CREWS Report 99-01

To appear in
Requirements Engineering Journal, 1999

Scenario Management: An Interdisciplinary Approach

by

Matthias Jarke
RWTH Aachen, Lehrstuhl Informatik V, 52056 Aachen, Germany
jarke@informatik.rwth-aachen.de

X. Tung Bui
University of Hawaii, College of Business, 2404 Maile Way, Honolulu, HI 96822, USA
tbui@busadm.cba.hawaii.edu

John M. Carroll
Virginia Tech, Center for Human-Computer Interaction, Blacksburg, Va 24061-0106, USA
carroll@cs.vt.edu

Scenario Management: An Interdisciplinary Approach

Matthias Jarke
RWTH Aachen, Lehrstuhl Informatik V, 52056 Aachen, Germany
jarke@informatik.rwth-aachen.de

X. Tung Bui
University of Hawaii, College of Business, 2404 Maile Way, Honolulu, HI 96822, USA
tbui@busadm.cba.hawaii.edu

John M. Carroll
Virginia Tech, Center for Human-Computer Interaction, Blacksburg, Va 24061-0106, USA
carroll@cs.vt.edu

Scenario management (SM) means different things to different people, even though everyone seems to admit its current importance and its further potential. In this paper, we seek to provide an interdisciplinary framework for SM from three major disciplines that use scenarios – strategic management, human-computer interaction, and software and systems engineering – to deal with description of current and future realities. In particular, we attempt to answer to the following questions: How are scenarios developed and used in each of the three disciplines? Why are they becoming important? What are current research contributions in scenario management? What are the research and practical issues related to the creation and use of scenarios, in particular in the area of requirements engineering? Based on brainstorming techniques, this paper proposes an interdisciplinary definition of scenarios, frameworks for scenario development, use and evaluation, and directions for future research.

1 Introduction

A *scenario* can be defined as a description of a possible set of events that might reasonably take place. The main purpose of developing scenarios is to *stimulate thinking* about possible occurrences, assumptions relating these occurrences, possible opportunities and risks, and courses of action. Given the renewed interest in scenarios, recent surveys of scenario research and practice suggest that scenarios management means different things to different people, even within disciplines [5, 9, 22, 58, 73]. Clearly, however, scenarios are not just abstract artifacts but a critical representation of the realities as seen by those who create them.

Historically, researchers and practitioners from other disciplines have long used scenarios. The scenario concept came into research via military and strategic gaming but found its origin in theatrical studies [4; 7]. Economists have successfully used scenarios for long range planning. Management scientists use them for strategic decision making. Policy makers use them to weigh the consequences of their actions. Scenarios are also used as a means to examine the interplay among economic, social, and technological issues.

Since the late 1980's, researchers in HCI (Human-Computer Interaction) use scenarios as representation of system requirements to improve communication between developers and users. Software engineers look at scenarios as an effective means to discover user needs, to better embed the use of systems in work processes, and to systematically explore system behavior – under both normal and exceptional situations. In the past few years, scenarios have gained enormous popularity through Ivar Jacobsen's Use Case approach which is now feeding into the efforts to establish a Unified Modeling Language (UML) for systems engineering based on the object-oriented approach [30].

The great variety of scenario usage in many different disciplines is probably the reason for the lack of a unified research framework of the field of scenario management. Indeed, despite some efforts to bring together various SM approaches in the last few years [9], there is not yet a coherent scenario management research community. The purpose of this paper is to provide a synoptic view of scenario management. The proposed interdisciplinary framework should shed some insights for researchers and practitioners for their own work. This paper draws on background research of the authors in their respective fields, as well as on findings of a workshop the authors organized at Dagstuhl Castle, Germany, in February 1998, in an attempt to promote mutual understanding and research focus across disciplines[1].

The paper is organized as follows. We first propose an interdisciplinary framework characterizing the role of scenarios in the context of change management. Then, for each of the three disciplines of HCI, requirements engineering, and strategic management, we discuss their theoretical perspective and practical results on what scenarios are good for, what properties they should have, and how they should be managed. We next summarize questions and results concerning four research issues the workshop participants considered crucial across disciplines. Finally, we synthesize the main research issues claimed by participating experts, and the research methods adopted for evaluating these claims. The goal is to provide a comprehensive research method that could be used as a guide to foster interdisciplinary research in scenario management.

2 Scenarios as Enablers of Change

Why have scenarios become so popular in the 1990's? Many proponents claim that scenarios are taking center stage in a problem area that appears prominent in all the disciplines represented at the workshop (and in many others): *management of change*.

In strategic management, turbulence of the organizational environment is driven by the interacting forces of globalization and technological progress [29]. In systems engineering, the need for more customer orientation, as well as continuous adaptation to organizational, environment, legal, and technological change places new demands on requirements engineering, system architectures, and

[1] The initiative for this effort stemmed from a European Long Term Research Project within the ESPRIT program of the European Union, called CREWS (Cooperative Requirements Engineering With Scenarios), which is coordinated by the first author. Workshop participants listed in the acknowledgments at the end of this paper have contributed to these results in the stimulating atmosphere of Dagstuhl Castle. The workshop was also organized in cooperation with the IFIP Working Group 2.9 (Requirements Engineering), the RENOIR Network of Excellence, and with the RE groups in the British Computer Society and the German Informatics Society. The workshop convened leading researchers and practitioners from various disciplines, in order to cross-examine the *effectiveness and efficiency of using scenarios* as a tool for modeling, design, development and (technical and organizational) implementation. A corollary theme was the question how the three different disciplines *manage scenarios as complex artifacts* throughout the planning and systems lifecycle. Selected individual research results by the workshop participants and other researchers have been collected in special issues of the *IEEE Transactions on Software Engineering* [31] and the *Requirements Engineering Journal*

traceability of development processes [17]. With the usability goal in mind, HCI researchers for a long time have paid attention into improving user-designer communication in change situations. For example, they focus on problems such as design fluidity, the mutual influence between user tasks and system object models, and rapid progress in interface technologies [9].

In this section, we view scenarios as tools to enable change in both the HCI and Information Systems communities, with a varying degree of interpretation and use.

In particular, we refer to a framework used for a long time in system evolution [45, 12] and made explicit in [33] as in figure 1a. According to this framework, change management involves four basic tasks:

(a) re-constructing the concepts and rationale behind the current system,
(b) defining the desired change at the conceptual level,
(c) implementing the changed concepts to reach the new system while
(d) taking the legacy context into account.

In practice, the cycle in Figure 1a is but one step in a *continuous change process*. Traceability across multiple changes in reality and concepts is essential. This is also reflected in decision theory, which often considers multiple related decisions or sequences of decisions.

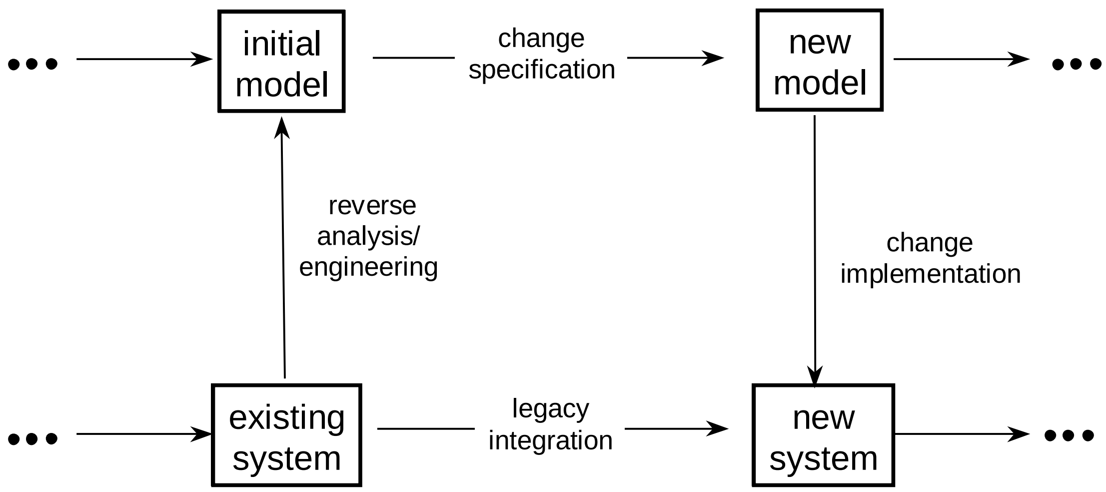

Figure 1a: Model-based change management [33]

However, this focus on modeling has faced two major critiques:

(a) How can we make sure that the models enable a sufficiently deep and *shared understanding* of the contextual issues, and preserve it over a long period of time?
(b) How can we capture and preserve the “vision” that *drives and focuses* the change cycle [33]?

This kind of critique of purely model-based approaches is not new. In Operations Research, it goes back to the 1970’s [1, 14]. Scenario-based approaches provide one promising response to these critiques. By offering a down-to-earth middle-level abstraction between models and reality, they promote shared understanding of the current system, and joint creativity towards the future. But the number of possible scenarios is even greater than the number of models for a given system. Therefore, many researchers have begun to recognize the need to make the goal hierarchies driving scenario-based processes explicit (e.g., [2, 16, 49]).

By combining these two extensions, we obtain a basic framework for scenario-supported change processes, as shown in Figure 1b. In this figure, *current-state and future-state scenarios* are placed as intermediate abstractions between immediate reality and abstract models. *Goals* (usually refined into more detailed *requirements*) focus the definition of change and the selection of scenarios, but also conversely, scenarios may help in goal discovery.

This framework covers a broad variety of different techniques found in research and practice. For example, market-oriented development organizations tend to follow a development cycle that contains just general constraints (e.g., "must have a market of at least one million copies") and *change envisioning* via future scenarios, but no models. The operational *goal* is to achieve the agreed collection of future-system scenario. Re-interpreted as test cases, these scenarios are complemented by concrete (initially mock-up) system usage demos at design and implementation stage.

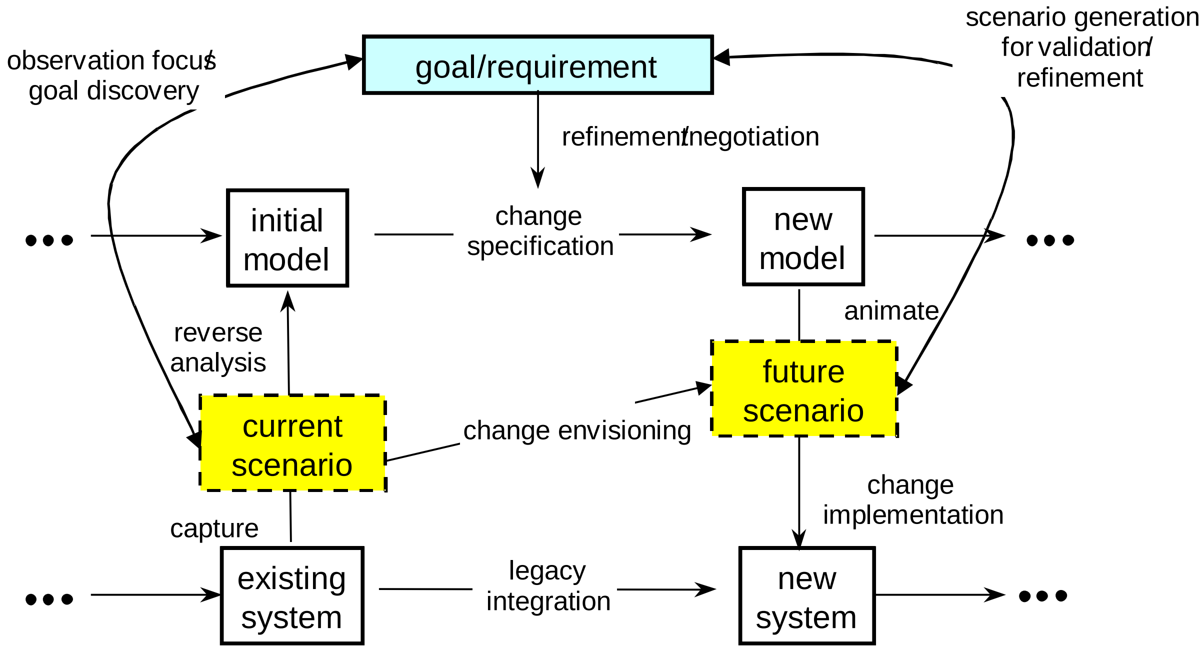

Figure 1b: Goal-driven change process with scenarios

At the other extreme, decision theory abstracts reality to the values of a few variables in some equation system which is then optimized according to multiple objective functions under uncertainty. Scenarios in this setting are still highly abstract, namely the decisions and outcomes resulting from a particular set of assumptions about future (system-external) events.

To illustrate the full process shown in Figure 1b, consider the scenario-based requirements management tool suite under development in the CREWS project. In CREWS, *current state scenarios* are technically supported by real-world scenes captured in multimedia. The capture of these scenes, and their abstraction to conceptual models, is driven by *hierarchically organized goals*. Traceability is maintained to all the artifacts captured this way, in order to ground a persistent, shared understanding [25], that will enable stakeholders later on to recognize modeling errors, impacts of changing assumptions, etc. For the special case of textual scenario descriptions, additional support is provided through structuring/authoring guidelines and partially automated natural language understanding and indexing [57].

To define change and future system elaboration, the CREWS tools support user-developer teams firstly by checklists to create broad-brush *future state scenarios* of normal-case and exceptional case behavior of system interaction and system environment based on reusable, partially domain-dependent classes of functional and non-functional requirements [67]. This is complemented by more automated animation support for critical portions of system interaction and system organization that allows users and developers to investigate consistency and impact of the system requirements definition in a simulated usage setting [27].

Scenarios can also be categorized according to the content scope they address. In the most frequent case (e.g., in the Use Case approach [30]), scenarios focus on the *interaction* between the system and its environment (case B in Figure 2). However, scenarios can also address an organizational work *context* [39] without considering the system to be designed (C). Equally, they can represent the internal *interplay of system components* within system (A). Interaction scenarios, in turn, can be studied in an in-bound direction (*what constraints does the environment place on the system?)* or in an outbound direction (*what impact will the system have on its environment?).* Inbound interaction scenarios are called *blackbox scenarios* if they do not consider any system internals, whereas combinations of interaction with internal scenarios are called *whitebox scenarios*.

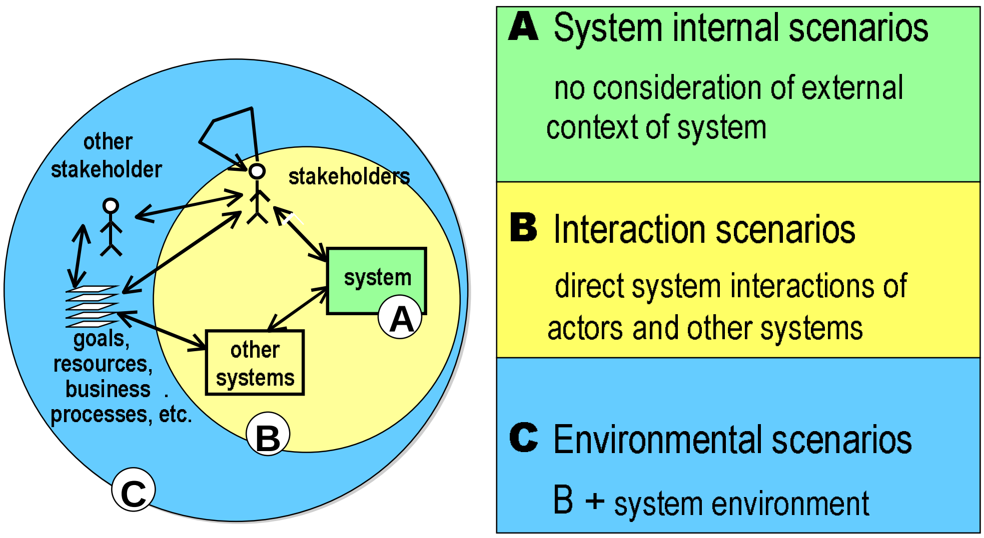

Figure 2: Scenario types and system boundaries

Scenario research and practice in each discipline have explored specific kinds of scenarios, and theories underlying their usage, in a complementary but partially overlapping manner. In the next section, we shall review their frameworks and results, before focusing on shared interdisciplinary research issues and agendas.

3 Three Disciplinary Perspectives on Scenario Management

An enormous and difficult-to-classify number of approaches and techniques for scenario-based analysis and scenario management exist in several fields. Many of these emerge in a haphazard fashion, formed by some methodological prejudices or theories from one specific domain but often without a broad grounding. While general frameworks such as the one proposed in the previous section may be useful for achieving the necessary conceptual overview, we believe that *theories* that allow us to evaluate scenario-based approaches in a systematically robust manner are still missing. The main question remains unanswered:

(under what circumstances) is my scenario-based approach reasonable?

An overall theoretical basis does not, and may not for a while, exist. However, many disciplines can bring about certain facets. The Dagstuhl workshop brought together experts with considerable experience in using and managing scenarios from three broad fields: *human-computer interaction* (HCI) where scenarios have been used for interface specification; *requirements engineering* where scenarios are used to illustrate current problems as well as future reality of software-intensive systems; and *strategic management* where scenarios have been used to explore a purposeful set of alternative futures. The workshop included brief tutorials intended to clarify the perspectives and main research results within each of these research areas. We also asked selected participants which theories from their area of expertise they would consider most relevant for designing and evaluating scenario-based techniques. The scope of theories mentioned in this section is illustrated in Figure 3.

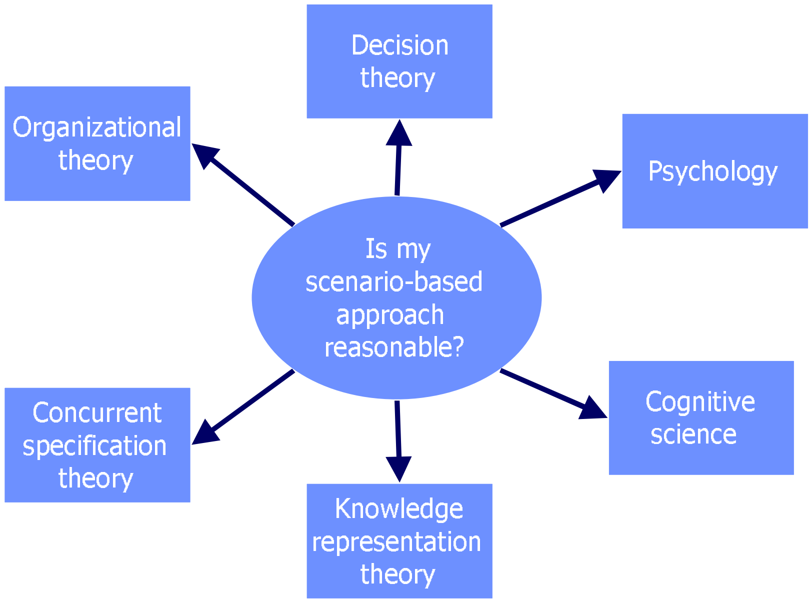

Figure 3: Theories related to scenario management

3.1 The Role of Scenarios in Human-Computer Interaction

In human-computer interaction (HCI), scenario-based design has emerged as a paradigm reconciling a longlasting methodological conflict : formal modeling approaches proved too narrow to provide effective guidance to designers, whereas purely experiential approaches could not be verified, replicated, or explained. Scenarios were also proposed as working design representation of user experiences with, and reactions to system functionality in the context of pursuing a task [11].

In HCI, scenarios therefore describe key situations of use in the form of narrative, with the goal of making design objects concrete. Designers and users develop, and reason about, these descriptions throughout the lifecycle in a variety of media, purposes, and views, either to discuss existing options or to stimulate imagination. Five key properties of scenarios motivate their widespread use in HCI [10]:

1. First and foremost, scenarios focus design efforts on *use.* What people can do with the old/new system, and the consequences for themselves and for their organizations, is described and analyzed prior to detailing the system functions and features that enable this use. Scenario descriptions of use provoke designers to reflect upon the concrete circumstances and experiences of users throughout the design process.

2. Scenarios *suspend commitment* but support *concrete progress*: They vividly document an analysis of task essentials, explaining why a system is needed by showing what it is used for. They also specify an analysis of design alternatives, by detailing how the system is used. But scenarios are also rough: they are incomplete; they suggest alternative approaches and "what if?" lines of reasoning. Iteration between requirements definition and scenario-based envisioning is rapid, easy, and cheap.
3. Scenarios provide a task-oriented design decomposition that can be used from *many perspectives*, including usability consequences and trade-offs, usability specifications and iterative development, and manageable software design object models. They provide a framework of concrete user tasks for developing design rationale and documentation, describing causal relationships implicit in a design and providing a analysis to which evaluation data can be subsequently adduced.
4. Scenarios *codify design knowledge* as a "middle-level" abstraction, in the term of [47]. This makes them somewhat less grand as science, but it allows the integration of design knowledge in a form more suitable for reuse.
5. Finally, scenarios are an ideal medium for *participatory design*: They allow design discussions to be carried out in a common language. Users may have difficulty describing their goals and visions in the language of features and functions, as traditional problem description languages and functional specifications are a language barrier to users. But all stakeholders in a design project can "speak" the language of scenarios.

These five points are summarized in Figure 4 (starting from the top-left).

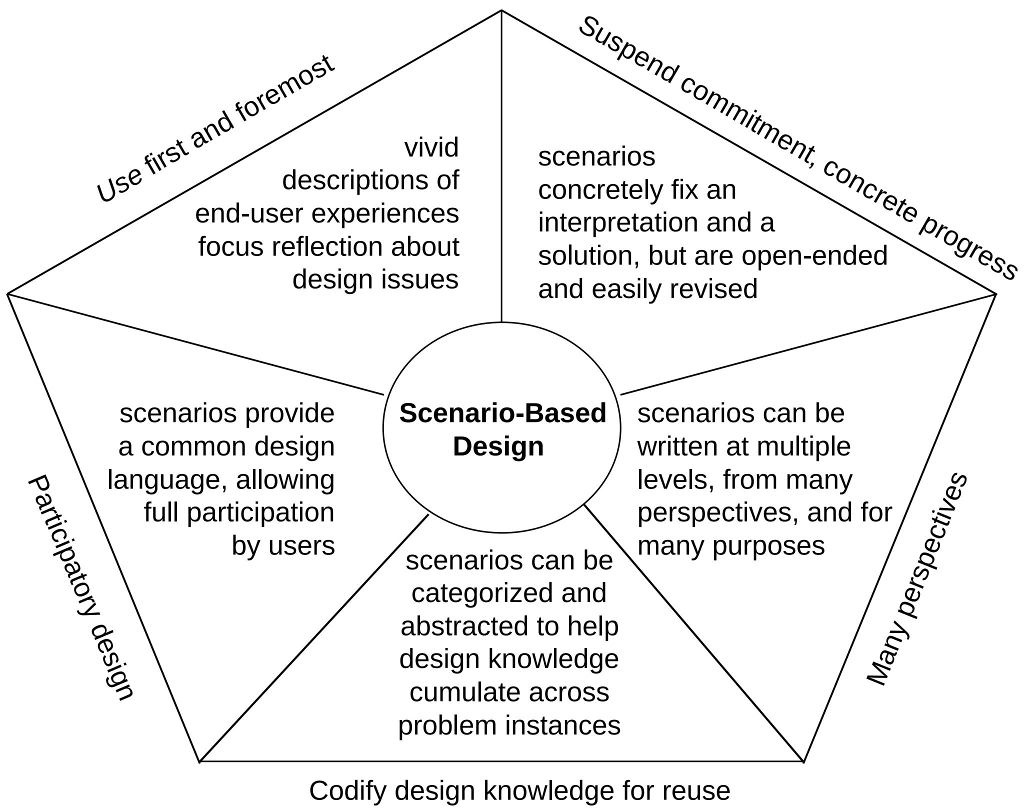

Figure 4: Five key properties of HCI scenario-based design

These properties of scenarios suggest an *"ideal" scenario-centric process* in which the design of a system is influenced by scenarios from two directions (Figure 5). On the requirements side, observation scenarios, selected according to the orienting goals of the design project, help identify issues and criteria, which can then be validated against further observation scenarios and against scenario abstractions reused from prior work. This subprocess drives the design through the definition of needs and opportunities. Once a prototype (including a user interface) is available, *scenario-based evaluation* can complement and validate the requirements work. Observation scenarios allow the analysis of the transformed situation and thereby the evaluation of the prototype. From such evaluations, further design abstractions can be induced, which collectively form a theory that informs future requirements processes and thus simplifies subsequent design.

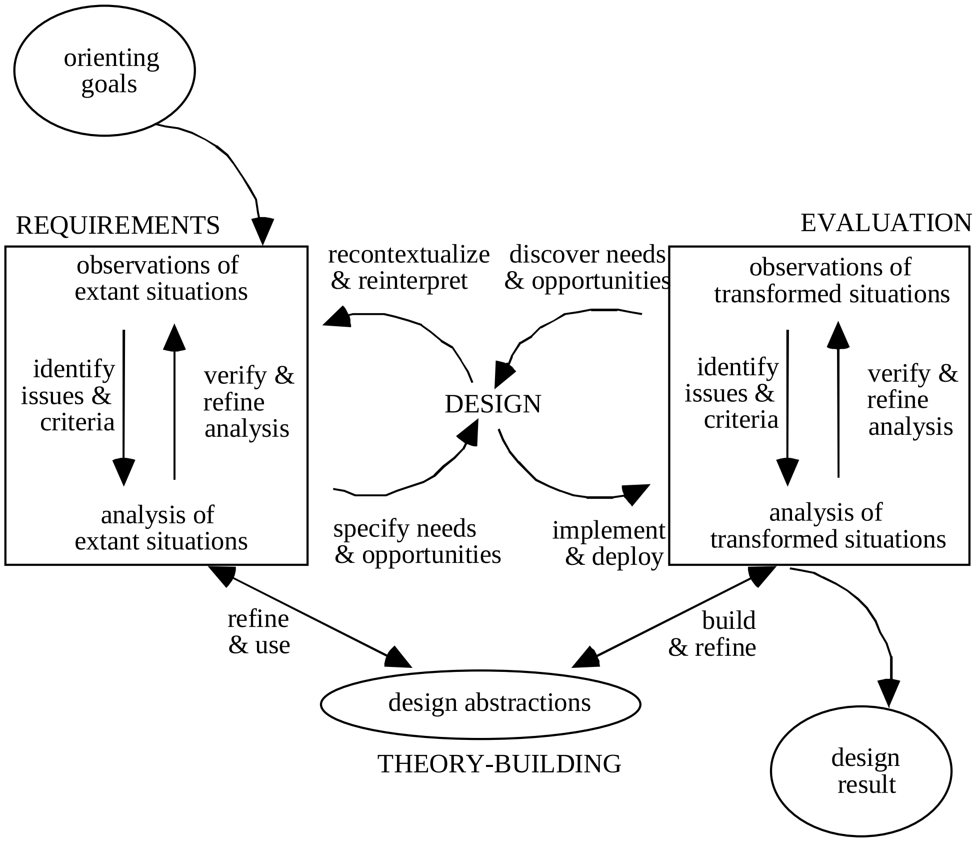

Figure 5: A desirable scenario-based design process

At the Dagstuhl workshop, this proposal evoked an interesting discussion of the relationship between requirements engineering and HCI: Several people argued that this process *is* a description of scenario-based requirements engineering. A subsequent panel discussion focused on the relevant cognitive science theory that underwrites scenario-based design. Fairly diverse sources of theory — ranging from Sigmund Freud and Levi Strauss to Julian Orr and Roger Schank — agree that narratives are privileged cognitive structures. Carroll suggested that this convergence could be explained by *principles from cognitive psychology*:

- *Concrete material* is cognitively accessed and interpreted more easily and more thoroughly. For example, people can remember a prototypical instance far better than they can remember the definition of the category to which that instance belongs [46; 59].
- *Incomplete material* is elaborated with respect to one's own knowledge when it is encountered. This process of elaboration creates more robust and accessible memories, relative to memories for more complete material [74].
- *Narrative structures* appear to be universally understood and employed by people from all cultures; within cultures, narrative form became extremely articulated and semantically overloaded [23; 42; 55].

- When people communicate, they follow a convention that has been called the *given-new contract* [26]. They first summarize or allude to relevant background information, and then present what is novel. This structure cues the listener or reader as to what the speaker or writer considered to be novel information, easing comprehension and analysis. Narratives, including scenarios, tend to follow this structure.
- Scenarios can address the *representational bias* in human cognition: People tend to overestimate the relevance of things that are familiar to them [35; 69]. This tendency is extremely difficult to mitigate, but can be managed by making exceptional patterns vivid. Narratives represent an excellent vehicle for managing this phenomenon.

Domain-specific theories of how scenarios are used by designers in the lifecycle of systems could lead to significant progress beyond the current ad-hoc development of scenarios with very modest guidance of scenario authoring in the small. Using theories which determine the content, scope and granularity of domain-specific patterns of system usage, specific scenarios could be rapidly and systematically developed [67]. Such theories would have to be developed bottom-up from particular scenario analyses, though they might have more general roots in script theory, natural categories, schema theory, theories of reasoning by analogy, and the like.

3.2 The Role of Scenarios in Software and Systems Engineering

Through the emergence of object-oriented software engineering [Jacobsen 1995], scenarios have gained enormous popularity in the practice of software engineering. As often in the SE field, research follows with some delay. The CREWS project has conducted surveys of scenario research and practice, with an emphasis on the requirements engineering task within software and systems engineering. Conceptually, an information system can be defined as being composed of four interacting basic perspectives or "worlds" [32]. As a product (Figure 6), an information system can be modeled as a human-machine *system* which provides *users* information or control over a *subject* domain (often called Universe of Discourse) which is denoted by the information objects. Users can be studied in two complementary roles: as individuals with cognitive problems of understanding, and as social units exploiting the information system as a communication and coordination medium to support their tasks, interests, formal roles, etc.

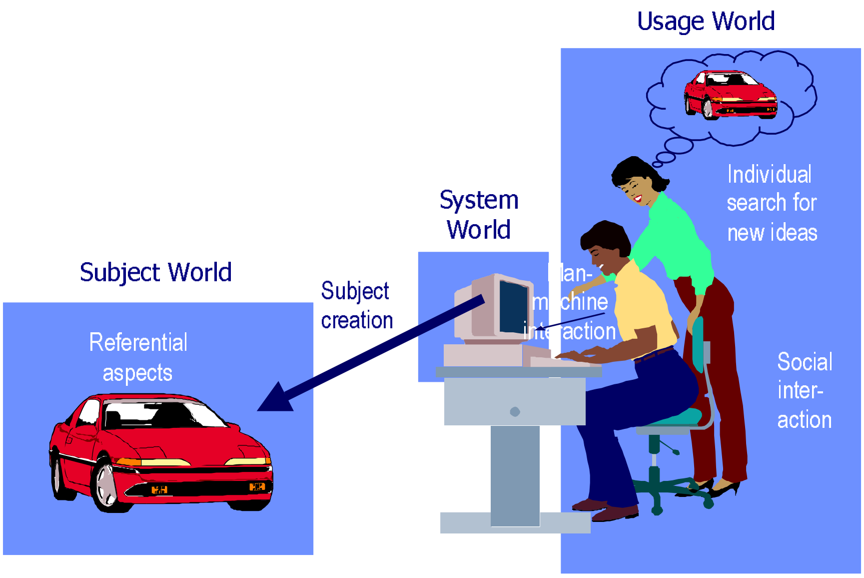

Figure 6: A conceptualization of an information system as a process of creating subjects

The product triplet <*system world, usage world, subject world*> is subject to an evolutionary change process in the *development world*. At a meta level, the development world can be seen as a *change information system* (Figure 7). It controls the product information system as its subject domain, has the development team as its users and the development environment with its intermediate artifacts as the system itself. Scenarios are a particular kind of design artifact, intended to facilitate shared understanding (in the development world) of the target system, its interaction with users and subject domain, and its larger context, as already illustrated in Figure 2.

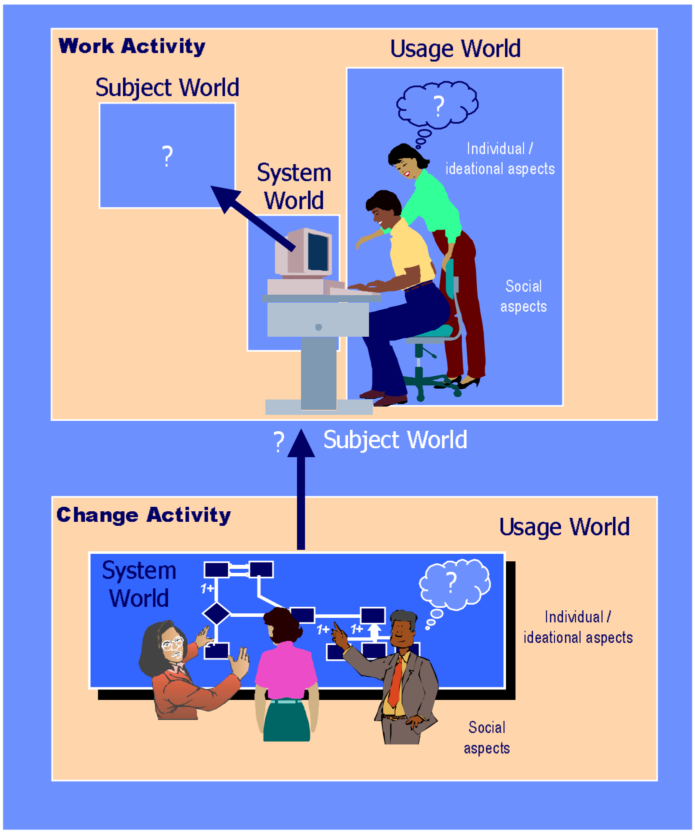

Figure 7: Change management as a meta information system

A review of the scenario literature [58] shows that this framework also provides a reasonable starting point for classifying scenario-based approaches. Looking at the work activity as the subject domain and scenarios as one kind of development system artifact in the change activity, we obtain four views:

- What part of the work activity is captured in a scenario (*content view*) ?
- How is it represented in the development system (*form view*) ?
- For what usage in the design process is it captured (*purpose view*) ?
- How is it developed and evolved (*life-cycle view*) ?

The resulting scenario classification framework is shown in Figure 8. This framework is also intended to serve as a basic structure in which one could manage knowledge about scenario-based approaches, and actual scenarios, in a method repository (cf. the section on method integration of scenarios, below). In [Rolland et al. 1998], each of these four basic views is further elaborated into detailed facets, and applied to classify more than a dozen well-known proposals in the literature, including, for example, Jacobsen's initial Use Case approach and various proposed extensions.

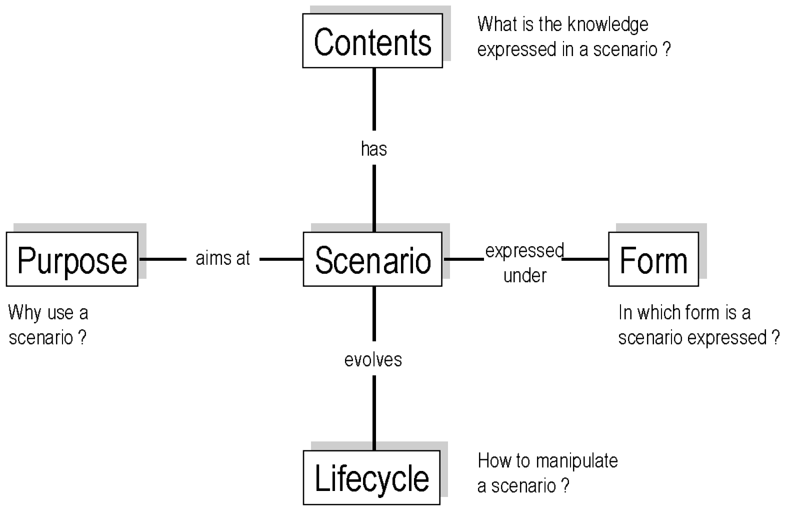

Figure 8: The CREWS framework for describing scenario-based approaches [58]

Concerning the *form view* in Figure 8, the question arises how scenarios are related to *formal specification models*, as they are proposed for requirements and high-level design of safety-critical systems. Consistent with Figure 1b, researchers in knowledge representation and requirements engineering see scenarios as an intermediate artifact between specifications of system behavior (class level) and fully instantiated lives of example objects which obey these specifications (traces).

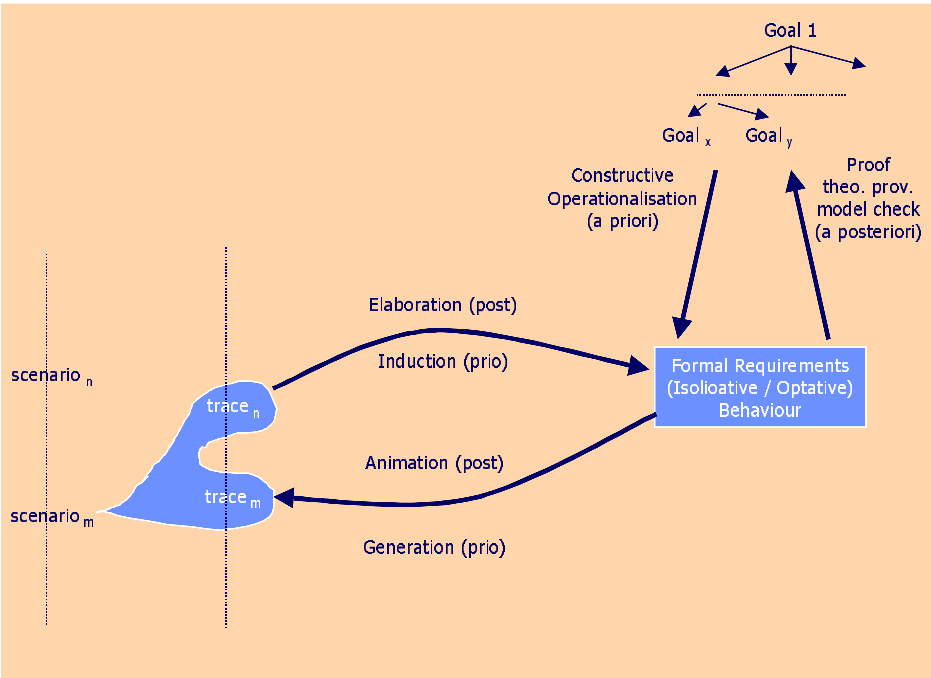

Figure 9: Scenarios as groupings of traces linked to formal specifications

Figure 9 elaborates the formal relationships between scenarios and models in Figure 1b. Scenarios represent *abstracted groupings of traces* which highlight specifications from a particular viewpoint considered important by designers or users. They thus speed up the interaction between designer and customer in requirements elicitation and validation. They also provide a middle ground between declarative and operational styles of specification. Considering the purpose view of scenarios, this approach allows a formally supported treatment of scenarios and their relationships to specifications, such as elaboration, property induction, generation and animation [28, 27, 41].

Perhaps surprisingly, scenarios and formal methods are not as far apart as one might think. In fact, *theories of reactive systems* (typically based on process algebra or temporal logics) start from sets of scenarios/traces as boundary conditions on system behavior. However, without a theoretical framework ensuring coherence, scenario-based specifications can deal with individual scenarios only. Without a notion of atomicity of actions, they run, for instance, into the danger of overlooking critical interference between concurrent execution of multiple scenarios. A theoretical framework studied in [38] comprises four major elements:

(a) An execution model for specification simulation offering atomicity and non-determinism
(b) A language for multi-partner actions which allows the modeling of collective behaviors at arbitrary levels of abstraction
(c) A design approach based on refining and composing such specifications
(d) A temporal logic of action for reasoning about their meaning.

Using these ingredients, operational specifications can be formally understood as canonical formulas in a Temporal Logic of Action [40] that can be derived incrementally. At each stage, the resulting scenarios can be both simulated and formally reasoned about, and the achieved properties are preserved when more detail is added or the level of abstraction is lowered.

In addition to serving as a classification scheme for the scenario research literature, the framework of Figure 8 was also elaborated into a set of questionnaires and semi-structured interviews in order to determine the *state-of-practice* in scenario-based software engineering [73]. More than 30 projects, covering a variety of sizes and application domains, were studied. Comparing the research literature with the situation found in practice, we observe that there is insufficient overlap and thus a need for improved two-way communication:

- *Scenario content*, while focused on scenarios of interaction between users and systems, also extend to environment scenarios describing the context independent of the system and, on the other extreme, to interaction scenarios between distributed systems components, e.g. in technical systems such as telecom applications. These different scenario content types, elaborating the interplay between usage world and system world, have already been displayed in Figure 2.
- While much research has focused on aspects associated with the *form view*, this view has so far received relatively little attention in practice, as most projects use (at best structured) textual representations. Users expect researchers to take this seriously and to provide authoring guidance for the structured text scenarios.
- The number of *scenario purposes*, and the impact on scenario usage on the whole project, was much bigger than expected from the research literature. While research did discuss the application of scenarios for making abstract models concrete, to reach partial agreement and consistency of understanding, practitioners also use scenarios as a decomposition mechanism for managing complex projects, as a linkage mechanism between development phases, and as design aids and boundary conditions for object models.
- Probably as a consequence of this wide-ranging usage, *the life-cycle* view of scenarios found in practice is also much more complicated than covered by current research. The structuring and evolution of scenarios are seen as major problems, especially if multiple views on scenarios (e.g., developer, user and manager view on the same scenario) and the traceability of scenarios across project phases (e.g., interplay between scenarios and prototypes, elaboration of scenarios into test cases) are considered. In these latter areas, practice has no solutions but poses this as an important challenge to research and vendors which is currently not addressed adequately.

In summary, problems and solutions found in RE partially try to bootstrap from experiences in fields that have started a bit earlier, such as HCI, but are faced with significant additional challenges due to the large size and long duration of many complex software engineering projects. Linkages to strategic scenarios have hardly been considered yet (cf. section 4.1).

3.3 The Role of Scenarios in Strategic Management

Strategic management defines the basic directions an organization wants to go. To identify these directions, and to move towards them, strategic managers make decisions that are *purposeful but novel*, under conditions of *ambiguity and uncertainty*, and result in *high impact outcomes*. This informal, organization-theory interpretation of strategic management sees scenarios as narrative descriptions which define, in a visionary sense, the outcome of strategic decisions. In a narrow sense, a scenario is a description of future situations of an organization [5]. In the broad sense, it consists of (1) assumptions and hypotheses about processes and actions, (2) models and procedures used to determine the elements of the scenario, (3) quantitative and qualitative factors, and (4) decisions, situations and interpretations.

In decision theory [54], scenarios are the answer to the combinatorial explosion of strategic options in decision trees. As Figure 10 shows, a decision tree typically illustrates a “game against nature”, i.e., actions react to expected events while events impact the outcome of actions [Raiffa 1968]. A scenario is then simply a (not necessarily complete) set of conditional actions consistent with the decision tree, e.g., $<a_o, a_1 \text{ if } e_{11}, a_2 \text{ if } e_{12}>$ in Figure 10.

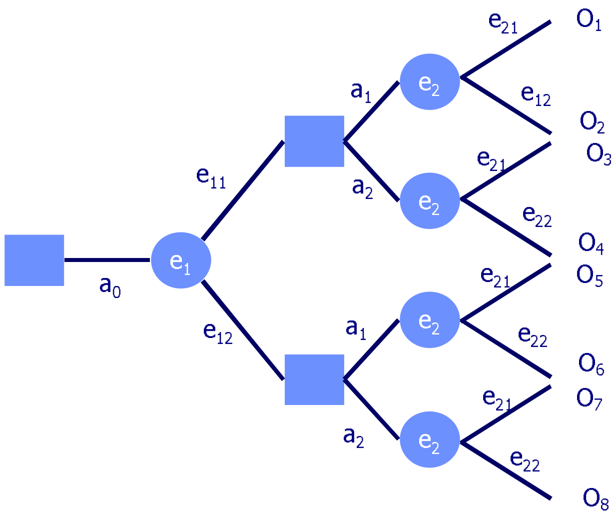

Figure 10: Example of a decision tree

Case-based reasoning from artificial intelligence is considered a promising strategy for linking decision trees to scenarios based on similar situations analyzed in the past [37,68]. However, even more strongly than in HCI or software engineering, the risk of *bias in scenario selection* is ever present in strategic decision making; numerous active measures are being researched to counter it. Decision-makers have a strong tendency towards optimistic assumptions, ignorance or overemphasis of small-probability scenarios, over-emphasis of recently occurred problems, continuation of present trends, etc. [43, 70].

Assuming a reasonable set of scenarios has been found, there are at least two different ways how they are actually used beyond simple envisioning. Of particular interest in strategic decision analysis is the question of *decision robustness*: is there a common prefix of action sequences that results in "good" outcomes with respect to the set of all plausible future events [60] ? A related meta-strategy for strategic risk control is *re-ordering of decisions,* such that risky decisions are postponed until more is known about the events.

In terms of the general framework shown in Figure 1b, the usage of scenarios in strategic management clearly focuses on future state scenarios as an aid in defining and implementing change. Generating a future-state scenario of an organization and its environment helps bring to light the direction where one would like to go, and the necessary actions that need to be taken to get closer to that future state. Typically, a large number of scenarios is constructed reflecting different assumptions and hypotheses about the environment and the applied models and procedures. Once the scenarios have been obtained, a sorting process must be conducted to weed out those which are impractical, leaving those which are feasible. From this smaller grouping, one or a few scenarios are picked and become the basis for strategy development.

Scenarios that are considered plausible, but are not selected as most-likely-to-happen, are not discarded. Instead, they provide an element of flexibility to the chosen plan as a means of escape or fallback should it be necessary. Taken together, these fallback scenarios provide a list of indicators that management needs to monitor with respect to a possible strategy modification. The passage of time will show how closely some views depict the future, while others will prove to be quite inaccurate. The scenario chosen as the most likely view of the future is used to plan a step-wise approach to achieve the desired ends. Projects or actions required to implement the scenario are broken into manageable phases, and management makes the decision to proceed from one phase to the next over time. Of primary importance is the recognition of unexpected changes that require assessment in the context of the scenario. A by-product that scenario creation can have is the protection against errors of judgement, by flushing out mindsets or basic assumptions which, over time, are no longer valid. During the implementation of the plan, some old beliefs based on common happenings in the plan may no longer apply [8].

From a practical point of view, there are a number of *methods* that can be used *for scenario generation in the decision-theoretic framework*: trend impact analysis, cross impact analysis, intuitive economic forecasts, implicit assumptions affecting business, and the intuitive logic method. These methods are in essence very much alike, differing only in the viewpoints researchers choose for analysis [7; 71, 72]. Among them, the *intuitive logic method* seems to offer the most clearly structured procedure to define the scenarios. It involves five steps [8]:

1. *Analyze the organization decisions*: For scenarios to be useful in decision making, they must be decision-focused. That is, their analysis of alternative futures must zoom in on the specific issues that are important to the organization's strategy, concerning both present and future decisions. This ensures that the resulting scenarios are focused on those trends, events and uncertainties that are strategically relevant to the decision-making process. It defines the scope of the analysis by concentrating on key organizational decisions with long range consequences such as capital allocation, diversification, infrastructure investment and market strategies.
2. *Identify key decision factors*: Once the key decision set is defined, factors which most directly influence decision outcomes must be identified. The more is known about these factors, the better the quality of decision making. Standard management analysis tools usually suffice for identifying these factors. The factors must form the basis for the scenarios.
3. *Analyze environmental forces*: This step will shape the future business strategy. Environmental forces may be analyzed in two categories: micro level forces which most directly impact the key decision factors, and macro level forces that set the overall (global) context for the business

environment. The analysis may utilize environmental monitoring and scanning systems, business models, special information services, general literature about the future, and outside consultants. Another relevant categorization of the environmental forces distinguishes forces on which the organization has some influence (e.g., market, main competitors, new product development), from those which cannot be controlled and are exogenous to the organization (e.g., government regulations, political situation, resource availability). Often, the organization's decisions but not the macro-level forces can influence the micro-level forces.

4. *Define scenario logic*: This step establishes the basic structure of the scenarios. Scenario logic involves organizing themes, principles, hypotheses and assumptions that provide each scenario with a coherent, consistent and plausible logical underpinning. Scenario logic should encompass most of the conditions and uncertainties identified in the preceding steps. Trial and error is usually necessary in arriving at useful scenario logic. The logic does not simply consist of optimistic or pessimistic scenarios. Instead, it describes future alternatives. This step involves also the selection of models and procedures used to determine the environmental factors and their implications on the organization's status.
5. *Analyze implications for decisions and strategies*: Determining the implications of each scenario has on the decisions and strategies are a critical step for management, planning and control. Typical questions that might arise include, but are not limited to:
    - What do the scenarios imply for the design and timing of particular strategies?
    - What threats and opportunities do the scenarios suggest to the future environment?
    - What critical issues emerge from the scenarios? and
    - What kind of flexibility do the scenarios suggest are necessary from the organization's planning perspective?

Despite a long tradition in the military sector, scenario development in strategic management is still an art rather than a science. Partly, these problems in scenario development are due to the very nature of strategic management:

- *Lack of well-defined objectives*: Scenarios are often used to deal with non-routine situations that are not documented in organizational standard procedures. The unforeseen context reflects the missing of a well formulated, unified set of objectives that stakeholders need to rely on to develop scenario assumptions and possible courses of action.
- *Lack of sound assumptions*: Even if organizational objectives are clearly articulated, assumptions can be vaguely formulated, based on unreliable data, and biased. Assumptions should then be regarded as a critical component of a robust scenario. Quality tests can obviously only increase confidence in the scenario but cannot prove anything about the future. Well-known tests include coherence of model structure and parameters, reproducibility and other aspects of model behavior, including robustness with respect to various kinds of changes.
- *Lack of structure*: Management scenarios are often considered rich in content and flexible in format. However, a free-format scenario can also be seen as a weakness in that the inherent lack of structure may be source of misrepresentation and miscommunications.
- *Granularity*: Granularity expresses the level of detail described in a scenario. A scenario with low granularity, that is, with a high level of abstraction and generality, is easier to construct but may loose its practical appeal. Conversely, a scenario with high granularity may contain detailed but likely inaccurate assumptions and information.
- *Discrete scenarios versus continuous reality*: Scenarios often describe discrete events, or at best a series of major predicted events that unfold one after another. The reality is often more continuous -- facts and actions gradually evolve over time. The inability of scenarios to capture continuity might lead the organization to embark on a wrong course of action.

- *Exhaustivity of scenarios*: Managers often have incomplete recall of the past, selective recollection of their experience, and subjective view of the future. When assigned to create scenarios, the managers often end up with a set if scenarios that runs the risk of not being complete to describe all possible future events. As discussed earlier, an incomplete set of scenarios could also be the result of a non-exhaustive set of assumptions.
- *Scenarios as processes and not outcomes*: Decision-makers in organizations often look at defined scenarios as targeted outcomes. And their natural action is to direct the organization's policies and resources to accomplish the scenario. In fact, a scenario should be seen, as an instantiation of the organization's decision making, should the situation develop as described in the scenario. Uncontrollable environmental factors and controllable courses of action are only metaphors of a future reality that one seeks to capture. As such, scenario management should be used as a means to learn how to deal with uncertainty, rather than to adopt a line of action based on a set of yet-to-happen events.

Other problems in scenario construction and modification encountered in strategic management include lack of continuity and the maintenance of the organizational memory; inability to quickly alter decision factors and accurately assesses their impacts. Further, the management and evolution of scenarios pose significant problems in terms of questions such as contribution structures (*from whom do the scenarios come?*), and quality criteria (*how do we decide if a scenario is useful?*). The scope of strategic management scenarios is much larger than in HCI, so rapid feedback from prototyping is usually impossible to achieve.

Emphasizing one or more of these issues, skeptics levy little credence in the scenario-based approach since no one is able to consistently forecast the future. The view has some validity since the environment is constantly changing, and the technology base is always in flux. There are, however, ways to alleviate this problem. The analysis we are discussing expects changes. Continuous reviews and corrections are an integral part of the scenario process. As the future unfolds into the present, scenarios are reviewed and assessed to determine whether the current plan must be modified or if a new approach is needed. The key issue here is that the analysis, revision and modification of scenario can be conducted in an efficient and responsive manner.

The above concerns should therefore not discourage organizations from using scenarios for strategic planning. Examples are ample to demonstrate their importance, and success, in capturing complex future uncertainties while avoiding unfounded extrapolation. Perhaps, the most important contribution of scenarios in strategic management is their ability, as a change agent to support mentality shift required to discover alternative futures. As a link to the role of scenarios in HCI and in software systems engineering, scenario development – when used according to the management approach – should be viewed as a process that developers use to help:

- recognized unexpected changes
- protect against judgment errors by flushing out mindsets or basic assumptions that seem to be no longer valid
- use the most plausible ones as a basis for development
- monitor fallback scenarios for possible modification of development strategy.

4 Interdisciplinary Research Topics

In email discussions preceding the workshop, participants had selected four issues that plague research and practice across the disciplines, and are of sufficient importance to warrant in-depth interdisciplinary discussion. These topics were selected from a larger set of questions emerging from the practice surveys in [73], as well as from suggestions by the participants themselves. The selected topics were:

- *Systematic capture and generation of scenarios*: What should a scenario contain information about, under different contingencies? Are there systematic ways to create normal-case and exception-case scenarios? What does it mean for a set of scenarios to be complete with respect to a particular task context?
- *Representational issues of individual scenarios*: When should scenarios be represented formally, when informally? How should the two-way transition between formal and informal representations be managed? What is the appropriate level of abstraction in a scenario, given a certain purpose?
- *Fitting scenarios to existing methods*: Do scenario-based techniques replace or complement existing methods in planning, requirements engineering, and design? What are the limits of scenario-based techniques, i.e. where should they not be used? Where do these techniques fit into the established organizational processes for these tasks? What adaptations of these processes, and of proposed scenario techniques, are needed to improve the fit? How do we make knowledge about these improved methods known in work settings, how in education?
- *Scenario management in the large*: How do we manage families of scenarios, their version and configuration structures, their traceability through to test cases down to design, implementation, and use? How do we handle scenario change in such integrated settings?

Each of these topics was discussed in an interdisciplinary working group coordinated by two senior researchers from different backgrounds. The main results are summarized in the next subsections. Based on these discussions, an additional plenary brainstorming session took place in order to determine a shared view on two important pre-requisites for making scenario management an interdisciplinary but coherent research area:

- What is an appropriate definition of the term “scenario”?
- What are the most critical research problems that have not yet been addressed adequately?

4.1 Systematic Capture and Generation of Scenarios

Scenario capture and generation must be grounded in empirical fact – usage practices and attitudes towards use. However, such a statement offers only limited guidance in designing the basic work processes and interrelationships involved with capturing and generating scenarios,. Such guidance should ideally synthesize observations of informal practice (at different levels of granularity) with proposals made by proponents of more formal approaches. At the Dagstuhl workshop, this question was first attacked following traditional top-down software engineering, then expanded towards a business policy perspective. First, a strawman process of scenario-based requirements analysis was defined. In this process, the obstacles faced in each step basically determine the need for the next step. In large projects, selected *“rich-picture” scenarios* are initially created. These are typically informal, close-to-reality examples, and driven by the immediate problem causing the change process. They need to be contrasted with semi-formal *context models*, describing the most relevant components and relationships in the system environment from a rather global, architectural perspective. The context models are then elaborated into more and more *detailed models*, following a Structured or Object-oriented approach.

However, this refinement quickly leads to a *combinatorial explosion* in the number of possible system behaviors. The formal interpretation of *scenarios as traces* or threads of detailed behavior is typically employed at this level. The challenge is how to bridge the big gap between the initial goal scenarios expressed in terms of work context, and the very detailed ones typically used in formal approaches. One step towards these goals is the enhancement of conceptual modeling techniques by agent, task and goal concepts which add aspects such as responsibility [75], authority, competence, strategic dependencies, goals and obstacles to the traditional three perspectives of structure, function, and behavior.

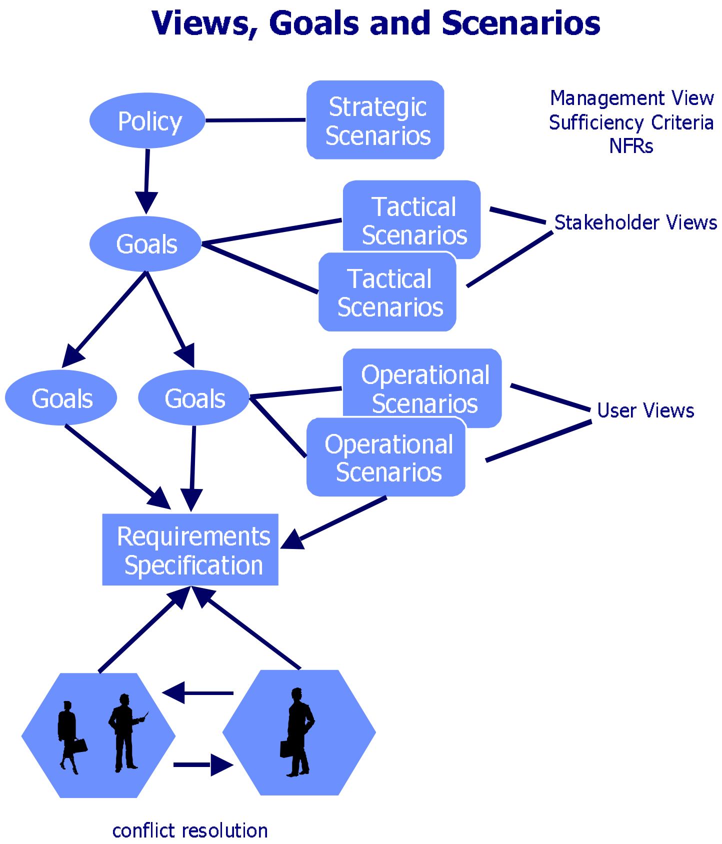

Figure 11: The problem of linking strategic, tactical, and operational scenarios to requirements

Elaborating on this theme from a business perspective, the second part of the discussion focused on the question how to make the relationships between organizational goal hierarchies and scenarios more explicit (cf. also Figure 1b). Figure 11 links policies, goals, and requirements to Anthony's [3] hierarchy of strategic, tactical, and operational levels, each expressed by their own kind of scenarios. This picture can also be understood as a potential bridge between the hitherto separate work on scenarios in strategic management and scenarios in requirements engineering.

Figure 10 also conveys another important observation. There is serious inherent conflict of interest within and between the levels. Within each level, there are multiple viewpoints, for instance, between multiple departments at the tactical levels, or between different user groups at the operational levels). In addition, there are basic conflicts between organizational goals (such as profit, rule following, policy implementation, etc.) and the work practice goals of users (flexibility and convenience, social needs, etc.) [17].

Any scenario management framework must, at the metalevel, make the different viewpoints and their relationship to scenarios explicit to set the context [51]. This teamwork-oriented traceability is orthogonal to the method-related descriptions of scenarios discussed in Figure 8, and in the section on method integration, below.

Once such basic process maps have been established, the next important research question is how to systematize their application. For the construction of the initial steps mentioned above, checklists, heuristics, and dialog-based interaction tools appear appropriate. At the more detailed levels, automatic scenario generation for coverage and, conversely inductive inference mechanisms for synthesizing details to more abstract requirements is needed, focusing on those scenarios considered most relevant in the application domain. Typical analysis techniques include case-based reuse, detection of redundant actions or events, and pathway coverage as in test case generation.

As mentioned earlier, a key problem is the combinatorial explosion as soon as scenario developers get into even a modest degree of detail. On the other extreme, situational bias, tacit knowledge and implicit assumptions may narrow the search space to less than the really important scenarios. Participants did see potential for addressing at least some aspects of this problem by formal abstractions, e.g., moving from the instance-level to the type level, decomposing the search space by specifying concurrency rather than listing all conceivable interference scenarios explicitly, and directing search by captured problem-oriented checklist to reduce bias.

4.2 Representing Scenarios: Informal vs. Formal

Most representations of scenarios in practice have been found to be “informal” in a Computer Science sense. Nevertheless, the group identified a wide variety of meanings to this term. The spectrum of scenario representations found in research and practice includes at least the following:

- Raw information: e.g., video recordings, literal transcripts
- Free format data, e.g., pictorial descriptions, free form text
- Structured representation, e.g., structured texts, templates/forms – probably the most important form
- Semi-formal syntax with some semantics, e.g., process maps in system analysis, message sequence diagrams, state charts with embedded text, pseudo code
- Formal languages with well-defined semantics, e.g., state charts, Petri nets, logic of action, etc.

The discussion proceeded from the assumption that the degree of formality required depends on the purpose of scenarios and on the intended audience; a list of drivers and inhibitors of formality is provided in Table 1. Some of the main arguments can be summarized as follows:

- Great uncertainty about requirements encourages rapid and informal scenarios. However, if these scenarios become too many, too broad or too deep, it is time for more general conceptual models validated against individual scenarios in a more formal manner. However, formalization is by no means synonym to greater coverage or more detail. Often, semi-formal representations will be sufficient to represent structural constraints within and between scenarios adequately.
- There is no clear distinction between scenarios, conceptual models, and process representations. Scenarios are almost never single-instance, but rather partially grounded fragments; in our framework scenarios are characterized as middle-level abstractions grounded in reality. However, not every partially grounded model fragment should be called a scenario. Scenarios in practice tend to have a step-to-step connotation, with no separate external frame of reference. They should be largely self-explanatory, at least in the context in which they are set.

Formality without tool support was generally considered impractical. However, it seems largely unclear what scenario tools are necessary or viable, even which ones are presently available[2].

Another important open question is whether there are different formality requirements for current-state and future-state scenarios. Considering the framework shown in Figure 1b, what is the effect of existing legacy documentation (code, design, specifications, text scenarios) on the representation of new current-state and future-state scenarios?

Drivers	**Inhibitors**
Representing the result of an agreement process	Eliciting a specific view
Consolidated view needed	Set of particular views sufficient
Focus on type/class level	Understanding without special training
Separation and linkage among components	Rapid feedback cycles between scenario author and domain experts/users
Strong need for traceability	
Strong need for unambiguity	
Assessment of given measurable properties	

Table 1: Factors promoting and inhibiting the formality of scenario representations

4.3 Fitting Scenarios to Existing Methods

Many practitioners and researchers complain that it is difficult to reconcile the systematic usage and management of scenarios with the standard methods applied in planning, analysis and design. Starting from the purpose view in Figure 8, the group looked at different phases or rough work tasks in the lifecycle, and collected arguments why scenarios should or should not be used in these tasks. The results are summarized in Table 2. They indicate that participants saw the role of scenarios predominantly in the analysis phase very early in the development process. Scenarios should also take a strong role in quality management. The potential of scenarios in subsequent design tasks was also recognized, but here, their added value is balanced to some degree by the significant additional management effort incurred.

After defining thus the rough positioning of scenarios in the systems lifecycle, another challenge resulting from the practice surveys was discussed. How can we provide more detailed method guidance in developing and using scenarios? The group adopted the view of methods as tools for the developer, which should not overly constrain the actual work process[3]. As a consequence, methods in general should be defined as collections of situated chunks that the development team could invoke and possibly adapt when desired. This, in turn, makes it relatively easy to integrate chunks of scenario method knowledge into the process. Examples of high-level "good practice" chunks from industry include:

- to structure functional and non-functional requirements along large use cases
- to assess the impact of a new commercial-off-the-shelf (COTS) software product using business scenarios

[2] A number of prototypical tools are described in this Special Issue, and in [31]. Examples include traceability support, advice for the choice or structuring of scenarios, semi-automatic synthesis of formal specifications from collections of formally represented scenarios, and animation of formal specifications by future-state scenarios.

[3] As Jim Odell put it, *"there is never just one way of doing something. Never has been, never will be, and never should be."*

- to test system compliance with requirements, and to develop user documentation, based on usage scenarios
- to optimize system performance through technical impact analysis of the most frequently occurring scenarios.

Development Task	**Pro-Scenario Arguments**	**Contra-Scenario Arguments**
Analysis	Uncover hidden requirements	Coverage problem: how many scenarios?
	Envision future system usage	Content problem: how much to capture?
	Provide rationale for design proposal	May result in overlooking concurrency
	Make requirements behavioral content more concrete	Requires much domain knowledge
	Enriched context information helps uncover risk, org. problems, etc.	
	Help envisage the potential of a problem	
Design	Illustrate trade-off between design solutions	Management of scenarios becomes complex
	Validate design using scenarios	
Quality Management	Communication aid between stakeholders	May oversimplify problems and project risks
	Facilitate documentation	Cost, time, and manpower intensive
	Verify/validate fitness for use	
	Justify needs	
	Understand and resolve conflicting quality requirements	

Table 2: Advantages and problems of using scenarios in specific development tasks

The group proposed to set up a web server infrastructure for the collection and dissemination of such “chunks of best scenario practice”, characterizing each chunk by the domain and other situational factors, the target output product, the type of scenario content, and a typical intention of reuse. Based on such a server, after a task analysis characterizing the situation, the intended reuse process could work in four steps:

1. write a current scenario in the users’ language (as a story)
2. run it against a (possibly domain-dependent) checklist to evaluate its completeness
3. validate the initial understanding gained from the thus enhanced scenario
4. identify the scope for improvement over the current scenario.

4.4 Scenario Management in the Large

As discussed earlier, individual scenarios are well accepted and easy to use. Small “chunks of best scenario practice” seem relatively obvious and well structured. In contrast, practitioners complain that maintaining large sets of possibly complex scenarios with different viewpoints over long periods of a system life cycle as a serious management nuisance. This problem is hardly addressed by research. The workgroup therefore delved into an initial exploration how one could even approach investigating the problem of proper scenario administration.

When setting up a scenario management framework, the key issue is *cost-effectiveness of scenarios in different tasks of the system lifecycle*, or even in the broader organizational change management cycle. Tackling this problem requires expanding on the purpose view of scenarios first.

Scenarios are tools for understanding (cognitive aspect) and communication (social aspect). Similar to rapid prototypes in the design stage, they reduce uncertainty about organizational, usage, or technical requirements. Cost-effectiveness should thus relate to system quality as well as quality of the change process supported by the scenario-based approach.

Major *benefits* include the following. Similar to good business process models, scenarios do not need to be complete but still help to give an overall intuitive picture. They do this by focusing on critical issues and on differences between current-state and future-state, without requiring a complete description of either the old or the new system. On the *cost* side, scenarios are so fluid that it is hard to provide structure within a large set of scenarios. It is even harder to maintain the relationships (consistency, conflict, evolution, etc.) without substantial version and configuration control effort (for which no good methodology is known to date).

To further elaborate on this point, the many roles of scenarios were summarized using the metaphors of “oil” and “glue” to point out the opportunities, but also the challenges of scenario management in the large. As a fluid and easy-to-manipulate concept *(oil),* scenarios support design decisions and help derive requirements. They assist the evaluation of requirements for incompleteness and inconsistency. They illustrate the requirements themselves as well as the potential costs of satisfying them, including unintended side effects of proposed solutions. They help define roles and responsibilities in the process. Last but not least, they serve as regression tests for old requirements in the evaluation of proposed adaptations.

Scenarios also fix required linkages (*glue*) between design artifacts and design decisions. This has an often underestimated impact on projects and thus require formal management. Scenarios determine the connection between parts of systems in a dynamic sense. As [73] show, they allow to structure and plan projects: division of labor, estimation of effort, focusing of inspections, and the validation of end-to-end functioning from the user perspective are all determined by scenarios. Project managers are known to have been fired for not satisfying important scenarios!

Summarizing, the systematic usage of scenarios has a profound but poorly understood impact on the whole change management process, ranging from requirements analysis through development and integration all the way to project planning, training and motivation of both project personnel and users. Novel approaches to version and configuration management must take into account the weak formal semantics but strong pragmatic role of scenarios, such that the relationships between scenarios and other artifacts can be maintained at all the necessary abstraction layers of the different stakeholders, and throughout the system lifecycle. Metrics of both cost and benefit are needed to determine and improve the quality of such a scenario-centered lifecycle process in a systematic manner, considering, e.g., average cost of scenario development, retrieval and usage, trade-offs between completeness and cost-effectiveness, and system quality.

4.5 Interdisciplinary Definitions and Research Questions

Following the working group summaries, participants from different disciplines were asked to summarize what they had learned across the discussions. In addition, a brainstorming session was conducted to see if participants could agree on a common definition of scenarios, and where they saw the most pressing research issues for further work.

In the brainstorming session, the group first attempted to converge towards a shared definition of scenarios. From a collection of individual properties, which were then voted concerning their importance, we finally synthesized the following definition:

> *A scenario is a description of the world, in a context and for a purpose, focusing on task interaction. It is intended as a means of communication among stakeholders, and to constrain requirements engineering from one or more viewpoints (usually not complete, not consistent, and not formal).*

In addition, critical issues in need for more research were identified. After collecting about twenty topics, and discussing them at length, it became clear that the biggest gap in our knowledge of scenarios can be summarized by the question:

> "*how to get the best value for the money invested in scenario-based techniques*."

Cost-effectiveness of scenario usage is poorly understood, thus there is no economic basis for *problem-driven use* of scenarios yet. Regardless of the detailed cost-benefit analysis, it is a shared perception among the working groups that scenario management in the large is least developed so far. Thus, questions of how scenarios evolve, how they are related to each other, and how they are linked to specifications are high-priority research issues.

In the final discussion from the management side, the hierarchy of scenarios exhibited in figure 11 was emphasized as a challenge, i.e., linking the use of scenarios for deciding strategy through to their usage in designing system interaction, and possibly back to stimulating novel uses of existing systems. From research in organizational sociology, it is not even clear whether such a linkage exist or whether unexplained emergent phenomena prevent such a bridging.

From the software engineering side, participating doctoral students pointed out that the workshop had indeed helped them to better position their work, in particular concerning the relationships of scenarios to more formal conceptual models. Such relationships include, among others, quality management of scenarios via conceptual models, scenarios as boundary objects and viewpoints constraining design, suitability of scenarios vs. models as externalized memories of development processes, and different kinds of analysis/animation/simulation relationships which allow the exploration of dynamic concepts.

From the viewpoint of HCI, dealing with scenarios (narrative, rich, non-formal descriptions) is not considered a choice but forced on research by practice. The workshop findings can, according to this viewpoint, be grouped according to four frequently asked key research questions:

- How do we deal with collections of scenarios, i.e. collections of only weakly structured text?
- How do we deal with coverage (writing an exhaustive set of scenarios)?
- What (instance/detail) in a scenario is essential, what is inconsequential?
- What are boundary conditions for applicability of scenario-based design?

Possible answers are listed in Table 3, together with some caveats when and to what degree these answers might be right or wrong.

Key research question	Typical/possible answers	Caveats
How do we deal with collections of scenarios?	For indexing/retrieval, learn from Information Retrieval	Where do classification schema, keywords come from?
	Conceptual models as indexes which offer reasoning as a side benefit	
	Minimal metadata for each scenario	Beware creeping modeling urge (over-formalization)
How do we deal with coverage?	Scenarios will only cover focal paths plus implicit set of error scenarios	But what about safety-critical systems?
	Formal descriptions (FSM family) help generate scenarios	Problem similar to test case coverage
	Scenarios abstraction, reuse facilitated by multimedia database	Beware formalism for tool's sake!
What detail is necessary?	Shared background tells us if we keep users around	But might get lost when users/ domain experts are no longer participants
	Conceptual modeling may help to ask the right questions	
What are boundary conditions for scenario applicability?	Scenarios focus on action/event stories	(some) non-functional requirements
	Scenarios support linearization	Highly parallel, non-transactional applications
	Scenarios broaden thinking in action	Parameter-fitting/optimization (standard engineering design)

Table 3: FAQ's for scenario-based design

5 Networking Research Hypotheses and Evaluation Methods

The preceding sections demonstrate a certain convergence concerning frameworks and theories for understanding scenarios, and shared concerns of interdisciplinary research and practice. However, the variety of underlying assumptions, specific research claims and methods remains substantial. In order to develop a coherent research community, it would be valuable to understand in the interdisciplinary setting what are the relationships between these *claims and hypotheses*, and what are the relationships between the *evaluation methods* used in the various disciplines to evaluate these claims and hypotheses.

In order to explore these questions, we first distributed a short questionnaire to all workshop participants with the following text:

> *"Each of us has proposed some theory, model, structure, language, ontology, formalism ... with some purpose in mind. We want to know (with a maximum of 25 words):*
> 1. *In your work, what is your most important claim/hypothesis/conjecture about scenarios?*
> 2. *How do you propose to justify/test/prove it? (Please mention if you are using some kind of tool/environment as a demonstration or test bed.)."*

The 24 responses to these questions were then synthesized into strawman “causal networks”, one intended to show the hypothesized relationships between scenario factors across respondents (figure 12), the other intended to show possible paths of research projects/evaluation methodologies tend to follow (figure 13).

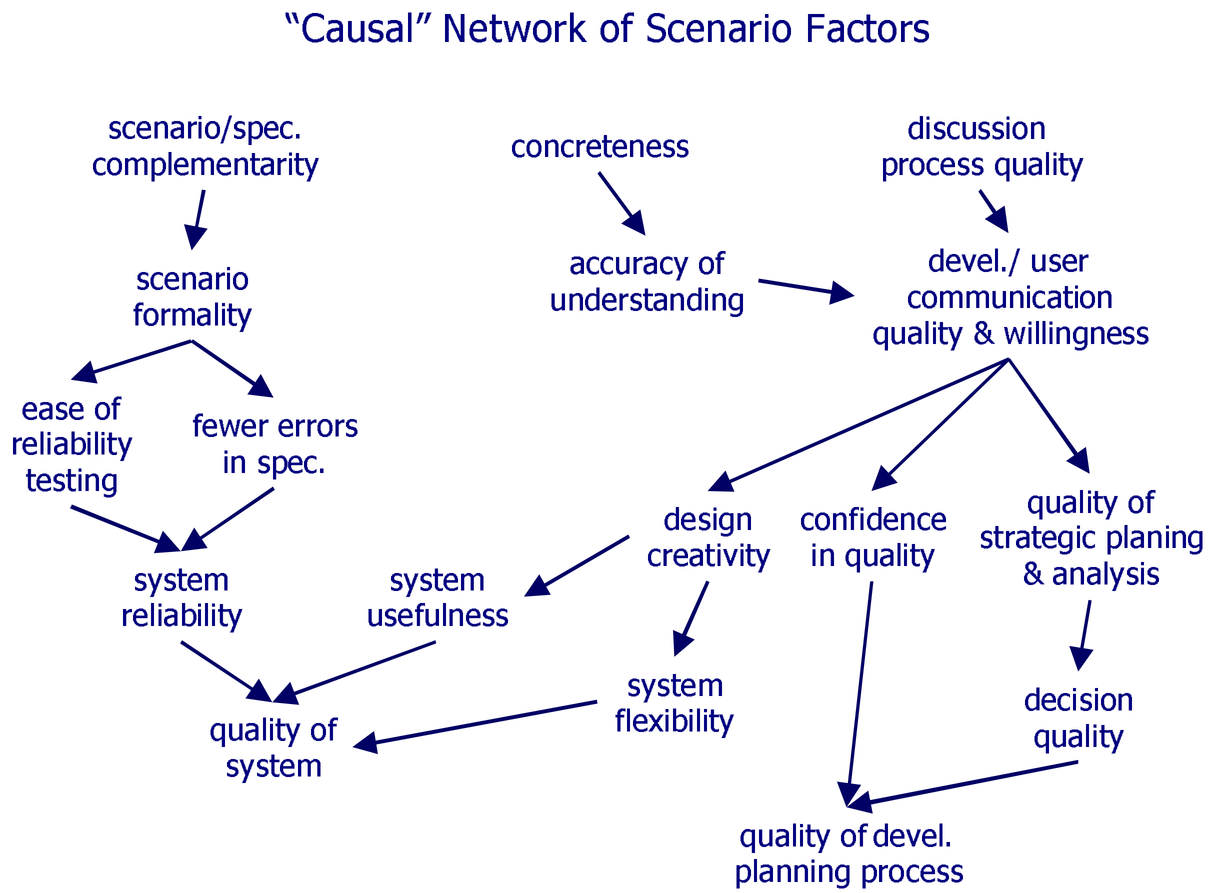

Figure 12: A causal network of scenario research constructs

Briefly, the research claim network in Figure 12 concentrates on two key factors which many researchers address in different ways: scenario formality and communication quality. The left part suggests that the degree of *scenario formality* depends on the complementarity between scenarios and specifications in a particular context. Scenario formality will facilitate reliability testing, and may reduce the errors in specifications, both positively influencing *system reliability*. But only together with knowledge about system usefulness and flexibility (which is not necessarily improved by formality of scenarios), the overall quality of systems can be assessed.

These two latter factors are influenced by *process-oriented scenario factors* rather than representation, most prominently the willingness and quality of *developer/user communication*. In turn this is conjectured to promote factors such as design creativity, subjective confidence in quality, and (strategic) decision quality. Communication quality is influenced by accuracy of understanding promoted by scenario concreteness, and by process factors of how the discussion process proceeds. Hidden in this network, we recognize the three well-known dimensions of requirements engineering [Pohl 1994], namely representational aspects (left part of the figure), depth of understanding (middle part), and quality of teamwork and agreement (right part).

In terms of *evaluation methods*, two major approaches are shown in Figure 13. The first one involves *tool building* as a starting point for demonstrating and testing claims. These are then exposed to expert critiques or lab experiments, prior to their use in *industrial case studies* which either construct industrial prototypes for further development into the commercial arena, facilitate/monitor ongoing specific projects, or try a rational reconstruction of a past process.

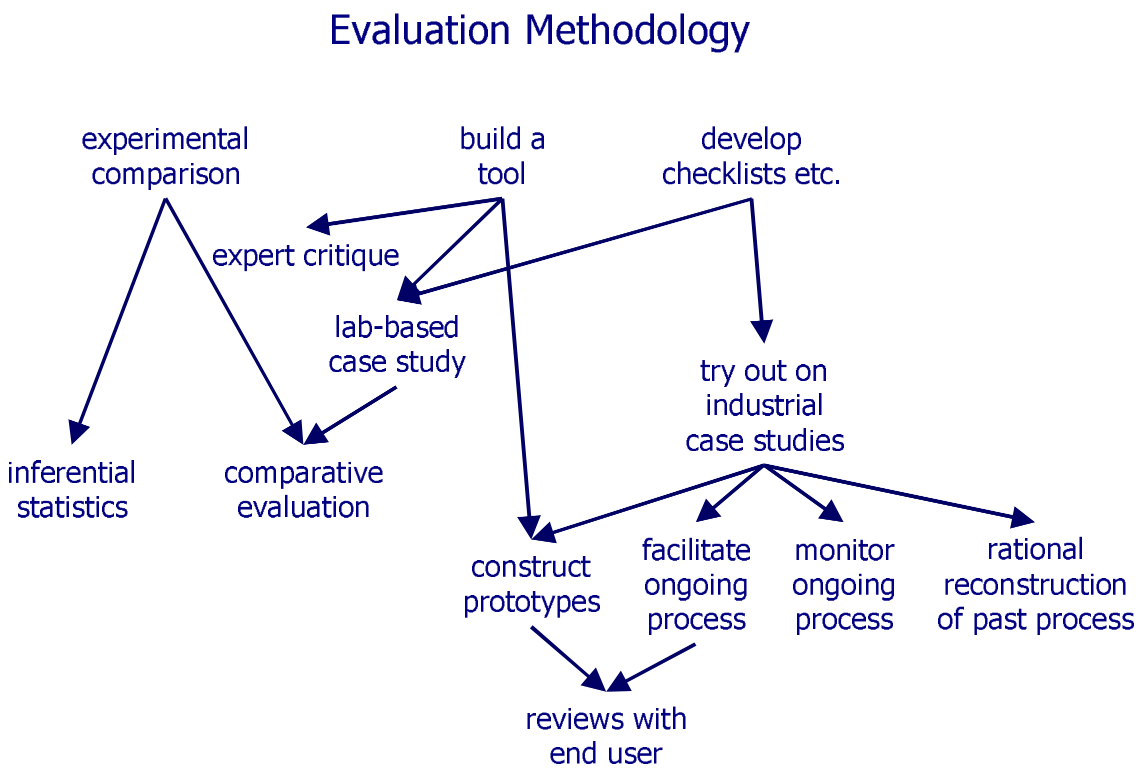

Figure 13: Possible evaluation paths in scenario research

The second evaluation approach investigates scenario-based methods independently of support tools. Often, the research claim, or a deeper theory underlying it, is elaborated into *checklists* which can directly be applied to laboratory experiments or industrial case studies, without necessarily going through a mediating tool.

In both approaches, valuable insights can be drawn from *comparative evaluation* with competing claims, tools, or checklists. The design of comparative studies in the scenario field is, however, particularly difficult due to the complexity of problems addressed by scenario-based approaches, and there have been few such studies to date. More often than they should, evaluation has therefore been restricted to the conceptual level.

Taken together, these two networks provide prolegomena towards an interdisciplinary research program in scenario management. We hope that they will help researchers to better relate their findings, and to build on each others' methodologies within and across disciplines.

7 Summary

In this paper, we reviewed scenario management from three major disciplines: strategic management, human-computer interaction, and software and systems engineering, and propose an interdisciplinary framework for scenario management. In addition to synthesizing our own previous research, we drew on findings of an interdisciplinary workshop including various brainstorming techniques.

Across all disciplines, scenarios are recognized as indispensable tools to comprehend future states of the world. We have outlined conditions under which scenarios can be best used for each of the three disciplines. As scenarios take different forms to fit in a particular application context, they invariably provide a coherent framework for analysis of how various elements of a problem at hand (e.g., defining systems specifications) impinge on one another and interact. Furthermore, they also serve as a vehicle to foster creativity, stimulate discussion, and help focus attention on specific points of interest.

With some diversity in terminology and use, two particular qualities of scenarios emerge from this study. First, a scenario is a context-dependent and purposeful description of the world with a focus on task interaction. Second, scenarios are a means of communication among stakeholders.

We look at scenarios as enablers of change. From that perspective, four research issues were discussed: systematic capture and generation of scenarios, representational issues of individual scenarios, fitting scenarios to existing methods, and scenario management in the large. Our study has helped identify (1) factors promoting and inhibiting the creation of scenarios, (2) advantages and problems of using scenarios in specific development tasks, (3) practical issues in developing scenarios, and (4) research constructs for designing and evaluating scenario and scenario management.

In all of the issues addressed in this paper, it should be remembered that scenario development, analysis and management are mainly practical processes that depend on creative participation and inputs from individuals, and no attempt is made here to propose a rigid methodology. The value of scenarios is that they serve as catalysts for such processes. We hope that the interdisciplinary discussion presented in this paper can facilitate the use of scenario approaches and make scenario studies and scenario use as interesting and effective as possible.

Acknowledgments. The Dagstuhl Workshop on Scenario Management, February 9-13, 1998, was supported in part by the Dagstuhl Foundation, and by the European Commission via ESPRIT Long Term Research project 21.903 (CREWS) and via the RENOIR Network of Excellence in Requirements Engineering. Participants included C. Ben Achour, D. Berry, S. Brandt, X.T. Bui, J.M. Carroll, G. Chin, D.R. Corrall, E. Dubois, W. Dzida, M. Feblowitz, M. Francksson, M. Glinz, S. Greenspan, P. Haumer, P. Heymans, P. Hsia, M. Jarke, S. Jungmayr, H. Kaindl, J. Koenemann, R. Kurki-Suonio, A. van Lamsweerde, J. Leite, K. Lyytinen, N. Maiden, S. Minocha, A. Oberweis, B. Paech, K. Pohl, J.-C. Pomerol, C. Potts, B. Regnell, C. Rolland, M.B. Rosson, K. Ryan, R. Sprague, A.S. Sutcliffe, and V. Thurner. The help of the group leaders and discussion recorders in the various subgroups is greatly appreciated.

References

1. Ackoff, R.L. 1979. Resurrecting the future of operations research. *Journal of the Operations Research Society 30*(3), 189-199.
2. Anton, A.I., McCracken, W.M., Potts, C. 1994. Goal decomposition and scenario analysis in business process re-engineering. *Proc. 6th Intl. Conf. Advanced Information Systems Engineering (CaiSE 94),* Utrecht, NL, Springer-LNCS, 94-104.
3. Anthony, R. 1985. *Planning and Control Systems: A Framework for Analysis*. Division of Research, Graduate School of Business, Harvard University, Boston, Mass.
4. Becker, H.A. 1983. The role of gaming and simulation in scenario project, in Stahl, ed., *Operational Gaming: An International Approach*, International Institute for Applied Systems Analysis, Laxenburg, Austria, pp. 187-202.
5. Blanning, R.W. 1995. A decision support framework for scenario management. *Proc. Intl. Symp. Decision Support Systems*, Hong Kong, vol. 2, 657-660.
6. Brooks, F. 1995. *The Mythical Man-Month: Essays on Software Engineering.* Reading, MA: Addison-Wesley, Anniversary Edition (originally 1975).
7. Brown, S. 1968. Scenarios in system analysis, in E.S. Quade and W.E. Boucher, eds., *Systems Analysis and Policy Planning: Applications in Defense*, Elsevier, New York, NY, 298-309.
8. Bui, X.T., Kersten, G., Ma, P.-C. 1996. Supporting negotiation with scenario management. *Proc. 29th Hawaii Intl. Conf. System Sciences*, Wailea, HI., vol. III, 209-219.
9. Carroll, J.M., Ed. 1995. *Scenario-based Design: Envisioning Work and Technology in System Development.* New York: John Wiley and Sons.
10. Carroll, J.M. 1999. Five reasons for scenario-based design. *Proc. 32nd Hawaii International Conference on Systems Sciences*, Wailea, HI.
11. Carroll, J.M. & Rosson, M.B. 1990. Human-computer interaction scenarios as a design representation. *Proc. 23rd Hawaii Intl. Conf., on System Sciences, vol. II,,* Kona, HI, 555-561.
12. Carroll, J.M. & Rosson, M.B. 1992. Getting around the task-artifact cycle: How to make claims and design by scenario. *ACM Transactions on Information Systems, 10,* 181-212.
13. Checkland, P.B. 1981. *Systems Thinking, Systems Practice*. New York: Wiley.
14. Churchman, W. 1970. Operations research as a profession. *Management Science, 17(2),* 37-53.
15. Chin, G., Rosson, M.B. & Carroll, J.M. 1997. Participatory analysis: Shared development of requirements from scenarios. In S. Pemberton (Ed.), *Proceedings of CHI'97: Human Factors in Computing Systems.* (Atlanta, 22-27 March). New York: ACM Press/Addison-Wesley, pp. 162-169.
16. Dardenne, A., van Lamsweerde, A., Fickas, S. 1993. Goal-directed requirements acquisition. *Science of Computer Programming 20*, 1, 3-50.
17. deMichelis, G., Dubois, E., Jarke, M., Matthes, F., Mylopoulos, J., Papazoglou, M., Schmidt, J.W., Woo, C., Yu, E. 1998. Cooperative information systems: a manifesto. In *Cooperative Information Systems: Trends and Directions* (Papazoglou/ Schlageter, eds.), Academic Press, 315-363.
18. Duval, S. & Wicklund, R.A. 1972. *A Theory of Objective Self-Awareness*. New York: Academic Press.
19. Erikson, E.H. 1980. *Identity and the Life Cycle*. New York: Norton.
20. Festinger, L. 1957. *A Theory of Cognitive Dissonance.* New York: Harper & Row.
21. Festinger, L., Riecken, H.W. & Schachter, S. 1956. *When Prophecy Fails*. Minneapolis,: University of Minnesota Press.
22. Filippidou, D. 1998. Designing with scenarios: a critical review of current research and practice. *Requirements Engineering 3*, 1, 1-22.
23. Freud, S. 1900. *The Interpretation of Dreams*. Standard Edition, Vol. IV. London: Hogarth.
24. Gagne, R.M., & Briggs, L.J. 1979. *Principles of Instructional Design.* New York: Holt, Rinehart and Winston.
25. Haumer, P., Pohl, K., Weidenhaupt, K. 1998. Requirements elicitation and validation with real-world scenes. *IEEE Transactions on Software Engineering*, Special Issue on Scenario Management, December 1998.
26. Haviland, S.E. & Clark, H.H. 1974. What's new? Acquiring new information as a process in comprehension. *Journal of Verbal Learning and Verbal Behavior, 13*, 512-521.
27. Heymans, P., Dubois, E., 1998. Scenario-based techniques for supporting the elaboration and the validation of formal requirements. *Special Issue on Interdisciplinary Uses of Scenarios, Requirements Engineering Journal 3*(3), this volume.
28. Hsia, P., Samuel, J., Gao, J., Kung, D., Toyoshima, Y., Chen, C. Formal approach to scenario analysis. *IEEE Software*, March 1994, 33-41.
29. Huber, G.P. McDaniel, R.R. 1986. Exploiting information technologies to design more effective organizations. In Jarke, M. (ed.): *Managers, Micros, and Mainframes: Integrating Systems for End Users*, Wiley Series on Information Systems, John Wiley, 221-236.

30. Jacobsen, I. 1995. The use-case construct in object-oriented software engineering. In J.M. Carroll (Ed.), *Scenario-based design: Envisioning Work and Technology in System Development.* New York: John Wiley & Sons, pp. 309-336.
31. Jarke, M., Kurki-Suonio, R. (eds.) 1998. Special Issue on Scenario Management. *IEEE Transactions on Software Engineering*, December 1998.
32. Jarke, M., Mylopoulos, J., Schmidt, J.W., Vassiliou, Y. 1992. DAIDA – an environment for evolving information systems. *ACM Trans. Information Systems 10*, 1, 1-50.
33. Jarke, M., Pohl, K. 1994. Requirements engineering in 2001: (virtually) managing a changing world. *IEE Software Engineering Journal*, *6*, 5.
34. Karat, J. & Bennett, J.B. 1991. Using scenarios in design meetings – A case study example. In J. Karat (Ed.), *Taking Design Seriously: Practical Techniques for Human-Computer Interaction Design.* Boston: Academic Press, pages 63-94.
35. Kahneman, D., Tversky, A. 1972. Subjective probability: A judgement of representativeness. *Cognitive Psychology, 3*, 430-454.
36. Kahneman, D, Lovallo, D. 1993. Timid choices and bold forecasts: a cognitive perspective on risk-taking. *Management Science 39*, 17-31.
37. Klein, G.A., Orasanu, J., Calderwood, R., Zsambok, C.E. (eds.) 1993. *Decision Making in Action : Models and Methods*. Ablex Publ.
38. Kurki-Suonio, R. 1996. Fundamentals of object-oriented specification and modeling of collective behaviors. In *Object-Oriented Behavioral Specifications* (Eds. H. Kilov, W. Harvey), Kluwer Academic Publishers 1996, 101-120.
39. Kyng, M. 1995. Creating contexts for design. In J.M. Carroll (Ed.), *Scenario-based design: Envisioning work and technology in system development.* New York: John Wiley & Sons, pp. 85-107.
40. Lamport, L. (1994)The temporal logic of actions. *ACM Trans. Programming Languages and Systems 16*(3), 872-923.
41. v. Lamsweerde, A., Willemet, L. 1998. Inferring declarative requirements specifications from operational scenarios. *IEEE Transactions on Software Engineering*, Special Issue on Scenario Management, to appear.
42. Lévi-Strauss, C. 1967. *Structural Anthropology*. Garden City, NY: Anchor Books.
43. March, J.G., Shapira, Z. 1987. Managerial perspectives on risk and risk taking. *Management Science 33*, 1404-1418.
44. McLuhan, M. 1994. *Understanding Media: The Extensions of Man.* Cambridge, MA: MIT Press (original edition, 1964).
45. McMenamin, S.M.,Palmer, J.F. 1984. *Essential Systems Analysis*. Prentice Hall.
46. Medin, D.L. & Schaffer, M.M. 1978. A context theory of classification learning. *Psychological Review, 85*, 207-238.
47. Mills, C. W. 1959. *The Sociological Imagination.* New York: Oxford University Press.
48. Muller, M.J., Tudor, L.G., Wildman, D.M., White, E.A., Root, R.A., Dayton, T., Carr, R., Diekmann, B., & Dystra-Erickson, E. 1995. Bifocal tools for scenarios and representations in participatory activities with users. In J.M. Carroll (Ed.), *Scenario-based Design: Envisioning Work and Technology in System Development.* New York: John Wiley, pp. 135-163.
49. Mylopoulos, J., Chung, L., Nixon, B. 1992. Representing and using non-functional requirements: a process-oriented approach. *IEEE Trans. Software Eng.18*, 6, 483-497.
50. Nisbett, R.E. & Wilson, T.D. 1977. Telling more than we can know: Verbal reports on mental processes. *Psychological Review, 84*, 231-259.
51. Nissen, H.W., Jarke, M., 1999. Repository support for informal teamwork methods. In Lyytinen/Welke (eds.): *Special Issue on Meta Modeling and Method Engineering, Information Systems 24*, 2.
52. Orr, J.E. 1986. Narratives at work. *Proceedings of CSCW'86: Conference on Computer-Supported Cooperative Work.* (Austin, TX, December 3-5, 1986). pages 62-72.
53. Pohl, K. 1994. The three dimensions of requirements engineering: a framework and its applications. *Information Systems 19*, 3, 243-258.
54. Pomerol, J.-C. 1997. Artificial intelligence and human decision making. *European Journal of Operations Research 99*, 3-25.
55. Propp, V. 1958. *Morphology of the Folktale*. The Hague: Mouton (originally published in 1928).
56. Raiffa, H. 1968. *Decision Analysis*. McGraw Hill.
57. Rolland, C., Ben Achour, C. 1997. Guiding the construction of textual use case specifications. *Data & Knowledge Engineering 25*, 1/2, 125-160.
58. Rolland, C., Ben Achour, C., Cauvet, C., Ralyte, J., Sutcliffe, A., Maiden, M., Jarke, M., Haumer, P., Pohl, K., Dubois, E., Heymans, P. 1998. A proposal for a scenario classification framework. *Requirements Engineering Journal 3*, 1, 23-47.
59. Rosch,E., Mervis, C.B., Gray, W., Johnson, D., Boyes-Braem, P. 1976. Basic objects in natural categories. *Cognitive Psychology, 7,* 573-605.
60. Roy, B. 1997. Un chainon manquant en RO-AD, les conclusions robustes, Cahier LAMSADE 114, Universite Paris-Dauphine.
61. Schank, R.C. 199?. *Tell me a Story***:** *Narrative in the Context of Intelligence.* Evanston, IL: Northwestern University Press.
62. Schön, D.A. 1967. *Technology and change: The New Heraclitus.* New York: Pergamon Press.

63. Schön, D.A. 1983. *The Reflective Practitioner: How Professionals Think in Action.* New York: Basic Books.

64. Scriven, M. 1967. The methodology of evaluation. In R. Tyler, R. Gagne, & M. Scriven (Eds.), *Perspectives of Curriculum Evaluation.* Chicago: Rand McNally, pp. 39-83.

65. Schriver, K. 1997. *Dynamics in Document Design*. New York: John Wiley and Sons.

66. Sutcliffe, A. 1998. Scenario-based requirements analysis. *Requirements Engineering Journal 3*, 1, 48-65.

67. Sutcliffe, A., Maiden, N.A.M., Minocha, S., Manuel, D., 1998. Supporting scenario-based requirements engineering. *IEEE Transactions on Software Engineering*, Special Issue on Scenario Management, December 1998.

68. Tsatsoulis, C., Cheng, Q., Wie, H.-Y. 1997. Integrating case-based reasoning and decision theory. *IEEE Expert 12*, July, 46-55.

69. Tversky, A., Kahneman, D. 1974. Judgements under uncertainty: Heuristics and biases. *Science, 185*, 1124-1131.

70. Tversky, A., Wakker, P. 1995. Risk attitudes and decision weights. *Econometrica 63*, 1255-1280.

71. Wack, P., 1985a Scenarios: unchartered waters ahead, *Harvard Business Review*, September/October.

72. Wack, P., 1985b Scenarios: shooting the rapids, *Harvard Business Review*, November/December.

73. Weidenhaupt, K., Pohl, K., Jarke, M., Haumer, P. 1998. Scenario usage in software development: current practice. *IEEE Software*, March 1998, 34-45.

74. Wertheimer, M. 1938. Laws of organization in perceptual forms. In W.D. Ellis (Ed.), *A Sourcebook of Gestalt Psychology*. London: Paul, Trench, Trubner.

75. Wirfs-Brock, R. 1995. Designing objects and their interactions: A brief look at responsibility-driven design. In J.M. Carroll (Ed.), *Scenario-Based Design: Envisioning Work and Technology in System Development.* New York: John Wiley, 337-360.