Reasoning with Diagrams: Final Report

John Howse, Richard Bosworth, Andrew Fish and John Taylor
Visual Modelling Group, University of Brighton

Peter Rodgers, Gem Stapleton and Simon Thompson
University of Kent, Canterbury

1 Introduction

The standard notation for modelling software systems is the Unified Modelling Language (UML). This consists of a suite of mainly diagrammatic notations. However, UML's constraint language, OCL, is entirely textual, making the notation less accessible to practitioners. The Reasoning with Diagrams project aims to build the foundation for more accessible notations for practitioners, building on and consolidating existing theoretical and practical work on diagrammatic reasoning. Work on reasoning about diagrams expressing logical or set-theoretical properties has a long history, which has been reinvigorated in the last decade. In seminal work, Shin demonstrated that diagrammatic reasoning systems could be provided with the logical status of sentential systems. Many other diagrammatic reasoning systems have since been developed; however, most of them are not sufficiently expressive to be used as practical software modelling languages.

The aim of this research project was to develop a framework to support reasoning with a combination of diagrammatic and textual constraint notations, suitable for use by practitioners. The specific objectives were:

1. To develop sound and complete systems of rules for individual notations.
2. To develop a framework to support reasoning about constraints expressed using a combination of notations.
3. To prototype a family of tools to support reasoning with a combination of notations.
4. To establish the feasibility of extending the formal framework and tools to handle dynamic constraints.

We have met and exceeded our major objectives, 1 and 3, and have met the other two objectives. The quality of the research produced on this project firmly places us as world leaders in our field.

The main diagrammatic notations considered on this project were *spider diagrams* and *constraint diagrams*. Constraint diagrams are designed to be a formal diagrammatic alternative to the OCL. The software tools developed in this project include a diagram editor, diagram generation and layout functionality, diagrammatic theorem provers, and translators. In this project theory and tooling are strongly interlinked. The tools are built on the specification given by the theory, but the development of tools has driven forward the theory, particularly in the theorem proving and diagram layout areas.

All the papers referenced in this report have been produced by project team members (or their research students) and are directly related to the project. No other citations are made in this report.

2 Key Advances & Supporting Methodology

2.1 Individual notations

In this subsection, we show how objective 1 has been met and exceeded. In order to meet objective 1, we needed to formalize the individual notations and define reasoning rules that give sound and, where possible, complete systems. In addition to doing this, we have, for example, established expressiveness results for some systems, thus showing that we have exceeded objective 1. Furthermore, we have developed more sound and complete reasoning systems than was envisaged in our original proposal.

Euler diagrams form the basis of spider and constraint diagrams. Various formalizations of Euler diagrams have been investigated [7, 15] indicating deficiencies in previous formalizations. We have formally defined corresponding regions in Euler diagrams [28], generalizing the approach Shin used for Venn diagrams. The notion of correspondence is used in defining the semantics of constraint diagrams. Euler diagrams can be built up by *nesting* one component into another; the concept of nesting has been formalized [18, 19]. Further syntax can be added to Euler diagrams that enables them to make statements more concisely in a less cluttered manner [30]. A measure of clutter in Euler diagrams is suggested in [31], in which a clutter reducing algorithm is presented. We have developed four sound and complete systems based on Euler diagrams augmented with shading [46], which will be discussed in section 2.3.3.

The formalization of *spider diagrams* has been finalized. We have developed a sound, complete and decidable reasoning system [29]. Proving that the individual rules are valid is, in some cases, not straightforward. The completeness proof strategy is to convert the premise and conclusion diagrams to a normal form and then reason about that normal form. This proof strategy is used for other systems that we have developed. We augmented the spider diagram language with constants and extended the reasoning rules accordingly to give a sound and complete system [47].

The expressiveness of spider diagrams has been determined to be equivalent to that of monadic first order logic with equality (MFOLe) [44, 50]. To show this equivalence in one direction, for each diagram we construct a sentence in MFOLe that expresses the same information. For the significantly more challenging converse we prove that there exists a finite set of models for a MFOLe sentence S that can be used to classify all the models for S. Using these classifying models, we construct a diagram expressing the same information as S. Spider diagrams are, therefore, more expressive than Shin's Venn-II system which is equivalent to MFOL (without equality). The approach that Shin used to prove this equivalence for Venn-II does not extend to the spider diagram case. Augmenting the spider diagram language with constants was shown not to increase expressiveness [43, 47].

Constraint diagrams extend spider diagrams by incorporating additional syntax to represent relations and explicit universal quantification. The formalization of constraint diagrams has been completed [9], but was more challenging than expected. The semantics of constraint diagrams are defined by translating them into first order predicate logic (FOPL) sentences. Constraint diagrams contain explicit existential quantification and universal quantification; in general, it is not possible to determine the order in which to read the quantifiers, sometimes rendering a diagram ambiguous. This ordering problem was solved by augmenting the language with *reading trees*, essentially a partial order on the quantifiers, to disambiguate the diagrams [8, 9]. Furthermore, the reading tree provides additional information that is essential for the construction of the FOPL sentence used to interpret the diagram. The tree determines where the brackets are placed and, in conjunction with the diagram, the scope of the quantifiers.

Each constraint diagram has many reading trees, the set of which can be determined from a *dependence graph* [9]. There are some circumstances where a quantifier must be in the scope of other quantifiers. It is this information that is captured by the dependence graph, from which the set of reading tress can be automatically generated [10]. We have defined a set of sound rules for constraint diagrams augmented with reading trees and produced a strategy for developing further rules [6].

To simplify the notation for users, a default reading for constraint diagrams was proposed [11] and an evaluation of its intuitiveness was undertaken [13, 14]. We concluded from this study that the proposed default reading did not match every user's intuitive interpretation. A comparison, using cognitive dimension-like techniques, of constraint diagrams with a visualization of OCL, called VisualOCL, has been performed [12]. This study concluded that, in comparison with VisualOCL, constraint diagrams are very good at expressing set theoretic and relational properties.

Two sound, complete and decidable fragments of the constraint diagram language have been developed [39, 41, 42]; these are the first sound and complete systems produced for significantly expressive fragments of the constraint diagram language. The diagrams in these fragments do not require reading trees, but still include relational information and one of them includes explicit universal quantification. Some of the reasoning rules for these two systems extend those defined for spider diagrams. Many additional rules are also defined to give complete systems. The proofs of completeness for these systems are difficult and complex [39, 41]. Whilst the basic strategy is similar to that used to prove completeness in the spider diagram system [29], the details are challenging. Constraint diagrams augmented with reading trees are more expressive than both of these fragments.

A survey of reasoning systems based on Euler diagrams has been produced [40], placing the systems produced in this project into context. In all of the reasoning systems we have developed, we make a distinction between the concrete syntax (the drawn diagrams) and the abstract syntax (a mathematical abstraction of concrete diagrams); this distinction is not evident in purely textual logics. All of the reasoning takes place at the abstract level, for reasons presented in [15, 26]. We have defined mappings between the concrete and abstract syntax [26, 29, 39]. Using an abstract syntax brings with it, importantly, a level of precision and rigour that is not present in diagrammatic systems developed by others.

2.2 Heterogeneous Systems

In this subsection, we show how objective 2 has been met. In order to meet objective 2, we needed to formalize a framework that allows the use of heterogeneous notations.

A flexible, modular framework for modelling with heterogeneous notations has been developed and formalized [20]. This framework is extensible, allowing users to mix notations and incorporate their own notations and forms the basis for representing diagrams joined using logical connectives.

We can use this framework to build a sound and complete heterogeneous reasoning system based on spider diagrams and MFOLe. The work presented in [50] provides an algorithmic method for converting between MFOLe sentences and spider diagrams. This algorithm together with the reasoning rules for spider diagrams [29] and for first order logic give a sound and complete system. Furthermore, we can use this framework to build sound and complete heterogeneous reasoning systems incorporating FOPL and any of the diagrammatic languages that we have developed.

A new hybrid language, called visual first order logic (VFOL), which mixes textual (symbolic) and diagrammatic notations, has been defined [48]. VFOL modifies and extends the constraint diagram language. A sound and complete reasoning system for VFOL has been developed and the language has been shown to be equivalent in expressive power to FOPL [49]. There is an algorithmic method for converting a statement made in VFOL into a statement made in FOPL and vice versa. We can use this algorithm to provide a sound and complete hybrid system based on VFOL and FOPL.

2.3 Tools

In this subsection, we show how objective 3 has been met and exceeded. In order to meet objective 3, we needed to prototype a family of tools to support reasoning with the notations considered in this project. In addition to implementing such tools, we have put effort into enhancing the functionality of the tools. For example, the work discussed below on improving the layout of diagrams is not necessary to achieve objective 3, but significantly improves the quality of the tools we have produced. The software produced in the project was developed in Java to ensure platform independence and is open source; it is available both on SourceForge.net and at http://www.cs.kent.ac.uk/projects/rwd/.

2.3.1 Diagram editor

A diagram editor has been built. For formal diagrammatic modelling and reasoning to be taken up in industry, good tool support is essential. As a result, a significant amount of effort on the project went into developing a robust and effective editor. The editor provides diagram drawing facilities, editing, cut and paste, and zooming functionality for all the diagrammatic notations developed in this project. Diagrams can be laid out automatically and stored in XML format.

Users can access the reasoning functionality in the tool from the editor. The interface provides access to theorem provers and allows users to write their own proofs. Tableaux [36] give users a way of visualizing the meaning of a particular diagram, by showing the ways that a diagram can be satisfied. In particular tableaux provide decision procedures for diagram satisfiability and validity. Dependence graphs and reading trees for constraint diagrams can be generated by the editor. In order to support this varied functionality the software provides sophisticated support for representing and modifying both abstract diagrams (without layout information) and concrete diagrams (with layout information). In addition, translations between diagrammatic and textual representations have been implemented. A framework has been developed for unifying translations from constraint diagrams to other notations by implementing a mapping from constraint diagrams to abstract syntax trees. These trees can then be converted into FOPL or OCL (or other notations) using KMF (see section 2.3.4).

2.3.2 Drawing Infrastructure

The layout of diagrams is of fundamental importance to diagrammatic reasoning systems. Previous to the work in this project, the appearance of diagrams used within such logical frameworks had to be hand generated. The drawing infrastructure produced in this project allows abstract descriptions of diagrams to be automatically converted to a concrete representation on a screen. This automated drawing facility is used in two important areas. First, diagrams that are the result of applying reasoning rules can be visualized without a user having to draw the diagram. Secondly, users can be aided in developing effective layouts for their drawings.

The automatic layout of spider and constraint diagrams posed several non-trivial challenges. The problem of generating concrete Euler diagrams from abstract descriptions is hard. This problem has been solved by the project team: concrete diagrams can be generated subject to some well-formedness conditions [16, 17]. During 2004, Stirling Chow (University of Victoria, Canada) visited the project team. Whilst at Kent, he adapted his Euler diagram embedding mechanism for use in the project's software. This mechanism relaxed our well-formedness conditions, enabling the drawability of some abstract diagrams that we did not allow using our approach. However, his mechanism is not robust enough to draw all of the abstract diagrams that can be drawn using our approach. The integration of these two mechanisms enhances the diagram generation framework. This framework was further enhanced by utilizing the theory developed on nested diagrams [19].

The automatically generated Euler diagrams are typically not very readable and can be visually unattractive. Hence, a function was implemented to make the diagrams more usable by modifying their layout, whilst maintaining their abstract syntax [23].

The diagrammatic notations developed in this project are Euler diagrams with additional structure. In particular, graphs are embedded in the diagrams. Further layout work introduced a force based method for laying out graphs in Euler diagrams [32], enabling the layout of spider and constraint diagrams. Furthermore, a key application of the layout work is to visualize sequences of diagrams, such as proofs. For this application, it is desirable to make subsequent diagrams look as similar as possible to previous diagrams. A mechanism to achieve this was implemented [37].

The first research on empirically validating the layout characteristics of Euler diagrams was performed in this project [4]. This study resulted in the conclusions that smoothing contours, avoiding contour lines that are close together and equalizing areas within diagrams is likely to improve users understanding of Euler diagrams. The study also tentatively concluded that the relative importance of these characteristics is in the order in which they are listed above.

2.3.3 Automated theorem proving

We have implemented and evaluated an automated theorem prover that uses four Euler diagram reasoning systems [46]. The theorem prover uses heuristics to guide it through the search space to find shortest proofs. We empirically evaluated the theorem prover in terms of time taken to find a shortest proof, using each of the rule sets. From this evaluation, we conclude that in order to find a shortest proof most quickly, the rule set used is dependent on the proof task [46]. Euler diagrams are the underlying notation of many diagrammatic reasoning systems (not just those developed by the project team). Therefore, this work on automated reasoning lays the foundations for efficient proof searches to be conducted in many other diagrammatic systems.

For spider diagrams, a direct proof writing algorithm can be extracted from the completeness proof strategy given in [29]. We improved and implemented this proof writing algorithm including functionality to produce counter examples whenever there is no proof [24]. The proofs produced by this algorithm can sometimes be unnecessarily long. In [22] (best paper at Diagrams 2004) we utilized the A^* search algorithm to produce shortest proofs in a fragment of the spider diagram language. This work has been extended to the full spider diagram language [21]. We have suggested extensions to the spider diagrams syntax that may assist the theorem prover when searching for proofs [45].

Further reasoning rules specifically for constraint diagrams augmented with reading trees have also been implemented. This functionality allows users to construct their own proofs interactively. We are currently extending this to a fully automated constraint diagram theorem prover.

2.3.4 KMF: the Kent Modelling Framework

A set of tools has been built to support model driven software development. At the core of the KMF [1, 2, 3, 33, 34, 35] is KMFStudio, a tool to generate modelling tools from the definition of modelling languages expressed as metamodels. KMFStudio is supported by two Java libraries, OCLCommon and OCL4KMF, which allow dynamic evaluation of OCL constraints. These libraries provide built-in support for checking well-formedness of models for tools generated using KMFStudio. KMF also includes a Java library that implements the XMI standard.

2.4 Case studies and dynamic constraints

A case study [27] has been developed to model a video rental store, a variation on the traditional library model, showing how constraint diagrams can be used to specify software systems. The case study uses a schema notation to specify operation pre-conditions and post-conditions. Constraint diagrams are used within this schema notation, showing that they can handle dynamic constraints; thus we have achieved the main goal of objective 4.

3 Project Plan Review

The project proceeded broadly according to the plan in the proposal. We put more effort than planned on formalizing notations and developing reasoning rules. In part this was due to underestimating the difficulty in formalizing the semantics of constraint diagrams but also because we developed more reasoning systems than originally planned. Furthermore, we widened the scope of the project and performed a deeper analysis of the theoretical issues than was envisaged. Much of the work on implementing reasoning tools was shared between Kent and Brighton, rather than primarily performed at Kent, due to the specialist expertise at both sites. The work on developing viewers and editors was considerably extended, due to the perceived importance of having a usable visualization mechanism and a robust editor for the notations. Case studies were developed throughout the project and some empirical and analytical evaluation was performed towards the end of the project.

3.1 Staff Issues

At the start of the project we appointed Octavian Patrascoiu as RA at Kent and Dr Andrew Fish as RA at Brighton. The RAs originally named on the proposal did not take up their posts; this was because they had already accepted other posts before funding for this project was confirmed.

The Kent Principal Investigator, Dr Stuart Kent, left for a job with Microsoft as Senior Project Manager on the Visual Studio project in April 2003 and was replaced as PI by Dr Peter Rodgers. Dr Kent's main focus of expertise in the project was in the area of Model Driven Development. This change resulted in more emphasis being placed on diagram layout than would otherwise have been the case.

Brighton co-investigator, Dr Jean Flower, left for a job with Autodesk as a Software Engineer working on geometric modelling in October 2004 and was replaced as co-investigator by Richard Bosworth, who worked mainly on tool development. Flower is a Honorary Faculty Research Fellow at Brighton and has continued to collaborate with the project team.

Patrascoiu left for a job in industry in February 2005. Due to the short timescale left on the project it was not considered feasible to appoint a new researcher to contribute to the software effort. Instead, Dr Gem Stapleton was appointed as the RA at Kent in February 2005. She was a research student under the supervision of Professor Howse and Dr Taylor at Brighton and was closely involved in the project. She also worked with the Kent co-investigator, Professor Simon Thompson. She has contributed to a substantial number of the key research papers resulting from the work on the project.

Other academics contributed to research on the project. Professor Sun-Joo Shin (Yale) worked with the team on the importance of distinguishing between abstract and concrete syntax in diagrammatic systems [26]. Dr Manuel Barrio Solórzano (University of Valladolid) worked on implementing a spider diagram system in PVS [38]. Dr Judith Masthoff (University of Aberdeen) worked on the generation of diagrammatic proofs [21, 46, 22] and on empirical evaluation of constraint diagram default readings [13, 14]. Jane Southern (University of Brighton) worked on evaluating the theorem prover [46]. Dr Gabriele Taentzer (Technical University of Berlin) worked on a comparison of constraint diagrams with VisualOCL [12]. Dr Florence Benoy (University of Kent) worked on the empirical validation of layout [4]. Chris John, a research student supervised by Howse, Taylor and Fish (replacing Flower), worked on measuring clutter in diagrams [30, 31]. Dr Bernd Meyer (Monash University, Australia) worked on methods of reducing clutter in Euler diagrams [25]. Dr David Akehurst (University of Kent) worked on the software infrastructure for the reasoning tools [2, 3]. Professor Steve Schuman (University of Brighton) worked on case studies [27]. Paul Mutton, a research student of Rodgers, worked on the layout of diagrams [23, 32, 37]. Stirling Chow (University of Victoria, Canada) visited both the Brighton and Kent sites with an EPSRC Visiting Researcher grant (GR/T28874), on which Rodgers was PI and Flower and Howse were co-investigators; he worked on area-proportional Venn and Euler diagrams [5]. All the collaborations listed above are ongoing.

4 Research Impact and Benefits to Society

Professor Alan Bundy (University of Edinburgh), in his nomination of Dr Gem Stapleton's PhD thesis [39] for the British Computer Society Distinguished Dissertation Award 2005, for which she was a runner-up, said: "This work is important because diagrams of this kind are the most popular specification language used by computer systems designers, e.g. via UML. Formalization provides a theoretical underpinning for such diagrammatic specification languages. It is needed to ensure that the languages are unambiguous, and to provide automated support tools, e.g. for verification, transformation and synthesis. It will assist ICT systems designers to build dependable and maintainable systems, and help reduce the plague of ICT failures with which we have all become painfully familiar."

By extension this quotation refers to the work of the whole project as does this quotation from a reviewer of her thesis for this award: "This work is likely to represent a benchmark or exemplar for any researcher attempting a similar formalization of a diagrammatic reasoning system or language. As such this thesis is comparable to that of Sun-Joo Shin, which has become a seminal work in this field. Similarly, it is to be hoped that a work of this clarity and quality will encourage those numerous developers, and future developers, of diagrammatic languages who have not seriously considered or attempted even lightweight formalizations of their semantics to make such an attempt."

Attention to detail and rigour is fundamental to all the work in this project. This aspect of our work is frequently noted by reviewers of our papers; international recognition of our work places us as world leaders in our field. We have raised the standard of research in this field. For example, a reviewer of one of our papers states: "This is a model for theoretical work on diagrammatic notations."

The software from the project has been made open source and is available on SourceForge.net. The notations and tools developed on this project can be used to formally model software systems diagrammatically. The automated reasoning facility within the tools provides support for verification and validation of models. Prior to their development the only tool support available for formal modelling was for textual notations. Our novel work on diagram generation and layout is vital to the reasoning functionality; this layout work also has many applications outside the scope of this project, for example in data representation, that could be revolutionized by the theory and techniques that we have developed.

We have developed new logical systems together with new formalization techniques and proof strategies. These techniques and strategies may well be applicable to other systems.

The project team has been instrumental in setting up an international workshop series on Euler diagrams. The first workshop was held during September 2004 in Brighton, and was chaired by Rodgers and Flower. It attracted delegates from seven countries. The second workshop was held in Paris during September 2005 and plans for the third workshop are underway. One outcome from the workshop series is a fruitful collaboration with Professor Frank Ruskey's group at the University of Victoria on Euler diagram generation and layout.

Patrascoiu was part of a team that integrated the OCL implementation into the Eclipse development tool as part of an IBM Faculty Partnership Award. He also helped to organize the Tool Support for OCL and Related Formalisms: Needs and Trends Workshop at Model Driven Engineering Languages and Systems, 2005; the OCL and Model Driven Engineering Workshop at UML 2004; and the Second European Workshop on MDA, EWMDA-2, 2004.

The results from the project have been disseminated widely at conferences, in journals, by delivering seminars and by direct personal contact. We have targeted research groups working in our general area, but have also spread our results more widely to broaden the research impact of our work.

We have disseminated our work by delivering seminars or by personal contact to teams at the following universities: Monash, Australia; Victoria, Canada; Technical University of Madrid and Valladolid, Spain; Middle East Technical University, Turkey; Yale, USA; Aberdeen, Cambridge, Edinburgh, Open University, St Andrews, and Sussex, UK.

Members of the team have presented research from the project at the following international conferences and workshops: Diagrams, Visual Languages and Human-Centric Computing, Applications of Graph Transformations with Industrial Relevance, Graph Transformations and Visual Modelling Techniques, Information Visualization, Visual Languages and Computing, UML, Metamodelling for MDA, Enterprize Distributed Object Computing, Euler Diagrams, Visual Languages and Formal Methods, and Computing: the Australasian Theory Symposium. Many of these conferences have a typical acceptance rate of around 30%. We were awarded best paper at Diagrams 2002 [17] and Diagrams 2004 [22], and best paper in the Visual Languages and Formal Methods strand at Human-Centric Computing 2003 [8].

We have published in the following journals: Visual Languages and Computing, Software and Systems Modeling, Logic and Computation, and LMS Journal of Computation and Mathematics. We are awaiting the outcome of papers submitted to the following journals: Automated Reasoning, Formal Aspects of Computing, Symbolic Logic, and Theoretical Computer Science.

5 Explanation of Expenditure

The travel budget was spent on project meetings and conference attendance. We had regular six-monthly major project meetings which alternated between sites and many smaller meetings involving travel between sites. These meetings were very successful. Project management at each site and overall was very effective. The equipment and consumables budgets were small and the money was used to provide equipment for the RA's and other provisions.

6 Further Research & Dissemination

In addition to the published output and dissemination activities from the project itself, the project team is engaged in a number of follow-on activities to develop further the work of the project and to disseminate the results. Stapleton was awarded two research fellowships: an Early Career Fellowship from the Leverhulme Trust and the other by the Royal Commission for the Exhibition of 1851. Stapleton's fellowship proposals built directly upon the work of the project. She accepted the Leverhulme Fellowship which runs from September 2005 to August 2007 held at Brighton. The University of Brighton has extended Fish's contract for a year. The retention of Stapleton and Fish assists the project team in maintaining its world leading status.

We are currently developing follow-on research proposals. A joint proposal from the project team, Generating Euler Diagrams, will be submitted to EPSRC soon; a joint project between Brighton and Aberdeen (Dr Kees van Deemter) on Hybrid Generation of diagrams and text is in preparation; and another proposal from the project team, on Rigorous Domain Specific Modelling, is at a preliminary stage as is another joint project between Brighton and Aberdeen (Dr Judith Masthoff) on Automated Reasoning in Diagrammatic Systems.

Another outcome of the project is a new collaboration between the Brighton group and Professor Peter Cheng at the University of Sussex. This has resulted in the joint supervision (by Howse and Cheng) of a research student, starting in 2006, considering the usability of the diagrammatic notations.

The 2006 IEEE Symposium on Visual Languages and Human-Centric Computing will take place in Brighton with Howse as General Chair.

7 Conclusion

We have developed formal diagrammatic modelling notations that are sufficiently expressive to be used in software specification on an industrial scale. The tools implemented in this project are a major advance towards providing sufficient support for the use of these notations in industry.

This project has been highly successful. We have met all of our objectives and exceeded most of them. The quality of the research produced on this project firmly places us as world leaders in our field.

References

[1] D. Akehurst, S. Kent and O. Patrascoiu. A Relational Approach to Defining and Implementing Transformations in Metamodels. Software and Systems Modeling, 2(4) pp 215–239, 2003.

[2] D. Akehurst and O. Patrascoiu. OCL.2.0: Implementing the standard for Multiple Metamodels. Proc. OCL2.0: Industry standard or scientific playground? Workshop, UML'03, ENTCS 102, pp 21–41, 2003.

[3] D. Akehurst and O. Patrascoiu. Tooling metamodels with patterns and OCL. Proc. Metamodelling for MDA Workshop, 2003.

[4] F. Benoy and P. Rodgers. Evaluating the comprehension of Euler diagrams. Proc. Euler Diagrams 2005.

[5] S. Chow and P. Rodgers. Constructing area-proportional Venn and Euler diagrams with three circles. Proc. Euler Diagrams 2005.

[6] A. Fish and J. Flower. Investigating reasoning with constraint diagrams. Proc. Visual Languages and Formal Methods, ENTCS 127, pp 53–69, 2004.

[7] A. Fish and J. Flower. Abstractions of Euler diagrams. Proc. Euler Diagrams 2004, ENTCS 134, pp 77–101, 2005.

[8] A. Fish, J. Flower, and J. Howse. A reading algorithm for constraint diagrams. Proc. Human Centric Computing, pp 161–168, 2003.

[9] A. Fish, J. Flower, and J. Howse. The semantics of augmented constraint diagrams. Journal of Visual Languages and Computing 16, pp 541–573, 2005.

[10] A. Fish and J. Howse. Computing reading trees for constraint diagrams. Proc. AGTIVE '03, Applications of Graph Transformations with Industrial Relevance, pp 260–274, 2003.

[11] A. Fish and J. Howse. Towards a default reading for constraint diagrams. Proceedings of Diagrams 2004, LNAI 2980, pp 51–65, 2004.

[12] A. Fish, J. Howse, G. Taentzer, and J. Winkelmann. Two visualisations of OCL: A comparison. Technical Report, University of Brighton, 2005. `www.cmis.brighton.ac.uk/research/vmg/publications`.

[13] A. Fish and J. Masthoff. Do monkeys like elephants or do elephants like monkeys? Technical Report, University of Brighton, 2005. `www.cmis.brighton.ac.uk/research/vmg/publications`.

[14] A. Fish and J. Masthoff. An experimental study into the default reading of constraint diagrams. Proc. Visual Languages and Human Centric Computing, pp 287–289, 2005.

[15] A. Fish and G. Stapleton. Defining Euler diagrams: choices and consequences. Proc. Euler Diagrams 2005.

[16] J. Flower, A. Fish, and J. Howse. Euler diagram generation. Submitted to Journal of Theoretical Computer Science, 2005.

[17] J. Flower and J. Howse. Generating Euler diagrams. Proc. Diagrams 2002, LNAI 2317, pp 61–75, 2002.

[18] J. Flower, J. Howse, and J. Taylor. Nesting in Euler diagrams. Proc. Graph Transformation and Visual Modeling Techniques, pages 99–108, 2002.

[19] J. Flower, J. Howse, and J. Taylor. Nesting in Euler diagrams: syntax, semantics and construction. Software and Systems Modelling 3 pp 55–67, 2004.

[20] J. Flower, J. Howse, J. Taylor, and S. Kent. A visual framework for modelling with heterogeneous notations. Proc. Human Centric Computing, pp 71–73, 2002.

[21] J. Flower, J. Masthoff, and G. Stapleton. Generating proofs with spider diagrams using heuristics. Proc Visual Languages and Computing, pp 279–285, 2004.

[22] J. Flower, J. Masthoff, and G. Stapleton. Generating readable proofs: A heuristic approach to theorem proving with spider diagrams. Proc. Diagrams 2004 LNAI 2980, pp 166–181.

[23] J. Flower, P. Rodgers, and P. Mutton. Layout metrics for Euler diagrams. Proc. Information Visualisation, pp 272–280, 2003.

[24] J. Flower and G. Stapleton. Automated theorem proving with spider diagrams. Proc. Computing: The Australasian Theory Symposium (CATS'04) ENTCS 91, pp 116–132, 2004.

[25] J. Howse and B. Meyer. Conjunction labels in Euler diagrams. Proc. Euler Diagrams 2004.

[26] J. Howse, F. Molina, S-J. Shin, and J. Taylor. On diagram tokens and types. Proc. Diagrams 2002, LNAI 2317, pp 146–160, 2002.

[27] J. Howse and S. Schuman. Precise visual modelling. Journal of Software and Systems Modeling 4, pp 310–325, 2005.

[28] J. Howse, G. Stapleton, J. Flower, and J. Taylor. Corresponding regions in Euler diagrams. Proc Diagrams 2002, LNAI 2317, pp 146–160, 2002.

[29] J. Howse, G. Stapleton, and J. Taylor. Spider diagrams. LMS J. Computation and Mathematics, 8 pp 145–194, 2005.

[30] C. John. Reasoning with projected contours. Proc. 2004, LNAI 2980, pp 147–150, 2004.

[31] C. John. Projected contours in Euler diagrams. Proc. Euler Diagrams 2004, ENTCS 134, pp 103–126, 2005.

[32] P. Mutton, P. Rodgers and J. Flower. Drawing graphs in Euler diagrams. Proc. of Diagrams 2004, LNAI 2980, pp 66–81, 2004.

[33] O. Patrascoiu. Mapping EDOC to Web Services using YATL Proc. Enterprise Distributed Object Computing Conference (EDOC 2004), 2004.

[34] O. Patrascoiu. YATL: Yet Another Transformation Language. Proc. 1st European MDA Workshop, MDA-IA, pp 83-90, 2004

[35] O. Patrascoiu and P. Rodgers. Embedding OCL Expressions in YATL Proc. OCL and Model Driven Engineering Workshop, UML'04, 2004.

[36] O. Patrascoiu, S. Thompson, and P. Rodgers. Tableaux for diagrammatic reasoning. Proc. Visual Languages and Computing, pp 279–286, 2005.

[37] P. Rodgers, P. Mutton, and J. Flower. Dynamic Euler diagram drawing. Proc. Visual Languages and Human Centric Computing, pp 147–156, 2004.

[38] M. Barrio Solórzano and J. Howse. Theorem proving in spider diagrams using PVS. Proc. Euler Diagrams 2004.

[39] G. Stapleton. *Reasoning with Constraint Diagrams*. PhD thesis, University of Brighton, 2004. To be published by BCS.

[40] G. Stapleton. A survey of reasoning systems based on Euler diagrams. Proc. Euler Diagrams 2004, ENTCS 134, pp 127–151, 2004.

[41] G. Stapleton, J. Howse, and J. Taylor. A constraint diagram reasoning system. Proc. Visual Languages and Computing, pp 263–270, 2003.

[42] G. Stapleton, J. Howse, and J. Taylor. A decidable constraint diagram reasoning system. Journal of Logic and Computation. To appear, 2005.

[43] G. Stapleton, J. Howse, J. Taylor, and S. Thompson. The expressiveness of spider diagrams augmented with constants. Proc. Visual Languages and Human Centric Computing, pp 91–98, 2004.

[44] G. Stapleton, J. Howse, J. Taylor, and S. Thompson. What can spider diagrams say? Proc. 2004, LNAI 2980, pp 112–127, 2004.

[45] G. Stapleton, J. Howse, and K. Toller. On spiders' feet. Proc. Euler Diagrams 2005.

[46] G. Stapleton, J. Masthoff, J. Flower, A. Fish, and J. Southern. Automated theorem proving in Euler diagrams systems. Submitted to Journal of Automated Reasoning, 2005.

[47] G. Stapleton, J. Taylor, J. Howse, and S. Thompson. The expressiveness and completeness of spider diagrams augmented with constants. Submitted to Formal Aspects of Computing, 2004.

[48] G. Stapleton, S. Thompson, A. Fish, J. Howse, and J. Taylor. A new language for the visualization of logic and reasoning. Proc. Visual Languages and Computing, pp 263–270, 2005.

[49] G. Stapleton, S. Thompson, A. Fish, J. Howse, and J. Taylor. Visual first order logic. Submitted to Journal of Symbolic Logic, 2005.

[50] G. Stapleton, S. Thompson, J. Howse, and J. Taylor. The expressiveness of spider diagrams. Journal of Logic and Computation, 14(6) pp 857–880, 2004.