A Toolkit for Negotiation Support Interfaces to Multi-Dimensional Data

Michael Gebhardt
Informatik V
RWTH Aachen, Ahornstr. 55, D-52056 Aachen
gebhardt@informatik.rwth-aachen.de

Matthias Jarke
Informatik V
RWTH Aachen, Ahornstr. 55, D-52056 Aachen
jarke@informatik.rwth-aachen.de

Stephan Jacobs
TMOS
Ericsson Eurolab Germany
eedsja@eed.ericsson.se

Abstract

CoDecide is an experimental user interface toolkit that offers an extension to spreadsheet concepts specifically geared towards support for cooperative analysis of the kinds of multi-dimensional data encountered in data warehousing. It is distinguished from previous proposals by direct support for drill-down/roll-up analysis without redesign of an interface; more importantly, CoDecide can link multiple views on a data cube for synchronous or asynchronoous cooperation by multiple analysts, through a conceptual model visualizing the problem dimensions on so-called tapes. Tapes generalize the ideas of ranging and pivoting in current data warehouses for the multi-perspective and multi-user case. CoDecide allows the rapid composition of multi-matrix interfaces and their linkage to underlying data sources. A LAN version of CoDecide has been used in a number of design decision support applications. A WWW version representing externally materialized views on databases is currently under development.

1 Introduction

Decision making based on multi-dimensional information has been a popular approach to decision support systems for more than a decade. Regular conferences on multi-criteria decision making have been held since the late 1970's. Already in the early 1980's, it was also recognized that there is a potentially nice mapping between extensions of the relational view concept and the idea of multi-criteria decision-making. For example, in the MEDIATOR project (Jarke et al. 1984), we developed a multi-criteria negotiation support system based on a relational „data warehouse" which was regularly downloaded from the heterogeneous operational databases to serve as a basis for negotiation between the marketing, design, and manufacturing departments of a French car manufacturer.

SIGMOD '97 AZ,USA

Though useful, these kinds of systems tended to be clumsy both at the data management and at the user interface level. In particular, the switching between different granularities and perspectives required significant database programming effort, thus hindering the creativity of decision makers.

1.1 Problems with current spreadsheet approaches

On the data management side, data models, query language extensions, and optimization techniques for on-line analytic processing (Codd et al. 1992) have begun to change this picture. In these models, relational attributes are reinterpreted as dimensions in an n-dimensional data cube. Query language extensions such as the Data Cube aggregation operator (Gray et al. 1995) allow to define, in a single query, views on such data cubes which contain both base data and various levels and perspectives of aggregates over them. A lot of research is currently ongoing concerning the efficient computation (Harinarayan et a. 1996) and maintenance of such collections of views (Quass et al. 1996).

On the user interface side, the picture is less clear. The traditional spreadsheet is a natural starting point because of its great popularity but is limited in at least two ways. Firstly, the spreadsheet is essentially a two-dimensional representation. Extensions to multiple dimensions such as Lotus Improv were initially considered too complicated even though they were fixed to a pre-defined set of dimensions.

Behind these difficulties may have been a second major problem. As pointed out by Isakowitz et al. (1995), spreadsheet systems tend to mix logical, physical, and presentational aspects and thus force early design decisions which prevent easy re-organization.

In some spreadsheet extensions, this problem is partly remedied. For example, when you design a pivot table in Excel (Microsoft 1994), you have to inform the pivot table assistant in advance about a set of data dimensions from some relational table (possibly simulated physically through spreadsheet arrays), thus introducing a kind of schema notion into the language. A report representing a subset of a

data cube (with grouped aggregates) is then constructed by selecting hierarchies of dimensions for the rows and columns. The MetaCube approach of Informix (Stanford Technology Group, 1996) goes one step further by introducing an explicit meta model similar to a semantic network, together with an efficiently implemented multi-dimensional data structure on top of the basic relational (or possibly object-oriented) model. In particular, in MetaCube, you cannot just arrange data dimensions in a hierarchy on top of each other, but also represent hierarchies within a single data dimension (e.g. from an object-oriented database).

Still, the goal of all these approaches is the definition of a single analysis report. Multiple perspectives on an analysis problem, such as drill-down or roll-up, must be introduced by re-design even if the drag-and-drop facilities of modern spreadsheet packages make this relatively easy and quite efficient. Moreover, the cooperative analysis via multiple interrelated perspectives on a multi-dimensional database, as demanded by Codd et al. (1992) as well as in the literature on group decision and negotiation support (Klein 1993, Nunamaker et al. 1991, Ramesh and Sengupta 1994, Sycara and Lewis 1991), is not supported at all.

1.2 An overview of CoDecide

CoDecide, an experimental user interface toolkit developed in our group, is an attempt to close these gaps.

Like MetaCube, CoDecide maintains a multi-dimensional cache defined via a meta model and queries to underlying data sources, where each dimension can be hierarchical in itself.

The main difference to previous systems lies in the way how user interfaces are defined. In contrast to the pivot table approach, we do not construct a single matrix from the involved dimensions but arrange multiple matrix segments on a so-called tape, thus creating a *family of interlinked views* on the problem. These views can be looked at (e.g. scrolling, drill-down/roll-up) and manipulated (e.g. adding information) together. Moreover, they can be distributed across workstations with different access rights to the overall structure and different degrees of synchronization, thus enabling a wide variety of cooperative work support options.

A second difference is that *hierarchy operations* such as drill-down and scroll-up are not separated out in a design interface but *directly embedded in the matrix interface*, taking advantage of the opening/ closing option familiar from the Macintosh interface. Thus, CoDecide can be thought of as a user interface equivalent of the Data Cube operator (Gray et al. 1995), except that multiple related views on a cube are supported.

A local area network version of CoDecide has been operational since 1994 (Gebhardt 1994). It was used in a number of design decision support applications with moderate database sizes, including factory layout planning (Gebhardt and Jacobs 1996), software requirements traceability (Pohl and Jacobs 1994), and business process analysis (Jarke et al. 1996). This version was implemented using Tcl (Ousterhout 1994) and C on an X11 platform. It maintains its own multi-dimensional data cache, and allows real-time synchronous collaboration ('what you see is what I see') for up to about five simultaneous negotiators on a LAN. Interfaces for downloading from external data sources had to be built on a one-by-one basis though some elementary support was provided.

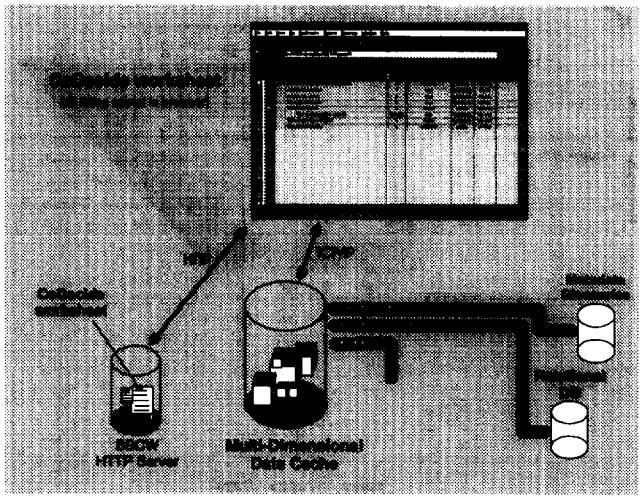

Figure 1. Architectural overview of the WWW version

In the context of a European project on cooperation support via the World Wide Web, we are currently completing a WWW version of CoDecide. In this version, the view cache maintained by CoDecide clients is seen as an externally materialized view on server databases which can be incrementally maintained using the algorithms presented in (Staudt and Jarke 1996). The resulting analysis documents are maintained in the BSCW shared workspace environment for asynchronous sharing and security support (Bentley et al. 1997). Metadata and additional glossary information are maintained in a separate object base, using the ConceptBase system (Jarke et al. 1995). A rough architecture of the WWW version is shown in figure 1.

1.3 Paper structure

In the following, we first define the basic components offered by CoDecide and link them to the Data Cube operator. We then describe the composition and functionality of CoDecide worksheets, and show how they realize externally materialized views. Finally, we present CoDecide's multi-user support which permits both synchronous and asynchronous cooperation modes, as well as full or partial information sharing in any modes. Several existing and planned application areas illustrate the concept. The presentation in this paper is mostly informal; a formal algebraic specification of the basic concepts can be found in (Jacobs, 1996).

2 CoDecide Components

CoDecide supports the development of hierarchy-sensitive multi-matrix user views on multi-dimensional data. We first explain the basic formal framework, then detail the associated visualization components.

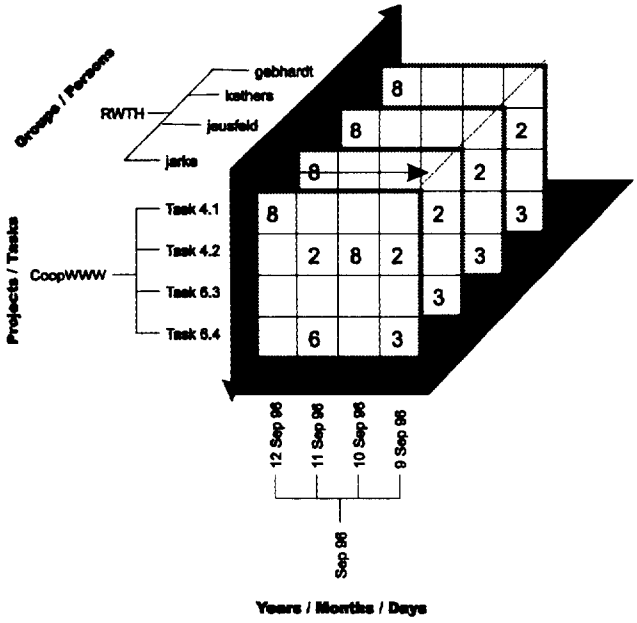

Figure 2. Data cube for timesheet management

2.1 Formal foundations

We assume that a negotiation problem is scoped by some n-dimensional space of data it considers; users can add aggregates or augment these data with their own subjective assessments. As an example, figure 2 shows a data cube for timesheet management in a distributed organization where people are both members of a line organization („group") and of certain projects. Typical negotiation problems in such a setting may include the time trade-offs as seen by the employees themselves, their group managers, and the leaders of the projects they are involved in.

Underlying such a cube is a conceptual database schema, such as the entity-relationship diagram of figure 3. Some dimensions may be in themselves hierarchical, e.g. time divided in years, months, days, etc. or projects divided in tasks.

Each CoDecide negotiation tool links multiple perspectives on such a negotiation problem. A *perspective* may represent the viewpoint of a particular negotiator, a particular aspect of a problem, or a combination of both. It can be initialized (and possibly maintained) as a database view which can be hierarchically composed from one or more data dimensions.

In the user interface schema of CoDecide, perspectives are represented as infinite *tapes* which can overlap (if they contain shared data dimensions), or intersect each other. A two-dimensional intersection is called a matrix, and represents a kind of cross-tab between the corresponding dimensions. Team cooperation will later be defined by specifying the visibility of data and operations on different pieces of tapes.

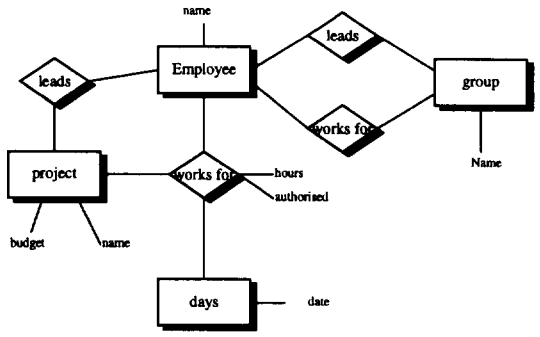

Figure 3. Schema for the project database

Each tape comprises a variable number of *tracks*. Thus, each tape T is a set of tracks $T = \{t_1, \ldots, t_n\}$. Each track t_i can be described by several attributes, e.g. projects tracks p_i of the project tape $P = \{p_1, \ldots, p_n\}$ can have a budget using the function $budget: P \rightarrow \mathbb{R}$.

The most important operations on tapes include

- insertion and deletion of tracks
- changing the sequence of tracks (e.g. sorting)
- scrolling on tracks

In contrast to spreadsheets which have a similar basic structure, CoDecide offers the possibility to build up *hierarchical structures* within a tape, using the child function $c: T \rightarrow \wp T$. For example, $c(t_6) = \{t_8, t_7\}$ describes that the tracks t_7 and t_8 are hierarchically below track t_6.

The operations on hierarchies are the same as those on tapes, plus the opening (*drill-down*) and closing (*roll-up*) of hierarchy levels.

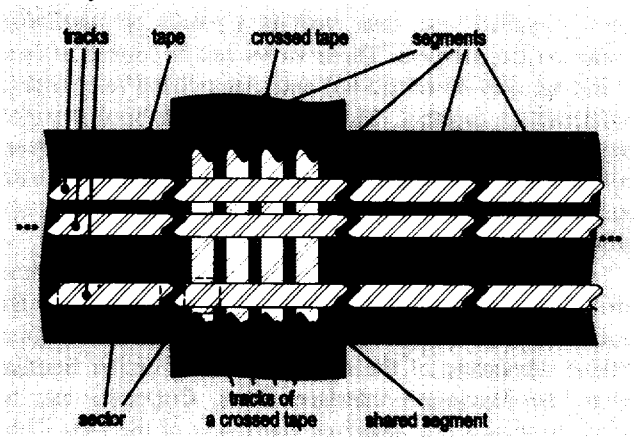

Figure 4. A tape structure with two crossed tapes

Matrices represent the intersection of two or more (not necessarily different) tapes. For example, a two dimensional matrix S of the tapes T and P is a set of sectors. $S(T,P) = \{(t_i, p_j) \mid t_i \in T, p_j \in V\}$ (cf. Figure 4).

Operations on matrices are indirectly applied via their underlying tapes, i.e. a matrix representation would scroll, drill-down, or roll-up when these operations are applied to their tapes.

A specific intersection is called a *segment.* Segments are called *shared* if they represent one-or more-dimensional crossings of tapes. Segmentation divides a track into *sectors* which represent the individual data entries in tapes, hierarchies, or matrices.

Segments can be associated with semantic knowledge defined by *logical or numerical functions* which compute the values of all sectors within the segment.

In the timesheet example (see figure 2) the worked hours should be aggregated by week and by project/task according the to the hierarchical structure within the tapes *Time T* and *Project P.* Thus the aggregation function $work{:}T\times P\rightarrow\mathbb{R}$ for the two-dimensional crossing representing the work per time and project can be defined as follows

$$work(t,p)=\begin{cases} x_{t,p}, \text{if } c(t)=\varnothing \text{ and } c(p)=\varnothing \\ \sum_{t_i\in c(t)} work(t_i,p), \text{if } c(t)\neq\varnothing \text{ and } c(p)=\varnothing \\ \sum_{p_i\in c(p)} work(t,p_i), \text{else} \end{cases}$$

where $x_{t,p}$ is a arbitrarily assigned value, i.e. in this case the hours a person worked on for the project an that day.

An example of a logical function is an integrity query which shows the violators of certain integrity constraints in the matrix. The hierarchy definitions imply the underlying group-by operations or quantifier scopes over which the functions are applied.

The hierarchical structure is inherited to the sectors of a segment. If both tapes *T,P* of two-dimensional segment are re organized hierarchically there are two additional structures offered beneath the one described by the matrix. The child function is defined for sectors:

$$c_{sec}((t,p))=c(t)\cup c(p).$$

2.2 Basic visualization components

To visualize a web of tapes and segments, CoDecide offers four kinds of basic visualization components by which you can create interactive worksheets: row (column), matrix, triangular half-matrix (roof), and hierarchical list (cf. figure 5).

Row components are used to visualize simple nonshared segments, representing attributes like the name or the weight of a track. They can be seen as a simple special case of *hierarchy* components which visualize hierarchical structures. In terms of the multi-dimensional data model, rows and hierarchies represent flat respectively hierarchical dimensions, but rows may additionally represent additional information attributes.

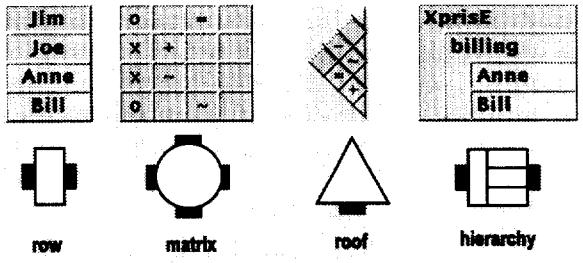

Figure 5. Examples and symbols for basic components

A *matrix* is used to visualize two-dimensional intersections. If intersections with more dimensions occur, several matrices have to be used. Each matrix then represents a two-dimensional view on the intersection.

The *roof* is used to represent self-intersections, i.e. a an intersection of tape *T* with itself. As in this case the tuples (t_i, t_j) and (t_j, t_i) are identical, there is only demand for the triangular half-matrix. Operations on matrices and roofs are inherited from their underlying tapes.

3 CoDecide Worksheets

Together with standard user-interface components like scrollbars, listbox, buttons, ..., multi-matrix interfaces are developed in the form of worksheets. As described above, some of the methods offered by worksheets are performed on the tapes, whereas others are performed on the segments or sectors.

3.1 Composition of worksheets

A CoDecide worksheet is composed by arranging instances of the components shown in figure 5 along tapes or on their intersections. Furthermore, these elements must be linked to information sources which will fill in the information, and possibly to aggregate functions. These information sources can be existing data structures from which the information shown on the screen is loaded, or information can be entered interactively by the user.

In the example shown in figure 6 (Jarke et al. 1996), the data have been loaded from a meta database of business analysis data. The rows describe activity hierarchies in a business, the crosses in the „roof" to the left mark the sequencing of these activities. The columns define information transport media, such as forms or telephone calls, and their data contents, for example the product number of a position in an invoice form. Thus, the columns show a hierarchy of two dimensions (media and data), one of which (the data) is itself a hierarchical structure. This illustrates a *hierarchical grouping of different database dimensions* on the same tape, as in NF2 relations.

Matrix entries describe operations the activities perform with the media respectively with the data. The matrix entries represent values of several different analysis views ap-

plied to the database. For instance, the highlighted sections in figure 6 mark conflicts between the perspectives detected by an integrity query defined in the business analysis strategy (here, e.g.: actions are defined on data structures which do not appear on the form). The left upper corner under the heading of the worksheet can be used for glossary rows or hierarchies which explain the symbols shown on the matrix; an example is given later in the paper.

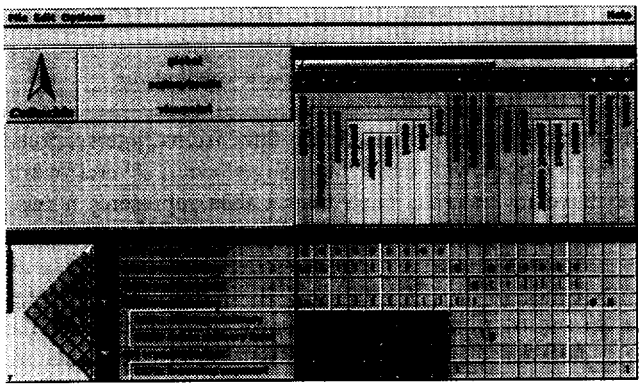

Figure 6: CoDecide interface confronting an action and a document perspective on a business process model (Jarke et al. 1996)

Scrolling is a typical method on a tape. The naturally expected effect of using the scrollbar in figure 6 is that not only the columns of the left row but also the columns of the corresponding matrix and of the left roof scroll together. Thus, scrolling has to effect the whole action tape, not only a single segment.

Interfaces such as this one can be described by a graphical design language. It arranges the base components on tapes and tape intersections, augmented with specifications how to retrieve, compute, and visualize the different sector entries. In the LAN implementation, Tcl scripts generated from this graphical representation set up the worksheet.

The graphical design pattern corresponding to figure 6 is shown in figure 7. It shows that the interface is composed from two hierarchical tapes and two crossings, one of them a self-crossing of the action tape representing the follows relationship between actions.

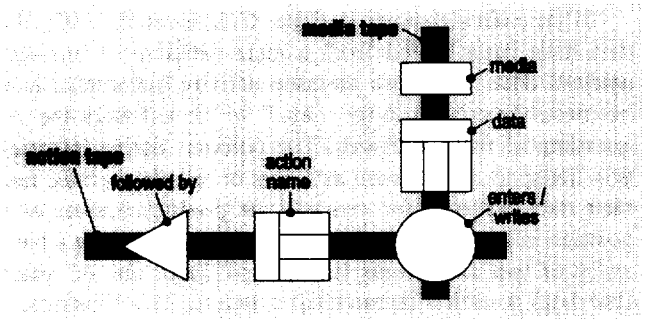

Figure 7: Action/media perspective as a tape web

Figure 8 indicates a more complicated example of a worksheet. It shows the specification and implementation of a so-called House of Quality, a popular representation for quality planning in manufacturing (Hauser and Clausing 1988). The definition shows three interacting tapes, one (the matrix rows) representing customer wishes (e.g. aggregated from a marketing research database), the columns represent technical product features (e.g. from a design or manufacturing database), and the third one descriptions of competitors.

The example demonstrates that tapes need not be visualized along straight lines. The competitor tape must cross both the customer wishes tape (how is the competitor perceived by our customers?) and the product tape (how do their product features compare to ours?). To visualize both crossings adequately, the tape turns around the corner of the customer/product matrix. This implies that scrolling also goes around the corner!

Three-way crossings are also possible. This is visualized by clicking through several matrices on top of each other. Obviously, this could be expanded by some animation facilities but we have not done this yet.

3.2 Worksheets as database views

A CoDecide worksheet defines a family of user views on an underlying global data cube representing the data dimensions mentioned on the worksheet specification. The degree of variation in the family is given by the operations on tapes and sectors (e.g. scrolling, drill-down/roll-up, choice of aggregate function). To achieve on-line analytic processing, the local data cube representing the whole family is cached in the current CoDecide implementation. It is likely that further optimization of this will be necessary to handle very large cubes.

The arrangement of CoDecide components on tapes and intersections determines the queries that fill these components, in a manner very similar to that used by pivot tables in Excel. We briefly sketch the case where the underlying database is relational. The object-oriented case, as represented in the implemented link of the LAN version to the ConceptBase system, is described in (Jarke et al. 1996) while Staudt and Jarke (1996) discuss a related approach to incremental view maintenance on the database server side.

Consider a tape such as the media tape in figure 7. The sequencing of rows, hierarchies, and matrices on the tape implies a hierarchy of the loading queries. The query related to the outermost object on a tape simply retrieves the different values of this dimension (e.g. from an attribute) or a hierarchy (from a collection of related attributes). For example, in our example, values of the columns correspond to the available documentation media in the business organization.

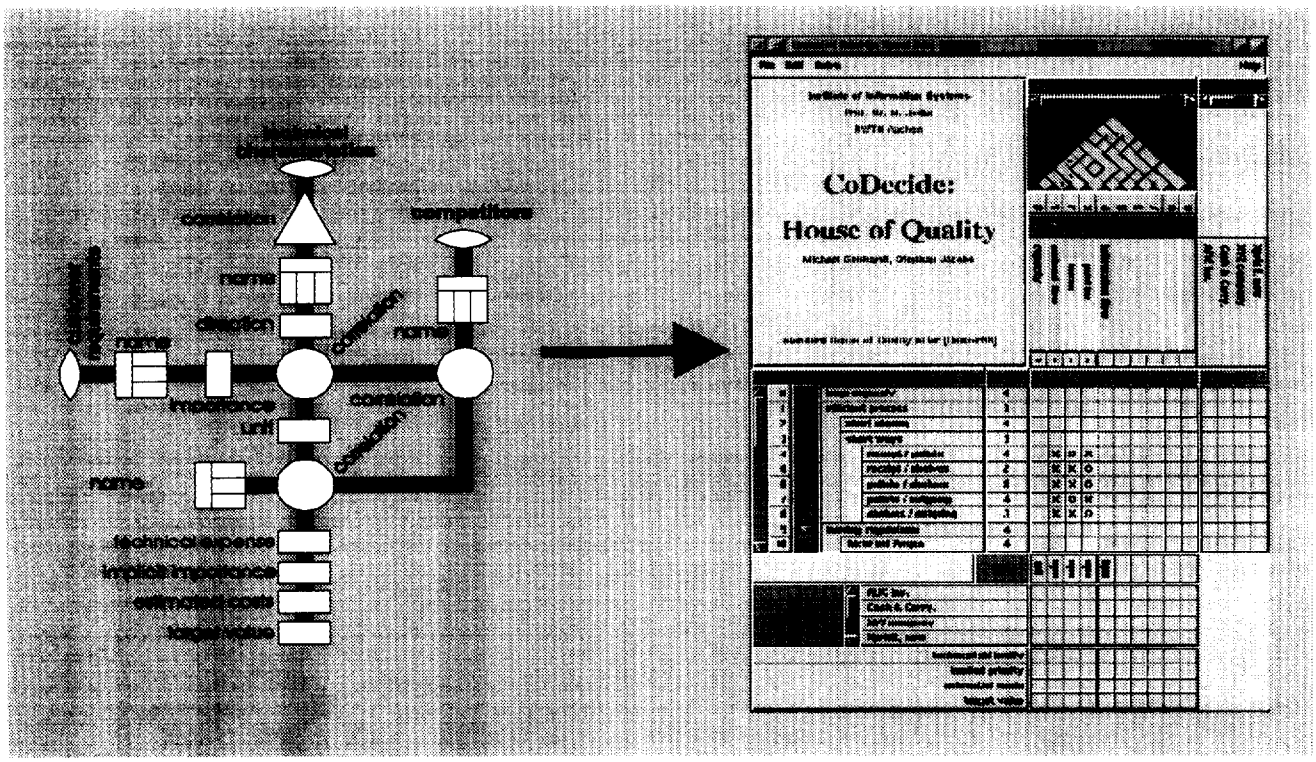

Figure 8: A more complicated CoDecide worksheet

Objects on a tape below the top-most one are again related to a (possibly hierarchical) dimension but in addition a group-by according to the values of all outer columns is added. For example, in figure 7, the hierarchy of data structures is grouped by the media on which they are captured. Sometimes, this grouping can be associated with aggregate functions, or simple retrieval of functionally dependent attributes, as can be seen in the House-of-Quality in figure 8.

Intersections of multiple tapes naturally inherit the groupings of all tapes they are involved in. So, the matrix entries of figure 6 are grouped by a hierarchy of actions as well as by media and (within that) of data. Again, output functions can be linked to such a query, in particular to define the aggregate values. Color and other symbols can be used to show the results of multiple analyses on the same screen. If many analysis functions are involved (in the above business analysis example, there were around 80!), it is convenient to be able to separate their computation from their visibility, as our architecture (cf. next section) does, thus allowing the users to focus on particular problems of interest at a time. Incidentally, this possibility to restrict the range of visible values or queries on the user interface, is also offered for the row and hierarchy constructs; it is called ranging in the data warehousing literature.

Taken together, this approach supports a broad variety of OLAP operations. Through the hierarchy, drill-down and roll-up are supported without switching to design mode, whereas operations such as ranging and rotate are specified through the design language.

4 Team Support with CoDecide

The central idea underlying team support in CoDecide is that tapes can run across multiple matrices, and in particular *across multiple analyst workstations*. As a consequence, change propagation across multiple visualizations of the same view or across different problem perspectives is naturally represented. We first present two examples of interlinked tapes, then discuss the underlying synchronization strategy.

4.1 Tapes across user interfaces

By linking the tapes of multiple user interface definitions, CoDecide can support multiple synchronized views on the same underlying data cube. In this case, the cube is (naively) chosen as the cube containing the union of all mentioned data dimensions as dimensions. Obviously, again optimizations are needed here but will not be discussed for brevity.

First, we consider again the time management example of section 2. Three different types of perspectives on the problem are shown in figure 9. The first one is that of an employee looking at his own timesheet, possibly updating it. The second one is that of the group manager who sees the timesheets of all her employees. The third one is that of a project manager, viewing those parts of employee's timesheet related to the given project. All of these interfaces are parameterized by the employee, group, or project in question.

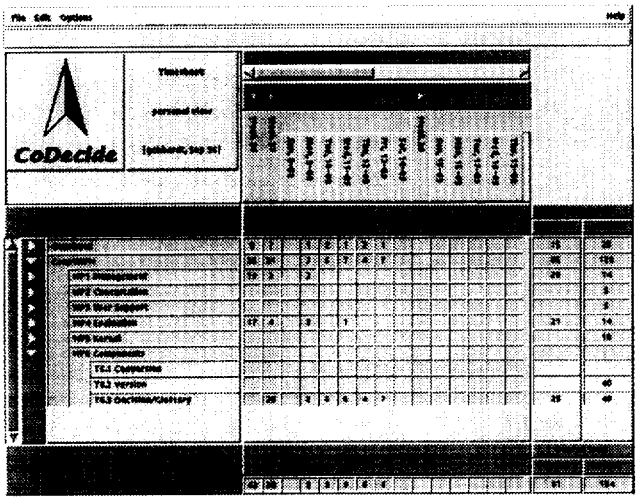

Figure 9a: Employee timesheet perspective

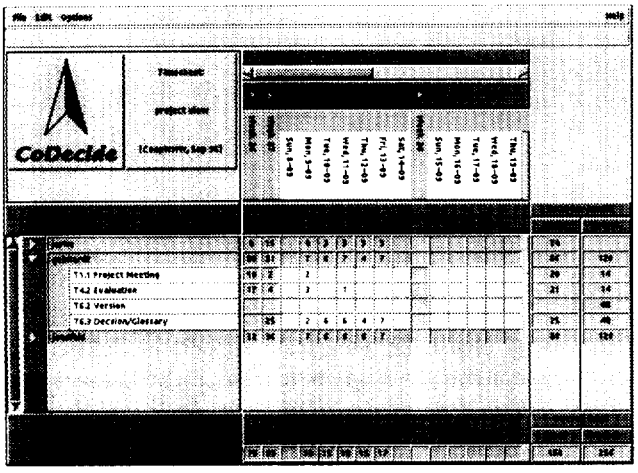

Figure 9b: Group manager timesheet perspective

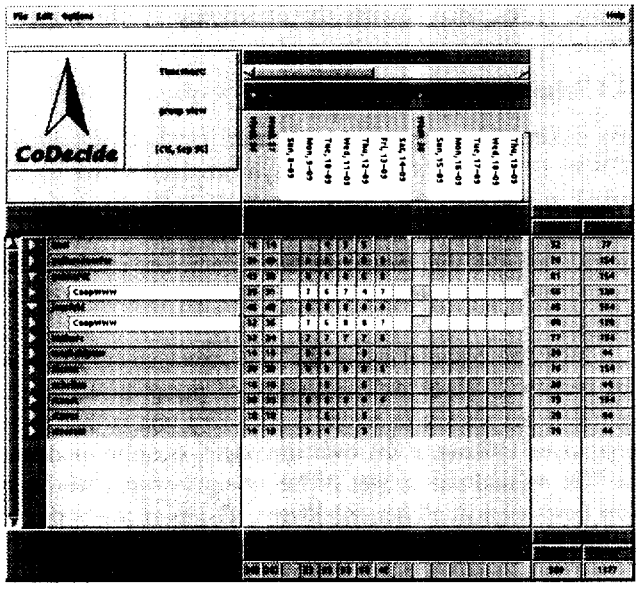

Figure 9c: Project manager timesheet perspective

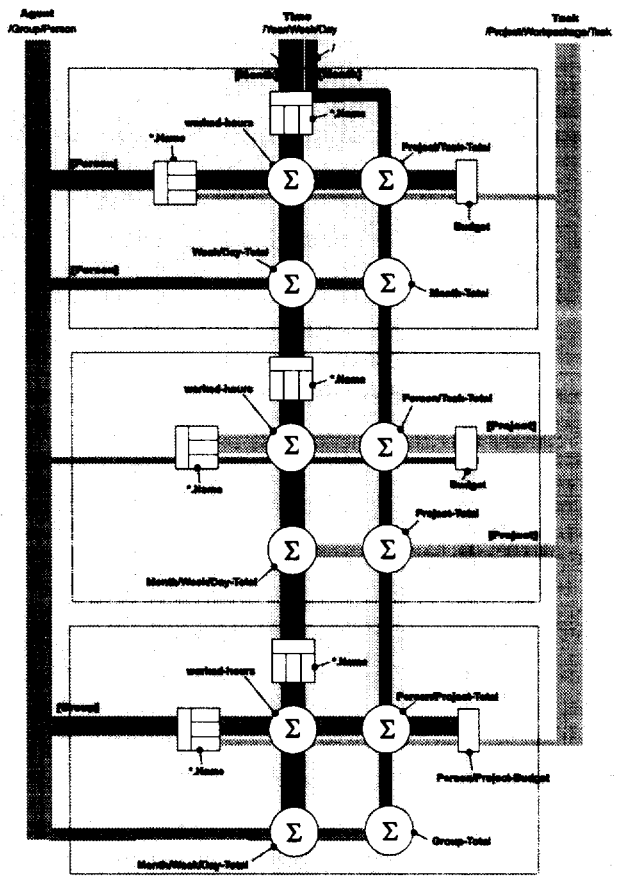

Figure 10: Tape schema of timesheet application

The tape design for this setting is shown in figure 10. It implies, for example, that an employee could discuss his timesheet with the group manager, and parts of it also with the project manager. This will be supported because user interface operations such as highlighting a particular entry or scrolling will, in full synchronization mode, be still operations on tapes and thus visible to the discussion partners. On the other hand, these data are then presented in a context that may not be visible for the negotiation partners.

As a more heterogeneous example, consider again the House of Quality in figure 8. Figure 11 shows a design how this has been linked to a commercial factory layout environment which determines whether the desirable technical properties are actually achievable (Gebhardt and Jacobs 1996). The factory layout environment supports multicriteria design decisions based on a database of reusable equipment components and valuation functions from engineering theory. CoDecide was used as a kind of wrapper on this environment. This allowed the addition of subjective evaluations from various stakeholders, version management on different layout options considered, and a feedback link to the quality planning stage. In this example, not only the user perspectives but also the underlying schemas and data sources overlap only partially, thus enabling mutual learning in addition to different interpretation of the same facts.

4.2 Group synchronization modes

The examples show that several different modes of team interaction via CoDecide may be useful depending on the negotiation setting. Based on the conceptual model of interlinked tapes and their visualization, a hierarchical architecture for cooperation modes at the physical level has been developed, as shown in figure 12.

Generalizing SmallTalk's Model-View-Controller approach to user interface management (Krasner and Pope 1988), the composition architecture consists of four levels (middle part of figure 12).

The lowest level defines to what degree the underlying data sources are shared (what part of the world is seen by the different users). This is defined by the tape structure and their linkage to underlying data sources. If these sources differ for the various users, we have the synchronization situation 4 in the figure. This situation where users see different data is typically used if there is either a strong difference in expertise or in interests among the different users.

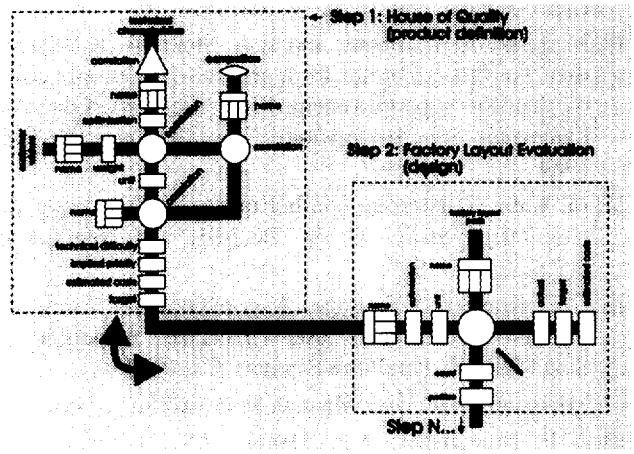

Figure 11: Tape schema linking planning and design

At the next level, transformations are specified, in particular including the choice of aggregate functions, but also other aspects such as sorting and grouping.

The third level determines how these transformed components are actually visualized on the different user screens (situation 2). In this situation, changes on data are visible across interfaces but may be visualized differently. For example, one user might see the situation at a greater level of detail than the other; in the timesheet example, the employee might enter data at the task level where also the project manager can see them, but the group manager is only interested in the project level summaries.

Finally, the fourth level determines the degree to which visual control operations (such as scrolling, roll-up, drill-down) are synchronized. Full synchronization means that all user interface operations are directly propagated and visible in the same way on the other user's screen.

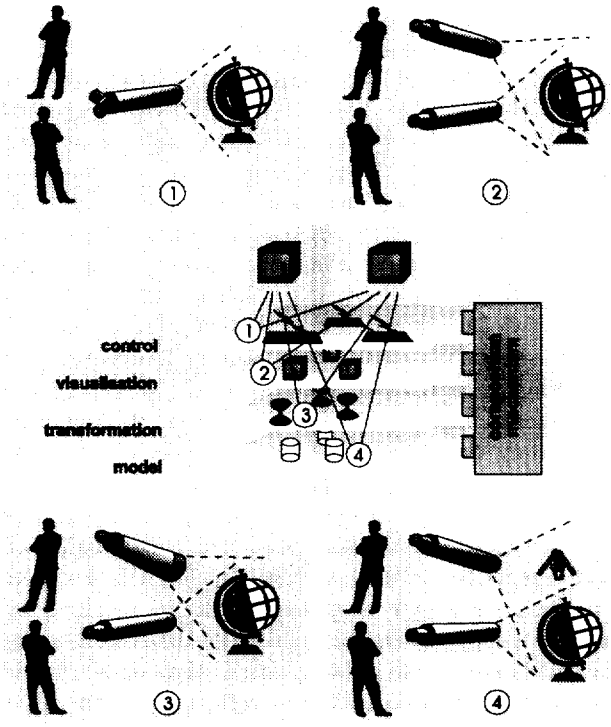

Figure 12: Levels of synchronization in CoDecide

5 Discussion and Outlook

In this paper, we have presented a group user interface to multi-dimensional data which can be seen as an extension of the spreadsheet approach for the specific demands of teamwork in data warehousing.

Distinguishing features of CoDecide include the rather direct representation of the different abstraction layers of the data cube operator in a single interface, and – more importantly – the linking of multiple analyses of the same data cube through the metaphor of interwoven, potentially infinite tapes.

Within the area of group decision and negotiation support systems, we see our approach as complementary to the one taken in meeting support systems such as GroupSystems (Nunamaker et al. 1991). While their approach supports a group process that is basically synchronous in each step, we maintain multiple opinions simultaneously. In fact, the two systems could easily have been used together and would have in fact profited from each other.

A number of positive experiences have been gained with applying CoDecide in design decision making with engineering and business analyst users.

In these applications, CoDecide was mostly used to support analysis and negotiation about meta data, usually not with more than a few thousand objects. This is similar to the range that tools like Excel can currently handle with reasonable efficiency, but may not be sufficient for data analysis and data mining upon huge amounts of base data. For such applications, it will be central to take advantage of

better management strategies for the multi-dimensional cache associated with each CoDecide application. Complementary to the idea of organizing such small data marts as self-maintainable views (Quass et al. 1996) on the database client side, we have been investigating their server-side maintenance through the idea of externally materialized views, as described in (Staudt and Jarke, 1996). This strategy is geared towards an application in the World Wide Web where clients cannot usually be expected to have database capabilties. However, experiments with the WWW version of CoDecide will have to show to what degree such a server-side solution can scale. Such experiments are in preparation as formal user studies with the WWW version in a company and a scientific organization, using applications in timesheet management and conference organization.

While our experiments have shown significant potential for the integration of database-centered and visually oriented negotiation formalisms, more research will be needed to increase the degree of automation in the mapping between the two, and thus the usability of the corresponding tools. Moreover, our artefact-based negotiation support needs to be coupled with active workflow advice on the negotiation process itself, once we actually know something about this process.

Acknowledgments. This work was supported in part by the European Commission under ESPRIT Long Term Research Project DWQ (Foundations of Data Warehouse Quality) and EU Telematics project CoopWWW (Basic Cooperation Support on the World Wide Web).

References

1. Bentley, R., Appelt, W., Busbach. U., Hinrichs, E., Kerr, D., Sikkel, S., Trevor, J. and Woetzel, G.: Basic Support for Cooperative Work on the World Wide Web. Intl. J. Human-Computer Studies, spring 1997.
2. T.X. Bui and M.Jarke. Communications design for CoOP: a group decision support system. ACM Trans. Office Information Systems 4(2):81-103, 4 1986.
3. S. Chaudhuri, U. Dayal. Decision support, data warehousing, and OLAP. Tutorial Notes, 22nd VLDB Conf., Mumbay, India, 1996.
4. E.F. Codd, S.B. Codd, C.T. Salley. Providing OLAP (on-line analytic processing) to user-analysts: an IT mandate. E.F. Codd and Associates, 1992.
5. M. Gebhardt. Coherent Design by Coupling of Views. Diploma Thesis, RWTH Aachen, 1994 (in German).
6. M. Gebhardt, S. Jacobs. Conflict management in design. 8th Intl. Symposium on Quality Function Deployment. Detroit, Mn, 1996.
7. J. Gray, A. Bosworth, A. Layman, H. Pirahesh. Data Cube: a relational aggregation operator generalizing group-by, cross-tab, and sub-totals. Microsoft Technical Report MSR-TR-95-22, July 1995.
8. V. Harinarayan, A. Rajaraman, J.D. Ullman. Implementing data cubes efficiently. Proc. ACM-SIGMOD Intl. Conf. Management of Data, Montreal, Canada, 1996, 205-216.
9. J.R. Hauser, D. Clausing. The House of Quality. Harvard Business Review, 63-73, May 1988.
10. T. Isakowitz, S. Schocken, H.C. Lucas. Toward a logical/physical theory of spreadsheet modeling. ACM Trans. Information Systems 13(1):1-37, 1995.
11. S. Jacobs. Conflict Visualization in Cooperative Design. Doctoral Thesis, Informatics Dept., RWTH Aachen, Germany, 1996 (in German).
12. M. Jarke, R. Gallersdörfer, M.A. Jeusfeld, M. Staudt, S. Eherer. ConceptBase - a deductive object base for meta data management. Journal of Intelligent Information Systems 4(2):167-192, 1995.
13. M. Jarke, S. Gebhardt, S. Jacobs, H.W. Nissen: Conflict analysis across multiple viewpoints: formalization and visualization. Proc. 29th Hawaii Intl. Conf. System Sciences, Wailea, Hw, 1996, vol. III, 199-210.
14. M. Jarke, M.T. Jelassi, and E.A. Stohr. A data-driven user interface generator for a generalized multiple criteria decision support system. Proc. IEEE Workshop on Languages for Automation, New Orleans, 127-133, 1984.
15. M. Klein. Supporting conflict management in cooperative design teams. Group Decision and Negotiation 2(3):259-278, 1993.
16. G.E. Krasner, S.T. Pope. A cookbook for using the model-view-controller user interface paradigm in SmallTalk-80. ParcPlace Systems, Ca, 1988.
17. Microsoft Excel User Manual. Microsoft Inc., 1994.
18. J. F. Nunamaker, A.R. Dennis, J.S. Valacich, D.R. Vogel, and J.F. George. Electronic meeting systems to support group work. Comm. ACM, 34(7):40-61, 1991.
19. J. K. Ousterhout. Tcl and the Tk Toolkit. Addison-Wesley, 1994.
20. D. Quass, A. Gupta, I.S. Mumick, J. Widom. Making views self-maintainable for data warehousing. Technical report, Stanford University, 1996.
21. B. Ramesh, K. Sengupta. Managing cognitive and mixed-motive conflicts in concurrent engineering. Concurrent Engineering Research and Application 2(3), 1994.
22. Stanford Technology Group, Inc. Informix MetaCube. http://www.informix.com, April 1996.
23. M. Staudt, M. Jarke. Incremental maintenance of externally materialized views. Proc. 22nd VLDB Conf., Mumbay, India, 75-86, 1996.
24. K.P. Sycara, M..C. Lewis. Modeling group decision making and negotiation in concurrent product design. International Journal of Systems Automation: Research and Applications 1:217-238, 1991