Viewpoints for requirements elicitation: a practical approach

I. Sommerville, P. Sawyer and S. Viller
Computing Department, Lancaster University, Lancaster, LA1 4YR, UK
Tel: +44 1524 593780; Fax: +44 1524 593608
[is, sawyer, viller]@comp.lancs.ac.uk

Abstract

This paper introduces an approach to multi-perspective requirements engineering (PREview) which has been designed for industrial use and discusses our practical experience in applying PREview. We have developed a flexible model of viewpoints and, using examples from an industrial application, show how this can be used to organise system requirements derived from radically different sources. We show how 'concerns', which are key business drivers of the requirements elicitation process, may be used to elicit and validate system requirements. They are decomposed into questions which must be answered by system stakeholders. We briefly describe the process of using PREview which has been designed to allow incremental requirements elicitation. Finally, we discuss some practical considerations which emerged when the approach was applied in industry.

Keywords: viewpoints, requirements elicitation, multi-perspective specification
Category: Research and experience

1. Introduction

It is now widely accepted that a multi-perspective approach to requirements engineering can, potentially, lead to requirements specifications which are more likely to satisfy the needs of a diverse set of system stakeholders. Good requirements engineers will always consider these different perspectives but viewpoint-oriented requirements engineering methods which *explicitly* identify and separate different system viewpoints [1-3] are not widely used in practice.

One reason for this is that, in our experience, most viewpoint-oriented methods do not cope well with the messy reality of requirements engineering. A method may be conceptually elegant but this elegance is a straitjacket when requirements engineers attempt to apply these methods to real systems. It is not easy to define and discover viewpoints, stakeholders insist on using their own notations and terminology to describe their requirements, requirements engineering is carried out to a tight schedule so it is not always possible to collect all desirable information and political and organisational factors influence technical requirements.

To address these difficulties, we have developed a flexible model of viewpoints which can be adapted to a wide range of industrial settings. The work was carried out in the context of a collaborative project where we cooperated with consultancies and user partners from the aerospace and railway industries and was guided by the requirements of industrial partners in the project. We were particularly concerned with providing support for requirements *elicitation* rather than analysis or system modelling. In this respect, our work complements rather than competes with other modern viewpoint-oriented approaches which provide support for consistency checking in the later stages of the RE process.

An overview of the approach which we have developed (called PREview) has appeared elsewhere [4] In this paper, we focus on its application and we illustrate its principal concepts using real industrial examples and discuss experiences of users when applying the approach.

2. Viewpoints in PREview

The earliest requirements engineering method which used explicit viewpoints was CORE [5] where viewpoints were informally defined and, in practice, were mostly seen as sources or sinks of data. This method has been fairly widely used in the UK aerospace industry but has received little attention outside of that domain. A notable feature of CORE which we believe contributes to its usability is that fact that its viewpoints are flexible - users of the method can interpret them in the way which is most appropriate to their needs.

More modern techniques of viewpoint-oriented requirements engineering [1, 3, 6] have attempted to define viewpoints more precisely as structured entities. For example, Finkelstein and Nuseibeh have the notion that a viewpoint represents an engineering perspective and that different viewpoints should encapsulate the different types of requirements model which are natural to different stakeholders. They have examined how to manage inconsistencies in these models and derive an agreed set of system requirements. Other approaches, such as that described by Greenspan [2] identify a fixed set of viewpoints namely services, workflows, organisation and systems which should be used in the requirements elicitation process.

The problem which we have found with these more structured approaches is that they are too rigid. They are based around the idea of a single type of viewpoint and require the specification to be fitted around this concept. While this is often *possible*, the reality is that stakeholders and end-users are reluctant to adapt to the structures imposed by methods. They have their own ideas and do not have time to think about how to present them in what is (to them) an artificial framework.

Therefore, in our model of viewpoints we don't impose any particular types of viewpoint or notation. We propose a flexible approach which accommodates diverse viewpoints and which allows users to define viewpoints which are appropriate to their application. While we provide some guidance on what we think are appropriate types of viewpoint these don't have to be followed. However, we do insist that if users decide to define their own type of viewpoint, they should do so explicitly and should specify the perspective of that viewpoint.

A viewpoint in PREview consists of the following components:

- The viewpoint *name*. This should be a meaningful identifier for the viewpoint.
- The viewpoint *focus*. A definition of the perspective taken by the viewpoint. As discussed above, this is up to the users of the approach. To provide some assistance, we have identified three different types of foci in our approach namely 'interactor foci', 'indirect stakeholder foci' and 'domain foci'. Interactor foci are the perspectives of people or other systems which interact directly with the system being specified e.g. a train driver in a signalling system. Indirect stakeholder foci are the perspectives of people, organisations or systems which have some stake in, and which may influence, the requirements e.g. a safety regulator or a maintenance manager. Domain foci are perspectives which encapsulate domain information e.g. the braking characteristics of a train, the EMC of equipment, etc. This attribute is included to help discover viewpoints which are potentially overlapping and which are therefore more likely to have common or conflicting requirements.
- The viewpoint *concerns*. This is a list of all the concerns applicable to the viewpoint. Concerns are the drivers of requirements elicitation. We discuss them in the next section of the paper. Concerns are included so that business drivers can be explicitly taken into account in the RE process.
- The viewpoint *sources*. These explicitly identify the sources of the requirements associated with the viewpoint. They may be individuals, roles, other systems, or documents. Maintaining sources is important for the backwards traceability of the requirements.

- The viewpoint *requirements*. This is the set of requirements elicited from the sources and from analysis of the system from the viewpoint's perspective. These may be expressed in whatever notation is preferred by the sources. This means that there is no need for requirements sources to adapt their requirements to suit an inappropriate notation.
- The viewpoint *history*. This records changes to the information recorded in the viewpoint over time. Again, this is useful for backwards traceability - we can see where the requirement has come from.

This viewpoint model is deliberately flexible and informal. Viewpoints can be adapted to specific organisational practice and standards as can the notations used to describe system requirements. Viewpoints may be used during the early stages of a requirements engineering process as a structuring mechanism for requirements elicitation and analysis. Identifying viewpoints and organising information around them at this stage reduces the possibility that critical information will be missed during requirements elicitation and provides a traceability mechanism for linking requirements with their sources.

Unlike the viewpoint model proposed by Finkelstein *et al.* [3] [6], our model is geared to elicitation and is *not* primarily intended for requirements validation. In their approach, they support requirements conflict analysis through cross-viewpoint consistency checking. We have not addressed this issue in any depth PREview although we believe that it could be adapted for analysis. However, when discussing the development of PREview, our industrial partners did not consider support for conflict analysis to be a high priority.

We illustrate PREview using examples from the specification of an on-board train protection system called "TCS". This is a system which is currently being implemented for installation on suburban trains. The train is principally controlled by a driver but TCS can intervene under certain circumstances. Its role is to ensure that the driver respects the operational rules and to take corrective action when the driver breaks these rules and provide diagnostic information. Essentially, if the driver allows the train to go too fast or to illegally cross a "stopping point" (such as a signal at stop), TCS will cause the emergency brakes to be applied and the train stopped. The parameters of the operational rules (e.g. the speed limit) vary from one track segment to another and, as the train enters a new track segment, the relevant data for that segment is broadcast to the train as it passes a track-side transmitter.

TCS must be retrofitted to existing trains and integrated with an existing execution environment and other on-board systems including a more sophisticated train protection system which is used on busy underground (central) sections of the lines. A module called "Hardware Systems Interface" (HSI) exists through which the TCS software communicates with other systems. This provides an interface of functions permitting (e.g.) emergency braking to be invoked, and data permitting TCS to poll for train speed, distance to next stopping point, etc.

Eight TCS viewpoints were initially identified:

1. *HSI* The hardware systems interface which is used for communication
2. *Central section* Transitions to and from central sections of track equipped with an alternative protection system
3. *Driver* The train driver's viewpoint
4. *Integration* Integration of TCS with existing on-board software
5. *Architectural integration* Integration of TCS with existing on-board software at the architectural level
6. *Standards* Quality and development standards which are applicable to the system development
7. *Emergency braking* The viewpoint of the emergency braking system
8. *Braking characteristics* The method used to compute braking distances

There are clearly overlaps here and subsequent refinement of the list resulted in a single viewpoint which combined the *Integration* and *Architectural integration* viewpoints. We were surprised at some of these viewpoints, such as *the Central section* viewpoint, as this did not fit our intuitive model of viewpoints mapping to stakeholders or domain phenomena. It is, essentially, a way of partitioning requirements and we were gratified that the viewpoint model was sufficiently flexible to be used in this way.

Now let us look in more detail at three viewpoints which were identified in the specification of the TCS system. The *Emergency Braking* viewpoint (Figure 1) focuses on the safety functions incorporated in the system which ensure that the train does not exceed the specified speed limit for the current track segment and does not overshoot red signals. As the system is being retrofitted to work with existing equipment, there is not a single 'Emergency Braking' subsystem. The safety functions are part of several subsystems. This viewpoint does not fit neatly into any of the viewpoint classes discussed above. Notice here that one of the sources of the requirements is the existing protection system on the train.

Name	Emergency Braking
Focus	The protection system of the train which must detect dangerous conditions and apply emergency braking to bring the train to a safe state.
Concerns	Safety Compatibility
Source	Systems design group A Functional specification of existing protection system software (ref XYY)
Requirements	*SS1* (Detection of excess speed): If the speed of the train exceeds the safe speed for the current track segment by more than 6 kph emergency braking shall be applied. *SS2* (Detection of overshooting): If the signal sensor indicates a danger setting and the front of the train has entered the signalled track segment, emergency braking shall be applied. *SS3* (Frequency of invocation): Detection of excess speed, detection of overshooting and determining the necessity of emergency brake application shall be performed once every iteration of the on-board software application cycle.
Change history	

Figure 1 The Emergency Braking viewpoint

The *Braking characteristics* viewpoint (Figure 2) is an example of an application domain viewpoint which provides essential information on how computations are to be carried out by the system (we have simplified the actual computation for this example). This information is used in the computation of the actual speed limit for a track segment. Notice here that the requirements are expressed using mathematics and cannot be easily related to a system model.

This type of background information is common in requirements documents. However, many requirements engineering methods do not allow for this type of requirement, which cannot be easily classified as functional or non-functional, to be expressed within their formalisms.

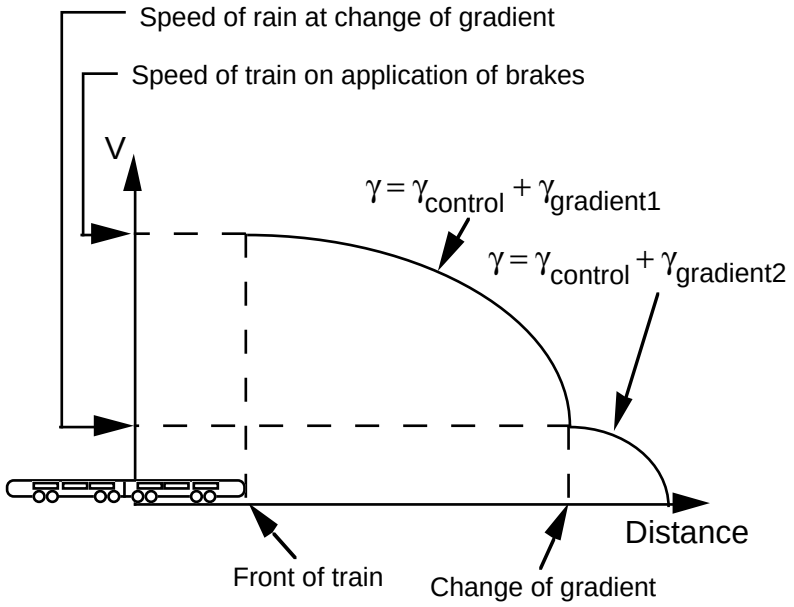
Name	Braking characteristics
Focus	The physical characteristics of the train's braking system.
Concerns	Safety Compatibility
Source	Train dynamics handbook
Requirements	The following information shall be used to compute the distance required to bring the train to a complete stop. The deceleration of the train shall be taken as: $\gamma_{train} = \gamma_{control} + \gamma_{gradient}$ where: $\gamma_{gradient} = 9.81\ ms^{-2} *$ compensated gradient / alpha and where the values of *9.81 ms$^{-2/}$ alpha* are known for the different types of train. $\gamma_{control}$ is initialised at 0.8 ms^{-2} - this value being parameterised in order to remain adjustable. The figure below illustrates an example of the train's deceleration by using the parabolas derived from the above formula where there is a change in gradient before the (predicted) stopping point of the train. 
Change history	

Figure 2 The Braking characteristics viewpoint

The *Driver* viewpoint (Figure 3) is an example of a viewpoint representing an end-user which interacts with the system. The TCS is an automatic system so the driver's functionality is limited but there is a need for the driver to be able to receive warnings from the system and to monitor its operation. Therefore, most of the requirements from the Driver viewpoint are 'non-functional' requirements.

Name	Driver
Focus	Usability and ergonomic requirements of drivers' interaction with the system
Concerns	Safety Compatibility
Source	Train drivers (ref XXY) Operating company driver safety regulations (ref XYX) Driver ergonomics recommendations (ref YXX)
Requirements	• *D1* (Visual indicator) The TCS shall provide a visual indication in the driver's cabin when it is operational. • *D2 (*Speed warning) The TCS shall provide a visual and audible warning when the train speed exceeds the track segment speed limit by more than 0.5 kph. • *D3* (Alarm) The TCS shall provide a visual and audible warning when the emergency brakes are automatically applied. • *D4* (Reset) The train driver shall only be able to reset the TCS to normal operation when the train is at a complete halt.
Change history	

Figure 3 The driver viewpoint

3. Concerns

In a review of open issues in requirement engineering, Zave [7] identified the following as an outstanding problem area:

Generating strategies for converting vague goals (e.g. "user friendliness", "security", "reliability") into specific properties or behaviour.

This is a particular problem where the vague goals are critical to the success of the enterprise. Such goals often represent high-level, non-functional requirements of the principal stakeholders. For example, security and availability are overriding goals of banking applications, while usability is an essential goal of mass-market desktop applications.

These broad system objectives must be identified at the project's outset and managed to ensure that they exert the appropriate influence on the system's development. They must:

- be elaborated into detailed non-functional and, ultimately, functional requirements which can be explicitly addressed by subsequent design and verification phases.

- act to constrain the rest of the requirements process so that they are not subverted by conflicting but otherwise valid requirements. Failure to detect and resolve such conflicts must inevitably lead to rework or worse [8].

In PREview, we have addressed the problem of deriving requirements from vague goals by introducing the notion of 'concerns'. Concerns represent high-level criteria which are business or mission-critical and are used as drivers of the requirements elicitation process. Concerns are "global" in the sense that they have a wide scope and, potentially, affect every aspect of the system. If a "concern" does not meet this criterion, it is not a concern. Concerns are not decomposed into *detailed* system requirements but they influence the requirements which are derived in each system viewpoint. Constraints which are derived from concerns generally override viewpoint requirements and must be considered when viewpoint requirements are derived.

The global nature of concerns is what distinguishes them from viewpoints. Figure 4 shows the conceptual relationship between concerns and viewpoints as a socio-

technical pyramid. At the apex of the pyramid are the viewpoints which interact directly with the system. At its base are those viewpoints which have the most indirect association with the system, but nevertheless have a stake in it. Viewpoints at each level impose requirements which are related to the particular type of viewpoint. Concerns, however, cut through all of these because they potentially constrain any requirement, regardless of its level.

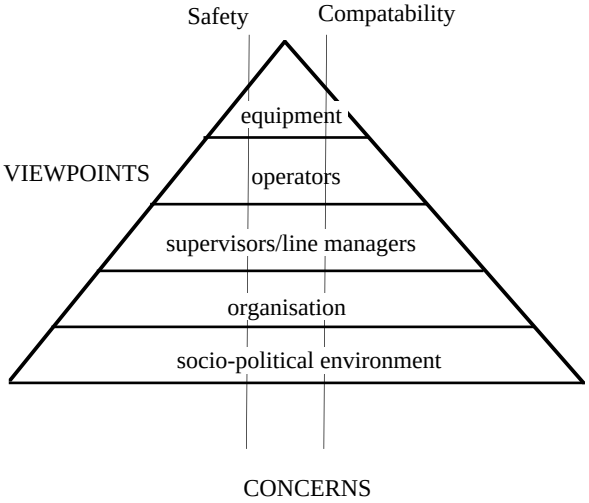


Figure 4 The orthogonality of viewpoints and concerns

We use the term *concern* rather than *goal* to distinguish our approach from so-called 'goal-oriented' methods of requirements engineering. We don't refine concerns through goals to requirements - rather, we use concerns as a means of identifying critical information which must be collected during requirements elicitation. Goal-oriented approaches to requirements engineering[9] [10] are based on refining vague objectives into concrete formal goals then decomposing these further into sub-goals until a set of primitive goals which can readily be expressed as system requirements has been derived. These approaches have the advantage that they expose the different goals from different stakeholders and they provide a structured approach to assessing alternatives. However, goal-oriented approaches are still immature and, in our view, are not yet ready for widespread industrial application.

Concerns are analogous to Critical Success Factors (CSF) [11]. These are the conditions necessary for a system to enhance the customer's business. They are usually defined by management with a strategic view of the business within the customer's organisation. A methodology to support CSF has been developed primarily for commercial applications. PREview, by contrast, has been developed for industrial applications such as control software within systems engineering projects. Here, a top-down approach is inadequate because requirements emerge not only from stakeholders such as the customer and users, but also from the application's domain and environment. Hence, a requirements method for such applications needs to be both top-down (to ensure that concerns are satisfied) and bottom-up (to ensure that each source of requirements is identified).

PREview defines a process which commences with the identification and elaboration of concerns which are then used to drive the subsequent requirements activities. The strategy which we have adopted to relate concerns to system properties and requirements has been influenced by Basili and Rombach's GQM paradigm [12]. This approach, which was developed for system measurement, starts off by identifying goals and sub-goals, deriving questions related to these goals and, finally, identifying metrics which provide answers to these questions. Our approach has a comparable structure:

1. Identify the concerns which affect the system (goals).
2. Derive a set of questions which will ensure that the information required to satisfy the concerns is collected.

3. Elicit and negotiate requirements which ensure that the system satisfies the identified concerns.

Concerns must be identified at the outset of a project by discussion with the principal stakeholders; typically the client and developer. In the TCS application, two concerns apply: *Safety* and *Compatibility.* Safety is a concern because it contributes to train safety and is important for certification. Compatibility is a concern because TCS must be integrated with the existing systems' execution cycle and because the HSI module provides the interface through which all communication with other software and hardware modules is made.

Once identified, concerns must be elaborated into a form which facilitates their analysis with respect to other requirements. The first stage in this analysis is to decompose the vague global concerns into a more specific set of sub-concerns. The way in which this is done must depend on the type of concern. For example, if safety is a concern, each identified hazard might be a sub-concern; if security is a concern, then each type of threat to the system (virus, unauthorised access, etc.) could be the sub-concerns which are identified.

Figure 5 shows part of the decomposition of concerns to questions for the TCS. The safety and compatibility concerns are decomposed to more specific sub-concerns (hazards in the case of safety, different types of compatibility in the case of compatibility) and then to questions which may be asked to requirements sources during the elicitation process.

Concerns are decomposed into associated questions which are put to stakeholders during the process of requirements elicitation. These questions generally fall into two categories:

1. Questions which identify essential information which must be elicited. We can see an example of this under the safety concern in Figure 5 where we have the question 'what does excess speed mean in reality?'. Simplistically, you might think that excess speed is anything greater than the segment speed limit but, in practice, the definition is very much more complex than this. Excess speed is any speed above which it cannot be guaranteed that the train can proceed through the track segment in safety and, in an emergency, can be brought to a safe stop. It depends on the type of train, the weather conditions, the track gradient and whether or not there is a train in the following segment.

2. Questions which identify potential constraints on other requirements. For example, the question 'Does the requirement need data which isn't available through the HSI interface' in Figure 5 means that potential requirements inconsistencies can be avoided. This question is asked when a requirement is proposed.

In our TCS example, consider the case where a stakeholder articulates a requirement to calculate a minimum braking distance to a high degree of accuracy. Such a requirement means that real-time data on train speed, mass, line gradient and track surface conditions must be available. Applying the concern questions to this requirement would prompt checking that this data was indeed available through the HSI interface. In fact, track surface conditions are not monitored via the HSI module so the unfeasibility of the requirement in its current form would be revealed avoiding the costs and time required for later analysis.

In some cases, the concern may result directly in a high-level system requirement. We see this in Figure 5, where these requirements are enclosed in boxes. In PREview terminology, these abstract requirements which are derived from concerns are called 'external requirements' to distinguish them from requirements which are elicited from one or more viewpoints. These external requirements represent the first stage in converting the abstract concerns into concrete functional and non-functional requirements for the system. Of course, as we can see, this may involve the generation of further questions which are put to requirement sources during the elicitation process.

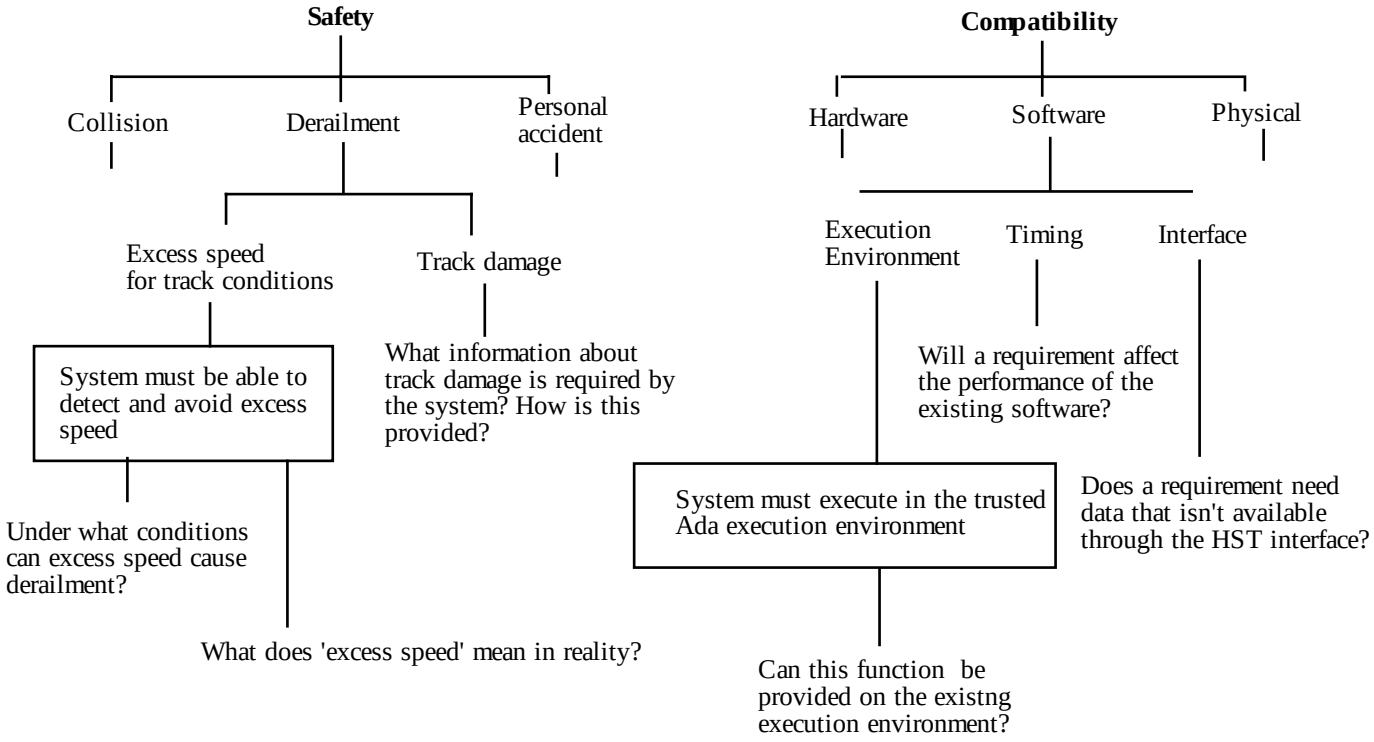


Figure 5 Concern decomposition

Some classes of concern employ other techniques to assist with their elaboration. For example, a safety concern might be elaborated by using hazard analysis techniques such as fault-tree analysis or Hazops [13] to identify the specific hazards against which the system must provide a defence.

4. The PREview process

The objective of the requirements process is to deliver a requirements specification document which defines the system to be developed. However, the process must recognise that the quality of requirements information never attains perfection. The point at which the requirements process terminates is a matter of judgement about when and whether the collected requirements are 'good enough' to get started on development. Models of the requirements process must define activities aimed at identifying and resolving requirements defects while coping with those which inevitably emerge at later stages.

We must emphasise here that PREview is not a requirements engineering method with associated rules and guidelines. We found that potential users already had methods in place (e.g. SADT) and they did not want to face the problem of trying to integrate these with some new method. However, they did ask for suggestions as to what *process* might be used with PREview and this is really all the methodological support that we provide.

Boehm [14] has proposed a requirements process model based on his spiral model of software development [15], augmented to include provision for establishing stakeholders' "win" conditions. This means the provision of steps to facilitate identification and negotiation of requirements trade-offs. These are necessary to ensure that all relevant factors (technical, economic and political) exert the appropriate influence on resolving stakeholders' conflicting requirements. Potts *et al.* [16] have also proposed a cyclical model, called the Inquiry Cycle. This consists of three iteratively repeated activities; expression, discussion and commitment. The

inquiry cycle is interesting because the expression activity includes provision for handling enterprise goals which are essentially the same as concerns.

The generic process model adopted by PREview (Figure 6) is an adaptation of these. It is a spiral in that the requirements information which emerges from successive iterations does so in the context of the requirements information which emerged from previous iterations. Hence, for example, requirements information which emerges in the first iteration may constrain requirements which emerge in later iterations. They may also need to be modified in the light of information which emerges later. Like the inquiry cycle, the identification and elaboration of concerns forms an integral part of the process.

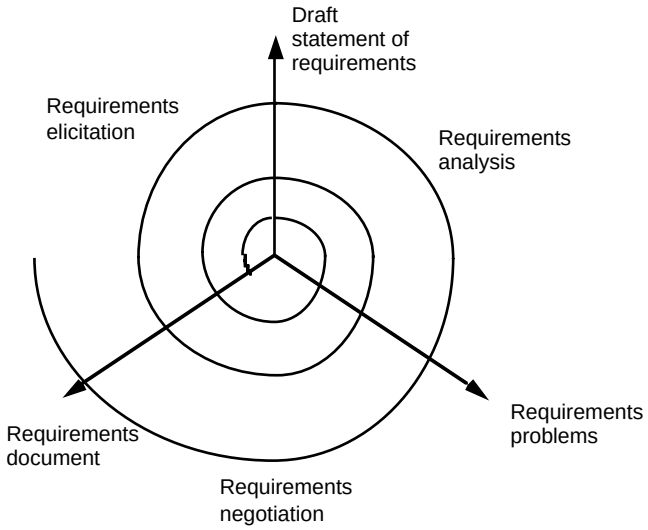


Figure 6 The top-level PREview process model

The three basic activities performed each cycle are:

1. *Requirements elicitation* Given a statement of organisational needs and other inputs, various different sources are consulted to understand the problem and the application domain and to establish their requirements. These requirements may not be complete and may be expressed in a vague and unstructured way.
2. *Requirements analysis* The requirements discovered during the elicitation phase are integrated and analysed. Usually, this will result in the identification of missing requirements, inconsistencies and requirements conflicts.
3. *Requirements negotiation* Inconsistencies and conflicts discovered during analysis need to be resolved. The analysts and stakeholders consider the problematic requirements to try to reach a consensus about their resolution and hence reach acceptable "win" conditions for all stakeholders. These trade-offs may necessitate the elicitation of further requirement information.

Before the first iteration of the spiral, the process commences with identification and elaboration of the concerns. To be manageable, the number of concerns for a particular application will be small; more than 5 will be difficult to handle and may result in inter-concern conflicts. The concerns set an agenda by which requirements are discovered, analysed and documented with constant reference to the need for compliance with, and satisfaction of, the concerns.

Figure 7 illustrates the PREview process in more detail. The three activities in Figure 6 are represented by the top three shaded boxes in Figure 7. The ellipses within these represent individual activities. The box at the bottom represents the output from the PREview process and the transition to a detailed requirements specification document.

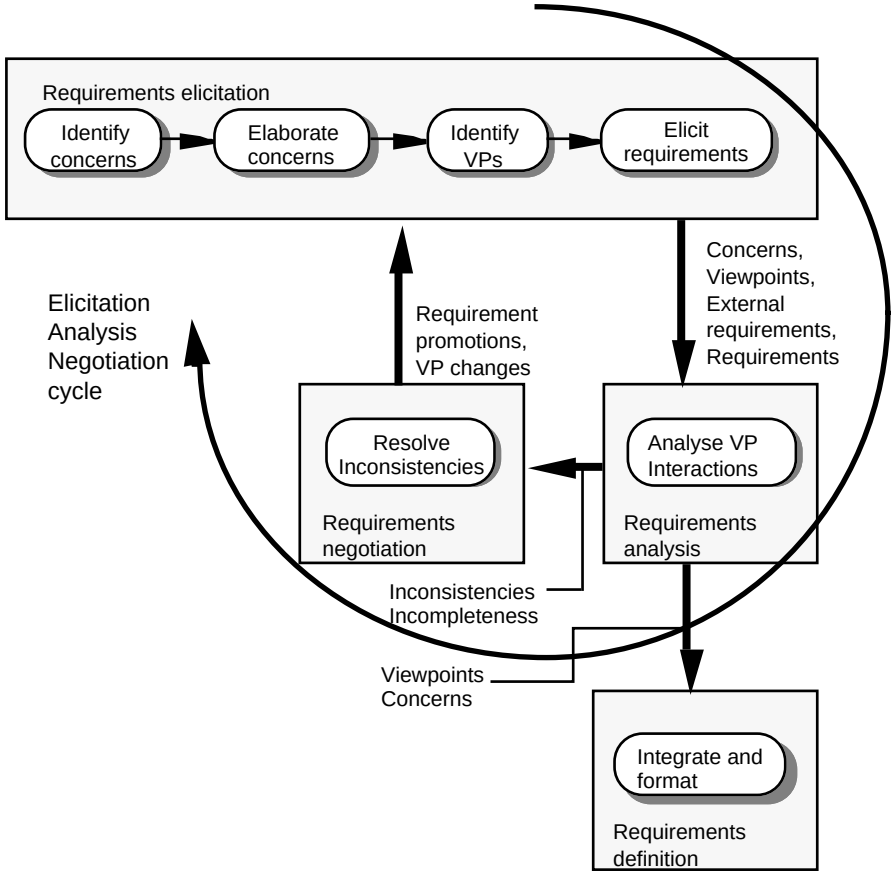


Figure 7 The PREview process in detail

A key process activity is identifying viewpoints which are appropriate to the application. The process we use to help discover viewpoints is shown in Figure 8. Viewpoint identification starts with informal modelling of the system's operational and organisational environments. Analysis of the operational environment reveals users of the system, the system's technical context and, with input from domain experts, domain constraints. The organisational environment helps identify the different people and roles within the organisation who have an indirect stake in the system.

The desire to represent each possible perspective on the system must be tempered by the need to keep the number of viewpoints small. Practically, viewpoints become hard to manage if there are more than 10. If more than this number emerge, commonalities should be identified and related viewpoints should be integrated into more generic viewpoints.

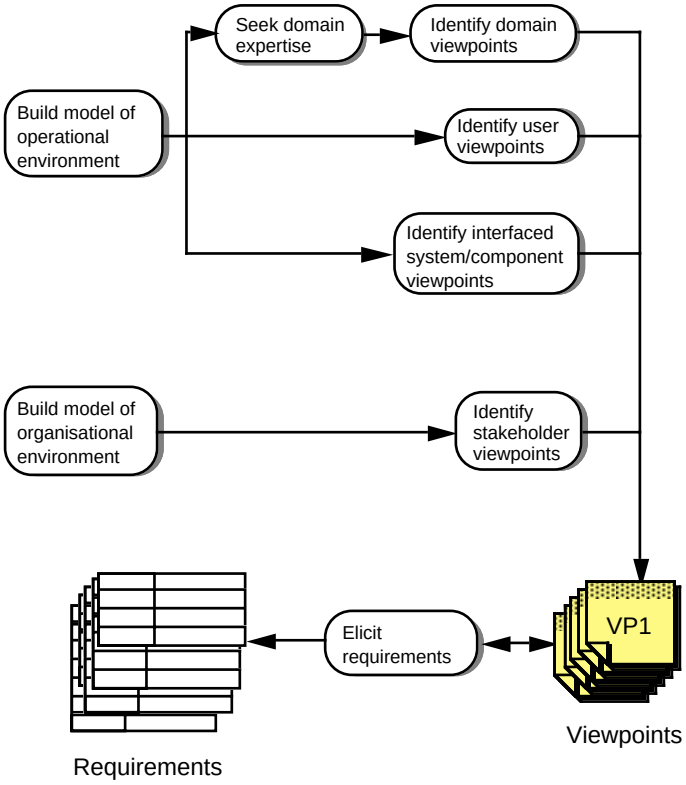


Figure 8 Identifying viewpoints

5. PREview in practice

The reality of applying new requirements engineering methods is usually quite different from that intended by the designers of the method. Not only do real applications introduce unexpected problems, the method has to be applied by busy people who often don't have sufficient time to develop a complete understanding of the approach. In this respect, PREview is no different from any other method although we hoped that it did not share some of the usability problems of other methods developed in academic research laboratories.

So far, the experiments with PREview have been confined to the specification of small to medium-sized safety-critical systems. This reflects the environment in which it was developed (the target application domain of the project where PREview was developed was safety-critical systems) and the lack of special-purpose tool support for our approach. We had only a set of standard word processor form templates to support the development.

We carried out a number of evaluations of PREview; in some of these, application and method developers worked side-by-side; in others, the method was used solely by the application engineers. It isn't possible to describe these in detail here but some key points which emerged from the practical application of PREview were:

- The type of viewpoints which were identified were unexpected such as a track segment viewpoint (*Central section*) in the railway example and a water level viewpoint in a boiler control system example. Surprisingly, users did not find it difficult to identify relevant viewpoints nor did they find the lack of specialised tools a disadvantage. By and large, viewpoints were not associated with human stakeholders but we think this is a consequence of the domain (safety-critical embedded systems) where PREview was applied.

- The viewpoint identification process was found to be helpful as a starting point but once users developed an understanding of how viewpoints could be applied, they derived their own approaches to viewpoint identification.
- The focus of viewpoints was defined in one or two words. In essence, the requirements engineers understood what a viewpoint was and did not see the need to document this. Clearly, we failed to communicate the need to document the focus so that future readers of the requirements could understand the rationale behind the viewpoint.
- No-one was interested in the change history and no information on this was ever provided. Nevertheless, we have retained this in PREview as we believe that it is potentially useful for requirements management and traceability. However, this probably needs special-purpose tool support to be usable.
- The fact that there was no pre-defined notation was appreciated and users described requirements in a variety of different ways ranging from informal diagrams through SADT to mathematical models.
- Concerns caused some problems initially as (understandably), there was some confusion about the differences between concerns and viewpoints. However, when we demonstrated that the essence of concerns was to derive questions which are then used within viewpoints to derive requirements, users of PREview found them to be a useful concept.
- Our initial idea was that all concerns should be relevant to all viewpoints and that all questions derived from concerns should be used during elicitation. In fact, users made a judgement about which concerns were relevant to a viewpoint and only asked these questions.
- The lack of tool support was less of a problem than we anticipated. It meant that users could continue using their current tools with no compatibility problems. In their experience, the difficulties of getting tools to work together often outweighed the advantages of specialised tools.

In general, the response to PREview was fairly positive. The notion that viewpoints could be flexible was much appreciated. Partners had examined other viewpoint-oriented methods and had rejected them simply because they could not accommodate the way they wished to work. They found that PREview could be adapted to their existing processes and could integrate with the existing methods (SADT) which were used.

6. Conclusions

PREview is a pragmatic adaptation of an old idea - the use of viewpoints for requirements engineering. The need for a viewpoint-oriented approach which could be easily learned and was not too restrictive was established by our industrial collaborators at the outset of the REAIMS project and has been the method's underpinning philosophy. It has been designed to be adaptable to an organisation's existing requirements process. It can be adopted in an evolutionary way rather than requiring a revolutionary change to a development organisation's existing practice. As PREview is not prescriptive about notations or methods, it can be integrated with existing requirements methods as a front-end process for elicitation.

The explicit recognition of the importance of organisational needs and priorities - the concerns of the principal stakeholders - is an important new feature of PREview. This emphasis on stakeholders' concerns and the integration of these with viewpoints is consistent with priorities identified at a recent industrial workshop on requirements engineering [17]. Here, the management of customer information was identified as one of the most problematic areas for reasons which include:

- *Requirements are gathered from a number of viewpoints*.
  This is implicit in almost any requirements process but PREview (and other

methods) make it explicit. PREview provides explicit guidance on how to identify and manage viewpoints.

- *Details tend to get lost in the generality of the application*.
  Viewpoints provide a structure for associating requirements with contextual information regarding their sources, the focus they have on the application and the concerns which constrain them.
- *It is not always clear when elicitation should cease*.
  The set of requirements is almost never complete. PREview's spiral model embodies an acknowledgement that this is the case and supports the iterative elicitation and analysis of evolving requirements.
- *Requirements are not always "there" to be elicited*.
  To cope with this, requirements have to be actively sought. PREview recognises the disparate nature of requirements sources. It embodies the idea of domain viewpoints for which no "stakeholder" may exist.

In summary, PREview helps improve the quality of requirements specification by providing a framework for the early phases of the requirements process and enshrines the need to comply with and to satisfy the overall, binding concerns defining the success or failure of a project. Ongoing work in the development of PREview includes the development of a prototype toolset for requirements management and the integration of the approach with other viewpoint-oriented methods for requirements engineering.

7. Acknowledgements

The work described here was partially funded by the European Commission in the REAIMS project (8649). We gratefully acknowledge the contributions made to this work by our collaborators on the REAIMS project: Adelard, Aerospaciale Aeronautique, Aerospaciale Protection Systemes, Digilog, GEC Alsthom Transport, RWTÜV and The University of Manchester.

8. References

[1] Kotonya, G. and Sommerville, I., "Requirements engineering with viewpoints". *BCS/IEE Software Eng. J.*, 1996. **11**(1): 5-18.

[2] Greenspan, S. and Feblowitz, M. "Requirements Engineering Using the SOS Paradigm". in *Proc. RE'93*. 1993. San Diego, CA, IEEE Press.

[3] Finkelstein, A., Kramer, J., Nuseibeh, B., and Goedicke, M., "Viewpoints: A Framework for Integrating Multiple Perspectives in System Development". *Int. J. of Software Engineering and Knowledge Engineering*, 1992. **2**(1): 31-58.

[4] Sommerville, I. and Sawyer, P., *Requirements Engineering:A Good Practice Guide*. 1997, Chichester: John Wiley & Sons.

[5] Mullery, G. "CORE - A Method for Controlled Requirements Specification". in *4th Int. Conf. on Software Engineering.* 1979. Munich.

[6] Nuseibeh, B., Kramer, J., and Finkelstein, A., "A Framework for Expressing the Relationships between Multiple Views in Requirements Specifications". *IEEE Trans. on Software Eng.*, 1994. **20**(10): 760-73.

[7] Zave, P. "Classification of Research Efforts in Requirements Engineering". in *Proc. RE'95*. 1995. York, England.

[8] Hutchings, A. and Knox, S., "Creating Products Customers Demand". *Comm. ACM*, 1995. **38**(5): 72-80.

[9] Fickas, S., Van Lamsweerde, A., and Dardenne, A. "Goal-directed concept acquisition in requirements elicitation". in *Proc. 6th Int. Workshop on Software Specification and Design*. 1991. Como, Italy.

[10] van Lamsweerde, A., Darimont, R., and Massonet, P. "Goal-Directed Elaboration of Requirements for a Meeting Scheduler". in *Proc. RE'95*. 1995. York, England.

[11] Rockart, J., "Chief Executives Define their own Data Needs". *Harvard Business Review*, 1979. **57**(2).

[12] Basili, V.R. and Rombach, H.D., "The TAME project: Towards Improvement-Oriented Software Environments". *IEEE Trans. on Software Eng.*, 1988. **14**(6): 758-773.

[13] Leveson, N.G., *Safeware: System Safety and Computers*. 1995, Reading, Mass.: Addison Wesley.

[14] Boehm, B., Bose, P., Horowitz, E., and Lee, M.-J. "Software requirements as negotiated win conditions". in *Proc. ICRE'94*. 1994. Colorado Springs.

[15] Boehm, B.W., "A Spiral Model of Software Development and Enhancement". *IEEE Computer*, 1988. **21**(5): 61-72.

[16] Potts, C., Takahashi, K., and Anton, A., "Inquiry-based Requirements Analysis". *IEEE Software*, 1994. **11**(2): 21-32.

[17] Morris, P., Masera, M., and Wilikens, M., *Industrial Workshop on Requirements Engineering - a report on the results obtained.*1996, ISPRA, Italy: EUR 16360EN.