Comparing Usability of One-Way and Multi-Way Constraints for Diagram Editing

MICHAEL WYBROW, KIM MARRIOTT and LINDA MCIVER
Monash University
and
PETER J. STUCKEY
NICTA Victoria Laboratory, University of Melbourne

We investigate the usability of constraint-based alignment and distribution placement tools in diagram editors. Currently one-way constraints are used to provide alignment and distribution tools in many commercial editors. We believe the limitations of these constraints lead to serious usability issues, and thus suggest that such tools be implemented using multi-way constraints. We have conducted two usability studies, the first studies we are aware of that examine the relative usefulness of interactive graphical tools based on one-way and multi-way constraints. They provide strong evidence that multi-way constraint-based alignment and distribution tools are more usable than one-way constraint-based alignment and distribution tools.

Categories and Subject Descriptors: H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*Graphical user interfaces (GUI)*; *Interaction styles*; *Evaluation/methodology*; D.2.2 [**Software Engineering**]: Design Tools and Techniques—*User interfaces*

General Terms: Design, Performance, Human factors

Additional Key Words and Phrases: Constraints, diagram manipulation, layout tools

1. INTRODUCTION

When editing diagrams and other graphical documents it is often useful to be able to specify geometric relationships between the elements, for instance "left-align these three objects" or "equally space the selected objects between the outer two". A once-off movement to fulfill this relationship can be done by simply adjusting the positions of shapes. However, one would often like this relationship to be preserved during subsequent editing. Tools that set up such persistent geometric relationships are generally implemented using constraints.

A constraint specifies a relationship among element attributes that should be

A preliminary description of the results from the first study appeared in [Wybrow et al. 2003].
Authors' addresses: M. Wybrow and K. Marriott, Clayton School of Information Technology, Monash University, Wellington Road, Clayton, Victoria 3800, AUSTRALIA; email: {Michael.Wybrow,Kim.Marriott}@infotech.monash.edu.au; P.J. Stuckey, NICTA Victoria Laboratory, Department of Computer Science and Software Engineering, University of Melbourne, 3010, AUSTRALIA; email: pjs@csse.unimelb.edu.au.
Permission to make digital/hard copy of all or part of this material without fee for personal or classroom use provided that the copies are not made or distributed for profit or commercial advantage, the ACM copyright/server notice, the title of the publication, and its date appear, and notice is given that copying is by permission of the ACM, Inc. To copy otherwise, to republish, to post on servers, or to redistribute to lists requires prior specific permission and/or a fee.
© 2007 ACM 1073-0516/2007/0000-0001 $5.00

maintained. For instance, vertical alignment of three boxes A, B and C can be specified by

$$A.x = L.x$$
$$B.x = L.x$$
$$C.x = L.x$$

where L is an "alignment guideline". Over the last four decades there has been considerable effort in developing efficient constraint solving techniques for interactive graphical applications [Badros 2000; Hower and Graf 1996].

One-way (or data-flow) constraints are the simplest and most widely used approach [Vander Zanden et al. 2001]. They form the basis for a variety of commercial products including widget layout engines and the customizable graphic editors Visio[1] and ConceptDraw.[2] A one-way constraint is exactly like a formula in a spreadsheet cell. It has the form $x = f_x(y_1, ..., y_n)$ where the formula f_x details how to compute the value of variable x from the variables $y_1, ..., y_n$. Whenever the value of any of the y_i's changes, the value of x is recomputed, ensuring that the constraint remains satisfied. Thus in the above example they will ensure that if the alignment line is moved then the boxes will follow it. One-way constraints are simple to implement and can be solved extremely efficiently. They are also very versatile since f_x can be any function.

The main limitations of one-way constraints are that constraint solving is directional and that cyclic dependencies are not allowed, i.e., an attribute cannot be defined in terms of itself. Thus, for instance, in the above example if box B is moved the other boxes and alignment line will not follow it since the constraints only compute values for $A.x$, $B.x$ and $C.x$, and only as a result of changes to the value of $L.x$. A change to B effectively overwrites the formula that caused its position to depend on L. The formula for $B.x$ will also be overwritten if another constraint is applied to $B.x$, such as for instance requiring B's center to be vertically aligned with some other objects. Thus, with one-way constraints it is generally not possible for multiple dependent constraints to apply to the same object.

As a result of these limitations, so-called multi-way constraint solving techniques have been developed. In multi-way constraints, all variables can potentially be output variables—any variable can be calculated from the values of the other variables. With a multi-way constraint solver if B is moved then the other boxes and alignment line will follow. The various approaches to multi-way constraint solving are listed in Section 2.1 along with details of previous constraint-based diagram editors.

Despite the large amount of research in the area of constraints and graphical editors, most mainstream, commercially available diagram editors only provide tools that perform once-off placement, and those editors that do provide constraint-based placement tools, such as Visio and ConceptDraw, use one-way constraints rather than multi-way constraints.

We believe there are three main reasons why tools based on other, potentially more powerful constraint solving techniques such as multi-way constraints, have

[1]Visio, Microsoft Corporation, `http://office.microsoft.com/visio/`
[2]ConceptDraw, Computer Systems Odessa, `http://www.conceptdraw.com/`

not made their way into commercial graphical editors other than CAD applications. First, one-way constraint solvers are simple to write and extremely efficient. An efficient multi-way constraint solver is more complex to write and may require considerable numerical programming expertise. However, with the development of efficient algorithms for solving multi-way constraints and open source implementations of these algorithms this reason has become less important. Second, there is little or no evidence of their value. Graphical editors are unlikely to provide multi-way constraint-based tools without compelling evidence that they are going to be useful to users. Third, previous multi-way constraint-based prototype systems have proposed or suggested user interfaces for constraints, but these have varied greatly and often have perceived problems, or have never been exposed to real user testing.

There have been virtually no usability studies which investigate the value of the various constraint-based systems that have been presented. In particular, there has been no investigation of the general claims that multi-way constraint-based tools are better than one-way constraint-based tools, e.g., see [Hill 1993; Sannella et al. 1993]. This is the main contribution of this article. We have conducted two experiments comparing the usability of one-way and multi-way constraint-based alignment and distribution tools.

In our first experiment we designed and implemented a set of multi-way placement tools as an add-on for Microsoft Visio 2002. Visio already provides tools which allow users to set up persistent alignment and distribution relationships that are implemented using one-way constraints. An extensions framework is provided, which allowed us to plug a multi-way constraint solver into Visio. Using this platform we conducted a usability study to compare the usefulness of once-off tools, one-way constraint-based tools and multi-way constraint-based tools.

This study showed some statistical significance and general trends favoring multi-way tools over one-way constraint-based tools and over once-off placement tools. In particular, it showed severe usability issues with the one-way placement tools. We felt, however, on further examination that some of these issues might be a result of the user interface provided by Visio, rather than intrinsic limitations of one-way constraints.

There were two main issues that we felt might decrease usability of the Visio implementation of the one-way constraint-based placement tools. The first issue is that Visio effectively only allows an object to be in a single alignment/distribution constraint. However since the horizontal and vertical position of standard graphic objects are independent of each other, there is no inherent reason why an object cannot have a one-way constraint on its horizontal position and another one-way constraint on its vertical position.

The second issue is the degree of feedback provided during direct manipulation. Like most graphic editors Visio allows the user to drag an object or collection of selected objects to a new position, providing feedback by showing an outline of the selected objects as they follow the cursor. However, the position of other objects may indirectly depend upon the position of the selected objects because of constraints between them. In Visio the position of these other objects is not updated until the user has completed the action. So if the user moves a guideline with shapes attached then these shapes are only moved once the user releases the

mouse. We call this *delayed feedback*.

Some interactive constraint-based graphical systems provide what we call *immediate feedback* in which the user sees all changes to the diagram immediately as they happen. This includes showing how unselected shapes move when they are connected by constraints to the selected shapes. It has been suggested that this behavior is preferable since it allows users to understand a system of constraints more easily as they observe the diagram change from one state to another as a result of their actions [Gleicher and Witkin 1994]. Similarly, it should allow them to notice mistakes more quickly. An example of this is when dragging guidelines, the user would be able to see immediately if they were dragging a different set of shapes than the ones they believed were attached to the guideline.

In consequence, we ran a second experiment with improved versions of the one-way and multi-way tools. The Visio add-on interface was not flexible enough for our needs, so we created a new editor, Dunnart.[3] This provided basic diagram editing features, as well as revised versions of the one-way and multi-way placement tools without the previous limitations. Dunnart additionally allowed us to compare the effect of immediate and delayed feedback on the usability of both the one-way and multi-way tools.

The second study showed there to be a statistically significant difference between the times taken to complete a range of diagramming tasks for multi-way compared with one-way based placement tools. It was found that participants using multi-way tools completed tasks significantly faster than their one-way counterparts. Somewhat surprisingly, the study did not show significant difference between the immediate feedback and the delayed feedback groups.

Section 2 provides motivation for the research by examining past constraint solving in diagram editors as well as introducing and discussing shortfalls and limitations of existing once-off and one-way constraint-based alignment and distribution tools. Section 3 and 4 describe the design of the multi-way constraint-based tools as well as the experimental design, procedure and the results of the usability studies. Section 5 concludes.

2. BACKGROUND

This section explains the motivation for the research by describing previous research on constraint usage in interactive graphical applications as well as the shortfalls and limitations of once-off and one-way constraint-based placement tools in existing diagram editors.

2.1 Constraint solving in diagram editors

As discussed, one-way constraint solving is used to provide placement tools in the diagram editors Visio and ConceptDraw, but there has also been considerable research into multi-way constraint solving and its possible use within diagram editors.

Multi-way constraint solving approaches fall into four main classes: local propagation based (e.g., [Sannella et al. 1993; Freeman-Benson et al. 1990; Vander Zanden 1996]); linear arithmetic solver based (e.g., [Gleicher 1993; Borning et al. 1997; Marriott and Chok 2002]); geometric solver-based (e.g., [Fudos 1995; Kramer 1992]);

[3]Dunnart, Michael Wybrow, `http://www.csse.monash.edu.au/~mwybrow/dunnart/`

and general non-linear optimization methods such as Newton-Raphson iteration (e.g., [Nelson 1985; Heydon and Nelson 1994]).

The earliest example of an interactive constraint-based systems is Sketchpad [Sutherland 1963], which allowed the user to set atomic constraints, e.g., specifying that lines be parallel or perpendicular, or that a point lie on a line or circle. Sketchpad was also the first system to integrate constraint-based techniques with *direct manipulation*.

IDEAL [Van Wyk 1982] and METAFONT [Knuth 1979] were both systems that employed a textual declarative constraint language to constrain the position of points in a diagram. Several systems have looked at the idea of interactively inferring constraints: Chimera [Kurlander and Feiner 1993] with persistent constraints; and Pegasus [Igarashi et al. 1997] with once-off position adjustment. Penguins [Chok and Marriott 1998] provided persistent constraints with syntax-based inference.

Juno [Nelson 1985] and its successor Juno-2 [Heydon and Nelson 1994] offered "double-view editing" where the user could textually modify constraints as well as graphically, via direct manipulation. Briar [Gleicher 1992] allowed creation of constraints by letting the user drag objects into new relationships and then giving them the option of making the relationship persistent. Briar further explored the idea of alignment objects that are not part of the diagram itself, and exist only as objects to which other shapes can be constrained. GLIDE [Ryall et al. 1997] also took this approach, using "indicator" objects superimposed on the diagram to represent constraints. The authors of all these direct manipulation systems note the importance and difficulties in designing an interface for constraints that both allows the user to easily understand as well as control the constraints.

Constraints have enjoyed success in commercial Computer-Aided Design (CAD) software. Constraint-based CAD (or parametric CAD) software like Solidworks[4] and Pro/ENGINEER[5] have employed constraint solving techniques for handling changes in object features that are propagated to dependent objects, as well as geometric constraints placed on objects or features themselves.

2.2 Once-off alignment and distribution

Most diagram editors provide once-off alignment and distribution tools that adjust the positions of the selected objects. Some examples of software that offer these tools are Visio, ConceptDraw, OmniGraffle[6], SmartDraw[7], Dia[8] and Xfig[9].

When shapes have been aligned through once-off alignment their physical layout on the page will have changed, but the editor will not subsequently treat them any differently. In most systems, alignments work by adjusting the positions of all the shapes in the selection to align with a lead object. Figure 1 shows changes to a diagram as a result of left-aligning all shapes with shape B.

[4]SolidWorks, SolidWorks Corporation, `http://www.solidworks.com/`
[5]Pro/ENGINEER, Parametric Technology Corporation, `http://www.ptc.com/products/`
[6]OmniGraffle, The Omni Group, `http://www.omnigroup.com/omnigraffle/`
[7]SmartDraw, SmartDraw.com, `http://www.smartdraw.com/`
[8]Dia, Alexander Larsson, `http://live.gnome.org/Dia`
[9]Xfig Drawing Program, `http://www.xfig.org/`

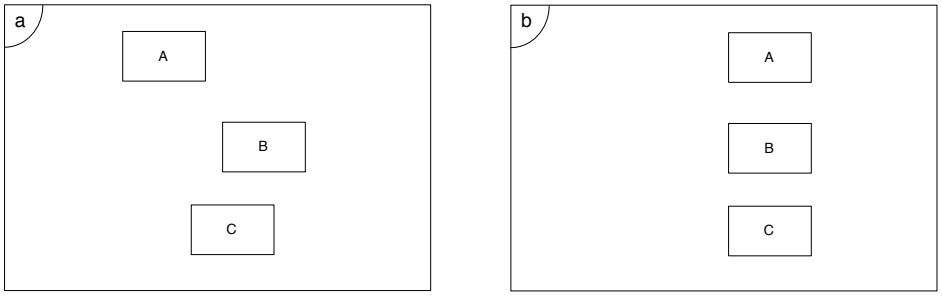

Fig. 1. Effect on layout due to once-off left-alignment of shapes A and C with shape B. Shapes A and C are moved into line with shape B.

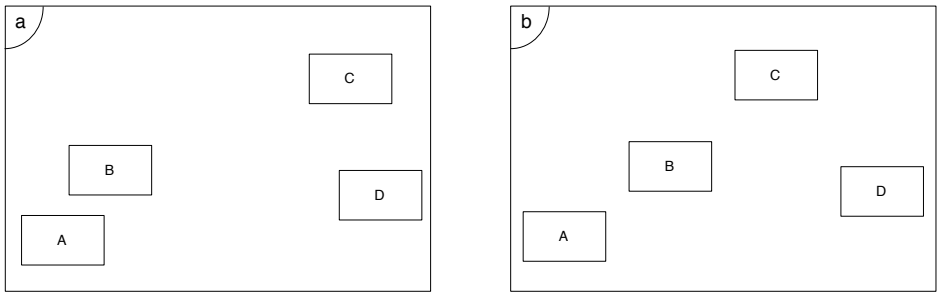

Fig. 2. Effect on layout due to horizontal distribution of shapes A, B, C and D by their center. The outermost shapes, A and D, remain in place, while the remaining shapes are moved so they are spaced equally between the outer two.

Distribution leaves the two outermost objects where they are and distributes the remaining objects in the selection equally between them. The user has control over the specific type of distribution used. Once again, no lasting relationship is created. An example of distribution is shown in Figure 2 where all shapes have been horizontally distributed by their center.

The key limitation of once-off placement tools is that they are memoryless. Thus, if the user aligns some objects and then selects one of these objects as part of a new distribution, that object will be moved to satisfy the distribution and they may need to realign the original objects again.[10]

2.3 One-way alignment and distribution

In addition to once-off placement tools, Visio and ConceptDraw provide a persistent form of alignment through the use of guidelines. Guidelines are purely placement

[10]The "group shapes" feature of most diagram editors could be seen as a method of preserving alignment. However, persistent alignment would allow us to move a single shape vertically along a vertical guideline without the other shapes in the alignment being moved, a grouped alignment does not allow this. Also, shapes can usually still be individually dragged within a group (without moving the other shapes in the group), which allows them to become unaligned, albeit though manual intervention.

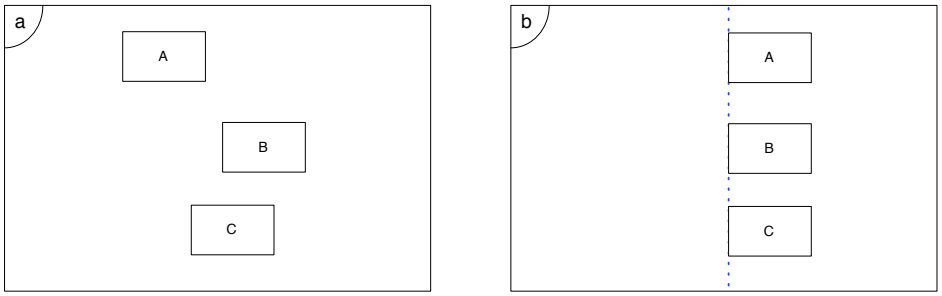

Fig. 3. Effect on layout due to one-way left-alignment of shapes A and C with shape B. A guideline is created in left-alignment with shape B, and shapes A and C are moved to align with the guideline. All three shapes become attached to the guideline.

aids; they act like normal manipulable objects on the page but are not part of the final diagram, i.e., they will not be visible on printed versions of the diagram. Visio also provides a guideline-based persistent distribution tool.

One-way constraint-based alignment tools work by creating a guideline connected to the lead object in the selection. The tools then adjust the positions of all the other selected objects to bring them in line and glue them to the guideline, as shown in Figure 3.

Often "snap-dragging" is provided as a means of attaching shapes to existing guidelines. Snap-dragging [Bier and Stone 1986] is a technique which uses a gravity metaphor where, as the user drags a shape, it will snap and connect to significant objects such as guidelines.

Once shapes have been glued to a guideline, they can be moved by moving the guideline. Unfortunately, one-way constraints only allow us to specify that the shape is constrained to align with the guideline. They do not specify that the guideline is constrained to align with the shape. As a result, moving shapes directly (rather than via guidelines) will always overwrite their position formulae which are used to describe the constraint-based relationship. This unglues them from guidelines.

The alignment and distribution tools themselves are required to move shapes to set up relationships. This breaks shapes from their prior alignment or distribution relationships.

Compounding this problem, there is no visible indication, at least in Visio or ConceptDraw, that a shape is glued to a guideline unless that shape is currently selected. This means that since the shape has not visually moved away from the guideline, the constraint will be broken without any visual feedback to the user. Such behavior means the user is unable to fully understand the state of the diagram from its on-screen representation.

These problems are illustrated in Figure 4 where two alignment relationships are set up, both involving shape B, a vertical alignment in Figure 4(a), followed by a horizontal alignment in Figure 4(b). In creating the second relationship, shape B's position is altered to be dependent on the position of the horizontal guideline—an action that invisibly removes the shape from the vertical alignment relationship.

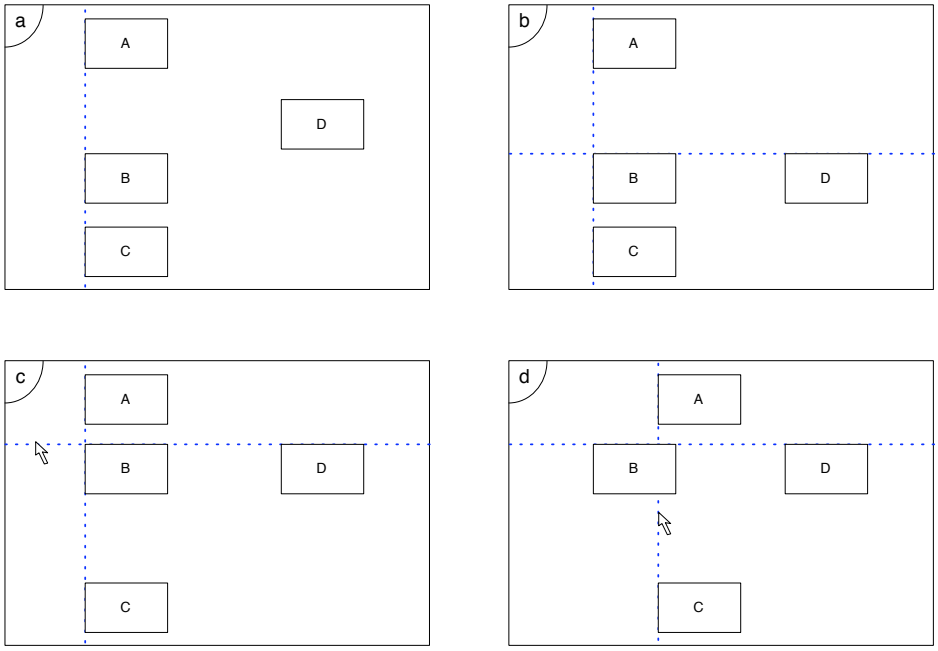

Fig. 4. Example of unexpected constraint breaking in Visio's one-way constraint-based alignment tools. Initially, in (a), the three shapes A, B, and C are left-aligned. Shape D is then top-aligned with shape B in (b). Next, in (c), the user drags the horizontal guideline up, moving shapes B and D along with it. When the vertical guideline is moved right (d), the user discovers that shape B is no longer attached to the vertical guideline.

Moving the most recently created alignment in Figure 4(c) works as expected. Then in Figure 4(d), manipulating the older alignment relationship, we see that shape B is no longer constrained to follow the guideline. This behavior is undesirable, since it makes it hard for the user to predict the response of the system. Obviously relationships will behave in different ways depending on the order in which they were set up.

This particular example clearly illustrates problems with shapes being invisibly broken from guidelines. We believe the major weakness of one-way constraint-based placement tools is that relationships can be broken by direct manipulation or by any other tools that affect the positions of shapes.

In Visio, shapes will only be attached to their most recently established placement relationship (and guideline), except when they have been explicitly placed in both a vertical and horizontal relationship through snap-dragging. It should be noted that in the example presented in Figure 4, the action that caused shape B to be broken from the vertical alignment—the creation of the horizontal alignment—does not actually require the first constraint to be broken. Using one-way constraints it is possible to have both the x and y position of a shape constrained to follow different guidelines. This particular behavior appears to be a bad design choice in Visio. Regardless of this, constraint breakage will still be unavoidable in many

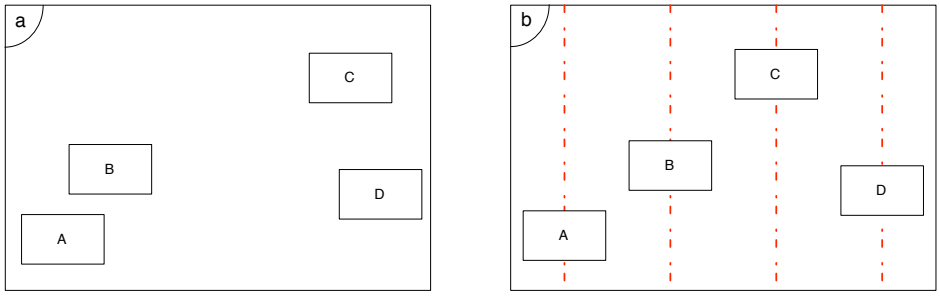

Fig. 5. Effect on layout due to one-way horizontal distribution of all shapes by their center. Guidelines are created for all shapes, and the shapes are attached to these. The guidelines (and shapes along with them) are spaced equally between the outermost two.

cases due to formulae being overwritten in one-way tools, i.e., vertically aligning an object that is already vertically aligned.

Like alignment, Visio's persistent distribution tools are implemented using guidelines. It creates a guideline for each selected shape and glues the shape to it. The tool then takes the two outermost guidelines and distributes the other guidelines (and attached shapes) equally between these, as shown in Figure 5.

As a result of using the tool, we end up with a persistent relationship that can be further manipulated. The outermost guideline on each end of the distribution can be dragged, effectively resizing the entire distribution. The other guidelines in the distribution cannot be dragged, because they are dependent on the positions of the outermost guidelines.

This behavior is sufficient for the basic case of distributing shapes, but we again run into the limitations of one-way constraints when trying to distribute shapes involved in alignment relationships. Unless we explicitly select the guidelines themselves for distribution the tool acts on and moves the individual shapes, effectively ignoring (and removing them from) any alignment relationships they are already part of. In Visio, the user can only distribute aligned groups of shapes by their guidelines if they wish to preserve existing alignments. This behavior violates usability principles that suggest systems should allow the user to arbitrarily substitute equivalent values for each other. Not only this, but the required action violates the usability concept of Familiarity or "closeness of mapping" [Green and Petre 1996], i.e., a user wishing to distribute shapes A, B and C, should be able to do so by selecting these shapes, making the interface closer to real world manipulation.

We can see that from the user's perspective that one-way constraint-based placement tools have a serious drawback: alignment and distribution relationships can break due to manipulation of objects involved in the relationship or because more than one constraint is applied to the same object. While some issues are introduced by specific implementations of the tools, the bigger problem is inherent in one-way constraints—each constraint has a fixed direction and an attribute can only have a single formula associated with it.

We therefore hypothesize that placement tools would be more usable if they provided truly persistent alignment and distribution relationships—two shapes put

into an alignment relationship should stay aligned through all further editing until the relationship is explicitly removed. The tools could then accurately use the metaphor of an alignment relationship, without the user needing to think about them as shapes glued to guidelines. As one-way constraints cannot support this, we must consider tools that are based on multi-way constraints.

3. STUDY 1

These considerations lead to our first usability study in which we compared the usability of once-off, one-way and multi-way constraint-based alignment and distribution tools.

We chose Microsoft Visio Professional 2002 as the platform for this usability study since it provides support for developer plug-ins and in particular allowed us to extend it with multi-way constraint-based alignment and distribution. Most commercial diagram editors do not provide support for developer plug-ins. Visio was chosen over the alternative of modifying an open source editor (such as Xfig or Dia) for three reasons. Firstly, it is widely used in industry, which makes the outcome of the research relevant and interesting to a greater group of people. Secondly, it is heavily customizable and provides support for writing add-ons that can neatly extend Visio's own tools and features. Thirdly, being an Office application it shares the common Microsoft Office interface, meaning it will already be partially familiar to anyone who has experience with Office products. The relatively wide exposure of Office applications meant that using Visio for the development and accompanying study, we were less likely to confound the measurement of the tools' usefulness with interface usability issues.

3.1 Multi-way constraint-based alignment and distribution tools

We first describe the multi-way constraint-based alignment and distribution tools that we integrated with Visio. Our tools were written in C++ and compiled as a Visio add-on Dynamic Link Library (DLL) with Microsoft Visual C++ 6.0.

Our implementation of multi-way tools made use of a multi-way constraint solving toolkit, QOCA [Marriott and Chok 2002]. QOCA allows us to create and solve systems of multi-way linear equality constraints. Multi-way constraints provide the ability to set up the initial alignment relationship so that moving the guideline moves the group of shapes attached to it, and moving any or all of the shapes also moves the entire group (including the guideline) where this still satisfies any other active constraints—the aligned group will stay aligned throughout all further editing. QOCA was chosen over the alternative of an incremental local propagation based toolkit such as DeltaBlue [Freeman-Benson et al. 1990] since it is able to handle cycles of constraints whereas DeltaBlue is not.

3.1.1 *Alignment.* The creation of an alignment relationship works the same way as existing tools in Visio—a guideline (if one does not exist) is created and aligned with the lead object, and all other shapes in the selection are moved to align with the guideline. It is only during subsequent manipulation of the diagram that the difference between the multi-way and one-way versions of the tools becomes evident.

The multi-way nature of the created relationship is clearly visible in Figure 6 where, when shape B is moved down and to the left, the two alignment relationships

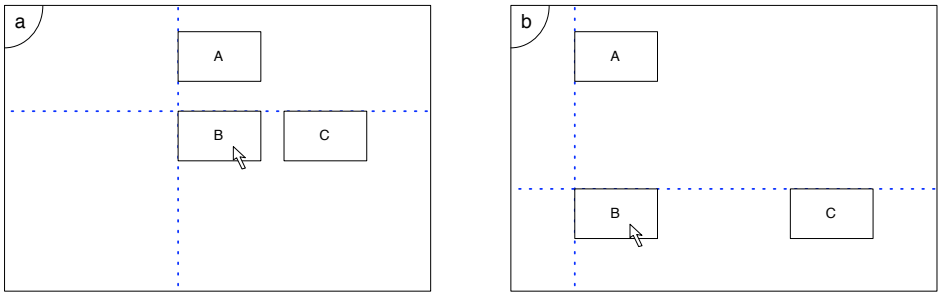

Fig. 6. Effect on layout due to moving shape B, which is involved in two multi-way alignment relationships, both down and to the left. As a result, the vertical guideline and shape A are moved left, and the horizontal guideline and shape C are moved down.

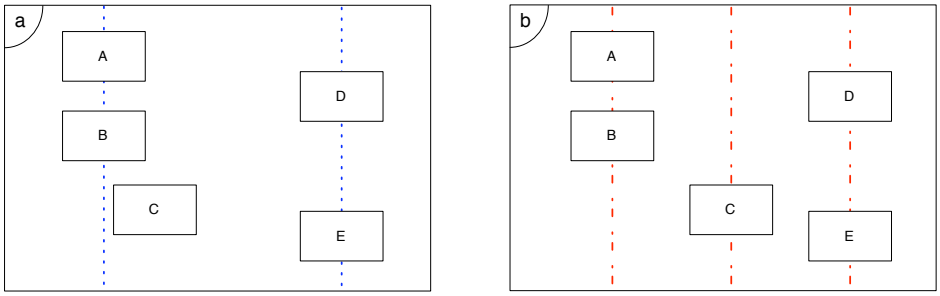

Fig. 7. Effect on layout due to multi-way horizontal distribution of all shapes by their center. Shapes A and B as well as shapes D and E are already center aligned. As a result, they are treated as a column and their existing guidelines are used for the distribution.

cause both shape A and C to move.

An alignment relationship can be removed by deleting the visible indicator of the relationship—the alignment guideline. A shape can effectively be added to an alignment relationship by aligning it with any (or every) shape in the existing relationship. This an be achieved by snap-dragging a shape on to a guideline as well as via the Align Shapes dialog.

3.1.2 *Distribution.* The initial effect of the distribution tool is again similar to existing tools—it considers all the shapes in the current selection, spacing them all equally (by their center, left or right) between the two outermost shapes. The difference is that the user can distribute shapes involved in alignment relationships and those relationships will stay active. Basically, when the user applies this tool, any group of aligned shapes will remain aligned, appearing to be treated as a single object for the purpose of distribution. This behavior is shown in Figure 7, where all shapes have been selected and distributed horizontally by their center. Internally, a variable (s) is used to represent the separation between two adjacent guidelines ($g1, g2$) in the distribution. A multi-way constraint is created between each pair of adjacent guidelines to constrain the separation between them to match

the distribution's separation: $s - (g1 - g2) = 0$.

Selected shapes without associated guidelines have new guidelines created for them and these guidelines are the subject of the actual constraints controlling the distribution relationship. Distribution guidelines are given a different color to distinguish between pure alignment guidelines and those involved in a distribution. The change in color is indicated in Figure 7 using a variation to the line stroke, where the alignment guidelines in Figure 7(a) change when they become part of the distribution in Figure 7(b).

Once a distribution relationship has been set up, dragging an outer guideline (either by direct manipulation, or movement of a shape "attached" to one) has the effect of growing or shrinking the entire distribution. Moving the very center guideline (or guidelines) of the distribution has the effect of moving the entire set of objects involved in the distribution. Moving any other inner guidelines partially resizes as well as moves the distribution. Since this behavior might not be completely predictable we would have liked to disallow manual movement of inner guidelines as Visio does, but this was unfortunately not possible with our add-on.

A distribution relationship can be removed by selecting and deleting all the distributed guidelines. When the user deletes just a single guideline involved in a distribution, the distribution relationship is removed. The other involved guidelines remain, but become (or revert to being) plain alignment guidelines, changing color to indicate this.

It is possible to place the same shape in several different types of alignment, e.g., left-aligned with some shapes while right-aligned with others. The alignment guidelines can themselves be part of distributions. In this way it is possible for the user to set up complex systems of constraints. Since alignment and distribution are described as "placement" tools it was decided that shapes should never be resized to force constraints to be satisfied. In the case where a shape would need to be resized in this way, or of constraints having no solution due to a new conflicting constraint, the last action is undone and a dialog box presented to the user explaining that their action created a conflict and could not be accomplished.

3.2 Method

3.2.1 *Design.* The placement tools are intended to aid the user in creating and modifying diagrams quickly. Therefore, in the user evaluation we are concerned with how long a participant takes to make changes to a diagram—obviously we would expect more usable tools to lead to shorter completion times. We are also interested in the relative number of errors found in the "completed" diagrams created by participants using the different tools. Here we expect more usable tools to result in fewer errors. In the study we used both exercise completion times and diagram correctness to measure the "comparative usefulness" of the tools.

The study consisted of a set of exercises in which the participants were asked to create, modify and manipulate diagrams resembling simple flowcharts. Flowcharts were chosen because they are a reasonably well-known and simple notation (at least for our participants) that capture the characteristics of other kinds of network-like diagrams.

The focus of the exercises was shape placement and overall diagram layout. We wanted the exercises to be simple enough not to require any prior knowledge of

flowcharts though we did not want them to be so simplistic that they seemed contrived. To this end, the diagrams given in the exercises were realistic flowcharts and the layout changes requested in the exercises were presented as aesthetic improvements to the diagram.

Participants in the study were randomly assigned to one of three groups. Each group was provided with a different set of constraint-based tools for alignment and distribution. The three groups were:

— **Group OO—once-off**: Once-off alignment and distribution tools were available. These move the involved shapes but do not create lasting relationships.

— **Group OW—one-way**: Visio's native form of persistent alignment and distribution tools based on one-way constraints were available.

— **Group MW—multi-way**: Persistent alignment and distribution tools based on multi-way constraints were available.

All participants were given exactly the same exercises, but could use only the particular tools offered to their group. Each group was trained on the particular set of tools available to them. In all other respects the training was identical for all groups.

It was hypothesized that the persistent state of the relationships set up by the tools in Group OW would make them faster and less error-prone than the once-off Group OO tools. Likewise, we hypothesized that the multi-way nature of tools in Group MW would make them faster and less error-prone than the one-way constraint tools of Group OW.

3.2.2 *Participants.* Thirty people were tested; ten in each of the three groups. There were no requirements for participants other than that they be computer-literate adults. All participants were undergraduate university students who were native speakers and readers of English, with normal or corrected-to-normal vision. Participants were not reused across groups.

3.2.3 *Equipment.* All tests were carried out in private, the investigator performing the experiment with a single participant at a time. The environment for the experiment was a usability lab in which the participant sat at a computer while the investigator sat behind them, observing and taking notes.

A record of each participant's interaction with Visio during the tests was obtained by taping a video feed of the test computer's display to VHS cassette. A small amount of audio data from post-test debriefing and discussion was also captured to the tape. In addition to this, the start and finish time for each exercise was taken down by the investigator. This included the time taken to comprehend the instructions. Other notes taken by the investigator summarized the strategy and method taken by the user to carry out the task, as well as problems they experienced.

Short pre- and post-test surveys were used as a means of obtaining some additional qualitative and quantitative data about participants' prior experience with related tools, how difficult they found the exercise and suggestions they had for the software's improvement.

At the beginning of each experiment the participant was shown a 15 minute training video. This consisted of a common introduction to Visio, as well as a specific

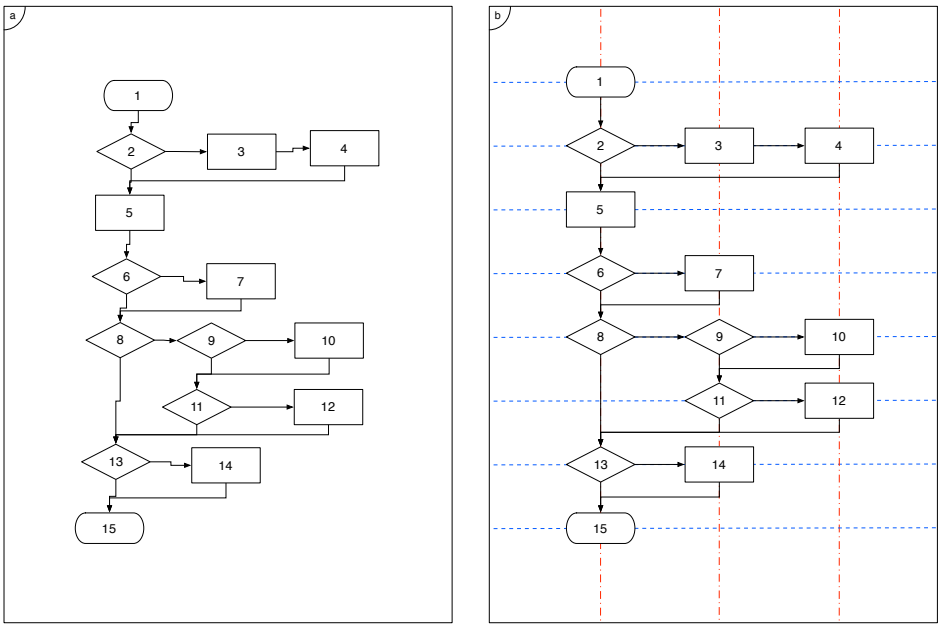

Fig. 8. Initial (a) and target (b) diagrams for the "Manipulation 1" exercise. The exercise requires participants to create several alignment relationships and a single distribution.

introduction to the tool set they would be using. Following this, the participant was asked to carry out some training tasks in an informal environment where the investigator would answer questions related to the software. The last of these tasks required the participant to construct a specific 16 shape diagram from scratch. When the participant had completed these tasks and was comfortable with Visio and its tools they proceeded to the timed exercises.

3.2.4 *Materials.* In the exercises, participants were required to modify some simple flowcharts. The exercises required the participant to make layout changes to the diagrams—spacing the objects on the page or aligning them to make the diagram more aesthetically pleasing. Some of the instructions and final diagrams showed a generic representation of alignment or distribution relationships, in this case the participant was required to enforce these relationships. They were also free to make use of additional placement relationships if they felt this would make the task quicker or easier.

The exercises were done one at a time, in fixed order. For each exercise, the participant was given a three page instructional handout.[11] The first page showed a typed description of the task written in point form in plain English. The second page showed the initial diagram, and the third showed the target diagram (i.e., the

[11]Complete copies of exercises, instructions and surveys for both studies are available online: `http://www.csse.monash.edu.au/~mwybrow/wybrow-tochi-2007-materials.tar.gz`

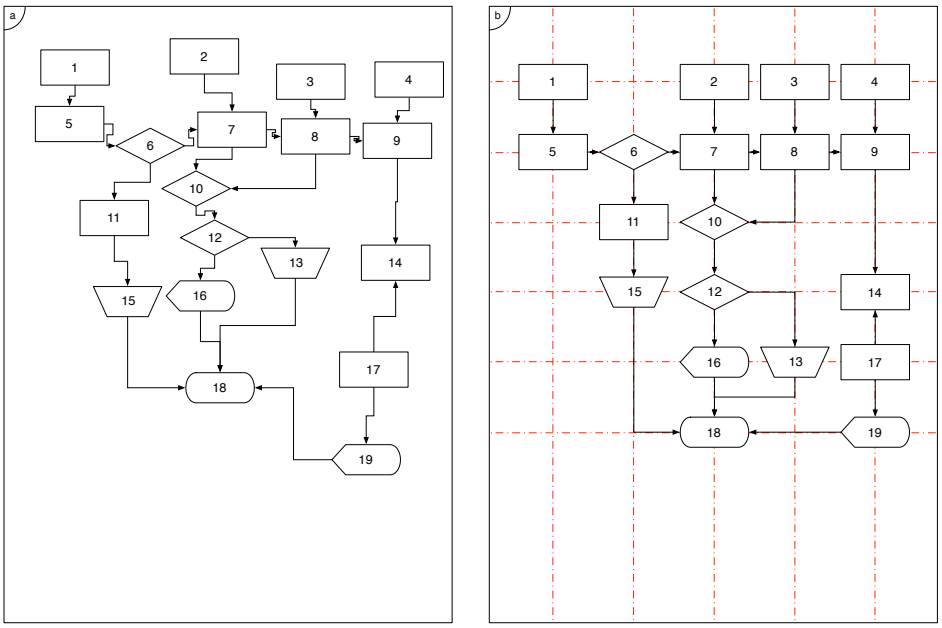

Fig. 9. Initial (a) and target (b) diagrams for the "Grid" exercise. The exercise requires participants to set up vertical and horizontal alignments as well as both vertical and horizontal distributions, creating a grid-like arrangement of shapes, making use of the available page space.

result of applying the specified instructions to the initial diagram).

The five timed exercises were:

— **"Editing"**: A simple exercise intended to increase familiarity with the editor and the available tools. Participants were required to make changes to a diagram resembling the one constructed during the training. They had to add two shapes to the diagram, reroute several connectors, and ensure two pairs of shapes were center aligned. Since the exercise was quite short and mostly a general editing task, it was not expected that performance on this task would show significant difference between the groups.

— **"Choice"**: Another editing task, beginning again with the training exercise diagram. Participants were required to remove three shapes, repair several connectors, and rearrange the diagram to make use of the entire page. Placement relationships were not explicitly mentioned in the instructions but these relationships could be inferred from the target diagram given. This was another short exercise, giving the participant more experience with general editing and a further chance to use the placement tools.

— **"Manipulation 1" and "Manipulation 2"**: These two exercises were designed as a pair. The first exercise required the participant to add some alignment relationships and a single distribution to a pre-constructed diagram. The initial and target diagrams for this exercise are shown in Figure 8.

The second exercise required the diagram to be resized to take up all of the available page. In this exercise no modification to the diagram was required apart from moving the objects in it. The required alignment and distribution relationships remain unchanged.

— **"Grid"**: The fifth and final exercise required modifications to another pre-constructed diagram, specifically the participant was required to set up vertical and horizontal alignments as well as both vertical and horizontal distributions. The final diagram shows a grid-like arrangement of shapes which makes use of most of the available space on the page. The initial and target diagrams for this exercise are shown in Figure 9.

3.3 Results

We consider completion times for the exercises, as well as errors in the completed diagrams. For the analysis we use well-known statistical techniques [Snodgrass et al. 1985]. To determine overall statistical significance we use a one-way randomized Analysis of Variance (ANOVA), where we consider $p < 0.05$ to be statistically significant. In the case of unequal group variances, as determined by Levene's test, the comparison of differences between means is instead achieved with a one-way ANOVA using the General Linear Model (GLM) in Minitab. As there has been no prior empirical analysis in this area, we are concerned where among the groups significant differences (if any) lie. For this reason we use Tukey's HSD test, a form of post hoc comparison, with p set at 0.05.

In our analysis we have excluded the results of exercises wherever the participant did not finish that exercise. It is interesting to note that in total five people did not finish all of the exercises, four from Group OW, one from Group MW, none from Group OO. The only exercises that were not always finished by participants were "Manipulation" and "Grid".

3.3.1 *Completion times.* The average completion times for each exercise are shown in Figure 10. To determine where the statistical significance lies we perform an ANOVA for each exercise.

A one-way ANOVA shows borderline significance for the first two exercises. The first exercise ("Editing", $F = 3.48$, $p = 0.046$) was the basic editing requiring only optional use of the alignment or distribution tools. Further analysis, by applying Tukey's HSD test, reveals there to be no significant difference between groups in times for the first exercise.

The second exercise ("Choice", unequal group variances, $F = 3.52$, $p = 0.045$) involves optional use of the tools. Tukey's test does reveal a significant difference between the times for Group OO and Group OW. This difference may be explained by participants in Group OW who chose to experiment with the use of the tools during this exercise, increasing their completion times. Placement tools would not be expected to have an effect on basic editing (excluding shape placement), it therefore is not surprising that more statistically significant results were not seen in these exercises.

We do find there is significant difference in the completion times for the exercise "Manipulation 1" (unequal group variances, $F = 7.19$, $p = 0.004$). Times for this exercise are summarized in Figure 11. Figure 11 is a standard box-plot, showing

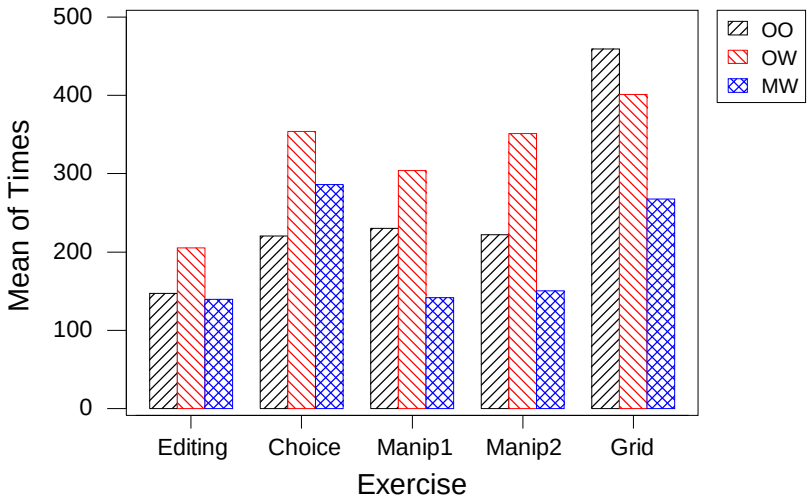

Fig. 10. Mean completion times (in secs.) for each exercise.

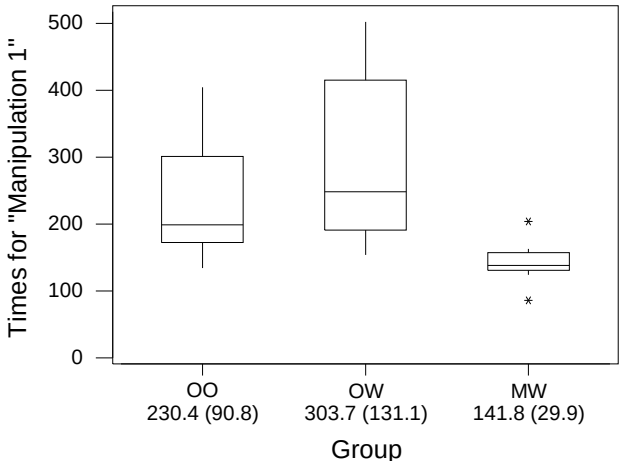

Fig. 11. Box-plot of completion times for "Manipulation 1" exercise.

a measure of spread. The boxes in the figure show the range of the middle 50% of the data, while the whiskers stretch to the largest and smallest values that are not "outliers". Outliers, those points more than 1.5 times outside the range of the middle 50%, are marked with a '*'. The mean completion time and standard distribution (in brackets) for each group are given below the box-plot.

To see exactly where the significance lies we use Tukey's HSD test to consider all pairwise differences between group means. Using this method we find that the only significant difference is between Group OW and Group MW. In this exercise the multi-way constraint-based tools of Group MW offer significant benefit over the

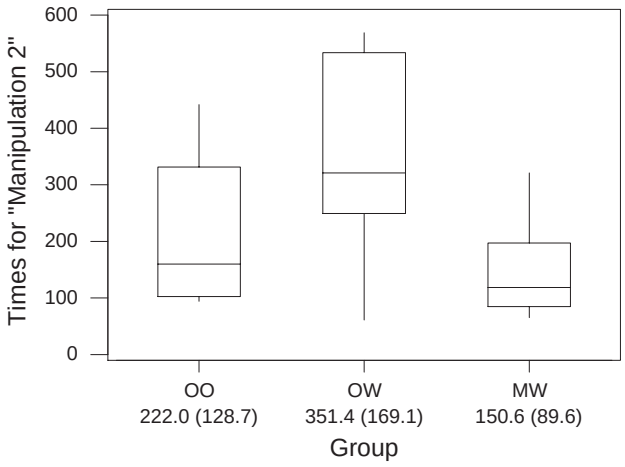

Fig. 12. Box-plot of completion times for "Manipulation 2" exercise.

one-way constraints of Group OW.

We again determine there is significance in the completion times for the exercise "Manipulation 2" ($F = 5.61$, $p = 0.010$). Times for this exercise are summarized in Figure 12. Once again, using Tukey's HSD test, we find that the only significant difference is between Group OW and Group MW. This exercise required that participants manipulate relationships they had set up in the previous exercise. We see that Group MW also benefits over Group OW in this aspect of editing.

In the study we made several observations that might explain why Group OW offered no significant benefit over Group OO for the "Manipulation" exercises. Participants in Group OO participants had to reuse the tools repeatedly to keep objects in the desired relationships. Group OW participants tended to have to do the same. Some shapes stayed in relationships, but a large number became unglued, leading not only to disassociated shapes but also to disassociated guidelines that no longer carried any meaning. Such objects cluttered the page and manipulation of them tended to be misleading and confusing for participants. In fact, some participants found it easier to delete such guidelines and continually recreate the relationships, effectively mimicking the usage of the once-off tools.

The final exercise ("Grid") also showed significant difference in completion times (unequal group variances, $F = 7.01$, $p = 0.004$). Times for this exercise are summarized in Figure 13. Tukey's HSD test showed that there was significance between times for Group OO and Group MW, and also between times for Group OW and Group MW. This supports that the multi-way constraints of Group MW are more beneficial for construction of heavily aligned and spaced diagrams than the once-off Group OO and one-way Group OW tools.

We next determine whether there was any interference between the groups and the exercises, i.e., whether differences seen between groups were due only to the tasks carried out for a particular exercise. Figure 14 shows group means as an interaction plot with error bars. An absence of interaction is illustrated by the relatively parallel lines of Group OW and Group MW for the final three exercises.

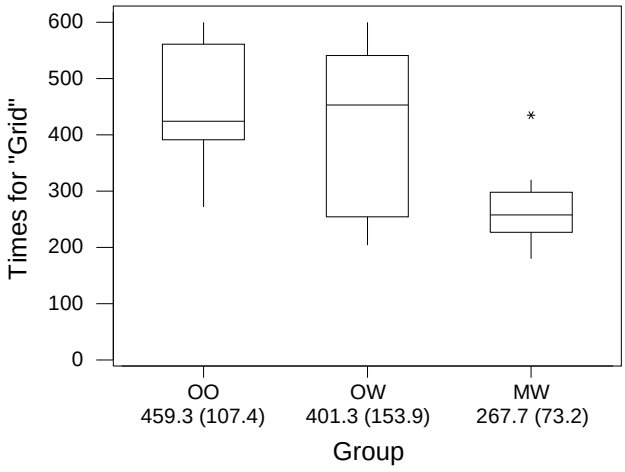

Fig. 13. Box-plot of completion times for "Grid" exercise.

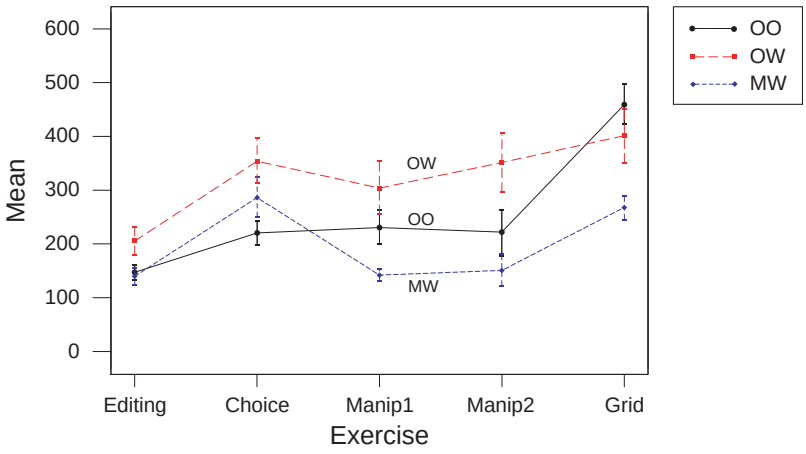

Fig. 14. Interaction plot for Groups OO, OW and MW.

This suggests that where we have seen significance, it is not due to the benefit of the tools for the particular individual exercises, but rather it is a benefit seen across all tasks.

Perhaps the most interesting result is the interaction between Group OO and Group OW. The plot shows that while Group OO out-performs Group OW (by means) on most exercises, the result is reversed for the final exercise. Since the Group OW tools are a persistent form of the Group OO tools, we had expected Group OW to out-perform Group OO across the tests. We found no significant evidence to support this. In fact, the time values in Figure 14 suggest that Group OW tools provide worse performance on all but the final exercise. This supports the

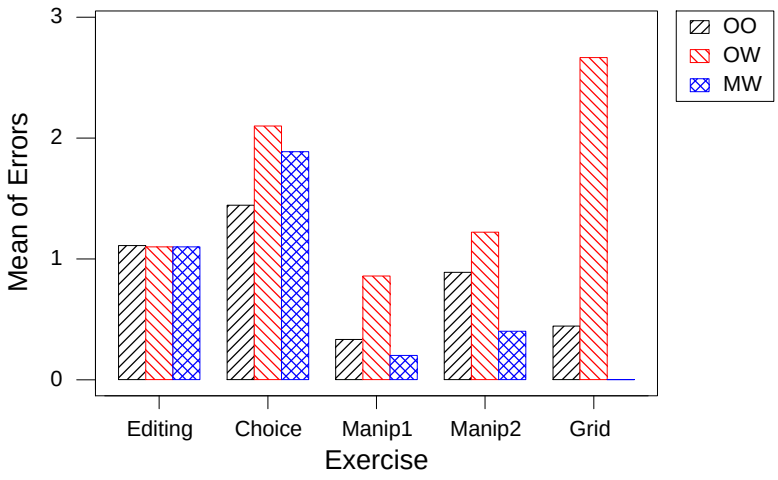

Fig. 15. Mean errors in final diagrams for each exercise.

observation that these tools suffer from usability problems. Though, even for us, it was surprising the extent of these problems on the tools' usefulness in terms of diagram editing time.

The "Grid" exercise is the only exercise that appears to show any kind of positive difference between Group OW and Group OO. An explanation of this might come from the fact that the exercise does not involve any manipulation of relationships once created. We also observed that many participants had by this exercise learned the quirks of the one-way constraint-based tools and had devised a particular order in which they could use the tools that would minimize the breaking of placement relationships.

3.3.2 *Error rates.* We also collected information about the number of errors present in participants' final diagram for each exercise. Diagrams were compared by eye to the target diagram. We classified as errors failure to carry out particular task instructions, as well as shapes not part of required alignment or distribution relationships—easily determined by the presence of kinked connectors.

The raw averages for errors in the final diagrams are shown in Figure 15. Apart from Group MW having significantly less errors than Group OW in "Grid" ($F = 5.79$, $p = 0.009$), these results were not statistically significant. Though by looking at the graph we can see that Group MW mostly leads to less errors than Group OO and Group OW. Here again, in the exercises requiring real use of placement tools, we see that the one-way constraint-based tools of Group OW are again more detrimental to performance than their simple once-off Group OO counterparts.

3.3.3 *Participant feedback.* Some interesting qualitative results come from participant feedback provided by the post-test questionnaire. As well as being asked to give general comments or suggestions participants were specifically asked the following two questions:

> "Could you please describe any parts of the exercises you found particularly difficult?"
>
> "Based on the exercises you were asked to perform, if you could change or add anything to Visio what would it be?"

In general, the comments supported the hypothesis that the multi-way constraint-based version of the alignment and distribution tools were more usable than either the once-off or the one-way constraint-based versions.

The thrust of the comments from participants in the once-off group was that they wanted the tools to create permanent relationships, i.e. they wanted constraint-based tools. Seven out of nine participants stated they had problems with the distribution tool not operating on alignments. A typical comment was that "aligning and distributing objects requires too much manual work". Additionally, they expressed difficulty with knowing "the correct order... so I don't have to redo steps". As a result, eight participants suggested adding some form of "better" distribution that "doesn't affect alignments", maybe using a kind of "sticky" alignment, or providing some way to "lock" groups of shapes or alignments. Interestingly, during the exercises many participants still expected the tools to create permanent relationships, even though the training was very specific in stating this wasn't the case. This was echoed in their comments.

Feedback from the participants who had used the one-way constraint-based tools described the expected usability issues. Seven out of ten participants stated they had problems with shapes becoming unglued from guidelines, "shapes breaking out". Two others reported general difficulties with alignment and distribution, one blaming the clarity of instructions and the other blaming themselves—specifically the order in which they used the tools. A final participant reported the activity of manually attaching nodes to guidelines as "tedious". Three participants stated difficulties with distributions acting on all objects rather than alignment groups. Their suggestions were to "have shapes always stay on guides", or change Visio so that it "remembers all alignments, and if you want to detach them you can do so manually" and that "alignment relationships be respected when applying new distributions and when moving the object as opposed to the guideline". In other words they wanted the one-way constraint-based tools to behave like the multi-way constraint-based version.

Very few participants using the multi-way constraint-based placement tools had problems with the tools or expressed suggestions for their improvement. Two participants had difficulties with not being able to distribute guidelines directly rather than just shapes. This was basically a design oversight. As we described earlier, our distribution constraints actually operate on guidelines rather than shapes themselves so it would have been easy to implement this behavior. It is worth noting that the error message generated in this case caused participants to change their strategy for distributions and caused them no further problems. One participant using the multi-way tool encountered a problem where they couldn't both left-align and center-align two objects of apparently equal width that actually had very slightly different widths. This is essentially a reporting problem since the error message only tells them that the action they were attempting would break an existing placement relationship. It would be much more useful to be able to let

them know the set of existing constraints that were preventing the action. Three participants suggested the ability to lock shapes at a particular position, once they were happy with their layout.

Four participants using the multi-way constraint-based placement tools and four participants using the one-way tools reported problems with clutter from guidelines obscuring the underlying diagram. This supports previous claims that clutter of onscreen constraint indicators could be problematic for users [Heydon and Nelson 1994; Gleicher and Witkin 1994]. A frequently suggested solution from users was to use a different kind of visual indicator or a tool to easily hide guidelines. Another comment was that it was difficult to determine whether a guideline indicated an alignment or a distribution relationship. These problems suggest that more research is needed in order to find a visual representation for placement relationships that will scale up to large diagrams without cluttering them.

3.4 Discussion

Our results support our hypothesis that placement tools based on one-way constraints have usability issues and that multi-way constraint-based placement tools offer significant benefit over one-way constraint-based tools for tasks requiring the alignment and distribution of shapes.

Interestingly, our results show persistent placement tools based on one-way constraints offer no significant advantage over the simple, once-off tools offered by nearly all diagram editors. The one-way tools can be thought of as an extension of the once-off tools, yet our results suggest that they provide no added value to the user for general editing and layout tasks. In fact, it appears that one-way constraint-based tools mostly lead to slower times and more errors in the finished diagram than once-off tools.

Multi-way constraint-based tools were not found to offer a statistically significant advantage over once-off tools in all tasks, though in tasks requiring alignment and distribution of shapes they consistently resulted in faster average completion times and fewer errors in the final diagram. Given this, we believe that significance would be seen given further testing.

However, as discussed in Section 1 we felt that some of the usability issues for the one-way constraint-based placement tools might be a result of the user interface provided by Visio, rather than an intrinsic limitation of one-way constraints. The first issue (as exemplified in Section 2.3) is that Visio essentially only allows an object to be in a single alignment/distribution constraint. However, since the horizontal and vertical position of standard graphic objects are independent there is no inherent reason why an object cannot have a one-way constraint on its horizontal position and another one-way constraint on its vertical position.

The second issue is the degree of feedback provided during direct manipulation. Visio only provides delayed feedback during direct manipulation, meaning that if the user moves a guideline with shapes attached then these shapes are only moved once the user completes the action by releasing the mouse. It has been suggested that immediate feedback of constrained objects during dragging is valuable to the user for understanding and debugging constraints [Gleicher and Witkin 1994; Ryall et al. 1997].

4. STUDY 2

As a result, we felt a follow-up usability study was required. It was designed to validate the results of the first study by removing confounding factors due to the design of Visio's one-way constraint-based placement tools. It was also intended to test the value of immediate feedback for both one-way and multi-way constraint-based placement tools.

For this study we decided to write our own diagram editor, rather than modifying Visio. We had found that the interface that was provided for interacting with Visio was fairly limited. Since we wished to change the behavior and on-screen representation of Visio's one-way constraint-based placement tools this would have required us to reimplement these tools as plug-ins. Even more importantly, it did not seem possible to modify Visio's behavior to provide immediate feedback since any add-on can only ever act in response to an event and Visio does not post events during dragging, only posting a "shape move" event when the user has completed the action and dropped the shape at its new location.

For this reason we required an editor that was built from the beginning with constraint solving and immediate feedback in mind. Rather than trying to write this on top of an existing code base that may not prove to be suitable, we chose to write a simplified diagram editor to be used exclusively for the usability testing. The interface and available features were kept to a minimum, reducing the possibility of participants experimenting with or being confused by unneeded menu options or unnecessary tools. Having written the editor, we were able to instrument it in ways that were useful for testing. For example, we are able to replay a participant's actions in the editor, watching the mouse move around the screen, much like a video replay. This saved us having to record the experiments with traditional cameras or screen capture. We could also collect statistics about the type and number of actions that participants used to complete the tasks.

4.1 Software tool design

In this section we describe slightly revised one-way and multi-way constraint-based alignment and distribution tools that we provided in Dunnart, our new diagram editor.

Dunnart itself is written in C++ and compiles and runs on Windows, Linux and Mac OS X. It has a simple interface and uses a custom User Interface toolkit that mimics the look and feel of Visio on Windows. Dunnart allows all the standard interaction with the diagram that you would expect from a diagram editor; you can add shapes to the page, move, resize and label them. You can cut, copy and paste selections, and the editor allows undo and redo commands. Dynamic connectors are available that will reroute themselves as a result of manipulation of the shapes to which they are attached.

4.1.1 *Alignment.* The creation of an alignment relationship, accessed through the "Align Shapes" toolbar button and corresponding dialog box, results in a guideline (if one does not exist) being created and aligned with the lead object. All other shapes in the selection move to align with and become attached to the guideline.

For the one-way tools the user can then move the guideline to move all the attached shapes. Since this is achieved with one-way constraints, any shape can be

attached to at most one vertical guideline and one horizontal guideline. Vertically aligning a shape that is already aligned (with other shapes) by a separate edge will cause this formula to be overwritten and will leave it attached to the more recent relationship's guideline. Likewise, if a shape is moved then its position formula will be overwritten, causing it to break from existing relationships.

In contrast, when shapes involved in a multi-way relationship are moved or resized the other shapes involved in the alignment (and the guideline itself) will also move. This is subtly different from grouping the objects in that, for example, a vertically aligned shape is able to slide up and down the guideline without vertically moving any of the other objects attached to the guideline, as shown in Figure 6.

We allow the user to add a shape to an existing guideline by dragging the shape over the guideline. While the shape is hovering, aligned with the guideline the guideline will be highlighted red. At this point if the user then releases the mouse button the shape will be attached to the guideline (added to the alignment relationship).

We also allow the user to free shapes from multi-way relationships. Rather than having to delete a guideline to remove a relationship, the user is able to enable "free-dragging" by holding ALT while they move a shape. This breaks the shape from any relationship it is part of, allowing it to be dragged free of all alignments. This is similar to Briar's constraint *ripping* feature [Gleicher and Witkin 1994]. While the ALT key is held the dragged shape will not try to attach itself to other guidelines either.

Additionally, we allow shapes to be freed from individual alignment relationships via a shape's context menu; the menu will show an item for each relationship that a shape is part of. Clicking the menu item breaks the shape from that single relationship, leaving the others intact, while leaving the shape stationary.

When use of the alignment tool causes multiple guidelines to be aligned they are merged, rather than being discarded and left on the diagram. This avoids some of the problems we saw in the first experiment which led to many extra guidelines being dropped and subsequently littering the page.

4.1.2 *Distribution.* The new distribution tool has the same basic behavior as other versions. The selected shapes are spaced equally between the outer two, and all shapes without associated guidelines will have guidelines created for them. Again, these guidelines will be the subject of the actual constraints controlling the distribution.

Rather than changing the color of guidelines, we have added a distribution indicator object as shown in Figure 16. This indicator can be used as a way of interacting with the distribution; it can be dragged to move the entire distribution and while it is selected it has a handle at each end, that can be clicked and dragged to resize the distribution. Deleting the indicator causes the distribution relationship to be removed, leaving the guidelines intact.

Deleting a guideline involved in a distribution created with either of the new tools, whether one-way or multi-way, has the same effect: the guideline is removed from the distribution and the remaining guidelines' positions are recalculated so that they remain equally distributed between the two outermost guidelines. If either of the outer guidelines is deleted, the distribution is not resized, but continues with

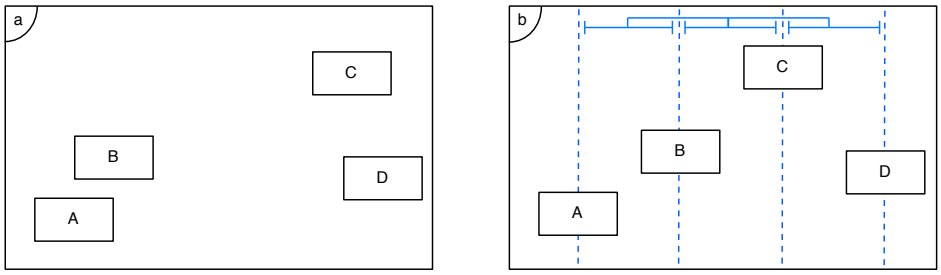

Fig. 16. Effect on layout due to multi-way distribution of all shapes. A distribution indicator object is created in addition to the four guidelines. This object represents the distribution relationship and may be used to interact with it.

one less guideline (the next outermost guideline becomes the outer guideline). Once there are less than three guidelines remaining in the distribution, the distribution relationship will itself be removed, though the remaining guidelines will not be removed.

For multi-way distributions, moving any distributed guideline or any shape moves the entire distribution, without resizing. This includes moving either of the outer guidelines. Distribution indicators are intended to be a physical on-screen representation of distribution relationships. For this reason, the size of distributions are controlled via handles on the distribution indicators, rather than through manipulation of the outermost guidelines.

Dragging a guideline involved in a one-way distribution breaks it from that distribution. The behavior of the distribution in this case is the same as for a deleted guideline. In multi-way distribution relationships, the user can intentionally break a guideline from a distribution by holding the ALT key as a guideline is dragged.

If any user action results in unsatisfiable constraints, that action is automatically undone and a dialog is shown to the user reporting the conflict.

All the tools are activated through a toolbar button and their options set with an associated dialog box containing buttons with icons representing the type of alignment/distribution to be created. The placement tools ignore connectors for the purpose of alignment and distribution.

4.2 Method

4.2.1 *Design.* Our second experiment was a more focused series of diagram manipulation/layout tasks in which the participants were given an initial diagram and then asked to modify this diagram in various ways. The starting diagram for a task was always the diagram the participant had constructed in the preceding task. The tasks were designed so they maximized the use of the tools.

The initial exercise in the experiment required the user to take an existing diagram without any existing constraints and to set up new placement relationships. The reason for this, especially in the case of the one-way group, was that the user should know that they had constructed all of the relationships they would later have to manipulate, so that they wouldn't feel we had set them up with deliberately difficult relationships.

In this study we measured the usefulness of the tools with the total exercise and individual component task completion times. We didn't consider the number of errors since during this study we stressed the importance of constructing a correct diagram rather than finishing quickly. This was done primarily so that subsequent tasks would have the correct starting diagram.

As with the first study, we used basic flowcharts as our diagram type for all exercises. Unlike the first study, the flowcharts had meaning and defined a real process—coffee making. Participants were not required to understand the meaning of the flowcharts. Still, we felt using realistic diagrams made the modification exercises a little more realistic and less contrived. To this end we gave a scenario for each exercise that described the participants' motivation for making the modifications to the diagram. These were such things as beautifying the diagram or fitting the existing diagram into a particular region of the page.

Participants in the study were randomly assigned to one of four groups. Each group was provided with a different set of constraint-based tools for alignment and distribution. The four groups were:

— **Group OW/DF—one-way, delayed feedback**: One-way alignment and distribution tools are available. Outlines showing the change in position of the selected objects are shown during dragging, but movement of objects connected by constraints is not shown until the mouse is released.

— **Group OW/IF—one-way, immediate feedback**: One-way alignment and distribution tools are available. Immediate feedback is shown during all interaction, including "live" changes to the actual objects being manipulated and the position of objects modified through constraints.

— **Group MW/DF—multi-way, delayed feedback**: Multi-way alignment and distribution tools are available. Outlines showing the change in position of the selected objects are shown during dragging, but movement of objects connected by constraints is not shown until the mouse is released.

— **Group MW/IF—multi-way, immediate feedback**: Multi-way alignment and distribution tools are available. Immediate feedback is shown during all interaction, including "live" changes to the actual objects being manipulated and the position of objects modified through constraints.

All participants were given the same exercises but only access to the particular tools offered to their group. The training differed slightly for each group to ensure participants knew how to use the tools available to them.

It was hypothesized that the persistent state of the relationships set up by the multi-way tools in Group MW would make them more usable than the one-way Group OW tools. We also hypothesized that the immediate feedback (IF groups) would be more usable than delayed feedback (DF groups) regardless of constraint type, because the participants could see the result of their interaction immediately. It is important to note that constraint solving for both the one-way and multi-way tools in Dunnart is fast enough to allow responsive direct manipulation when working with diagrams of the size used during the study.

4.2.2 *Participants.* Thirty-two people were tested; eight in each of the four groups. There were no requirements for participants other than that they be

computer-literate adults. All participants were university students who were native speakers and readers of English, with normal or corrected-to-normal vision. Participants were not reused across groups.

4.2.3 *Equipment.* All tests were carried out in private, the investigator testing a single participant at a time. The environment for the experiment was a private office in which the participant sat at a computer while the investigator sat behind them, observing and taking notes.

Interactions made by each user during their session were recorded via the logging mechanism of Dunnart so that their actions could be played back for reference purposes. Also collected was a log file for each test containing the times and type of every action the participant made in completing the exercises. From this we were able to accurately retrieve the start and finish times (the time of the first and last "action") for each exercise. Other notes taken by the investigator summarized the strategy and method taken by the user to carry out the task, as well as problems they experienced. These were used to prompt discussion during the debriefing.

Again short pre- and post-test surveys were used as a means of obtaining some additional qualitative and quantitative data about participants' experience with related tools, how difficult they found the exercise and suggestions they had for improving the tools.

At the beginning of each experiment participants were taken through a 25 minute scripted training exercise, that introduced them to Dunnart (referred to as "the editor"), described its basic features while requiring them to interact and experiment with it. The placement tools they were to use were explained and the training required them to set up and interact with these relationships. This was all conducted in an informal manner where the participant was able to interrupt the training and ask for clarification on specific points or spend a little more time on any aspect of the training they felt needed extra attention. When participants completed the training, had no further questions and indicated that they were comfortable to go on, they proceeded to the timed exercises.

4.2.4 *Materials.* In the exercises, participants were required to make layout changes to the diagrams—spacing the objects on the page or aligning them to make the diagram more aesthetically pleasing. The instructions and final diagrams showed alignment or distribution relationships, which we required participants to enforce in their final diagram.

There were three component tasks that were all part of a single large exercise, i.e., each task led on to the next and they were done one at a time, in fixed order. For the second and third component tasks the participant's own output from the previous task was used as their starting point. Emphasizing the importance of diagram correctness for the exercises helped minimize the effect this had on results. It was felt important that participants work with a diagram and set of constraints that they had set up themselves.

For each task, participants were given a three page instructional handout. The first page showed a typed description of the task including justification for the task as well as written instructions. The second page showed the target diagram, the result of applying the specified instructions to the initial diagram. The third page

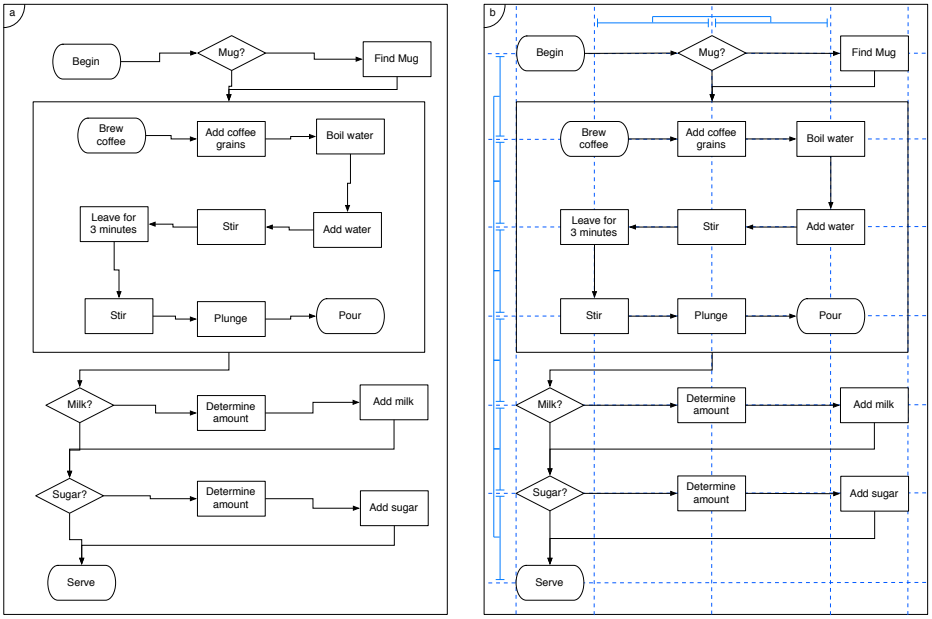

Fig. 17. Initial (a) and target (b) diagrams for the "Beautify" task. The task required participants to set up relationships, with minimal manipulation of constraints. The second diagram (b) was also the initial diagram for the "Reorient" task.

again showed the target diagram, this time in "print preview" mode, without any guidelines or distribution indicators visible. This was provided so that participants could check the paths of connectors and the overall structure of the diagram without the clutter of the alignment aids.

The three timed component tasks were:

— **"Beautify"**: This task required the participant to take an existing flowchart with no constraints and to add constraints to enforce multiple interconnected alignment and distribution relationships. It required little manipulation of constraints once they were created. The initial and target diagrams for this task are shown in Figure 17.

— **"Reorient"**: The aim of this task was to study user manipulation of existing constraint relationships. It required the participant to change the layout of the diagram without changing the relationships between objects. The initial diagram for this task is shown as Figure 17(b) and the target diagram for this task is shown in Figure 18(a).

— **"Rearrange"**: The primary aim of this task was to study removal and addition of objects to relationships (e.g., taking shapes from alignments and putting them in other alignments, taking guides from one distribution and putting them in another). There was also a smaller amount of manipulation of guidelines and distributions. Part of the purpose of this task was to determine whether the more per-

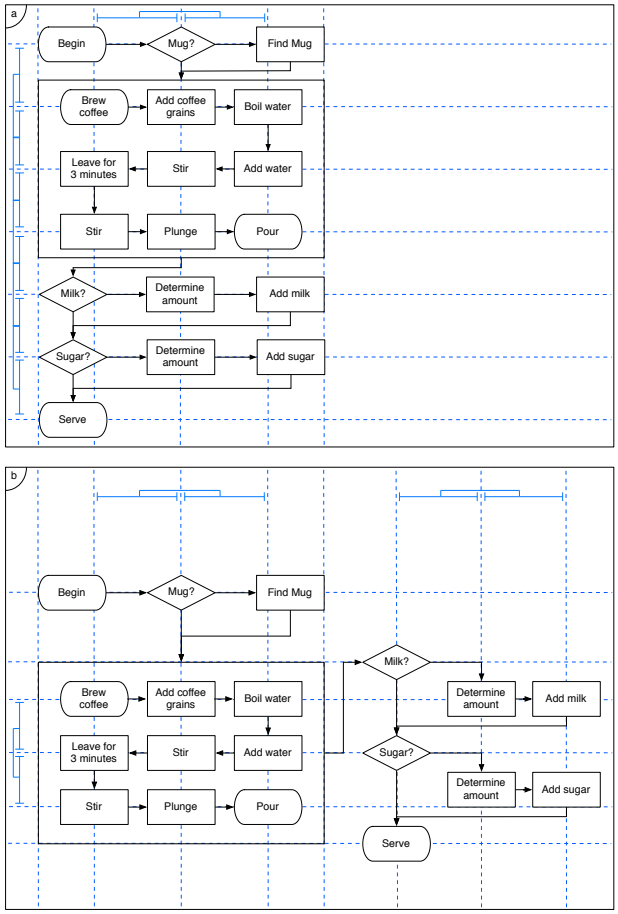

Fig. 18. Initial (a) and target (b) diagrams for the "Rearrange" task. The task required participants to remove and add objects to existing constraint relationships, along with some manipulation. The first diagram (a) was also the target diagram for the "Reorient" task, which required manipulation of existing constraint relationships.

sistent nature of multi-way constraints as compared to one-way constraints would make them less useful when placement relationships needed to be altered, or to have selected shapes broken from them. The initial and target diagrams for this task are shown in Figure 18.

4.3 Results

In this section we present and discuss the results of the usability study, examining the times taken to complete each task. We used the same statistical analysis methods as those used in Section 3.3. All participants completed the tasks.

The average completion times for each component task as well as for the complete

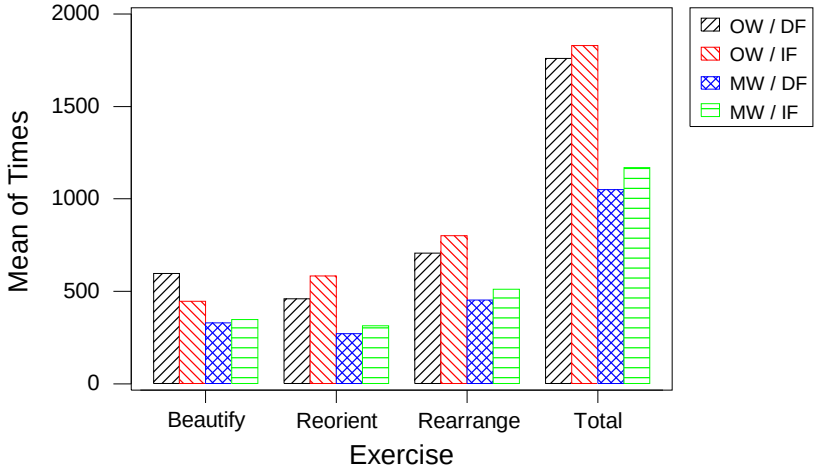

Fig. 19. Mean completion times (in secs.) for each exercise.

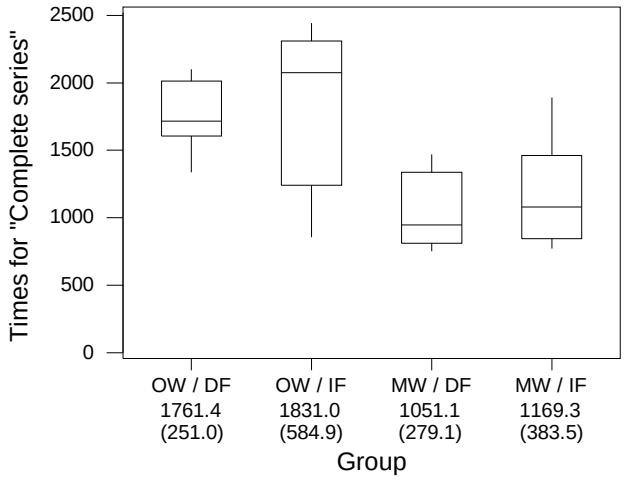

Fig. 20. Box-plot of completion times for the complete exercise.

exercise are shown in Figure 19. To determine where the statistical significance lies we perform an ANOVA for each.

The complete exercise series showed significant difference in completion times ($F = 8.12$, $p < 0.001$). Times for this exercise are summarized in Figure 20. Tukey's HSD test showed that there was significant difference between times for Group OW/DF and the multi-way tools, Group MW/DF and Group MW/IF, as well as significance between times for Group OW/IF and both multi-way tools. This shows that the multi-way constraints, with or without immediate feedback, were more beneficial than either of the one-way based groups. Rather surprisingly, it also shows that providing immediate feedback did not make a significant difference

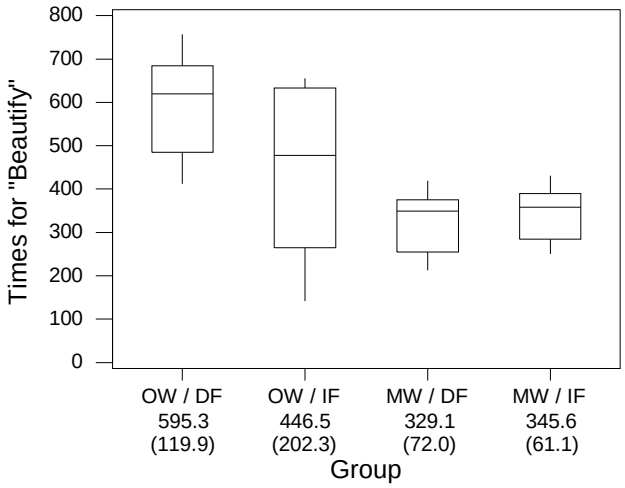

Fig. 21. Box-plot of completion times for "Beautify" task.

to completion times of users for either the one-way or the multi-way tools. Indeed, the trend is that immediate feedback increases completion time.

Next we consider the individual component tasks. In general these support the results for the overall exercise. A one-way ANOVA shows clear statistical significance to differences between groups in the first task "Beautify" (unequal group variances, $F = 7.46$, $p = 0.001$). Times for this task are summarized in Figure 21, a standard box-plot, as described in Section 3.3.

To see exactly where the significance lies we use Tukey's HSD test to consider all pairwise differences between group means. Using this method we find that the only significant differences are between Group OW/DF and Group MW/DF, as well as between Group OW/DF and Group MW/IF. In this task it is clear that the multi-way constraint-based tools of Group MW/DF and Group MW/IF definitely offer some benefit over the one-way constraints, without feedback, of Group OW/DF. There is no significant difference though between the multi-way tools and the one-way tools with feedback.

We again determine there to be significance in the completion times for the second task "Reorient" ($F = 5.04$, $p = 0.006$). Times for this task are summarized in Figure 22. For this task, using Tukey's HSD test, we find that the only significant differences are between Group OW/IF and Group MW/DF, as well as between Group OW/IF and Group MW/IF.

This task saw participants using existing placement relationships (set up in the previous task) to resize the diagram. We see that the multi-way tools of both Group MW/DF and Group MW/IF offer significant benefit over the one-way tools of Group OW/IF, though not those of Group OW/DF. The interesting result here is that it is the immediate feedback version of the tool that is significantly slower than the multi-way tools, rather than the non-feedback version, as we had expected. A possible explanation for this is presented at the end of this section.

The third task in the series, "Rearrange" also showed a significance in completion

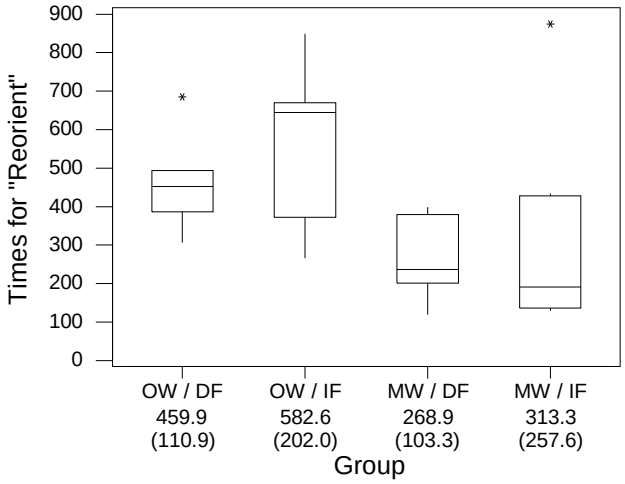

Fig. 22. Box-plot of completion times for "Reorient" task.

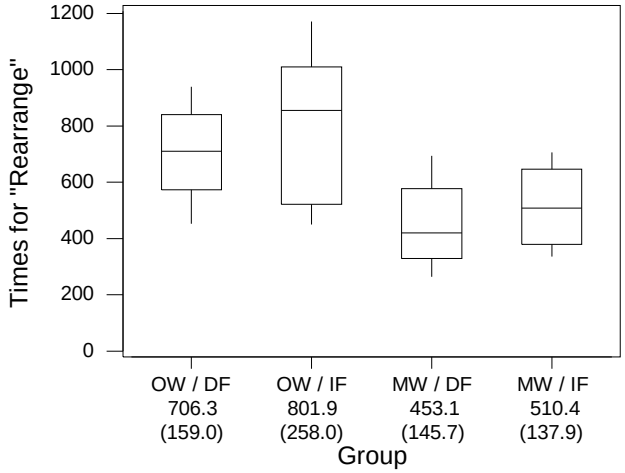

Fig. 23. Box-plot of completion times for "Rearrange" task.

times (unequal group variances, $F = 6.49$, $p = 0.002$). Times for this task are summarized in Figure 23. Tukey's HSD test showed that there was significance between times for Group OW/DF and Group MW/DF, and also between times for Group OW/IF and the multi-way tools, Group MW/DF and Group MW/IF. This shows that the multi-way constraints without feedback of Group MW/DF are more beneficial for manipulation and alteration of existing constraint-based placement relationships than either of the one-way based tools. In addition, the multi-way Group MW/IF tools are also significantly more beneficial than one-way feedback Group OW/IF tools.

We are also interested in whether there was any interference between the groups

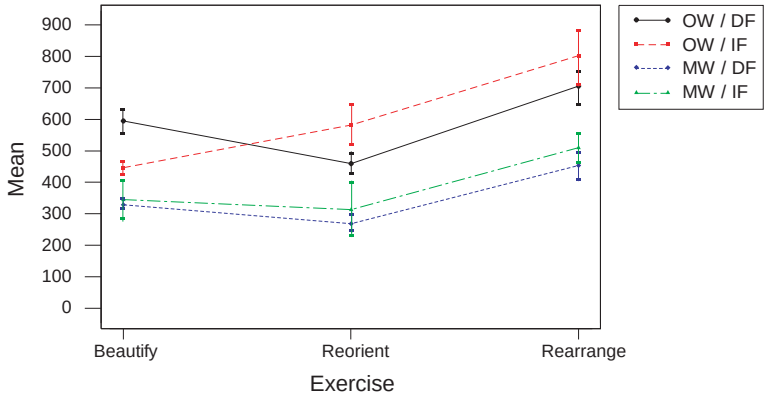

Fig. 24. Interaction plot for Groups OW/DF, OW/IF, MW/DF and MW/IF.

and the individual component tasks. Figure 24 shows group means as an interaction plot with error bars. An absence of interaction is illustrated by the consistent separation of the lines for the one-way lines versus the lines representing the multi-way group. Since each task was designed to test a different type of interaction with the constraint tools, this shows that the benefit of the multi-way tools compared with the one-way tools was not limited to a particular task. Ultimately, multi-way tools can be considered more beneficial in terms of task time than one-way tools.

Probably the most surprising result of our experiment was that there was no significant difference between the feedback version and the version without feedback of either type of tools in any of the component tasks. In fact, very surprisingly, the feedback versions performed worse than the non-feedback versions of the tools for all cases except the one-way tools in the first task (see the line crossing of Group OW/DF and Group OW/IF in Figure 24). This is likely because when setting up relationships and testing them, it is beneficial to have feedback for the one-way tools where relationships can break easily.

The two later tasks required considerably more movement of objects. This meant that participants often had to undo accidental actions or actions that inadvertently affected other constrained objects. We observed that when this happened participants with immediate feedback were more likely to undo the move by manually dragging objects back to their last position, rather than using the undo command. Since precisely placing objects by dragging is slow, whereas the undo command is instant, it would seem that the times for the feedback group could have been increased as a result of the feedback.

The version of Dunnart used during the study lacked an "escape" option, present in tools of many diagram editors that allows the user to cancel the current action, while it is in progress, by hitting the Escape key. While we are unable to determine exactly how many participants attempted to cancel actions by hitting escape (several mentioned the omission in the debriefing), we conjecture that the presence of such an option may have partially avoided the slowdown we observed from undo-dragging.

4.4 Discussion

Careful examination of the replays of the experiments revealed interesting information about participants' interaction with the tools as well as indications for possible improvements to the placement tools and to Dunnart itself. In particular, we examined actions with unexpected consequences. These were identified as actions made by the participant which caused an observed result and were undone immediately, or corrected manually with an opposite action. To add further strength to the argument that the participant expected a different result, these actions were often followed immediately by the same or similar attempt at the same action.

Reinforcing our hypothesis that the multi-way based tools are more usable than one-way based versions was the observation that only participants using the one-way tools unintentionally broke placement relationships, i.e., shapes becoming detached from guidelines or guidelines accidentally becoming detached from distributions. Even with our improved one-way tools this was still very frequent, happening between three and twelve times for *all* participants in the one-way group.

Roughly 60% of participants dragged an object and unexpectedly found one or more other shapes moving as a result. The breakdown of these participants was OW/DF: 4, OW/IF: 3, MW/DF: 6, and MW/IF: 6. As was expected, this was less common in the one-way groups where it could only happen when dragging placement indicators. In these groups dragging shapes always breaks them from relationships, and therefore never causes objects to be unintentionally dragged. The problem for one-way participants could sometimes be attributed to shapes accidentally being dropped and attached to a guideline, which became noticed later while moving the guideline. For the multi-way groups, where this problem was more frequent, participants had difficulty understanding exactly why a large group of shapes were moved as a result of moving a single object. They probably did not realize that relationships acted in chains—that moving a shape would move everything attached to it, and in turn everything that was attached to each of those secondary objects, and so on. This reaffirms previous claims of the difficulty in how best to communicate indirect connections between objects as a result of constraints. Further research is required in this area.

One common action was for participants to attempt to resize distributions by dragging their outermost guidelines. While this behavior is the behavior shown by Visio's tools, this alone cannot explain it since 62.5% of participants attempted this, and only roughly 20% of these had used Visio in the past. Less than one-third of these participants were from the immediate feedback groups. One explanation for this might be that participants better understood distribution indicator objects and their use from the training, due to the presence of the immediate feedback. Interestingly, participants who found the action didn't work with one outer guideline would often immediately undo and try to resize the distribution with the other outer guideline. Resizing via the outer guidelines is certainly behavior that could be implemented with a mix of constraints and code, at least for the standard case where there aren't overlapping distributions.

A final, unexpected observation was that 80% of participants had the expectation (confirmed by discussion in the debriefing) that the software would reason about the current positions of shapes and treat what appeared to be rows and columns

as groups, and align or distribute these accordingly. E.g., if there were six objects roughly in two columns, then applying a left alignment to the entire six objects would result in two alignment relationships rather than just the one. The breakdown of these participants was OW/DF: 6, OW/IF: 7, MW/DF: 6, and MW/IF: 7. Many participants attempted this repeatedly. This suggests that it could be worth adding a feature that allows automatic inference of constraints.

5. CONCLUSIONS

Despite the large amount of research in the area of constraints and graphical editors, there have been few if any formal studies to compare the usability of the various constraint-based systems that have been presented. In particular, there has been no investigation of the general claims that multi-way constraint-based tools are better than one-way constraint-based tools.

We have described two experiments comparing the usability of one-way and multi-way constraint-based alignment and distribution tools in diagram editors. The results from our two experiments provide strong support for our hypothesis that multi-way constraint-based placement tools are more usable than one-way constraint-based placement tools.

We believe the reason that the multi-way constraint-based placement tools are more usable than the one-way constraint-based placement tools is that one-way constraints have a fixed direction and an attribute can only have a single formula associated with it. This means that alignment and distribution relationships can silently break due to manipulation of objects involved in the relationship or because more than one constraint is applied to the same object.

Of course this is not to say that multi-way constraints are better than one-way constraints in other tools or other applications. For purposes in which the direction of constraint solving is fixed such as widget layout [Myers et al. 1990; Myers et al. 1997; McCormack et al. 2004] and more generally adaptive page layout [Weitzman and Wittenburg 1994; Hurst et al. 2003; Jacobs et al. 2004] or incrementally updating views of data [McDonald et al. 1990; Myers and Kosbie 1996], one-way constraints seem preferable to multi-way constraints because of their simplicity, efficiency and expressiveness.

In our second experiment we also investigated the impact of visual feedback provided during direct manipulation. We tested delayed feedback in which the position of objects connected by constraints to the selected objects being moved is not updated until the user has completed the action. We compared this with immediate feedback in which the user sees all changes to the diagram immediately during direct manipulation. Unexpectedly, providing immediate feedback appeared to slow users down. While this slowdown was not statistically significant it is interesting as it goes against the general belief that immediate feedback is purely beneficial [Gleicher and Witkin 1994; Ryall et al. 1997].

Comments solicited from participants in our study suggest that further research should focus on how to best represent constraints so that the diagram does not become too cluttered. Additional research should investigate how to best provide better visual and non-visual feedback during interaction to aid the user in comprehending complex systems of constraints and explaining unexpected interactions

between constraints. It should also examine the usefulness of automatic inference of placement relationships. We plan to examine all these issues for placement tools within diagram editors with future user studies.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their helpful feedback on an earlier draft of the article. We thank Laurent Tardif who helped in the design and development of the initial user study. We thank Bernd Meyer for his input into the initial experiment design and the other members of the Adaptive Diagrams Research Group for the useful comments and suggestions.

In addition, we thank the Department of Information Systems' Interaction Design Group at the University of Melbourne for the use of the IDEA lab, where the first usability study was conducted.

REFERENCES

Badros, G. J. 2000. Extending interactive graphical applications with constraints. Ph.D. thesis, Department of Computer Science and Engineering, University of Washington, Seattle, WA 98195-2350.

Bier, E. A. and Stone, M. C. 1986. Snap-dragging. In *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press, 233–240.

Borning, A., Marriott, K., Stuckey, P., and Xiao, Y. 1997. Solving linear arithmetic constraints for user interface applications. In *Proceedings of the 10th Annual ACM Symposium on User Interface Software and Technology*. ACM Press, 87–96.

Chok, S. S. and Marriott, K. 1998. Automatic construction of intelligent diagram editors. In *Proceedings of the 11th Annual ACM Symposium on User Interface Software and Technology*. ACM Press, 185–194.

Freeman-Benson, B. N., Maloney, J., and Borning, A. 1990. An incremental constraint solver. *Communications of the ACM 33,* 1, 54–63.

Fudos, I. 1995. Geometric constraint solving. Ph.D. thesis, Purdue University, Department of Computer Sciences.

Gleicher, M. 1992. Integrating constraints and direct manipulation. In *Proceedings of the 1992 Symposium on Interactive 3D Graphics*. ACM Press, 171–174.

Gleicher, M. 1993. A graphics toolkit based on differential constraints. In *Proceedings of the 6th Annual ACM Symposium on User Interface Software and Technology*. ACM Press, 109–120.

Gleicher, M. and Witkin, A. 1994. Drawing with constraints. *The Visual Computer: International Journal of Computer Graphics 11,* 1, 39–51.

Green, T. R. G. and Petre, M. 1996. Usability analysis of visual programming environments: A 'cognitive dimensions' framework. *Journal of Visual Languages and Computing 7,* 2, 131–174.

Heydon, A. and Nelson, G. 1994. The Juno-2 constraint-based drawing editor. Technical Report 131a, Digital Systems Research Center, 130 Lytton Avenue, Palo Alto, California 94301. Dec.

Hill, R. D. 1993. The Rendezvous constraint maintenance system. In *Proceedings of the 6th Annual ACM Symposium on User Interface Software and Technology*. ACM Press, 225–234.

Hower, W. and Graf, W. H. 1996. A bibliographical survey of constraint-based approaches to CAD, graphics, layout, visualization, and related topics. *Knowledge-Based Systems 9*, 449–464.

Hurst, N., Marriott, K., and Moulder, P. 2003. Cobweb: a COnstraint-Based WEB browser. In *ACSC '03: Proceedings of the 26th Australasian Conference on Computer Science*. Australian Computer Society, Inc., Darlinghurst, Australia, Australia, 247–254.

Igarashi, T., Matsuoka, S., Kawachiya, S., and Tanaka, H. 1997. Interactive beautification: a technique for rapid geometric design. In *Proceedings of the 10th Annual ACM Symposium on User Interface Software and Technology*. ACM Press, 105–114.

Jacobs, C., Li, W., Schrier, E., Bargeron, D., and Salesin, D. 2004. Adaptive document layout. *Communications of the ACM 47,* 8, 60–66.

Knuth, D. E. 1979. *TEX and METAFONT*. Digital Press.

Kramer, G. 1992. A geometric constraint engine. *Artificial Intelligence 58,* 1–3 (dec), 327–360.

Kurlander, D. and Feiner, S. 1993. Inferring constraints from multiple snapshots. *ACM Transactions on Graphics (TOG) 12,* 4, 277–304.

Marriott, K. and Chok, S. S. 2002. QOCA: A constraint solving toolkit for interactive graphical applications. *Constraints 7,* 3-4, 229–254.

McCormack, C., Marriott, K., and Meyer, B. 2004. Adaptive layout using one-way constraints in SVG. In *Proceedings of the 3rd Annual Conference on Scalable Vector Graphics (SVG Open)*.

McDonald, J. A., Stuetzle, W., and Buja, A. 1990. Painting multiple views of complex objects. In *Proceedings of the 1990 ACM Conference on Object-Oriented Programming: Systems, Languages, and Applications and the European Conference on Object-Oriented Programming*. Ottawa, Canada, 245–257.

Myers, B. A., Giuse, D. A., Dannenberg, R. B., Vander Zanden, B., Kosbie, D. S., Pervin, E., Mickish, A., and Marchal, P. 1990. Garnet: Comprehensive support for graphical, highly-interactive user interfaces. *IEEE Computer 23,* 11 (nov), 71–85.

Myers, B. A. and Kosbie, D. S. 1996. Reusable hierarchical command objects. In *CHI '96: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM Press, 260–267.

Myers, B. A., McDaniel, R. G., Miller, R. C., Ferrency, A. S., Faulring, A., Kyle, B. D., Mickish, A., Klimovitski, A., and Doane, P. 1997. The Amulet environment: New models for effective user interface software development. *IEEE Transactions on Software Engineering 23,* 6 (jun), 347–365.

Nelson, G. 1985. Juno, a constraint-based graphics system. In *SIGGRAPH '85: Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press, New York, NY, USA, 235–243.

Ryall, K., Marks, J., and Shieber, S. 1997. An interactive constraint-based system for drawing graphs. In *Proceedings of the 10th Annual ACM Symposium on User Interface Software and Technology*. ACM Press, 97–104.

Sannella, M., Maloney, J., Freeman-Benson, B. N., and Borning, A. 1993. Multi-way versus one-way constraints in user interfaces: Experience with the DeltaBlue algorithm. *Software—Practice and Experience 23,* 5 (May), 529–566.

Snodgrass, J. G., Levy-Berger, G., and Haydon, M. 1985. *Human Experimental Psychology*. Oxford University Press, New York.

Sutherland, I. E. 1963. Sketchpad: A man-machine graphical communication system. Ph.D. thesis, Massachusetts Institute of Technology.

Van Wyk, C. J. 1982. A high-level language for specifying pictures. *ACM Transactions on Graphics (TOG) 1,* 2, 163–182.

Vander Zanden, B. 1996. An incremental algorithm for satisfying hierarchies of multi-way dataflow constraints. *ACM Transactions on Programming Languages and Systems 18,* 1 (jan), 30–72.

Vander Zanden, B. T., Halterman, R., Myers, B. A., McDaniel, R., Miller, R., Szekely, P., Giuse, D. A., and Kosbie, D. 2001. Lessons learned about one-way, dataflow constraints in the Garnet and Amulet graphical toolkits. *ACM Transactions on Programming Languages and Systems (TOPLAS) 23,* 6, 776–796.

Weitzman, L. and Wittenburg, K. 1994. Automatic presentation of multimedia documents using relational grammars. In *MULTIMEDIA '94: Proceedings of the 2nd ACM International Conference on Multimedia*. ACM Press, 443–451.

Wybrow, M., Marriott, K., McIver, L., and Stuckey, P. J. 2003. The usefulness of constraints for diagram editing. In *Proceedings of the 2003 Australasian Computer Human Interaction Conference (OZCHI)*, S. Viller and P. Wyeth, Eds. Information Environments Program, University of Queensland, Queensland, Australia, 192–201.

Received March 2005; accepted April 2007