CU/USW Research Collaboration Mid-Project Report: 1991-1992

Ed Freeman & Clayton Lewis

CU-CS-607-92 August 1992

CU/USW RESEARCH COLLABORATION MID-PROJECT REPORT: 1991-1992

Edited By

Ed Freeman
U S WEST Advanced Technologies

Clayton Lewis
CU Boulder

CU/USW RESEARCH COLLABORATION FIRST YEAR REPORT

1. INTRODUCTION

The CU/U S WEST research collaboration was formed in December of 1990, as a three year research project. The Collaboration is composed of a team of scientists from U S WEST and the University of Colorado who are working together to develop advanced computing technology for U S WEST.

The ultimate, long-term goal of this particular collaboration is to facilitate the implementation of a framework for Direct Computer Usage (DCU), thus making it possible for people who need computing or information services to define and create the computing solutions they need, without having to work through support intermediaries. The phrase "Direct Computer Usage" is intended to suggest an analogy with "Direct Distance Dialing" (DDD). The implementation of DDD has made it possible for telephone users to place their own long distance calls, rather than working through telephone operators, as was once necessary. This "break-through" in long distance dialing has greatly increased the individual's ability to efficiently and effectively, solve their own long distance communications problems. We would like to have the same type of impact on U S WEST computer usage.

In order to develop the technology for DCU in a way that will maximize benefits for U S WEST, the Collaboration is developing a collection of strategic case studies, upon which applied research efforts will focus. These case studies are concerned with specific situations in U S WEST's business, where users must now work through intermediaries to obtain computational solutions and where significant cost reductions or other benefits would result, if users could work more directly with the particular computational medium required to solve their problems. The Collaboration is using the case studies to achieve two primary objectives: (1) to define the requirements and architecture for viable DCU technologies and (2) to plan the development of the required technologies in a form that will permit the selected case study problems to be solved.

It should be emphasized that DCU is a long-term strategic goal requiring fundamental advances in Computer Science as well as in other disciplines. Therefore even though valuable insights, consulting input, and problem solutions have been successfully provided in certain limited situations, the Collaboration cannot guarantee working solutions to most case study problems, in the near term. However, working with the Collaboration can offer benefits to organizations whose problems are adopted as case studies. Given that each case study will be the subject of a careful technical assessment and a technology plan, there will be a much stronger chance that significant development resources may be allocated to case studies for which cost effective solutions can be identified.

This document presents a read-out of our progress during the seventeen months of this research effort. Based on a request from USWAT management for a fast ramp-up period, we modified our original plans to conduct extensive (and potentially time consuming) U S WEST-wide interviews, before picking an initial set of case studies. Instead, we decided to capitalize on the ground work already performed by USWAT directors and project leaders. Accordingly after considerable discussion, our preliminary set of case studies was chosen from existing USWAT projects that were deemed significant to U S WEST and were identified as already having strong customer support. Our longer term working strategy is to continue our original interview plans, but to do this in parallel with our work on the preliminary set of case studies. In this way, we hope to maximize potential short term impact, while continuing to search for new projects that will maximize our potential for significant long term impact.

Reflections on the First Year:

At the close of the first year of work we offer the following observations on what is working well and not so well in the collaboration.

Four projects are in progress, differing substantially in the way in which they embody collaboration between US West and the University. The Voice Dialog project seems to embody most fully our hopes for the collaboration. Here CU students and faculty have been working closely with Mike King to develop a comprehensive design aid for voice dialog applications. A prototype visual editing environment has been built, and much has been learned on both sides about the requirements for design support and how this might be provided. It is too early to assert that an operational tool can be built on the lines now envisioned, and the demands of producing an "industrial strength" tool are not to be minimized, but it is hard to see how better progress towards these goals could have been made.

The OMIE project has focussed on the development of a framework for access to multiple databases by nonspecialist users. This project too has featured close interaction between CU and US West people and is making good technical progress. This project now faces the challenge of making a transition between application areas because the parent project within US West is not continuing, in its original form. The underlying technology is still of importance, and the work is continuing as part of a new project in the domain of remote access to multimedia documents.

The War Room project contributed to the conceptual design of a model-centered communication and decision support system for enterprise managers. In a sense, the war room scenario development efforts were a precursor to the more extensive scenario development projects conducted this year, and provided significant input and insight into the potential of a real time decision support war room. The project has also developed new ideas of wider potential application in the representation of complex business models, especially those requiring a mix of traditional financial data with qualitative influences like changes in competitive environment. Unfortunately the level of direct collaboration suffered when Warren Kuehner, the original AT partner in the project, moved to Spectrum Enterprises. Some work is continuing on the CU side, but there is a need to identify new points of contact within US West for this work, or to redirect this effort.

The graph editor project is the smallest and least directly collaborative: here a CU doctoral student is doing thesis work on a generic graph editor. This technology is of broad importance to US West, and there has been input to the thesis by a number of U S WEST AT staff members, representing a number of US West application areas. At this point, however, the work is primarily a CU effort. Technology transfer plans are currently being put in place, to move this technology into the USWAT mainstream.

We draw the following lessons from our experience so far.

First, the effort required to establish working projects can easily be underestimated. We had very ambitious plans at the start of the collaboration for soliciting participation from many parts of US West, extending well outside of AT. It became clear, however, that establishing such relationships would require a great deal of time and effort, and further that much of this investment would have to be made by US West staff, with little help from

CU people. For example, we could not ask CU graduate students to locate and initiate discussion with managers in US West. Thus the resources of the collaboration itself could not be effectively mobilized for this job.

Our response to this problem was to change to a strategy of working with existing AT projects. This allowed us to get started more quickly, at the cost of foregoing the aim of reaching outside AT. We think the new program structure in AT may make this cost less important: if we can build the right relationships between the collaboration and AT programs this should bring with it the relevance to US West at large that we were seeking originally.

A second lesson is that matching the rhythms of activity at US West and the University is a challenge. While graduate students are busy people, they normally expect to be able to devote as much as half time to a single project, and have that project remain fairly stable for a period of two years or more, a common funding period for a research grant. Demands of business make it hard for projects in US West to remain stable for that long, as has been illustrated in two of our projects. Further, the life of a US West project may have gaps in it in which decision makers are being contacted, presentations made over a considerable time, people are preempted by other activities, and the like. Because US West people are typically working on several projects concurrently, these gaps may not be too disruptive. They do, however, cause trouble for students, not because students have nothing gainful to turn to, but because these alternative activities aren't likely to be part of the CU/USW collaboration.

We believe that one possible response to this problem is to change somewhat our expectations about CU's contributions. While the Voice Dialog project shows that a sustained, in depth involvement is certainly possible, we should allow for more "light weight," high-impact interactions to occur. For example even though the future of CU's participation in the War Room project is currently unclear, CU collaborators were able to contribute some very useful ideas in the form of possible decision support scenarios. The collaboration could try to promote more white paper/scenario development activities such as this, at least until more substantial and stable projects are established. This would keep the graduate student resource more fully occupied while at the same time maintaining a flow of useful ideas to US West. Obviously the collaboration would still remain strongly committed to full participation in U S WEST projects whenever possible.

In the following sections, we first provide an overview and a status report on the four DCU case studies that have been selected for our preliminary work. Summaries are then provided for two common DCU-related technology threads, which cut across the various case studies. We close with a discussion of "next steps" and provide an overview of what we would like to accomplish for the remainder of 1991. For those readers who would like more detail, an appendix has been included, which contains detailed write-ups on each of the current DCU case studies.

CASE STUDY OVERVIEWS

Generic Network Editing

<u>Participants</u>

US WEST:	Mike Epstein,
CU:	Jeff McWhirter, Gary Nutt

Related Projects

Mission Control Facility, PRONET, SPS, MOBIAS, GEB, Voice Dialog Development

Project Summary

Generic Network Modeling: There are many common requirements shared among network model based graphical interfaces and systems. These requirements fall into such categories as graph visualization, network structures, network algorithms and interface functionality. It often requires considerable effort to develop a visual interface to suit the particular needs and requirements of a network model. This level of effort often precludes the development of visual interfaces to network model based applications. For those applications that depend on a visual interface the construction of the network editor may constitute a large portion of the application development effort. Much of the effort involved in the process of developing and exploring the uses of a network model based system is devoted to the development of the network editor.

As part of the US West/CU collaboration we have been exploring issues related to generating network model based systems. These issues involve identifying and characterizing common network model properties and system requirements. There are currently two projects which relate to generating network model based systems: PRONET and GEB. PRONET, developed at US West, is a Prolog based tool which is an instantiation of the Generic Network Model System (GNMS). The major focus of GNMS is the support of graph analysis mechanisms such as path analysis, reachability, and pattern detection.

The GEB system is currently under development at CU. GEB takes a systemic view of network editor generation and is based on a very general characterization of the structure and visual representation of network models. This results in the applicability of GEB to a large set of network model based applications. The focus of GEB is on the underlying data model and the visual representation of the data model for a wide variety of complex graph model constructs and visualizations. The concepts that underlie GEB and GNMS will mesh well. GNMS focuses on graph analysis capabilities while GEB addresses model usage and visualization.

Status

The graph model characterization framework on which the GEB system is based has been completed. This framework allows one to describe a wide variety of structural, visual and system aspects of graph models. We have begun a development effort aimed at realizing this framework. Our goal is to allow for the specification of the structural and visual aspects of a graph model. From this model specification C++ software that supports the model specific components of a graphic editor will be generated. These parts support such functionality as internal model storage, model I/O, consistency, event propagation, and model representation. These model specific components will be combined with an editor template to realize a system that supports the target graph model.

References

McWhirter, J. D. and G. J. Nutt, *A Characterization Framework for Visual Languages*, University of Colorado, Boulder, Dept. of Computer Science Technical Report #CU-CS-586-92, March 1992.

The Mission Control Facility: Usability engineering for a decision support facility.

Participants

US WEST:	Warren Kuehner, Mike Epstein
CU:	Clayton Lewis, Nick Wilde

Related Projects

US WEST:	Strategic Planning System, PRONET
CU:	Cognitive Walkthroughs

Description

US West is developing a decision support facility for strategic level managers within US West Spectrum Enterprises. The users of the facility will be high level managers with little or no computer experience. Clearly providing this class of users with a computer based modeling facility of the type envisioned is a DCU-like task.

The Mission Control Facility is envisioned as a place that strategic level users can go that provides both easy to use modeling tools and support for accessing the wide variety of data that may be needed to make sound business decisions. It should allow users to capture and work with their own conception of the dynamics of the US West business environment - the "physics of the business", as well as to communicate that conception to others. Data access should be provided not only through traditional means of organizing and accessing computer based information, but also by organizing data along model-based lines. In this way, information supporting a particular model assumption or implementation can be uncovered ("drill down").

Researchers at CU have been developing a methodology (dubbed the "Cognitive Walkthrough") for developing walk-up and use applications for other application areas. We feel that this methodology can be used to gain valuable insights into the types of modeling software best suited to this application. To this end, we are applying cognitive walkthrough analyses to two different modeling packages already in use within US West.

We are also working closely with one of the proposed users of the Mission Control Facility to develop an implementation plan and a usage scenario which fits within the user's conception. By iterating between an implementation plan (the "implementors story") and a usage scenario (the "users story") we feel we should be able to constrain both products to represent a facility that is truly possible with today's technology, as well as useful to the user.

Status

Initial walkthroughs on the both SPS and Excel/@Risk based models revealed *information representation* to be of primary importance for the ability of using and comprehending models in both packages. Accordingly, we have spent much of our efforts in clarifying and understanding the issues involved with effective information representation in modeling tools such as SPS or Excel.

To do this, we have employed a variant of the walkthrough approach aimed at evaluating alternative model representations in light of their information presentation abilities. The approach was aimed at discovering weaknesses in both the spreadsheet-based (Excel) representation and the graph-based (SPS) representation, with an eye towards suggesting improvements, or a different representation that would combine the particular strengths of each. Several difficulties of each representation were identified, and alternative representations designed to alleviate the difficulties suggested (See the technical report reproduced in the appendix for further details).

Several global issues on information representation came out of this effort:

- **The presentation of information *in context.***

  One of the primary lessons learned was the importance of presenting information in a global context, as opposed to isolated chunks of information in answer to specific queries. Even when the user is looking for a specific piece of information in response to a direct query, presenting such information in the global context of the model can have benefits in facilitating the *uploading* of related information that may benefit the user later.

- **The use of 3D representations to increase available screen real estate.**

A fundamental problem in the design of any computer-based information representation is the effective use of available screen real-estate. Some designs (i.e. graph representations) present important information as to the structure of the underlying model in a relatively natural way but waste large amounts of space in doing so. Others (i.e. spreadsheet based designs) use the available space much more efficiently but end up hiding much of the model's structure out of sight, where the user must dig for it. The use of *3D representations* appears to us to be a natural way of breaking this deadlock, allowing space to be used in a more efficient manner while still making much of the structure available to the user in a relatively natural manner. We were surprised by how easily the notion of 3D representations fell out as the solution to some of the problems identified in the walkthrough analyses, without our (to the best of our knowledge) being predisposed to the use of 3D in the first place.

- **The use of multiple/different representations in concert.**

  During our analysis, it became clear to us that no one representation will do everything well: i.e., present exactly the right information, in exactly the right order, with exactly the right emphasis for the task at hand. Rather, different representations have different strengths, and should be used when the task at hand clearly calls for the particular strengths of one representation over another. Clearly, the *combination* of different representations to provide the strengths of each when needed has the potential for synergy, provided the different representations are combined in a coordinated and integrated manner. We are currently investigating the potential for such representations by looking at the possibility of fluid graph-based to spreadsheet conversions, and back again.

Future directions

As noted above, we have found the notion of a combined graph/spreadsheet based model representation an intriguing one, based on some of our investigations of both representations alone. We are currently investigating the potential for a combined graph/spreadsheet representation by looking at the problem of graph/spreadsheet *interconvertability.*

It is a fairly trivial matter to build a graph from a spreadsheet. To build a useful graph from a spreadsheet is another matter. Spreadsheets already contain a great deal of model-structure related information in the global arrangement of cells: although this is not formally enforced in the spreadsheet model, spreadsheet builder's often use globally organizing principles in constructing their models. We are currently looking at heuristic-based approaches to a graph based view of a spreadsheet that would preserve such information while augmenting it with additional relevant information, and that would allow such spreadsheets to be built from the graph-based view in the beginning.

References

Wilde, N. and Lewis, C., *Inventing new information representations through task analysis*. University of Colorado, Boulder, Dept. of Computer Science Technical Report #CU-CS-549-91, December 1991.

The oMIE Project

Participants

U S WEST:	Steve Laufmann, Pam Drew
CU:	Roger King, William McIver, Michael Novak, Pam Drew

Related Projects

U S WEST:	BROOM, MOBIAS, Profit, RMIE, Mission Control Facility,
CU:	A la carte, FaceKit, Heterogeneous Database Seed Projects

Project Summary

oMIE is an application of the Coarse-grained Agent system to real world problem solving involving the RMIE database. The Coarse-Grained Agent (CGA) system is the implementation of an agent metaphor for distributed problem solving. It is hoped that this metaphor can be used to achieve virtual resystemization of U S WEST databases and applications by providing a means to wrap them with agents/objects to solve heterogeneity and ease-of-use problems. This should also make access to RMIE and other databases available to *non-expert* users. Currently, this class of users must go through an analyst to get any database information desired.

The oMIE project involves a multitude of overlapping research areas many of which could potentially involve DCU either directly or indirectly. A major concern is how to build a system capable of uniformly accessing heterogeneous databases without specific user knowledge of these databases. This system must also be capable of accommodating many users with different skill levels and interest areas. In order to make oMIE meet these criteria, integration of the interface and the heterogeneous database access right from the beginning seems crucial. Therefore, we have chosen to focus on creating a high level design for the interface and for heterogeneous database access, since aside from being crucial to oMIE, these areas are also among those that we feel are the most naturally suited to a DCU contribution.

The main focus of our interface design is to produce a set of techniques for users to create a query and to produce a corresponding set of techniques for displaying query results. These goals are being addressed through the development of an interface called MultiView. It is based on a context-driven approach to building user views. MultiView uses knowledge about users and application domains to provide a uniform data retrieval interface with customized views tailored to specific users and groups of users. This includes knowledge about different applications domains and about various users and classes of users. Users, including non-experts, will be able to build queries relevant to their application domain, in a terminology they are familiar with. The user will start building a query within a a broad context and then MultiView will use all the information known about the user and the context, to guide the user to the query actually desired. This will allow users to quickly focus queries, without having to learn a lot of database terminology. Queries built with MultiView will be passed to oMIE in IQL (intermediate query language).

The oMIE system will provide heterogeneous database access using a community of coarse-grained agents. The main focus of our heterogeneous database access work will be to design the most crucial mechanisms and policies required to enhance the current capabilities of the Coarse-grained Agent (CGA) system and oMIE. Persistence and fault tolerance mechanisms must be built into the Coarse-grained Agent system. Mechanisms must be designed to allow CGAs to efficiently name (identify), locate, and communicate with other CGAs in a large network environment. Security and authentication mechanisms must be built into the CGA paradigm and oMIE. The goals of building efficient naming and communications capabilities into the CGA paradigm are being addressed in an effort to implement a prototype Unix version of the system. The goals of building persistence and fault tolerance into the CGA paradigm are being addressed through the design and implementation of an algorithm for the recovery and continuation of failed agent discourses.

Status

User Interface - An initial design for MultiView has been finished. The design includes an example of how an oMIE user would form database queries using MultiView. The next phase of the MultiView project will be to implement a prototype, including an extensive sample application.

Heterogeneous Database - A report surveying heterogeneous database and database security problems was produced. A Unix implementation of the CGA paradigm is underway and nearing completion. This implementation attempts, in part, to address the problems within this report, particularly those which are not dealt with in previous CGA implementations. Analytical work on the recovery algorithm for agent discourses has been completed and will be verified within the Unix version of the CGA system.

References

Novak, M., King, R., Laufmann, S. C., and L. Thompson, *MultiView: A Case Study in Context-Driven Approaches to Heterogeneous Database Interfaces.*

McIver, W. J. Jr., *Heterogeneous Database and Database Security Problems: Brief Overviews.*

McIver, W. J. Jr., *Designing Recovery into Course-grained Agent Discourses.*

Voice Dialog Design Environment

Participants

U S WEST:	Mike King, Lynda Bain
CU:	Tammy Sumner, Susan Davies Peter Polson, Gerhard Fischer

Related Projects

U S WEST: Voice Dialog Systems (Mike King)
CU: Critics: An Approach to Knowledge Based Human Computer Interaction (G. Fischer and A. Lemke),
Design Rationale (G. Fischer and A. Lemke),
Multifaceted Architecture for Design Environments (G. Fischer),
Cognitive Walkthrough (C. Lewis and P. Polson),
Programming Walkthrough (C. Lewis and B. Bell)

Project Summary

U S WEST currently has a need to streamline the design and development of voice dialog applications. The current design environment used by the Human Factors group consists of unintegrated Macintosh applications. Several applications are used because no single application fulfills the task requirements. Much redundant information is required in maintaining designs in several different forms and applications. Design specifications often need updating which requires modifications in many different databases and applications. Usability is a major concern and challenge in voice dialog application design. A large component of the design process involves prototyping and testing the proposed interface. Prototyping is essential because current design tools do not help designers to bridge the medium gap. The medium gap is unique to voice dialog design and means that it is difficult to anticipate what the final product will sound like when looking at a paper-based design representation.

STATUS

We have constructed a prototype voice dialog design environment. The voice dialog design environment, as it stands today, is a sophisticated construction-kit approach that integrates functionality previously distributed between separate design tools. It provides an on-screen gallery of voice dialog design units, such as menus and prompts, and a work area for design construction and simulation. At any time, the behavior of the design can be simulated. Design simulation consists of a visual trace of the execution path combined with an audio presentation of all prompts and messages encountered. This simulation component helps the designer to bridge the medium gap between the visual representation and the audio artifact.

Currently, our work is focussed on taking the design environment beyond the prototype stage such that it can be evaluated in a naturalistic setting; i.e., used by U S WEST voice dialog designers. Specifically, we are:

- Enhancing the design language to allow designers to specify more complex conditional clauses such as those typically found in existing voice messaging services.

- Providing high quality audio so that the environment may be used for participatory design and usability testing.
- Creating a catalog component containing designs of existing voice dialog interfaces to support design by modification. The first catalog entry the designers have chosen to create is the interface design of the current U S West residential Voice Messaging Service.

References

Sumner, T., Davies, S., Lemke, A. C., and P. Polson. *Iterative Design of a Voice Dialog Design Environment.* Poster to be presented at CHI '92 - Conference on Human Factors in Computing Systems, Monterey, CA, May 3-7 1992.

Repenning, A., and T. Sumner. *Using Agentsheets to Create a Voice Dialog Design Environment,* University of Colorado, Boulder, Dept. of Computer Science Technical Report #CU-CS-576-92, Jan 1992.

Appendices

A Characterization Framework for Visual Languages. J.D. McWhirter and G. J. Nutt,

Designing Recovery into Course-grained Agent Discourses. W. J. McIver, Jr.

Heterogeneous Database and Database Security Problems: Brief Overviews. W. J. McIver, Jr.

Inventing New Information Representations through Task Analysis. N. Wilde and C. Lewis.

Iterative Design of a Voice Dialog Design Environment. T. Sumner, S. Davies, A. C. Lemke, and P. Polson.

MultiView: A Case Study in Context-Driven Approaches to Heterogeneous Database Interfaces. M. Novak, R. King, S. C. Laufmann, and L. Thompson.

Using Agentsheets to Create a Voice Dialog Design Environment. A. Repenning and T. Sumner.

GROUPS INVESTIGATING COMMON RESEARCH THREADS

User Interface Issues Group

John Reiman

Affected DCU Projects: all

Participants: John Rieman (all projects), Susan Davies (voice dialog), Nick Wilde (war room), Jeff McWhirter (SPS), Clayton Lewis

Research Thread Description: User interface issues lie at the heart of the DCU effort. The project's goal is to provide end users with facilities that they can use "directly," i.e., without the assistance of specialist intermediaries. This requires more than a good interface added to a powerful underlying system: it requires that the underlying system itself be structured for usability.

Two related user interface issues cut across all the projects. First, the user interface must present data in a form that matches the user's existing mental picture of what the data represents. For example, if the SPS user typically imagines the effects of decisions flowing in a graph-like model, the interface should present a similar picture, rather than forcing the user to reconceptualize the data in spreadsheet terms. A difficulty with several of the projects is that different user groups conceptualize data in different ways.

The second issue, related to the first, is that DCU systems must provide immediate and salient feedback showing the effects of the user's actions. This supports what Fischer's group refers to as "the situation talking back." It allows the computer to be an interactive tool, much like a whiteboard, that can become the central focus of group discussions and decision making.

Status: The DCU project's early focus on users puts us in an excellent position to address the interface issues described above. Past and ongoing research by CU participants is also pertinent. Providing a good, high-level match between the user's model of a situation and the model presented by the system has been an ongoing concern of Fischer's group, while Lewis and Polson's recent work on the cognitive walkthrough provides a methodology for identifying lower-level mismatches between user goals and specific interface features.

Heterogeneous Database Access Group

William J. McIver, Jr., Michael Novak

Affected DCU Projects: All

Participants: Bill McIver, Mike Novak, Steve Laufmann

Research Thread Description: The need for heterogeneous database access was recognized as a requirement that most of the case studies in the DCU project have in common. To minimize duplication of effort in this area, it was decided that an activity be carried out to develop a common set of design guidelines and software component specifications for heterogeneous database access. This effort is to take place in parallel with the individual case study activities.

The CU resources for this effort are Roger King, Pam Drew, William McIver, and Michael Novak. The US WEST resources are Pam Drew, Ed Freeman and Steve Laufmann. The US WEST liaison will be Steve Laufmann.

Time Line & Deliverables: The proposed time line for the heterogeneous database access group effort is summarized below:

Stage 1: Define Heterogeneous Database Access Requirements

During this stage, the heterogeneous database (HDB) access requirements for each DCU case study will be determined. This will entail in-depth technical discussions with each DCU case study team and the maintainers of the actual database management systems they propose to access. This will be an iterative process.

The duration of this stage is estimated to be four (4) months.

The deliverable for this stage will be a detailed document consisting of guidelines and options for designing heterogeneous database access into DCU case study systems.

Stage 2: Develop Individual High Level HDB Access Specifications

During this stage, each DCU case study team will proceed with the development of their high level specifications, consulting the HDB document when appropriate.

The duration of this stage is estimated to be three (3) months (based on the oMIE portion).

The deliverables for this stage will be design documents from each case study team consisting of the high level specifications for the HDB access aspects of their projects.

Stage 3: Identify Commonality

During this stage, the design specifications produced in the previous stage for each case study will be examined to identify areas of commonality in the HDB access specifications. The intention here is to gather information sufficient to propose a set of software components and tools that can be used for more than one DCU project.

The duration of this stage is estimated to be two (2) months to propose a set of common HDB access software components and tools, plus three (3) months to produce high level specifications for each.

The deliverables for this stage will be a set of high level design specifications for each entity in this set.

Publication Plan: Each deliverable discussed above will be published as CU and US WEST technical reports. When appropriate, extracts from each technical report will be produced for journal, conference, and workshop publications. In general, the experiences from each stage discussed above will render additional ideas for publications.

4. NEXT STEPS for the CU/USW COLLABORATION

In addition to pursuing the case studies described above, the Collaboration will be addressing further activities in the second half of 1991. *Publication* is a key objective, both as a means of establishing useful communication with the wider research community and as a requirement for the Collaboration to meet the professional goals of the participants. We hope to prepare at least some reports of the work for submission to conferences in Fall 1991. Plans also include a USW/CU *workshop* on DCU, in which outside speakers would be invited to present their work to the US West and University audience and to be briefed on the work of the Collaboration.

Finally, as pointed out above, *developing new projects* is an important activity to be carried out in parallel with the projects now underway. Experience early in the Collaboration showed that developing a project from scratch, without the benefit of an already existing Advanced Technologies project, takes a certain amount of effort but a greater amount of time. Establishing appropriate liaisons outside of AT can, in general only be initiated by AT participants, and finding the right time, place, and people for discussions is challenging. Since the long-term impact of the Collaboration depends on developing additional projects that reach out beyond those already undertaken, we plan to overlap this liaison development with our ongoing work.

APPENDIX I

PROJECT DETAILS

PROJECT 1

Voice Dialog Design Environment

Susan Davies
Tammy Sumner

Problem Statement

US West needs to streamline the development of its voice dialog applications. The current design environment used by the Human Factors group consists of unintegrated Macintosh applications. Much redundant effort is required to update these disparate information sources. The goal is to develop an integrated design environment for voice dialog applications that will provide intelligent support for the design process.

History / Current Practices

There are currently at least three groups developing voice dialog applications within US West. All three groups employ different design methods. One internal group relies on traditional written specifications. A sales group uses "specially printed gray stickies" as their primary design tool. Mike King, our problem owner, is currently pioneering his own design methodology that attempts to overcome the limitations of both of these methods.

Voice dialog application development follows a traditional waterfall development process. Mike King, whose has the task of designing these systems, works with the marketing department and the customer to elucidate the problem requirements. Next, he engages in the design process. The end goal is to produce a product specification that the customer will approve with an interface that meets US West design standards. This specification is then given to the implementation group. After implementation, the product is tested by a separate testing department. The testing department relies on the specifications to generate test scenarios.

Mike's department develops approximately ten voice dialog applications per year, generating about 50 million dollars in revenue per annum. He is currently working on a very large design which, in its unfinished form, is represented by over 200 states in a state database and a 36 page flow chart. Such a project takes about one year to develop.

As mentioned previously, Mike is pioneering a design methodology that attempts to overcome the limitations of a traditional textual specification. He is using an assemblage of Macintosh applications to generate a variety of views of the design. The three primary views are a structure chart, a state description table, and a phrasal database. Most of the components in these views are named. It is the component names that (logically) link these objects across views. Illustrations of these views are provided in the appendix.

Structure Chart. This view is currently maintained using MacDraw. It is a variation on a flow chart. Mike uses rectangles to represent states. These states offer menus, process input, or manage passwords and time-outs. These states are connected by arrows representing the user's input or internal error traps. Diamonds are used to represent decision junctions.

State Description Table. This table is stored as records in a database - one record per named state. Stored are the state's inputs, outputs, and long and short prompts.

Phrasal Database. This is also stored as one record per named state in another database. Associated with each state is a particular phrase and the phrase attributes. Phrase attributes include: whether it is for the long or short prompt, which menu item, and whether that menu item is dynamically available. These phrases are concatenated into prompts at run-time in the final product. There is a lot of phrase redundancy in prompts. For storage efficiency reasons, prompts are broken into these phrases which are stored separately. In his current design, there are 2300 phrasal entries in this database.

Some of the problems Mike as cited with his methodology are the following:

> 1) There is a lot of *redundancy* between the different views - both in terms of data and effort. A given state is represented in both the structure chart and the state description database. Any changes to this state must be carried out in both views separately and this involves using different Macintosh applications.
>
> 2) A lot of effort is spent trying to guarantee *consistency* between the different views. A change in one view requires manual updates to all other views. When a prompt is changed in the structure chart, Mike must verify that all necessary phrases are present in the phrasal database.
>
> 3) Each of the three views can get quite large - a 36 page flow chart is difficult to comprehend. With the current tools, there is no way for the designer to generate *dynamic perspectives* of subsets of one of these views. For instance, Mike would like to see only the error flows in the structure chart or only the back-up / cancellation flows.

Mike's primary desires for an intelligent design augmentation tool are:

> 1) assistance from the tool in maintaining consistency between views.
>
> 2) support for dynamic perspective generation.
>
> 3) support for design simulation. Namely, Mike would like integrated support for voice and graphics. Currently, he is designing in a purely graphical world and has no way to actually hear how the voice dialog application sounds. Especially, he needs to hear how the concatenated phrases sound in the final prompts. He feels this would be immensely helpful for preliminary user testing.

While Mike King is our primary client for the design tool, other possible clients include the implementation and testing groups, the sales group, and ultimately could include end-users by allowing them to design their own applications.

Challenges for the Design Tool

One of the primary challenges is that there is no standard, accepted design representations. None of the other voice dialog designers use Mike's representations. This is the first time such representations have been used and no one knows whether these views will be accepted by the other groups involved in voice dialog development; namely the implementation and testing groups.

The ideal environment would support large designs; i.e. those containing large numbers of states and phrases. Another, more immediate challenge is overcoming proprietary restrictions. For Mike's manager to feel comfortable about Mike sharing particular designs and design decisions with us, all involved researchers need to sign non-disclosure forms. Even with this precaution, Mike is concerned that certain parties within US West will still be hesitant about sharing explicit information.

On a more mundane level, US West would prefer Macintosh-based tools. However, our development environment is Symbolics-based and we really don't have the hardware or software resources for a MACINTOSH development environment.

Interesting Issues

Enforcement versus Support. How much should the system enforce versus support the user interface design guidelines mandated by US West? Legally, US West cannot impose these guidelines on customers. How should the system track exceptions to these guidelines taken by the designer?

> Possible scenario--develop system that provides design guidance consistent with in-house guidelines. The designer would have the ability to ignore/override the guidance, but would be warned of possible negative implications. Ideally, the system would keep track of instances where guidance has been ignored, in hopes of maintaining consistency throughout the application.

Design Representations. What design representations should the system support? The three views used by Mike King? Others? There are no standard design representations for voice dialog application development so the system probably shouldn't lock-in a particular set of distinctions.

Knowledge Bases versus Databases. How is design knowledge stored in the system? How are underlying components managed within the system? How is view consistency achieved?

Augmentation versus Replacement. How much shall the tool rely on the designer to provide expertise? Is there any place for an expert-systems based replacement approach?

Understanding the Practice and Organizational Implications. What are the practices we are trying to enable? Are we simply augmenting the designer's activities? Are we trying to promote a less rationalized (waterfall) development process? Are we trying to promote more user interaction in the development process? Are we trying to empower end-user in designing their own applications?

End-User Modification. How can end-users add design knowledge? How can end-users specify dynamic perspectives? How can end-users create new design representations?

Participatory Design. What methods can we use to enable participatory design between us and Mike King, et al.? How can our design tool promote/ enable participatory design between voice dialog application designers and their customers?

System Architecture. Layered? Multi-faceted?

Understanding the Users. Who are the actual user we are supporting - designers, end-users, other development groups, or all of the above?

Palette of Design Components. What would a palette of voice dialog design components look like? Is the 'palette' approach appropriate in this context?

Scenarios

Scenario #1 - An Domain-Specific Design Environment

Goal: Adding a feature to CU Connect that allows the user to find out the course rating for a particular class.

The designer is initially presented with a collection of icons representing particular design case histories. Our designer, Mike, is interested in extending the current CU Connect features. Mike double-clicks on the CU Connect design icon and is presented with the (default view) top hierarchical level of the structure chart.

The top view shows only a state representing the possible action codes the user may choose from with output transition arrows terminating in a resultant state name. Since only the Boulder campus will support this new feature, Mike double clicks on the resultant output state labelled "Boulder".

This causes the next hierarchical structure level to be displayed illustrating the possible term codes for the Boulder campus. It's been several years since Mike has looked at the design so he is content to step through the various design levels. He double-clicks on an output state labelled "get password". This brings up a structure chart covering the states for getting the student number and PIN code. Mike double-clicks on an output state labelled "accept - main menu".

Finally, he is at the structure level he is looking for. The state describing the main menu choices is shown:

2	ADD
3	DROP or DELETE ALL
5	LIST SCHEDULE
6	REQUEST A PRIMARY
8	EXITS
*	CANCEL

The numbers 1, 4, 7, 9, and 0 are available. Mike decides to use the number 0 for this new feature. He chooses "Add New State Transition" from the *Structure* menu. A dialog box appears and Mike enters the required input string "0" and a name for the resultant output state, "ratings". He clicks on "DONE". A critic dialog box appears informing Mike that: 1) using the number 0 for a menu item violates standard US West user interface design guidelines; and, 2) this would result in a menu with 7 options when the recommended length is 6. Mike can't remember why he is not allowed to use the number 0 so he presses the EXPLAIN button. A dialog box appears with the explanation, "0 is reserved for operator assistance only." He chooses to ACCEPT this criticism; the explanation disappears and the input field in the "Add State" dialog box is now blank.

Mike still has one critic message to deal with. He doesn't feel that splitting up the 7 items into two levels of menus would be a good design choice for this particular application. He chooses to REJECT this critic message. The critic box goes away and another dialog box appears asking him to enter a textual explanation for why he is rejecting this design rule. Mike enters his reasoning and presses OK. All explanations of why exceptions to standard design rules are taken are added to a special area in the specifications document.

Mike enters the number 4 for the required input and presses DONE. The new state transition is added to the structure chart.

Mike double-clicks on the "ratings" state label; the next level of structure chart consists only of a blank rectangle. He chooses "Show States" from the *Show* menu. A dialog box appears, Mike accepts the default hierarchic level of "current" and enters the string "*" under required input for resultant state. A new window overlays the structure chart showing a 'bird's eye view' of how states "2" and "3" both transition on "*" to a resultant state labelled "Get Class Code". This look promising. Mike selects the resultant state and chooses "Show Documentation" from the *Show* menu. A dialog box containing the documentation for this state appears. It says, "This state and its resultant transitions prompt the user to enter a class code. The code is verified before returning." This is indeed the behavior Mike wants to add after the "ratings" state so he chooses to "Copy Entire Branch". He goes back to the structure chart showing the blank rectangle and pastes this new branch.

Next, he chooses "Add New State Transition" and enters "#" for the required input and "ratings verification" for the resultant state name. This new transition is added to the current structure chart. Mike double-clicks on the state name "ratings verification". The next level contains a blank rectangle shape. From the *Structure* menu, Mike chooses "Attach Function". A dialog box appears and Mike enters a function written in a domain-specific programming language,

```
If "ClassCode" = 0,
        then goto state "ERROR" with prompt "noClassCode"
        else goto state "Accept Ratings".
```

He chooses DONE and the syntax of his expression is verified. A message appears stating that prompt "noClassCode" is already defined. Mike chooses to USE the current definition and the dialog box disappears. An ERROR transition and an "Accept Ratings" transition have been added to his "ratings verification" state. Mike double-clicks on "Accept Ratings" and the next level of structure chart appears containing a blank rectangle.

Mike attaches a function to this state using "Attach Function" in the Structure menu:

```
LOOKUP "ClassCode" in Class.Database ("ClassName", "ClassRating")
```

Next, Mike chooses "Attach Prompt" from the Structure menu. A dialog box appears and Mike enters the following prompt:

Class "ClassName" has a class rating of "ClassRating".

Functions are always evaluated before a prompt is spoken for a given state. Mike presses the "Verify Phrases" button and a message appears:

Prompt contains three defined phrases:

1) *Class "ClassName"*
2) *has a*
3) *of*

Mike chooses to USE these predefined phrases and RECORDs the new phrase - *class rating*. Next, Mike presses the PLAY button to hear how the new prompt sounds. Since this is not a full-fledged simulation and only a preliminary check, the variable names are simply replayed as the spoken word *variable*. The *of* phrase is recorded in a different voice than Mike's and the two phrases *of* and *class rating* concatenated together sound strange. Mike chooses to not use the predefined *of* phrase and records a new phrase, *class rating of.* Mike presses DONE and the phrasal database is updated to reflect these additions.

Mike now wishes to simulate his design. He returns to the structure level showing the main menu where he had added the new menu item "4". He chooses the *Simulate* menu and a dialog box containing a telephone key pad appears. Mike presses the number "4" and the structure diagram advances to the next level. He enters a "*" followed by a test class code of "99999". The structure diagram advances to the "ratings" state level. Mike enters the "#" key and an error message appears - "No field named "ClassRating" exists in Class.Database". The error was generated when the simulator tried to perform the lookup function attached to the "ratings" state. Mike chooses "Show attached function" from the *Show* menu and double-clicks on "Class.Database". A dialog box appears showing the stub database created for testing purposes. Mike adds a new field, ClassRating, and some test data to the database.

Mike chooses "Continue Simulation" from the Simulate menu and he hears his new prompt spoken, complete with variables bound to the appropriate spoken dialog. Mike pulls down the *View* menu and chooses to look at the "State database". Sure enough, the system has automatically generated state entries consistent with the structure chart. The system also performs a similar function for maintaining the phrasal database.

Scenario #2 - End User Programming

Clayton Lewis

I propose that in addition to exploring the knowledge-based design environment approach to this application that we also consider end-user programming approaches. Can we create tools with which Mike King could create his own design environment, rather than having us (or some other support group) create one?

I am not suggesting this because I think the knowledge-based design environment approach is not the right one, but rather because I think we should find a way to explore the envelope of end-user programming, and this seems like a good place to start.

I will outline an approach using concepts from an end-user programming system I am cooking up, called Formulons (at least for now). But somebody (like Ed for example) might want to take a crack using other approaches to end-user programming.

The Formulons system is a kind of generalized spreadsheet. (Some readers may have heard some discussion of a system called Computons. Formulons is a new name for this, chosen to get around the fact that "computons" is obscene in Spanish.) Values in Formulons have associated formulae which refer to other objects; when a value changes any dependent values are recomputed. Values are placed in a 3D space, and can be stuck together to represent complex objects. They can also be connected by links. (The Formulon design borrows ideas from Lemke and Gance, Repenning, and Hudson, as well as the precursor NoPumpG system).

Here is a sketch of how the functionality needed for the case study might be provided. The basic representation of a dialog would be a network of nodes and links. Having a 3D space to work in might be convenient in laying out paths.

Selective views would be created by shrinking nodes and links that are not part of a selected view. This could be controlled simply and not very flexibly, by tying the size of objects to one or other value associated with some class to which it belongs, or more flexibly by tying the size of an object to a formula which the user could create to represent any desired dependency.

The phrasal database would be represented by a collection of values. A formula would be written that would take a prompt as input and compute a decomposition into phrases in the database and any remnant.

Getting the dialog to play would rely on functions with side effects in formulae. Functions to place prompt text in a specified spot and to produce synthesized sound output would be provided.

Providing design criticism would be interesting to attempt, but might get beyond plausible end-user programming. One issue is how to refer to any parts of a dynamically built structure in a formula.

PROJECT 2

The oMIE Project

William J. McIver, Jr., Michael Novak

University of Colorado
Department of Computer Science
Boulder, Colorado 80309-0430

1. Overview

oMIE is an application of the Coarse-grained Agent system to real world problem solving involving the RMIE database. The Coarse-Grained Agent (CGA) system is the implementation of an agent metaphor for distributed problem solving. It is hoped that this metaphor can be used to achieve virtual re-systemization of US WEST databases and applications by providing a means to *wrap* them with agents/objects to solve heterogeneity and ease-of-use problems. This is known as surround technology.

U S West currently operates a very large database known as RMIE. Access to RMIE is largely limited to a few *analysts*. Since end users must go through these analysts for RMIE data, the turnaround time for reports is slow. Also, data is often redundant and inaccurate.

The general operation of oMIE is as follows:

Step 1.

The user selects an area of expertise that will act as a reference point in interacting with oMIE. This is known as a *context*. Examples of contexts are: revenue, marketing, services, and international information. A set of agents appropriate for the selected context are located at this stage. An agent communication language corresponding to the selected context is used at this stage.

Step 2.

An agent is selected from among those identified in the search.

Step 3.

The user formulates and enters a query by selecting and entering values into various fields of the interface. For example, the user may be asked to specify the region type for which data is desired (e.g. LATA, Metro-area, or state).

Step 4.

The resulting query is displayed on the interface. The query is displayed in both the English and agent languages. For example, in the revenue context the revenue data access language (RDAL) is used by the agents. The message in the chosen agent language is then sent to the agent selected in Step 2. for processing.

Step 5.

The query results are received by the remote agent. A brief summary of the records that were found, if any, is displayed by the user's agent. The actual data is cached in the responding (remote) agent's memory. The user then has the option to print or save the data, or to create a subset of it with yet another query.

Step 6.

The user then has the option to print or save the data, or to create a subset of it with yet another query. When the user requests one of these options, the data is transferred from the remote agent's cache to the appropriate destination.

Goals

Success can be measured in the oMIE project through: improved accuracy of data and reporting, greater access with less knowledge, faster turnaround time, a greater number of users accessing RMIE, and eventually, an application of the technology to a variety of US WEST databases.

The oMIE project involves a multitude of overlapping research areas many of which could potentially involve DCU either directly or indirectly. A major concern is how to build a system capable of uniformly accessing heterogeneous databases without specific user knowledge of these databases. This system must also be capable of accommodating many users with different skill levels and interest areas. In order to make oMIE meet these criteria, integration of the interface and the heterogeneous database access right from the beginning seems crucial. Therefore, we have chosen to focus on creating a high level design for the interface and for heterogeneous database access, since aside from being crucial to oMIE, these areas are also among those that we feel are the most naturally suited to a DCU contribution.

The potential research areas are discussed in the next section. This is followed by a description of the specific research scenarios we intend to pursue and our conclusions.

2. Research Areas

The oMIE project involves a multitude of overlapping research areas many of which could potentially involve DCU either directly or indirectly. We have chosen to examine three that we feel are the most naturally suited to a DCU contribution.

2.1. User Interface

Currently, the interface allows a user to pick a context to work in and then provides an agent communication language for the selected context. The user's agent than looks for agents that speak this language. The user query is sent to the first agent on this list, then the other agents if necessary. When the query is completed, the results are sent back to the user agent, then displayed to the user.

There are a host of interface issues to be looked at here. These include correct context selection, search domain specification, the user language(s), and interface transitionality[3]. Some of these issues have been partially addressed and some have not been addressed at all. In the following sections we would like to identify what we feel are the major problem areas that need to be attacked.

Context Selection

When specifying a query, the user needs to select a context. This gives the system a starting point when looking for an agent that can retrieve the desired data. In reality, however, the user may not know the context of the query. For example, is the information desired stored in the context of *marketing* or *international information*? The system should be able to determine some of the context from the query, or at least help the user to determine it.

The goal here should be to find a way for the system to determine context with as little help from the user as possible (unless, of course, the user wishes to specify it).

Search Domain Specification

The search domain of a query is closely related to the context. Currently, an agent is selected from those that are identified as appropriate to the context. However, as the agent set becomes large, this problem get more and more difficult. There needs to be a mechanism that allows the user to specify the search domain, without having to be very familiar with what is stored where. The system should also aid in this by having a set of heuristics for selecting a search domain automatically, should the user have no idea of the domain.

This problem involves both the interface issue of finding the appropriate domain and the underlying issue of creating some kind of cross referencing that links contexts with domains.

User Language(s)

The user language is currently context specific. When a context is chosen, an agent communication language appropriate to the context is selected. This could present problems if the context is not known by the user or if the query extends across more than one context. The issue of how to combine these languages or whether to use one global user language (with subsets for different contexts) instead is an issue that needs some looking into. It also need to be determined whether the language(s) will be SQL like [17], graphical [6,13,14], some combination thereof, etc.

Aside from determining the user language(s), the underlying issue of what agent language the user query will translate into must also be addressed.

Interface Transitionality

Another area closely related to the query language(s) is interface transitionality. The queries a novice creates will generally be quite different from those which an expert creates. One approach to this is a language that is functionally complete, allowing the expert to do any query the system is capable of, while the novice would only use a small subset of the simpler commands. Another, probably more elegant, approach would let the user select a level or perhaps select from a set of interfaces.

Not only does the full range of queries from novice to expert need to be provided for, but the transition as a user becomes more experienced should be as painless as possible[16].

2.2. Agent Development Environment

Currently, all oMIE agents have been created manually and all connections between agents and the entities they serve have been hard coded. An agent development environment needs to provide some tools for easily creating agents and connecting them to the appropriate entity. This process should involve as little coding as possible and provide a reasonable amount of agent consistency.

The specific areas that need to be addressed here are the agent data model(s), agent creation and evolution, user interface creation and evolution, and the role of the knowledge base. Once again we would like to point out what the specific problem areas we wish to address are.

Agent Data Model

Although this is not an obvious DCU issue, the agent data model is so important to the rest of the agent development environment that it needs to be very carefully designed. There are several main issues here. First, is one data model appropriate for all agents, or should there be different data models for different types of agents. Second, if different agents use different data models, how is the data translated from one model to another. Third, how is database data translated into agent data and vice versa.

Aside from the data model(s) and the translation protocols that need to be designed, a heuristic for creating new protocols (such as when a new type of database and corresponding agent is added to the system) also needs to be designed.

Agent Creation and Evolution

Tools need to be provided that allow for the easy creation of new agents and that allow agents to evolve as new information is added to the system. These tools need to be as automatic as possible. If, for example, every new agent addition involves a major amount of coding, then not only will the cost of adding an agent be extremely prohibitive, but it will be almost impossible to maintain system consistency. Along with this, tools to connect new agents to the databases they will be serving also need to be provided. This becomes quite important when a new database is added to the system, especially if the database is of a new type.

The major tasks here will be to provide tools for adding agents of a known type and to provide tools and heuristics for creating agents of a new type without sacrificing system consistency.

User Interface Creation and Evolution

As the system evolves, there will undoubtedly be a need for both new user interfaces and a change in the existing interfaces. In order to maintain system integrity, some constraints must be enforced when creating or modifying user interfaces. Tools for creating and updating interfaces need to be easy to use and they need to automatically enforce these constraints. Tools to connect new interfaces to their user agents must also be provided. Also, interface evolution that is directly related to data or agent evolution should be as easy and as automatic as possible. For example, if an agent used to return all data as single precision now returns it as double precision the interface needs to adapt.

The main problem here is to make interface creation and modification as easy as possible while making violation of system integrity as difficult as possible.

Knowledge Base

Another issue that is directly related to agent creation and evolution (as well as database creation and evolution) is the knowledge base [18]. Not only does it need to be easily accessible and updatable, but there needs to be some process for determining whether information goes into the knowledge base, gets built into a specialized agent [1], or goes into a database. Also, will the knowledge base ultimately be global or distributed? If it will be distributed, how do we determine what is the appropriate location to add new knowledge to.

Since the structure of the knowledge base directly affects the creation and evolution of the other components of the system, these issues need to be looked at simultaneously with the issues mentioned above.

2.3. Heterogeneous Database

The community of coarse-grained agents that will make up oMIE will essentially form what is known as a heterogeneous database management system (HDBMS). A HDBMS consists of a collection of autonomous DBMSs distributed throughout a network environment [12]. We call these DBMSs constituents of the HDBMS. Data models and software systems may be different for each constituent. However, in this type of system the heterogeneity of the underlying data models, query languages, schemata, and transaction management is disguised. A single view of the entire collection of DBMSs is presented to users. A HDBMS also insures that queries and updates are consistent across the entire system.

Research into HDBMS falls several major categories: schema integration, query processing, and transaction management and recovery. An additional research issue that is not exclusive to HDBMSs is database security. Each of these issues is discussed below.

Schema Integration

In a HDBMS there must be a mechanism to map the user views of the database into the individual local schemata of which the HDBMS is logically composed. Schema integration includes the translation of data models, names, representations, and syntax into common entities. If the constituent DBMSs employ different data models, each must be mapped onto a common data model. If several local schemata employ different names and representations (e.g. attribute names and relation definitions) for the same conceptual entity, they must be translated into common names and representations. Finally, if several local schemata employ different data types for the same conceptual entity (e.g. floating point vs. fixed point, or 20 byte vs. 40 byte fields), they must be translated into common data types [12, 19].

Query Processing

A query in a HDBMS is composed of one or more local queries that are executed on the HDBMS's constituent DBMSs. The aspects of query processing in a HDBMS include: the distribution, translation, and execution of query operations at the HDBMS level into appropriate combinations of subqueries at the constituent DBMS level; managing the retrieval of results from multiple sites; and the optimization of queries [19]. The distribution of subqueries to constituent DBMSs can either be handled by a single distributed query manager, or applications can be made responsible for sending subqueries to the appropriate DBMSs. In general, distributed query optimization should make use of information such as disparate communication bandwidths, processor speeds, and work loads among the constituent databases. Also, distributed query optimization approaches may involve exploiting parallelism and moving data
to certain locations before processing [19].

Transaction Management and Recovery

A transaction in a HDBMS is composed of one or more local transactions that are executed on the DBMSs that make up the HDBMS. A HDBMS transaction must guarantee the transaction properties of atomicity, durability, and isolation on a global level. Atomicity and durability ensure that either all operations within a transaction complete and commit, or none of them do. This is more complicated in a distributed environment where a local database failure could leave a datum inaccessible, and therefore, unrecoverable for an indefinite amount of time. Isolation ensures that the concurrent execution of transactions is equivalent to some serial schedule of transactions. In general, the transaction manager of an HDBMS must provide commit and recovery mechanisms and ensure serializability of its transactions on a global
level [19].

Database Security

Clearly, some groups of users must be restricted from explicitly requesting certain data (e.g. salary information). Users should also be prevented from *inferring* restricted information in spite of other security constraints. For example, in spite of being restricted from obtaining individual salary information, a user might be able to construct a query using aggregate information to approximate an individual's salary (e.g. find the average salary of all employees of age 43 in department ABC). This is known as the *inference problem* [10, 11]. Several types of object level security constraints must be supported by a DBMS to prevent or minimize these threats [20] *(NOTE: The Inference problem has not been solved. Approaches to minimizing it have been proposed. Yet they are generally considered unsatisfactory [10, 20].).*

In general, object level security constraints dictate the values and contexts under which information can be read from a database. For example, retrieving attribute values for NAME and SALARY together might be made a restricted operation. Security constraints must be enforced during data and schema updates as well. Clearly, a user must not be able to write information into a part of the database that requires a higher security level than they are authorized for. It is not so clear, however, what the policy should be for users with high security levels relative to the part of the database they are updating [20]. The security levels must either be made to match the lower one, or the data must be replicated for both security levels. Finally, in systems that support inheritance, users must not be able to define new, unclassified schemas/classes that inherit classified information.

Many object level security constraints can be violated without additional security constraints at the query processing level by posing multiple queries. For example, suppose a context-based constraint exists that prevents the displaying of NAME and SALARY together. A user might be able to find the correct NAME/SALARY relationships by posing two queries: one for NAME and ADDRESS, and a second for SALARY and ADDRESS. A query processor would have to maintain security related information between queries to guard against this type of threat. Enforcing query processing level security in a HDBMS further complicates this problem for several reasons. First, the individual security constraints of the constituent DBMSs must be integrated into a global representation for use by the HDBMS's query processor. Second, constraints that span more that one constituent database can possibly be violated by accessing each database locally (i.e. not through the HDBMS interface). Finally, a user could be authorized at different security levels for the different constituent databases in the HDBMS.

2.4. Systems

There are a variety of computer *systems* policies and mechanisms that the CGA system will require for its operation. Many of these issues do not fall exclusively in the domain of database issues discussed above. Most fall within the operating systems and data communications research areas. The specific mechanisms and policies of interest are discussed below.

Cache Management in Coarse-grained Agents

During the operation of applications of the CGA model, agents are required to cache information that they have retrieved. At some later point in time an agent may be asked to deliver a specific piece of information that was cached earlier; however, an agent may never be asked to deliver the result of a request. In the interim, this agent may accept and service other requests, whose results must also be cached. In an environment with large numbers of agents and requests between them, serious consideration must be given to the cache policies [2].

Broadcast (or multicast) Algorithms for Locating Agents

During the operation of applications of the CGA model, an agent may be required to search for and identify unknown agents. Searches are to be carried out using some broadcast protocol. In large communication domains, broadcasting messages throughout the entire domain can adversely affect performance. Protocols should be designed to judiciously broadcast messages to appropriate *subdomains* of a community of CGAs. These are called multicast protocols [5].

Agent Naming

A system is needed to effectively manage the name space of agents. This naming system must provide an efficient and secure means of mapping agent identifiers to agents and their locations. Also, a mechanism is needed to assign names to newly created agents [9, 15]. Multicast algorithms can be designed by taking advantage of the nature of a particular naming system.

Intra-agent Scheduling and Synchronization

It may be necessary to have multi-processing internal to coarse-grained agents. This capability necessitates the presence of intra-agent scheduling mechanisms and policies to facilitate efficient and fair execution of processes within agents. Furthermore, the existence of interleaved or parallel processing within an agent requires the presence of appropriate synchronization mechanisms within agents to insure safe and fair access to information shared among intra-agent processes [7, 8].

Fault Tolerance and Recovery

Each coarse-grained agent must be both highly *available* and *reliable*. Each coarse-grained agent must also be able to function correctly in the presence of hardware, software, and operational faults. A framework within which coarse-grained agents can effectively identify and recover from exceptional conditions that occur in their environment must be designed.

Communications Security and Authentication Mechanisms

A community agents that communicates over a shared medium such as a network must be protected against the threat of unauthorized activities performed by *clandestine agents* or other entities on the network. At least two types of threatening agents may be present in such an environment: unauthorized listeners and unauthorized requesters. Agents that receive request must have proof of the origin of the request and that it received the original content of the request [4]. Agents that send requests need proof of delivery and that the original content of the request was delivered. In general, inter-agent communications should be unreadable by any unauthorized entities on the network.

Database Support for Agents

Agents must be persistent and recoverable. A DBMS may be a natural choice for managing the information that an agent manipulates. In this scenario agents would have special support DBMSs available to them at each site which they may be present.

Agent Load Balancing and Subsystem Tuning

CGAs can be mobile. To achieve efficient operation of a large community of CGAs, the optimal assignment of agents and information they access to sites is crucial. Also, it is crucial that the interaction between all of the mechanisms proposed above be examined for potential performance conflicts. This is important for presenting the user with an environment that is free of disruptions and other unpredictable behavior.

3. Scenarios

Based on the research issues examined earlier, followed by some in-depth discussion between the CU and US WEST people involved in this project, we have identified the **interface** and some **heterogeneous database** issues as the most important to oMIE *at this time*. Therefore, we are going to attack the following two scenarios as our contribution to the oMIE project. The process charts for each scenario are given in *Figure 1*.

3.1. User Interface

The goal is to define a set of interfacing rules or standards around which all specific interfaces will be built. Using these standards, we will design a high level specification of the oMIE interface. The interface specified will initially be geared to provide access to RMIE data through the oMIE CGA system. This design will, however, be general enough so that future versions of oMIE can also provide access to databases other than RMIE.

Some tasks that this entails are:

1. Determining what kind of access oMIE needs to provide for both naive and expert users and what form that access should take. This task will involve talking to a variety of US WEST employees to understand the needs of the user community. These employees should include the full spectrum of those who currently use RMIE data, all the way from those who may have no knowledge of how to access RMIE (people in marketing, sales, etc.) to the analysts who get the data for them.

2. The definition of a *covering set* of information contexts, along with their interactions is needed. Experimentation with various approaches to selecting contexts should be done. Also, various levels of contexts need identified and some way of dealing with overlapping contexts must also be addressed.

3. Exploration of a set of interfacing techniques, especially as they relate to varying levels of expertise with the interface and the context and as they relate to varying domains of expertise (e.g., presentation of database concepts (i.e., a CS topic) to an MBA) is needed. These techniques should allow the interface to adapt to the changing sophistication of individual users with respect to tools, contexts, and domains. Also, as much consistency as is possible must be maintained between different portions of the interface.

4. Defining a query language or set of query languages that will be output by the interface. The language(s) must be context sensitive and consistent. Development of a set of user-level concepts relevant to each language is also necessary. For example, the user may be given a graphical way of specifying a request. This request is then translated by the interface into RAQL, an intermediate query language for oMIE.

3.2. Heterogeneous Database

The objective of this scenario is to design the most crucial mechanisms and policies required for the operation of the Coarse-grained Agent (CGA) system and oMIE. In general, each of these tasks will involve interviewing US West database and systems administrators to understand the operating environment of oMIE and synthesizing appropriate techniques from the research literature.

1. Persistence and fault tolerance mechanisms must be built into the Coarse-grained Agent System. This task will involve designing mechanisms for: storing process states, detecting and gracefully recovering from internal CGA error conditions, and detecting and gracefully recovering from inter-CGA communication error conditions.

2. Mechanisms must be designed to allow CGAs to efficiently name (identify), locate, an communicate with other CGAs in a large network environment. This task will involve developing: multicast algorithms, agents that will act as name servers, and

strategies for deploying name server agents to achieve efficient communication.

3. Security and authentication mechanisms must be built into the CGA system and oMIE. Two levels of security constraints are envisioned here. First, database level security to enforce data related security constraints during oMIE query processing activities is needed. This may involve developing a trusted agent to enforce a comprehensive set of database security constraints. Second, communications level security and authorization mechanisms to protect against unauthorized access or disclosure to malevolent or pernicious agents is needed. This may involve using DES, Public Key, or RSA encryption for inter-agent communication along with other techniques such as traditional passwords [4]. US West employees must be consulted on the technical, procedural, and legal aspects of security that are particular to the company.

4. Conclusions

As stated earlier, the main goal of oMIE is to provide users with a wide range of skill levels and areas of expertise easy access to a heterogeneous database environment. In view of this, we feel that the early integration of the interface and the heterogeneous database access is crucial to oMIE. Such an approach to designing a heterogeneous system also will provide some insight into the general problem of providing database access to non-experts. In specific, the two ideas within our work that we see as the most promising are the context sensitive approach to the interface and the incremental integration of heterogeneous databases using the Coarse-Grained Agent System.

Generally, work in providing database access to non-experts only concentrates on the level of experience the user has with the interface and the database. The oMIE interface will also take into account the context within which the user is working and the users expertise in that area. Knowledge of a context will allow the interface to translate queries based on user terminology into database specific concepts.

Using many heterogeneous databases simultaneously can be extremely complicated if users must learn the schemas of all of the databases they need to access, and if they they must be involved in the process of locating and accessing all of these databases to obtain the data they desire. The oMIE system attempts to minimize, and thereby ease, user involvement in these tasks by encapsulating knowledge of database locations, schema, and access methods inside of a community of cooperating agents.

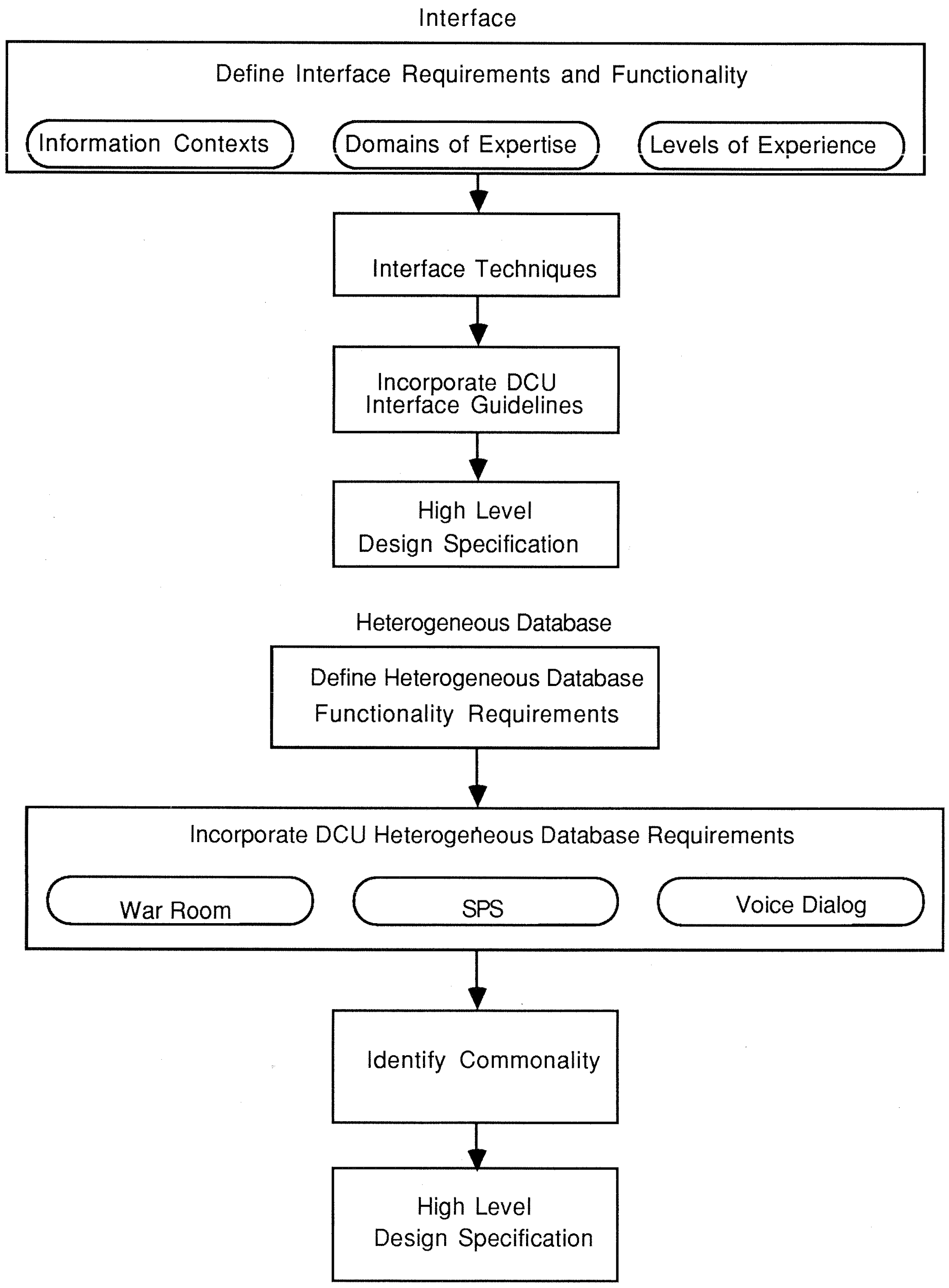
Interface
Define Interface Requirements and Functionality
Information Contexts
Domains of Expertise
Levels of Experience
Interface Techniques
Incorporate DCU Interface Guidelines
High Level Design Specification
Heterogeneous Database
Define Heterogeneous Database Functionality Requirements
Incorporate DCU Heterogeneous Database Requirements
War Room
SPS
Voice Dialog
Identify Commonality
High Level Design Specification

Figure 1. oMIE Design Plan

References

1. S. S. Adams and A. K. Nabi, ``Neural Agents - A Frame of Mind", *OOPSLA Proceedings*, October 1989.
2. R. Alonso, D. Barbara and H. Garcia-Molina, ``Data Caching Issues in an Information Retrieval System", *ACM Transactions on Database Systems* ,15, 3 (September 1990), 359.
3. A. Badre, ``Designing Transitionality into the User-computer Interface", *Proceedings of the First USA-Japan Conference on Human-Computer Interaction*, Honolulu, August 1984.
4. T. Berk, ``Security and Authentication", *Lecture Notes, CS5673*, Boulder, Colorado, April 1990.
5. K. Birman, A. Schiper and P. Stephenson, ``Fast Causal Multicast", *Cornell Computer Science Technical Report Tech. Rep.-1105*, April 1990.
6. D. Bryce and R. Hull, ``SNAP: A Graphics-based Schema Manager", *IEEE Conference On Data Engineering*, 1986, 151-164.
7. A. P. Buchmann and M. Hornick, ``A Transaction Model for a Distributed, Heterogeneous, Active Object System", *Workshop on Multidatabases and Semantic Interoperability*, 1990.
8. A. P. Buchmann, ``Modeling Heterogeneous Systems as an Active Object Space", *Proceedings of the 4th International Workshop on Persistent Object Systems*, Martha's Vineyard, September 1990, 279-290.
9. D. R. Cheriton and T. P. Mann, ``Decentralizing a Global Naming Service for Improved Performance and Fault Tolerance", *ACM Transactions on Computer Systems* ,7, 2 (May 1989), 147-183.
10. D. E. Denning, P. J. Denning and M. D. Schwartz, ``The Tracker: A Threat to Statistical Database Security", *ACM Transactions on Database Systems* ,4, 1 (March 1979), 76-96.
11. D. Dobkin, A. K. Jones and R. J. Lipton, ``Secure Databases: Protection Against User Influence", *ACM Transaction on Database Systems* ,4, 1 (March 1979), 97-106.
12. A. K. Elmagarmid and C. Pu, ``Introduction to the Special Issue on Heterogeneous Databases", *ACM Computing Surveys* ,22, 3 (September 1990), 175.
13. K. J. Goldman, S. A. Goldman, P. C. Kanellakis and S. B. Zdonik, ``ISIS: Interface for a Semantic Information System", *ACM SIGMOD Conference Proceedings* , May 1985, 328-342.
14. R. King and M. Novak, ``FaceKit: A Database Interface Design Toolkit", *VLDB Conference Proceedings*, Amsterdam, The Netherlands, 1989, 115-123.
15. B. W. Lampson, ``Designing A Global Name Service", *Proceedings of the Fifth ACM Symposium on the Principles of Distributed Computing*, August 1986, 1-10.
16. H. Mozeico, ``A Human/Computer Interface to Accommodate User Learning Stages", *Communications of the ACM* , 25, 2 (February 1982).
17. J. Sayles, *SQL spoken here*, QED Information Sciences, Wellesley, MA, 1989.
18. Y. Shyy and S. Y. W. Su, ``K: A High-level Knowledge Base Programming Language for Advanced Database Applications", *ACM SIGMOD Conference Proceedings* , May 1991, 338-347.
19. G. Thomas, G. R. Thompson, C. W. Chung, E. Barkmeyer, F. Carter, M. Templeton, S. Fox and B. Hartman, ``Heterogeneous Distributed Database Systems for Production Use", *ACM Computing Surveys* , 22, 3 (September 1990), 237.
20. M. B. Thuraisingham, ``Security in Object-Oriented Database Systems", *Journal of Object-Oriented Programming*, March/April 1990, 18-25.

PROJECT 3

Decision Support War Room

PROBLEM STATEMENT

US West AT has been tasked to develop a decision support facility (dubbed the "war room") for US West Spectrum Enterprises. This facility will be used to present, integrate, and analyze management information and to provide support for strategic level decision making.

Warren Kuehner, et al., the developers of the war room, need to provide the seamless integration of modelling techniques and data access procedures in a facility that will be used by non-technical, high-level users. This class of users, in particular, has very little tolerance for high training times and little to no experience interacting with modern computer systems. This is combined with a great need for powerful modelling techniques in concert with sophisticated data management and retrieval capabilities. Because it involve enabling computer-naive users to do sophisticated computer-based modelling, the war room project clearly falls under the rubric of "Direct Computer Usage" (DCU).

BACKGROUND

Vic Pavlenko, Vice President of Spectrum Enterprises, is the motivating force behind the war room project. As part of US West's overseas operations, Spectrum Enterprises has a great need to bring together information from different countries and sources in order understand the "big picture" of how the company's overseas operations are succeeding (or not).

In the past, much of this information was obtained by literally travelling from country to country and getting briefed by those responsible, e.g. about marketing operations in Czechoslovakia, or about the regulatory climate in Yugoslavia, etc. The war room project started as a physical facility envisioned where strategic level decision makers could go to have the information they needed brought to them, rather than the other way around.

A physical realization of the war room concept is still very much a goal, and in fact construction of such a facility is underway. However, it has become evident that bringing this quantity of management information together in one place provides some significant opportunities as well as significant challenges. W. Kuehner, et al., are interested in providing new and better modelling facilities as part of the on-going war room effort.

The idea is to provide technologies that would enable a planner to combine their idea of the "physics of the business" with the actual data from all its various sources. In this way, the war room should help to provide a cohesive "big picture" of the state of the world from the planner's perspective. Such a picture will include not only data from various physical sources (i.e. spreadsheets, year-end reports, end of the month financial analysis etc.), but various assumptions about how the world external to US West operates and effects the business. Such assumptions need to be called out explicitly in the model(s) to enable US West planners to take them into account. The planners must take into account the uncertainty involved with such assumptions, and plan for the event that one or more assumptions turn out erroneous. A key concept here is "drill down", (a phrase coined by Apple Computer), in which a user who wishes to may easily access the supporting information for a particular datum - i.e. "drill down" below the substrate of the presented analysis to the bedrock (or quicksand) of supporting information underneath.

There are several different types of modelling software available today, ranging from highly quantitative spreadsheet-based packages to the more qualitative types such as SPS and its predecessor OMNI/MOD. Clearly an important component of the war room project involves not only integrating such modelling packages with the varieties of data available, but also with each other, and with the way the user thinks about the business.

RESEARCH ISSUES

There are many research issues involved in the war room project that appear to fall squarely in the DCU domain. These include:

Data Input/Access:

- Providing easy methods of data input from a variety of sources that may already exist today e.g. spreadsheets, existing databases, existing hardcopy, etc.

- Providing seamless access to many different forms of data

- Providing browsing/organizational capabilities to allow a user to discover:

  What data is available.
  What data supports a particular contention ?
  What data refutes a particular contention ?

- Providing physical connectivity, and data consistency in a distributed information environment.

  The information providers must be using the same information and models as the information receivers.

Modelling

- How can existing modelling software be best used and/or adapted to serve the needs of strategic level decision makers ?

  -- Can we adapt the spreadsheet based / "At Risk" approach to provide more qualitative models, as well as provide better overall support for model building within a spreadsheet.

  -- Can we adapt the SPS approach to provide some of the tools available in a spreadsheet - I.E. provide the ability to do quantitative analysis based on tables of data, etc.

- How do we provide models / modelling software that best allow a business user to input their conception of the world into the model ?

  -- Should we (attempt) to change the user's viewpoint to fit mathematical models ?

  -- Should we attempt to fit models to the user's view of the world ?

SOLUTION SCENARIO (Relevant Technologies)

There seem to be 3 parts to any solution scenario: data management facilities, modelling facilities, and a "glue" or interface module that serves to bring together the data management and modelling facilities.

On the data management side, there are 2 problems that must be attended to: providing convenient means of data input to such a facility, and providing adequate and convenient methods of data access once the information is in place.

For data input current OCR and DIS (Document Imaging System) technology could be used to provide a ready means of getting data that already exists (in the form of office-memos, year end reports, etc.) into the system. Capabilities to read and make use of popular data formats, such as spreadsheet files and word processing files, should also be built into the system. While it is probably not realistic to expect to support every word processor or spreadsheet package in existence, object-oriented methods could be used to provide an extensible system in which new data types can be added with a minimum of effort.

Andreas Lemke pointed out that much of the environmental data - i.e. data that is not specific to US West, may already be found in commercial news and data services such as Dow-Jones, Compuserve, or Genie, and that it may prove beneficial to provide "hooks" into on-line services such as these to avoid unnecessary manual entry of information.

Once the data is input into the system, the relevant questions become those of data access and navigation. Current hypertext (or, in this case "hyperobject") capabilities could be built into such a system so that links between objects and their supporting information can be provided. Indexing via temporal and spatial origin (i.e. did this memo come from Yugoslavia or Czechoslovakia) attributes of the data should be provided at the very least. It is anticipated that the structure of the data will evolve over time, as the user's model of the business evolves and is communicated to the system.

Since the war room will be used to bring together information from geographically separated points, and will be used to provide to high-level management the information necessary to make strategic level decisions, it is vital that the origin of all data be tracked. A concept that may prove useful is the "owner" of a particular piece of information - the person whom, when a question is raised concerning the validity of the information, can be asked to provide reference material and supporting evidence. While the hope is that much of the supporting data will already be in the system, it is quite possible that the high-level user of the war room may ask for supporting information that is not readily available on-line. In that case an "interrupt" should be generated to let the information-owner know that further input is required, and of what form it should take. Once the owner furnishes the requested information, another interrupt will be generated to inform the requester that the new information has been provided.

Much of the data for the war room will be of the variety that gets generated on a repeating basis - i.e. monthly and year-end reports, analyses, etc. Thus, the ability to use existing information structures as templates for adding new information is vital.

Modelling Technologies

On the modelling side, their are two avenues we would like to explore: extensions to current spreadsheet technology, and the use of more specialized causal modelling and risk analysis software, such as SPS .

Refinements to existing spreadsheet technology could provide means for more top-down, structured problem definition than is available in current spreadsheet. While current commercial spreadsheet products are evolving in this direction, we believe better methods for analyzing existing spreadsheet models, and for documenting spreadsheet based models as they are developed, are needed.

SPS is a qualitative modelling tool with clear application to the war room facility. Adapting SPS to more closely fit the war-room model would involve providing data input/output capabilities from the rest of the war room, and providing a common language that would allow SPS to work in concert with other more quantitative business models or perhaps extending the SPS model itself to better handle "spreadsheet" type data (i.e. tables of numbers).

An interesting question to explore is the "fit" between the user's internal models of the business (their sense of "the physics of the business") and the models that tools such as SPS and spreadsheet models allow one to build. The question is one of what models are POSSIBLE vs. what models are NATURAL. It may turn out that the most natural models of US West's business for strategic planners are not captured easily by any existing modelling software. In this case a different, or perhaps hybrid, modelling tool may be called for.

Since the design of a business model in any of these tools is clearly a task-driven activity, it seems natural to adapt cognitive walkthrough techniques to the design of modelling software for the war room. Not only could cognitive walkthroughs provide leverage on building the next generation of modelling software itself, some adaptation of the walkthrough techniques might provide leverage on designing the models themselves so that they are more easily communicated from user to user.

Glue

A third aspect of any proposed solution involves the interface between the data management and modelling aspects of the war room. Clearly, models without data are about as much use as data without a coherent model to use it.

A possible solution here is to use the evolving model as a form of structure for the data itself - thus navigation of the data takes the form of traversing the model in some sense. As the model evolves, the structure of the data in the war room evolves along with it. The concept of "drill down", as mentioned previously plays an important role here. The data should be structured in such a way as to provide a supporting environment for the model - it should be possible to query any assumption in the model and find any and all data available that support this particular assumption (as well as any and all data that refute it). Some of the EUCLID work at the CU, on representing argumentation in such a way as to provide automated support for the argument, may be applicable here.

Some of recent work by Lemke, et al., on issue-based hypertext systems for design, may also be of use in structuring the information in such a way as to make it accessible. Although at the moment this work only deals with highly-structured information, research is ongoing to allow it to deal with more loosely structured and unstructured information such as might be found in the war room project.

USAGE SCENARIO

(NOTE BENE: the following is pure Sci-Fi, included for the purposes of illustrating what a system like the one envisioned may be like. Part of the real research task, obviously, is to come with a vision more firmly grounded in reality than this one.

Dateline: 1 April, 2010

Our intrepid user, I. User, walks into the war room with a furrowed brow and a facial expression that clearly states "I'm worried". For several months now, I. User has been hearing rumors about the upcoming election in one eastern European country, and what it would do to US West's burgeoning cellular telephone service in Eastern Europe if the incumbent party there was ousted by the newly formed Nationalists party. If the Nationalists were able to sway the upcoming election, it may mean the nationalization of all cellular telephone service in this country. What would such a turn of events mean to US West's overall Eastern European operations ? Would they be able to absorb the financial loss ? How about the network itself ? Although US West could conceivably still use the nationalized network, a policy decision had been made years ago not to rely on any non-US West owned wires without have a contingency plan in effect. If this country's facilities were suddenly cut off from them, could US West re-route all traffic through the neighboring countries without interrupting service ?

I. User takes a seat at the war room console. Selecting "Global-Overview: Eastern Europe", I. is presented with a map of Eastern Europe on the screen in front of him. Clicking on the country in question, a pop-up menu gives him several choices of information repositories to look at: political situation, revenues, labor unrest, etc. Selecting "political situation", I. is presented with a newspaper clipping from one of the country's major newspapers discussing the upcoming elections. Obviously, someone in the field office thought this particular clipping important enough to scan into the information base and flag for I. User's retrieval. Scanning the column, I. finds a section marked in red that discusses a recent speech by the leader of the Nationalist party in which she advocates the nationalization of all communications and broadcasting infra-structure. I. clicks on this section, which brings a small window to the front of the screen: A note from R. Jones, I.'s second in command in the field office, noting that he believes the Nationalists have both the determination and capability to carry out their threat IF they are successful in the upcoming election.

Since I. is interested in how the Nationalists could muster enough governmental labor and technical prowess to keep the communications network running if indeed they did decide to nationalize it, I. selects "justification" from a pull down menu after highlighting the relevant text on the screen. A dialogue appears stating that the "no information is available, would you like to generate request ?" I. answers in the affirmative, and an e-mail note to R. Jones is automatically dispatched asking for information to back up his assertions about the Nationalist's capabilities.

Having finished assessing the political situation, and deciding that there is, indeed, cause for concern, I. turns to the financial picture. Back at the map of Eastern Europe, I. selects "revenues" and is presented with a graphed tree of all of US West's money-making concerns in Eastern Europe. He clicks on the box labelled "Cellular Operations - Eastern Europe". The display promptly zooms in on that box revealing that contained within it is a complete causal analysis graph and model of US West's revenue produced by selling cellular telephone services in Eastern Europe. Noting with satisfaction that it is solidly in the black, I. selects a node representing this particular country's contribution to the year-end revenue picture. A small spreadsheet pops up in the lower corner of the screen, detailing the financial picture of US West's cellular services organization in the country. "Heck, that's one of my most profitable areas", I. grouses, as he clicks on the number representing sales of cellular services to international clients with organizational arms in the country. "Its possible, if they do nationalize it, we can keep our international clients by routing through other countries, and only rely on local networks for the final link", I. muses. Another information missing dialogue appears, and I. quickly generate a request to the chief financial officer in the field office asking for detailed information on his international clients.

Getting back to the causal analysis graph of I. selects the node representing the country's contribution, and selects a menu item titled "Suppose...", giving the node a (temporary) value of 0, I. examines what losing this revenue would do to the bottom line. Noting that, the E. European Cellular company is still in the black, even without the this revenue, I. is a little happier. Although I. is still not ecstatic, because the margin of profit is uncomfortably close to the range of uncertainty calculated automatically by the modelling software from assumptions underlying the model.

At that moment, an alarm rings indicating incoming information flagged for I's personal attention. Answering the alarm, I. discovers it is the information from the field office he had requested. The resulting spreadsheet tells him what he had hoped for: fully 75% of US West's revenues in this country are from international clients who could reasonably be expected to move the majority of their operations elsewhere if the Nationalist party did win the election. Going back to the causal analysis graph, I. re-does the "suppose" item to include the revenue from international clients but to specifically exclude the revenue that couldn't be saved if the phone service was nationalized. Now the picture looks brighter. Clearly, the key lies in keeping those international clients if indeed the worst-case political scenario plays out.

The question foremost in I's mind now is "can we deal with the resulting network loss, re-routing through neighboring countries ?" Although network planning and prediction aren't I's particular forte, the network gurus fortunately kept their model of the E. European operations integrated into the war room company model. Going back to the map of E. Europe, I. selects "network model" and is present with a map showing a present (and future) trunk lines in Eastern Europe. Selecting the country in question, I. chooses "suppose" again from the menu and deletes all trunk lines through this country from the network model. The resulting traffic analysis takes a few minutes to settle down, but the end result is that although a severe strain is put on the neighboring countries traffic load, the network could handle it without undue perturbation provided everything else stayed the same.

Selecting "attach" from a menu, I. quickly attaches a note to the E. European network model asking the network planning guys to make sure there is nothing else planned in the future that will throw off his analysis. I. also attaches a note the to the E. European financial picture asking his administrative assistant to draft a letter to all international clients detailing US West's concerns and pledging to help them with the transition should it become necessary to move their operations.

I. then elects to save the complete scenario and model just worked out as justification in case his decision is questioned in the future, and a snapshot of the current model and all supporting data is filed in the appropriate place for later retrieval.

DCU and the War Room: current plans and possibilities

The War Room project is not yet at a juncture where detailed planning as to potential DCU contributions is possible. However, after a second meeting with the war room project leader, it has been decided to pursue 2 parallel activities as part of the case study.

Activity #1 Usability Analysis of current modelling software, and design of the next generation.

Description: The war room project is looking at two sets of available modelling tools for inclusion in the war room. These are the Microsoft Excel/@Risk packages and the SPS package. Cognitive walkthrough methods will be employed to better characterize the strengths and weakness of these packages from a usability standpoint. This will be done both from the viewpoint of using the packages to create models, as well as from the viewpoint of using the packages to interact with pre-existing models. Once the usability characteristics of these packages are better understood, a design activity will be carried out for the next generation of modelling software.

Key personnel: Warren Kuehner and Mike Epstein (US West)
Clayton Lewis and Nick Wilde (CU)

Hardware/Software Environments: Since no software is envisioned to be produced at this time, there are no hardware and software environment issues. The current packages run on Macintosh and IBM PC equipment.

Deliverables: The final deliverable will be a design document on the next generation of modelling software, based on current SPS and/or Excel/@Risk packages. An interim deliverable will be a listing of strengths and weaknesses uncovered by the cognitive walkthrough analysis of the existing packages.

TimeTable:

Mid-May	- W. Kuehner provides examples of Excel/@Risk models to DCU - M. Epstein provides an SPS solution patterned after the Excel/@Risk example provided by Warren.
Beg-Jun	- Task(s) to be used for cognitive walkthroughs specified by C. Lewis, N. Wilde
Beg-Jul	- Cognitive Walkthrough analysis of the 2 packages completed by C. Lewis, N. Wilde
Mid-Jul	- Write-up on strengths and weaknesses of current packages completed by DCU folks and presented to war room project (folks)
Mid-Aug	- Design of new package, based on above analysis, completed by war room project folks and DCU folks in a cooperative effort.

Activity #2 Iterative development of user requirements/ design of the war room facility.

Description: Good iterative design requires user input early in the process, both before the design activity can begin and while it is underway. To facilitate this, we propose to develop 2 "stories" that deal with the war-room facility as a whole. One, a user-based "story", gives the perspective of a user of such a system as to what it should look like, how one should interact with it, etc. The other, an implementor-based "story" will deal with what the system is actually built of, the functionality of the various modules, etc. It is thought that by focusing on these two early in the process, and alternating between successive versions of the two stories, one can be used to constrain the other - i.e. the user's "story" can be used to constrain the implementation to better provide what the user's actually want, while the implementors "story" can be used to constrain the user's expectations as to what is actually possible. The DCU group proposes to work with the war room project to develop these stories. Warren will work with Vic Pavlenko and other executive potential users to insure the implementation story is constrained to provide a design they are satisfied with, while the DCU group will help Warren to insure the user's story is at least technically feasible.

Key personnel: Warren Kuehner and Mike Epstein (US West) Clayton Lewis and Nick Wilde (CU)

Hardware/Software Environments: TBD. Part of the development of the implementation story should involve what hardware/software environment the envisioned system should run on.

Deliverables: The final deliverable will be the two documents spelling out the envisioned war room from the two different viewpoints. Hopefully, if we've done our job correctly, it will be evident that the two stories describe the same system. Interim deliverables will be the successive versions of the two "stories" as the war room developers in conjunction with the DCU folks labor to bring them in synch.

TimeTable:
TBD - Perhaps a good estimate would be to aim at holding a joint war room - DCU meeting once every two weeks over the summer, to provide and compare successive versions of the two stories. The aim should be to provide a complete document from both the users perspective and the developer's perspective by the end of the summer.

PROJECT 4

Graph Editor Generation
Jeff McWhirter
jeffm@cs.boulder.edu

Graph Models

Graph models are constructs that are used within computer based applications to represent objects and relationships among objects in a formal, structured manner. There often exists some intuitive visual representation of a graph model which is a very useful abstraction, greatly facilitating human interactions with the application. The ability of graph models to represent object/relationship abstractions in a visual manner make them well suited to be the basis for activities such as system modeling, visual programming, network design and computer aided software engineering.

The target activity of a graph model places certain requirements on the characteristics and behavior of the graph model. This results in the development of a large number of diverse graph models, each particularly suited to some task or set of tasks. With the increasing use of graphical computer human interfaces, previously unforeseen applications of graph models and the development of new graph models continue to arise.

The visual representation of a graph model implies the need for a visually oriented support system or graph editor. Graph editors allow for the construction, manipulation, analysis and storage of instances of their corresponding graph model. These graph editors often share a major disadvantage in that the structure, syntax, semantics and graphical representation of the graph model is encoded within the system, making for an inflexible, model specific graph editor.

It often requires much effort to develop a graph editor to suit the particular needs and requirements of a graph model. This level of effort often precludes the development of visual interfaces to graph model based applications. For those applications that depend upon a visual interface the construction of the graph editor may constitute a large portion of the application development effort. Much of the effort involved in the process of developing and exploring the uses of a graph model is devoted to the development of the graph editor.

DCU Collaboration

Many of the case studies of the USWest/CU collaboration use graph models and graph model based systems. The War Room project, SPS, and the Voice Dialog System (VDS) all have a graph model component. While the structure, functionality and end use of these graph models differ the requirements they place on their respective support systems are very similar. The following is a list of some of the common requirements which each of these graph models may place on their respective support systems.

* Model creation, manipulation and annotation.
* Model persistence.
* Arbitrary viewing.
* Visual hierarchies.
* Graph algorithms - path analysis, reachability, etc.
* Alternative representations - graph representations, spread sheet, textual.

* Interfaces to other applications - databases, voice synthesizers, simulators.

The focus of this research is to develop a framework which can be used to describe the structural and visual aspects of a broad range of graph models. The end result of this research will be a system which will aid in the process of graph editor development by automating much of the construction of graph editors. This work will enable a system designer to rapidly develop new graph editors which are specifically suited for some particular graph model. The use of a system such as this will greatly facilitate the development of, and experimentation with, new graph models and graph model behaviors, and in will free the system designer of much of the time consuming task of interactive system development. There are currently two projects, PRONET and Diegest, that are addressing the issues of graph editor generation.

GNMS/PRONET

The Generic Network Modeling System (GNMS), and its instantiation PRONET have been developed at USWest. PRONET is used to model network structures. Many of the general requirements of graph model based systems are addressed by GNMS and PRONET. For example, PRONET has the ability to contain a network inside of a node or edge. This ability greatly facilitates the development of large graph models. PRONET is an object oriented system and has been developed using Prolog. Each component of a graph inherits all of the abilities of a Prolog object providing for rich component functionality.

Diegest

There is an ongoing research effort at the University that is also involved with automating the process of graph editor development. A system, Diegest (Distributed, Extensible, Graph Editor, Specification and Transformation), is being developed which will be used to automatically generate interactive graph editor systems from a high level specification of a graph model. Diegest is composed of two parts: the Graph Model Specification Language (GMSL) and a set of tools Graph Editor Builder (GEB). Diegest will allow for the specification of a broad range of diverse graph models and the automatic construction of the concomitant graph editor. The overall goal of the Diegest system is to not limit the domain of graph models that Diegest is applicable to. For this reason the characterization of graph models that Diegest is based upon is very general. Very few assumptions have been made concerning the functionality and use of graph models in the domain of Diegest.

Graph editors generated using Diegest will have such abilities as multiple representations, arbitrary graph hierarchies, view filters and graph constraint checking. Diegest generates a full C++ class hierarchy conforming to the graph model specification. This allows for arbitrary attribute specification abilities as well as providing a structural basis for such model specific functionality as voice synthesis, program generation, path analysis etc.

The underlying architecture of the graph editors produced by Diegest provides an environment that supports multiple users, real time alternative representations and distributed critics. The architecture is also well suited for extending the overall functionality of the system to incorporate model-specific analysis in a distributed, real time manner. An important use of Diegest is its application to previously existing systems. It is often the case that a graph model based system lacks a visual interface due to the effort needed to develop the interface. Taking advantage of the underlying architecture one can use Diegest to quickly build visual interfaces to existing applications.

Results

Tools such as PRONET and Diegest will enable the system designer to rapidly prototype and develop new graph editors that are specifically suited for some particular graph model. We believe the use of these tools will greatly facilitate the development of, and experimentation with, new graph models and graph model behaviors, freeing the designer of the time-consuming task of interactive system development. Through the use of these tools various graph editors can be developed to support SPS and VDS as well as other future graph model based case studies. The use of these tools will allow for the experimentation with various model constructs and interface designs.

Milestones

September 91 - Complete the development of the graph model characterization framework. Begin development of the generic graph editor.

Winter 92 - Prototype of generic graph editor completed. Prototype graph editors for various DCU case studies.

Summer 92 - Refine generic graph editor prototype.

Conclusion

During the course of this research, we will be addressing fundamental issues concerning the general characterization and specification of graph models. We take a systemic view of the problem of graph editor generation, focusing on a wide variety of issues involving the creation of complete systems. As work in this area progresses prototype graph editors will be developed for the various case studies of the collaboration. This will provide value insight into the merits and drawbacks of the graph editor generation systems.