

Structured graphs

a visual formalism for scalable graph based tools
& its application to software structured analysis

Submitted by Mark J. Sifer to meet the requirements of the Doctor of Philosophy in the
School of Computing Sciences at the University of Technology, Sydney in 1996.

CERTIFICATE

I certify that this thesis has not already been submitted for any degree and is not being submitted as part candidature for any other degree.

I also certify that the thesis has been written by me and that any help that I have received in preparing this thesis, and all sources used, have been acknowledged in this thesis.

Signature of Candidate

Production Note:

Signature removed prior to publication.

Acknowledgement

I am indebted to my principal supervisor John Potter for both his guidance and collaboration in this work. I also wish to thank my co-supervisor John Leaney for his support. Some of the concepts in the second half of this thesis had their genesis while I was a research student at Hiroshima University, under the supervision of Tadao Ichikawa.

This work has also benefited from discussions and collaborations with Peter Eades, Masahito Hirakawa, David Lowe, and former colleagues in the Computer Systems Engineering Division of Computer Sciences Corporation (Australia). I also wish to thank the thesis examiners for their detailed comments. Addressing these, has significantly improved the clarity and presentation of the thesis. The work was funded by an Australian government postgraduate award with a top-up from the School of Computing Sciences.

Abstract

Very large graphs are difficult for a person to browse and edit on a computer screen. This thesis introduces a visual formalism, *structured graphs*, which supports the scalable browsing and editing of very large graphs. This approach is relevant to a given application when it incorporates a large graph which is composed of named nodes and links, and abstraction hierarchies which can be defined on these nodes and links.

A typical browsing operation is the selection of an arbitrary group of nodes and the display of the network of nodes and links for these nodes. Typical editing operations is: adding a new link between two nodes, adding a new node into the node hierarchy, and moving sub-graphs to a new position in the node hierarchy. These operations are scalable when the number of user steps involved remains constant regardless of how large the graph is. This thesis shows that with structured graphs, these operations typically take one user step.

We demonstrate the utility of structured graph formalism in an application setting. Computer aided software engineering tools, and in particular, structured analysis tools, are the chosen application area for this thesis, as they are graph based, and existing tools, though adequate for medium size systems, lack scalability.

In this thesis examples of an improved design for a structured analysis tool, based on structured graphs, is given. These improvements include scalable browsing and editing operations to support an individual software analyst, and component composition operations to support the construction of large models by a group of software analysts.

Finally, we include proofs of key properties and descriptions of two text based implementations.

Contents

Certificate	ii
Acknowledgement	iii
Abstract	iv
1 Introduction and Review	1
1.1 Limitations of current Graph Models and Visual Formalisms	1
1.1.1 Interacting with large graphs on a computer	2
1.1.2 Graph models	2
1.1.3 Visual formalisms	4
1.1.4 The need for a new visual formalism	5
1.2 Limitations of current graph based tools	5
1.2.1 Hypertext tools	6
1.2.2 CASE tools	6
1.2.3 Structured analysis tools	12
1.3 Levelled Data Flow Diagrams as the basis of a new formalism	14
1.4 Structured graphs	15
1.5 Discussion	16
1.6 Thesis overview	16

Part I Structured Graphs

2 Viewing a Structured Analysis Model	18
2.1 The model	18
2.2 Browsing and editing the model	20
2.3 Discussion	27
3 Browsing and Editing Structured Graphs	28
3.1 Network Abstraction	29
3.2 Browsing Models	35
3.3 Incomplete Models	38
3.4 Editing Models	40
3.5 Model Limitations	41
3.6 Summary	45
4 Ordered Sets : Background	46
4.1 Standard Terminology	46
4.2 View Orders	49
5 Structured Graph Formalism	53
5.1 Structured Graphs	54
5.2 Compact and Abstract Models	54

5.3 Properties	57
5.4 Model Editing	58
5.5 Model Viewing	58
5.6 Limitations	59
5.7 Discussion	61

Part II Structured Graph Components

6 Building with Structured Analysis Components	62
6.1 Example Components	62
6.2 The composed model	67
6.3 Data flow meshing	69
6.4 Viewing a component	71
6.5 Discussion	72
7 Typed Link Orders	73
7.1 Tree typed link orders	74
7.2 General typed link orders	75
7.3 Link schema constraints	77
7.4 Summary	81
8 Component Composition	82
8.1 Limitations of model composition	83
8.2 Components	84
8.3 A component composition example	85
8.4 Component merge	86
8.5 Component composition	87
8.6 A component limitation	91
9 Typed Order Formalism	93
9.1 Labelled orders	93
9.2 Typed orders	96
9.3 Typed tree schema	97
9.4 Typed order schema	97
10 Structured Graph Component Formalism	101
10.1 Pre-components	101
10.2 The merge operator and components	102
10.3 The component mesh operator	105
10.4 The component composition operator and proper components	109

11 Conclusion	110
11.1 The contribution of part one: structured graphs	110
11.2 Limitations of structured graphs	111
11.3 Future work for part one	112
11.4 The contribution of part two: structured graph components	112
11.5 Limitations of structured graph components	113
11.6 Future work for part two	113
11.7 A component schema example	113
11.8 Final Discussion	115
Bibliography	116
Appendices	123
A Glossary	123
B Structured graph proofs	125
C Gofer implementation of structured graphs	131
D A console based structured graph tool	160
