mix of information and time flow becomes even larger than before: information flow arrows would be duplicated wherever phase duplication occurs because of the time detailing.

2.4 The Rational Unified Process (RUP)

RUP groups activities by defining the following workflows: Business Modeling, Analysis and Design, Implementation, Testing, Deployment, Configuration and Change Management, Project Management and Environment. Workflows are similar to (a group of) boxes in the V-Model. RUP

explicitly does not impose an ordering in time of the workflows.

RUP places activities in the workflows but does not group activities together like in the boxes of the Waterfall and V-Models. This we consider an omission, as this level of abstraction is in fact the most relevant to depict artifact stability and information flow. In RUP, the dependencies between activities of a workflow are visualized with a kind of Activity Diagrams. …..

The timing or process part is described separately in RUP; timing is given as steps: one step is a full iteration through all workflows. The time order of the workflows is not indicated, but the amount of time spend on the workflows per iteration is described. Time order is indicated between activities.

The iterations are grouped in phases: Inception, Elaboration, Construction and Transition.

(Further detail is, that RUP may use different perspectives on the workflows, e.g., Technical and Managerial; the perspectives synchronize on deliverables. The synchronization is more driven by the availability of the results from the two perspectives than by timing; i.e., the information flow rather than the time flow is decisive. Deliverables are Conceptual Prototype, Architectural Prototype, Architectural Baseline, Releases and Deliveries.)

One traversal of these phases, a cycle, produces one software generation. A new traversal of the cycle produces a new generation (in RUP this, somewhat confusingly, is also called a phase: Evolution).

2.5 Conclusions from the analysis and some new ideas

Idea 1: Separate time and information flow. Therefore, define an information flow model and, for this model, a timing (process) model.

Idea 2: Use activities as basic ingredients, group these into Tasks, partly driven by what belongs together by nature (groups of workers, development of one kind of object, what allows best picturing of information flow, ...).

Idea 3: Use Vs to order groups of activities to represent information flow. Our motivation for the V-shape is, that an important (information flow rather than timing) ordering on the workflows (in the various perspectives) is, whether the activity is decompositional or compositional in nature. Corresponding decomposition/composition steps need information transfer; this is easy to depict when they are at opposite positions in the V.

Note, that the Vs are quite close to the RUP workflows but that we consider the construction activities to belong together and therefore group them into a Realization V.

Furthermore, there are some ideas from other approaches that we want to employ.

To make explicit what is used and produced in these activities, input and output artifacts are defined for each phase. These come both form previous phases as well as from the outside. The latter in, at least, two flavors: extra detail like design decisions and extra constraints that do not follow from previous phases but are fixed environment constraints. An example of the first is

deciding to use a certain design pattern, an example of the second is to use a certain platform because it is the company standard. Evidently, the two may influence one another.

As to timing (process), we observe that to provide assessment points during development, anchor points are used: in [Bo96] anchor points for evaluation are defined that occur at the unfolding spiral. These anchor points are not located at fixed points of each cycle but are milestones in the full lifecycle: life-cycle objectives, life-cycle architecture and initial operational capability. RUP employs a similar device in grouping iterations (comparable to one traversal in a spiral model) into phases. The boundaries of the phases then correspond to the anchor points.

3 The multiple V-model

3.1 The multiple V-model for information flow

We outline "the next" model. We present the model as a template of ingredients where specific concrete ingredients can be chosen depending on the situation at hand.

The entities in the information flow model are activities (that transform artifacts). The relationships are information flow arrows that represent the input and output information between (groups of) activities. This relation can occur at various levels: artifacts or groups of artifacts.

3.1.1 Entities

1. At the lowest level of detail the following entities are identified.

An activity: the basic unit of work.

Examples are: user requirements elicitation, deciding on requirement attributes, implementing tests, establishing connections for tracing, applying a metric to code and handling of certain change requests.

Input for an activity is a set of artifacts (documents, code, tests, ...)
Output from an activity is a set of artifacts.

We take the activities as the leading entity for grouping.

The activities are then grouped in various ways.

2. A Task: a group of activities.
- A Task groups a set of activities that are closely related and that cooperate to produce a common set of output artifacts. Information flow between activities within a box is informal. For example, the activities static and dynamic problem modeling and documenting these models are executed iteratively until the model and documentation are complete. The common input artifacts for these activities are requirements and use cases; the output artifact is the documentation of the problem analysis.

The result of these activities at the end of the development process or of, e.g., a RUP phase is a set

of baselined output artifacts. At such moments, the information dependencies dictate the consistency of dependent artifacts with the artifacts that these depend on. At the end of the development process, it is impossible to distinguish what kind of process has been executed to produce the artifacts: it could have been a Waterfall or a RUP process.

Input for a Task is a set of artifacts (namely, the ones needed for the activities of that phase).
Output for a Task is a set of artifacts (namely, the ones produced by the activities of that phase).

Examples of Tasks are: requirements engineering, integration testing, or requirements management for a particular Task.

NB The activities within a Task may well have an iterative character. This is not indicated at the information flow level of the multiple V-Model, but is described at the Process Model level.

3. A V: groups Tasks by nature.

Our motivation for the V-shape is, that an important (information flow rather than timing) ordering on the Tasks (in the various perspectives) is, where the Task occurs in the course of decompositional (downward left in the V) or compositional (upward right in the V) development. Corresponding decomposition/composition steps need information transfer; this is easy to depict when they are at opposite positions in the V. As mentioned above, we group several of RUP's workflows into a single Realization V. Given the dependencies, the information flows between these workflows seem adequately modeled in a single V.

4. A set of Vs: gives total development

Because the development tasks differ in nature, several Vs are used. For example, a realization-V, a Test-V, a Requirements management-V, a Project Management-V, a Configuration and Change Management-V. Note that also information flow arrows between the Vs occur. For example, effort registration from each Task to the Project Management-V.

3.1.2 Relationships (in progress)

Input for an activity is a set of artifacts (documents, code, tests, ...)
Output from an activity is a set of artifacts.

This information can come from or be used by various activities and therefore flows between phases and between Vs.

We take the activities as the leading entity for information flow, but further detail which artifacts are used.

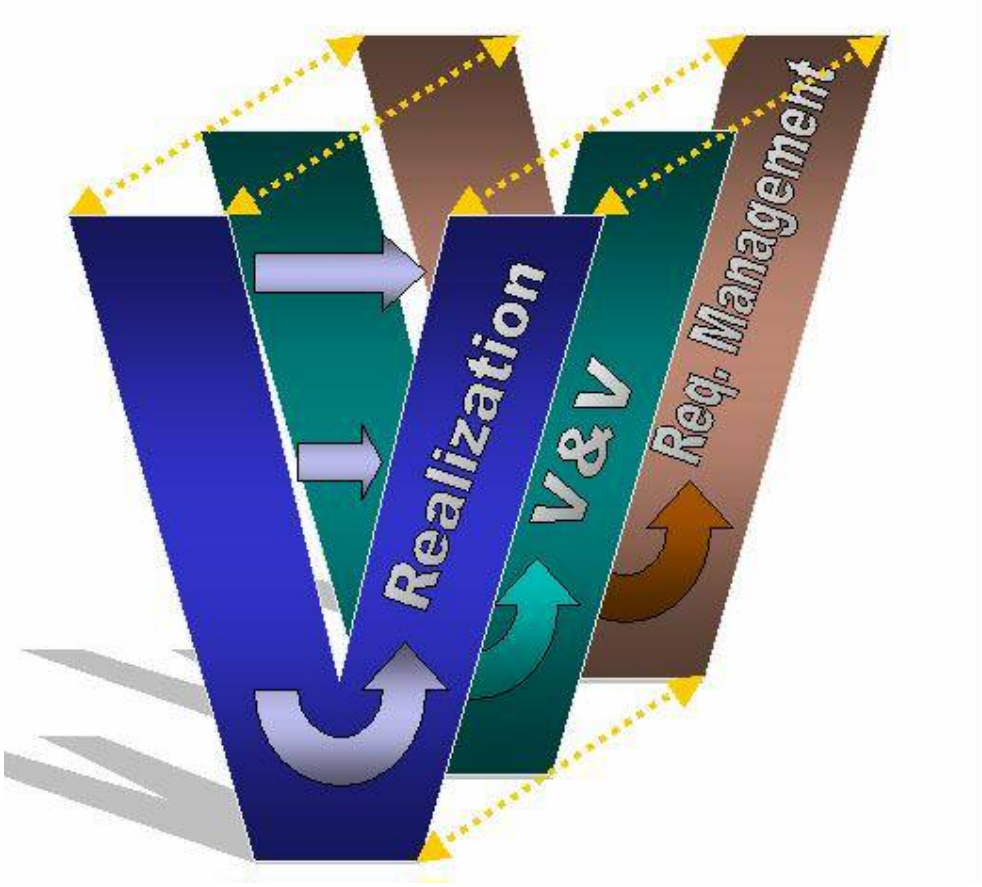
Relationship arrows indicate information flow; there are three (relevant) levels of detail at which arrows can connect entities:
- between Artifacts;
- between Activities;
- between Tasks.

This leads to arrows at different levels of detailing:
- at the task level, an arrow between tasks indicates that output from one phase is used as input for the other task.

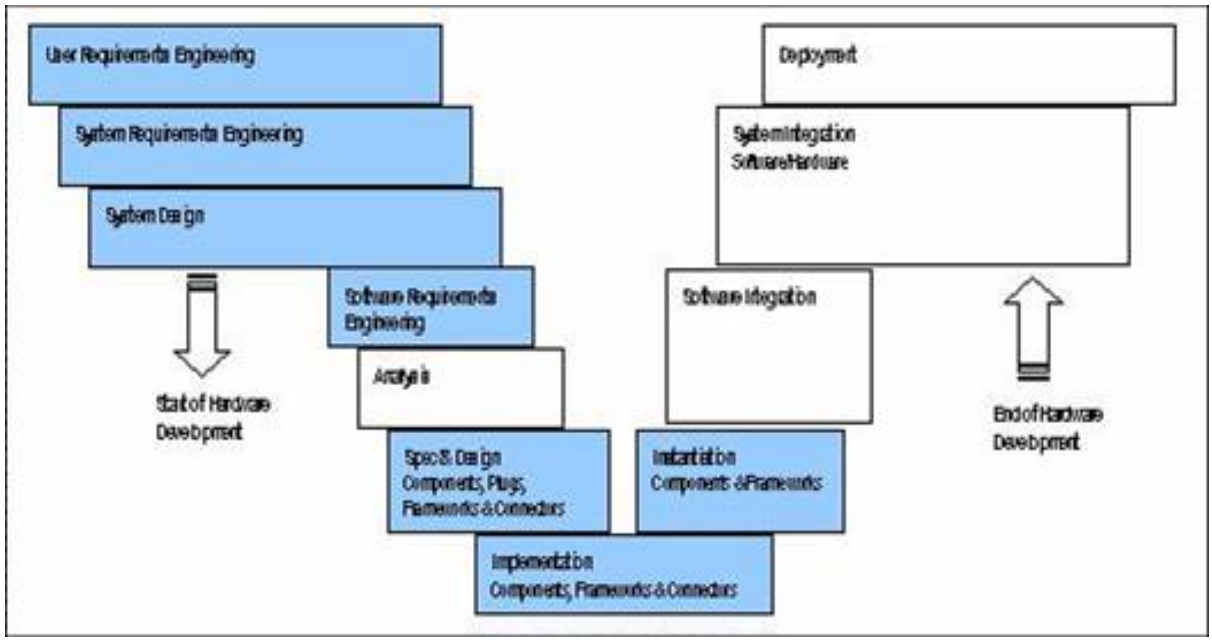
- at the activities level, an arrow between activities means that the output of one activity is used as input for the other activity.
- at the artifact level, an arrow indicates that an artifact is used as input for an activity. It might even be possible in certain cases to identify that an input artifact is relevant for certain but not all output artifacts of an activity.

Note, that for the management of change impact it makes a difference at which level the information flow arrows are available.

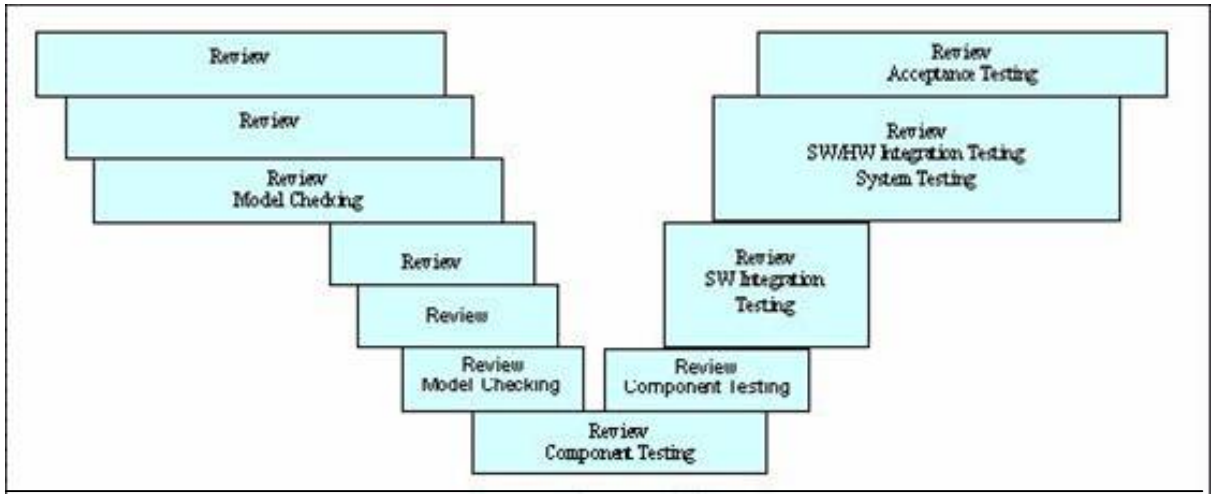
NB It is quite possible to choose the activities, tasks, Vs and relationships in any order.



Multiple V



Realization-V



Verification and Validation-V

3.2 Process (in progress)

The timing is treated separately, as various kinds of process, e.g., top-down, bottom-up, spiral, extending the multiple V model with time arrows.

NB Many issues to be considered:
- what is the difference between a spiral model and several iterations through a waterfall? maybe that in the spiral case there is info flow from the same phases on different rings of the spiral. This is interesting, as it is information flow that cannot be depicted in the model without time, as there are no rings there.

Processes

Tooling

At least for separate Vs, but more challenging:
1. To join Vs
2. To handle also process (i.e., timing)

4 Conclusions

References

[Bo76] Boehm, B.W., Software Engineering, IEEE Transactions on Computers, C-25, pp 1226-1241, 1976.
[Bo88] Boehm, B.W., A Spiral Model of Software Development, and Enhancement, Computer, (21,5) pp 61-88, 1988.
[Bo96] Boehm, B.W., Anchoring the Software Process, IEEE Software, July 1996, pp 73-82, 1996.
[Br86] Rook, P., Controlling software projects, Software Engineering Journal, January 1986, pp 7-16, 1986.
[Co95] Cockburn, A., "Unraveling Incremental Development", Object Magazine, Jan. 1995, pp. 49-51
[Co97] Using "V-W" Staging to Clarify Spiral Development, http://alistair.cockburn.us/crystal/articles/vws/vwstaging.html
[IABG95] http://www.v-modell.iabg.de/ (start page), http://www.v-modell.iabg.de/kurzb/vm/b-vm.doc (process model).
[JBR99] Jacobson, I., Booch, G., Rumbaugh, J. The Unified Software Development Process, Addison-Wesley, 1999.
[Kr00] Kruchten, P., The Rational Unified Process: An Introduction (2nd Edition), Addison-Wesley, 2000.
[!!] ITEA site
[??] some reference to more-Vs model.

1. This is for component DEVELOPMENT, now change also: take care of
   What is changing and what is touched.

2. How to match this with the ideas of IEESE and Stefan?

3. How to use Deifel in here?

4. How to use Antje von Knethen's ideas?

5. How to use TUE's incremental verification?

6. How to use Selliers' ideas?

7. What about FOCUS?

8. What about ISpec?

UNUSED IDEAS

NB Subdivision of workflow Vs into phases: according to granularity of artifacts; also information flow is decisive.

NB Traversal of cycle should also be possible for obtaining iteratively 1 generation.

NB Idea of emphasis on cycle phases is close to customizable cycle, but also gives management info about resources needed.

OLD STUFF

1.

The multiple V-model

The following is based on results from the ITEA-DESS project [!!]

From the above observations it follows that in fact three things are modeled:

I. The development activities and their grouping
.....

This regards information transformation.

1. activities - smallest unit of development activity

2. phases - groups of activities; grouped according to I/O artifacts that address same entity ***is

that so, and should I/O be only for phases or also for activities?***

3. workflows - V-shaped groupings of phases; (ordered according to information flow ***maybe under II?***), grouped according to same nature
NB Depending on situation, various choices for the Vs can be made. Basic: realization V, Validation V, Test (or Verification ***you were right about the naming conventions, Erik, I looked it up in the IEEE glossary - verification is testing, blurf...***), Requirement Management V. Less basic: Process management V, Quality management V, C***?*** and Change Management V.

II. Information flow between phases in workflows
.....

This regards information dependency. NB Arrows go from output to input, selecting from a set of output artifacts the artifact that needed in a set of input artifacts: many arrows may go in.
***NB Don't forget to mention somewhere that feedback is considered a different kind of information flow than during development: no clear I/O artifacts,
treated as only occurring during instable situation***

III. Time order between development activities.
.....

This is process: for a given V add a separate timing schedule. See below.

We propose to separate information flow and timing.

This leads to modeling 1 and 2 together, and separately model timing of the various phases as various processes: bottom up, top down, middle out, spiral.

1. a model, the triple V-model for the information flow between phases of workflows.

2. a separate description, in terms of the (workflows and?) phases of the triple V model of the timing.

*** I am not convinced that all perspectives are Vs: for example, project management or versioning may well have another relation to some set of Vs
rather than being a V itself.

To be considered: what happened to the classical notions of decomposition and abstraction?
decomposition: split of activities inside phases over components, maybe orthogonal part of Vs? Or organized by RM?
abstraction?
maybe present in how detailed one looks at the artifacts in the Vs? And how does that relate to the information flow? maybe something like a view - different from a perspective (perspective is a certain area of work, a view is than the level of abstraction chosen to work in that perspective.)