Defence Research and Development Canada
Recherche et développement pour la défense Canada

DEFENCE R&D DÉFENSE

Critiquing systems for decision support

H. Irandoust
DRDC Valcartier

Defence R&D Canada – Valcartier
Technical Report
DRDC Valcartier TR 2003-321
February 2006

Canada

Critiquing systems for decision support

H. Irandoust
DRDC Valcartier

Defence R & D Canada - Valcartier

Technical Report

DRDC Valcartier TR 2003-321

February 2006

Author

Hengameh Irandoust

Approved by

Éloi Bossé
Section Head, Decision Support Systems

Approved for release by

Gilles Bérubé
Chief Scientist

[Work unit 13dm22]

Abstract

The purpose of this document is to provide a general introduction to critiquing systems and to discuss their particularities as decision support tools. Critiquing systems (critics) are software programs that take the problem description and the user's partial or final result as inputs and provide a critique of the user-proposed solution as output. This specific type of human-computer interaction, and the fact that critics can operate with limited knowledge, makes these systems very appropriate for open-ended domains where it is impractical to embed complete knowledge or where users are unwilling to cede decision-making.

This report provides definitions of critiquing and related concepts; describes the general mechanisms of critiquing systems; examines the adequacy of the critiquing approach with regard to the nature of user tasks and application domains; presents the different intervention strategies used by software critics as well as their user perception; discusses the user-adaptation of critiques and the joint use of critiques and explanation/argumentation; and finally identifies the distinctive features of critiquing systems in terms of cognitive support, which comprise transfer of decision control to the user, collaborative problem-solving, and detection and correction of errors and biases.

Résumé

L'objectif de ce document est d'offrir une introduction générale aux systèmes de critique et de mettre en évidence leurs particularités en tant qu'outils d'aide à la décision. Les systèmes de critique (critiques) sont des logiciels qui prennent comme entrée la description du problème ainsi que le résultat partiel ou final de l'utilisateur et qui produisent en sortie une critique de la solution proposée par l'utilisateur. Cette configuration particulière au point de vue interaction personne-machine et le fait que les critiques puissent fonctionner avec une connaissance incomplète du domaine rendent ces systèmes très appropriés à des domaines à problématiques ouvertes où on ne peut spécifier les connaissances de manière exhaustive ou à des contextes dans lesquels l'utilisateur ne souhaite pas céder son pouvoir décisionnel.

Ce rapport offre des définitions de la critique et des concepts reliés; décrit les mécanismes généraux des systèmes de critique; examine l'applicabilité de cette approche par rapport à la nature de la tâche utilisateur et au domaine d'application; présente les différentes stratégies d'intervention utilisées par les systèmes de critique et leurs effets sur la perception utilisateur; aborde la question de l'adaptation des critiques au profil utilisateur ainsi que l'usage conjoint des critiques et de l'explication / argumentation; et enfin, identifie les traits distinctifs des systèmes de critique en tant que support cognitif, ce qui comprend : le transfert du contrôle décisionnel à l'utilisateur, la résolution coopérative de problèmes et la détection et la correction des erreurs d'exécution et de jugement.

This page intentionally left blank.

Executive summary

The purpose of this document is to provide a general introduction to critiquing systems and to discuss their particularities as decision support tools. Critiquing systems (critics) are software programs that take the problem description and the user's partial or final result as inputs and provide a critique of the user-proposed solution as output. This specific type of human-computer interaction, and the fact that critics can operate with limited knowledge, makes these systems very appropriate for open-ended domains where it is impractical to embed complete knowledge or where users are unwilling to cede decision-making.

In this report, we first provide definitions of critiquing and related concepts. We then sketch the global architecture of critiquing systems, present their general features and describe the two types of evaluation procedures by which the system can critique a user-proposed solution.

Next, we examine the adequacy of critiquing systems relative to the nature of the user task and the application domain. We clarify the aspects of the knowledge employed by the user that are tested by the critic; we explain how critics can operate in ill-defined domains; and finally, we try to identify the tasks for which the critiquing approach can be used.

We then discuss aspects relative to the critic's feedback: the intervention strategies used by critics and their effects in terms of user perception; the adaptation of critiques to the profile and needs of the user; and the use of explanation and argumentation for improving user acceptance of criticism.

In the last section, we try to identify the defining features of critiquing systems in terms of cognitive support. This highlights those characteristics of critics that distinguish them from other types of advisory systems: critics transfer decision control to the user, support collaborative problem-solving, and make users notice their errors and biases.

We conclude the report with a brief discussion.

Irandoust, H. 2006. Critiquing systems for decision support. TR 2003-321 DRDC Valcartier.

Sommaire

L'objectif de ce document est d'offrir une introduction générale aux systèmes de critique et de mettre en évidence leurs particularités en tant qu'outils d'aide à la décision. Les systèmes de critique (critiques) sont des logiciels qui prennent comme entrée la description du problème ainsi que le résultat partiel ou final de l'utilisateur et produisent en sortie une critique de la solution proposée par l'utilisateur. Cette configuration particulière au point de vue interaction personne-machine et le fait que les critiques puissent fonctionner avec une connaissance incomplète du domaine rendent ces systèmes très appropriés à des domaines à problématiques ouvertes où on ne peut spécifier les connaissances de manière exhaustive ou à des contextes dans lesquels l'utilisateur ne souhaite pas céder son pouvoir décisionnel.

Dans ce rapport, nous offrons une définition de la critique et des concepts reliés, nous esquissons l'architecture globale des systèmes de critiques, présentons leurs caractéristiques générales et décrivons les deux types de procédures d'évaluation par lesquelles le système peut critiquer la solution proposée par l'utilisateur.

Nous discutons ensuite de l'applicabilité des critiques relativement à la tâche utilisateur et au domaine d'application. Nous clarifions d'abord quels sont les aspects de la connaissance mise en œuvre par l'utilisateur qui sont examinés par le critique; nous expliquons ensuite comment le critique peut être opérationnel même avec une connaissance incomplète du domaine; et enfin, nous essayons d'identifier les tâches qui se prêtent le mieux à l'approche critique.

Nous discutons ensuite des aspects relatifs à la rétroaction du système : les stratégies d'intervention utilisées par les critiques et leurs effets au point de vue perception utilisateur; l'adaptation de la critique au profil et aux besoins de l'utilisateur; et l'intégration de l'explication et de l'argumentation dans la critique afin d'améliorer l'acceptation de la critique de la part de l'utilisateur.

Dans la dernière partie, nous identifions les propriétés définitoires des systèmes de critique en tant que support cognitif. Cela fera ressortir les caractéristiques qui distinguent les critiques des autres systèmes de conseil : les critiques transfèrent le contrôle décisionnel à l'utilisateur, aident à la résolution coopérative de problèmes et signalent aux utilisateurs leurs erreurs d'exécution et de jugement.

Nous concluons le rapport par une brève discussion.

Irandoust, H. 2006. Critiquing systems for decision support. TR 2003-321 DRDC Valcartier.

Table of contents

Abstract i

Résumé i

Executive summary iii

Sommaire iv

Table of contents v

List of figures vii

List of tables vii

1. Introduction 1

2. Critiquing: attempting a definition 3

3. Architecture of a critiquing system 5

4. Evaluation procedures 9

5. Scope 12
   - 5.1 Knowledge dimensions 12
   - 5.2 Operating with incomplete knowledge 14
   - 5.3 Task properties 18

6. Critic's feedback 20
   - 6.1 Intervention strategies 20
   - 6.2 User-tailoring of critiques 22
   - 6.3 Use of explanation and argumentation 24

7. Critiquing and cognitive support 28
   - 7.1 Decision control 28
   - 7.2 Cooperative problem solving 31
     - 7.2.1 Reflection-in-Action 32

7.2.1.1 Context shifts 34
7.2.1.2 Ergonomics 34
7.2.2 Visibility 36
7.3 Detecting and correcting errors and biases 37
7.3.1 A model of human error 38
7.3.2 Use of alternative strategies 40
8. Conclusion 44
9. References 46
Distribution list 50

List of figures

Figure 1. Expert system versus critiquing system 6

Figure 2. Architecture of a critiquing system, from [1] 7

Figure 3. Testing the credibility of knowledge, from [1] 13

Figure 4. Critiquing and argumentation, from [23] 26

Figure 5. Role of critiquing in design, from [16] 33

Figure 6. Human expert's knowledge and goal of critiquing, from [12] 39

List of tables

Table 1. Comparative approach versus analytical approach 10

Table 2. Differences between expert and critiquing systems 15

This page intentionally left blank.

1. Introduction

There are many domains in which problems cannot be completely specified in advance and optimal solutions cannot be found algorithmically. Expert systems are inappropriate for these contexts because they need a clear problem specification to which they offer a complete solution. Moreover, users will have to decide upon (accept or reject) a solution that they have taken no part in elaborating and that they might not even fully understand.

In principle, critiquing systems (critics) can provide an answer to both of these issues: the brittle case problem and the decision burden problem. Critiquing systems are partial problem solvers that cooperate with human users to enhance their solution. It is this different dynamic of critiquing systems and its effects on different decision-making issues that we investigate in this document.

More specifically, the purpose of this report is to provide a general introduction to critiquing systems and to discuss their particularities as decision support tools. This will prepare the ground for a more detailed investigation of the use of critiquing systems in the command and control context, as planned within the framework of the related Technology Investment Fund Project, entitled *CoA Critiquing System for the Improvement of the Military Estimate Process*.

The document is organized as follows.

First, in Section 2 we attempt a new definition of critiquing and provide definitions of related concepts. Next, we outline the global architecture of critiquing systems and present their general features (Section 3). In Section 4 we describe the two types of evaluation procedures, analytical and differential, that critics use to provide feedback on the user-proposed solution.

We then discuss the adequacy of critiquing systems relative to the nature of the user's task and the application domain (Section 5). First, we determine the different aspects of the knowledge employed by the user that are tested by the critic. Then, we explain how critics can operate in ill-defined domains, and finally we try to identify tasks for which the critiquing approach can be used.

In Section 6, we discuss issues relative to the feedback provided by critiquing systems: the different intervention strategies used by critics and their effects in terms of user perception; the ways in which the critic's feedback can be tailored to the profile and needs of the user, and how user acceptance of critiques can be enhanced by means of explanation and argumentation.

In the last section, we try to identify the defining features of critiquing systems in terms of cognitive support. This will underscore those characteristics of critiquing systems that distinguish them from other types of advisory systems: critics transfer

decision control to the user, support collaborative problem-solving, and help the user notice their errors and judgement biases.

We conclude the report with a brief discussion.

2. Critiquing: Attempting a Definition

Our research in the field of argument studies yielded no definition of critiquing as a distinct type of discourse having its own purpose, structure and dynamics. Using the theoretical framework of argumentation, we attempt a definition based on the study of a corpus of critiques in different domains.

Our position is that a critique is the assessment of a claim, product, fact or situation in order to provide a judgement. This assessment requires a thorough analysis of the product, which is probably the characteristic that distinguishes a critique from a simple evaluation.

Moreover, a critique has a pragmatic orientation: a critique is explicitly or implicitly addressed to the individual(s) or entities that are behind the outcome being assessed. This accounts for the fact that critiquing is often associated with the dialogue schema. Even self-criticism, which can be seen as the judgement of an individual by himself or herself, evokes a dialogue where the same person embodies two distinct participants. Thus, critiquing is part of a dialectical process, which is not true for simple evaluations.

In the critiquing systems paradigm, the dialogue structure is often emphasized, probably because interactivity is a desirable feature for intelligent systems. But there is also another motivation that is specific to the critiquing approach: designers do not want the system's critiques to be viewed as a sermon or a challenge to the user's authority. As Silverman [1] puts it, the critiquing process is "a two-way communication', a 'mutual exchange of viewpoints between the expert human and the expert critic".

In certain contexts, the dialogue takes place between two diverging, if not conflicting, opinions. Thus, a *critical discussion* is a discussion in which a proponent must defend a claim against challenges of various kinds by an opponent or critic [2, 3]. This context might exclude positive critiques, which reinforce the opinion or the fact they are directed at.

Another related concept is that of *critical thinking*, which is defined as "the ability to analyze, criticize, and advocate ideas; to reason inductively and deductively; and to reach factual or judgemental conclusions based on sound inferences drawn from unambiguous statements of knowledge or belief." [4]. Competency in critical thinking helps individuals to distinguish fact from judgement or belief from knowledge. As we will see, one kind of decision support that critiquing systems offer, although no explicit reference to critical thinking is made, is to point out users' judgement errors.

In the critiquing systems literature, definitions are generally expressed in terms of the expected or observed effects of critiquing rather than its inherent characteristics. For example, it is claimed that critiquing has positive outcomes such as deeper reflection, better situation analysis, learning, etc. For Fischer et al. [5], critiquing produces many

benefits, including the growth of knowledge, error elimination, and the promotion of mutual understanding by all participants. Silverman [1] claims that critics help the user to acquire knowledge through the iterative interactions of criticism dialogues.

In fact, to a certain extent, such assumptions have proven to be valid in the human-system environment. Above all, in this context, contrary to the human-human context, criticism is not negatively perceived. As [6] point out, critiques coming from an intelligent computer assistant might be more tolerable than critiquing from a human because they are handled as a private matter between the user and the computer. Nevertheless, critiquing can raise some personal and social issues. As these authors note, like recommendations from colleagues, critics can be seen as helpful or hindering, as supportive or interfering with the accomplishment of goals. Therefore, critiquing systems should be integrated into the work environment in such a way that users welcome their involvement.

One of the greatest advantages of critiquing, compared with other discourse exchanges triggered by disagreement, is that a critique is always centred on the proponent's claim. In the human-computer context, the critiquing system does not explain or argue so that the system's solution is understood, but examines the credibility of the user's knowledge. The critic acts as a foil that helps users improve their performance. Therefore, establishing a critiquing dialogue implies that "one must vest power in the hands of the users. They will have a say in what descriptive, normative, and/or prescriptive knowledge remains" [1].

Helping the user validate a body of knowledge without taking control of the problem-solving process is how one can best define the transposition of the critiquing paradigm to advisory systems. In the following, we discuss how this is realized in critiquing systems and which challenges remain.

3. Architecture of a Critiquing System

The term ”critic” was first used by Miller [7] in 1986 to refer to a software program that critiques human-generated solutions.

Different definitions of critiquing systems reflect the way their designers see the role and application of critiquing. Silverman and Mehzer [8], being concerned with different categories of human errors, propose the following definition: ”Expert critiquing systems are a class of program that receive as input the statement of the problem and the user-proposed solution. They produce as output a critique of the user’s judgement and knowledge in terms of what the program thinks is wrong with the user-proposed solution.”

In creative fields such as design, which is one of the most fertile application domains for critiquing systems, the latter are seen as seeking opportunities to improve the user’s work. Yet, these opportunities are not interpreted as signs of underlying limitations of the user. Following these lines, Robbins [9] defines a ”design critic” as “an intelligent user interface mechanism embedded in a design tool that analyzes a design in the context of decision-making and provides feedback to help the designer improve the design.”

A critiquing system has a different dynamic than other advisory tools such as expert systems in that it receives two inputs: (1) the problem description provided by the user or displayed by the computer as the result of a previous process, and (2) the user’s solution to the problem (diagnosis, design, document, etc.). This second entry is the distinctive feature of a critiquing system. It distinguishes a critiquing system from an expert (advisory) system, which computes its own solution and offers it as output to the user. The critic examines the user’s solution in order to provide feedback specific to that solution. Both the expert system and the critiquing system may provide explanations to clarify rationale or justify criticism. This is shown in Figure 1.

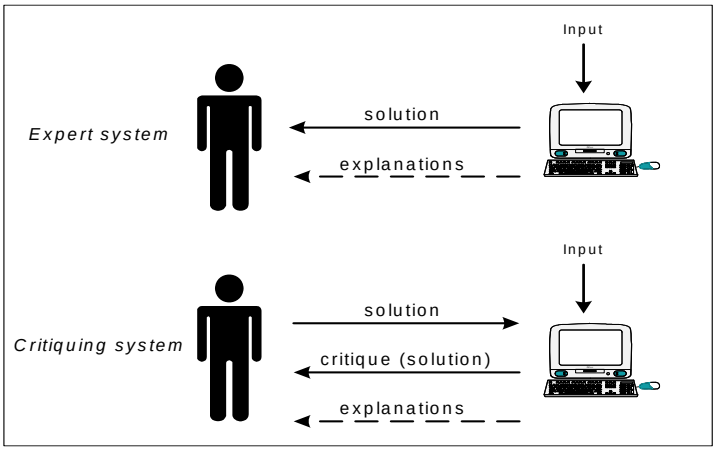

Figure 1. Expert system versus critiquing system

To critique the user's solution, a critiquing system must have (a) a model of expert performance, (b) a model of the types of errors to detect, (c) a means to detect the errors, and (d) a means to notify the user of a detected problem [10].

Basically, the critiquing system uses the mechanism shown in Figure 2: the critiquing system, embedded in a larger software, takes the user's solution and the problem description as inputs. The critic may also have the capability to infer the user's goal (the goal can also be explicitly input by the user). Then, by means of a differential analyzer, the critic compares the user's task result with the one produced by the expert module, which is often a knowledge base of rules. Differences beyond an acceptance threshold are passed to a file of errors that is read by a dialogue generator, parsed into a readable form for the user and displayed on the screen. The feedback (criticism) output to the user may report errors, point out incompleteness, suggest alternatives, or offer heuristic advice. All critics detect breakdowns but not all of them suggest solutions. Some critics require the user to determine how to resolve the problem by making changes, others are capable of suggesting alternatives. These are *solution-generating critics* (ex. JANUS).

The system can rely on the user to specify the type of critique needed or attempt to infer the relevance and timeliness of the critique from the state of the task being performed or user models. The user can then use this criticism and the related explanations to improve performance.

The basic model can be enriched with different mechanisms. Ramachandran and Wilkins [11] propose a model in which critiquing is combined with the use of blackboard architectures, temporal control structures and data manipulation in order to deal with time-critical and resource-constrained domains where priorities of action change over time. A mechanism for knowledge acquisition can also be added, enabling the system to exploit the user's rationale.

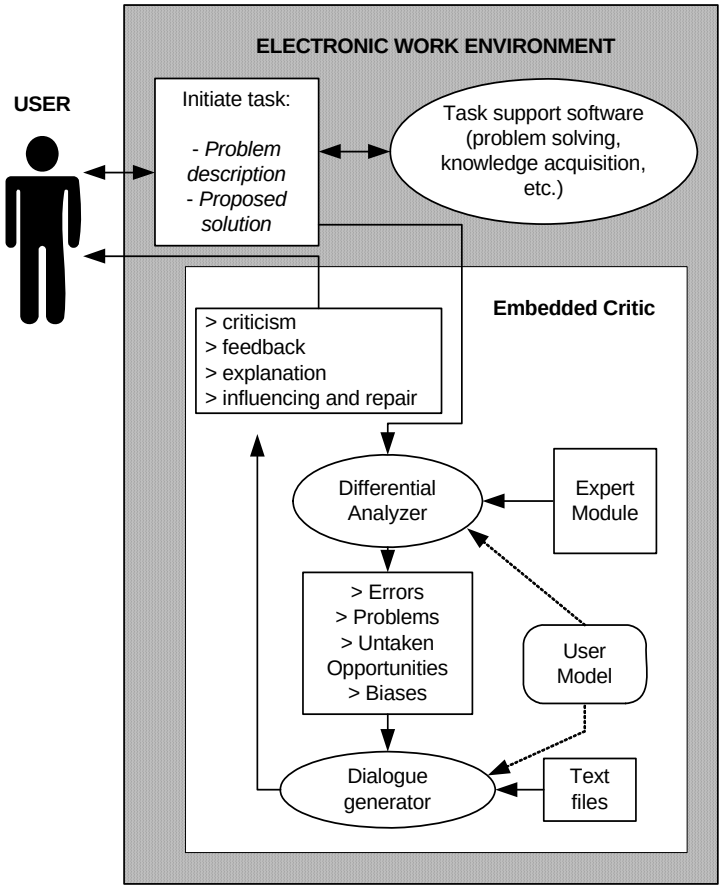

Figure 2. Architecture of a critiquing system, from [1]

Critiquing systems can have as many critics as there are issues to be addressed. As we will elaborate under diverse topics in the following sections, critics may be deployed with consideration for timing (before, during or after the task), process (incrementally or by batch), mode (active or passive), evaluation procedure (analytical or differential), depth of knowledge (shallow or deep), algorithm, attitude (positive appeal, negative condemnation), strategy (querying, hinting, suggesting defaults or analogues, repetition, persuasion, repair and others), and interface dimensions (textual, command-driven, iconic or direct manipulation) [12, 13].

Critiquing systems have been used in several fields, mostly medical applications and (engineering) design. The following are some of the well-known critics and their domain applications. We will refer to some of these in subsequent sections.

- ONCOCIN: clinical consultation system
- ATTENDING: medical support
- JANUS: kitchen design
- Framer: user interface window layout
- KRI/AG: graphical user interface design
- CLEER: placement of antennas on military ships
- VDDE: voice dialogue design
- TraumaTIQ: treatment of medical trauma cases
- AIDA: antibody identification
- UIDA: user interface design assistant
- SEDAR: civil engineering
- ARGO: software development
- ICADS: design of residential apartments
- Critter: engineering design

4. Evaluation Procedures

There are two general approaches to critiquing: differential and analytical. With the first approach, the system generates its own solution, compares it with the user's solution and points out the differences. The system illustrated in the previous section, which uses a differential analyzer, performs such *differential* or *comparative critiquing*. With the second approach, *analytical critiquing*, the system does not generate any solutions on its own but instead analyzes the user-proposed solution for errors and deficiencies.

An example of a comparative critic is the ATTENDING system (anaesthesiology), which first parses the user's solution into a goal/subgoal hierarchy and then evaluates each node in a top-down manner using its own solution generator.

As Robbins [9] points out, comparative critiques can be confusing when several valid solutions exist but are very different from each other. Some domains allow radically different but equally valid solutions. Comparative critiquing can be problematic because it reports a warning when the user's solution is not optimal, thereby hindering different alternatives that may be acceptable. Of course, users in technical fields with only a few possible alternatives in each situation may accept such rigorous systems. However, as [14] remarks, in many areas, such as medical or military applications, the situation is different; in many cases the possible alternatives cannot be judged right or wrong.

Also, as [6] points out, comparative critiquing can be frustrating if the system generates its solution without regard to the user's solution approach. If the user and system solutions differ fundamentally, the critic can only state that the system solution achieves good results, but it cannot explain why the user's solution is less than optimal.

However, a critiquing system may provide comments on the user-proposed plan and propose its own solution(s) as alternatives to it. This approach was illustrated by a critiquing system developed to assist in planning the operations of the San Francisco Water Department [15]. The authors observe that this type of critiquing can be useful in subjective decision-making situations where several valid problem-solving approaches and solutions exist.

Another consequence of the comparative approach in certain contexts may be to lead the user to make their work more like the one proposed by the system [9]. Naturally, this approach can be quite successful in contexts like knowledge-based training systems, where one can assume that the expert system's expertise is superior to the user's. But in other areas, it may hamper creativity or promote overconfidence.

Analytical critics, unlike comparative critics, "guide the users *away* from recognized problems rather than guiding them *to* known solutions" [9]. Systems using *analytical critiquing* detect error occurrences that they then turn into assistance opportunities

[9]. These systems rely more on the user's proposed plan of action sequences (or focus of attention) as a basis for the generation of critiques, and the critiques they formulate are often based on domain knowledge for general problem-solving approaches. Given that their knowledge is not directly solution-dependent but rather based on alternate approaches to handling a given problem, these critics can function in domains that are relatively more incomplete [11].

In the analytical framework, critics do not need to have a complete understanding of the product. An analytical critic checks the user's output with respect to predefined features and effects. For example, JANUS is an analytical critic that uses a set of rules to identify undesirable spatial relationships among kitchen design units, but it does not identify all possible problems within a kitchen design [6].

The analytical approach works well in domains in which well-defined procedures are applied. Such systems can be used to check, for example, whether any mandatory constraints are violated by a proposed solution. This approach seems perfectly suited to design tasks where it can be used to check if all requirements have been met.

These two approaches are compared in the following table.

Table 1. Comparative approach versus analytical approach

COMPARATIVE APPROACH	ANALYTICAL APPROACH
• Generates its own optimal solution and compares it with the user-proposed solution	• Critiques the user-proposed solution with regard to criteria and constraints
• Uses a differential analyzer	• Uses a set of general problem-solving rules
• Hinders acceptable, equally valid alternatives that may be proposed by the user	• Relies on the user-proposed solution for the generation of critiques
• Less interference	• More interference
• Needs a complete problem specification	• Can function in incomplete domains
• Guides the user *to* known solutions	• Guides the user *away from* recognized problems

It should be noted that, ideally, any type of critiquing requires at least a limited understanding of the intentions of the user. Even in a procedural domain where analytical critiquing can be used successfully, having only domain knowledge without any understanding of the particular goals of the user, a critic can only reason about characteristics that pertain to all products.

Interestingly enough, in a study in which human critics were observed, Sumner et al. [16] also identified two evaluative procedures used by designers: an analytical procedure for assessing the solution's features with regard to criteria and constraints,

and a comparative procedure for assessing the solution's features by comparison with the features of another solution. The evaluation criteria used by designers were consistency, compliance with criteria and constraints specified in guidelines, and completeness.

Rhein-Desel and Puppe [14] attempt a synthesis of differential and analytical critics for a medical expert system where the former uses the knowledge of a standard diagnostic problem-solver and the latter checks for a minimum performance standard and makes sure that the user's solution contains no obvious mistakes stored in a library.

The choice of an evaluation procedure for a critiquing system depends largely on its application domain, the characteristics of the task it supports, and the needs of the user in terms of cognitive support (learning, problem-solving, etc.). Furthermore, this choice has important consequences on the user-system interaction modes. The role of an analytical critic is to walk the user through their task, which means more interference, while a comparative critic intervenes only when the user has reached a semi-final result. The intervention strategies of critics and their user perception will be discussed in Section 6.1.

5. Scope

In this section, we discuss the scope of critiquing systems in terms of knowledge management. First, we determine what are the dimensions of the knowledge employed by the user that are tested by the critic, in other words, what is it that the critic checks for. Then, we explain how critics operate with only partial domain knowledge. Finally, in the last part, we try to determine the nature of the tasks for which the critiquing approach can be used. In brief, we try to clarify how the critic can assess the user's knowledge given its own incomplete understanding of the domain, and under which conditions this can work.

5.1 Knowledge dimensions

As Silverman [1] states, the goal of the critic is to critique the credibility of the knowledge used in forming the product, rather than proving the correctness of the user's task result.

This credibility will be questioned if the knowledge used fails any of the following tests:

- *Clarity* (Is the knowledge presented ambiguous?)
- *Coherence* (Is it incoherent, incomplete, inconsistent?)
- *Workability* (Does the knowledge lead to prescriptions or descriptions that are unworkable?)
- *Correspondence* (Does the knowledge agree with reality? Has relevant information been omitted?)

The critiquing system may test one, a few or all of these aspects. For example, word processing critics (spelling or grammar checkers) find clarity and consistency errors but cannot catch semantic incoherencies.

These tests are represented in Figure 3.

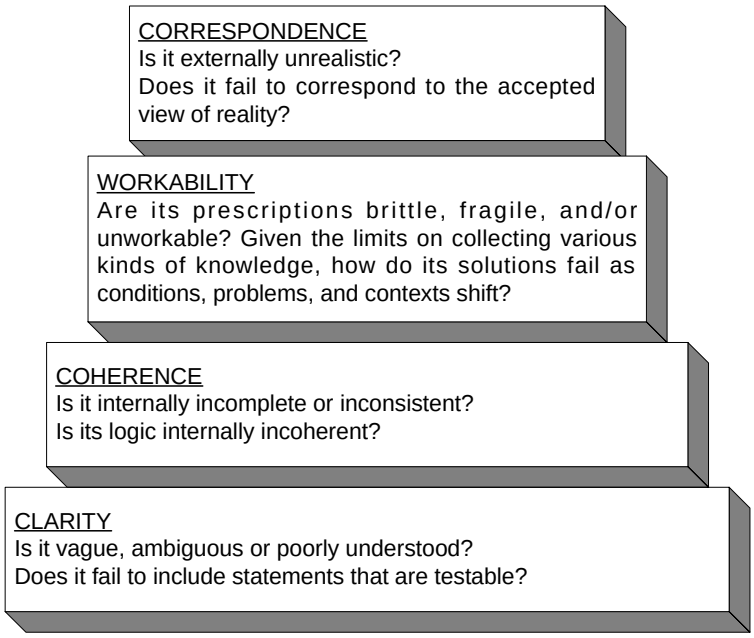

Figure 3. Testing the credibility of knowledge, from [1]

Naturally, this testing implies criticism dialogues, which entail learning on the part of the user. Yet, as Silverman [1] points out, the critic's role is not to teach but to remind an expert of a skill he or she neglected to apply. Learning can of course occur as a by-product of critiquing but must not be considered as one of its objectives.

Critics, contrary to tutors, are designed for expert users. Although critics are very efficient for training like tutors, most tutors are geared to correct errors in the use of new knowledge and skills, while most critics are geared to correct errors in the judgement or use of skills presumably already mastered [1].

In fact, the principal source of difference between critics and tutors lies in the open-endedness of the domains that critics must work in. As [1] explains: "tutors are usually built in narrow domains where all of the errors that the student can possibly make can be identified and added to the expert module. This makes it possible to do a more thorough job of knowledge communication, and much of the technology of tutoring hinges on the completeness of this enumeration of possible user errors. Critics, by contrast, have adopted less complicated technologies because they have to relax the completeness constraint – also, because they do not need all the apparatus of tutoring to remind an expert of a skill he or she neglected to apply. To use a metaphor, a tutor is a power tool useful to solve large problems while a critic is more of a surgical instrument appropriate for smaller repairs".

Critics often are experts on only some aspects of the problem domain. Also, the testing is often performed by several critics, each watching for specific problems. This is why, in a given domain, the more constraints and considerations there are, the

finer the categorization of critics will be. This is reflected in Robbins' classification of design critics [9].

1. Correctness critics detect syntactic and semantic flaws.

2. Completeness critics remind the designer to complete design tasks.

3. Consistency critics point out contradictions within the design.

4. Optimization critics suggest better values for design parameters.

5. Alternative critics prompt the architect to consider alternatives to a given design decision.

6. Evolvability critics address issues such as modularization, that affect the effort needed to change the design over time.

7. Presentation critics look for awkward use of notation that reduces readability.

8. Tool critics inform the designer of other available design tools at the times when those tools are useful.

9. Experiential critics provide reminders of past experiences with similar designs or design elements.

10. Organizational critics express the interest of other stakeholders in the development organization.

A critiquing system can use as many critics as there are knowledge issues to be checked. Therefore, all these critics can be used concurrently. Also, because critics are structured to be independent entities, adding or modifying a critic does not affect the behaviour of the remaining critics [5].

5.2 Operating with incomplete knowledge

The main application domains for which critiquing systems or prototypes have been developed are medical support, design, engineering and word processing (some of the most familiar critiquing systems are spelling or grammar checkers.) Details about the characteristics of different critiquing systems and prototypes can be found in [9] and [1].

In all these domains, critiquing systems operate with partial knowledge. This is another distinctive feature of critiquing systems in comparison with expert systems. Expert systems are infeasible in complex situations in which the potential background knowledge relevant to the task cannot be wholly articulated. Also, expert systems cannot account for the evolution of domain knowledge. Shifts in the problem domain can make an expert system inoperative. Furthermore, these systems only work with complete and accurate problem specifications [5]. In certain fields like design, the

problem specification evolves as the designer continuously reframes the problem. Changes in the problem specification can completely invalidate the expert system's solution.

Critics, in contrast to expert systems, do not need a large or comprehensive rule-base to be effective. A critiquing system can be functional in situations where only partial knowledge is available because it tries to infer solutions using only the information at hand. The critic simply checks the reliability of the data and the solutions and helps the user not to overlook facts or relevant hypotheses.

In fact, a critiquing system can critique a product without knowing how to design it; it can check the spelling and grammar in a document without knowing how to make a sentence. This is of course true for the human context too, although not for the same reasons. A movie critic doesn't necessarily know how to make a movie and a literary critic does not necessarily know how to write a novel; however, this is not because of a lack of domain knowledge but rather of creativity. The software critic does not have a general understanding of the problem, but it knows the rules of the game and can recall them if asked to do so.

Table 2. Differences between expert and critiquing systems

EXPERT SYSTEM	CRITIQUING SYSTEM
• Takes the problem specification as input	• Takes the problem specification and the user's solution as input
• Provides a solution to the problem as output	• Checks the reliability of the user's solution. Reports errors, points out incompleteness, suggests alternatives, or offers advice
• All relevant background knowledge must be articulated	• Does not need a comprehensive rule-base
• Does not resist shifts in problem domain	• Can operate with partial knowledge
• Problem must be accurately specified	• Problem can be reframed

This makes critics well-suited for design tasks in complex problem domains. Design problems are ill defined, they do not have an optimal solution, and the problem cannot be precisely specified before attempting a solution [6]. Yet, at each stage, the critic can provide support by applying generic design knowledge.

However, as [14] remarks, this approach is not very promising for complex domains such as medical diagnosis. As these authors observe, one reason for the missing breakthrough of critiquing systems despite broad acceptance might be the complexity of critiquing knowledge in vague domains.

As a matter of fact, a missing element in the critiquing systems literature is the definition of domains in which the critiquing approach *cannot* operate. It is certainly not enough to know that critics can be functional in ill-defined domains because they provide generic knowledge; we need to know the characteristics of those domains in terms of goal definition, reasoning processes, problem structure, etc. Only then can we clearly determine why the critiquing approach is effective for a complex and vague domain like design and not for another complex and vague domain like medical diagnosis.

It is interesting here to look at the way Simon et al. [17] describe the two application domains of medical diagnosis and design. Besides clearly structured puzzle-like programs, which are noncomplex optimization problems, they distinguish two types of complex problems, one with well-specified decision problems and the other with ill-specified decision problems. Medical diagnosis is of the first kind. This is a task that, for all the knowledge it calls upon (large bodies of semantic information are involved), is still well structured, with clear-cut goals and constraints.

Design, on the other hand, offers a good example of what is involved in solving ill-structured problems. A problem-solving task is ill-structured where the goals themselves are complex and sometimes ill-defined, and where the very nature of the problem is successively transformed in the course of exploration. Ambiguous goals and shifting problem formulations are typical characteristics of design problems. The initial goals are modified and substantially elaborated as the designer proceeds with the task. New criteria, new possibilities and new requirements are suggested as the task progresses. Throughout the whole process of design, the emerging conception provides continual feedback that reminds the architect of additional considerations that need to be taken into account.

The authors write: “With the current state of the art, it is just beginning to be possible to construct programs that simulate this kind of flexible problem-solving process. What is called for is an expert system whose expertise includes substantial knowledge about design criteria as well as knowledge about the means for satisfying those criteria. Both kinds of knowledge are evoked in the course of the design activity by the usual recognition processes, and the evocation of design criteria and constraints continually modifies and remolds the problem that the design system is addressing. The large data bases that can now be constructed to aid in the management of architectural and construction projects provide a framework into which AI tools, fashioned along these lines, can be incorporated”.

Critiquing systems begin to be useful when expert systems cease to operate effectively for the reasons mentioned above. Critiquing systems provide a tool for these complex and ill-defined domains precisely because their role is not to offer a solution but to provide feedback on a given state of affairs. Therefore, our answer to the question of how one can characterize the application domain of critiquing systems is that a critic can operate in a “vague” domain as long as problem-solving within that domain is performed on a constraint-satisfaction basis. When discussing the role of critics in design in Section 7.2.1., we will see how these systems help designers by considering only one state of the task at a time.

Of course, if critiquing systems are less successful in domains like medical diagnosis, it is not because the problem is well defined but rather because partial knowledge is not helpful enough in such domains.

A simulation study by Kuilboer et al. [18] shows how critiquing can function for medical diagnosis despite missing information … with human critics. Considering that the available data for the critiquing system to be designed will be limited to the data that the general practitioner (GP) is able and willing to enter into his/her electronic medical record, the authors were faced with the decision between a noninquisitive and an inquisitive system design. However, their simulation study with human reviewers showed that limited availability of data did not impair the generation of useful critiquing statements. Referring to Miller, the authors observe that there may be reasons why it is good to be generic because, for example, comments are less likely to be wrong. Fischer and colleagues [5] also report that one study involving both amateur and expert kitchen designers showed that HYDRA's generic critiques helped both categories of designers even though its rule-base contained only 24 critique rules.

Kuilboer et al. observed that it is possible for human reviewers to generate critiquing comments despite the fact that they often missed information (90 requests for further information were stated over 87 visits). They also observed that the majority of the comments/critiques (74%) were left unchanged by the reviewers after the participating GPs had provided additional information for 64% of the requests. On the other hand, 26% of the comments were changed and 15 new comments were made, showing that additional information may change some comments or give rise to additional ones.

They concluded that the fact that requested additional information was available in many cases supports the option to build an inquisitive system; however, given its higher intervention rate, such a system will have a greater impact on the physician's normal routine, and therefore will run a greater risk of being rejected. This idea is reinforced by their observation that the majority of comments (critiques) remained unchanged when the additional information requested became available. However, software critics may not be able to extrapolate from partial knowledge as easily as their human counterparts in the first place.

Other studies show that using a critiquing system in such cases is not only efficient, but also safer compared with other automated systems. The results of the empirical study by Guerlain et al. [19] in the domain of antibody identification suggest that cooperative problem-solving is superior for brittle cases (when knowledge is incomplete) when using the critiquing model of decision support. The study compares two conditions, one in which the computer critiques the user at work and one in which the computer performs that task. There was no statistical difference in outcome errors for cases in which the computer's knowledge was competent. Yet, in a case in which the system's knowledge was brittle, misdiagnosis rates increased by 29% for users of the automated system. This can probably be explained by the way decision control is handled in the critiquing approach (see Section 7.1): the critic does not assume the problem-solving responsibility; therefore the user stays in charge and vigilant.

5.3 Task properties

It seems therefore that the complexity of domain knowledge can be compensated to a certain extent if the tasks to be performed with the help of a critiquing system are well structured. For example, the critiquing system can be used to help the user manage multiple factors or parameters. If engineering design is a fertile ground for the use of critiquing systems, it is partly due to the fact that critics can point out many weaknesses pertaining to failure in specification or requirement satisfaction, failure in life-cycle system analysis, flaws in reliability, maintainability, interoperability, susceptibility, usability, etc.

Miller himself, who was the first researcher to see great potential in the use of critiquing systems for decision support (ATTENDING was the first prototype system), observed that (1) it is important to choose a constrained domain, which led him to use the technique for hypertension rather than the broader field of anaesthesiology; and (2) critiquing systems are most appropriate for tasks that require that the practitioner remember or consider a lot of information such as procedures, risks, benefits, side effects and costs.

Critiquing can be very effective for such well-structured tasks, especially if the critiquing strategy is based on an error model of the domain. AIDA, the Antibody Identification Assistant [19] uses five types of error-checking mechanisms: checking for errors of commission, checking for errors of omission, checking for an incomplete protocol, checking that the data are consistent with the answer, and checking that the answer is plausible given prior probabilities.

AIDA monitors for errors of commission and errors of omission and ensures that practitioners follow sound protocol (using independent, converging evidence to rule out all nonconfirmed antibodies), that the final solution set makes sense (i.e., that there are no unexplained reactions and that the pattern of reactions is consistent with the types of antibodies confirmed), and that the final solution set does not violate constraints inherent in the domain (e.g., anti-E alone in an Rh-negative patient is very rare).

Given the objectives of our project as stated in the introduction, the question that comes up is whether command and control can be a good candidate for the use of critiquing systems. One definition of command and control is “that integrated function which enables the establishment of objectives and associated selection of action alternatives for force deployment. The basic C2 functions are information gathering, situation assessment, action selection, response planning and execution, and monitoring of the implementation of alternative courses of action” [20].

Silverman [1] writes about this environment: “[This is] yet another domain where automation levels are high and expert critics rather than expert advisory systems should prove to be useful. For example, in the pilot’s cockpit or in the control room consoles of power stations, air traffic towers, satellite control centres, and the like, the architecture of the critiquing process is eminently well suited to silently observing the human process, and only interrupting when a problem arises. One research question

for this class of domains that does not arise in others is the difficulty of having the computer discern what stream of inputs from the user are associated with which class and/or problem. […] Command and control environments are continuous and real-time. There is no point at which the user says now I am done with that anomalous condition and let me move on to the next one. The critic would need to infer this."

However, in a high-level task like course of action evaluation and selection, the critiquing system can help the user test different alternatives against a whole array of requirements and constraints. Mezher et al. [13] describe the decision-making process as follows: The first step in the decision analysis process is identifying the problem. The second step is to identify objectives and alternatives. The third step is to model the problem and to evaluate alternatives. The fourth and fifth steps are to choose the best alternative and to perform a sensitivity analysis. The final step is to implement the chosen alternative. According to Silverman and Mezher [8], expert critiquing systems can be used to help managers at different stages of this process. First, critics compare the current decision with a store of common erroneous decisions. Second, critics check the current decision for violations of good decision rules, cues and constraints that managers tend to overlook or misuse. Finally, critics provide feedback and situated tutoring on syntactic-semantic-pragmatic specialties missing from typical manager training.

It seems therefore that it is not the domain as such but the common tasks and subtasks performed within that domain that determine whether or not the critiquing approach is appropriate. Mezher et al. [13] remark that critics appear most useful when inserted into existing automated environments for users at competent/proficient skill levels attempting semistructured tasks, and for intermediate users attempting well-structured tasks.

In brief, the critiquing system must operate within a framework that is sufficiently structured to permit confirmation that norms and standards have been respected, that there are no obvious errors, and that significantly suboptimal conditions do not exist. Nevertheless, further investigation of task properties (as in *generic tasks*) is needed in order to better define the applicability of critiquing systems at a general level.

6. Critic Feedback

The feedback provided by a critiquing system can be a message displayed in a dialogue box or feedback pane or it can be visually indicated in the document itself, as is the case with spelling or grammar checkers. The feedback message can be a canned or static text, a hierarchically planned text as for explanations, or a dynamic text generated as differences between the user-proposed solution and the system-generated solution are detected (if differential critiquing is performed).

Apart from the format and the provision mechanisms, a critic can also be characterized relative to its intervention strategy. Defining the intervention strategy for a critic is determining when and how the critic should signal a breakdown. In the first part of this section, we will present the general dimensions along which critics are categorized with regard to intervention strategies and their impact in terms of user perception. We discuss the ergonomics of different intervention strategies in Section 7.2.1.2.

In the second part of this section, we present works that discuss how critic feedback can be adapted to the user's profile and needs. Finally in the last part, we touch on the user acceptance of critiques through the use of explanatory and argumentative discourse.

6.1 Intervention strategies

Critic feedback can be classified along the following dichotomies:

- *Active vs. passive*: active critics continuously critique the design, whereas passive critics wait until the user requests a critique. This feature corresponds to the proactive vs. on-demand options in the more general context of interface design.

- *Reactive vs. proactive*: reactive critics critique accomplished work while proactive critics try to guide the user before the user makes a specific decision.

- *Positive vs. negative*: positive critics praise while negative critics criticize the user's work.

Active critiquing agents continuously monitor user actions. Passive critics usually evaluate the (partial) product of a design process, not the individual user actions that resulted in the product. Passive critics, less expensive computationally speaking, can be used if criticism is needed only at certain times.

A protocol analysis study [21] shows that passive critics were often not activated early enough in the design process to prevent designers from pursuing solutions known to be suboptimal. Subjects invoked the critic at a time when the effort of repairing was too expensive. However, Fischer et al. [5] state that their interactions with professional designers showed that active critics are not a perfect solution either:

they can disrupt the designer's concentration on the task at the wrong time and interfere with creative processes. Interruption becomes even more intrusive if the critic signals breakdowns at a different level of abstraction compared with the level in which the task users are currently engaged.

Thus, for active critics, intervention strategies must be further specified for timing. Intervening immediately after a suboptimal or unsatisfactory action has occurred (immediate intervention strategy) has the advantage that the problem context is still fresh in the user's mind and the user still knows how they arrived at the solution. The problem can be corrected immediately without causing dependent problems. A disadvantage of active critics is that they may disrupt a cognitive process, causing short-term memory loss. The user then needs to reconstruct the goal structure that existed before the intervention. Both modes can be used concurrently as in JANUS, which is an active critic but can also be invoked by the *critique all* command.

The interface design is another factor to consider because, among other things, it can help active critics be less disruptive. When critiques are displayed in a separate window, users can choose to read and process the message immediately or do so after completing an action in progress.

Intrusiveness is one of the dilemmas of critic interface design: critiquing systems can either interfere too much or fail to provide sufficient help. Based on their experience with Framer, Fischer et al. [6] observe that critics should intervene when the user has made a potentially suboptimal choice and later choices are likely to depend on it. By pointing out problems early, critics can avoid the cost of revising whole chains of decisions.

A slightly different problem arises in learning contexts. As noted in [22], the demand to learn cannot originate with users when they are unaware that additional functionality exists in the system. The system must take the initiative. To avoid the problem of these systems becoming too intrusive, a metric is necessary for judging the adequacy of the user's action… "A major challenge for this class of systems is to keep them quiet most of the time. A coach or a critic must be capable of diagnosing the cause of a student's misunderstandings and then judiciously deciding, on its own, when to interrupt and what to say. Interrupting too often can destroy motivation, but too few interruptions results in learning experiences being missed or floundering on the part of the user".

The other interface feature that critic designers can decide upon is the proactive or reactive nature of its assistance. Critiquing by reacting occurs when [7]:

1. Specific rules can be written for each type of wrong answer;

2. The rules for assessing user solution optimality are objective and few;

3. Only one or two possible correct outcomes exist for that task;

4. Each subtask can be critiqued independently of the others.

In the reactive mode, the analyzer looks up the feedback in a table or knowledge base of rules. As Silverman [1] summarizes: if conditions (1) through (3) deteriorate and the domain has numerous feasible solutions and primarily subjective standards for evaluation, then critiquing requires evaluation of the risks and benefits of a number of alternative solutions. These are dynamically instantiated with the values of the current problem to determine if the user-proposed solution falls within the array of acceptable solution sets. Miller calls this critiquing by *local risk analysis*. Finally, if condition (4) does not hold for a given domain, then critiquing can simply address an isolated portion of the proposed solution. Critiquing by *global plan analysis* occurs when the critic cannot address each sub-decision in isolation but must adopt a global view to examine the totality of what the user is suggesting. This is a form of critiquing neither Miller nor anyone else has yet attempted.

Finally, negative vs. positive critics can be used to counter the natural tendencies of decision-makers, as illustrated by Vahidov and Elrod's work [23] (see Section 6.3). Vahidov and Elrod's model provides a way of controlling the intensity of positive/negative critique: "If we emphasize negative critique by diminishing positive feedback, this will stress pessimistic bias in decision-makers, thus forcing the proposer to search for more alternatives. Therefore, the negative critique would tend to stress divergent behaviour. On the other hand, if we diminish negative critique, the positive feedback would introduce an opposite (optimistic) kind of bias in decision-makers, making them more lenient in accepting the proposed decisions. Hence, the positive critique is likely to stress convergent behaviour. We can therefore manipulate the intensity of one or the other type of critique to counteract the user's natural bias on the divergence-convergence scale."

Furthermore, Fischer et al. [6] note that positive performance critics help users retain the good aspects of a product in further revisions; positive educational critics reinforce the desired behaviour and aid learning.

6.2 User-tailoring of critiques

Mastaglio, who is one of the pioneers of critiquing systems, was the first researcher to discuss the adaptivity of computer-based critics by means of user modelling [24]. He writes that to avoid repetitive messages and to accommodate different user preferences and users with different skills, a critiquing system needs an adaptation capability. Critics can be adaptable or adaptive. Systems are called adaptable when the user can change the behaviour of the system. An adaptive system is one that automatically changes its behaviour based on information observed or inferred. This feature, desirable for critiquing systems, can be implemented by simply disabling or enabling certain rules, by allowing the user to modify or add rules and by making the critiquing strategy dependent on an explicit, dynamically maintained user model.

Silverman [1] discusses the use of overlay models for different skill levels so as to make criticism more oriented toward the specific needs, current areas of skill, and specific cognitive biases of the users. Hence, the dialogue generator could better tailor its text to say the right thing at the right time, and a rigorous basis would exist for

deciding which critiquing strategy to invoke and which level of depth of knowledge to use. User modelling, he recognizes, is a worthy long-term research objective of the critiquing paradigm because it can indeed control the kind of critique, its form and timing. Yet, almost none of the existing critiquing systems use a user model to tailor its feedback to the expertise or preferences of the user. In fact, integrating a user model in the critic's architecture can make it much harder to implement, not to mention the cost involved or the frustrations caused by it: a user model is never entirely satisfactory and it must always be improved [1]. Indeed, two of the principal dilemmas encountered in user modelling are dealing with uncertainty about the information represented in the model and maintaining its consistency as the information evolves [25].

However, simpler techniques have been used to attune the critiquing system's feedback to the user's needs. For example, in JANUS and TraumaTIQ, the critic addresses the user's goals. In the former, the user explicitly states their goals by filling out a form, while in the latter, the goals are inferred from the user's actions.

In ONCOCIN, a user-selectable agenda is displayed. The user can select certain terms from the menu that correspond to the parameters used by the system and obtain an explanation. Thus the explanation tree, which is in fact the hierarchy of rules that fired to detect the sub-optimal elements of the user's plan, can be traversed according to the needs of the user.

Fischer et al. [5] discuss the complementary roles of three embedded critiquing mechanisms – generic, specific and interpretive – for design critics. These are described as follows.

A generic critiquing mechanism can be prototyped that will be knowledgeable about commonly accepted design principles and standard design practices (accepted standards or regulations).

A specific critiquing mechanism supports the process of problem reframing. It enables only those critics pertinent to the current partial specification, and as such, embodies domain knowledge concerning situation-specific design characteristics that not every design will share.

Finally, the interpretive critiquing mechanism supports designers with varying interests and differing areas of expertise to work together by allowing design knowledge to be defined and bundled into personal or topical groupings. HYDRA contains a collection of critics that embody different areas of domain expertise, different design styles, and often diverging opinions.

Along the same lines, Nakakoji et al. [26] propose to offer a perspective-based critique by means of a perspective manager that would function as a filter for the knowledge base. At the request of the critic manager, the perspective manager uses the explicitly represented perspectives to enable only relevant critic rules from the knowledge base. Hence, when critiquing the design of a kitchen, this component recommends that the dishwasher be placed on the right side of the sink only if the

user is not left-handed. Using this knowledge about design intentions, the critiquing system can tailor its analysis and argumentation activities to better support human judgement.

The study by Rothschild et al. [27] points out some of the complexities of tailoring critiques. It shows that in addition to accuracy, there are at least three other axes possible in evaluating prose critiques: scope, level of detail, and wording. We have already discussed scope, which is concerned with the area of a knowledge domain that the system chooses to encompass. As to the level of detail, they observe that experts differ on how much information is appropriate for a given discussion and as to which points are vital and which are not. They observe that there are also competing requirements that the critique be succinct for efficient use in an interactive setting while being detailed enough to provide an explanation for its conclusions. Finally, when it comes to wording, they note that evaluators may consider that clarity can be increased through the application of wholly subjective standards.

6.3 Use of explanation and argumentation

Although explanation facilities are mainly used to show the correctness and usefulness of system recommendations [28], they must be integrated somehow into the critiquing system's environment. A critiquing system tests the credibility of the user's solution by examining the knowledge and judgement used to reach the solution. To justify its criticism, the critic needs to provide explanations.

As discourse structures, critiques and explanations are related in many ways. First, a critique is often an explanation of the rationale underlying a disagreement. When describing their system ONCOCIN, Langlotz and Shortliffe [29] say: "A critique is an explanation of the significant differences between the plan that would have been proposed by the expert system and the plan proposed by the user."

Conversely, explanations are a component of critiques as for any other argument structure. Justifying a critique by explanation is important for several reasons. First, justification is necessary for user acceptance of the critique. A user will not accept a critique if it is not supported by valid reasoning. (This, of course, applies more to comparative critiquing where the system computes its own solution. In the analytical approach, the critic simply reminds the user of knowledge that he or she possesses but has simply overlooked.)

Explanations can also provide insight into possibilities the user has not foreseen. In the decision-making context, an explanation component capable of presenting different alternatives and opinions and their corresponding advantages and disadvantages can be very useful.

Finally, as Robbins [9] remarks, increasing support for the presentation of advice helps designers make more effective use of the feedback they are given. One of the obvious benefits is learning. Users learn best when they are situated in the actual context of their work and are able to receive explanations from an expert who can

clear up misconceptions and clarify understanding. This helps them restructure their own knowledge.

In order to foster cooperative work, Fischer and Mastaglio [22] recommend that critics contain domain knowledge represented in a form that is applicable to both problem-solving and explanations. This would create an integrated environment for both learning and problem-solving.

The generation of explanations can require more or less complex techniques. However, hypertext techniques have been shown to be very efficient for providing contextualized explanations [30]. One contribution of Fischer and colleagues' work has been the incorporation of hypertext into the critic's feedback loop and the creation of what they call "minimalist explanations". Via hypertext jumps, the user can obtain more in-depth explanations.

Argumentation is another technique used in cooperative settings based on the use of discourse. Vahidov and Elrod [23] propose a framework for an active DSS with both negative and positive critiquing agents. They further introduce the use of argumentation and debate between these agents as a potential way to better inform the decision-making process. The use of antagonist agents is based here on the intuition that human decision-makers tend to assess their choices in terms of advantages and disadvantages, pros and cons. The positive critic plays the role of the proponent of the user-proposed solution by defending, substantiating and augmenting it. The negative critic is the opponent of the user's solution and plays the role of the devil's advocate. These critiquing agents, respectively called the angel and the devil, enhance quality decision-making through polarization. The angel tries to identify the strengths of the proposed solution, while the devil tries to find possible violations, drawbacks and problems. The way they substantiate their claims is by the use of argumentation. When asked to do so, they relate their critiques to warrants. The critiques and their warrants, as well as the criterion they are associated with and the triggering conditions of the critique, are represented in the system in a critique frame.

An interface of their system is represented in Figure 4.

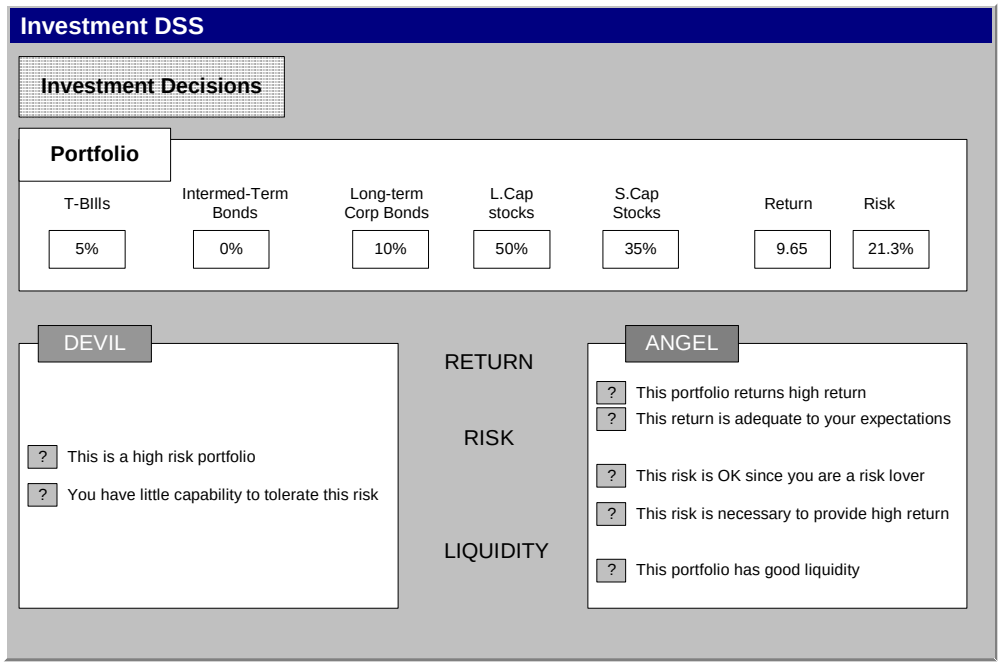

Figure 4. Critiquing and argumentation, from [23]

In this example of decision-making for investment, the devil and the angel provide objective-related (with regard to the goal), preference-related (with regard to the user's preferences) and reactive critiques (in reaction to the other agent's criticism) relative to the criteria of return, risk and liquidity. The dialogue on risk illustrates all these critiques: while being reactive, remarks concerning high return are objective-related and those concerning risk tolerance are preference-related.

Fischer and his colleagues have also used the dialectical technique [5]. In HYDRA, each critic rule is linked to the argumentation component that provides the design rationale. Because design rationale contains design issues accompanied by positive and negative argumentation, critic explanations in this form help the designer understand why the current design situation may be significant or problematic.

Finally, the use of explanation and argumentation can also be beneficial for the system. Interestingly, an empirical study of a critiquing system [16] found that experienced designers often explained their decisions in response to criticism with which they disagreed. This means that critiquing does in fact engage the user in an argumentative exchange. Although this may be perceived as a negative result in that the user may not carry out the suggested changes, it does show that critiquing promotes reflection and makes designers externalize their design rationale and expertise. This can be exploited to support knowledge acquisition for critiquing.

Ramachandran and Wilkins [11] discuss how, by analyzing the user's rationale, an interactive knowledge acquisition process aimed at capturing the missing or incorrect piece of information can correct the incompleteness and inaccuracy in domain knowledge.

Using the right mechanisms to feed the critic with the user's knowledge will not only increase the assistance capabilities of the computer, but will also make the critiquing dialogue in the human-computer environment as close as it can get to its human-human counterpart in an ideal situation. Silverman [1] describes this as "a mutual exchange, a search for truth, and a dialogue in which both parties can benefit, not just the recipient of the criticism".

7. Critiquing and Cognitive Support

Critiquing systems provide cognitive support in many ways. Certain aspects of it have been discussed in the literature in order to justify the utility of critiquing systems or a specific approach to critiquing. We have structured this information under three main topics which highlight the critics' distinctive features: critics transfer decision control to the user, support collaborative problem-solving, and help the user notice errors and judgement biases.

The first dimension is concerned with the "attitude" of critiquing systems: critiquing a user-proposed solution, as opposed to making the user accept the system's recommendations, has the advantage of leaving decision-making control within the user's hands. The second dimension is concerned with how the critic can augment the user's cognitive processes and improve their mental model of the situation: by signalling breakdowns and providing explanations, the critic promotes reflection, which enhances the quality of the user's task and gives him or her the opportunity to learn. Finally, the last dimension is concerned with a specific type of decision support: pointing out knowledge and judgement errors can avoid costly consequences for the user.

Throughout this section, we report the empirical studies, if any, which provide evidence for or refute these claims.

7.1 Decision control

There is evidence that experts make early judgements about a problem, rapidly generate partial solutions [31], and anticipate the consequences of their solutions and the constraints they must satisfy [32]. With his recognition-primed decision model [33], Klein shows that in time-compressed environments, experienced people make rapid decisions using situation assessment to generate a workable course of action and mental simulation to evaluate that course of action. Therefore, in certain circumstances, rather than deciding upon a system-generated solution or advice, many competent users may prefer to interact with a critiquing system that can give them a second opinion about their own solution.

It has also been pointed out [34] that advice-giving systems place the user in the paradoxical and extremely difficult role of monitoring, and overruling when appropriate, the recommendations of a machine whose competence is presumed to exceed that of the user. When Langlotz and Shortliffe [29] convert their diagnostic expert system, ONCOCIN, into a critiquing system, they report "The most frequent complaint raised by physicians who used ONCOCIN is that they became annoyed with changing or 'overriding' ONCOCIN's treatment suggestion." The designers found that since the doctor's treatment plan might differ only slightly from the system's treatment plan, it might be better to let the physician suggest his or her treatment plan

first and then let the system decide if the difference was significant enough to mention to the doctor.

Thus, as Robbins [9] says, the critiquing concept arose from the realization that the system should support doctors without infringing on their decision-making authority.

As Fischer et al. [6] observe, automation is a two-edged sword. "At one extreme, the system is a servant, relieving humans of the tedium of low-level operations and freeing them for higher cognitive functions... At the other extreme, automation can reduce the status of humans to 'button-pushers' and strip their work of its meaning and satisfaction. People's willingness to delegate tasks depends on the extent to which they trust they will receive satisfactory solutions. Critics allow – and indeed force – them to exercise a great deal of personal control over, and to take responsibility for, the design of the product".

The problem of responsibility is a major one indeed. Knowing that they will be held responsible for their decisions, a user will not accept solutions that they have not participated in elaborating. If, on top of that, the recommendations are not fully understood, they may simply discard them.

The critiquing system can be an effective alternative in that, rather than placing the user in the position of evaluating the quality of a system-generated result, it provides feedback that the user is free to accept or reject. Also, the critiquing process engages and challenges the user's own positions and assumptions, thus supporting their understanding of the problem.

On the other hand, a tendency to become overconfident with automated systems has also been observed. Complacency and lack of trust have been identified as two opposite reactions to automated environments, each of which can result in ceasing use of the system or suboptimal performance [35]. Several conditions under which complacency may occur were reported in [36].

Similarly, Guerlain et al. [10] write that a serious concern with having users decide about a decision support system answer is that they may become overly reliant on the computer or may be led 'down the garden path', failing to adequately evaluate the computer's conclusion.

However, the problem of over-reliance does not seem to apply to all user categories. The results of a study by Guerlain et al. [37] show that the use of a partially automated system can cause practitioners to make more errors on cases where the computer's knowledge is inappropriate than when using a critiquing system. This was uniformly true regardless of the practitioner's level of competence on that strategy, although it was most likely caused by different factors. For practitioners without prior knowledge of the strategy used by the computer, failure to detect the inappropriateness of using the strategy was likely due to the human's lack of the knowledge required to apply that strategy. Practitioners who had prior knowledge of a strategy similar to the computer's may have had an incorrect mental model of the computer's knowledge because they were not getting periodic feedback from the

computer as were the users of the critiquing system. For practitioners with the same level of knowledge as the computer, over-reliance may have been the cause of worse performance for the case where the computer was incompetent.

This team shows [19] that degradation of results (in cases where the computer's knowledge is brittle) did not occur when the computer used its knowledge to critique a user who was performing the problem-solving task while the computer "looked over their shoulder". According to the authors, these results provide initial evidence that placing an expert system in a critiquing role may be a more effective form of decision support than asking the user to critique the computer.

Kuilboer et al. [18] show that leaving control with the user brings about better reactions on the part of the user. In a simulation study, they investigated the kind of comments that could be made on general practitioners' work and the way the latter assessed those comments. The majority of comments (made by human reviewers) were those critiquing the medication prescribed and the GP's therapeutic strategy in general. Interestingly, these were judged very positively by GPs. The authors note: "This observation seems to be in contradiction to studies that have shown that diagnostic systems have had little impact on daily clinical practice. Possibly the kind of diagnostic support that is appreciated by physicians (support with diagnostic work-up) differs from the kind of support that diagnostic systems have provided in the past (support with differential diagnosis). When describing major obstacles to the implementation of decision support systems, Taylor identified 'loss of clinical control' as one of the reasons diagnostic systems have achieved so little. The fact that critiquing leaves the physician in control could account for our finding that general practitioners appreciated the diagnostic comments."

However, they also observe that it is difficult to make the physicians accept these recommendations. Indeed, the authors noticed that in a number of cases the GP could see that a comment was relevant, but they could still strongly disagree with its content.

In their study on critical reasoning biases, Klaczynski et al. [38] show that when presented with pertinent information, individuals apply a wide range of reasoning tactics to preserve the integrity of their goals and beliefs. They readily assimilate belief-enhancing information without engaging in critical scrutiny, but they engage in extensive data analysis and deep processing when facing belief-threatening information. The use of sophisticated strategies in the latter case can result in refutation of the data.

In the case of their experiment, Kuilboer et al. [18] simply conclude that more insight is needed to determine the reasons for rejection and to be able to make a distinction between reluctance of the physician to accept advice and disagreement of the physician with the content of the advice.

7.2 Cooperative problem-solving

One aspect of the cooperative attitude of critiquing systems is that they interact with the user through criticism dialogues. Naturally, to be successful, cooperative problem-solving systems must settle issues such as what role each partner should play, when to take the initiative and how to communicate with the partner [6]. These issues pertain to the intervention strategies of critics.

Yet, more important than this communication issue is the way critics empower the user with the system's knowledge. The core task of critics is to recognize and communicate debatable issues concerning a product within the problem-solving context [6]. As we already discussed, the critiquing system does not solve the problem for the user but acts as a cognitive amplifier in that it helps him or her to better manage the problem-solving constraints. Fischer and Mastaglio [22] describe the cooperative human-computer setting as follows: "humans use common sense, define the common goal, decompose problems into sub-problems and so on. Computers provide external memory for the human, ensure consistency, hide irrelevant information and summarize and visualize information."

By providing "contextualized access" to knowledge [30], the critic does exactly what an intelligent support system should do, that is, "facilitate access, application of knowledge, and user learning" [39]. The critic is tightly integrated into the user's task and can therefore provide feedback relevant to the context of decision-making (unless of course critiques are provided in batch mode). As Fischer et al. [5] observe, in situations of information overload, the critical resource for users is not information, but rather the attention with which to process information. This has been noted for many contexts: availability of knowledge is not sufficient; the user must be given access to relevant knowledge in a timely manner [40]. Moreover, by providing contextualized access to knowledge, critics can reduce the cost of learning and influence the effort-accuracy trade-off involved in accessing domain knowledge.

Finally, by providing contextualized access to information, critics facilitate learning: users understand the purposes or uses of the knowledge they are learning; they learn by actively using knowledge rather than passively perceiving it, and they learn at least one condition under which their knowledge can be applied [6]. Hence, learning occurs as a natural by-product of the problem-solving process.

The critiquing system satisfies all cooperation requirements, given that, as Fischer et al. [6] enumerate, they (1) save users the trouble of explicitly querying the system for information; (2) notify users of situations indicating the need to reflect and provide access to information fueling reflection; (3) deliver relevant information of which users were unaware; and (4) afford learning on demand by letting users access new knowledge in the context of actual problem situations.

In fact, in the human-critic setting, all the ingredients are there for the user and the system to form a "joint human-machine cognitive system" [31].

In the following paragraphs, we discuss two principles of the cooperative problem-solving paradigm as instantiated by critiquing systems. In the first part, we describe how critiquing systems support the user's reflection-in-action during the problem-solving activity. Given the nature of the process described, this part will focus more on design critics. Within this framework, we will also discuss how critics support the user when switching from one mental context to another, as well as the ergonomic aspects of joint human-critic problem-solving.

In the second part, we discuss another parameter of cooperative problem-solving, which is visibility. Through their interactions with the user, critiquing systems not only enhance performance and learning, they offer a unique opportunity for the user to construct a mental representation of the system.

7.2.1 Reflection-in-action

Fischer et al. [5], who have extensively explored the domain of design critics, define critiquing as "a dialogue in which the interjection of a reasoned opinion about a product or action triggers further reflection on or changes to the artefact being designed".

As such, critics can be very useful in environments where the final product is the result of continuous refinement of initial, abstract specifications, as in design tasks. Design problem-solving, as Bonnardel [41] defines it, requires designers to be creative and express evaluative judgements. During iterative problem-solving, designers (i) construct partial solutions based on their current understanding of the design goals and specifications, and (ii) evaluate those solutions with respect to various criteria and constraints. The results of these evaluation steps feed back into the designers' understanding of the problem and lead designers to focus their attention on specific features of the current solution, which guides further development or modification of these features. Thus, evaluation plays a major role in design because each successive evaluation step guides the course of design activity. The final solution to a design problem is obtained progressively, through iterative cycles of solution generation and solution evaluation.

In such environments, a critiquing system would promote Schoen's action-breakdown-reflection cycle [42]. A breakdown occurs when the designer realizes that non-reflective action has resulted in unanticipated consequences – a realization described as "the situation talks back". This process takes place within the time period during which the decision to act has been made but the final decision about how to act has not. This is the time period during which reflection can still make a difference in what action is taken. The critiquing system optimizes this situation by (1) pointing out problematic situations that may remain unnoticed, (2) communicating debatable issues and different perspectives surrounding a design solution, and (3) helping designers learn about relevant criteria and issues [5].

Sumner et al. [16] define the critiquing system's role in this context as shown in Figure 5.

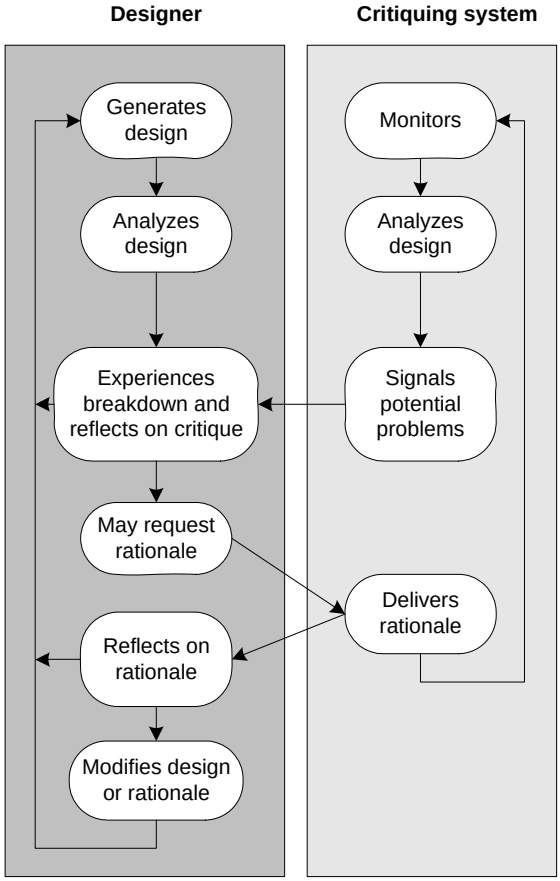

Figure 5. Role of critiquing in design, from [16]

Along these lines, Robbins [9] proposes a five-phase critiquing process model, called ADAIR for activate, detect, advise, improve and record. The model comprises the detect-advice component which is the core of all critiquing systems, but it allows for the evaluation of the relevance of the feedback to the designer's current task and support for guiding or making design improvements.

In the activate phase, a subset of available critics is selected for activation. In the detect phase, active critics detect assistance opportunities and generate advice. Then, in the advise phase, feedback is provided. This feedback improves the user's understanding of the status of the design, while the explanation provided improves their knowledge of the domain. In the improvement phase, the user resolves the identified problems or changes the

goals in reaction to an improved understanding of the problem. Finally, in the record phase, the resolution of each feedback item is recorded so that it may inform future decision-making.

7.2.1.1 Context shifts

Critics not only provide knowledge relevant to the task at hand, they can also support the user by actualizing the different dimensions of prior contexts. Indeed, while shifting to a new task, the user switches to a new mental state, thus losing information that becomes more difficult to recall.

Robbins [9] describes this situation in the design environment as follows.

Since designers cannot make all decisions simultaneously at the beginning of the design process, they are likely to use simplifying assumptions and place-holder elements that serve to structure the design and narrow the set of options considered. These initial decisions are tentative and must be considered later. Furthermore, as designers progress through the series of decisions, they come to understand more of the implications of the problem and the relevant features of the solution domain; this new understanding allows them to re-evaluate previous decisions. Mental context switches occur as the designer changes from one task to another. When starting a new step or revisiting a former one, designers must recall schemas and information needed for the task that were not kept in mind during the immediately preceding task. Inconsistencies can evolve undetected. Some requirements may be overlooked or forgotten as the designer focuses on more engaging ones.

The critic can help the user deal with such situations by delivering contextual information that helps the designer reconstruct prior mental states. At any point, the critic can assist designers by informing them of potential problems, possible corrective actions and pending decisions.

7.2.1.2 Ergonomics

A major challenge for critic designers is to create active design support systems that are cognitively ergonomic, i.e., systems that effectively support designers' cognitive work without hindering their creative process [16]. Such systems should positively influence both the design process and the design product.

A positive influence on the designer's behaviour could involve [16]:

- Modifying design solutions for the better
- Reflecting on issues raised by the critic

- Recording design rationale provided by the critic
- Taking into account criteria that were previously unknown
- Learning the information presented by the critic.

Conversely, negative behaviour can include: changing designs for the worse and/or being annoyed or disrupted by critic signals. As Sumner et al. [16] note: “Clearly, the two system goals of noticing potential problems and not being disruptive are diametrically opposed. These competing goals create a very difficult design situation which has been previously articulated as the challenge to ‘say the right thing at the right time’ ”.

Through their empirical investigations, the authors make several very interesting observations. They notice that during their study:

- All designers (highly skilled or not) anticipated critics prior to their firing. They looked for deficiencies in their solutions that might make the critics fire. Sometimes they even took preventive steps in order to avoid this. “Thus a key finding is that critics exert an indirect influence on designers’ cognitive processes that can lead to awareness, anticipation, and avoidance of system activity”.
- Critics do appear to promote reflection during design, however, the quality of reflection, particularly in the area of considering trade-offs, appears to be deeper for more experienced designers.
- Critics do influence designers to modify design products, though often the influence indirectly arises from critic anticipations.
- Only highly skilled designers were able to anticipate critics they had not been presented with during the current design session. Designers of low and medium skill anticipated only critics that had been previously presented.
- Redundant critics disturbed the highly experienced designers, who expected these rules to be locally disabled once they had explained why they had broken the rule and recorded their design rationale.

The authors note that an understanding of the designer’s traditional processes can be used to guide system design in order to create a system better adapted to the designer’s cognitive processes. They also note that a pre- and post-system approach is useful since the evaluation of complex systems is confounded by the fact that their use substantially changes the traditional task. As noted by Burkhart et al. [43], an analysis of the designer’s existing activities can serve as a baseline for assessing how the system has changed the task and whether the changes are for the better.

7.2.2 Visibility

Critiques are not only useful for enhancing the user's performance, but also for providing insight into the knowledge scope, resolution strategies and functionalities of the system.

For example, in AIDA (antibody identification assistant), a written checklist provides an explicit, high-level representation of the system's goal hierarchy. The design of the system is such that users can apply additional strategies without interference from the computer and can override a critique from the computer, but the checklist makes clear what steps the computer expects the user to perform before completing a case. The computer also allows the user flexibility in deciding what order to follow when completing the subgoals listed on the checklist (except when the sequence is critical to successful problem-solving) [10].

The checklist primarily reminds users of the steps necessary to solve a case correctly. But it also makes the high-level goal structure of the computer's knowledge base explicit for the user. This representation helps the user and the system establish a common frame of reference.

A similar concept can be found in the explanations literature under the name "strategic explanations" [44], which shed light on the system's higher-level control and planning information. These explanations clarify the system's problem-solving strategy, that is, they indicate why information is gathered in a certain order, why one knowledge piece is invoked before others and how reasoning steps contribute to high-level goals.

However, it has been shown [45] that these explanations, provided on the user's demand and aimed at explaining the system's results, are perceived as being given at the wrong level since they do not directly support the system's conclusions. With the critiquing approach, the system's role is precisely to check if the user's resolution steps comply with the general problem-solving approach. Thus, the critic's feedback in this case *is* strategic information. Yet, given that this information is provided as guidance for the problem-solving process, it is perceived as relevant and useful.

The importance of communicating the goal structure underlying the problem-solving task has also been underscored for the design of tutors within the framework of ACT-Theory [46]. With critics, the objective is more functional than instructional: the critic enables users to gain insight into the system's problem-solving approach rather than help them learn to solve a problem.

Critics provide insight into the system's resolution strategies and domain knowledge independently of the nature and presentation of the critique. The simplest error-checking mechanisms can give the user valuable information on the system's knowledge level. In Guerlain et al.'s study [37], behavioural data showed evidence that subjects using the critiquing system had the

potential to form a better mental model of the computer's level of competence, since they got feedback from the computer whenever it detected an error in their performance.

One of the significant obstacles encountered by users when using complex systems is that they are unaware of the kind of information contained in the system and the ways by which they can efficiently exploit it. A discrepancy exists "between the user's perception of an information space and the information actually contained in a high-functionality system" [6]. Most of the time, the problem persists because the effort required to find information outweighs the perceived benefits of doing so. Moreover, it is through extensive experience that users develop a grasp of the high-level functionalities of the system.

With critics, users can access this hidden information without knowing that either a certain type of knowledge or the tools that can retrieve it exist. "The critic is a knowledgeable agent that helps the users map their situation model into the available resources represented in terms of the system model" [22]. Through their interactions with the user, critics can reveal a lot about the system's knowledge and objectives, and this can be further enhanced through concurrent use of different intervention strategies.

7.3 Detecting and correcting errors and biases

Another way by which critics can assist users is by improving their judgement in decision-making situations. Since the late 1980s, Silverman has used a generic prototype shell (COPE) to explore whether cognitive bias in expert practitioners can be detected and corrected by the critiquing process. The research objective is to exploit a rich theoretical base of principles about the general weaknesses of human cognition in order to derive, test and validate a theory of expert error. The other complementary long-term research goal of his project is to develop a theory of criticism.

In [12], Silverman states that the design of critics must be guided by a theory of bugs and repairs and must result from the merger of expert critiquing with decision science techniques. He presents an error model that accounts for two major types of errors: (1) errors in the expert's knowledge base due to missing concepts or missing knowledge, and (2) errors in the expert's reasoning processes due to cognitive bias or systematic selection of poor judgement heuristics.

Before discussing this model further, it would be interesting to compare it with Reason's [47] classification of error types. Reason presents evidence for the existence of three basic error types: skill-based slips and lapses, rule-based mistakes and knowledge-based mistakes. Mistakes, rule-based or knowledge-based, are associated with problem-solving. Problem-solvers first proceed by automatic pattern-matching. If a pattern is recognized and some kind of match is performed, a previously established *if (condition) then (action)* rule-based solution is applied. Only when this procedure fails to provide an adequate solution will they move to the mode of making

inferences from knowledge-based mental models of the problem space. He argues that rule-based mistakes are either due to misapplication of good rules or to the application of bad rules. Knowledge-based mistakes have their roots in two aspects of human cognition: bounded rationality and the fact that knowledge relevant to the problem space is nearly always incomplete and often inaccurate. Therefore, in Reason's categorization, mistakes can be due to missing knowledge and/or reasoning flaws. It is the mental processes involved that distinguish the two types of mistakes.

Reason talks of several "pathologies" associated with knowledge-based problem–solving, which are selecting the wrong features of the problem space, being insensitive to the absence of relevant elements, confirmation bias, overconfidence, biased reviewing of plan construction, illusory correlation, halo effects and problems with causality, with complexity and with diagnosis in everyday life. Some of these issues can be directly related to the biases that Silverman tries to counter by means of a critiquing tool.

7.3.1 A model of human error

Silverman defines judgement as a person's ability to come to an understanding, opinion and sense of things. Distinct from domain knowledge, judgement relies on informal reasoning and includes the values, preferences and intuitions that set the stage for the decisions one ultimately makes. Humans routinely use judgement heuristics in order to save time and reduce complexity. This is acceptable as long as no errors occur. But recurring errors of judgement arise when the decision-maker systematically neglects more formal methods. When such tendencies are likely to be repeated, cognitive bias exists.

Unaware of their judgement biases, problem-solvers and planners tend to be overconfident in evaluating the correctness of their knowledge. They will tend to justify their chosen course of action by focusing on evidence that favours it and by disregarding contradictory signs. Also known as the confirmation bias, this strategy consists in seeking information consistent with current beliefs and avoiding contradictory evidence.

Silverman affirms that the human task practitioner focuses on two categories of knowledge; one is part of the normative and correct knowledge required to accomplish the task successfully (call it *B*), and the other is irrelevant knowledge not germane to the task because of bias in judgement (call it *A*). But the task practitioner often ignores two further categories of knowledge that are critical to successful task outcomes. These are *C* (overlooked knowledge) that the human ignores for reasons of biased judgement, and *D* (missing knowledge) that the practitioner is unaware of due to incomplete or out-of-date training. This is summarized in the upper part of Figure 6.

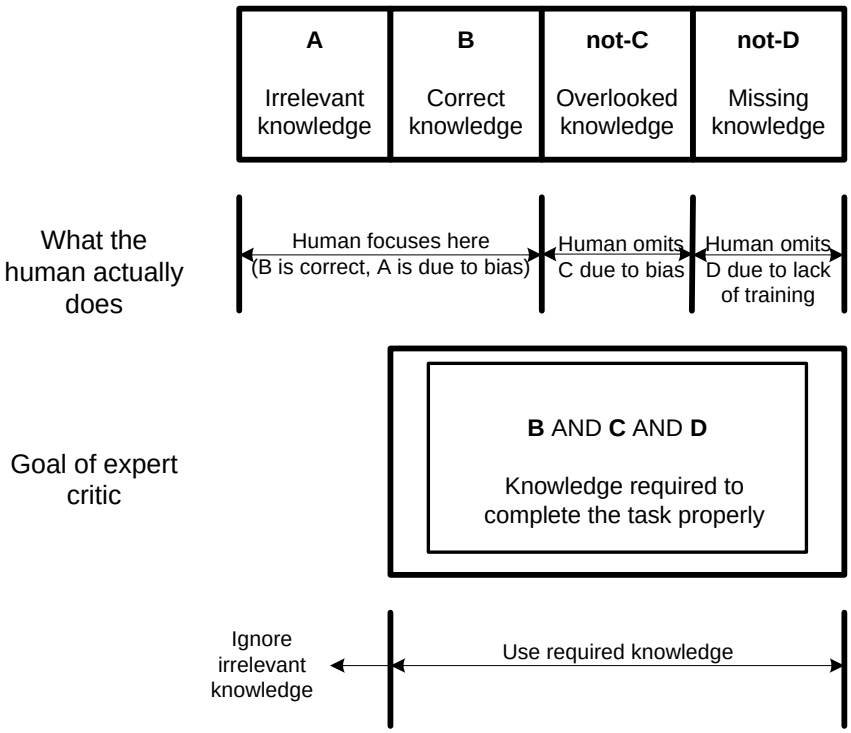

Figure 6. Human expert's knowledge and goal of critiquing, from [12]

Omitting *C* is a selective perception bias, which is very important. If one defines the problem incorrectly to begin with, no amount of processing or computation can compensate for the initial error.

According to Silverman, one way to assist decision-makers is to help them regress, by means of a critiquing system, to distributional data. The distinction between single-case and distributional information was first made by Kahneman and Tversky [48]. Single-case information, or case data, consists of evidence about the particular case under consideration. Distributional information, or base-rate data, consists of knowledge about the distribution of outcomes in similar situations. It has been observed that individuals are insufficiently sensitive to distributional data even when available.

Silverman gives examples of this in the domain of army forecasting: for army forecasters, past examples of degradation and damage from threats they encountered and studied in their own discipline are the most vivid and easy to recall. For example, attack helicopter specialists most vividly visualize potential damage from enemy attack helicopters and not the damage resulting from encounters with other threats.

Given the short timetable for completing the overall task and the shortage of available, concrete threat information, the forecaster replaces formal reasoning with weak, biased heuristics. Moreover, the forecasters experience no loss of confidence in their results due to this use of "common sense reasoning".

More generally, Silverman draws our attention to conditions that lead to biases of type A and not-C, such as:

1. Availability bias in the acquisition of distributional information

2. Base-rate biases that promote the attractiveness of the concrete information that forecasters possess in their personal experience base

3. Habit

4. Non-regressive effects

5. Adjustments from deficient anchors

6. Ease-of-recall biases.

In fact, trained professionals in a variety of disciplines exhibit mental myopia due to force of habit, selective perception, and availability bias [1]. For example, Silverman [49] shows that qualified statisticians fail to recognize everyday situations as classic probability and statistical problems due to a human tendency to disregard abstract data.

The goal of the critic, as shown in the lower part of Figure 6, is to help the human notice errors of judgement and bias in categories A and C. The goal of the judgement critic is to cause the human to divert his or her attention away from A and to refocus on C. The goal of the knowledge critic is to acquaint the human with missing knowledge. This helps to bring factor D into his inference processes. The former requires the critic to use a variety of strategies to influence and debias the human, which we present in the next section. The latter requires a critic to provide explanations or perform situated or on-the-spot tutoring.

Along these lines, Silverman recommends that judgement heuristics and substantive knowledge be treated separately. The principles of substantive knowledge of domain are of less interest to the critic than the heuristics the experts use to manipulate it [12].

7.3.2 Use of alternative strategies

While many systems (ATTENDING, ONCOCIN, CRITTER) use after-task criticism based on an intuitive argument that this improves user acceptability

or user performance, some empirical results show greater improvement from before-task influencing.

The use of both proactive and reactive methods of critiquing can be seen in [8], where the authors propose decision networks of critics as a new direction for critiquing systems. A decision network of critics is a cluster of critics that uses several influencing strategies. These critics are:

- *Influencers*, which provide tutoring before or during a task (in an incremental mode, with heuristic reasoning);
- *Debiasers*, which provide negative feedback when mistakes are made (self-activate incrementally);
- *Clarifiers*, which present feedback to designers in graphical or textual form; and
- *Directors*, which provide task-specific support for carrying out improvements (step-by-step procedures that walk a user through the process of using a cue).

These are related groups of agents that work together on a given major type of error and are activated by a dialogue generation component based on a task model.

Placed before the task, influencers make positive appeals to the user's intelligence. These are preventive critics that divulge common errors and the kind of decisions that are expected before the user begins the problem-solving process. In [12], Silverman argues that it is useful to have influencers warn experts about nonregressive and overconfidence biases and explain how to avoid them.

Biases, like perceptual illusions, are often hard to remove even after hearing about them. It is therefore necessary to have debiasers note that an erroneous judgement remains. Debiasers give the user negative criticism; they are more corrective strategies that are applied to the actions that have already been performed by the user and their suggestions are based on errors already committed. Debiasers conduct users towards a more correct reasoning strategy. They check for failures, explain their effects, explain their causes and then attempt repair actions.

Directors are task models that encourage the human to act as a simple data gatherer. Triggered when users choose their nonregressive solution, they transfer control from the user, who is left to record data in the system. The director requires the user to gather (abstract, distributional) data, conduct the steps of a regressive analysis, and compute the correct answer.

In [50], Silverman gives the following examples of strategies that can be used by these critics (if a cue fails, the next one is used).

Influencers:

- Leading question asking (e.g., "*Would you like to input the effectiveness, suitability or other issue?*")
- Socratic hinting (e.g., "*A good report normally contains one of each of the effectiveness and suitability issues but it may contain as few as one and as many as six issues if conditions require it*.")
- Tutoring (when hints fail; exposes the user to a hierarchy of notecards of interlinked principles where the user navigates as deeply as needed)
- Default reasoning (a default issue phrase that satisfies all cues and thus is a subtle attempt to anchor the user in good practice; e.g., "*Can the truck system sustain its logistics transport mission in harsh battlefield environments?*")
- Analogue reasoning (past relevant cases)

Debiasers:

- Repetition (the same criticism is restated differently to help reach distracted or differently oriented users)
- Persuasion (usually takes the form of explaining the cause of error and its effect if it goes unchanged)
- Argumentation (explaining and presenting alternatives, only mentioned in [12])
- Feedback: helps the user bring their results more in line with the prescribed cues.

In a study on the use of critiquing systems for probability problems [51], Silverman noticed that when only *debiasers* (critics that provide feedback only after they find that the user's conclusion was incorrect) were used, performance improved significantly (69% correct answers after the third critique compared with 4% for the control group), but that when *influencers* were also used (critics that give before-task explanations), performance improved to 100% correct answers after the third critique. Silverman proposes the use of both of these critics as well as director strategies for efficient error and bias correction.

Silverman [50] proposes a methodology for evaluating and refining critics which focuses on the measurable aspects of how critics help users minimize their knowledge and judgement errors. He observes that as more evaluation methodologies are applied, critic designs begin to focus on (1) various degrees of coupling knowledge versus judgement critics as a promising avenue that needs to be further explored and evaluated, particularly in the presence of non-thinking reactions by experts; (2) the prospect of tailoring the critic to various characteristics of the human partner by integrating limited degrees of user modelling into the critic's architecture; and (3) use of critics that assist the expert before he or she psychologically commits to an erroneous solution (e.g., by joint use of before-task and after-task critics, incrementally triggered persuaders and feedback agents).

8. Conclusion

This report provided a broad view of the use of critiquing systems as decision support tools. Critics can make the decision-maker's task less challenging cognitively with no loss of decision control, and avoid costly consequences by alerting him or her to problematic issues. As a side effect, critics also promote learning and help users improve their understanding of the system's knowledge and goals.

However, the critiquing paradigm has its own pitfalls. Users might find it very annoying to have a critic constantly watching for their errors and may feel the need to guard against criticism. One of the studies presented here reported cases of critic rejection, which may be due either to reluctance to accept advice or to disagreement with the content of the advice. More insight must be gained into such issues through empirical studies. One thing has been established, however: it is not enough to point out differences; the critic must provide support with solution work-up. The user's perception of critics, as we saw, can also be enhanced through the use of explanatory/argumentative discourse, tailored critiquing strategies and ergonomic human-computer interfaces.

The other challenge for critic design is the obtrusiveness of the system. A critic must intervene at the right time without hampering the user's creativity or interfering with his or her thought process ("say the right thing at the right time"). Evaluation and intervention strategies must be designed to challenge the user without being too intrusive. Ideally, users will be allowed to control the critic's intervention strategy.

The next step is to specify what type of critiquing parameters and strategies work best in specific domains and for specific decision-makers. One can also choose to look at the problem on a more general level first; that is, try to determine the scope of the critiquing approach with regard to problem-solving approaches and task structures used across application domains, in order to define a general framework for the use of critics.

Reading the literature on critiquing systems reveals why critiquing systems are suitable for certain application domains; however, it does not indicate in what respects these domains are similar or what characterizes them, other than fuzziness. Our explanation is that critiquing systems are appropriate for constraint-satisfaction problem types. Yet, for any potential application domain, the CoA estimate process in this case, one should be able to characterize it in terms of domain knowledge, task structure, problem-solving methods, reasoning processes (inductive, deductive) and even thought cycles (e.g., How can the insight provided by the critic be incorporated into the new version of the user's work? What effect would contextualized access to knowledge have in the decision situations characterizing that domain?), not to mention external factors (What would a goal-oriented collaborative human-critic interaction look like in a time-sensitive environment?).

One can also choose to base the use of the critic on an error model of the domain. As we saw in Silverman's work, a long-term research goal can consist in exploring how domain-independent theories of systematic errors of judgement can be used when designing critics. For Silverman [12], critic engineers must use psychological and decision science models of error processes as a basis for different design factors.

Cue usage, and more generally, performance improvement through the use of critiquing feedback, will have to be evaluated by means of a lens model or other measurement tool. The sensitivity of users to the user-system configuration in the critiquing approach, as opposed to traditional expert systems, is also a factor on which the user's verdict is needed. Within the critiquing paradigm, it is the role of the critiquing system to bring relevant knowledge to the user's attention; it is the role of the user to evaluate the trade-offs and make the final decisions. Can we take it for granted that this is what the user wants? If the answer is affirmative, then one can draw important conclusions for the design of the interactive features of future advisory systems.

Finally, even after one designs and inserts critics into a given task environment, one should closely monitor the reactions of users to the various critiquing parameters and strategies in order to improve them further. However, as discussed before, there is a risk that the system will influence or change the way the user performs the task.

This concludes our overview of the potential benefits of critiquing systems and the main challenges that await critic designers.

9. References

1. Silverman, B.G. (1992). Survey of Expert Critiquing Systems: Practical and Theoretical Frontiers. *Communications of the ACM*, 35(4), 107-127.

2. Van Eemeren, F.H. and Grootendorst, R. (1984). *Speech Acts in Argumentative Discussions*. Dordrecht and Cinnaminson: Foris.

3. Walton, D. (1996). *Argument Structure: A Pragmatic Theory*. University of Toronto Press.

4. Freeley, A.J. and Steinberg, D.L. (2000). *Argumentation and Debate. Critical Thinking for Reasoned Decision Making*. Belmont: Wadsworth.

5. Fischer, G., Nakakoji, K., Ostwald J., Stahl, G., and Sumner T. (1993). Embedding Critics in Design Environments. *The Knowledge Engineering Review Journal, Special Issue on Expert Critiquing*, 8(4), 285-307.

6. Fischer, G., Lemke, A.C., Mastaglio, T. and Morch A.I. (1991). The Role of Critiquing in Cooperative Problem Solving. *ACM Transactions on Information Systems*, 9(2), 123-151.

7. Miller, P. (1986). *Expert Critiquing Systems: Practice-based Medical Consultation by Computer.* New York: Springer Verlag.

8. Silverman, B.G. and Mezher, T.M. (1992). Expert Critics in Engineering Design: Lessons Learned and Research Needs. *AI Magazine*, AAAI 13 (1), 45-62.

9. Robbins, J.E. (2003). Design Critiquing Systems. http://www.ics.uci.edu/~jrobbins/papers/CritiquingSurvey.pdf

10. Guerlain, S.A., Smith, P.J., Obradovich, J.H., Rudmann, S., Strohm, P., Smith, J.W., Svirbely, J., Sachs, L. (1999). Interactive Critiquing as a Form of Decision Support. *Human Factors*, 41(1), 72-89.

11. Ramachandran, S. and Wilkins, D.C. (1997). Temporal Control Structures in Expert Critiquing Systems. *Proceedings of the 4th International Workshop on Temporal Representation and Reasoning*. Daytona Beach, FL, USA. 160-167.

12. Silverman, B.G. (1992). Judgement Error and Expert Critics in Forecasting Tasks. *Decision Sciences*, 23(5), 1199-1219.

13. Mezher, T. Abdul-Malak, M.A., Maarouf, B. (1998). Embedding Critics in Decision-making Environments to Reduce Human Errors. *Knowledge-based Systems*, 11, 229-237.

14. Rhein-Desel, U. and Puppe, F. (1998). Concepts for a Diagnostic Critiquing System in Vague Domains. *Advances in Artificial Intelligence - Proceedings of the 22nd Annual German Conference on Artificial Intelligence*. Bremen, Germany. 201-212.

15. Shepherd, A. and Ortolano, L. (1994). Critiquing expert systems for planning and management. *Computers, Environment and Urban Systems*, 18, 305-314.

16. Sumner, T., Bonnardel, N., Kallak B.H. (1997). The Cognitive Ergonomics of Knowledge-based Design Support Systems. *Proceedings of the 1997 Conference on Human Factors in Computing Systems* - CHI. Atlanta, GA, USA. 83-90.

17. Simon, H. and Associates (1986). Report of the Research Briefing Panel on Decision Making and Problem Solving, by the National Academy of Sciences. Washington: National Academy Press.

18. Kuilboer, M.M., Van Der Lei, J., De Jongste, J., Overbeek, S.E., Ponsioen, B., Van Bemmel, J.H. (1998). Simulating an Integrated Critiquing System. *Journal of the American Medical Informatics Association*, 5(2), 194-202.

19. Guerlain, S., Smith, P.J., Obradovich, J., Smith, J.W., Rudmann, S. and Strohm, P. (1995). The Antibody Identification Assistant (AIDA), an example of a cooperative computer support system. *Proceedings of the 1995 IEEE International Conference on Systems, Man, and Cybernetics*. New York: International Institute of Electrical and Electronics Engineers. 1909-1914.

20. Sage, A.P. (1987). Human Information Processing Principles for Command and Control. In: J.L. Boyes and S.J. Andriole (eds.) *Principles of Command and Control*. Washington, DC: AFCEA International Press.

21. Lemke, A.C. and Fischer, G. (1990). A cooperative problem solving system for user interface design. *Proceedings of AAAI-90*, 8th National Conference on Artificial Intelligence. Cambridge, MA, 479-484.

22. Fischer, G. and Mastaglio, T. (1991). A Conceptual Framework for Knowledge-based Critic Systems. *Decision Support Systems*, 7, 355-378.

23. Vahidov, R. and Elrod, R. (1999). Incorporating Critique and Argumentation in DSS. *Decision Support Systems*, 26(3), 249-258.

24. Mastaglio, T. (1990). User Modelling in Computer-based Critics. *Proceedings of the 23rd Annual Hawaii International Conference on System Sciences*. Kailua-Kona, HI, USA. 403-412.

25. McTear, M.F. (1993). User Modelling for Adaptive Computer Systems: A Survey of Recent Developments. *Artificial Intelligence Review*, 7, 157-184.

26. Nakakoji, K., Sumner, T. and Harstad, B. (1994). Perspective-based Critiquing: Helping Designers Cope with Conflicts Among Design Intentions. *Proceedings of the 3rd Conference on Artificial Intelligence in Design*. Lausanne, Switzerland. 449-466.

27. Rothschild, M.A., Swett, H.A., Fisher, P.R., Weltin, G.G. and Miller, P.L. (1990). Exploring Subjective vs. Objective Issues in the Validation of Computer-based Critiquing Advice. *Computer Methods and Programs in Biomedicine*, 31, 11-18.

28. Irandoust, H., Moulin, B., Bélanger, M. (2001). De l'usage de l'explication et de l'argumentation dans les systèmes à base de connaissances. *In Cognito*, 23, 3-22.

29. Langlotz, C.P. and Shortliffe, E.H. (1983). Adapting a Consultation System to Critique User Plans. *International Journal of Man-Machine Studies*, 19(5), 479-496.

30. Mao, J.-Y. and Benbasat, I. (2001). The effects of contextualized access to knowledge on judgement. *International Journal of Human-Computer Studies*, 55, 787-814.

31. Woods, D.D., Roth, E.M. and Bennett, K. (1990). Exploration in Joint Human-machine Cognitive System. In Robertson, S., Zachary, W., Black, J.B. (eds.) *Cognition, computing and cooperation*. Norwood: Ablex, 123-158.

32. Pollack, M.E., Hirschberg, J. and Webber, B.L. (1982). User participation in the reasoning processes of expert systems. *Proceedings of the 2nd National Conference on Artificial Intelligence*. Pittsburgh, Pennsylvania. 358-361.

33. Klein, G.A. (1995). A Recognition-Primed Decision (RPD) Model of Rapid Decision Making. In: G.A.Klein, J. Orasanu, R. Calderwood and C.E. Zsambok (eds.) *Decision Making in Action: Models and Methods*. Norwood: Ablex Publishing Corporation.

34. Muir, B.M. (1987). Trust between Humans and Machines, and the Design of Decision Aids. *International Journal of Man-Machine Studies*, 27(5), 527-539.

35. Endsley, M.R. and Kaber, D.B. (1999). Level of automation effects on performance, situation awareness and workload in a dynamic control task. *Ergonomics*, 42, 462-492.

36. Parasuraman, R., Mouloua, M. and Molloy, R. (1994). Monitoring Automation Failures in Human-machine Systems. In: M. Mouloua and R. Parasuraman (eds.) *Human Performance in Automated Systems: Current Research and Trends.* Mahwah, NJ: Erlbaum, 45-49.

37. Guerlain, S.A.E. (1993). Factors Influencing the Cooperative Problem-solving of People and Computers. *Proceedings of the Human Factors and Ergonomics Society*, 37th Annual Meeting. Seattle, WA, USA. 387-391.

38. Klaczynski, P.A., Gordon, D.H. and Fauth, J. (1997). Goal-oriented critical reasoning and individual reasoning biases. *Journal of Educational Psychology*, 89, 470-485.

39. Woods, D.D. (1986) Cognitive Technologies: The Design of Joint Human-Machine Cognitive Systems. *AI Magazine*, 6(4), 86-92.

40. Schwartz, D.G. and Te'eni, D. (2000). Tying knowledge to action with kMail. *IEEE Intelligent Systems and Their Applications*, 15, 33-39.

41. Bonnardel, N. (1992). *Le Rôle de l'Évaluation dans les Activités de Conception* [The Role of Evaluation in Design Activities], University of Provence, Ph.D. Dissertation, Department of Psychology.

42. Schoen, D.A. (1983). *The Reflective Practitioner: How Professionals Think in Action*. New York: Basic Books.

43. Burkhart, B., Hemphill, D. and Jones, S. (1994). The Value of a Base-line in Determining Design Success. *Proceedings of CHI'94* - Human Factors in Computing Systems. Boston, MA, 386-391.

44. Chandrasekaran, B., Tanner, M.C., Josephson, J.R. (1989). Explaining Control Strategies in Problem Solving. *IEEE Expert*, 4(1), 9-24.

45. Ye, L.R. and Johnson, P.E. (1995). The Impact of Explanation Facilities on User Acceptance of Expert System Advice. *MIS Quarterly*: *Management Information Systems*, June, 157-172.

46. Anderson, J.R., Corbett, A.T., Koedinger, K.R. and Pelletier, R. (1995). Cognitive tutors: Lessons learned. *Journal of Learning Sciences*, 4, 167-207.

47. Reason, J. (1990). *Human Error*. Cambridge: University Press.

48. Kahneman, D. and Tversky, A. (1973). On the Psychology of Prediction. *Psychology Review*, 80, 237-351.

49. Silverman, B.G. (1992). *Critiquing Human Error: A Knowledge Based Human-Computer Collaboration Approach*. London: Academic Press.

50. Silverman, B.G. (1992). Evaluating and Refining Expert Critiquing Systems: A Methodology. *Decision Sciences*, 23(1), 86-110.

51. Silverman, B.G. (1992). Building a Better Critic. Recent Empirical Results. *IEEE Expert*, 18-25.

Distribution list

INTERNAL DISTRIBUTION

DRDC Valcartier

1 - Director General
1 - Chief Scientist
3 - Document Library
1 - Head/DSS
1 - Head/IKM
1 - Head/SOS
1 - H. Irandoust (author)
1 - R. Breton
1 - J.M.J. Roy
1 - S. Paradis
1 - M. Bélanger
1 - A. Guitouni
1 - J. Berger
1 - A. Benaskeur
1 - A.C. Boury-Brisset
1 - P. Maupin
1 - F. Rhéaume
1 - A.L. Jousselme
1 - R. Rousseau
1 - LCdr E. Woodliffe
1 - Maj G. Clairoux
1 - Maj M. Gareau
1 - Maj B. Deschênes
1 - MCpl É. Tremblay

EXTERNAL DISTRIBUTION

DRDC Valcartier TR 2003

1 - Director Research and Development Knowledge and Information Management

1 - Director Research and Development Knowledge and Information Management (unbound copy)

1 - Defence Research and Development Canada

3 - Director General Operational Research

1 - 3d Thrust Leader: Chris McMillan, DRDC-Ottawa

4 - Director General Joint Force Development

1 - ST CCIS

2 - Director Science and Technology (Command and Control Information Systems)

2 - Director Science and Technology (Air)

2 - Director Science and Technology (Land)

2 - Director Science and Technology (Maritime)

2 - Director Science and Technology Human Performance

2 - Director Maritime Requirements Sea

1 - Director Maritime Requirements Sea 4

1 - Director Maritime Requirements Sea 6

1 - Director Maritime Requirements Sea 6-2

2 - D DCEI

1 - D LCSPM

2 - Director Air Requirements

1 - Director Air Requirements 4

1 - Director Air Requirements 3

2 - Director Maritime Ship Support

2 - Director Maritime Ship Support - 6

2 - Director Maritime Ship Support - 8

1 - Director Science and Technology Air – 3 (3d)

1 - Director Science and Technology Maritime – 2 (1b)

EXTERNAL DISTRIBUTION (cont'd)

DRDC Valcartier TR 2003

1 - Director Science and Technology Maritime – 3 (1a)

1 - Director Science and Technology Maritime – 5 (1c)

1 - Director Science and Technology Land – 2

2 - Director Land Requirements

1 - ADM (IM)

1 - Canadian Forces Experimentation Centre

8 - Defence Research and Development Canada - Toronto:

Attn: R. Pigeau

C. McCann

S. McFadden

J. Baranski

D. Bryant

A.R. Blais

LCdr K. Kubeck

K. Hendy

Maj Linda Bossi

4 - Defence Research and Development Canada - Atlantic:

Attn: J.S. Kennedy

B. McArthur

R. Kessel

M. Hazen

LCdr B. MacLennan

2 - Defence Research and Development Canada – Ottawa:

Attn: B. Bridgewater

1 - PMO MHP

1 - PMO HMCCS

1 - PMO CADRE

1 - PMO AURORA

1 - Canadian Forces Command and Staff College
Toronto
Attn: Commanding Officer

1 - Canadian Forces Maritime Warfare School
CFB Halifax
Halifax, Nova Scotia
Attn: Commanding Officer

DOCUMENT CONTROL DATA

1. ORIGINATOR (name and address)	2. SECURITY CLASSIFICATION (Including special warning terms if applicable)
Defence R&D Canada Valcartier 2459 Pie-XI BLvd. NOrth Val-Bélair, QC G3J 1X8	Unclassified

3. TITLE (Its classification should be indicated by the appropriate abbreviation (S, C, R or U)

Critiquing systems for decision support

4. AUTHORS (Last name, first name, middle initial. If military, show rank, e.g. Doe, Maj. John E.)

Irandoust, H.

5. DATE OF PUBLICATION (month and year)	6a. NO. OF PAGES	6b .NO. OF REFERENCES
2006	53	51

7. DESCRIPTIVE NOTES (the category of the document, e.g. technical report, technical note or memorandum. Give the inclusive dates when a specific reporting period is covered.)

Technical report

8. SPONSORING ACTIVITY (name and address)

9a. PROJECT OR GRANT NO. (Please specify whether project or grant)	9b. CONTRACT NO.
13DM22	

10a. ORIGINATOR'S DOCUMENT NUMBER	10b. OTHER DOCUMENT NOS
TR 2003-321	N/A

11. DOCUMENT AVAILABILITY (any limitations on further dissemination of the document, other than those imposed by security classification)

- ☒ Unlimited distribution
- ☐ Restricted to contractors in approved countries (specify)
- ☐ Restricted to Canadian contractors (with need-to-know)
- ☐ Restricted to Government (with need-to-know)
- ☐ Restricted to Defense departments
- ☐ Others

12. DOCUMENT ANNOUNCEMENT (any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in 11) is possible, a wider announcement audience may be selected.)

13. ABSTRACT (a brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual).

The purpose of this document is to provide a general introduction to critiquing systems and to discuss their particularities as decision support tools. Critiquing systems (critics) are software programs that take the problem description and the user's partial or final result as inputs and provide a critique of the user-proposed solution as output. This specific type of human-computer interaction, and the fact that critics can operate with limited knowledge, makes these systems very appropriate for open-ended domains where it is impractical to embed complete knowledge or where users are unwilling to cede decision-making.

This report provides definitions of critiquing and related concepts; describes the general mechanisms of critiquing systems; examines the adequacy of the critiquing approach with regard to the nature of user tasks and application domains; presents the different intervention strategies used by software critics as well as their user perception; discusses the user-adaptation of critiques and the joint use of critiques and explanation/argumentation; and finally identifies the distinctive features of critiquing systems in terms of cognitive support, which comprise transfer of decision control to the user, collaborative problem-solving, and detection and correction of errors and biases.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus, e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus-identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

Critiquing systems, critique, decision support, collaborative problem-solving, reasoning biases

Defence R&D Canada

Canada's leader in defence
and national security R&D

R & D pour la défense Canada

Chef de file au Canada en R & D
pour la défense et la sécurité nationale

WWW.drdc-rddc.gc.ca