Artificial Intelligence for Engineering Design, Analysis and Manufacturing (2002), *16*, 229–241. Printed in the USA.

DOI: 10.1017/S0890060402163086

Sharing design perspectives through storytelling

ANA CRISTINA BICHARRA GARCIA,[1] CARLOS EDUARDO CARRETTI,[1]
INHAÚMA NEVES FERRAZ,[1] AND CRISTIANA BENTES[2]

[1]Universidade Federal Fluminense, 22421-020 Rio de Janeiro, RJ, Brazil
[2]Universidade do Estado do Rio de Janeiro, Rio de Janeiro, RJ, Brazil

(RECEIVED October 20, 2001; ACCEPTED April 3, 2002)

Abstract

Design consists of analyzing scenarios and proposing artifacts, obeying the initial set of requirements that lead from initial to goal state. Finding or creating alternative solutions, analyzing them, and selecting the best one are expected steps in the designer's decision making process. Very often, not a sole designer, but a team of them is engaged in the design process, sharing their expertise and responsibility to achieve optimum projects. In a design team, most conflicts occur due to misunderstanding of one's assessment of specifications and contexts. Decision explanations play a key role in teamwork success. Designers are rational agents trained to follow rational methods. Acceptable justifications include value function, requirements, constraints, and criteria. Generally, explanations are delivered in a multimedia fashion, composed of text, graphics and gestures, to provide the audience the ability to perceive what was contextually imagined. The more spatial the reasoning is, the richer the explanation channel should be. This paper presents CineADD, a design explanation generation model based on cinema techniques such as animation, scripting, editing, and camera movements. The idea is to provide designers with a tool for describing the way their projects should be visually explained, as in a movie. Designers develop their projects in an active design document environment. Rationale is captured as a design model, so explanations can be generated instead of retrieved. The captured design model serves as a base to visually reconstruct design, giving emphasis and guidance by using movie storytelling techniques. CineADD was implemented for the domain of oil pipeline layout showing the feasibility of this approach. We expect CineADD to become a commodity attachable to any intelligent CAD system.

Keywords: Design; Cinema; Teamwork; Explanation; Scripting

1. INTRODUCTION

Human–computer interaction (HCI) consists of a dialog between users with a set of demands and computer systems with a set of affordances built into their codes by their designers. The interaction happens physically through input and output devices, such as a keyboard and printer. It also happens through information exchange that allows the emergence of cognitive distinction between players. Depending on the complexity and the way messages are delivered, it may become a challenge for users to understand them.

The communication involves the speaker, the listener, the channel, the content to be transmitted, the code used to make the content of a message, and the message itself. Texts, graphics, and pictures are the common codes employed by the computer to deliver the message. However, sometimes HCI demands an "immersive" experience (Lachman, 1997), as in movies, for users to efficiently perceive the overall, but sometimes hidden, information. A movie has the power to connect spatial and temporal information, make concrete one's perspective of the facts and processes, reconstruct human memory, and make the audience think.

Knowledge-based systems (KBSs) have been successfully used in CAD systems to assist users in developing design projects, either by offering design solutions or by verifying decision alternative solutions (ten Hagen & Tomiyama, 1987; Garcia et al., 1997). KBSs contribute to users finding efficient solutions, given a design context. Users' acceptance strongly depends on the credibility of computer suggestions. An active design document (ADD, Garcia, 1992) is an environment for developing engineering designs assisted by a computational agent trained for making decisions on

Reprint requests to: A.C.B. Garcia, Universidade Federal Fluminense, Rua Nascimento e Silva 538/501, 22421-020 Rio de Janeiro, RJ, Brazil. E-mail: bicharra@dcc.ic.uff.br

projects in a specific design domain. ADD allows users to develop their project while being monitored by its design agent.

Whereas the agent's knowledge base covers user decisions, explanation of those decisions can be derived without the user's guidance. Whenever a user's decision on a design project conflicts with the ADD's expectation, the computational agent will interact with the user to gather more knowledge to improve its knowledge base. Providing clear explanation is the key to this teamwork of user and computer agent. Furthermore, because a project is generally developed in teams, the availability of explanations for design decisions allows an understanding of individual perspectives on design issues.

Explanations vary from canned text (optionally, a multimedia message), working as prerecorded annotations, to on-demand generated explanations. Although a textual explanation is fundamental, there are domains in which spatial and temporal reasoning are crucial to decision making. For such domains, explanations composed only by text, diagrams, and pictures will not serve because the spatial and temporal transformation will not emerge. For instance, planning a kitchen layout in a 2- or 3-dimensional (2-D or 3-D) space consists of optimizing space distribution by obeying a set of norms, such as the fact that the refrigerator should not be placed beside the oven. The designer's task consists of moving, erasing, and reshaping objects (Fischer et al., 1991). The decisions in this domain are not well reported by either textual notes or figures. They need to be reported using actions. An event in time makes a difference in the possible understanding of the facts. When explanations reflect a set of actions or a process in a time frame, a sequence of scenes may be transformed into an animation, leading to a reconstruction (full or partial) of what happened. The introduction of another visual medium (animated scenes) brings up issues related to animation speed, scene selection, and user's attention guidance. Creating a scene from a system's interaction log is a matter of using computer graphics techniques such as rendering. The issues discussed in this paper concern building interactive narratives as the explanations for artifact designs. In this context, designers play the role of a movie director choosing the right framing to communicate their idea when creating an artifact. In addition to allowing designers play the director's role, it is important to let end users investigate the explanation from different perspectives in order to understand it.

In this paper we present CineADD, a design explanation model based on cinematic techniques. CineADD was planned for any CAD system; however, we develop our studies using ADD applied to engineering spatial layout domains. Our goal is to show the feasibility of using cinema and animation techniques to generate visual animated explanations that augment the end users' understanding of the designer's decisions and intentions. This visual presentation represents the design story, that is, the designers' perspective of their project. By adding special effects to the design story, designers can emphasize or hide details of the scenario. Our goal is to allow designers to create a script that describes the way they want their design to be explained to others. Designers work as filmmakers, creating a script for their design movie. In addition, the other participants may change the script to further investigate the design.

In our research, we investigated the use of cinematic techniques to empower user interface systems, allowing a greater volume of knowledge to be concisely conveyed to end users. The encoding mechanisms, which allow images and their interaction to carry meaning, must allow designers (filmmakers) to express their intentions and end users (audience) to perceive them.

This paper is organized as follows. In Section 2 we discuss the design task as an activity normally conducted by a group of people. Section 3 describes the ADD approach to design development and documentation. Section 4 presents some background on cinema and animation techniques. Section 5 provides a description of our model, called CineADD, to apply cinema techniques to the design process. Section 6 presents a case study of the CineADD model in the context of oil pipeline design in the ADDSub system. Finally, Section 7 provides our conclusions of this work.

2. DESIGN TASK AS GROUP ACTIVITY

An artifact generally emerges as a solution to a set of needs from a group of people willing to pay for it. Before the idea becomes concrete, a great deal of work must be done that usually involves people with different expertise. Although they are neither the longest nor the most expensive, the conceptual and preliminary design phases are crucial because the solution is conceived during this period. Mistakes can still be found and fixed at a low cost when compared with the remaining phases, such as the detailing and construction phases.

Design is a complex activity normally performed by a group of people with different types of expertise, who either work together to reach a good solution or work sequentially to carry out the project from the conceptual to the detailing design phases.

From the initial specifications, designers elaborate a project concept that is discussed by the design team and the end user. During the development of the project, the set of specifications grows and gets modified as a result of a deeper understanding of the problem being addressed by the project. Everybody in the design team shares responsibilities; however, those who sign the project get the fame or the blame for the success or failure of the artifact.

Design documents are produced to communicate a specification of the solution that designers intended for further construction. Frequently, due to the complexity of the artifact, there is more than one writer in the artifact's story. As expected, conflicts among them appear. In this context, the documentation is also used as a communication medium to allow mutual understanding.

https://doi.org/10.1017/S0890060402163086 Published online by Cambridge University Press

From the documentation, issues may be raised, leading to a group discussion on possible better overall solutions. In summary, there are two main types of users: the documentation builder or writer and the documentation consumer or reader.

The builder or writer is the design team. They are people with the same or different backgrounds who are hired to contribute their specific expertise but who are also committed to the final integrated solution. It is not rare that a designer must compromise the quality of his or her decision on a portion of a project to allow the best-integrated solution to be used. However, giving up a partial solution often requires much convincing. Consider the following scenario involving plumbing and air conditioning design. There is a great deal of overlap between the two designs, but both must be integrated in a house. Sometimes a designer has to decide on a less than optimal alternative solution to contemplate constraints of other areas of the project (local vs. global analysis). When writing the documentation, the designer writes the final solution. Unfortunately, the discussion is put aside. Consequently, when accessing the final documentation many alternative solutions that have already been discussed arise again, and designers have to rebuild the rationale for the final solution.

The consumer or reader consists of people interested in reading design documentation for many different reasons, such as:

- *Accept or reject a solution*: This scenario lets writers gain supporters to their point of view.
- *Approve or reject a solution*: This scenario lets writers share their responsibility with readers.
- *Build the artifact*: This scenario concerns making the specification concrete.
- *Understand a solution*: This scenario is the basis for all other scenarios (accepting, approving, or building the artifact).

Even though good documentation would save time, it has been neglected due to emphasis on generating a good quality solution. The importance of design rationale has been acknowledged, and research effort has been devoted to its capture and retrieval (Moran & Carroll, 1996).

Although design rationale capture and delivery are highly connected, this paper focuses only on design rationale retrieval. When using an intelligent CAD system environment, specifically an ADD environment, designers can gradually build design and documentation for further investigation. Design rationale capture is a subproduct of developing design. The issues discussed in this paper concern delivering an effective message that reveals the designers' perspectives when building their solution.

3. ADD

The ADD (Garcia, 1992) has been used as an intelligent CAD system, helping designers develop and document their projects (Vivacqua & Garcia, 1996; Garcia et al., 1997). The ADD approach uses the apprentice metaphor.

A computational agent, capable of developing a design in a specific domain, monitors a designer developing a project using the ADD environment. The computational agent creates expectations for design decisions, based on its knowledge base. Whenever an expectation fails, the agent interacts with users, presenting its rationale for its expectation. Based on the presented explanation, users may have a clue concerning the ADD knowledge representation for inputting modifications. Knowledge acquisition is restricted to scenes within a context. There is no commitment to an integrated knowledge base. The final knowledge base should cover the project developed by the design team and ADD. Generating explanations for design decisions is an important but easy task for the computational agent, considering its ability to generate a design decision expectation.

To work as an apprentice, ADD must start with an initial domain model that guides its decision making throughout the decision process. This initial model is implemented by a knowledge engineer, and it represents all necessary abstractions on the process and parameters of a given domain. The created model, however, does not always produce the same decisions that human designers do. This happens for a number of reasons, such as the system model does not cover all possible situations or the designer experience may exceed the knowledge captured in the ADD domain model. In any of these situations, designers may solve conflicts and differences by changing either the system's underlying model or their own mental model. This shows that ADD is actually a learning environment for the system and users. Figure 1 shows the ADD architecture with its seven components.

Anticipator is the inference engine of the apprentice agent. It monitors the design project for focus alteration. A design project is represented as a set of parameters with their val-

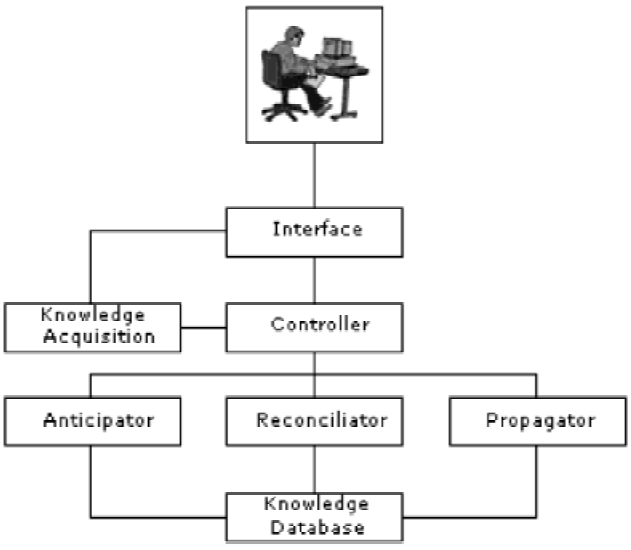

Fig. 1. The ADD model architecture.

https://doi.org/10.1017/S0890060402163086 Published online by Cambridge University Press

ues, available in a blackboard structure. Whenever a parameter gets the user's focus through the design interface, the anticipator triggers its knowledge base to create a valid expectation for that parameter value. In addition to a parameter value, the decision may involve the existence of the parameter. For example, suppose the task is designing a house. Although the owner may provide a set of specifications for a porch, later on we may conclude that there is no space for even thinking about one.

Reconciliator identifies conflicts between the system and users' decisions. Defining the similarity function is the main issue for this module.

Knowledge acquisition elicitor elicits from the designer changes to the initial design model. A mismatch diagnosed by the reconciliator triggers the action of the knowledge elicitor component. Before letting users change its knowledge base, it shows an explanation of how its expectation was reached. An explanation is composed of

- the sequence of the decisions already made,
- the dependency graph showing what parameters influence the current decision, and
- a trade-off table showing the alternative values and the performance of each alternative according to the set of pertinent constraints and criteria.

After presenting its explanation, the elicitor lets users include or exclude parameters, parameter dependencies, alternative parameter values, rules to produce new parameter values, rules to evaluate parameters, constraints, criteria, and evaluation functions.

After receiving the changes, the reconciliator checks if the changes are sufficient to erase the mismatch. The elicitation process continues till the reconciliator gets satisfied or the user wants to force a value with no explanation.

Propagator propagates the effects of design decisions. Whenever a decision changes, other changes may be requested to comply with the new scenario. The propagator stops when it reaches an untouched design space area or when it reaches a design decision with a value imposed by the designer that does not comply with the knowledge base (a break in the domain knowledge consistency).

Controller monitors the project blackboard and determines which module should be triggered.

Domain knowledge base contains the heuristics ruling the decision process in the domain. A dependency parametric network represents the domain knowledge. Primitive parameters are the input data. Derived parameters have a formula, either heuristic or mathematic, for determining their values. Decided parameters require a trade-off analysis, so alternative values must be generated. Constraints are applied to eliminate the unfeasible alternatives, whereas criteria are applied to order them. As a rational agent, the best alternative is preferred and selected. Sometimes, dependencies are dynamically assigned, increasing the complexity of the network processing; the same happens with parameters with mutual dependencies (i.e., finding the value of one parameter influences finding the value of the other and vice versa).

Design user interface consists of the CAD interface from which designers will develop their project.

This modular architecture allows ADD to capture all information needed to recover the design history, as well as the information needed to justify the decisions underlying the final product or artifact.

4. FRAMING CINEMA TECHNIQUES TO USE IN COMPUTER INTERFACE DESIGN

Cinema is an attractive medium for transferring thoughts. Using a set of techniques, filmmakers build a narrative to deliver a message that is communicated to an audience through a movie. Individual understanding requires a balance between the audience's and the filmmaker's ways of seeing the world. Similarly, computer interfaces have a message from the designer that must be understood and negotiated (Persson, 1999, 2001). Cinema language offers an interesting approach to enrich computer interfaces to augment users' reception. This section explains the set of techniques applied to enhance the ADD interfaces that are dedicated to present the designer's explanations of design projects.

4.1. Cinema and animation techniques

Cinema language is composed of cinema techniques (Davenport et al., 1991; Lester & Bares, 1997*a*). These techniques are heuristic rules that bring the real world to the movie screen with all its visual, temporal, and sound restrictions. Animation needs further techniques, because the interface is different: there is a transition from a real visual medium to an imaginary one.

Cinema techniques are classified in five groups (Silverstein & Huss, 1968):

- camera movements, such as zoom, pan shot;
- camera positioning, such as close-up, wide shot;
- edition, such as cut, cross-cut;
- style, such as fiction, silent movies, documentary; and
- narrative, such as slow motion, reordering, flashback.

Animation techniques can emphasize actions and physical processes that could not be perceived using other techniques (Lachman, 1997). Animated movements may help users to imagine (rationale reconstruction) what might have happened during a design process, making it easier to visualize concepts, objects, and thoughts. There are seven animation rules (Thomas & Johnston, 1984):

- *Anticipation*: The character movements are anticipated, so the audience knows in advance which movement will occur, generating expectation and attention.

https://doi.org/10.1017/S0890060402163086 Published online by Cambridge University Press

- *Deformation*: Some elements are deformed during collisions. These deformations must be anticipated and exaggerated, generating expectation.
- *Continued actions*: Two simultaneous actions that came from the same event must not begin or end together, thus focusing the attention to each one individually.
- *Secondary movements*: The effects of an action that occurred on an object must be propagated to the objects related to it.
- *Movement sequence*: When an action is initiated, it cannot be drastically finished. Postponing it allows emphasis.
- *Exaggerated movement*: Some actions may not be perceived if they appear as they occur in real world; however, when some movements are exaggerated, the action is emphasized.
- *Scenario creation*: The objects must be placed on the screen, so that the action of the characters can be observed clearly.

These rules are essential to emphasize, efficiently and pleasantly, the most important elements of a scene, without distracting the audience. Directing the audience's attention to specific actions or scene elements is one of the most important features of animation planning.

There are also special techniques for focusing the audience is attention on some aspect of an image (Blinn, 1994):

- *pointers*: like arrows pointing to the object that must be highlighted;
- *blinking*: objects blinking on the screen are not a very subtle way of emphasizing them, but it works; and
- *saw effect*: show some interleaving scenes with the state of an object before and after an action affected it.

A text can also be used to emphasize, clarify, and deeply analyze the information behind a movie or an animation. Including a text over the images is a common technique, even in movies, where there are captions and graphic animations. The inclusion of captions on an animation, however, has two main aspects: the size of the text and the screen position for it.

The animation and cinema techniques deal with focusing the audience's attention on relevant information that must be transmitted. The use of these techniques in a computational environment requires some precise definitions of its utilization and organization.

4.2. Idioms

Idioms are scripts that contain well designed rules for positioning and moving the camera. They also contain allocation time for scenes and takes and the behavior of the elements filmed. The rule set encompasses the filmmaker's expertise for the capture of an event sequence that tends to be repeated during the film (Christianson et al., 1996; He et al., 1996).

A frequent idiom used in movie production is the dialog idiom. This idiom defines what should be presented to the audience. This idiom applied to the dialog of two actors consists of three steps: introduce the world containing all participants, present each participant individually when he or she is speaking, and, show the world again. As illustrated in Figure 2, in the first shot the camera shows both actors in wide shot. After that the scene alternates a close shot of each actor individually. Finally, the scene ends with the same wide shot of the two actors.

The rules for constructing scenes or takes can be decomposed and grouped, creating generic idioms that incorporate specific information on a set of cinematography techniques. Incorporating idioms to script specifications of a film allows the creation of independence between the visual techniques used and the domain elements that will be shot.

4.3. Storytelling techniques and styles

The realm of storytelling is where the events take place. There are different storytelling styles that have been improved over the years. New heuristic techniques have been included as new media, like TV and sound, alter the art of making a film (Katz, 1991).

Cinema storytelling can be grouped in three categories (Arijon, 1976):

- *News flashes* deal with unpredictable facts whose final result is a set of disconnected images that must be edited.
- *Documentaries* deal with a sequence of situations with a common motivation.
- *Fiction films* deal with real events, but the events can be repeated until they capture the director's desire. There is no single point of view.

Fig. 2. An example of an idiom applied to a dialog scene between two actors.

https://doi.org/10.1017/S0890060402163086 Published online by Cambridge University Press

In Lester and Bares work (1997*b*), the storytelling styles are defined following the generalization of the users' profiles. These profiles are defined according to the cinema techniques adopted by users to visualize animations. The defined styles are incorporated into the system and cannot be modified.

Works on interactive movies Davenport (1996) and on automatic generation of cinematography storytelling Brooks (1997) have been very helpful in turning a computer into a storytelling agent, thus changing the limits between the filmmaker and the audience. The movie becomes interactive in the sense that the audience may change the script for producing the movie in order to get other perspectives on the facts to be revealed. There are two sides to analyzing the benefit of manipulating a story. The benefit is that it allows people to get a deeper understanding by trying different ways to explore a story, given a set of scenes. On the other hand, the audience may get confused and may miss the message that the filmmakers were trying to send through the movie. This issue is not unique to interactive movies. It extends from interactive textual narratives through hypertext advances.

5. CineADD: SHOOTING DESIGN DECISIONS SCENARIOS

CineADD is an extension to the ADD explanation model. It represents the way movie animation is automatically generated to augment an explanation on a design built in the ADD environment. Even though computer graphics techniques are used to actually produce the scenes, there is nothing new about them. The research focus is to let designers create explanation narrative scripts for generating design explanations.

CineADD, as illustrated in Figure 3, uses three input sources to work:

- *Design log* contains the sequence of decisions reflecting design project development. Playing this log replays the entire "design movie scenes." Generally, it lacks structure and a straight line of reasoning.
- *Domain knowledge base* contains an instance of the parametric dependency network, the assigned values, and the inference rules applied in a specific design case of a domain.

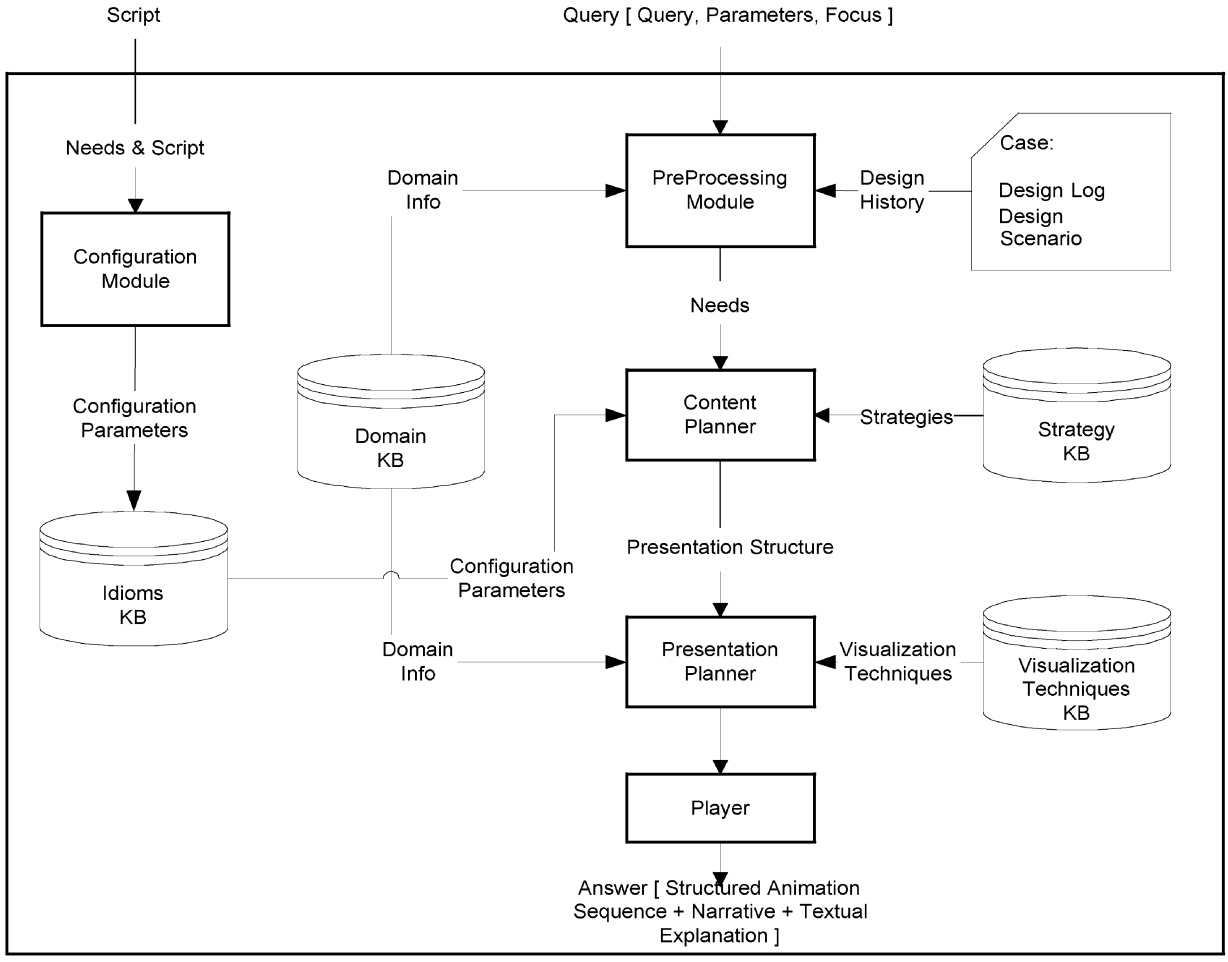

Fig. 3. The CineADD model as a complement of the ADD textual explanation generator.

https://doi.org/10.1017/S0890060402163086 Published online by Cambridge University Press

- *Query* represents the user's needs entered using the explanation interface. It contains the type of question, such as "Why" or "How"; the target decision to be explained, such as the entire design; and the focus of the explanation, such as the user's wish to understand a decision or need to approve the project.

The CineADD preprocessing module represents the filtering that must occur to determine the user's needs and the portion of knowledge and history that must play a role in the answer.

The strategy knowledge base contains heuristics for selecting the content from the design history to generate an explanation that fulfills the user's needs. For instance, a strategic rule for explaining a decision that was forced by the designer may consist of selecting the parameters that map to the decision, the immediate dependent parameters and their assigned values, what should be expected as the decision, a sign that the decision is not fully explainable, and, possibly, an annotation made by the designer on that decision.

Designers involved in project development may configure the way idioms should be selected, thus defining the message's format. For instance, a designer may want to make a zoom in an object that is under too many visual constraints to emphasize the difficulty of positioning it in 2-D space. He may build an idiom that shows the entire scene with all objects, followed by zooming in on the object that is overconstrained.

The content planner selects a set of strategies from the strategy knowledge base and a set of idioms that should be applied to build the scenes to satisfy the user's needs, and produce the presentation structure.

The presentation planner is responsible for creating the actual movie explanation. It selects visualization techniques, such as zooming or pan actions (Dufaux & Moscheni, 1996; McReynolds & Blythe, 1998), from the visualization techniques knowledge base and applies them to the presentation structure filled with information gathered from the design domain knowledge base.

After all these processes are done, the created movie is ready to be played. The visual answer works as a complement of the textual answer. Further research will look into whether the visual animations can be complemented by spoken language.

CineADD relies on an interaction history file, called the design log. The design log contains the user's actions while developing a design in the ADD environment and the domain knowledge base (represented as a parametric dependency network). From this raw material, CineADD applies cinema and animation techniques to organize and compose a visual presentation that works as a complement to the textual explanation generated by the system. Therefore, the end users' attention is guided to perceive the designer's intent.

Together with the design log, the domain knowledge base is used to reconstruct the design history. Rebuilding design can be accomplished through two approaches: replay and rebuild.

The replay approach consists of presenting the design history exactly as it was. The actions are shown to the audience in the same sequence in which they were made. In addition, even irrelevant actions are presented. This is a complete reconstruction of the facts. End users should watch the presentation carefully, observing each detail and interpreting the entire set of actions to understand what happened in the design process.

The rebuild approach consists of selecting a relevant set of actions from the design log (design scenes) to create a presentation that satisfies a specific user's question. Relevance is defined in the designer's mind. When defining idioms, designers indicate the type of scenes and the sets of actions to comply with explanation goals. The rebuild approach follows a method consisting of the following:

- interpreting the needs behind a user's question (preprocessing);
- selecting relevant scenes from the design log that plays an important role in building an answer (content planner);
- planning the visual presentation (visual planner); and
- presenting the movie answer to the observer (player).

The rebuild approach offers an explicit language with which the designers' and observers' plan design sequences can be interpreted and presented. There are three expected planner agents:

- the author, who is responsible for creating the presentation strategies;
- the designer, who is responsible for building the design and, consequently, the design history log; and
- the observer, who is responsible for studying and understanding the project, that is, the inquirer agent.

CineADD allows designers and observers to switch roles to discuss a project. It is expected that the presentation strategies be predefined and included in CineADD. The designer should select the strategy that he or she wants to apply to explain any specific decision. These strategies reflect the perspective that the designer wants to share with the audience.

CineADD also lets the audience further investigate an explanation in order to allow any details that may be hidden in the designer's explanations to be brought up.

6. CineADD APPLIED TO OIL PIPELINE LAYOUT DESIGN

In this section, we will present an example of using CineADD in a real design domain. Before explaining the use of the cinema techniques, we present the application domain in the context of an ADD system.

https://doi.org/10.1017/S0890060402163086 Published online by Cambridge University Press

6.1. The oil pipeline layout design task

Oil exploitation in deep water fields needs special processes. The oil is pumped from the bottom of the sea to offshore platforms from which it is treated to be exported to land. The oil pipeline layout problem belongs to the class of spatial layout problems. There is a set of objects in a 2-D space that have to be located and connected, considering a number of restrictions of the environment. Finding a solution to this problem is not a simple task, considering the complexity of the environment and the information involved. To handle the complexity of the oil pipeline layout problem, we divided the task into seven different subtasks.

Given a set of wells with their target areas and a number of oil exploration units (usually called platforms), the subtasks are the following:

1. finding the best well cluster, considering the relative distance among the geometric centers of the elements;
2. assigning each platform to wells clusters, considering the maximum oil processing capacity of a platform and the maximum number of risers;
3. locating the well heads in each target region, in order to minimize the distance to the platform;
4. locating each platform in a free area as close as possible to its cluster geometric center;
5. defining the exact source and destination of a pipeline, from wells to platforms;
6. defining the pipeline route for draining the oil from wells to the assigned platforms; and
7. defining intermediate draining elements to receive oil from wells and bends it to the platforms.

Locating the wells, the platforms, and the other oil draining elements, as well as designing the pipeline connecting them, are the activities involved in an oil pipeline layout project. This is the most expensive part of an oil field exploitation project. Even though optimizing the project saves a great deal of money, it is rarely accomplished due to the complexity of the involved reasoning.

All the above decisions are made with a consideration of the spatial constraints of the environment. Partial decisions are also considered for the domain model. For example, designers may locate only half the wells or they may create only one group of targets on which to focus their attention, leaving the remaining oil target areas to be grouped later.

6.2. ADDSub: A system to assist oil pipeline layout design

ADDSub system (Laboratório ADDLabs, 1998) is an intelligent CAD tool used to assist and document the designer's decisions during oil pipeline layout project development. ADDSub helps the users to optimize their project, but it is not merely a calculation tool to find the best solution for each of the subtasks described in the previous section. The problem is very complex, and the order in which each decision is made affects the overall solution of the problem. Therefore, to find one best solution for the whole problem would be computationally intractable. The ADDSub approach benefits from the partnership between the designer and the system (taking advantage of the computer's fast calculation and the designer's expertise and visualization facility).

ADDSub offers a friendly interface, presenting in a canvas active area the undersea topography and texture, as well as the existing objects. ADDSub offers a direct manipulation interface where objects are displayed and modified on this canvas. Figure 4 illustrates a small fictitious oil pipeline layout project developed with ADDSub. As we can see from the figure, there are seven oil target areas (big circles) and a big obstacle area (irregular polygon). The wells (small circles) are connected to the corresponding platform (rectangle) by a pipeline (thick lines), and the slim lines represent the undersea topography.

Following the ADD model, ADDSub observes the designer's actions and compares them with the rationale stored in its knowledge base. The system can disagree with the designer's action, presenting another suggestion for it. The designer chooses which actions or decisions will be adopted in the final project.

ADDSub operates in six different modes: data entry, suggest, verify, free, knowledge acquisition, and explanation. In the data entry mode, the designer inputs the data that configure the project to be developed. In the suggest mode, the designer requests suggestions from the system on each of the layout design subtasks. In the verify mode, the designer proposes a solution to a subtask and the system analyzes it. In the free mode, the designer imposes a solution without asking the system to analyze it; the system works as regular drawing software in this mode. In the knowledge acquisition mode, the designer may include or modify calculation methods and design criteria. Finally, in the explanation mode, the system provides explanations on the decisions made during the project development. In the next section we present a detailed description about the explanations given by ADDSub.

ADDSub was developed in C++ under Windows and is being successfully used by the Brazilian Oil Company (Petrobras) to develop pipeline layout projects. The use of ADDSub provides Petrobras with three important benefits: reduction of the project development time; reduction of the overall project cost (as it optimizes the project elements); and generation of automatic project documentation (with explanations).

6.3. Explaining design decisions in the ADDSub context

When designers create a project, they do not consciously create explanations of their rationale. ADDSub automati-

https://doi.org/10.1017/S0890060402163086 Published online by Cambridge University Press

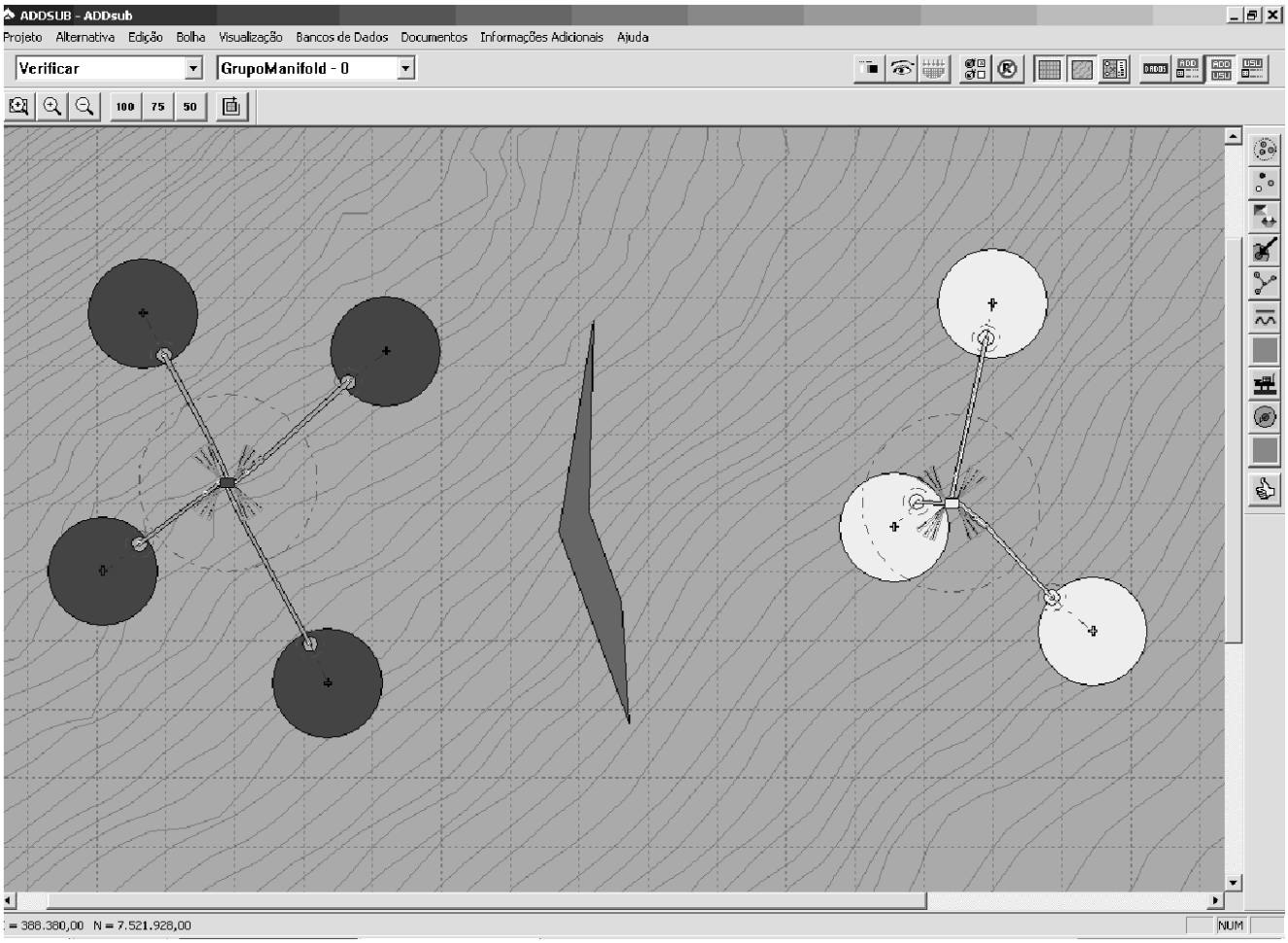

Fig. 4. An example of an oil pipeline layout project developed with ADDSub.

cally generates these explanations, as the system is used to make design decisions. To provide explanations, ADDSub records, in a design log, both the sequence of design decisions made in each of its operation modes and the agent (system or designer) that generates that decision. The design log provides a chronological view of the designer's actions in a project.

Using the design log information, the system can provide explanations for all the decisions made during project development. Each decision is represented as a parameter, its assigned value, the agent who assigned this value, and information about the decision compliance (or not) with the system model.

In the explanation mode, the designer (or the observer) interacts with the system by writing a question. ADDSub provides answers for three types of questions: "how," "why," and "which." With these questions designers or observers usually want to know why a system parameter has a certain value. Then the system provides a textual explanation answering the question. Besides this textual explanation, the system provides the design history, the dependence network, and the heuristics. The design history contains the sequence of decisions that occurred before the parameter value was calculated, which is a linear sequence of actions. The dependence network contains the part of the parametric network that refers to the parameter of the question. The heuristics contain the heuristics or formula used to calculate the value of the parameter asked.

The textual answer given by ADDSub is generated using natural language. The text generation uses narrative and rhetorical structure (Mann & Thompson, 1987) to build a textual explanation that delivers the knowledge behind the set of information pieces.

6.4. Augmenting explanations with CineADD

Oil pipeline layout design is mainly a visual task, as it involves locating and changing the location of elements in a 2-D area. The text explanations given by ADDSub may not clearly express the answer the user needs. When too many visual actions occur, a text or even pictures telling about design decisions does not suffice to let the informa-

https://doi.org/10.1017/S0890060402163086 Published online by Cambridge University Press

tion emerge. It keeps bouncing from wordy to concise sentences, causing a cognitive overload on users to create a mental image to understand the designer's intentions. Next we explain this argument using a very simple example of how CineADD can improve an explanation about one decision made by ADDSub concerning the location of a specific well.

Suppose a user wants to know why well number 5 was located at coordinate (x, y). The user formulates a question to the system as the following:

Why was well 5 located at (x, y)?

The system provides information about the design history, the heuristics used, the dependencies used to solve that question, a view of the well location, and a textual explanation, as follows:

Well 5 was located at coordinate (x, y) because:
it does not violate any spatial constraint
it is the closest location to platform 1

This answer does not give a clue to users whether other alternative locations were even tried. It may be the case that other solutions also leading to minimum distance were possible or even preferable. The system can deal with that flaw by showing a text explaining the other considered alternative solutions and why they were discarded. The answer would be:

Well 5 was located at coordinate (x, y); *Alternative 4*, because:
it does not violate any spatial constraint; and
it is the closest location to platform 1
Alternative 1: location $(x1, y1)$, distance to platform 1: K1. Discarded because it violates restriction: intercept existing element.
Alternative 2: location $(x2, y2)$, distance to platform 1: K2. Discarded because it violates restriction: it does not lead to minimum distance.
Alternative 3: location $(x3, y3)$, distance to platform 1: K3. Discarded because it violates restriction: intercept existing element.
Alternative 5: location $(x5, y5)$, distance to platform 1: K5. Discarded because it violates restriction: intercept existing element.

The text continues until all alternatives are listed.

We can see from this example that the concise text explains only about location (x, y), but does not provide the answer for "why not choose the other positions?" The complete text answer for this question, on the other hand, becomes wordy, making it very difficult for users to visualize it.

The CineADD model provides ADDSub with an enhanced explanation interface, as shown in Figure 5, because it gives the answer in an animated fashion. The user can see the decisions that were made without having to analyze an overloaded text. For example, for the same question on the localization of well number 5, the CineADD presents a movie showing why location (x, y) was chosen. Although it is very difficult to show the actual advantages of seeing a movie on paper, in Figure 6 we will try to show some scenes of the movie that answer this question.

The whole project is shown in Figure 5. This scene is cut and edited with the next scene, which is a pan on the project, to centralize well 5 on the screen. After centralizing the well, the next scene provides a zoom in the specified well. These scenes provide the user the notion of where the target area of well 5 is in the project. After that, the movie begins to show all the possible alternatives for locating well 5 on that target area. There is a sequence (which was omitted) showing each alternative for well placing. When the alternative violates any constraint, a square marks this violation and there is text explaining it in the text area of the screen. The last scene shows the chosen location of the well.

6.5. CineADD in ADDSub

As explained in Section 5, CineADD is based on the replay and rebuild approaches. In ADDSub, the replay approach is built based on the actions and decisions stored in the design log. CineADD creates scenes for each kind of action, like data entry and the creation of new elements. For the subsequent actions, like moving an element or suggesting new alternatives, the system creates takes. After looking over the entire design log, the system generates all the sequences of scenes and takes that compose the replay approach. This movie recomposes all the sequences of actions done during the project development.

The rebuild approach is built after the interpretation of the observer question, based on a list of techniques or idioms configured by the designer (or by default if none is specified). This feature allows designers to create the rebuild script, directing the design movie. For each decision type, or even a specific decision, the designer can eliminate or create scenes; choose camera effects; include different perspectives of a scene, texts in special boxes, and written frames; and select the gluing effects between scenes.

The techniques available in CineADD to create the rebuild script of an ADDSub project were chosen based on the characteristics of the specific domain studied. Oil pipeline projects usually involve a great number of graphically independent elements (e.g., platforms, wells, pipelines) present in a 2-D canvas area, where most of the actions taken on an element affect only a small part of the project. Therefore, CineADD uses techniques that are able to highlight a single element of the project; make it easy to visualize the actions taken on an element and to focus on details; and show up the relevant actions that affected a set of elements. The set of the techniques used is divided into three groups:

https://doi.org/10.1017/S0890060402163086 Published online by Cambridge University Press

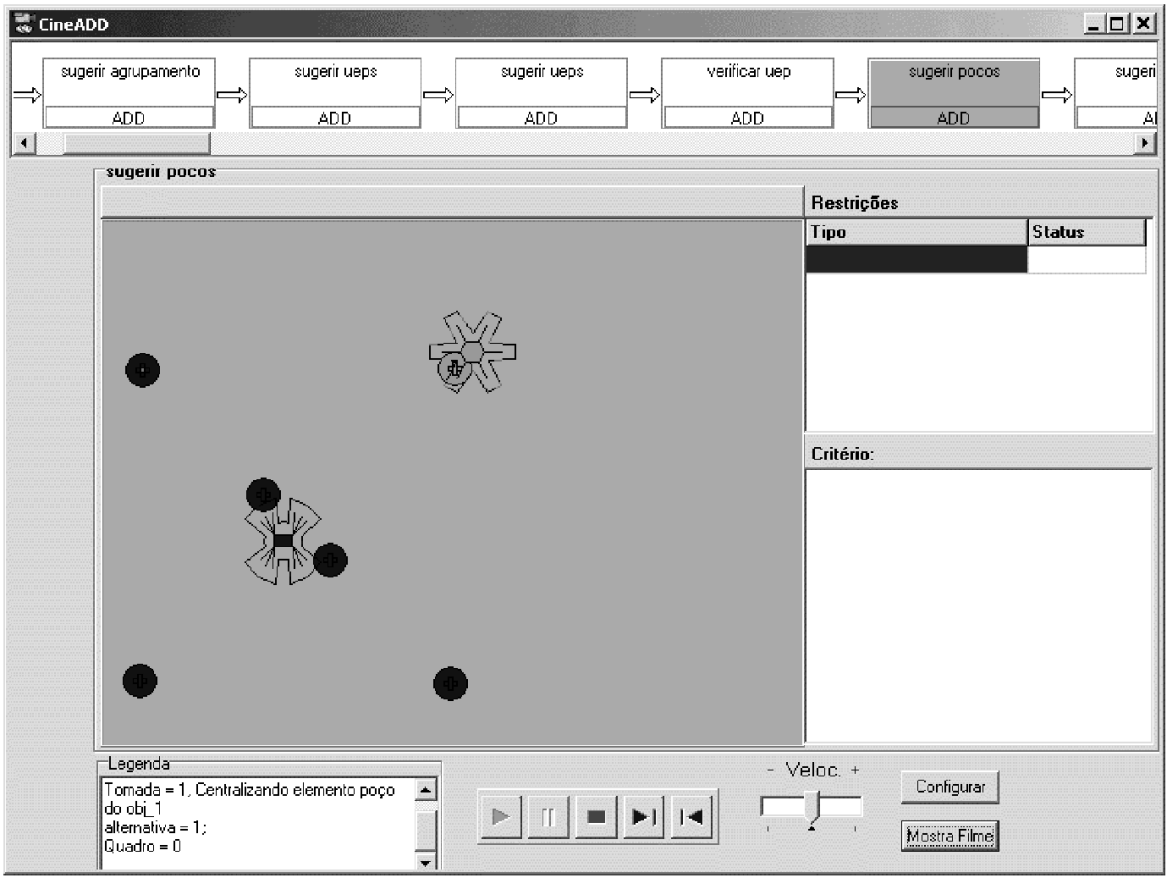

Fig. 5. The CineADD interface.

- *Cinema*
  - camera movement (zoom and pan): to emphasize small details and locate in the canvas elements that must be focused;
  - narrative (reordering, visibility, time of a frame): to reorder the sequence of actions (e.g., show the platform positioning before the well positioning), make some takes visible or invisible, and define the time for which a frame must appear (a frame that contains a constraints violation must stay longer than others, as the observer has to realize which constraint was violated);
  - style (silent movie): to draw attention to transitions between scenes, explaining briefly in text what will happen next; and
  - editing (abrupt cut): to cut some intermediary frames with no special effects in order to merge scenes;
- *Animation*
  - pointer and blinking: to highlight the elements that are the focus of the attention;
- *Text*
  - insertion of text: to complement the animation, giving additional information for some specific actions

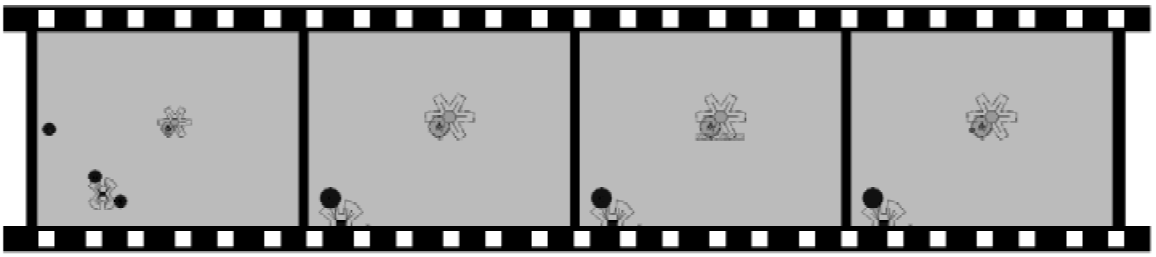

Fig. 6. The CineADD in progress.

https://doi.org/10.1017/S0890060402163086 Published online by Cambridge University Press

(e.g., explain which constraint is violated on the well positioning).

7. CONCLUSIONS

Storytelling is a very efficient way to explain a fact to an audience (Bers & Cassell, 1997; Gershon, 2001). It is much more effective to give the audience an interpretation of the fact than to give them all the facts and let them make sense of them. Delivering an interpretation lowers the audience's cognitive load. In addition, the audience can be conducted to a specific viewpoint.

Cinema is an interesting communication medium with which to tell a story and, consequently, send a message. This medium, when properly used, may permit long and complex messages to be transformed into simple ones (e.g., *Wuthering Heights* by Emily Brontë or *Anna Karenina* by Lev Tolstoy).

This paper presented a model that includes design cinematic techniques to improve computer user communication for the task of design rationale delivery. Because design documentation is developed as a design task subproduct in an ADD environment, the boring, time-consuming, and repetitive aspects of it can be substituted by a creative process, generating new perspectives for the problem. CineADD enhances traditional interaction by providing users with an immersive experience afforded by a cinematic environment. CineADD applies cinema and animation techniques to arrange and compose a visual presentation (a movie), which serves as a complement to the textual explanation generated by the system.

CineADD was implemented in an intelligent CAD system, called ADDSub, and used in a real design domain (oil pipeline layout). As this layout design is mainly a visual task and involves locating and changing the locations of elements in a 2-D area, the main contribution of CineADD in ADDSub is to provide an enhanced explanation interface, allowing designers to configure the way a visual explanation should be created. Designers play filmmakers in CineADD. They create idioms that define the explanation narrative, such as scene selection, scene sequencing, scene effects, and scene merge, to be created to explain each type of decision in a project.

Users may doubt an explanation and may pursue a further investigation by creating new idioms for playing the scenes. This action may lead to reveal new perspectives in the designers' narrative. However, this functionality may lead to more misconceptions and should require a deeper HCI study. Initial results of using CineADD to deliver design rationale in the oil pipeline design domain have shown that designers have some difficulties in learning how to build their movie; however, the potential increase of understanding by the design team and even the end users makes it worth learning.

CineADD was meant for designers to build design movies. However, because the design scenes are available and the movie builder allows a story to be easily constructed, end users can also take the director's role and uncover details or interpretations that might be hidden by the designers. Therefore, CineADD allows interactive cinema to be pursued in the context of engineering design.

REFERENCES

Arijon, D. (1976). *Grammar of the film language*. New York: Communication Arts Books, Hasting House.

Bers, M., & Cassell, J. (1997). Storytelling systems: Constructing the innerface of the interface. *Proc. IEEE Cognitive Technologies, CT'97 (2nd Int. Conf. Cognitive Technology)*, Los Alamitos, CA, pp. 98–108.

Blinn, J. (1994). Animation tricks. *SIGGRAPH '94*. Course notes for Animation Tricks course.

Brooks, K.M. (1997). Programming narrative. *VL'97, IEEE Symposium on Visual Languages*, Capri, Italy.

Christianson, D.B., Anderson, S.E., He, L., Salesin, D.H., Weld, D.S., & Cohen, M.F. (1996). Declarative camera control for automatic cinematography. *Proc. AAAI-96*, pp. 148–155.

Davenport, G. (1996). Smarter tools for storytelling: Are they just around the corner? *Visions and Views: IEEE Multimedia 4*, 10–14.

Davenport, G., Smith, A., & Pincever, N. (1991). Cinematic primitives for multimedia. *Proc. IEEE Computer Graphics and Animation*, pp. 67–74. Special issue on multimedia.

Dufaux, F., & Moscheni, F. (1996). Segmentation based motion estimation for second generation video coding techniques. In *Video Coding: The Second Generation Approach* (Torres, L., & Kunt, M., Eds.), pp. 219–263. Dordrecht: Kluwer.

Fischer, G., Lemke, A.C., McCall, R., & Morch, A.I. (1991). Making argumentation serve design. *Human Computer Interaction 6*, 393–419.

Garcia, A.C. (1992). *Active Design Documents: A new approach for supporting documentation in preliminary routine design*. PhD Thesis. Stanford University, Stanford, CA: Civil Engineering Department.

Garcia, A.C., Andrade, J.C., Ferreira, R., & Moura, R. (1997). ADDVAC: Applying active design documents for the capture, retrieval and use of rationale during offshore platform VAC design. *IAAI 97 Emerging Applications*.

Gershon, N. (2001). What storytelling can do for information visualization. *Communications of the ACM 44*, 31–37.

He, L., Cohen, M., & Salesin, D. (1996). The virtual cinematographer: A paradigm for automatic real time camera control and directing. *Proc. ACM SIGGRAPH '96*, pp. 217–224.

Katz, S.D. (1991). *Film Directing Shot by Shot*. Amsterdam: Elsevier.

Laboratório ADDLabs. (1998). Documentação do Projeto ADDSub. Niterói, RJ, Brazil.

Lachman, R. (1997). Experiments in mapping character animation to computer interface. *Proc. IJCAI Workshop on Animated Interface Agents*, Nagoya, Japan.

Lester, J.C., & Bares, W.H. (1997*a*). Cinematographic user model for automated realtime camera control in dynamic 3D environments. *Proc. Sixth Int. Conf. User Modeling*, pp. 215–216.

Lester, J.C., & Bares, W.H. (1997*b*). Realtime generation of customized 3D animated explanations for knowledge-based learning environments. *Proc. Fourteenth National Conf. Artificial Intelligence (AAAI-97)*, pp. 347–354.

Mann, W.C., & Thompson, S.A. (1987). Rhetorical structure theory: A theory of text organization. In *Report ISI/RS University of Southern California*, Marina del Rey, CA, pp. 87–190.

McReynolds, T., & Blythe, D. (1998). Advanced graphics programming techniques using OpenGL. *SIGGRAPH '98*. Course notes. New York: ACM Press.

Moran, T.P., & Carroll, J.M. (1996). *Design Rationale Concepts, Techniques and Use*. Mahwah, NJ: Erlbaum.

Persson, P. (1999). Understanding representations of space: A comparison of visualisation techniques in mainstream cinema and computer interfaces. In *Social Navigation in Information Space*, (Munro, A., Höök, K., & Benyon, D., Eds.), pp. 195–216. London: Springer.

Persson, P. (2001). Cinema and computers: Spatial practices within emergent visual technologies. In *New Technology and Space* (Munt, S., Ed.). New York: Continuum Publishers.

https://doi.org/10.1017/S0890060402163086 Published online by Cambridge University Press

Silverstein, N., & Huss, R. (1968). *The Film Experience*. Dell Publishing.
ten Hagen, P.J.W., & Tomiyama, T. (1987). *Intelligent CAD System I*. New York: Springer–Verlag.
Thomas, F., & Johnston, O. (1984). *Disney Animation: The Illusion of Life*. New York: Abbeville Press.
Vivacqua, A.S., & Garcia, A.C. (1996). The use of active design documents to assist conflict mitigation in concurrent engineering. *Third Conf. Concurrent Engineering and Research Applications*.

Ana Cristina Bicharra Garcia received her BS in civil engineering from Universidade Federal do Rio de Janeiro, Brazil, in 1983. She obtained her PhD degree from Stanford University, Stanford, CA, in 1992. After graduating, she returned to Brazil to work as a professor at the Computer Science Department at Universidade Federal Fluminense in Rio de Janeiro, Brazil. She has also been the director of the ADDLabs, an artificial intelligence (AI) lab, since 1994. Dr. Garcia's research interests include knowledge acquisition, agents, and HCI.

Carlos Eduardo Carretti received the BS degree from Pontifícia Católica do Rio de Janeiro and the MS degree from Universidade Federal Fluminense. His research interests include AI, cinema, and HCI.

Inhauma Neves Ferraz received a BS in civil engineering from Instituto Militar de Engenharia (IME), a BS in mathematics from Universidade Federal do Rio de Janeiro, an MS in mechanical engineering from Escola de Engenharia de Itajubá, and an MS in computer science from IME. He was Professor at the Instituto Tecnologico da Aeronautica from 1970 to 1979. From 1979 to 1991 he taught at IME. Since 1992 he has been with Universidade Federal Fluminense, where he is currently a professor of computer science. Professor Ferraz's research interests include AI, neural networks, and coding theory.

Cristiana Bentes received her BS in Mathematics from Universidade do Estado do Rio de Janeiro (UERJ). She obtained her MS and DS from Universidade Federal do Rio de Janeiro. Since 1993 she has been with UERJ, where she is currently Professor of the Department of Systems Engineering. Dr. Bentes' research interests include AI, optimization, HCI, and high performance computing.

https://doi.org/10.1017/S0890060402163086 Published online by Cambridge University Press