Requirements Engineering in the Year 2001: On (Virtually) Managing a Changing Reality

Matthias Jarke and Klaus Pohl
Informatik V, RWTH Aachen
Ahornstr. 55, 52072 Aachen, Germany
jarke@informatik.rwth-aachen.de

Trends in society and technology force requirements engineering (RE) to expand its role from a one-shot activity in the development process to a virtual image that accompanies the changing reality of a system. A maturing software market will require a better understanding of the differentiation in market segments also for RE, and standardization of methodologies within these segments. On the research side, this will require a coherent perspective of hitherto parallel research directions towards a comprehensive understanding of requirements processes, as well as the optimal exploitation of new technologies that support the main role of requirements engineering: mutual learning of all stakeholders concerned.

1 The Changing Role of Requirements Engineering

Many engineering and business disciplines are experiencing the need to understand better the early phases of their processes, and to maintain this information over time and across traditional technical and organizational boundaries. Among the many buzzwords (which will of course be forgotten in 2001!), *total quality management* refers to a comprehensive approach to quality consciousness, planning, and control, *business process re-engineering* to the analysis and re-design of activity or value chains in business, *concurrent engineering* to the simultaneous analysis of engineering problems from multiple perspectives, and *virtual manufacturing* to the management of manufacturing processes that span organizational boundaries[Yoshikawa and Goossenaerts 1993]. Similarly, there is a growing demand for making the early phases of processes in industrial research (e.g., drug synthesis in the pharmaceutical industry) more effective.

Within software engineering, requirements engineering (RE) has traditionally been responsible for the kind of task which transforms a fuzzy initial idea into a precise system specification. The good news is that none of the above disciplines seem better off than

software engineering in terms of understanding the actual experience, and offering adequate methods and tools for RE. The bad news is that even within software engineering, the meaning of RE is changing. We have to become more clear what we need and what we have to offer. The answer to the question what RE will be in the year 2001 will heavily depend on our response to this challenge.

1.1 The State of the Art and Advanced Practice

The classical RE cycle consists of the three steps *elicitation - expression - validation*. As pointed out by Brackett [1990], the choice of requirements methods and tools is usually determined by a single expected major source of complexity in the product to be designed: function (e.g., structured analysis and its derivatives [DeMarco 1978]), data (e.g., entity-relationship and its derivatives [Chen 1976]), or time-sequencing as inreal-time systems (e.g., statecharts [Harel 1987]). This emphasis also determines the typical strength of tools, e.g., traceability of product relationships in the ER-oriented tool RDD-100 of Ascent Logic Inc., or executability of specifications in the statechart-oriented STATEMATE of iLogix Inc. Object-oriented analysis methods (in-the-small) and enterprise integration architectures (in-the-large) look at several of the above perspectives simultaneously; but their formal relationships and therefore technical cooperation support remains poorly understood.

Several implicit assumptions seem to have accompanied much of the early work in RE:

- There exists a well-definable *problem* that can be clearly scoped, described (and subsequently solved) once and for all; once completed, the system remains largely stable. The structured methods developed in the 1970´s are prototypical of this assumption.
- The requirements specification forms the basis of a *contract* which is usually fulfilled by someone else who is normally not responsible for the subsequent use and maintenance of the system.
- Each problem is different from others. Therefore, successful systems development is an art. Requirements *engineering* means to many a rigorous way to do art.
- *Users* are typically computer-illiterate but domain specialists who tend to know well what they want and just need to be questioned the right way. There exist only a few specialists who understand computers and have to be implicitly trusted.
- *Methods* for the early phases are generalizations of methods for later phases in systems development, i.e., they stem from *computer science*. Structured methodologies went from structured programming via structured design to structured analysis, today we see the same historical pattern for the object-oriented approach.

1.2 The Challenge

The traditional approach was quite successful as long as the software market was immature, organizations and their software systems were relatively stable over time, and information technology (IT) was localized in temporally stable islands of automation. Today, it is becoming increasingly clear that these conditions hold no longer:

- ***coping with continuous change****:* Due to economic pressure, also as a consequence of new technology especially in the communications sector, organizations change much faster than current IT can accomodate; many now see IT as an impediment rather than a motor of progress. To facilitate goal-directed change in evolving organizations, requirements models must now be carried along throughout the life time of a system.
- ***systems embedded in the world***: Systems are distinguished from works of art by a particular purpose and usage environment, and they are often used as an indirect means to understand or manage other systems (e.g., information systems, process control systems). Therefore, RE must address the environment of a system as well as the system itself. The system boundaries are defined by the purpose of the system. The system may include networks of human as well as computerized hardware and software components, both inside its boundaries and in the environment. Terms such as *composite systems* [Fickas 1987] or *cooperative information systems* [Brodie and Ceri 1992] indicate this trend. The nature of the social processes surrounding RE is so poorly understood that ethnographics, a science originally meant to understand the life of remote and strange tribes, has been proposed to investigate what requirements really exist and how RE really works [Goguen and Linde 1993].
- ***cooperation and scalability***: System structures change rapidly according to new cooperation patterns within and among human organizations. Heterogeneously developed legacy systems must interoperate with one another in ever changing coalitions for ever changing business processes. RE must address the problem of evolving distributed system societies which are themselves specified by temporally and spatially distributed teams. We do not expect the legacy problem to go away, until the year 2001 (nor later). When we recently gave a talk about the future of RE, a very appropriate comment from the audience was: "you better solve the problems of the 1970´s than speculate on those of the 1990´s!"
- ***maturity of the software market*****:** While the traditional playground of RE was the large one-shot project, the literature gradually begins to acknowledge a differentiation in the software market and therefore the RE task. For example, Lubars et al. [1993] differentiate between customer-driven RE (the traditional approach) and market-driven RE. Similarly, Derroitte et al. [1994] observe different ways of working for RE in the CIM domain, taking advantage of architectures, nonfunctional goals, and availability of standard object libraries.
- ***technology-literacy*****:** In the year 2001, you may find more people who are computer-literate than those who are literate in the traditional sense! Software engineers, including requirements engineers, are surprisingly conservative in their choice of technologies. Programmable virtual reality (an extension to the rapid prototyping of today) or multimedia cooperative work will significantly broaden the ability of stakeholders to participate in RE tasks, as we can already observe in computer games, robotics, and interior architecture.

1.3 A Broader Definition of RE

We can summarize the above observations in three important conclusions.

The first one is that *the central role of computer specialists in RE is abandoned* in favor of broader participation, not only from users and their managers, but also from other potential "victims" of computerized systems; the implication is that requirements definitions must include context models, and traceability of all design decisions back to the central system vision as well as to the laws, policies, and habits of all the stakeholders.

The second one is that the traditional one-off approach to RE is being replaced by a requirements *engineering* practice which gains added value by putting together standard components in innovative fashions, rather than constantly proving their ingenuity by re-inventing the wheel.

The third observation is that *conceptual support for change management* is becoming the central task of RE. We can see requirements models as "virtual companions" which help (changing) organizations to pro-actively support their change processes through (changing) IT infrastructure and applications. We claim that requirements engineering in the year 2001 will be central to cope with a changing world through the adequate management of virtual representations of this world and its evolution.

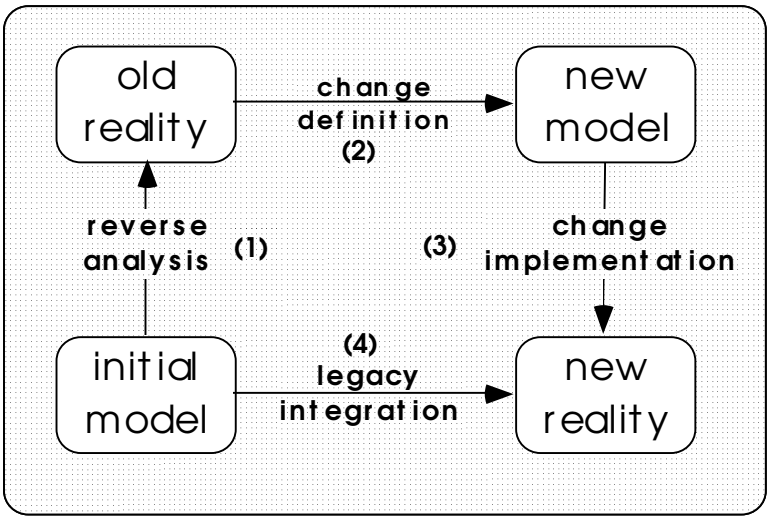

Figure 1: RE and the Management of Change

Let us look at one step in such a change process in some more detail, in order to understand what this proposition implies. Figure 1 shows that propagating a change needs four steps: (1) abstracting a model from current reality, (2) defining the desired change within this model; (3) implementing the changed model in the reality while (4) taking into account the existing context.

Like any expensive activity, such changes are not conducted out of the blue. Before an RE team is established, some need for change is defined by rationalising perceived opportunities or threats, or by political decision-making. The *vision* (the need for change) is typically stated in a simple manner. A classical example is John F. Kennedy´s *"send a man to the moon before the end of the decade"*. Thus, RE is not an undirected analysis process, but a process of transforming the vision into a requirements specification which can then, in the design and implementation tasks, serve as a framework for making the necessary changes in the real world.

Within the world in which the vision has to be realized, many habits exist. Some are based on formally stated goals, policies, or competing visions. Others are just regularly observable phenomena for which no predefined structure or reasons are known a priori.

The task of RE is therefore twofold. First, relevant habits must be analysed and the goals, policies and visions behind them must be made explicit. This is essentially a goal-directed abstraction process from existing practice [Dardenne et al. 1992]. Second, the new vision must be established as a mission in this context. Propagation of their consequences leads to the detection and resolution of conflicts among the different viewpoints [Finkelstein et al. 1990]. During this RE process, the vision often shifts. Large projects often appoint a "vision holder" to ensure that the vision does not get totally lost in the constraints of current practice. Noticing this necessary juxtaposition of vision and context in an always changing environment, we define ***requirements engineering as a process of establishing visions in context*** [Jarke and Pohl 1993].

Many RE methods emphasize one of these aspects at the expense of the other. Top-down decomposition methods such as Structured Analysis do not say how context influences decomposition, whereas bottom-up approaches (ethnographics is an extreme example) so far fail to provide guidance how analysis can be focused on the vision.

In section 2, we elaborate the definition into a more detailed framework for requirements engineering. In section 3, we look at each component of this framework in order to come up with predictions about requirements engineering in the year 2001.

2 A Framework for Requirements Processes

One source from which a framework for RE can be derived is the class of products for which requirements are to be determined. In the sequel, we shall focus on systems that control, manage, or inform about other systems for a particular usage purpose. We call this class *Information Systems* (IS) although it is admittedly broader than what is usually called IS. We first provide a structure for the context of these systems, the *four worlds* of information systems. From this, we derive a framework of three dimensions for describing the process of RE, which we then use for generating structured predictions about RE in the year 2001.

2.1 Structuring the Context

As mentioned above, the context in which the system vision has to be established is complex and continously changing. Establishing a vision in context remains an empty phrase if we do not understand what parts of the world are relevant and how they are related to the development process. Due to the diversity of RE tasks, even an open basic formalism such as the "rich pictures" of soft systems methodology [Avison, Golder and Shah 1992] is not enough. We need some kind of domain ontology — a basic understanding of what IS requirements engineering is concerned with. This ontology should have a simple structure to be acceptable to a broad class of developers in that it is easily understood and does not overly constrain their freedoms in the RE process.

2.1.1 Usage, subject, and system worlds structure IS requirements

The context for IS can be characterized as follows (cf. figure 2). First, the IS represents a certain part of the outside world which we call the *subject domain* — for instance, the domain of cars. Coming back to RE as a form of art, representing a subject domain is a property shared by IS with many works of art, but less emphasized in numerical or algorithmic systems.

However, IS are distinguished from works of art in that they are designed with the purpose of being used by people, e.g. car buyers, with a particular goal in mind. We call this the *usage world*. We further distinguish individual, cognitive aspects of usage, from social or organizational ones.

Third, the system itself is sufficiently complex that it needs some structured description; we call this the *system world*. Consider, for example, a system that supports buying a car. The system world is linked to the subject domain world by a denotational relationship, to the usage world by an individual as well as social usage relationship. Both sides as well as the technical aspects of the system itself, especially if it builds on an existing one, are substantial sources of requirements. Following the traditional notion of the semiotic triangle [...], the system has to represent its subject domain in a way that invokes the right understanding (semantics), individual usage (pragmatics), and social usage (semiotics) in the usage world.

As the arrows in figure 2 illustrate, information about the subject is represented in the system after making a conceptualization. Depending on the system vision this information may include information about the number plate of the car, its engine, the costs, the petrol needs etc. By providing special system functions, again based on the vision, the user is able to retrieve the represented information. Only if he is able to interpret the conceptualization and reflect it back to reality, the information system fulfils its purpose. It is not important if the subject is a existing object or an abstraction of reality shared by a certain community. In both cases, only if the conceptualization, the representation and the resulting presentation are well designed and fitted to the user, the system can be used as expected.

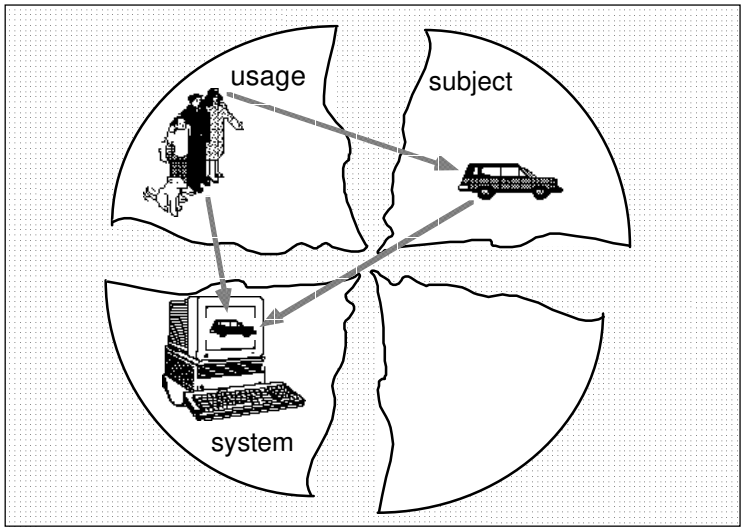

Figure 2: The context of an information system

2.1.2 The vision distinguishes system, usage, and subject worlds

The three worlds introduced above are neither static, nor do they cover the whole "real" world. Whether an object belongs to the usage, system or subject world (or even to two or three of them), is defined by the system vision (figure 3).

For example, if the vision is to build a library system, books on the library shelves are relevant for the subject world, whereas in an IS of a car repair shop, the same books could be parts of the system world; here, the book is part of the information system which represents cars for the usage purpose of repair). Hence, the relevant objects for each world and what is relevant about them during RE is defined by focusing the context using the vision. The vision guides the separation process that creates the three worlds from the infinite variety of real phenomena, and thus decisively drives the modeling process.

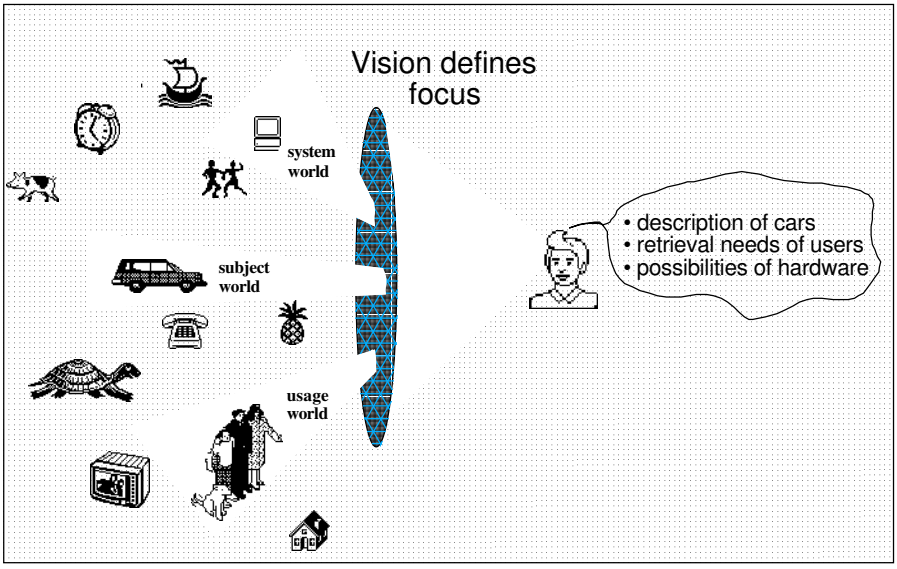

Figure 3 : The vision distinguishes subject, system, and usage world

2.1.3 The development world supports the RE process

But how does the vision define the focus in practice? Looking at RE from a social perspective, the RE process starts with a vision holder who establishes his or her vision in the social context through communication with other people, typically through drawing them into a project team for certain periods of time. As figure 4 shows, this team then becomes a distinguishable social reality of its own, drawing upon and interacting with the other three worlds; we call it the *development world*.

Based on the context partitioning expressed by the three worlds, the people who must be involved in the RE process can be identified. They have different backgrounds and opinions due to the context they come from. It is important to deal with this background, in order to integrate the different aims and goals. Therefore, it is essential to gather information about the relevant parts of the different worlds, evaluate this information against the system vision, detect conflicts, and resolve them during the RE process. If the initiative for these interactions does not come from the vision holder, it will eventually come from the other worlds -- but then as an expensive and difficult to change "requirements bug"!

During the RE process, knowledge about the context is structured in conceptual models or in instantiations of these models. This knowledge is acquired, conceptualized, and integrated within the development world, i.e. conceptual models about the subject, the usage, and the system world are developed. Typical modelling formalisms exist for each world (cf. figure 4). Knowledge about the worlds may be already available in existing models; for example, business models provide a source for information about an enterprise and (depending on the vision) may be part of the usage world. Such models provide sources for reusing abstract, information.

The vision, the need for change, is then integrated into the conceptual model, i.e. the needed changes in the reality are identified and described at the abstract level (cf. figure 1). During the design and implementation of the system based on this specification, the vision is integrated in the "real" context.

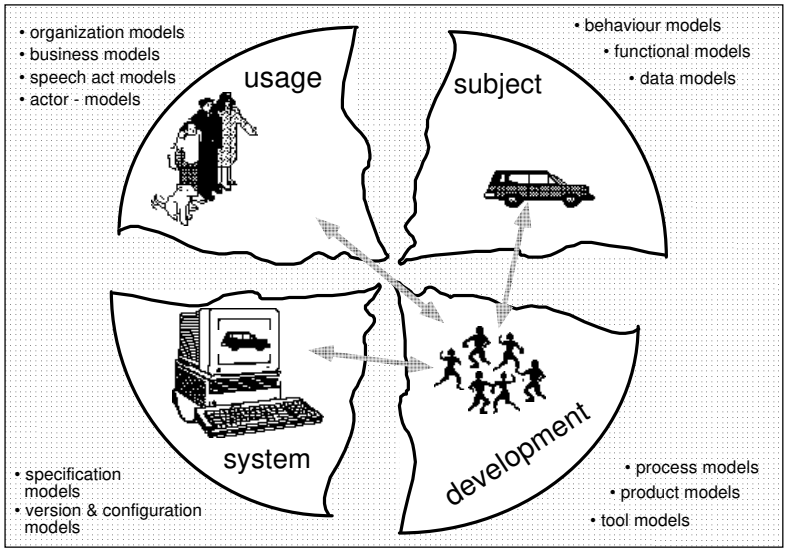

Figure 4: The four worlds of IS

From the roles of the four worlds, we can predict certain goals they will likely impose on the system. We can also predict certain standard representations and structures that could prove useful for describing these worlds, their evolution and interaction. As figure 4 indicates, we find some evidence of our earlier statements:

- A look in the current literature and practice shows that emphasis is shifting away from the computer science domain knowledge (system modeling), to usage world models describing the enterprise, business processes, and the like, in which IT models are embedded as one (but not necessarily the only or even the most important) component. Similarly, there is an increasing demand for analyses of the impact of technology which can be modeled as a relationship between the system world and its subject world.
- Within system models, we see a shift from traditional monolithic specifications organized according to a single paradigm for a single system, to composite system architectures enabling reuse by usually object-oriented libraries of components.
- Within the development world itself, emphasis on process modeling -- the basis for change management and quality improvement -- replaces the traditional emphasis on data or project management.

2.2 Self-Application of the Framework Explains RE Processes in the Year 2001

We can see the development world as an information system (the "virtual companion") about existing and intended change processes in reality. The four-worlds framework can therefore also be fruitful to understand the process of RE in the development world.

From the perspective of formal requirements modeling, the development world, though socially part of the world, must be considered a "meta world" in which change of the remaining worlds and their interrelationships (hence called the "work system") is managed — that is, the world in which the RE process takes place and, in fact, (virtually) creates the different worlds.

Assume that we (as RE researchers) define as our system vision: "find out how a RE process should be supported by an information system". Now, for writing the paper you are reading, we "only" need to know what context this vision shall face in the year 2001 !

Obviously, the *subject world* of our RE system is composed of information about the subject, usage, and system world of the target systems to be designed. We have already discussed their changing nature caused by increasing maturity in the software market and other factors.

The *system world* of our RE system comprises the technical support environment. It will obviously have to live with the legacy of current CASE technology while gradually adopting new technological facilities.

The *usage world* of our RE system has to consider individual, cognitive as well as motivational and social aspects of the RE team. It will heavily be influenced by the answer to the question who will be the players on these teams in the year 2001.

The *development world* of the RE system consists of RE researchers and method engineers. Given the current diversity in research directions, we only see the beginning of a field of study that can actually be called a coherent world. Giving RE a more prominent role in undergraduate computer science education could heavily influence this world.

If we accept the interpretation of the development world as an information system, the model predicts three problem areas which would be faced by a requirements process in the development world[1]. (figure 5):

- The complexity of the individual/cognitive usage aspect leads to the *specification dimension* which describes the degree of completeness of the requirements specification, typically measured against some standard, guideline, or domain model. This dimension has traditionally come from practice, emphasized by large procurement agencies, but has recently seen substantial progress in terms of more formal and detailed guidelines and reference models for various application domains.

---

[1] In [Pohl 1994], they have been called the "three dimensions of requirements engineering" and were independently derived from a survey of the RE literature and practice. The observation how they can be related to the four worlds was made in collaboration with our Swedish colleague Peter Holm.

- The social usage aspect leads to the *agreement dimension* which describes to what degree the members of the RE team agree on the requirements specification. This dimension has mostly been investigated by social scientists, all the way to ethnography.
- The system aspect leads to the *representation dimension* which describes how requirements are technically described, e.g. their degree of formal semantics. This dimension has mostly been studied by computer scientists.

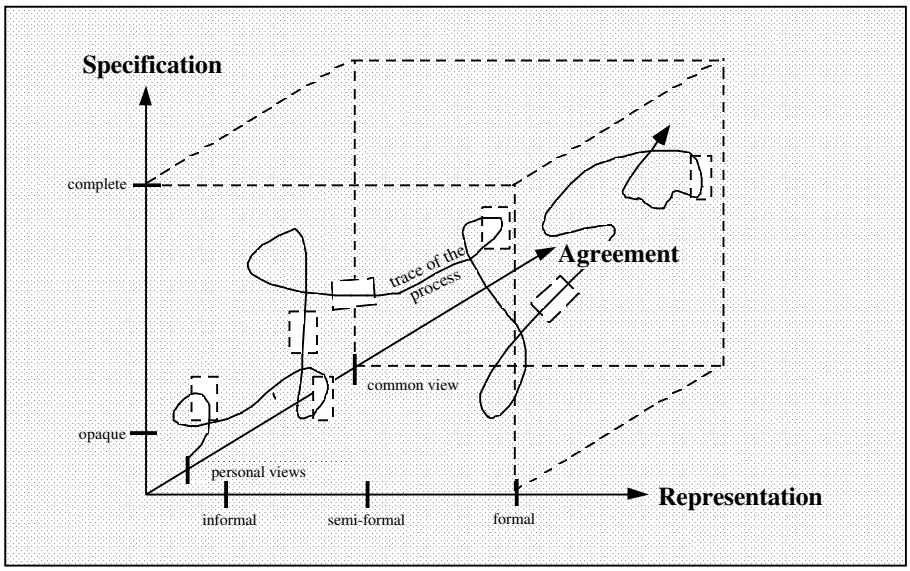

Figure 5 : The three dimensions of the RE process [Pohl 1993]

Desired or actual RE processes can be described according to their effect on these three dimensions, for instance, transformation along the representation dimension, refinement along the specification dimension, or decision-making along the agreement dimension. As indicated by the boxes within the figure, only small chunks of this process are currently understood well enough so that decent guidance can be offered.

The trace of the requirements engineering process is modeled as a curve within the three dimensions (cf. figure 5). The traditional requirements capture process would start somewhere near the origin of the framework with different personal views, opaque system understanding and a completely informal representation. It should ideally end with agreement on a well-understood and formal specification from which development can proceed. Movement may sometimes appear erratic because the outside context of the other worlds changes.

Summarizing, the RE process takes place in the development world and is described along the three dimensions. It gets its sources and must relate its results to the other three worlds, namely the usage, system, and subject world. Further, the definition of the worlds and the relevant information, as well as the definition of the RE process itself, are constrained and guided by the overall system vision, the original reason for establishing the RE process.

3 Requirements Processes in the Year 2001

With the three dimensions of RE processes, we have described a way how the RE process can be traced or defined. The most important demand on future RE methods and tools is that they address *all* of these dimensions rather than just one, as the current methods do. In the sequel, we first look at specific trends along each of the three dimensions, then at the question of integrated process support in the trade-off between vision and context.

One general observation stands out. Requirements specification is both too specific and too general to be attacked directly. On the one hand, RE talks about concepts that come to life only through example scenarios. On the other, each RE process is different so that direct requirements reuse is rarely possible. We can only offer metalevel architectures and domain models which give some generic guidelines or checklists from which a requirements specification can be derived or against which it can be validated. Thus, when moving along the three dimensions of RE, the RE process typically works outside-in, constantly switching between example mode and meta mode to achieve the specification in the middle.

This outside-in way-of-working appears both at the level of the work system and at the meta level of the development world. To help us in designing good requirements models in the three worlds of the "work system", we need to strengthen both the generic structures (reference models, process guidance, etc.) and find methods for the choice of intuitive example scenarios. In the development world, we need both better process and product meta models and much more experience with actual process traces.

3.1 Coverage in the Specification Dimension

The specification dimension addresses the question what the requirements specification should contain. With the shift away from a techno-centric view to the broader perspective indicated by the four-worlds model, the traditional facets of RE models have to be augmented with at least traceability to models of the context (system context, usage context, subject domain context, even development context), and to the vision and the goals of the various stakeholders.

In the sequel, we look at these two aspects in some more detail.

3.1.1 Reference models for the context

The first organizations to consider the specification dimension were large procurement agencies who needed a standardized structure that made development proposals comparable. National and international standards organizations followed with standards, guidelines, and basic reference models. However, these standards lack the detail to be truly useful as a guidance, and are purely geared towards what Lubars et al. [1993] have called *customer-driven RE*.

As the software market matures, we see a trend towards low-cost standard packages for which a traditional RE process is too expensive. For individualization of such packages, we need easy *do-it-yourself RE methods*. Unfortunately, software engineering has left this fascinating field so far to the trade literature and no sound methods or techniques are known to us.

For more complex problems, the solution will typically consist of a configuration of package functionalities, organized according to the business process to be supported. RE can be supported by reference models for various application domains, and each such reference model will offer various options for mapping its individual steps into meaningful configurations.

The first few environments constructed along these lines are appearing on the market. As a successful example in Europe, we mention the ARIS reference architecture for industrial information systems which is offered in combination with related standard software packages [Scheer 1994]. At present, the reference architecture is represented semi-formally, with multiple perspectives organized in hypertext style. To support search for reusable RE models in large requirements information bases with possibly heterogeneous vendors, also to ensure consistency of the generated models and to support analysis of their properties, more formal representations will be needed. The formal theory of domain modeling and its empirical validation will be a subject of research for quite a while [Maiden and Sutcliffe 1992].

3.1.2 Dealing with non-functional goals

While there has been some progress with metrics that *measure the achievement* of nonfunctional goals such as reliability, performance, accuracy, or security, we need to know more about how to incorporate them into the requirements and design process.

One proposed solution is again derived from a computer science concept, namely, the idea of planning with AND/OR graphs [Mylopoulos et al. 1992]. Goals are broken down into subgoals until known satisficing solutions for achieving a subgoal are reached. Goal trees can be used to represent general knowledge about how to reach a certain goal, or to record specific requirements refinement processes.

An alternative approach generalizes the idea of Essential Systems Analysis [McMenamin and Palmer 1984] to deal with domain-specific knowledge about individual types of goals [Derroitte et al. 1994]. It starts with an unconstrained functional description of the intended system, derived from the vision. Then, the methodology takes each goal into account in a particular sequence, by gradually considering ways how it could be violated. For example, in a distributed system, the initial version would assume fully reliable components and communication lines. Then the methodology would investigate if one or the other is in fact likely not to be reliable and, should that be the case, to add requirements for extra measures that ensure reliability anyway.

Two important extensions will be required to make these approaches truly useful. Firstly, the old wisdom of RE to define measurable tests along with requirements has to be taken into account. Secondly, the formal treatment of goals has to be linked to their social meaning, as discussed in the agreement dimension below. To support explicit decision-making based on trade-offs between competing goals, a link between the RE community and research in multi-criteria group decision support systems needs to be established.

3.1.3 Richer specifications through scenarios of virtual reality

Animation of requirements specifications and prototyping of design solutions have been among the more successful techniques to support knowledge acquisition and validation in RE. Typically, this works in combination with specific examples or test cases which instantiate the specification in order to provoke critiques and other comments.

In several domains, we are beginning to see animation and prototyping tools whose scope goes beyond the description of the system itself by including the usage context of the system. We expect that this will eventually cover all the four worlds. Advances in high-performance computing enable us to represent these scenarios in the rich interactive multimedia style of virtual reality technology. However, we have to go well beyond the present state of the art to create and manipulate virtual reality prototypes on-line, for distributed stakeholder teams, as part of a normal, low-cost RE process.

3.2 Broadening the Agreement Dimension

Social aspects of RE have always been a concern for a sub-community within RE research but their investigation has proceeded rather independently from the specification and representation activities. However, in our opinion, these aspects are part of a common whole and need to be investigated as such. For example, if one wants to use ethnographic methods in RE, one still needs to keep the general goals of RE processes in mind -- achieving a sufficiently precise and agreed specification of the envisioned change with limited resources in a limited time [Sommerville et al. 1993]. Conversely, when devising a new formalism, reasoning technique, or process model, its social aspects in terms of understandability, learning effort, or political acceptance need to be considered. Communication technology and cooperation models offer means for improvement -- but the current bottleneck is our lack of understanding of cooperation in real-world RE processes.

Within this general prediction, we should like to point out two specific trends.

3.2.1 Broader range of functions in the agreement dimension

Human interaction plays many different roles in RE:

- *group knowledge building and maintenance:* Team members have different backgrounds and change frequently over time. The RE environment should assist in rapid transfer of acquired knowledge to newcomers, but also stimulate mutual learning of the team as a whole.
- *conflict management:* Conflicts of many different kinds -- ranging from technical concurrency control to contract violation handling, from formal negotiations to hardly structured argumentations -- need to be recognized and dealt with in RE. Technical means help mostly in the detection, rarely in the resolution of these conflicts. However, the ex-post analysis of such conflicts is useful as a rationale for design decisions and their possible revision over time.
- *traceability:* It is often important to trace requirements back to their origin, for example, to determine their criticality. It is equally important to assign *accountability* for the fulfillment of certain requirements to identifiable team members [Ramesh and Edwards 1993].
- *acceptance of guidance:* Regardless of whether guidance in the RE process is provided by oral advice, handbooks, or technical means in the development environment, it is important that arguments are delivered to help the user in deciding among possible solutions or work strategies; argumentation theory says that these should be grounded in identifiable experiences.

Technically, these different functionalities will be available in a comprehensive cooperative work setting which includes synchronous as well as asynchronous, local as well as remote, message-passing as well as repository-oriented modes of cooperation.

3.2.2 Broader composition of RE teams

The four-worlds model indicates that we should broaden the participation in RE teams. Of course, user participation has been a long-standing concern of RE, but remains a difficult issue even just from the viewpoint of the question who are the "users" to be involved.

More importantly, many systems nowadays have impact beyond their usage and development environment. Technically, this impact tends to be related to the "subject world", e.g., the patients in a hospital information system, or the neighbors of a chemical plant to be controlled by the system under design. The widespread availability of computer-integrated communication facilities makes it easier to involve people who are potentially impacted by a system either socially, economically, ecologically, or politically. Subject world models in the specification dimension should include such aspects, at least to the degree needed for identifying the relevant stakeholders. In the representation dimension, ways must be found to help these stakeholders (who are neither domain specialists nor technical specialists!) with analysing the system impact from their own perspective.

3.3 Diversity in the Representation Dimension

The representation dimension of RE has traditionally been characterized by the debate of informal vs. formal, in which informal was considered to be the thing humans can handle, and formal the thing computers (or mathematicians, if a little more difficult) can handle. A conventional computer science wisdom is that the final requirements specification should be "formal", but in practice it is most often not. However, it is also the question whether 3.000 folders of text, graphics, and the like can be understood by humans, regardless whether they are technically "informal", "semi-formal", or "formal".

We believe that the relevant goals of representation are *human intuition and communication,* and *computer-supported analysis*, ideally achieved in combination. There seems to be a good chance of substantial progress over the next few years. It will, however, not result in a single closed formalism for RE but in a well-chosen combination of intuitive (graphical, perhaps multimedial and dynamic) representations with formally understood and largely automatically captured interactions.

The main impediments to achieving this goal are the complexity of the problems and thus of the requirements specifications, and incompatible specialist languages in the different worlds.

3.3.1 Complexity handling with abstractions

The explication of the basic abstraction principles of classification, aggregation, and specialization in the mid-1970´s has had a profound impact on RE methodologies. While the early and now widely used formalisms (structured analysis, entity-relationship, statecharts) just rely on one of these principles (decomposition), more recent proposals also consider the other two.

Originally proposed by [Borgida et al. 1984], stepwise specialization has become an important requirements development strategy not only in object-oriented analysis, but also in the essential systems analysis approach to non-functional requirements mentioned above.

Working with reference models implies *bottom-up reuse and composition of solution objects* rather than *top-down refinement of requirements*. This strategy can also be observed in other engineering disciplines. Engineers tend to think from the available components, their behavior, and their interaction. The actual goals and global functions of the system are just a derived property, not the starting point.

Although this approach has the danger of missing innovative opportunities gained from thinking more seriously about the problem itself, the analogy to engineering practice suggests that RE as an *analysis* task of problems will be replaced by a *synthetic* activity of solution properties. Because the properties of solutions are better known in advance than those of possible problems, this approach does not only facilitate reuse but also the preparation of animations, rapid prototypes, virtual reality, videos with demo applications, and other presentation media useful for human intuition.

This brings us back to the central role of the classification dimension already alluded to in the beginning of section 3. On the one hand, the standardization of knowledge in the specification dimension (section 3.1) emphasizes the need for a meta level representation which proposes the schema for individual requirements projects. On the other, the need for human intuition and communication (section 3.2) requires examples and scenarios which actually instantiate the schema under development. We do see a need for formal requirements language [Greenspan et al. 1994] that -- invisible to the developers and other stakeholders -- manages the relationships between meta level domain schema, actual specification, and instance scenarios of this specification.

3.3.2 Diversity in requirements perspectives

The discussion about the specification dimension showed that a requirements specification involves many different facets, whereas the discussion about the agreement dimension showed that these facets also represent the evolving opinions of different people or subgroups.

The traditional tools tend to concentrate on exactly one of these facets and offer a set of closely related representations around it. Object-oriented analysis is a first step towards a broader spectrum of facets. However, in practice, representing and interrelating these different views without massive training and tool support has proven difficult.

Moreover, the assumption that there is at any given moment a coherent object from which the views are derived forces unnecessary early design decisions concerning the grouping of objects. This may be acceptable in simple cases where RE will become synthetic rather than analytic, as discussed above. But it is arguably incorrect in the presence of multiple stakeholders with different opinions, and in complex analysis tasks where a lot of learning takes place.

We therefore expect that truly distributed approaches to RE, as exemplified by viewpoint-oriented software development [Finkelstein et al. 1993], will prevail over object-oriented analysis.

An existing example of semi-formal viewpoints is the ARIS architecture mentioned earlier. It offers about twenty different representations, organized along two dimensions: one concerning application requirements, system design and implementation perspective; the other concerning the facets of function, data, control, and organization [Scheer 1994]. And this architecture only addresses the system and usage worlds, without considering development and subject world representations -- so we can expect even more viewpoints in need of integration.

3.3.3 Managing static and dynamic relationships between perspectives

The price for the autonomy and realism of distributed requirements engineering is the need for managing the relationships among viewpoints. Meta modeling of static interrelationships has now reached a degree of maturity that a not-too-detailed mapping can be handled between most of the well-known representations. We are beginning to see some of these mappings supported in commercial CASE environments and expect this to become standard in the near future. To integrate the informal legacy documentation, it is important to also structure informal objects with limited costs. Slow but steady progress is made in natural language understanding and generation for RE, but more pragmatic solutions may come from innovative ways of using hypertext.

Even more important, but less understood, are the *dynamic dependencies* among RE perspectives (as well as to other tasks in system engineering, such as design). For example, the relationship between a formal representation and an informal text or video representation could be of type *derived-from, example-for,* or *opinion-about* -- all in either direction! These dependencies result from the process followed in RE, be it a defined process or a haphazard one. In any case, they can only be recorded during process execution, hardly reconstructed afterwards.

Static and dynamic relationships can be partially predicted by, and tested against, the meta level product and process models. The rest still needs substantial manual recording; reducing this effort by concentrating on just the relevant relationships and automating the recording is a major challenge for RE tool developers and method designers. Moreover, the question when and how to invoke the automatable checks also depends on human factors such as the interruption of work, awareness of parallel change, and the like. Solution proposals come both from the groupware field (group awareness in user interfaces) and from the active database community (detached trigger handling) but it may take beyond 2001 until these problems are fully under control.

4 Summary

Three trends seem to drive the direction RE has to take in the future:

(a) The old trick of reasoning backwards from the programming level is not adequate in a world which no longer accepts technology as its main driving force.

(b) We need to live with continuous change and to take advantage from it; and

(c) Maturing of the software market leads to a shift from analytic to synthetic ways of doing RE.

Especially the latter trend also allows better accumulation of experience in standards and reference models, although it is constantly being partially invalidated by the second trend.

Requirements engineering is challenged to take a much more important role in many domains, as its products -- metalevel domain models, conceptual specifications, and their instantiations in prototypical scenarios of virtual reality -- are becoming tools through which interdisciplinary teams can understand, communicate on, and change reality.

A comprehensive understanding of the RE process is critical to raise to the new application challenges. We do have some knowledge at the very fine-grained level of specific methods. There also exist basic approaches at the very coarse-grained level some of which were mentioned in this paper. However, in the middle, there is a big gap that can only be filled by careful analysis of existing and evolving practice. Not only to gain this bitterly needed experience, methods and tools for requirements traceability are central to our understanding what RE actually involves.

Despite progress with many individual problems, our overall understanding of the requirements engineering process is quite limited. RE has to overcome the current fragmentation of research interests along the different dimensions of RE, as well as the divergence between research and practice. This paper has therefore placed a major emphasis not only on making individual predictions about the future of RE, but also proposing a coherent framework within which these predictions can be positioned and combined.

Acknowledgments. The concepts presented in this paper result from a collaborative effort of the RE research team in Aachen. Many members of the NATURE project, led by Janis Bubenko, Panos Constantopoulos, Colette Rolland, and Alistair Sutcliffe, also contributed to the ideas presented in this paper. Thanks are also due to Eric Dubois, Anthony Finkelstein, John Mylopoulos, and Bala Ramesh for many stimulating discussions.

References

Avison, D.E., Golder, P.A., Shah, H.U. A tool kit for soft systems methodology. *Proc. IFIP WG 8.2 The Impact of Computer Supported Technologies on Information Systems Development*, Minneapolis, Mn, June, 1992, 273 - 287.

Blanning, R.W., Holsapple, C.W., Whinston, A.B.: Special Issue on Model Management Systems. *Decision Support Systems 9,* 1 (1993).

Borgida, A., Mylopoulos, J., Wong, H.K.T.: Generalization/specialization as a basis for software specification. In Brodie/Mylopoulos/Schmidt (eds.): *On Conceptual Modeling,* Sprinter-Verlag 1984, 87-114.

Brackett, CASE tools for requirements analysis and software design. Tutorial notes, *12th Intl. Conf. Software Eng.,* Nice, France, 1990.

Brodie M.L., Ceri S.: On intelligent and cooperative information systems: a workshop summary, Intelligent and Cooperative Information Systems 1, 2 (1992), 249-289.

Chen, P.: The entity-relationship model. *ACM Trans. Database Systems 1,* 1 (1976).

Dardenne, A., Fickas, S., van Lamsweerde, A.: Goal-directed concept acquisition in requirements elicitation. *Proc. 6th IEEE Workshop System Specification and Design*, Como, Italy, 1991, 14-21

DeMarco, T.: *Structured Systems Analysis,* Yourdon Press 1978.

Derroitte, M., Dubois, E., Du Bois, P., Petit, M., Michel, J.-P.: Requirements definition of manufacturing systems. Working paper, University of Namur, Belgium, 1994

Fickas, S.: Language support for the specification and development of composite systems. *ACM Trans. Programming Languages and Systems* 9, 2 (1987), 198-234

Finkelstein A., Gabbay, D., Hunter, A., Kramer J., Nuseibeh, B.: Inconsistency handling in multi-perspective specifications; *Proc. 4th European Software Eng. Conf.,* Garmisch, Germany, 84-99.

Goguen, J.A., Linde, C.: Techniques for requirements elicitation. *Proc. IEEE Symp. Requirements Engineering,* San Diego 1993.

Greenspan, S., Mylopoulos, J., Borgida, A.: On formal requirements modeling languages. *Proc. 14th Intl. Conf. Software Engineering,* Sorrento, Italy, 1994.

Harel, D.: Statecharts -- a visual formalism for complex systems. *Science of Computer Programming 8* (1987), 231-274.

Jarke, M. (ed.): *Database Application Development with DAIDA*. Springer-Verlag 1992.

Jarke, M., Pohl, K.: Establishing visions in context: towards a model of requirements processes. *Proc. 12th Intl. Conf. Information Systems,* Orlando, Fl, 1993.

Lubars, M., Potts, C., Richer, C.: A review of the state of the practice in requirements modeling.*Proc. IEEE Symp. Requirements Engineering,* San Diego 1993.

Maiden N., Sutcliffe A.: Exploiting reusable specifications through analogy. *Comm. ACM 35*, 4 (1992), 55-64

McMenamin, S.M., Palmer, J.F.: *Essential System Analysis.* Yourdon Press 1984

Petrie, C. (ed.): *Enterprise Integration Modeling,* MIT Press, Cambridge, Mass, 1992.

Mylopoulos, J., Chung, L., Nixon, B.: Representing and using nonfunctional requirements: a process-oriented approach. *IEEE Trans. Software Eng. 18,* 6 (1992).

Pohl, K.: The three dimensions of requirements engineering: a framework and its application. *Information Systems 19,* 3 (1994).

Ramesh, B., Edwards, M.: Issues in the development of a requirements traceability model. *Proc. IEEE Symp. Requirements Engineering,* San Diego 1993.

Scheer, A.-W.: *Enterprise-Wide Data Modeling*, Springer-Verlag, 4th edn., 1994.

Sommerville, I., Rodden, T., Sawyer, P., Bentley, R., Twidale, M.: Integrating ethnography into the requirements engineering process. *Proc. IEEE Symp. Requirements Engineering,* San Diego 1993.

Yoshikawa, H., Goossenaerts, J. (eds.): *The Design of Information Infrastructure Systems for Manufacturing* (Proc. JSPE-IFIP 5.3 Workshop, Tokyo 1993), North-Holland 1994.