

Enquiring and Reasoning Over Diagrams Using the Semantic Web

By
Zaineb BEN FREDJ

A thesis submitted in partial fulfilment of the requirements of
Oxford Brookes University
for the degree of
Doctor of Philosophy

Oxford Brookes University
Department of Computing
School of Technology
Wheatley Campus
Oxford, OX33 1HX, UK

May 2011

Abstract

This thesis is a contribution towards the representation of diagrams in a way that improves the ability to enquire and reason about the information on which the diagram is based. This research is directed towards the acquisition of the information behind carefully selected “formal diagrams”. The hypothesis underlying the proposed approach is that “if information on the structure and the semantics of formal diagrams were preserved, made ‘part of the diagram’ by willing authors at the creation stage, these diagrams would be more perceivable, operable and understandable and, as a result, suggest enhanced accessibility benefits for such diagrams”.

The main contribution of this research is the introduction of a novel approach called Graphical Structure Semantic Markup Languages. GraSSML reveals that “if the information of the diagram is made part of the diagram at the creation stage, it is possible to represent diagrams in a way that improves the ability to enquire and reason about the information on which the diagram is based, thus making diagrams more perceivable, operable and understandable and, as a result, suggesting enhanced accessibility benefits for such diagrams”. Graphical content is no longer thought of as ink on the paper or pixels on the screen but as the intrinsic structure and semantics of the information.

Three different evaluations, each assessing GraSSML from a different perspective, have been carried out. A technical perspective evaluation demonstrated the feasibility and applicability of GraSSML. The development and implementation of a full functional proof-of-concept prototype demonstrated GraSSML for two different types of diagrams selected from two different application domains presented as use cases. A methodology to apply the GraSSML conceptual architecture to a “formal diagram” in a given application domain has been developed. An author’s perspective evaluation provided information on the viability and applicability of the methodology developed for the application of GraSSML conceptual architecture. The application of the methodology has been tested, independently, by a third party during a double blind trial. And finally a user’s perspective evaluation, carried out with three sighted users and two blind users against a set of predefined tasks, demonstrated the perceivability, operability and understandability of diagrams using GraSSML and as a result validated the hypothesis.

To the loving memory of my father, Ali Ben Fredj
To my extraordinary mother, Faouzia Ben Fredj
To my eight sisters and two brothers
To my husband, Kamel Heus and two daughters,
Hannah and Jasmine

Acknowledgments

I would like to thank here some of the people without whose help and support this thesis would never have been completed. These people have been helpful in their support and encouragement even during the most difficult periods.

I owe my deepest gratitude to my supervisor Professor David Duce for his help, advice and encouragement. I am deeply thankful to him for his guidance and support from the initial to the final stages of this research.

I am also grateful to Professor Bob Hopgood and Dr Mary Zajicek, for their help, guidance, support and encouragement.

I would like to express my gratitude to Elizabeth Maynard for being there, helping and supporting me during the whole process.

To David Lightfoot, who guided me and supported me towards and during my studies at Oxford Brookes University.

Thanks should also go to staff in the Computing Department of Oxford Brookes University, and in particular to Samia Kamal, Anne Becker, Ken Brownsey, Fareena Salih, Rosemary Ostley and Professor Denise Morrey.

To my friends who were here when I needed them the most: Patricia Vilain, Pauline McKneow, Lamine Ouyahia, Dunya Bendahoud, Musbah Sagar, Emna Menif and Caroline Berard.

To my special parent who always encouraged me to study and never give up. I wish my father could have seen that I have completed it.

I thank my eight sisters (Beya, Soundes, Naila, Nozah, Samia, Akbel, Aida and Fattouma) and my two brothers (Mohamed-ali and Essamadi), without whom life wouldn't be the same.

Not least, thanks to my husband Kamel Heus, for loving and supporting me in all my endeavours, and to my lovely daughter Hannah for giving me the love, support and willingness never to give up, and finally to my new daughter Jasmine, who has kept me awake whenever she could to ensure the completion of the thesis.

Contents

Abstract	2
Acknowledgments.....	4
Contents.....	5
List of Figures	10
Introduction	13
1.1 Terminology	15
1.2 Motivation	16
1.3 Aims and Research objectives.....	18
1.4 Research Contribution	24
1.4.1 The GraSSML Conceptual Architecture	24
1.4.2 Fully working prototype	24
1.4.3 Methodology to apply the GraSSML approach.....	25
1.4.4 Credibility of GraSSML.....	25
1.5 Structure of the thesis	26
1.6 Publications	27
1.7 External Support.....	30
1.8 Originality.....	30
Diagrammatic Representations.....	31
2.1 Nature and importance of diagrams	31
2.1.1 Graphics.....	31
2.1.2 Diagrams.....	32
2.2 Graphicacy.....	36
2.3 Syntax, Semantics and Pragmatics of Diagrams	36
2.4 Taxonomies of visual representations	37
2.4.1 Blackwell Taxonomy of taxonomies.....	37
2.4.2 Lohse's Taxonomy	38
2.4.3 Blenkhorn and Evans's taxonomy.....	39
2.4.4 Takagi and Tatsuya's taxonomy.....	40
2.4.5 A Synthesised Taxonomy Selection of Diagrams	42
2.5 Diagrammatic Communication	43
2.6 Generating diagrams.....	45

2.6.1 A Presentation Tool (APT).....	45
2.6.2 BOZ	47
2.6.3 TRIP2	47
2.6.4 AVE.....	48
2.6.5 OntoDiagram	49
2.6.6 SemViz	49
2.6.7 PIC	50
2.7 Discussion and Conclusion.....	51
Accessible Diagrams: A Critical Review	52
3.1 Diagrams on the Web	52
3.1.1 Graphic formats	52
3.1.2 Guideline 1.1 and the WCAG 2.0 POUR Principles.....	54
3.2 Current approaches to presenting accessible diagrammatical content	55
3.2.1 Existing alternative modalities to access diagrams non-visually	55
3.2.2 Description of some current Bottom-up approaches	58
3.2.3 Limitations of Bottom-up approaches	65
3.2.4 Summary of issues.....	66
3.3 Diagrams processing	68
3.3.1 Visual processing of diagrams.....	69
3.3.2 Issues of Non-visual presentation of diagrams.....	70
3.4 Accessible Diagrams Requirements	73
3.4.1 Loss of information at diagram rendering	73
3.4.2 Lack of accessibility support for both image and vector formats	74
3.4.3 Presentation: modality issues	74
3.5 Discussion and Conclusion.....	75
The GraSSML Approach.....	78
4.1 Terminology	78
4.2 Approach	79
4.3 Conceptual architecture	81
4.4 Semantic level	82
4.4.1 Ontology	82
4.4.2 Data Model	83
4.5 Structure level: ZineML	84
4.6 Presentation level.....	87

4.6.1 Graphical representation.....	89
4.6.2 Textual representation	89
4.6.3 Query systems	90
4.6.4 Other representations.....	91
4.7 Transformations.....	91
4.7.1 Data model to ZineML	92
4.7.2 ZineML to Representation.....	92
4.8 GraSSML in practice	93
4.8.1 Initiation	95
4.8.2 Applying GraSSML.....	96
4.8.3 The authoring process.....	97
4.9 Conclusion.....	98
Evaluation: A Technical Perspective	101
5.1 GraSSML Prototype Architecture	101
5.1.1 Stage 1: Ontology	104
5.1.2 Stage 2: Data model.....	104
5.1.3 Stage 3: Notational conventions.....	106
5.1.4 Stage 4: Graphical Representation	108
5.1.5 Stage 5: Verbalisation model (Semantics)	112
5.1.6 Stage 6: Verbalisation model (Structure)	112
5.1.7 Stage 7: Query systems	112
5.1.8 Graphical User Interface.....	113
5.2 Use Case: Hierarchical Diagram “Organisation Charts”	114
5.2.1 Organisation Charts	114
5.2.2 Main Example	116
5.2.3 More examples	130
5.2.4 Hierarchical diagrams in different domains	132
5.3 Use Case: Process Diagram “UML Activity Diagram”	135
5.3.1 UML Activity Diagram	135
5.3.2 Main Example	135
5.3.3 Process diagrams in different domains.....	155
5.4 Conclusion.....	157
Evaluation: An Author’s Perspective	158
6.1 Introduction	159

6.1.1 The GraSSML FCP project	159
6.1.2 Objectives of the GraSSML FCP project	160
6.2 Evaluation methodology.....	161
6.3 GraSSML FCP.....	163
6.3.1 Class of diagrams: “Charts”	163
6.3.2 The GraSSML FCP approach.....	165
6.3.3 Semantic Level	168
6.3.4 Structure level.....	170
6.3.5 Presentation level.....	172
6.4 Results	178
6.5 Conclusion.....	180
Evaluation: A User’s Perspective.....	181
7.1 Heuristic evaluation.....	182
7.1.1 Participants	183
7.1.2 Method.....	183
7.1.3 Results	184
7.1.4 Conclusion.....	187
7.2 Blind users Evaluation.....	187
7.2.1 Participants	187
7.2.2 Equipment.....	188
7.2.3 Materials	188
7.2.4 Tasks.....	195
7.2.5 Methods	197
7.2.6 Results and discussion.....	200
7.3 Conclusion.....	202
Conclusion and Future Work.....	205
8.1 Main findings of the research	205
8.2 Contributions	205
8.2.1 The GraSSML Conceptual Architecture	206
8.2.2 Fully working prototype	207
8.2.3 Methodology to apply the GraSSML approach.....	207
8.2.4 Credibility of GraSSML.....	208
8.3 Open issues and future work	208
8.4 Conclusion.....	211

References	213
Graphical Formats on the Web.....	224
A.1 Raster formats.....	224
A.1.1 Graphic Interchange format (GIF).....	225
A.1.2 Portable Network Graphics (PNG).....	226
A.1.3 Joint Photographic Experts Group (JPEG).....	226
A.2 Vector formats	226
A.2.1 Computer Graphics Metafile (CGM or WebCGM)	227
A.2.2 Scalable Vector Graphics (SVG).....	227
A.2.3 Small Web Format (SWF).....	231
Accessibility.....	234
B.1 Definition	234
B.2 Legal Frameworks	235
B.3 Accessibility Guidelines	236
B.4 Blindness.....	237
B.5 Screen Readers.....	238
Accessibility of Diagrams.....	240
C.1 Diagrams on the Web	240
C.1.1 Diagram inserted as vector images	240
C.1.2 Diagram inserted as vector graphics.....	240
C.2 Guidelines for diagrams accessibility	241
C.2.1 WCAG 2.0 – Guideline 1.1 Text Alternatives.....	241
C.2.2 Limitations of Text Alternatives.....	242
C.2.3 Screen Readers and diagram accessibility	244
Accessibility of Diagrams and the WCAG 2.0	246
D.1 The POUR Principles of the WCAG 2.0.....	246
D.1.1 Perceivable.....	246
D.1.2 Operable.....	250
D.1.3 Understandable	251
D.1.4 Robust.....	251
D.2 Existing approaches in the context of the defined requirements	251

List of Figures

FIGURE 1: EXAMPLES OF “PROCESS DIAGRAMS”	21
FIGURE 2: EXAMPLES OF “HIERARCHICAL DIAGRAMS”	22
FIGURE 3: EXAMPLES OF “CHARTS”	23
FIGURE 4: THESIS STRUCTURE	26
FIGURE 5: GRASSML TIMELINE ALONG SIDE W3C TIMELINE OF W3C RECOMMENDATIONS	29
FIGURE 6: CATEGORIZATIONS OF COLLECTED GRAPHICS (TAKAGI AND TATSUYA, 2007)	41
FIGURE 7: DIAGRAMMATIC COMMUNICATION BASED ON (NARAYANAN, 1997)	45
FIGURE 8: TRIP2 MODELS (TAKAHASHI ET AL., 1991).....	48
FIGURE 9: A SIMPLE EXAMPLE « DO YOU SEE WHAT I MEAN? ».....	66
FIGURE 10: GRASSML APPROACH.....	80
FIGURE 11: GRASSML SYSTEM OVERVIEW.....	81
FIGURE 12: GRASSML CONCEPTUAL ARCHITECTURE.....	82
FIGURE 13: EXAMPLES OF BASIC SHAPES DEFINED BY ZINEML	86
FIGURE 14: GRASSML SYSTEM ARCHITECTURE	94
FIGURE 15: GRASSML PROTOTYPE ARCHITECTURE	103
FIGURE 16: POSSIBLE SOURCES OF INFORMATION OF THE RDF DATA MODEL	104
FIGURE 17: CREATION OF THE RDF GRAPH USING FORMS IN PROTÉGÉ	105
FIGURE 18: CREATION OF THE RDF GRAPH USING GRAPH WIDGET IN PROTÉGÉ	105
FIGURE 19: BASED ON ILOG JVIEWS DIAGRAMMER TOOL CHAIN	109
FIGURE 20: ARCHITECTURE OF JVIEWS DIAGRAMMER	109
FIGURE 21: SYMBOL EDITOR FOR UML ACTIVITY DIAGRAM EXAMPLE	111
FIGURE 22: ORGANISATION CHART EXAMPLE	115
FIGURE 23: CLASS HIERARCHY FOR VISIONS ORGANISATION CHART ONTOLOGY	116
FIGURE 24: VISIONS ORGANISATION CHART	117
FIGURE 25: CREATION OF THE INDIVIDUALS USING GRAPH WIDGET	118
FIGURE 26: ZINEML CODE FOR THE VISIONS ORGANISATION CHART EXAMPLE.....	119
FIGURE 27: VISIONS ORGANISATION CHART WITH DRAWING DIRECTION “RIGHT”	120
FIGURE 28: VERBALISATION MODEL TEMPLATE FOR ORGANISATION CHARTS	121
FIGURE 29: "MENU" AND “GENERAL DESCRIPTION” SECTIONS	122
FIGURE 30: "DETAILED PROSE DESCRIPTION" SECTION	123
FIGURE 31: "DETAILED LIST DESCRIPTION" SECTION.....	123
FIGURE 32: TEXTUAL REPRESENTATION OF THE STRUCTURE (MENU AND GENERAL DESCRIPTION SECTIONS).....	124
FIGURE 33: TEXTUAL REPRESENTATION OF THE STRUCTURE (DETAILED PROSE DESCRIPTION SECTION)	125
FIGURE 34: TEXTUAL REPRESENTATION OF THE STRUCTURE (DETAILED LIST DESCRIPTION SECTION) ..	125
FIGURE 35: SPARQL QUERY "WHO MANAGES DIRECTLY...?".....	126

FIGURE 36: RESULT OF QUERY “WHO MANAGES DIRECTLY DANIELLE PEPIN”	127
FIGURE 37: RESULT OF QUERY “WHO MANAGES INDIRECTLY DANIELLE PEPIN”	128
FIGURE 38: RESULT OF QUERY “WHO MANAGES DIRECTLY DANIELLE PEPIN”	129
FIGURE 39: RESULT OF QUERY “WHO MANAGE INDIRECTLY DANIELLE PEPIN”	129
FIGURE 40: OXFORD BROOKES UNIVERSITY ORGANISATION CHART EXAMPLE	131
FIGURE 41: OXFORD BROOKES UNIVERSITY EXAMPLE (DIRECTION=“RIGHT”).....	132
FIGURE 42: PROBABILITY TREE DIAGRAM EXAMPLE	133
FIGURE 43: WEBSITE STRUCTURE EXAMPLE.....	133
FIGURE 44: GENETIC FAMILY TREE EXAMPLE.....	134
FIGURE 45: UML AD ENROLLING IN THE UNIVERSITY FOR THE FIRST TIME	136
FIGURE 46: HIERARCHY OF THE CLASSES OF THE UMLAD ONTOLOGY	137
FIGURE 47: OBJECT PROPERTIES OF THE UMLAD ONTOLOGY	137
FIGURE 48: DATATYPE PROPERTIES OF THE UMLAD ONTOLOGY	137
FIGURE 49: TEMPLATE EXTRACTED FROM (TAYLOR, 2008B)	141
FIGURE 50: VERBALISATION MODEL TEMPLATE FOR UML ACTIVITY DIAGRAMS	141
FIGURE 51: TEXTUAL REPRESENTATION OF THE SEMANTICS (GENERAL DESCRIPTION SECTION).....	142
FIGURE 52: TEXTUAL REPRESENTATION OF THE SEMANTICS (DETAILED PROSE DESCRIPTION SECTION)	142
FIGURE 53: TEXTUAL REPRESENTATION OF THE SEMANTICS (DETAILED LIST DESCRIPTION SECTION) .	143
FIGURE 54: TEXTUAL REPRESENTATION OF THE STRUCTURE (GENERAL DESCRIPTION SECTION)	144
FIGURE 55: TEXTUAL REPRESENTATION OF THE STRUCTURE (DETAILED PROSE DESCRIPTION SECTION)	144
FIGURE 56: TEXTUAL REPRESENTATION OF THE STRUCTURE (DETAILED LIST DESCRIPTION SECTION)..	145
FIGURE 57: TEXTUAL QUERY SYSTEM FROM VIEW MODE INTERFACE	146
FIGURE 58: COMPLETING SELECTION OF AN OPTION	147
FIGURE 59: SELECTING OPTION FOR “ENROL IN UNIVERSITY”	147
FIGURE 60: RESULT OF QUERY “TRAVERSE GRAPH FROM ... TO ...”	148
FIGURE 61: SELECTION OF QUERY “SHOW ME ALL ELEMENTS OF TYPE...” WITH OPTION SELECTED “ACTIVITY”	149
FIGURE 62: RESULTS OF THE QUERY “SHOW ME ALL ELEMENTS OF TYPE ACTIVITY”	149
FIGURE 63: SELECTION OF QUERY “SHOW ME PATH TO FINAL ACTIVITY FROM...” WITH OPTION SELECTED “INITIAL ACTIVITY”	150
FIGURE 64: RESULTS OF QUERY “SHOW ME PATH TO FINAL ACTIVITY FROM ...”	151
FIGURE 65: ACTIVITY DIAGRAM EXAMPLE “BOWLING”	152
FIGURE 66: NESTED ACTIVITY DIAGRAM EXAMPLE	153
FIGURE 67: EXAMPLE OF SUB-ACTIVITIES	153
FIGURE 68: STARUML EXAMPLE “JUKEBOX”	154
FIGURE 69: A SIMPLE FLOWCHART EXAMPLE	155
FIGURE 70: ANOTHER EXAMPLE OF A FLOWCHART WITH DIFFERENT NOTATIONAL CONVENTIONS	155
FIGURE 71: EXAMPLE OF A FLOWCHART FOR COMPUTING FACTORIAL N (N!).....	156

FIGURE 72: BAR CHART EXAMPLE.....	164
FIGURE 73: COLUMN CHART EXAMPLE.....	164
FIGURE 74: PIE CHART EXAMPLE.....	165
FIGURE 75: THE GRASSML FCP CONCEPTUAL ARCHITECTURE	165
FIGURE 76: BALANCE SHEET EXAMPLE USED.....	166
FIGURE 77: THE GRASSML FCP SYSTEM ARCHITECTURE.....	167
FIGURE 78: DATATYPE PROPERTIES OF GRASSML FCP ONTOLOGY	168
FIGURE 79: OBJECT PROPERTIES OF THE GRASSML FCP ONTOLOGY	168
FIGURE 80: HIERARCHY OF THE CLASSES OF THE GRASSML FCP ONTOLOGY	169
FIGURE 81: COLUMN CHART STRUCTURE ELEMENT	170
FIGURE 82: FGML CODE SAMPLE	171
FIGURE 83: FGML “EVOLUTION OF THE CASH” EXAMPLE.....	172
FIGURE 84: COLUMN CHART GENERATED USING GRASSML FCP.....	173
FIGURE 85: TEXTUAL PRESENTATION OF FIGURE 84	174
FIGURE 86: TEXTUAL PRESENTATION OF FIGURE 84 (TABULAR VERSION).....	174
FIGURE 87: BALANCE SHEET ASSETS SUBSET EXAMPLE	175
FIGURE 88: SMART PIE CHART “NET CURRENT ASSETS FOR COMPANY A IN 2002” EXAMPLE	177
FIGURE 89: SMART PIE CHART “WHO IS SMALLER...” QUERY	178
FIGURE 90: SMART PIE CHART “WHO ACCOUNTS FOR...”	178
FIGURE 91: UMLAD FOR ATTENDING A COURSE LECTURE (DIAGRAM 1).....	189
FIGURE 92: UMLAD FOR EATING WHEN HUNGRY (DIAGRAM 2).....	190
FIGURE 93: UMLAD ENROLLING IN THE UNIVERSITY FOR THE FIRST TIME (DIAGRAM 3)	191
FIGURE 94: MY SIMPLE ORGANISATION CHART (DIAGRAM 4)	192
FIGURE 95: OXFORD BROOKES UNIVERSITY ORGANISATION CHART EXAMPLE (DIAGRAM 5)	193
FIGURE 96: VISION ORGANISATION CHART (DIAGRAM 6)	194
FIGURE 97: EXAMPLE OF SIMPLE MODIFICATION IN SVG.....	230
FIGURE 98: SCREEN READER USAGE (WEBAIM, 2009A).....	238

Chapter 1

Introduction

The World Wide Web plays an important role in modern society which thrives on information. Computer graphics is an important component of the web in conveying information via diagrams, maps, visualisations, illustrations, etc. There exists a wide variety of kinds of graphics (photographs, painting, maps, diagrams, symbols, etc.), each of them using different notations and carrying different kinds of information. Different graphics are indeed accessed and looked at in different ways.

This research will be restricted to the study of diagrammatic representations, which are structured visual representations of concepts and relationships between these concepts. Diagrams offer powerful advantages in presenting, accessing and processing information.

Until quite recently, browser support for graphics was restricted to image file formats such as GIF, PNG and JPEG. Today, most browsers also provide support for one or more vector graphics formats such as SWF, SVG and CGM. Despite the importance of vector graphics and the recent innovative advances in web technology, relatively little attention has been paid to making information of this kind available to a wide range of people accessing it in a wide range of situations.

The need for accessibility support on the web was recognised ten years ago and is now a legal requirement. Accessibility is not only for people with obvious disabilities but also for people who simply access information and learn in different ways (SLOAN et al., 2006). The mobile Web is a good example that illustrates this idea (HARPER et al., 2006).

According to the Web Content Accessibility Guidelines 2.0 (WCAG 2.0) (W3C, 2008c), four principles (POUR) provide the foundation for Web Accessibility: Perceivable (P), Operable (O), Understandable (U), and Robust (R). These principles *“lay the foundation necessary for anyone to access and use Web content”*. These four principles apply perfectly to visual web content such as diagrams but the actual guidelines defined, does not really indicate the requirements to make this type of content POUR.

This thesis intends to contribute to the representation of diagrams in a way that could improve the ability to enquire and reason about the information on which the diagram is based and thus making diagrams more perceivable, operable and understandable and, as a result, suggesting enhanced accessibility benefits for such diagrams. This information could then be presented, explored and adapted in a multimodal POU environment. Note that this research will address three of the principles defined by the WCAG 2.0. Graphical content is no longer thought of as ink on the paper or pixels on the screen but more as an abstract entity that has an intrinsic structure and semantics.

This research is directed towards the acquisition of the information behind carefully selected “formal diagrams”.

The hypothesis underlying the proposed approach is that “if information on the structure and the semantics of formal diagrams were preserved, made ‘part of the diagram’ by willing authors at the creation stage, these diagrams would be more perceivable, operable and understandable and, as a result, suggest enhanced accessibility benefits for such diagrams”.

The proposed approach named Graphical Structure Semantic Markup Languages (GraSSML) is as follows:

1. Large classes of diagrams (vector graphics) are defined with a clear set of rules and symbols from which they are created.
2. Such diagrams are defined via a data model conforming to an ontology and rule set that is mapped down onto a logical description of the vector graphics that is not constrained by final form rendering and layout constraints.
3. This logical description is then mapped to an appropriate final form for the graphical representation on the output device.
4. Perceivability, operability and understandability support is provided by alternative transformations of the semantic and logical representations to alternative final forms (e.g. text or aural equivalents) and query facilities that allow the end user to explore, enquire and reason over the information content of the diagram.

1.1 Terminology

The following terms are used throughout this thesis with these definitions:

- ❖ **GraSSML:** name of the proposed approach aiming at contributing to the representation of diagrams in a way that improves the ability to reason and enquire about the information on which the diagram is based. The approach is described in Chapter 4.
- ❖ **ZineML:** generic XML language that captures the structure of a diagram in the GraSSML approach. A description of ZineML can be found in Chapter 4 section 4.5.
- ❖ **Verbalisation model:** consists of a template, in the GraSSML approach, gathering essential information at an appropriate level of abstraction (structure or semantics), organizing and presenting it in a way that is easy to understand textually. A more detailed description can be found in Chapter 4 section 4.6.
- ❖ **Graphic:** A visual representation that is not just text. It includes charts, graphs, diagrams, flowcharts, drawings, symbols, maps, painting, and photographs. The word “graphic” and “graphical representation” are used interchangeably.
- ❖ **Diagram:** A simplified and structured visual representations of concepts and relationships between these concepts to represent and clarify a topic.
- ❖ **Image (raster graphic):** Raster graphic composed of a rectangular array of pixels where each pixel may have both colour and opacity defined.
- ❖ **Vector graphic:** Graphic composed of a set of geometric objects defined by the coordinates of paths and areas.
- ❖ **Vector image:** A vector graphic that has been rendered as an image.
- ❖ **Image format:** A standardised way of representing an image. The data describes the characteristics of each individual pixel. PNG, JPEG and GIF formats are the most often used formats to display raster graphics images on the web.
- ❖ **Vector graphic format:** A standardised way of representing a vector graphic. The data contain a geometric description of the objects the graphic is composed of. If displayed on a raster display, the vector graphic must be rasterized. This is not necessary if rendered on a vector display. WebCGM,

SVG and SWF formats are some examples of vector formats used over the web.

- ❖ ** element:** An HTML markup element for including images in web pages. Most browsers support the GIF, PNG and JPEG image formats.
- ❖ **<object> element:** An HTML markup element for including an open-ended range of external objects in web pages. Such objects can be text, images, vector graphics, audio, video, etc. Most modern browsers support the SVG vector format natively and the SWF and CGM vector formats via proprietary plugins.
- ❖ **Web accessibility:** This thesis follows the definition of accessibility used by the Web Accessibility Initiative (WAI) (WAI, 2009): *“Web accessibility means that people with disabilities can perceive, understand, navigate, and interact with the Web, and that they can contribute to the Web. Web accessibility also benefits others, including older people with changing abilities due to aging. Web accessibility encompasses all disabilities that affect access to the Web, including visual, auditory, physical, speech, cognitive, and neurological disabilities ... a key principle of Web accessibility is designing Web sites and software that are flexible to meet different user needs, preferences, and situations.”*

1.2 Motivation

This research started with a study aiming at understanding: what makes diagrams unique? Why are they difficult to make accessible? What is missing from the existing methods and finally what needs to be addressed to make them perceivable, operable and understandable?

Diagrams present information in a way which is easy to access and process. They offer an instant access to the information present in the diagram, offering the ability to gain an overview of the information presented as well as the ability to access details of that information. This flexibility reduces the load of the working memory allowing diagrams to act as an external memory support. Such representations simplify and facilitate search of information as their spatial grouping abilities allows related information to be accessed and processed simultaneously. Diagrams also facilitate recognition by making information that would be implicit in other forms of representation such as textual, more explicit. But this is true only if these diagrams are

“effective diagrams” meaning that the diagram matches well what it represents and the task for which it is intended, making inferences instant and obvious.

Accessibility of diagrams, represented as vector graphics, for blind users and users who access this type of graphics in an environment where visual representations are inappropriate was the initial motivation of the research. For these users, alternate methods of accessing this type of web content have to be offered as they are unable to use the visual communication channel. Vector graphics accessibility means for these users the ability to understand, interact and navigate information presented diagrammatically. As the research progressed, it was realized that the approach that had been developed was far more general in the sense that it would potentially benefit “ALL”, including computers.

The World Wide Web Consortium is the leading organization promoting Web Accessibility and developing guidelines and techniques to encourage the development of accessible Web content. The first version of the WCAG, WCAG 1.0 (W3C, 1999) provided techniques to achieve accessibility, these were mainly related to HTML. The second version, WCAG 2.0 (W3C, 2008c), which became a recommendation in December 2008 aimed at being more general by taking into account different advanced technologies that have made their place in today’s web content. The WCAG 2.0 is built on four principles considered essential for any web content to be made accessible (Perceivable, Operable, Understandable and Robust).

These guidelines have some applicability to images but do not address vector graphics per se. The regulations involve providing blind people the same message a sighted person would get from an image. In the case of vector images accessibility, it is required to include alternative text that should serve as an effective replacement for the information provided by the vector image. This must be the same information that is given by the vector image to people who can see it.

Alternative accessible representations of graphics are possible but recovering the necessary information is difficult and time consuming unless that information has been built-in at the time of creation. Some attempts have been made to provide access support to vector images but even though most of these approaches provide a partial solution to the problem, the main limitation resides in the absence of information “behind” the graphic. Accessible diagrams require the information behind the diagram to be preserved in a way that makes it possible for a user to enquire and reason over it.

Some approaches aimed at taking advantage of diagrams have recognized this need of preserving access to the original information behind the diagram. They aim at generating expressive and effective diagrams from high level descriptions of a given set of data. But none of these approaches addressed the perceivability, operability and understandability of the diagrams obtained.

The availability of the information behind the diagram allows the meaning to be explicit which assists perceivability, operability, understandability, though even this is not directly tested there is something akin to an inductive reason for having these three principles. By making diagrams perceivable, operable and understandable then by definition they are made more accessible. Due to time limitations and lack of resources it was not possible to test this directly by carrying out major user studies.

1.3 Aims and Research objectives

This research aims at exploring a different approach derived from the benefits and limitations identified in existing approaches.

It is believed that defining a “smart diagram system” that stores the original information, and provides details of the transformation performed to generate the filtered view of the information, and allows them to perform alternative transformations on the information to provide different presentations is a potential way forward for this research area. Such a system should aim at representing diagrams in a way that it is possible to enquire and reason about the information they carry, making them perceivable, understandable and operable.

A system aimed at retaining as much information as possible at the creation stage of the diagram would enable readers to use this information in as many ways as possible offering the ability to enquire and reason over the diagram. The aim is to define a general purpose architecture to explore the hypothesis and develop a methodology for applying that architecture to carefully selected classes of diagrams from specific application domains.

It is not expected that this approach will be applicable to all kinds of graphical content, but rather to well-structured “formal diagrams” that conform to well-defined conventions in specific formally defined domains.

A full description of different graphic classification systems is given in Chapter 2 Given the rich diversity of diagrammatic representations captured in these taxonomies and taking into account the time frame of the research, there is a need to

focus the research on some specific types of diagrams carefully selected as a basis to demonstrate that the approach proposed in this thesis is viable and applicable

The exemplars chosen are part of Lohse's (LOHSE et al., 1994) network chart category. Such diagrams show relationships between components using lines, arrows, proximity, etc. Their planar coordinate system does not convey any meaning whereas the relative arrangement of objects does. Blenkhorn and Evans (BLENKHORN and EVANS, 1998) call this class of diagrams "schematic diagrams" consisting of objects and the relationships between these objects. Takagi and Tatsuya (TAKAGI and TATSUYA, 2007) identified a separated sub-class for such diagrams: "formal diagrams". Formal diagrams are well structured and conform to conventions in specific domains. They are defined as having a well-defined data model behind their structure. For reasons discussed in Chapter 2 section 2.4.5, two representative classes were chosen to test the GraSSML approach:

- Process diagrams (Figure 1): describe the relationships between processes using a set of well-defined elements linked together using arrows representing flows. A good exemplar of this class is UML Activity Diagrams.
- Hierarchical diagrams (Figure 2): consist of elements organised into layers between concepts. Organisational charts are a good example of this class.

The third exemplar chosen by a third party to test the applicability of the proposed approach was:

- Charts (Figure 3): are concerned with the display of data using symbols. Financial charts were chosen by a user who independently evaluated the approach. They are required to be formally described and are significantly different from the two other exemplars.

The research aims and objectives can be summarized as follows:

1. Understand what makes diagrams unique. Explore various taxonomies of graphical representations and select the classes of diagrams this research will be applicable to (Chapter 2).
2. Why are diagrams difficult to make perceivable, operable and understandable? Establish what is missing from the existing methods and finally what needs to be addressed to express and present them in a way that

would provide the ability to enquire and reason about them? Identify the issues/ requirements this research will aim at addressing (Chapter 3).

3. Explore, design and develop a general purpose conceptual architecture to address the issues/ requirements and support the hypothesis Chapter 4).
4. Develop a methodology for applying our architecture to an application domain for a specifically selected class of diagrams (Chapter 5).
5. Implement a proof of concept tool (GraSSML prototype) to demonstrate the feasibility and applicability of the proposed approach and apply the methodology to carefully selected classes of diagrams from specific application domains (Chapter 5).
6. Assess the viability and applicability of the approach through independent evaluation by an external user of the system which also applied the approach to a different class of diagrams (Chapter 6).
7. Perform an evaluation with blind and sighted users by evaluating the developed prototype against a set of predefined tasks to assess the perceivability, operability and understandability of diagrams presented using GraSSML in order to validate the hypothesis (Chapter 7).

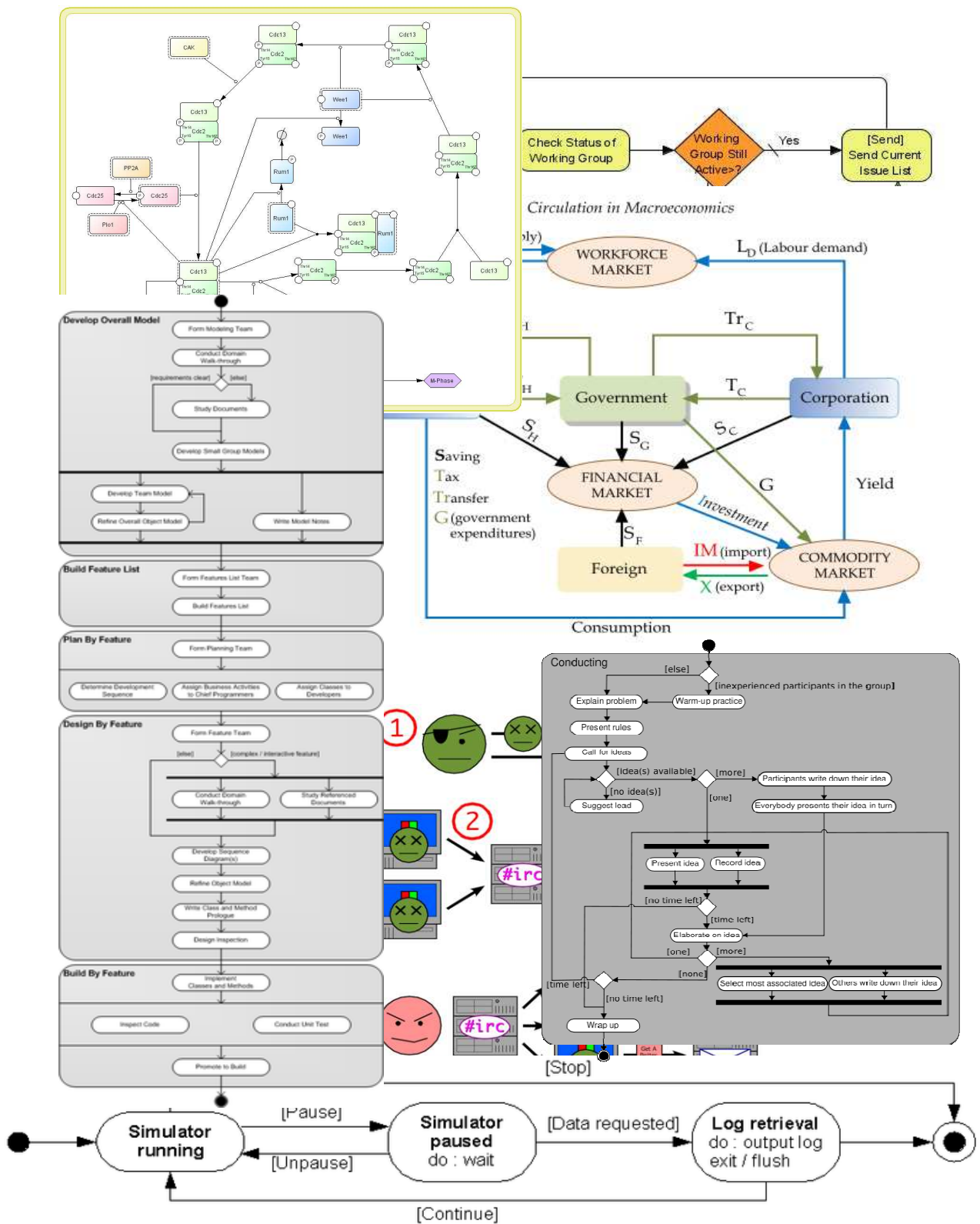
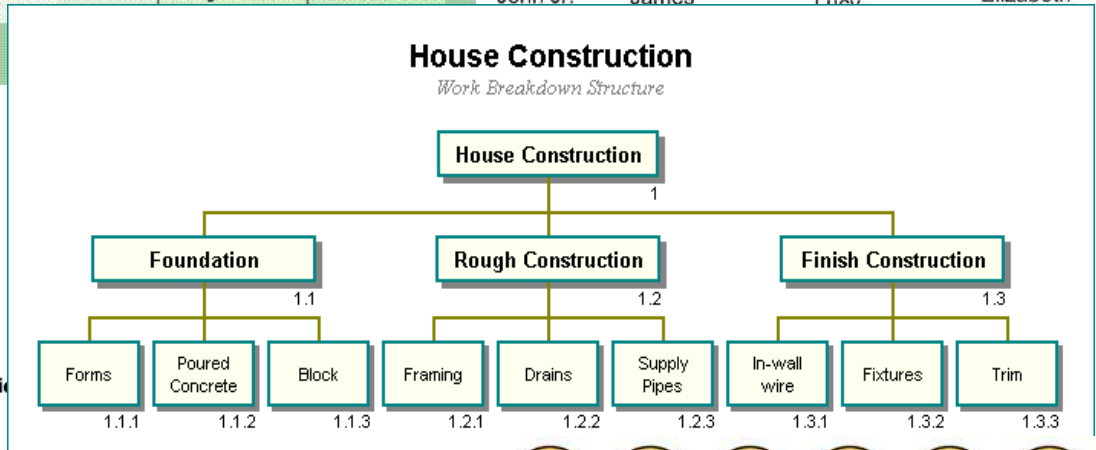
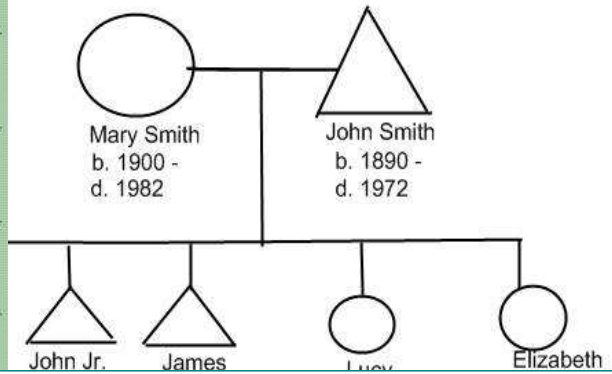
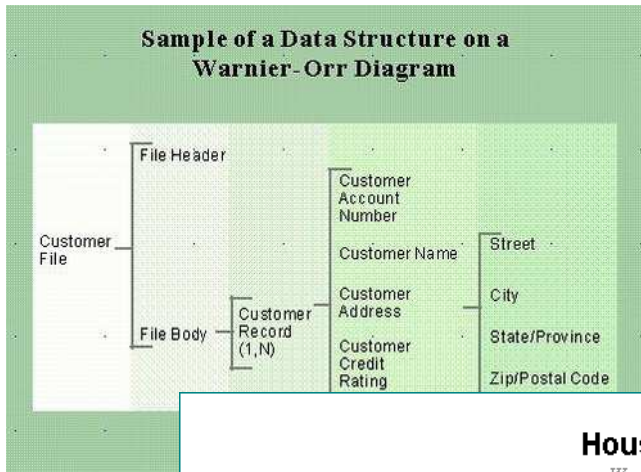


Figure 1: Examples of “Process Diagrams”



Home Page →

Main Sections →

Subsections →

Databases →

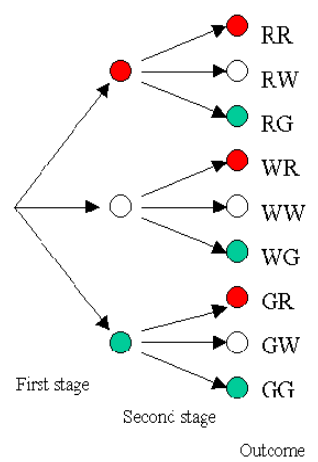
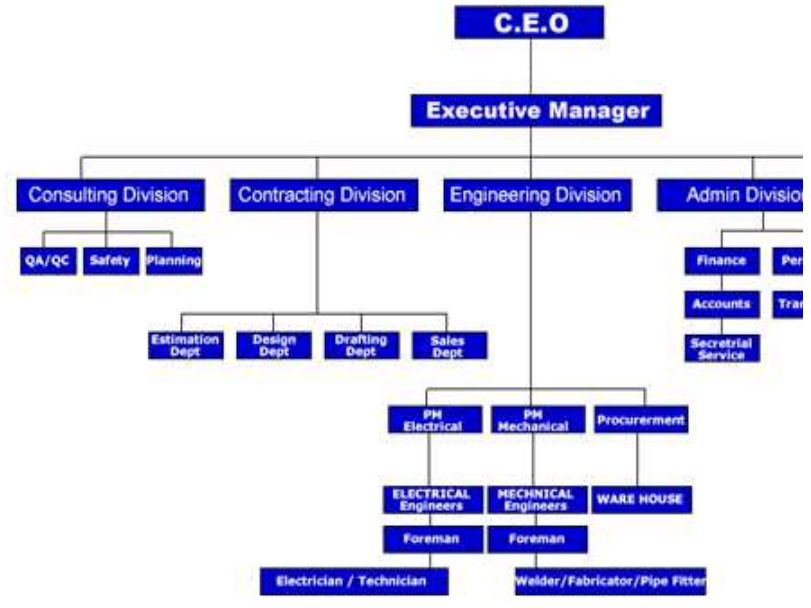
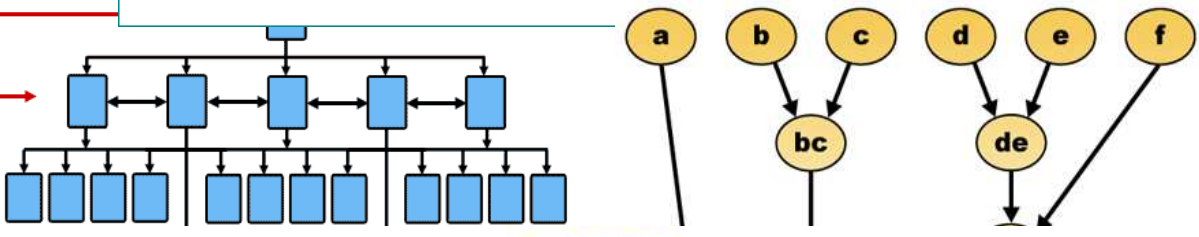


Figure 2: Examples of “Hierarchical Diagrams”

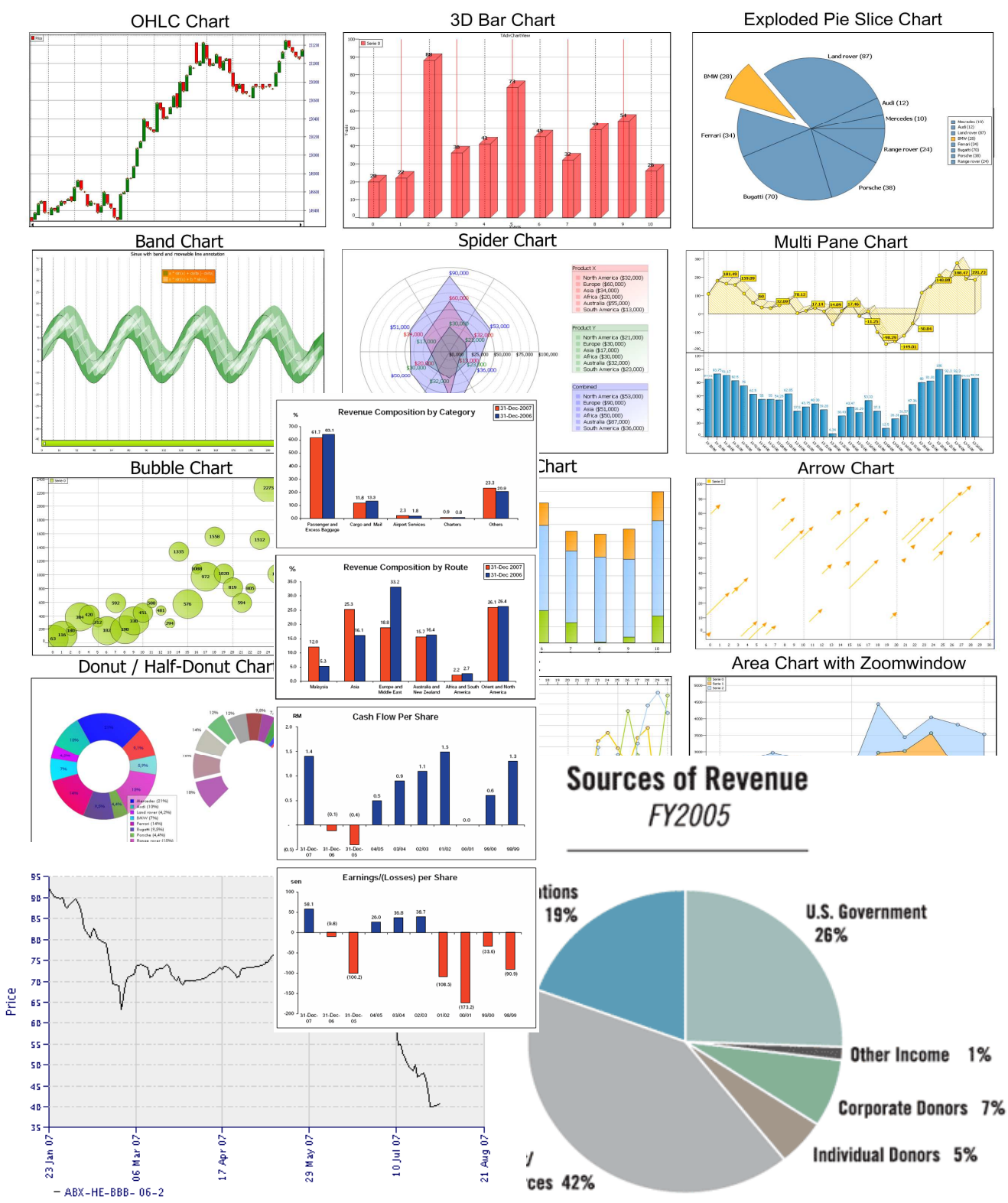


Figure 3: Examples of "Charts"

1.4 Research Contribution

The main contribution of this research is the introduction of a novel approach called Graphical Structure Semantic Markup Languages (GraSSML).

The contributions of the thesis are summarized as follow:

- The GraSSML Conceptual Architecture.
- A fully working prototype demonstrating the feasibility and applicability of the approach for three different application domains.
- A methodology to apply the GraSSML conceptual architecture to a given application domain.
- Credibility of GraSSML: evidence of the benefits of GraSSML through instances of evaluation of the working prototype with users.

1.4.1 The GraSSML Conceptual Architecture

GraSSML contributes to the representation of diagrams in a way that improves the ability to enquire and reason about the information on which the diagram is based and thus making diagrams more perceivable, operable and understandable and, as a result, suggesting enhanced accessibility benefits for such diagrams.

This framework relies on the presence of essential selected information provided by domain experts and the willingness of authors to allow the capture and access to such information while creating their diagrams upon which the approach relies. This exposes a new way of thinking about the authoring of diagrams.

1.4.2 Fully working prototype

The viability of the approach and its applicability for three different application domains has been demonstrated through a fully working prototype.

A fully functional implementation of the GraSSML conceptual architecture has been implemented in a proof-of-concept tool (GraSSML prototype). Three use cases have been considered, each for different classes of diagrams from different application domains: Process diagram “UML Activity Diagram” (Chapter 5 section 5.3), Hierarchical diagram “Organisational Charts” (Chapter 5 section 5.2) and Charts “Financial charts” (Chapter 6).

The system uses a combination of web technologies, including OWL, RDF, XSLT, GRDDL, along with some new XML based languages (ZineML), in an attempt to enable the authoring of diagrams based on their meaning (semantics) and

not their visual rendering/ presentation. This exposes a new way of thinking about the authoring of diagrams and is an interesting use of the semantic web technologies to assist this authoring process.

The GraSSML prototype has demonstrated that the GraSSML conceptual architecture is sound.

1.4.3 Methodology to apply the GraSSML approach

A methodology to apply the GraSSML conceptual architecture to a ‘formal diagram’ in a given application domain has been developed. An author’s perspective evaluation provided information on the applicability of the methodology developed for the application of GraSSML conceptual architecture.

Olivier BRAECKMAN, an MSc student, independently, applied GraSSML to financial reports in order to represent “financial charts” in a way that it is possible to reason and enquire over the information they carry thus making them more perceivable, operable and understandable and, as a result, enhancing their accessibility. The application of the methodology has been tested by Olivier during a double blind trial in which he, alone, performed the data gathering, processing and recording as well as an initial analysis. Afterwards the results and data obtained were analysed, interpreted and evaluated. The viability and applicability of the GraSSML approach was successfully assessed by Olivier who extended the system for a different class of diagrams. The evaluation of the process is presented in the evaluation chapter from an author’s perspective (Chapter 6).

1.4.4 Credibility of GraSSML

The credibility of GraSSML has been demonstrated through the ability to reason and enquire about the information behind diagrams. This ability to reason and enquire has been demonstrated through the proof-of-concept prototype. The three use cases implemented within the GraSSML prototype demonstrated the feasibility, viability and applicability of the GraSSML approach but did not provide any insight on whether GraSSML support the hypothesis. An evaluation of the GraSSML prototype, carried out with two blind users and three sighted users against a set of predefined tasks, allowed us to assess the perceivability, operability and understandability of GraSSML. This evaluation (Chapter 7) demonstrated the hypothesis and supports the ability for further studies, for greater work.

1.5 Structure of the thesis

Figure 4 illustrates the structure of the thesis.

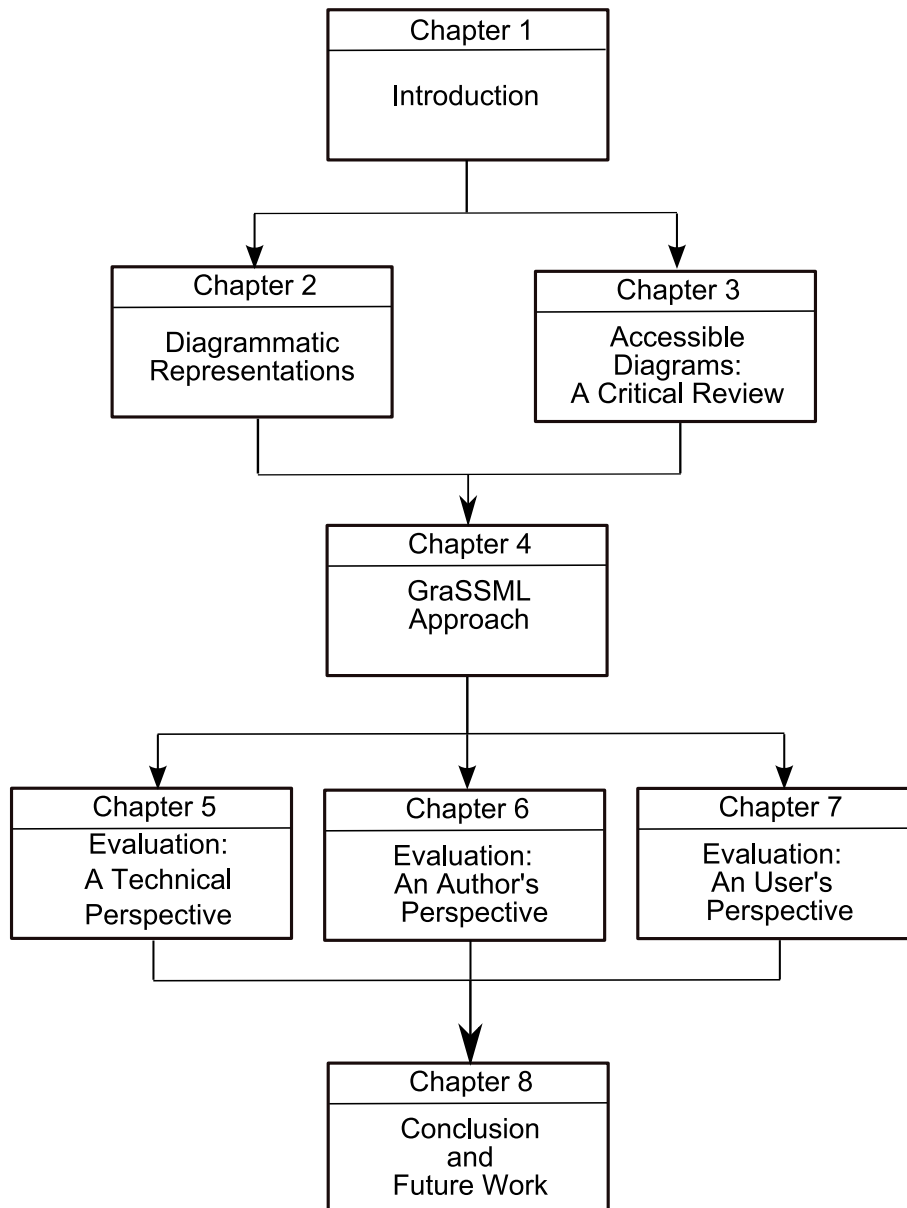


Figure 4: Thesis Structure

- **Chapter 2 (Diagrammatic Representations):** Reviews the published literature related to diagrammatic representations as well as taxonomies for graphical representations. This has led to the selection of the classes of diagrams the GraSSML approach will be applicable to.
- **Chapter 3 (Accessible Diagrams: A Critical Review):** Reviews some relevant work on the accessibility of diagrams on the web. This has led to

the identification of issues/requirements the proposed thesis will aim at addressing.

- **Chapter 4 (The GraSSML Approach):** GraSSML is described, in detail, introducing the GraSSML three-level conceptual architecture: semantic, structure and presentation and the transformations between these levels
- **Chapter 5 (Evaluation: A Technical Perspective):** Describes the system architecture of the proof-of-concept tool called the GraSSML prototype to demonstrate the feasibility and applicability of the GraSSML approach. The development and implementation of the full functional proof-of-concept prototype demonstrating GraSSML for two different types of diagrams, selected from two different application domains: Process diagrams (UML Activity Diagram) and Hierarchical diagrams (Organizational Charts), is then presented as two different use cases.
- **Chapter 6 (Evaluation: An Author's Perspective):** Describes the evaluation carried out to assess the viability and the applicability of the GraSSML approach to a different class of diagrams (in this case “charts”) in a different domain (“financial reporting”) by a third party. An MSc Student, Olivier BRAECKMAN, from Oxford Brookes University applied the GraSSML prototype to a different class of diagram “charts”, in the finance application-domain.
- **Chapter 7 (Evaluation: A User's Perspective):** Describes the evaluation, carried out with three sighted users and two blind users against a set of predefined tasks, to demonstrate the perceivability, operability and understandability of diagrams using GraSSML and to validate the hypothesis.
- **Chapter 9 (Conclusion and Future Work):** The thesis concludes by summarizing the original contributions and discusses ways the work could be extended.

1.6 Publications

A timeline (Figure 5) along side the W3C developments, has been created in order to illustrate the influence of developments in W3C on GraSSML. This project initially started in 2002 but was suspended for nearly three years for personal reasons. In the intervening period the World Wide Web started its evolution towards the

“Semantic Web” which emerged as a new vision. This led to changes such as: new technologies, new recommendations, new working groups and new tools. As these new technologies and supporting tools emerged, an “opportunity” to explore them with regard to diagram accessibility was noticed. The different publications present the GraSSML version at the time. The 2003 paper describes the initial ideas of our research project. These ideas evolved and were developed following W3C developments. The 2006 and 2007 papers describe this evolution and application. The current latest version of GraSSML has not been published yet outside this thesis. The main changes appeared at the semantic level in which the domain specific markup language has been replaced with RDF graphs.

The Publications are as follows:

- 2003: BEN FREDJ, Z., DUCE, D. A. “Schematic Diagrams, XML and Accessibility”, Proceedings of the Theory and Practice of Computer Graphics (TPCG'03).
- 2006: BEN FREDJ, Z., DUCE, D. A., “GraSSML: accessible smart schematic diagrams for all”, Proceedings of the 2006 international cross-disciplinary workshop on Web accessibility (W4A): Building the mobile web: rediscovering accessibility? Edinburgh, U.K.
- 2006: BEN FREDJ, Z., DUCE, D. A., “GraSSML: Smart Schematic Diagrams, XML and Accessibility”, Proceedings of the Theory and Practice of Computer Graphics (TPCG'06), Middlesbrough, United Kingdom
- 2007: BEN FREDJ, Z., DUCE, D. A., “GraSSML: accessible smart schematic diagrams for all”, Universal Access in the Information Society, 6(3), 233-247.
- 2008: BEN FREDJ, Z., DUCE, D. A., ZAJICEK, M.: “GraSSML: a new approach for Accessible Web Graphics for All”, tactile 2008, fourth international conference on tactile diagrams, maps and pictures for blind and partially sighted children and adults in education, work and daily life, Birmingham, UK.

GRASSML Project

W3C Recommendations

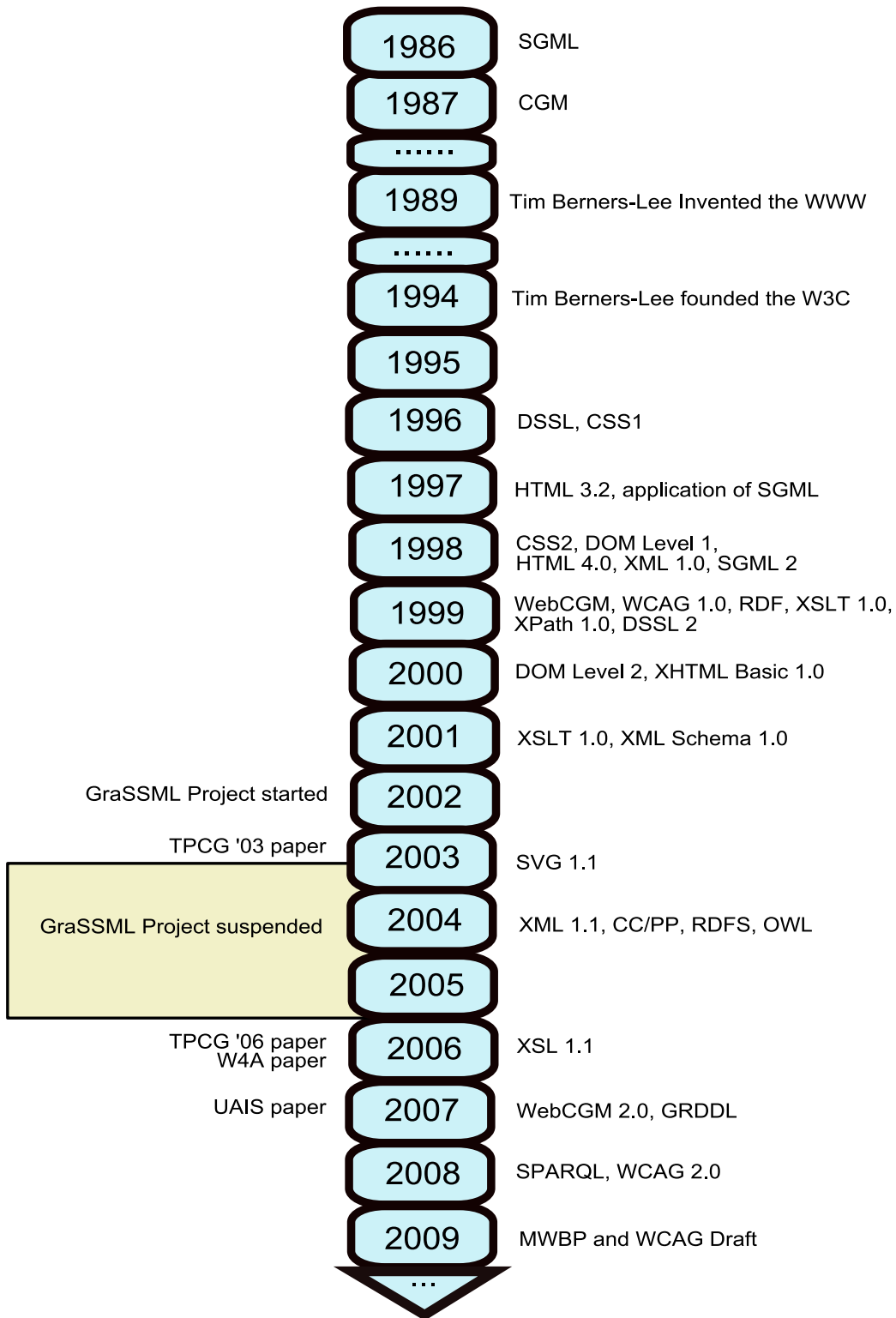


Figure 5: GraSSML Timeline along side W3C timeline of W3C Recommendations

1.7 External Support

A licence of 2 years for the product “ILOG JViews Diagrammer” has been kindly provided by ILOG (now IBM) for the realisation of the approach.

1.8 Originality

The Financial case study was done by Olivier BRAECKMAN a master student at Oxford Brookes University in 2007/2008. His Master’s dissertation entitled “GraSSML Financial Chart Project- Accessible Financial Reporting” is deposited in the Oxford Brookes University Library (BRAECKMAN, 2008). Apart from this the entire research is the author's own work.

Chapter 2

Diagrammatic Representations

This chapter reviews the published literature related to diagrammatic representations. The nature and the importance of diagrammatic representations as well as the necessity to access such representations are emphasized.

A number of taxonomies for graphical representations have been proposed, each following a specific classification scheme. A review of these taxonomies helped select the classes of diagrams the GraSSML approach will be applicable to. Some research work aimed at taking advantage of diagrammatic representations is presented. This is concerned with the generation of “effective” diagrams from high level descriptions.

2.1 Nature and importance of diagrams

2.1.1 Graphics

Graphics have had a long history, starting from as far as the prehistoric periods through cave paintings and engraved stones. They are visual representations which can be produced on different surfaces (e.g. paper, computer screens, or stone) aimed at conveying information visually to illustrate, inform or entertain. Examples of visual representations include charts (pie charts, bar charts, histograms, function graphs, scatter plots), diagrams (tree diagrams, network diagrams, flowcharts), drawings, symbols, maps, painting, photographs.

These play an important key role in representing, conveying and communicating information in many different areas of science, education and business. Graphics enhance understanding, analysis, exploration and memorization of complex information (TUFTE, 1997). Larkin and Simon (LARKIN and SIMON, 1987), and Lohse share the view that (LOHSE et al., 1994) “*Visual representations are data structures for expressing knowledge. As such, visual representations can facilitate problem-solving and discovery by providing an efficient structure for expressing the data.*” Tufte states that “*Graphics reveal data*” (TUFTE, 2001).

They are extensively used in many fields to either draw attention to important information, or serve as an extra support to understand certain concepts. Businesses profitably use graphics to advertise their products and/or services by using well

defined artwork. In business and finance, graphics have been used to create financial graphs and charts to highlight trends over time; these graphics are referred to as “business graphics”. Graphics are used by some to mislead or deceive using methods such as distortion, distraction or ambiguity to confuse the reader as to what is relevant. Graphics have also proved efficient in propagating political ideas (e.g. posters, graffiti, and flags). Science and education extensively use graphics to illustrate concepts (e.g. diagrams) which otherwise would be difficult to describe and explain (e.g. anatomy, maps).

Graphics are an integral part of the web and played an important role in the success, acceptance and growth of the World Wide Web and continue to occupy an important position in the use of the web (DUCE et al., 2002). In 1994, the Mosaic browser contributed to the transformation of the web. The html element `` introduced in the mosaic browser allowed images to be viewed within a web page and played a key role in the growth of the web. Indeed, the advantages of being able to communicate via graphics on the web did not go unnoticed by organisations which now could reach and provide information to millions of potential customers by communicating their logos, their products, their locations on maps, their art work (photographs, paintings, etc.) (GILLIES and CAILLIAU, 2000). Wilson (WILSON, 2008) states that “*the graphical nature of the Web is definitely one of its biggest selling points*”. As mentioned in the introduction, this research will be restricted to the study of diagrams.

2.1.2 Diagrams

A. Definition

It is not trivial to define what a diagram is and answers vary depending on specific perspectives (FATHULLA and BASDEN, 2007).

One definition proposed by (VISWANATH et al., 2006) is “*A diagram is a simplified, structured representation that describes the components of a system and their spatial relationships*”.

In the context of their research Larkin and Simon (LARKIN and SIMON, 1987) define diagrams as external representations recorded on some medium (e.g. paper, blackboard). A representation consists of both data structures and programs which operate on them through the processes of search, recognition and inference, to make new inferences. They define diagrammatic representation as a “*data structure in*

which information is indexed by two-dimensional location". In this thesis the following definition of a diagram is adopted:

"A diagram is a simplified and structured visual representation of concepts and relationships between them used to represent and clarify a topic. Diagrams are composed of shapes and text. The information concerning the concepts represented and their relationships is the important information whereas its layout is not".

B. Main Characteristics of diagrams

Many scientists have focused their efforts on discovering the powerful advantages diagrams offer in accessing and processing information.

❖ Powerful advantages of diagrams

Diagrams provide an intuitive mechanism to access, interact with and process complex information. They are considered as being a powerful tool. As Tufte (TUFTE, 2001) states "*...of all methods for analysing and communicating statistical information, well-designed data graphics are usually the simplest and at the same time the most powerful*".

Many researchers worked on identifying why such representations are powerful and what makes them different to other forms of representations in conveying information and solving problems (LARKIN and SIMON, 1987, TUFTE, 2001, KULPA, 1994, SUWA and TVERSKY, 2002).

Diagrams can be used to display a large amount of information (TUFTE, 2001) offering the user an overview of the information displayed. Indeed, diagrams are concise, they offer the ability to gain a bigger picture of the information presented and simultaneously offer the ability to explore by digging deeper into its details without losing this big picture. This ability to offer flexible constant access reduces the effort needed by the working memory as the diagram act as an external memory support (LARKIN and SIMON, 1987).

Diagrams allow effective communication which facilitates the representation of abstraction. They have the ability to present the information in a way which is easy to access and process (TUFTE, 2001): (LARKIN and SIMON, 1987) "*A diagrammatic representation permits information at or near one locality to be accessed and processed simultaneously*".

This simplifies and facilitates the search for specific information (distance, relationships between objects, size, colour, etc.) and reduces memory needed in the process (LARKIN and SIMON, 1987, SUWA and TVERSKY, 2002).

Larkin and Simon (LARKIN and SIMON, 1987) compared two informationally equivalent representations: sentential and diagrammatic representations. They state that the “diagrammatic” representations convey relationships among information more directly than “sentential” representations. Text differs from diagrams in the way it conveys information. They have found that diagrams facilitate the search of information as the spatial grouping ability of these representations allows related information to be perceived at once.

Diagrams have proved to be effective tools at promoting communication and visual thinking. Diagrams prove to be of great help in reasoning, problem solving and decision making (KULPA, 1994).

Diagrams are organised by location and often much of the information needed to make inferences is explicitly preserved and located in proximity, making search and recognition easier (LARKIN and SIMON, 1987). This form of presentation makes information more explicit than other forms of presentation (e.g. textual presentation) allowing direct retrieval of that information (KULPA, 1994).

Larkin and Simon believe that “*diagrams and the human visual system provide, at essentially zero cost, all of the inferences we have called perceptual*”(LARKIN and SIMON, 1987). They state that “*the great utility of the diagram arises from perceptual enhancement*” so “*that people can focus attention on part of a diagram, and that they can detect cues there and use them to retrieve problem-relevant inference operators from memory*”.

❖ **Not all diagrams are powerful**

As well as providing explanations on why diagrams can be superior to other representations, Larkin and Simon (LARKIN and SIMON, 1987) also highlight the need for appropriate knowledge that is required to effectively take advantage of such representations.

Not all graphics are considered as “good” graphics. In that respect not all diagrams are worth ten thousand words. It depends on the ability of the user to construct an effective diagram. The diagram should be constructed in a way which takes advantage of the features that makes them powerful, which is not always trivial. If badly constructed, a diagrammatic representation can conceal the information

message intended to be conveyed. Gurr (GURR, 1999) investigated the issues of diagrammatic communication in order to find out what makes an effective diagrammatic communication. He argues that an “effective diagram” is one that matches well what it represents and the task for which it is intended, making certain inferences instant and evident.

Of course, exceptions arise when an author creates a diagram with the intention to mislead or deceive and in which the diagram does not match well with what it represents. These kinds of diagrams could still be defined as “effective” as they are designed to confuse the reader as to what is relevant, which is exactly the intent of the author (e.g. some financial charts).

This difficulty of creating an effective graphical representation is discussed by Tufte in his books (TUFTE, 1997, TUFTE, 1990, TUFTE, 2001). Tufte (TUFTE, 2001) refers to “Graphical excellence” which consists of complex ideas communicated with clarity, precision and efficiency. This should give “*the viewer the greatest number of ideas in the shortest time with the least ink in the smallest space*”.

C. Summary

Diagrams, which are simple structured visual representations of concepts and relationships between these concepts, present information in a way which is easy to access and process. They offer an instant access to all the information present in the diagram, offering the ability to gain an overview of the information presented as well as the ability to access details of that information. This flexibility reduces the load of the working memory allowing diagrams to act as an external memory support. Such representations simplify and facilitate search of information as their spatial grouping abilities allows related information to be accessed and processed simultaneously. Diagrams also facilitate recognition by making information that would be implicit in other forms of representation such as textual, more explicit. But this is true only if these diagrams are “effective diagrams” meaning that the diagram matches well what it represents and the task for which it is intended, making inferences instant and obvious.

2.2 Graphicacy

Using effective graphical representations requires the acquisition of necessary skills called graphicacy skills.

Graphicacy (BALCHIN and COLEMAN, 1966), a word created by Balchin and Coleman in 1965, (ALDRICH and SHEPPARD, 2000) describes the capacity to comprehend and present information in the form of graphics. Graphicacy is identified as a complex form of communication (WILMOT, 1999): *“It requires that the reader/creator of graphic language possesses conceptual knowledge of the phenomena represented in the graphic representation, as well as spatial perceptual abilities and an understanding of spatial concepts; and it requires practical skills of being able to create graphic forms to communicate information to others.”*

The comprehension of information presented graphically requires prior knowledge. A person’s capacity to interpret certain types of graphics relies on his background knowledge. Two main types of background knowledge have been identified as being essential in comprehending graphics:

- Knowledge about the specific graphic system used to represent the information.
- Knowledge about the subject matter that is represented graphically. Personal knowledge of a domain influences the interpretation of the graphical information.

Research suggests that (NARAYANAN, 1997) *“diagram comprehension is a constructive process in which the individual attempts to use his or her prior knowledge of the domain, information presented in the diagram, and his or her reasoning skills to build a mental model”*.

2.3 Syntax, Semantics and Pragmatics of Diagrams

Effective diagrams are considered as expressions of visual languages, with their own syntax, semantic and pragmatic. The syntax of a diagram is characterized by the set of graphical elements, their properties and their relationships.

The semantics of a diagram concerns the relationships of graphical elements to the concepts these graphical elements are applicable to. It concerns the meaning of the diagram: the concepts, their properties and their relationships.

Pragmatics concerns the relationships between the graphical elements and the receiver interpreting them. It concerns the best appropriate usage of syntax to ensure an appropriate interpretation of the diagram's information by a human receiver. This information includes the implicit knowledge. The pragmatics helps to emphasize important aspects of the diagram. An author of a diagram starts with the information he wants to communicate and needs to find out the best way to communicate this information effectively. Examples of pragmatic issues are the choices in properties of graphical elements such as size, colour or position. Taking into account practical experiences and observations, decisions and actions are made to ensure an appropriate interpretation of the diagram. Such issues have no impact on the semantics of the diagram but will have an impact on its interpretation by a human receiver. Gurr (GURR, 1999) states that pragmatics helps to bridge the gap between "what is said and what is meant". The correct use of pragmatic features contributes to the understanding of the diagrammatic representation.

In order to comprehend information presented graphically it is important to know how to interpret graphics. Such interpretation requires graphicacy skills which, as well as oral, literal and numerical skills are essential to communicate effectively.

2.4 Taxonomies of visual representations

The investigation of different varieties of diagrammatic notations is of interest to researchers in psychology and cognition. Various classes of graphics can be identified depending on the chosen classification scheme.

Different diagrams types are looked at in different ways. The same type of diagram can be read in a number of different ways.

A wide range of classifications has been described. Taxonomies of visual representations have been developed in several different academic fields. Each of these taxonomies focuses on certain aspects of graphical representations.

2.4.1 Blackwell Taxonomy of taxonomies

Based on the analysis of some of these taxonomies, Alan F. Blackwell et al. (BLACKWELL and ENGELHARDT, 2002), proposed a classification divided into two sets of aspects: representation related and context related.

❖ Representation related aspects relate to:

The components of the diagram:

- (1) Basic graphic vocabulary
- (2) Conventional elements
- (3) Pictorial abstraction

The graphic structure of the diagram:

- (4) Graphic structure

The meaning of the diagram:

- (5) Mode of correspondence
- (6) The represented information

❖ **Context related aspects relate to:**

- (7) Task and interaction
- (8) Cognitive process
- (9) Social context

A common distinction between existing taxonomies involves the differences between “function” and “structure” of visual representations. Depending on the context of their use, either functional or structural taxonomies are considered for visual representation. In a “functional taxonomy”, the visual representations are classified based on their “intended use”. According to Blackwell et al., this refers to “(7) task interaction” and “(9) social context” aspects. In a “structural taxonomy”, the visual representations are classified based on their perceived structure. According to Blackwell et al., this refers to “(1) basic graphic vocabulary” and “(4) graphic structure”.

2.4.2 Lohse’s Taxonomy

An experimental classification of visual representations has been carried out by Lohse et al. Participants were asked to classify visual representations based on their structural similarities.

A first study (LOHSE et al., 1990) based on this experimental classification, identified six categories of visual representations: graphs, tables, networks, diagrams, maps and icons.

In the following studies (LOHSE et al., 1994), the initial basic categories have been confirmed and some more categories have emerged from the new experiments, a total count of 11 categories (Table 1).

Graphs	encode quantitative information. Some examples of common graph types are histogram, line chart, bar chart, pie chart
Tables	2D arrangement of words, numbers and symbols or a combination of them
Network charts	make use of arrows, proximity, etc. to show the relationship between components
Structure diagrams	describe the physical objects they represent
Process diagrams	describe the interrelationships and processes associated with the physical objects
Maps	symbolic representations of physical geography
Graphic tables	same as table but considered more attractive
Cartograms	spatial maps that show quantitative data
Icons	as a label, convey a single meaning
Time charts	similar to table but encode temporal data
Pictures	realistic depictions of an object or scene

Table 1: Lohse's taxonomy of visual representations

This classification is mainly based on the perceptual classification of these visual representations by the participants rather than the analysis and interpretation of unique features of these visual representations. Some of the categories obtained by Lohse's experiments fit the group categories of 2D visual representations presented by Bertin (BERTIN, 1983): Diagrams, Networks, Maps and Symbols.

2.4.3 Blenkhorn and Evans's taxonomy

For the purpose of their research on graphics accessibility more specifically on "Talking Tactile diagram", Blenkhorn and Evans (BLENKHORN and EVANS, 1998) considered that graphical information can be classified into the following five types (Table 2).

Real world images	This type refers to photographic images, pictures. Highly visual, they have proved to be a challenge to represent in any other modality.
Maps	2D views of objects where the absolute position, size and shape of each object are considered important. Relative position and size must be maintained for a map to be meaningful. They are defined as an abstract view of the real world emphasising the essential details. Examples include geographic maps; diagrams of mechanical components, medical drawings, etc. Most of these representations are successfully presented as tactile diagrams sometimes with speech added.
Schematic diagrams	These are structured diagrams which consist of objects and the relationships between these objects. As maps, they are also defined as presenting abstract information, however for this type of representation position and size of the objects of the representation are not as important as the explicit relationships between these components. Examples of such representations include software engineering diagrams, project planning charts, organisation charts, underground map, etc.
Charts	This type of representation presents structured tabular data. Pie charts and histograms are examples.
Graphical interfaces	This type of representations refers to windows, icon, menu, and pointer type of interface rather than the information which can be represented using such an interface.

Table 2: Blenkhom and Evans's (1998) taxonomy of visual representations

2.4.4 Takagi and Tatsuya's taxonomy

In an attempt to identify next generation accessibility features in graphics standards, (TAKAGI and TATSUYA, 2007) carried out a survey of “typical” existing business graphics. A set of 60 examples were classified according to their complexity and formality, their accessibility potential was also discussed by the authors. Four categories emerged from their survey (Table 3).

Simple graphics	This category is presented as simple to describe using an alternative modality such as Braille or speech output.
Formal diagrams	These diagrams are defined as having a well-defined data-model behind their structure. Examples of these diagrams include tree structures, Unified Modelling Language (UML) diagrams.
Table structures	These diagrams can be regarded as tables and consequently can be accessed as such by assistive technologies.
Composite diagrams and graphics	These accounted for 50% of the sample graphics categorized. They are composed of multiple types of diagrams and are mostly informal. These diagrams are a challenge in terms of accessibility.

Table 3: Takagi and Tatsuya's taxonomy (TAKAGI and TATSUYA, 2007) of visual representations

Figure 6 extracted from (TAKAGI and TATSUYA, 2007) shows an overview of the classification of the examples selected for their study, the categories identified are shown. The complexity of the graphics is shown on the horizontal axis and the formality on the vertical axis.

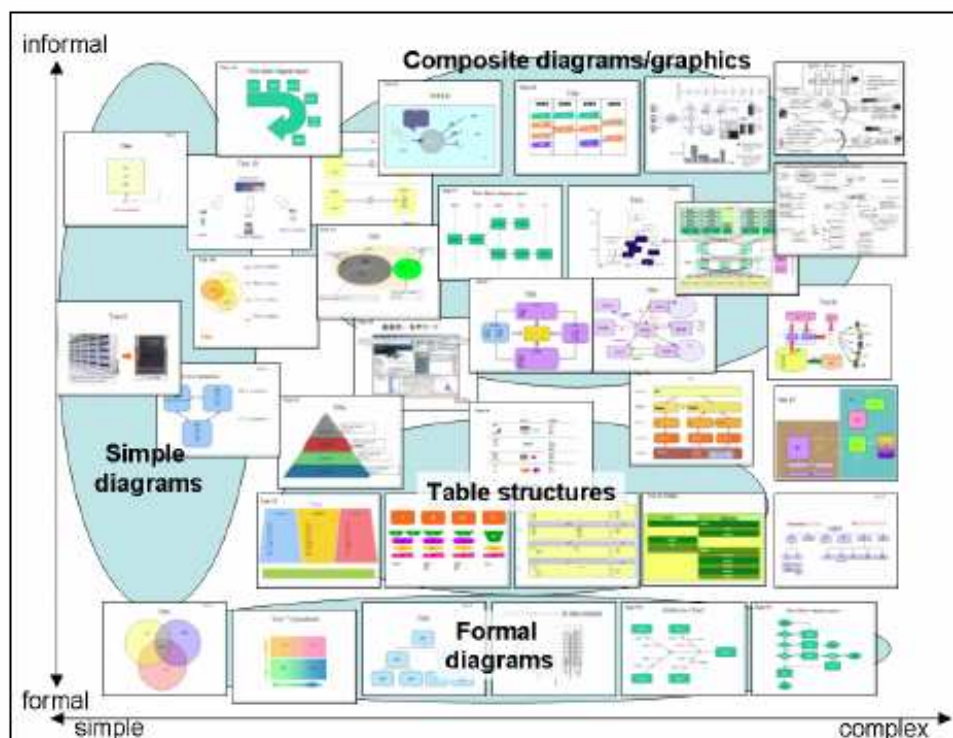


Figure 6: Categorizations of collected graphics (TAKAGI and TATSUYA, 2007)

2.4.5 A Synthesised Taxonomy Selection of Diagrams

The previously discussed taxonomies have been considered. Given the rich diversity of diagrammatic representations captured in these taxonomies and taking into account the time frame of the research, there is a need to focus on some specific types of diagrams carefully selected as a basis to demonstrate that the approach proposed in this thesis is feasible.

As a starting point, the structural taxonomy proposed by Lohse (LOHSE et al., 1994) has been used to identify some representative types of diagrams for which the proposed approach might be applied. Any of the previously presented taxonomies could have equally been chosen as a starting point as in this instance only the name of the category differs.

The types of diagrams selected are part of the “Network chart” category defined by Lohse. These diagrams show the relationships between components using lines, arrows, proximity, etc. Their planar coordinate system does not convey any meaning whereas their relative arrangement does. Network charts include flow charts, organizational charts, decision trees, pert charts and data models. Blenkhorn and Evans (BLENKHORN and EVANS, 1998) have named this class of diagrams “Schematic diagrams”. They define these diagrams as structured diagrams which consist of objects and the relationships between these objects. As maps, they are also defined as presenting abstract information, however for this type of representation the position and size of the objects of the representation are not as important as the explicit relationships between these components. Examples of such representations include software engineering diagrams, project planning charts, organisation charts, etc. The selection was further narrowed down to “formal diagrams”. Referring to the (TAKAGI and TATSUYA, 2007) taxonomy in terms of formality and complexity, “formal diagrams” are diagrams defined as having a well-defined data model behind their structure.

The approach explored in this thesis will apply to well structured diagrams that conform to well-defined conventions in specific formally defined domains. The selected classes are representative of a wider range of diagrams. It is believed that because of underlying similarities, demonstrating that the approach works on these selected classes would allow inferring the feasibility on a wider range of diagrams of the same type.

This led to the reasoned choice of selecting two classes of formal diagrams:

- Process diagrams: describes the relationships between processes and/or objects. They consist of elements linked together using arrows representing flows between the elements. Examples (Figure 1) of this class of diagram are business process diagrams, UML Activity diagrams, process flow diagrams, flowchart diagrams, UML state diagrams.
- Hierarchical diagrams: describes hierarchical relationships in levels/layers between concepts. They consist of a hierarchical structure of elements organised into different levels. These diagrams are often used to depict the structure of a model (e.g. enterprise, family, system, etc.). Examples of this class of diagram (Figure 2) are Organisation charts, Family Trees, Warnier-Orr Diagrams, Directory Trees, etc.

In the context of this thesis, two particular types of diagrams have been selected as a basis for experiments, one from each of the two classes: Organisation charts from the hierarchical diagram class and UML Activity diagrams from the process diagram class.

2.5 Diagrammatic Communication

As argued in the previous section diagrams can be useful and have an important place in communication as they play a key role in some disciplines such as science, geography or mathematics. But as mentioned by Larkin and Simon (LARKIN and SIMON, 1987), studies on understanding why diagrams are such a useful powerful representations lead to questions such as “what is the knowledge required for effective diagram use?”. Indeed, in their view *“diagrams can be better representations ... because the indexing of its information can support extremely useful and efficient computational processes. But this means that diagrams are useful only to those who know the appropriate computational processes to take advantage of them”*.

Among their advantages, the way diagrams encode, convey information and aid reasoning, has attracted much interest in many fields of research (NARAYANAN, 1997). These research interests have been classified in three categories:

- Nature of diagrammatic representations
- Cognitive processes of diagrammatic representations: perception, comprehension, reasoning, generation and manipulation

- Computational processes of diagrammatic representations: parsing, interpretation, compilation, execution, generation and manipulation.

Diagrammatic communication involves the use of diagrammatic languages for the creation of diagrams to communicate information. So diagrams are considered as expressions of visual languages.

Horn states “*Visual Language as one of the more promising avenues to the improvement of human performance in the short term (the next 10 to 15 years)*” (HORN, 2001). He introduces the major goals for the next 15 years:

- automatically create diagrams from text
- create tools for collaborative mental models based on diagramming
- improving the semantic web, by managing multiple meanings of visual languages in real time on the web
- get computers to understand the link between visual and verbal languages, etc.

Each diagram carries a message. Diagrams are created in order to communicate a specific intent and aim to make this intent understandable by a reader.

This intent needs to be accessed to discover the knowledge the diagram carries. As Lohse (LOHSE et al., 1990) writes “*visual representations carry no meaning without the decoding processes that interpret the visual representation. People must have rules to interpret features of visual representation.*” Once such rules become familiar, diagrams develop into an efficient tool for communication.

Such diagrammatic communication requires generation, comprehension, reasoning and interaction involving three entities (NARAYANAN, 1997): the diagrammatic representation representing the intended message, the communicator representing the idea diagrammatically and the receiver comprehending the meaning of the diagram. Figure 7 extracted from (NARAYANAN, 1997) illustrates such a model of communication. The communicator and receiver roles may be cognitive or computational agents. Computational processes involve diagram parsing, diagram interpretation, program execution, diagram generation and manipulation to convey the result of program execution. The cognitive processes are diagram perception, comprehension, inference and diagram generation and manipulation to convey the results of inferences. Based on the model (NARAYANAN, 1997) stipulates that the

utility of a diagrammatic communication rests on “its computational tractability and cognitive effectiveness”.

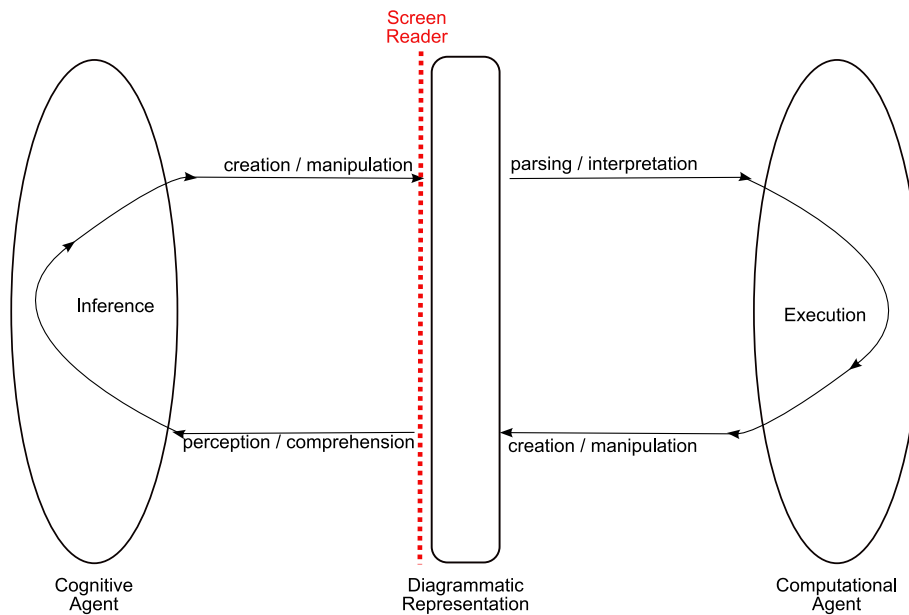


Figure 7: Diagrammatic communication based on (NARAYANAN, 1997)

2.6 Generating diagrams

The previous sections on diagrams led to the question of the production of diagrams on computers. Researchers working on visualization have been investigating the most appropriate way of displaying a given set of data on a computer. They have been exploring ways for generating diagrams from high-level representations. These approaches are mainly aimed at automatically generating effective diagrammatic representations based on the type of the data to be presented and/or the tasks it is intended for (DUKE et al., 2005) .

The following sections review different methods that have been explored over time to generate diagrams from raw data. Each of these methods explores a given intent. They start from high-level representations of a given set of data. They are aimed at generating expressive and effective diagrams based on the type of data and/or the tasks it is intended for.

2.6.1 A Presentation Tool (APT)

Mackinlay (MACKINLAY, 1986) was one of the first to contribute to the automatic design of visualizations. He explored the automatic design of expressive and effective graphical presentations for 2D static presentation of relational data such

as bar charts, scatter plots, etc. His approach is based on the assumption that “*graphical presentations are sentences of graphical languages, which are similar to other formal languages in that they have precise syntactic and semantic definitions*”.

Mackinlay highlights the fact that all communication is based on convention sharing. Indeed, to determine how messages are constructed and interpreted a group of persons have to agree and share a given set of conventions. In the case of communications of graphical representations, these conventions define how specific arrangements of graphical objects encode information. The formalization of these conventions is used as a basis for the logic program designing presentations automatically.

They have built “A Presentation Tool” (APT) that designs graphical presentations of given information automatically. APT extracts the information from a database, and then synthesizes a graphic design which is an abstract description indicating the graphical techniques used to encode the information it presents. Based on this graphical design the tool renders a graphical presentation.

The foundation of this research relies on the development of visual languages describing the syntax and semantics of graphical presentations.

Two set of criteria are used to codify graphic designs: expressiveness and effectiveness. The expressiveness criteria determine whether a graphical language can express exactly the given information. The effectiveness criteria determine whether a graphical language exploits the capabilities of the output medium and the human visual system. However, in this particular research, the author focuses on the generation of designs which can be accurately interpreted. APT generates optimal graphical designs by matching data types with respect to Cleveland and McGill’s recommendations (CLEVELAND and MCGILL, 1984).

APT presents a number of limitations. It is mainly intended for specific graphics which are easily interpreted. Its applicability and capabilities are restricted. APT does not take task into account, so the graphics generated may not be the best for a specific task hence based on Gurr’s definition of an effective diagram (GURR, 1999), they might not be effective diagrams.

2.6.2 BOZ

BOZ (CASNER, 1991) extends APT by rooting predicate in task. BOZ designs different graphical representations of the same information based on the nature of the task intended to be supported. It relies on the cognition theory of Larkin and Simon (LARKIN and SIMON, 1987) and shares Gurr's view on the effectiveness of a diagram depending on how well it matches both what it represents and the "task" for which it is intended.

One advantage of such a method is the achievement of an effective graphic well designed for the intended task. But when creating a diagram not only the information to be presented is needed but the intention of the receiver must be known and passed to the system. As mentioned by Casner, BOZ presents some limitations as this process involves human intervention. Task descriptions have to be prepared by hand and then submitted to BOZ. Also BOZ does not provide good enough time predictions.

2.6.3 TRIP2

TRIP2 (TAKAHASHI et al., 1991) is a prototype system based on a model of bi-directional translation between internal abstract data of applications and pictures to generate diagrams (e.g. graph diagrams). It is based on declarative mapping rules expressed in Prolog. Using these rules it becomes possible for changes made on a diagram to be fed back to the abstract structure data automatically by the system.

As the authors stress, the idea of bi-directional translation is basically simple. The difficulty arises in its applicability as a general model, due to the infinite amount of possible kinds of data and visual representations. This imposes the definition of specific dependant rules for every new combination of abstract data and picture.

To address this issue, they define four different representations (Figure 8):

- Application's Data Representation (AR): the application specific data.
- Abstract Structure Representation (ASR): the underlying abstract structure represented by relations of abstract objects from the data.
- Visual Structure Representation (VSR): the underlying structure of the picture.
- Pictorial Representation (PR): the representation of the picture to be rendered on the device.

Two universal representations, ASR and VSR, are used as pivots. Using these pivots, whatever changes occur on the abstract or pictorial data, will essentially not affect the mapping rules between ASR and VSR.

As pointed out by the authors the approach presents some drawbacks. The user has to write the mapping rules and their inverse, which is not straightforward using Prolog. A possible solution proposed is to derive one from the other or allowing the specification of the rules visually.

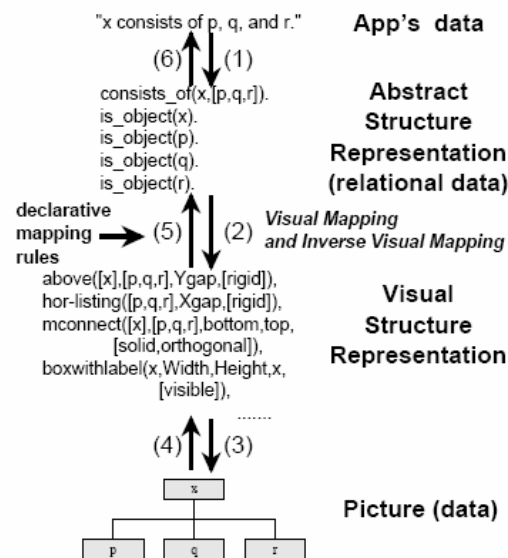


Figure 8: TRIP2 models (TAKAHASHI et al., 1991)

2.6.4 AVE

AVE (Automatic Visualization Environment) (GOLOVCHINSKY et al., 1995) generates diagrams automatically based on the theory that “*diagrams are constructed based on the data to be visualized rather than by selections from a predefined set of diagrams*”.

AVE uses basic diagram components corresponding to relations present in the data. The aim is to preserve relational properties of the data within the graphical presentation in order to reflect the structure of the data. AVE does not aim to be general. It is particularly useful for the generation of diagrams from heterogeneous data structures (GOLOVCHINSKY et al., 1995). A prototype that automatically constructs semantic networks diagrams has been developed. It demonstrates how, based on properties of binary relations, data stored in semantic networks have been

organised and the classification of the binary relations is exploited to visualize the data structure.

2.6.5 OntoDiagram

With the intent to provide a better care in paediatric cardiology, a research project called “OntoDiagram” (VISHWANATH et al., 2005) (Ontology based Diagram Generation) has explored the automatic generation of diagrams based on patient’s information stored in a cardiology database and by using clinical and spatial ontologies representing the human heart.

OntoDiagram used spatial and domain terminology common to domain experts to model the structural aspects of a diagrammatic description. This description is then used to generate the diagram. The heart is modelled as a hierarchical representation of concepts. The possible defects are modelled into a classification including for each its unique spatial aspects relative to the normal heart. The image descriptions of the components generated by the models are mapped to actual images of the components. To make the composition process more efficient and effective, the heart base is used to configure each component position. Annotations are used to add extra information to each heart component and heart defect, these are then classified depending on their physiology. This offers users the option to modify the amount of information on a representation depending on its preferences or the heart defect requirements. They have developed a query interface which allows heart defect descriptions to be queried and to generate diagrams.

An expansion (VISWANATH et al., 2006) of the project aimed at capturing the variability of heart defects was developed. By modularising the knowledge underlying the heart diagram, it becomes possible to apply transformation models to represent the effects.

2.6.6 SemViz

Gilson et al. (GILSON et al., 2008) propose a novel approach to produce an end-to-end automatic visualization from extracted semantic information of domain specific data retrieved from the web. Their approach combines ontology mapping and probabilistic reasoning techniques.

This approach pursues the same goals as Mackinlay (MACKINLAY, 1986) and falls into the same scope of (DUKE et al., 2005). The originality of the approach relies on the use of ontologies and ontology mapping.

A domain ontology is mapped to a web page. This domain ontology stores the semantics of the domain specific data represented in the web page. The domain ontology is then mapped to different visual representations ontologies which capture popular visual representations. Each of the visual representation ontologies captures the semantics of a visualization style (e.g. Graph, Networks).

A semantic bridging ontology is then used to map the domain ontology and the visual representation ontology. This ontology specifies the appropriateness of each semantic bridge by using some specific key knowledge “semantic equivalence” about the relationship between the data and the possible target visualizations. This information is based on expert knowledge about how domains can be “usefully” visualized by different visual representations. Based on the “analysis” of these three ontologies, different possible visualization designs are dynamically generated. These visualizations are scored, and the top best ones are then presented for the user to make a decision based on his needs. A prototype based on this approach, called “SemViz” and dealing with “music charts”, has been developed to demonstrate the potential of the approach.

2.6.7 PIC

Bentley argues that a language is “*any mechanism to express intent, and the input to many programs can be viewed as statements in a language*”. By viewing inputs in this way, it is possible to see how to decompose a computation into a sequence of processing steps in which the output of one step is fed into the input of the next. An example is Kernighan’s PIC language (KERNIGHAN, 1982, KERNIGHAN, 1991), a component of the Unix document production suite for constructing line drawings from textual descriptions. This language works by generating code that is interpreted by the TROFF typesetting system to produce the drawing on the page. Bentley shows how other Little Languages designed to express particular kinds of graphical representations, for example chemical structures, CHEM (BENTLEY et al., 1986) and graphs, GRAP (BENTLEY and KERNIGHAN, 1986), can be translated into PIC and hence rendered in a document.

Kernighan's PIC language is one possible way of generating diagrams. Although, PIC predates all of the latest new web technologies and does not make use of ontologies, it exposes a very powerful idea: "multiple transformations". Some people such as Jeni Tennison (TENNISON, 2005) have picked up on the powerful advantages of such multiple transformations. She highlights the advantages of structuring a workflow into XML pipelines to improve the manageability of XML publishing by specifying smaller transformations.

The research project presented in this thesis noticed and intends to explore the availability of the current powerful technologies in order to express the kind of intuitions behind PIC.

2.7 Discussion and Conclusion

This chapter has provided a detailed description of the powerful advantages (Section 2.1.2B) diagrams offer in presenting, accessing and processing information.

Diagrams are created in order to communicate a specific intent which needs to be accessed to discover the knowledge the diagram carries. The whole idea resides in the communication between the cognitive agent and the computational agent (Figure 7). It is all about a certain symmetry that allows going back up through some mediating intermediate abstract representation, to the original information, the "raw data". The techniques presented (section 2.6) recognize the need of "going back up". Some of these techniques have highlighted the importance the formalization of the conventions shared plays when communicating using diagrams. These methods rely on the development of visual languages expressing the syntax and semantics of diagrams.

A review of taxonomies for graphical representations helped select the classes of diagrams this research will be applicable to. The approach explored in this thesis will apply to well structured diagrams that conform to well-defined conventions in specific formally defined domains. For reasons discussed in section 2.4.5, two classes of formal diagrams have been selected: "Process diagrams" and "Hierarchical diagrams".

Chapter 3

Accessible Diagrams: A Critical Review

This chapter reviews some relevant work on the accessibility of diagrams on the web. It aims at exploring the accessibility of diagrams and exposing associated problems. The effects of such problems for blind people are covered but the problem is seen as more general as it concerns all people. Indeed, all of us could become blind in some contexts. This could happen while our eyes are needed for other things (e.g. driving), low lighting levels, etc. In such situations the use of visual representations becomes inappropriate. To overcome the barriers faced while accessing diagrams on the web, it is important to understand the issues and limitations involved in making diagrams perceivable, operable, understandable and robust.

The main findings, obtained through the presented methods in this chapter, are discussed, leading to the identification of issues/requirements the proposed thesis will aim at addressing.

3.1 Diagrams on the Web

It is a moral obligation as well as a legal obligation to provide accessible information on the web. Appendix B defines accessibility as well as the legislative framework related to it.

Graphics over the web, “Web graphics”, such as diagrams, charts and illustrations are essential components of the content of the World Wide Web as it exists today. Therefore, it is important to consider accessibility of web graphics if the WWW is to reach its full potential. Due to their visual nature diagrams present a challenge as they are hard to access for blind people and people who work in an environment where visual representations are inappropriate. This section aims at highlighting the barriers/constraints faced while accessing diagrams on the web.

3.1.1 Graphic formats

A full review of the most commonly found formats on the web is presented as well as their main characteristics and limitations in Appendix A.

Currently most diagrams are represented using raster formats. For the purpose of the thesis these diagrams were named “vector images” (image of a vector graphic

represented in a raster format). Raster formats are inappropriate for perceivable, understandable and operable diagrams. The information concerning the diagram (objects and relationships between the objects) is lost as only the data describing the characteristics of each individual pixel is available within the format used. Even though metadata can be injected in most raster formats, this facility imposes some constraints as it can only be added on top of the file format and not at the individual level of objects requiring the diagram to be seen as one entity. So the description has to be done for the complete image. The information added could be enormous if it attempts to describe the diagram in detail or if it is going to be used to describe the diagram as a whole.

Vector formats present more opportunities than raster formats to make the information of the diagram available for accessibility purposes. Vector formats such as SVG, SWF and WebCGM are available on the web. Even though they offer accessibility features (some inherited from being vector formats (zoomable, plain text format, metadata added at level of individual object) and other proper to each individual format (e.g. SVG alternative equivalent)), they do not guarantee accessibility. Indeed if not authored properly with accessibility in mind they can be a challenge to access.

Currently SVG (W3C, 2001, DOUG and CULLER, 2009), which comes with a set of accessibility features (title, desc, grouping, comments, class attributes for styling, etc.), is the most appropriate vector graphic format on the web. It is a low-level graphics standard but it is defined in XML which offers many advantages as it can be used in conjunction with other web technologies. Even though notes on the accessibility features of SVG (MCCATHIENEVILE and KOIVUNEN, 2000) have been published by the W3C, no guidelines exist on how to author accessible vector graphics. SVG stores structural information about the graphic as an integral part of the graphic but the amount of structure is mainly author dependent. SVG does not capture diagrams at a high enough level of abstraction. It is more a “final form” presentation, which has some drawbacks in the direct creation of complex, highly structured, diagrams. Furthermore, most editors do not author or encourage authoring accessible SVG by correctly grouping elements or promoting the use of “title” or “desc” for example. A full description of the limitations of SVG is presented in (Appendix A section A.2.2B).

3.1.2 Guideline 1.1 and the WCAG 2.0 POUR Principles

According to the WCAG 2.0 (W3C, 2008c), four principles (POUR) provide the foundation for Web Accessibility: Perceivable, Operable, Understandable, and Robust.

- **Perceivable:** Information and interface components must be perceivable. Users must be able to perceive the information presented using at least one of its senses.
- **Operable:** Interface components and navigation must be operable. Users must be able to operate the interface.
- **Understandable:** Information and the operation of the interface must be understandable. Users must be able to understand the information presented as well as the operations the system offers.
- **Robust:** Content must be robust enough that it can be interpreted by a variety of user agents. Users must be able to access the content with current and future user tools.

These principles “*lay the foundation necessary for anyone to access and use Web content*” (W3C, 2008c).

These four main principles of WCAG 2.0 apply perfectly to visual web content such as diagrams but the actual guidelines do not really indicate the requirements to make this type of content (vector graphics) accessible.

The only requirement found (WCAG 2.0 Guideline 1.1) is the provision of “Text Alternatives” which does not apply to vector graphics but rather to vector images. Furthermore, for various reasons discussed in (Appendix C section C.2.2) this is not always appropriate as it does not ensure accessibility of such web content. If the description is missing, if a meaningless description is provided or if the diagram is too complex to be described textually, blind users or users who work in environments where visual representations are inappropriate, find themselves denied access to possibly important information. The support of assistive technologies (e.g. screen readers (see Appendix B section B.5)) allowing access to such alternative information is also limited for some occasions (e.g. when fall back mechanism for the object element is not well supported (see Appendix C section C.2.3)).

3.2 Current approaches to presenting accessible diagrammatical content

Many researchers have explored different approaches to make diagrammatic content accessible to blind or visually impaired people by presentation in other modalities such as textual, auditory, tactile or combinations of them (e.g. Audio/Tactile). These approaches have been termed “bottom-up approaches”. They start with a vector image, attempt to infer or extract information from this, and then generate alternative presentations.

These approaches have greatly contributed to resolving some of the accessibility problems of graphical content faced by visually impaired users. They provide useful methods and tools to interpret, present and navigate graphical information. However, they do have some limitations which need to be considered for the accessibility of diagrammatic content.

The different possible alternative modalities to access diagrams for blind people are reviewed followed by a detailed review of current bottom-up approaches to presenting diagrammatic content to blind people by generating alternative presentations in other modalities.

3.2.1 Existing alternative modalities to access diagrams non-visually

Blind people, deprived of the sense of sight, have to rely on other senses to access information, mainly touch and hearing. Different modalities (BAILLIE et al., 2003) have been explored: Tactile, Audio, Haptic, Audio/Haptic and Audio/tactile.

A. Tactile presentation access

In certain circumstances, diagrams can effectively be converted into a tactile diagram when appropriate and well designed. In these cases, tactile diagram become an invaluable resource of information.

But as stated by the RNIB National Centre for Tactile Diagrams (NCTD, 2008) *“Converting the visual graphic to an appropriate tactile graphic is not simply a matter of taking a visual image and making some kind of “tactile photocopy”. The tactile sense is considerably less sensitive than the visual sense, and touch works in a more serial manner than vision. Therefore the visual graphic needs to be re-designed by experts to make sense in a tactile form.”*

There exist a number of tactile formats:

1. Embossed diagrams are produced on a Braille printer which punches dots to make the diagram (e.g. Tiger Tactile Graphics and Braille Embosser).
2. Swell paper consists of microcapsule paper which swells on black areas when exposed to heat.
3. Vacuum formed diagrams using a thermoform printer.

Tactile diagrams are limited by the amount of information they can express. Tactile devices do not allow high resolution. Tactile diagrams require some practice to read and interpret, even with the simplification process they went through. Very complex diagrams can be really difficult to read with the touch sense alone. When reading a diagram tactually, the diagram has to be discovered by scanning it tactually for a mental image of the whole diagram to be built, whereas while reading a diagram visually the overall diagram is looked at and then the details can be read if needed.

This kind of diagram can be expensive to produce. This modality requires important resources and cannot satisfy the large amount of diagrams available due to the lack of appropriate devices. In the case of Tactile diagrams which are not refreshable; once produced they need to be reproduced if a modification is required making them not too adequate for diagrams changing over time. Much research is being carried out in order to address the issues of tactile diagrams, some exploring the feature of SVG in the facilitation of generation of tactile diagrams (ROTARD and ERTL, 2004), others attempt to automate the transcription process needed to produce accessible tactile diagrams (GONCU, 2009).

B. Auditory description access

Auditory access is a popular way of conveying information for the blind. There are different kinds of auditory access: speech and non-speech auditory access.

Screen readers and voice browsers rely on speech auditory access to convey the information on the screen to the blind user.

Sonification, an emerging success in accessibility (FRANKLIN and ROBERTS, 2004), uses non-speech sounds (i.e. pitch, timbre, volume) to represent and convey information. This method has proved to be an excellent way to convey the trend of a graph for example (BENNETT, 2002).

Auditory access of information within a diagram presents some drawback due to their nature. Audio differs in the way it conveys information. It provides information

sequentially. Furthermore, audio does not offer any external support for the exploration of the information presented. Complex diagrams presented in sound could stretch the limit of the working memory.

C. Haptic access

Haptic interfaces combine tactile and kinaesthetic sensing (force feedback interface) (ROTH and PUN, 2003) to perceive spatial information such as proximity, texture and shapes.

A mix of different modalities have also been researched, this includes: Audio/Haptic and Audio/Tactile modalities.

D. Audio/ Haptic access

Another modality explored (ROTH and PUN, 2003) to make diagrams accessible is a combination of audio and haptic access, most of the time involving force feedback devices and non-speech sound and/or synthesized speech. It has successfully been explored for graphs (YU et al., 2002) more specifically pie charts, line chart and bar charts.

E. Audio/Tactile access

Many researches have focused on improving tactile diagrams. A combination of tactile diagrams with audio has been of great interest in research in making diagrams accessible (KENNEL, 1996, WALL and BREWSTER, 2006, RNCB, 2009, VIEWPLUS, 2009). The audio/tactile access method can be used to make diagrams accessible to many blind people. Well made audio-touch diagrams can contain additional information that helps make complex diagrams accessible too. It has been proved to greatly reduce the mental effort needed to access diagrams.

A well known example is AudioGraf (KENNEL, 1996) from Kennel. AudioGraf presents a diagram in an audio-tactile way; the diagram is displayed on a touch sensitive panel and explored by the user who gets access to different levels of detail depending on the pressure applied on the panel through speech and non-speech audio.

Currently the two most popular audio/tactile devices are IVEO Touchpad (VIEWPLUS, 2009) and the Talking Tactile Tablet (T3) (RNCB, 2009). Text labels are attached to regions of the tactile diagrams which are placed on a touch panel connected to a computer. Then the diagram is explored by the user who navigates

with his hand on the tablet, each region pressed outputs an audio description of the label attached to it. These techniques have proved to be very useful to explore graphs, maps, charts, etc.

Although popular, these techniques are limited by the amount of information tactile diagrams can express and so cause problems on complex and rich diagrams. One solution proposed by IVEO is the use of SVG which allows the diagrams to be zoomed and printed based on the area of interest (see the IVEO project in the next section). The origin of the audio information is important, the more comprehensive these audio outputs are the better the experience but if they are badly designed the user might get frustrated and give up as a result. Using this modality, the user needs to search and discover the information instead of consulting it. It might be difficult for a novice user who might get discouraged.

3.2.2 Description of some current Bottom-up approaches

These bottom-up approaches can be organised into two categories:

- Infer alternative presentations from the original one
- Explore accessibility features of SVG

A. Infer alternative presentations from original one

In the first category, there is an attempt to infer alternative presentations from the one given by an author, by either using a third human party (MIKOVEC and SLAVIK, 1999, KURZE et al., 1995), or by applying a semi-automatic analysis (ELZER and SCHWARTZ, 2007, HORSTMANN et al., 2004b, FERRES et al., 2008). This process infers additional information. Tools are provided to enable users to navigate and explore such information.

❖ Blind Information System (BIS) and Graphical User Interfaces for Blind People (GUIB)

The “Blind Information System” (BIS) (MIKOVEC and SLAVIK, 1999) and “Graphical User Interfaces for Blind People” (GUIB) (KURZE et al., 1995) are two interesting projects that have proposed methodologies to tackle the problem of graphics accessibility for blind users. These approaches have made a significant practical impact by allowing blind users to comprehend graphical information. Although their approaches differ in detail, they share a common basis in the fact that they depend on human intervention by a third party, “a moderator”, not necessarily the author, who provides a description of the graphical information.

The “Blind Information System” (BIS) is a project, developed at the Czech Technical University (MIKOVEC and SLAVIK, 1999). It is a system for the interpretation of graphical information. It develops a hierarchical picture description methodology based on an XML grammar which characterizes pictures by structure and semantics. The result obtained is a text (XML) document describing the semantic and structural view of the picture. The description obtained depends on the person creating it. This person, who will probably not be the author, might omit important information from the description and provide an idiosyncratic interpretation.

In the context of the “Graphical User Interfaces for Blind People” (GUIB) project (KURZE et al., 1995) a two phase methodology is introduced, which is supported by software which allows the work of one sighted person to be used by many blind people.

The two phases are:

- Phase 1: a modelling tool supports a sighted person (the moderator) in producing a model of the visual graphic taking into account the representational and presentational aspects of the graphics and the interaction facilities available to the blind person.
- Phase 2: a presentation tool allows the blind person to interact, explore and experience the graphics without the assistance of a sighted person. This approach presents an obvious drawback: the moderator, who most of the time is not the author of the graphics, has an important responsibility. He decides what information to convey and imposes his view when the graphic is being read.

❖ **Summarizing Information Graphically Textually (SIGHT)**

In SIGHT (ELZER and SCHWARTZ, 2007) there is an attempt to automatically infer and convey the intended message a graph carries. SIGHT (ELZER and SCHWARTZ, 2007) is a system, implemented as a browser extension, enabling visually impaired users to gain access to a textual summary of the hypothesized intended message of the graphic. The information of the graphic is summarized by the system around this inferred message and then provided to the user who accesses it using a screen reader. This approach hypothesizes that “...*the core message of an information graphic (the primary overall message that the graphic conveys) can serve as the basis for an effective textual summary of the graphic (ELZER and SCHWARTZ,*

2007)“. The system is currently limited to simple bar charts and investigation for line charts is ongoing.

A Browser helper object (BOH) is created the instant an instance of the browser is created. It then intends to establish which object is in focus on the browser. Once the graphic has been detected it is passed to a Visual Extraction Module. This module analyses the image file and produces an XML representation containing the information about the graphic (for the type “bar chart”, it will contain numbers of bars, labels of the axis, label of each bar, height of the bars, etc.). The author specifies that for the moment the VEM only handles electronic images with certain fonts and with no overlapping characters. The VEM is also limited as it assumes only standard placement of textual information such as captions, labels and axis headings. For example, if a bar chart is detected, the system attempts to infer its intended message. The XML file is then passed to a “pre-processing and caption tagging module”. The pre-processing adds extra information judged to be important in the XML file based on the analysis of its content and the caption tagging extract information from the captions and then passes the file to the Intention Recognition Module (IRM). The IRM is charged to recognize the intended message of the graphic. The IRM reasons on three communicative signals to determine the intended message of the graphic:

- Relative effort required for different perceptual and cognitive tasks. This signal is based on the hypothesis that “the relative difficulty of different perceptual tasks serves as a signal about which tasks the viewer was expected to perform in deciphering the graphic’s message”.
- Importance of entities in the graphic which draw attention to important aspects of the graphic (e.g. different colors used, different annotations, captions, etc.).
- Presence of certain verbs and adjectives in the caption. The aim being to extract communicative signals from these.

The evidence extracted using the communicative signals, is used by the IRM Bayesian inference system to infer the intended message of the graphic.

The system has been evaluated for “bar charts”. Despite the advantages presented by the author (provide access to the communicative intent of the graphic, does not necessitate specialized hardware, no action required by the web developer) the results showed that the intended message inferred was not always the one actually

intended by the bar chart. This might result in conveying the wrong intent, which could lead in a misinterpretation.

❖ **Technical Drawings Understanding for the Blind (TeDUB)**

The TeDUB (TEchnical Drawings Understanding for the Blind) project which began in 2001, explored the possibility of a semi-automatic analysis of diagrams (HORSTMANN et al., 2004a, HORSTMANN et al., 2004b). The system developed generates descriptions of certain classes of diagrams automatically (electronic circuit diagrams, UML diagrams and architectural plans) and allows blind people to read and explore them on a PC with the aid of a computer games joystick and a screen reader. The diagrams used in the TeDUB project are taken from different sources. They can be raster graphics, vector graphics (SVG), file format with semantic content (e.g. XMI, CAD format).

The system is composed of two main parts. The first is the “DiagramInterpreter” which is the knowledge processing unit. It analyses the diagram provided, operates on a network of hypotheses until a semantic description of the whole diagram is found and converts it into a representation that is used by the second main part of the system: the “DiagramNavigator”. The latter allows the blind users to navigate and annotate these diagrams through a number of inputs and output devices.

In the event that the automatic process might lead to wrong results, the “ImageAnnotator” is used to make corrections. Their approach for the presentation and navigation of diagrammatical information offers many advantages for blind users. But as the authors point out, due to noise and distortions in the original data, the (semi-) automatic analysis of the diagram information may produce wrong results. To overcome this issue, the Image Annotator tool is provided to allow a human to correct and to improve the internal description generated. Human intervention and time may thus be needed.

❖ **iGraph-Lite**

The iGraph-Lite system (FERRES et al., 2007) is an open source project aiming at providing automatic short verbal descriptions of given graphs and some means of interaction to interpret and query the graph’s information. iGraph-Lite (FERRES et al., 2007) attempts to make the information carried by the graph accessible to users so they can explore the graph information and infer its intended message depending on his own need.

Contrary to SIGHT, the main objective of iGraph-Lite is of Natural Language based interactivity with the information of the graph.

The graphs are created using some standard applications (SPSS, Microsoft Excel, OpenCalc) with a specific plug-in installed on the application. This plug-in allows the creation of an XML file containing all the information regarding the structure and the data provided to create the graph. The XML file is then translated into a text file or a web page which can be accessed by a screen reader.

The iGraph-Lite system consists of three subsystems:

1. The P-System, parses the XML file and generates a logic version of the given graph taking into account mathematical properties of the data (e.g. maximum, minimum).
2. The C-System stores different kinds of rules for describing and querying the logic version generated by the previous subsystem.
3. The L-System, which is the language subsystem, takes the outputs from the previous subsystems, including all possible inferences, and generates a natural language text (general description or response to a query) plus the querying system.

The XML file can be explored and the data can be queried “sequentially” while a textual description is provided. The query system is said to be a “sequential” querying process as it allows the user to move from one point to another at a time.

Some navigation commands allow the current position in the graph to be indicated and values to be skipped, the \rightarrow and \leftarrow keys allowing the user to travel around the data.

iGraph-Lite presents some limitations. Some evaluation results of the project showed that some of the users needed to get more semantics, some important information are not inferred and so not conveyed to the user. Some information at the P-System level is inferred based on algorithms from a concept repository which is limited to a set of algorithms which might not be sufficient. Users also would like to get more interactivity with the graphs; the presently “sequential” query system is not adequate when looking for exact information. The system depends on a plug-in installed on the graphical application. This approach might cause some problems as the plug-in might not be supported by some graphical application and the currently supported applications might be evolving at a faster pace than iGraph-Lite.

B. Explore accessibility features of SVG

The second category includes approaches that have explored the accessibility features of SVG and have attempted to overcome some of SVG's limitations. Additional information is embedded in an SVG document (e.g. metadata, constraints, and alternative descriptions) and alternative presentations are generated based on added information.

❖ The Science Access Project (SAP) and ViewPlus project

The Science Access Project (SAP) group at Oregon University aims at developing methods to make information accessible for visually impaired people. In 1997, SAP started a research project to make vector images accessible. They obtained some useful results and acquired an understanding of the possibilities and limitations of audio/ tactile access of vector images. With the emergence of SVG they stopped this research project concluding that “there is no way to make vector images that are more or less automatically accessible as a separate accessibility page needs to accompany each vector image” (SAP, 2005).

They then developed the ViewPlus project (BULATOV and GARDNER, 2004). The project's current hardware/software product line has a group of SVG applications which can provide excellent access “to most” SVG graphical information for all:

- ViewPlus IVEO Creator application
- ViewPlus embosser
- ViewPlus TouchPad

A blind user can use the ViewPlus IVEO Creator application to import a diagram from the Web, paper copy, or any electronic file, and create an accessible IVEO SVG version. This diagram can then be printed to a ViewPlus embosser to create a tactile version, and it is then placed on a ViewPlus touchpad to read. When a region of the diagram is touched, it speaks the information attached to it. For most diagrams, the audio feedback and the tactile layout of a standard size diagram is quite accessible. However for complex, information rich, diagrams if a better view is needed, SVG lets one zoom and make bigger copies.

Many accessibility features of SVG have been successfully explored in the ViewPlus project (BULATOV and GARDNER, 2004), but some of its limitations have also been identified (GARDNER and BULATOV, 2001).

The accessibility features are of benefit only if used in the document creation process. If a document is not properly structured (e.g., careful use of <g>elements), it becomes very difficult to provide alternative presentations even if alternative descriptions are provided for individual elements. Some SVG documents become less accessible when created without <title> and <desc> elements, so to overcome these drawbacks SAP has created an SVG editor that permits addition of a title and description to elements that do not have them. Some SVG documents are very badly structured and therefore less informative. So occasionally, re-ordering the hierarchical structure of the SVG to organize objects into proper groups becomes necessary.

In the ViewPlus project the authors have taken an approach based on exploring the information behind the picture, but the solutions proposed are expensive in terms of adding extra information and/or reorganizing it. This illustrates the fact that SVG is in a sense too low level and does not contain enough information about the structure of the graphical information.

❖ **Constraint Scalable Vector Graphics (CSVG)**

An extension to SVG, called Constraint Scalable Vector Graphics (CSVG) (BADROS et al., 2001), has been proposed. It partially addresses some of the limitation of SVG by proposing additional capabilities, semantic zooming, differential scaling and semantics preserving manipulation. CSVG allows attribute values to be expressions whose values are determined at display-time. These additional capabilities allow alternate layouts for the same logical group of components in a diagram, which greatly improves SVG's value. Whilst this constraint-based approach permits a more flexible description of graphical content, CSVG remains close to SVG, and still captures diagrams at a similar low level of abstraction.

❖ **The SVG linearizer tool**

The SVG linearizer tool (HERMAN and DARDAILLER, 2002) generates a textual linear representation of the content of an SVG file by using a metadata vocabulary describing it. The author has to describe the SVG-content using this RDF vocabulary and to add textual descriptions to all elements that constitute primary RDF resources. Then, from this information plus information contained in the SVG file itself, an HTML file is generated. As described by the authors, the generation of the RDF metadata is done mostly by hand and this could be facilitated by appropriate authoring tools. Indeed, adding the RDF annotations is an onerous task for the author, and this operation can be tedious and not very efficient for complex diagrams.

This method is too dependent on the creator's patience and willingness to produce appropriate metadata (at least in the proof-of-concept tool). As proposed in (LEWIS, 2006), one possible alternative to make the approach easier for authors and less prone to errors would be to provide this facility through a semantically rich markup language, rather than requiring the author to semantically enrich the diagram. Furthermore, as the authors of the SVG linearizer tool (HERMAN and DARDAILLER, 2002) point out, "the specification of the right vocabulary is undeniably the hardest research issue to evolve this approach further".

3.2.3 Limitations of Bottom-up approaches

The approaches proposed in BIS and GUIB contain, as their authors point out (KURZE et al., 1995), an intrinsic drawback, i.e., the resulting description of the picture relies heavily on the expertise, analysis and indexing of a third party (not the author of the vector image most of the time). An important responsibility placed on the moderator is to decide what information to convey, and thus the moderator indirectly imposes a view when the picture is being read (which may result, for example, in an idiosyncratic interpretation of the information or in the inadvertent omission of important information).

In the approaches of TeDUB, SIGHT and iGraph Lite, the automatic (ELZER and SCHWARTZ, 2007, FERRES et al., 2007) or semiautomatic (HORSTMANN et al., 2004b) analysis of the diagrammatic presentation (vector image) may produce wrong results due to noise and distortions in the original data presentation which may need time and human intervention to correct and/or improve the information extracted (HORSTMANN et al., 2004a, ELZER and SCHWARTZ, 2007) or due to limitations of algorithms in charge of recovering semantic information behind the diagram (FERRES et al., 2007).

In the second set of approaches described (ViewPlus, CSVG and SVG Linearizer tool), the authors have identified the need to preserve the information behind the diagram by storing it within the format representing it (in this case SVG). They all try to explore the accessibility features in SVG and also attempt to overcome some of its limitations (see APPENDIX section A.2.2): the author dependent low level nature of SVG and structure of the information. The solutions proposed involved sometimes reorganising the structure of the SVG if the original is badly structured, enriching it by adding extra information (adding title and desc element when missing

or inserting RDF annotations). These solutions are expensive and onerous for the author who is willing to invest time and patience.

An important point noticed while analysing these existing approaches is that most of the limitations they present are not due to the way they allow the user to access and explore information but are due to the pitfalls inherent in trying to recover this information from the diagram. The absence of information “behind” the vector image which is lost at the creation stage is the main problem.

The following section discusses an example that illustrates the idea.

3.2.4 Summary of issues

A. Loss of information at diagram rendering

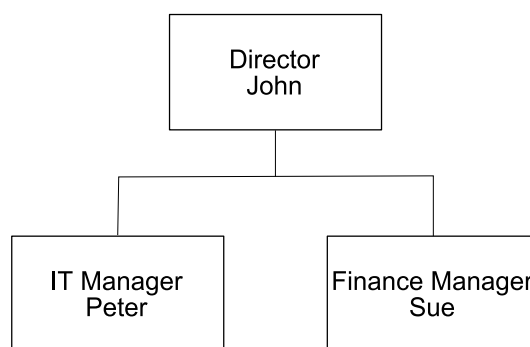


Figure 9: A Simple example « Do you see what I mean? »

Figure 9 is a diagram given without any additional context. This diagram is composed of three boxes organised into two levels and connected by undirected lines. Is it possible to infer what the boxes and the lines between them represent? What information is available about the relationship between Sue and John? One could assume that the three boxes represent three employees working for the same company. The connections might represent lines of command, implying a hierarchical relationship between the employees (e.g., John manages Sue). Another interpretation could be that this diagram represents salary grades. John is in the next salary grade above Peter and Sue, who are in the same grade. Many other interpretations are possible, e.g., a family tree, a floor plan. The brain infers information based on previous experiences, making one see and understand what one wants to see and was taught to see (NARAYANAN, 1997).

Without any contextual information provided with a diagram represented as a vector image it is difficult or even impossible to infer what the author of the diagram

intended it to mean. This is mainly due to the fact that the semantics of the diagram is not made explicit.

Over time many researchers highlighted the need of having this precious information behind the diagram available. In 1997, John A. Gardner (GARDNER et al., 1997) gave an overview of the concepts of “smart diagrams” (information behind a diagram) and intelligent diagrams browsers for accessing such information. He highlighted the fact that “*Nearly every part of smart graphics technology exists today, but to our knowledge there is no complete package that incorporates everything necessary to author a smart picture, incorporate it into an electronic document, and display it intelligently*”. In an attempt to identify “...next generation accessibility features in future graphics standards...” As mentioned in Chapter 2 section 2.4.4, Takagi et al. (TAKAGI and TATSUYA, 2007) made different surveys, one of which highlighted the complexity of existing business graphics which they categorized into four categories: simple graphics, formal diagrams, table structure and composite diagrams and graphics. (As mentioned in Chapter 2, the “formal diagrams” category is of interest in this thesis.) Takagi states: “*Formal diagrams are formally defined diagrams, which have a well defined data-model behind the visual structure, such as tree structures, Venn diagrams...UML diagrams. If non-visual conversion were defined for each data model and blind users learned the appropriate access method, then these diagrams should be accessible. In order to implement this approach, the graphics should have functions to preserve the data-model for each part of the figure*”.

Takagi also underlines that the most important information to be presented non-visually is the “represented information”, “*however no standards support bundling of the represented information in graphics. The idea of a “data-model attachment” is an important idea*”.

B. Author's willingness

The willingness of authors, to allow the capture and access of the diagram content information while creating their diagrams, is an important aspect to consider.

Given a diagram on the web, the sighted user can gain what information he can from its representation. The information that he can obtain depends on the author, the quality of the graphics system used and the appropriateness of the diagram or visualization techniques used. The information will in general, be tailored by the

author so that what the user sees and the information he acquires is what the author intended and what the author allowed him to acquire “What the author means”.

There is a moral and legal duty for the author to provide an equivalent perceivable view to the blind person. That also means that if the author has constrained the information provided to the sighted user for any reason then he should ensure that the same constraints are applied to the information provided to the blind user. For this to be possible, the author needs to generate an alternative equivalent view that does not need the diagram to be seen. To deal with situations where authors decline to do that, researchers have come up with a number of “bottom-up approaches”. Such approaches are the best that can be achieved given that authors refuse to provide access to the information behind the diagram and there is no jurisdiction to force them doing it. These “bottom-up” solutions would satisfy both the moral and legal requirements.

C. Graphical format

The alt-text approach and most of the other approaches described previously are as good as they could be for images but if the image is a vector image then better accessibility could be achieved by first using the appropriate format: “vector graphic format” and capturing the information behind it.

To summarise, if all one has to work with is a raster image, the bottom-up approaches (adding descriptive information to an existing diagram) are appropriate. In the “Do you see what I mean?” context, this is starting with what is seen, which is all that is available.

3.3 Diagrams processing

As argued in Chapter 2, diagrams are excellent tools for conveying and communicating information. They offer powerful advantages for our ability to access, process and memorize information presented.

These main features are what make diagrams different to other forms of representation such as text (LARKIN and SIMON, 1987) or sound (BROWN et al., 2004). Because of their sequential nature, text and sound do not present the same advantages when processed.

While diagrams can be used as external memory (SUWA and TVERSKY, 2002) presenting complex diagrams textually or aurally presents some challenges on

the working memory, especially if the diagram represented is complex. As introduced by Miller (MILLER, 1956) short term memory is limited at around 7 ± 2 “chunks” (elements). So as stated by Brown (BROWN et al., 2004) “*if a diagram contains more than 7 ± 2 items of information, it is unlikely that a user will be able to build a complete mental representation of the diagram unless some chunking takes place*”.

Having an idea about the powerful advantages diagrammatic representations offer over other representations such as textual or auditory and highlighting the limitations the currently used approaches present, the principle question arises of how to make these visual representations perceivable, operable and understandable? What is needed? What is missing?

A summary of how sighted people process and interact with diagrams is researched in order to gain an understanding of how the information presented diagrammatically is accessed by sighted people. The challenges of non-visual presentation of diagrams have then been researched.

3.3.1 Visual processing of diagrams

Shneiderman (SHNEIDERMAN, 1996) summarizes the basic principles of visual design as the Visual Information seeking Mantra: “Overview first, zoom and filter, then details on demand”. Through this mantra, the importance of an overview is highlighted as it is the first task of an interaction with a diagram.

A person performs different tasks when faced with a diagrammatic representation (SHNEIDERMAN, 1996, CARD et al., 1999). The seven tasks identified by (SHNEIDERMAN, 1996) are:

- **Overview:** gaining an overview of the entire diagram when needed.
- **Zoom:** Zoom in on specific elements of interest.
- **Filter:** Getting rid of unwanted elements by hiding them from the visual representation for users to be able to control the viewing of information and focus on specific elements that interests them.
- **Details-on-demand:** Getting detailed information from specific elements of interest.
- **Relate:** Viewing relationships among elements of the diagram.
- **History:** History of actions which consists of keeping track of actions performed by the user allows him to retrace their steps when needed by actions such as undo, replay or progressive refinement.

- **Extract:** Allowing the search and extraction of specific information through queries which allows users to get answer to specific questions.

Shneiderman emphasizes that a system should support the full task list, because this allow information to be presented rapidly and for rapid user-controlled exploration. He also specifies that such support would require a novel and specialized 'data structure' among other things. Most of these tasks explore the powerful advantages offered by a diagram and are easily performed by the human visual system when exploring the diagram visually. But having access to all these tasks in a non-visual environment (e.g. for visually impaired users) is a challenge with actual diagrams (represented as vector image or vector graphics). Thought needs to be given to how to allow these tasks to be performed on diagrams, what is needed? Or rather how to design diagrams in such a way that it is possible to enquire and reason about the information they carry, making them more perceivable, operable and understandable, thus allowing any user regardless of his abilities to perform these tasks to access and explore the information presented by the diagram.

3.3.2 Issues of Non-visual presentation of diagrams

What aspect of the visual representation of the diagram needs to be communicated non-visually? What kind of information needs to be conveyed? Many research projects have addressed these questions (BENNETT, 2002, BROWN et al., 2004).

The information presented will depend on the alternative modality used to communicate the diagram non-visually (e.g. tactile, audio, audio/tactile, etc.). When presenting the information of the diagram by touch the information concerning the position of the objects composing the diagram is conveyed as for the nature of the tactile modality, but is this essential information in the accessing and processing the information?

The benefit of presenting co-ordinate positions of the objects composing diagrams has been investigated by David Bennett (BENNETT, 2002) and has proved to provide no benefit in the non-visual presentation of diagrams in audio. This comes back to the definition of diagram in Chapter 2: "A diagram is a simplified and structured visual representation of concepts and relationships between them used to represent and clarify a topic. Diagrams are composed of shapes and text. The

information concerning the concepts represented and their relationships is the important information whereas its layout is not”.

So the information behind the visual representation of the diagram “the information content” is the information to be conveyed non-visually.

Aiming at demonstrating his hypothesis that “the way in which a blind person can access diagrammatic information will affect the ease of performing tasks”, David Bennett developed two models of navigation presenting the content information of a diagram. The first one presents the hierarchical structure of the information and the second one presents the connection-based structure. Through a set of experiments he validated his hypothesis which confirmed that the navigation model used has a significant effect on the performance of the user’s task.

The theory behind the result obtained by David Bennett has been explored by Andy Brown who (BROWN et al., 2004) looked at the non-visual presentation of graph based diagrams taking into account the cognitive science behind the use of such representations. Keeping in mind the main advantages diagrammatic presentations offer for sighted readers (Chapter 2), he identified a collection of issues any system presenting non-visual diagrams should take into account:

- **Recognition:** One main advantage of a diagram is its ability to make some implicit information, which would normally need to be deduced from other forms of presentation, explicit. A mechanism to provide access to implicit information within the diagram presented non-visually should be provided if this representation is to be an alternative equivalent to diagrams (in specific domains it should be possible to pre-determine a set of specific aspects to be looked at (e.g. cycles in graphs)).
- **Task dependence:** The recognition of implicit features in a diagram depends on the type of diagram and the information that is needed which is related to the nature of the task intended. What is meaningful depends on the nature of the information presented and the interests of the reader.
- **Overviews:** when using the sense of sight the whole diagram can be seen offering instant access to all of the information presented. The diagram provides an external reference which can be used as an external memory support. Some sort of support, substituting the loss of such external memory support, should be provided for the exploration of a diagram

presented non-visually. Andy Brown suggests that overviews can be integrated into a hierarchical data structure as presented and explored by David Bennett.

- **Search:** a facility to search for specific information within the diagram is essential in providing a substitute to the lost ability of scanning the diagram visually for specific information. Connection-based navigation, which allows finding related objects, is needed.
- **Representational constraints:** consideration should be taken into deciding on the resemblance between the diagrammatic presentation presented visually and its non-visual presentation. What needs to be presented? A presentation that resembles the visual diagram or the information the diagram represents? This would depend on the context the diagram will be used in and the modality it will be presented in (e.g. tactile, audio, etc.). If it is to be used in a collaborative environment where the diagram is used as an essential means of communication between members of a team to discuss and share ideas then a common language is needed to discuss the diagram in both presentations (visual and non-visual).

The previously presented factors give an overall picture of what would be required for a system to present diagrams visually or non-visually keeping in mind the advantages diagram offer when presented visually.

It is noticed that most of the tasks identified by Shneiderman as needed to allow user-control exploration of the information presented by a system, also apply to systems aiming at presenting diagrams non-visually.

Providing an overview as a hierarchical structure of the information of the diagram is important and provides a means to reduce working memory load. To facilitate searching and recognition of the information careful attention needs to be given in offering appropriate navigation models (hierarchical and connection-based). The importance of giving the user control over the information he wants to navigate and access is highlighted for both visual and non-visual presentation of diagrams.

3.4 Accessible Diagrams Requirements

The previous analysis helped in highlighting a number of issues and limitations current approaches present for the problem of making diagrams accessibility on the web. Online resources such as email discussion lists (WebAIM, BCAB, yahoo groups (blindwebbers, blindtech), freelists accessible image), forum proceedings (T2RERC, 2003), reviews of forum discussions, etc. have also provided valuable input.

Three main issues have been identified:

- loss of information at diagram rendering
- lack of accessibility support for both image and vector formats
- presentation modality issues

Hence a set of requirements that a new approach should satisfy in order to overcome these has been identified.

These requirements should be incorporated into a system aiming at representing diagrams in a way that it is possible to enquire and reason about the information they carry, thus making diagrams more perceivable, operable and understandable and as a result more accessible. Such an approach should not replace existing “bottom-up” approaches but rather complement and enhance them.

The requirements are organised around the three main issues identified and are summarised in Table 4.

3.4.1 Loss of information at diagram rendering

As mentioned by (RIBERA, 2008) “*the final format should allow its content, its presentation and its interactivity to be manipulated independently in order to personalize each one according to the user’s preferences*”. The problem with most current ways of creating diagrams is that information is lost, the only thing which remains is a visual graphical presentation presented through an array of pixels. The absence of information “behind” the diagram which is lost at the creation stage is the main problem. It would be more useful if the information behind the diagram was made available. This would ensure the information provided to the user is the same as the information that the author of the diagram intended to provide. In other words, no information is missing, distorted or inferred.

The following requirements have therefore been identified:

- Capture, encode and preserve the original information behind the diagram at the creation stage.
- Present the diagram from this.

3.4.2 Lack of accessibility support for both image and vector formats

The format used should be adapted to the nature of diagrams which are defined as structured sets of objects and relationships between these objects.

The following requirement has been identified:

- Use an appropriate format to store the information of the diagram.

3.4.3 Presentation: modality issues

The requirements identified for the presentation modality issues are proposed in order to address the issues identified by Andy Brown in section 3.3.2 (Recognition, Overviews, Search, Task dependence and Representational constraints).

The following requirements have been identified:

- Provide a mechanism to provide access to implicit information present in the diagram information (Recognition), by making it explicit.
- Provide mechanisms of exploration/ navigation of the information.
- Provide an overview of the information (Overviews).
- Provide a mechanism to process and search the information presented (Search).
- Provide the user with the power to decide which information he wants to obtain depending on the nature of the task he wants to achieve (Task dependence).

Issues	Requirements
Loss of information at diagram rendering	Information
	<ul style="list-style-type: none"> • Capture, encode and preserve the original information behind the diagram at the creation stage. • Present the diagram from this
Lack of accessibility support for both image and vector formats	Graphical format
	<ul style="list-style-type: none"> • Use an appropriate format to store the information of the diagram.
Presentation modality issues	Presentation modality
	<ul style="list-style-type: none"> • Provide a mechanism to provide access to implicit information present in the diagram information (Recognition), by making it explicit • Provide mechanisms of exploration/ navigation of the information • Provide an overview of the information (Overviews). • Provide a mechanism to process and search the information presented (Search) • Provide the user with the power to decide which information he wants to obtain depending on the nature of the task he wants to achieve (Task dependence)

Table 4: Accessible Diagrams Requirements

Once the requirements were identified, it was noticed that they are the same as those for non blind people accessing and processing information within a diagram (section 3.3.1). In (section D.2), a table has been created to provide an overview of all the existing approaches reviewed.

3.5 Discussion and Conclusion

This chapter has explored and exposed a number of issues and limitations current approaches present in to the problem of diagram accessibility on the web. The absence of the information “behind” the diagram which is lost at the creation stage has been identified as the main problem. This lead to the identification of a set of requirements a new approach should satisfy in order to overcome this.

The following table (Table 5) presents an overview of the main existing approaches against the defined requirements.

	Requirements	TeDUB	SIGHT	iGraph Lite	OntoDiagram	SemViZ
Information	<ul style="list-style-type: none"> Capture, encode and preserve the original information behind the diagram at the creation stage. 				✓	✓
	<ul style="list-style-type: none"> Present the diagram from this 				✓	✓
Graphical format	<ul style="list-style-type: none"> Use an appropriate format to store the information of the diagram. 	Vector image	Vector image	Vector image	Vector image	Vector image
Presentation	<ul style="list-style-type: none"> Provide a mechanism to provide access to implicit information present in the diagram information (Recognition), by making it explicit 					
	<ul style="list-style-type: none"> Provide mechanisms of exploration/ navigation of the information 	✓		✓	✓	
	<ul style="list-style-type: none"> Provide an overview of the information (Overviews). 	✓	✓	✓	✓	
	<ul style="list-style-type: none"> Provide a mechanism to process and search the information presented (Search) 			✓	✓	
	<ul style="list-style-type: none"> Provide the user with the power to decide which information he wants to obtain depending on the nature of the task he wants to achieve (Task dependence) 			✓	✓	

Table 5: Overview of existing approaches against the defined requirements

The TeDUB, SIGHT and iGraph Lite approaches have been created with accessibility in mind whereas OntoDiagram (Chapter 2) and SemViZ (Chapter 2 section 2.6.6) have been created in an attempt to automatically generate expressive and effective diagrams from high-level representations of a given set of “raw data”. Both categories of approaches present different benefits and limits in the fulfilment of the requirements.

TeDUB, SIGHT and iGraph Lite have produced some good results in terms of presentation of accessible diagrams non-visually but still have some limitations mainly due to the absence of information “behind” the image which is lost at the creation stage and the pitfalls inherent in trying to recover this information. In the “Do

you see what I mean context (section 3.2.4A)”, this is starting with what is seen, which is all that is available.

OntoDiagram and SemViz present other possibilities as the original information from which the diagram was generated is available. They are starting with “What I mean” and generating “what you see” providing links from the presentation to the underlying information. They have explored the use of ontologies in creating diagrams. But this only applies in a single medium. Indeed, none has taken into account or explored alternative modalities for presenting the information generated in a variety of presentation media. Furthermore, the possibilities offered by the use of data models and ontologies for enquiring and reasoning over diagrams have not been explored by any of these methods.

This research proposes a solution derived from the benefits and limitations of these approaches. It is hypothesized that defining a smart diagram system that stores the original information, and provides details of the transformation performed to generate the filtered view of the information, and allows alternative transformations on the information to provide different presentations is a solution to the problem of diagram (accessibility) perceivability, operability and understandability on the web.

Various accessible presentation forms could be generated but it is not the aim of the proposed approach to create a new type of accessible representation but more to draw inspiration from existing “bottom-up” approaches previously presented which are specialized at that level.

Furthermore, if authors decided to use such a system, the resulting diagram would be information rather than just a graphical presentation of that information and as a result would offer enhanced accessibility benefits without further work from the author. On another hand, if authors choose not to use the system, and it is assumed many would not for a variety of reasons (e.g. protect data, do not want to invest effort, etc.), then it is up to them to satisfy the accessibility issues by some other methods (e.g. existing approaches).

The requirements should be incorporated into a system aiming at representing diagrams in a way that it is possible to enquire and reason about the information they carry, thus making diagrams more perceivable, operable and understandable.

Chapter 4

The GraSSML Approach

This research proposes a solution to the issues identified that fulfils the requirements listed (Chapter 3 section 3.4). The hypothesis underlying the proposed approach is that “if information on the structure and the semantics of formal diagrams were preserved, made part of the diagram by willing authors at the creation stage, these diagrams would be more perceivable, operable and understandable and, as a result, suggest enhanced accessibility benefits for such diagrams”. Graphical content is no longer thought of as ink on the paper or pixels on the screen but more as an abstract entity that has an intrinsic structure and semantics.

This chapter presents this novel approach called Graphical Structure Semantic Markup Languages (GraSSML), a three-level conceptual architecture that captures and provides access to this “information behind the diagram”.

GraSSML considers the possibilities the semantic web vision offers to the ability to enquire and reason over diagrams and its benefit to problems such as accessibility of diagrams.

The chapter starts by specifying the terminology used in the remainder of this thesis. The proposed approach is then introduced giving a description of the main ideas. Then the three levels: the semantic level, the structure level and the presentation level and the transformations between these levels are described in more detail. Use cases presented in the next chapter illustrate the idea.

4.1 Terminology

The terminology is based in part on the glossary of the Web Content Accessibility Guideline 2.0 (W3C, 2008c).

- ❖ **Content:** the information and sensory experience to be communicated to the user by means of a user agent, including code or markup that defines the content's structure, presentation, and interactions. The qualified nouns *web content* and *graphical content* are used in an analogous sense.

- ❖ **Presentation:** the “rendering of the content in a form to be perceived by users” (e.g., as print, as a two-dimensional graphical presentation, as a text-only presentation, as synthesized speech, as Braille, etc.).
- ❖ **Representation:** used in the sense in which it is used in REST (REpresentational State Transfer), a phrase coined by Fielding (FIELDING, 2000) to describe the architecture of the World Wide Web (W3C, 2004a). Web resources are referenced using URIs and a *representation* of the resource is returned. In the context of this research representations are expressed in an XML-based data format (markup language).

4.2 Approach

The GraSSML approach is to start with the meaning behind a diagram (instead of its graphical presentation) and then generate one of its possible presentations without losing access to the initial meaning. There are thus a minimum of two levels in this approach (Figure 10). The “top” level represents the meaning of the information communicated by the diagram: its semantics. The “bottom” level represents the possible presentations of this semantics.

The existence of sets of (predefined) rules which allow the “bottom level” to be generated from the “top level” is posited. Two different sets of rules are identified:

- Domain specific structural rules (notational conventions): define the diagram components (primitives and attributes) required by the domain and the class of diagrams it belongs to.
- Preferential presentation rules: these rules express preferences concerning the representation of the diagram in terms of aesthetics preferences or practical preferences which depend on a user’s preferences and/or device requirements. These rules are considered important for implementation of “adaptability” features.

The identification and differentiation of these two sets of rules justify the existence of an intermediate structure level aiming at providing a logical description of the diagram. Once the “bottom level” has been generated using these rules, it becomes possible (if the rules are carefully defined) to travel from one level to another in a reversible manner.

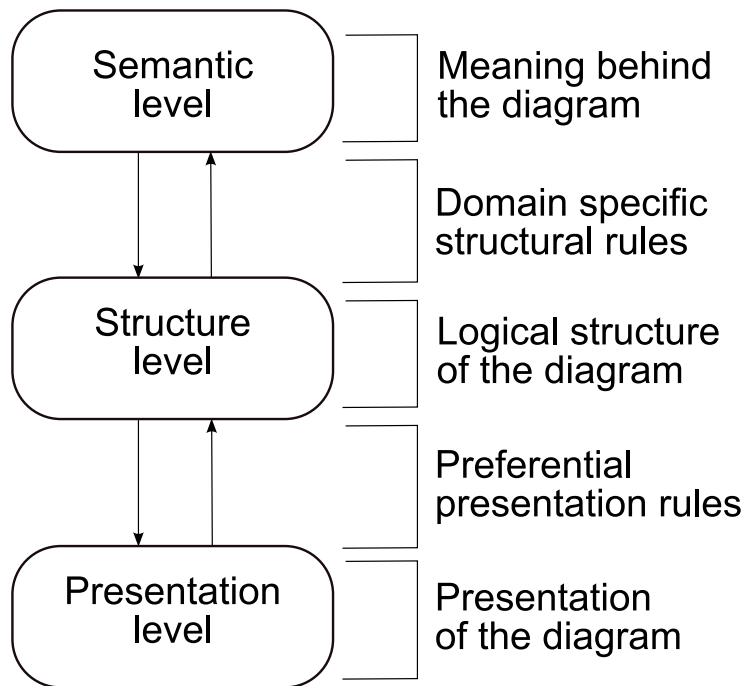


Figure 10: GraSSML approach

The GraSSML approach is not applicable to all kinds of diagrams but is aimed (see Chapter 2 section 2.4.5) at formal diagrams which have a well-defined data model behind their structure. This data model captures the information on which the diagram is based. These are well structured and conform to well-defined conventions. Two classes of formal diagrams have already been identified: “process diagrams” and “hierarchical diagrams”. Both classes considered are representative of a wider range of diagrams and have been used to demonstrate the feasibility and applicability of the proposed approach (Chapter 5). In addition to organizational charts (hierarchical diagram), UML activity diagrams (process diagram), GraSSML has also been applied to “Charts” (Chapter 6), more specifically charts in the financial domain (e.g. Bar charts, Pie charts).

An overview of the GraSSML system is presented in Figure 11.

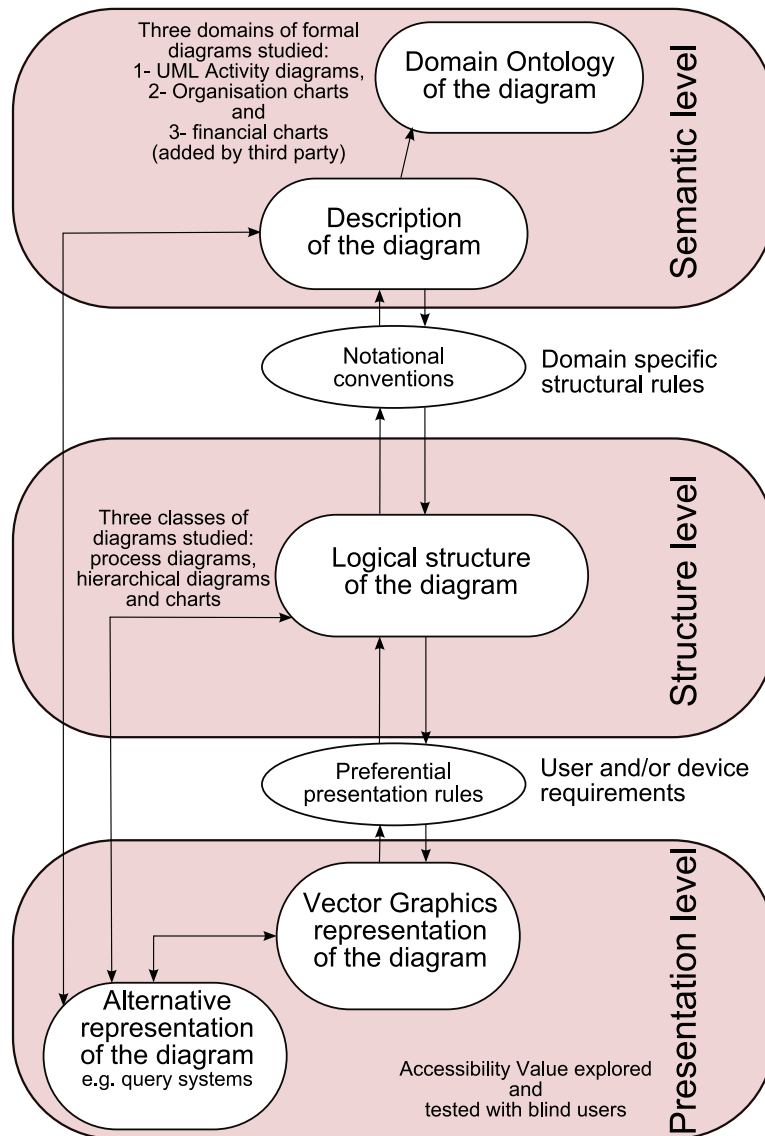


Figure 11: GraSSML system overview

4.3 Conceptual architecture

The conceptual model is presented in (Figure 12). Each GraSSML level captures a specific aspect of a diagram. The rendering processes that generate presentations from representations (e.g. an XHTML user agent) are not shown as these are external to GraSSML.

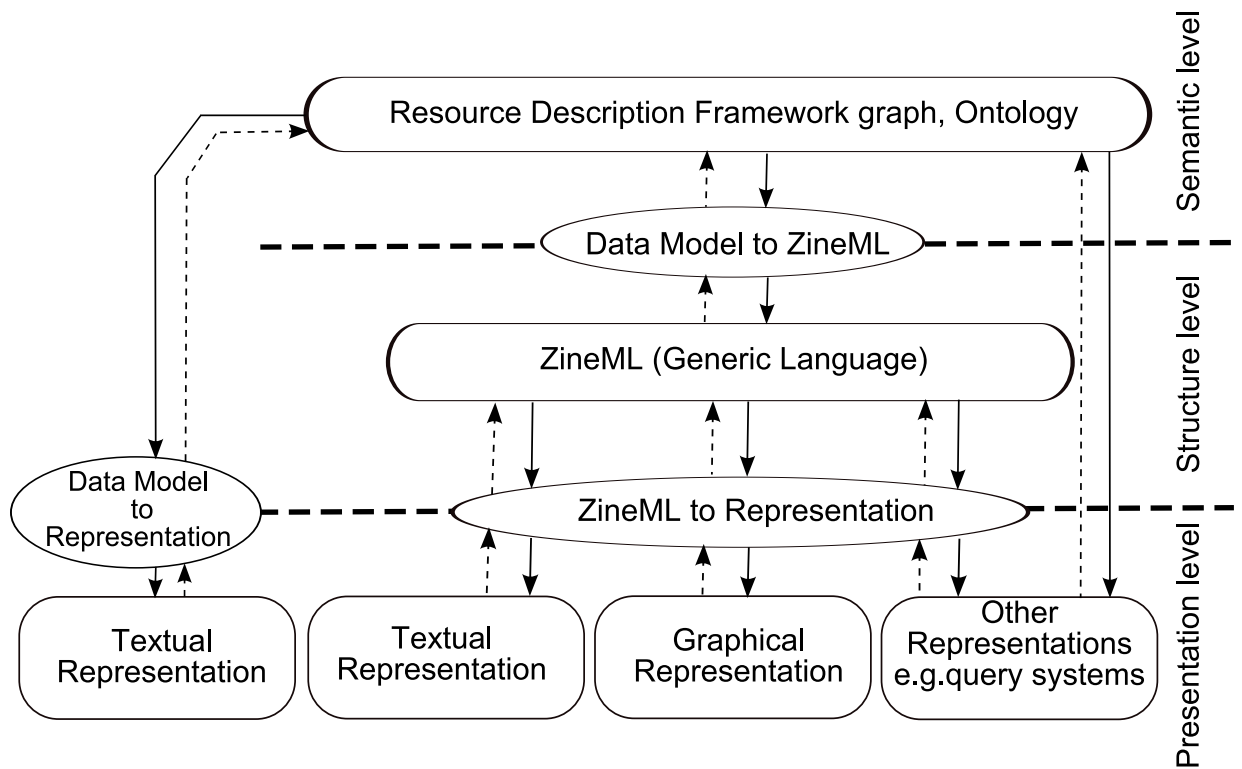


Figure 12: GraSSML Conceptual Architecture

This layered conceptual model reduces a task to a sequence of transformations between inputs and outputs expressed in different “Little Languages” (BENTLEY, 1986). The basic model does not depend on specific technologies but technologies are needed to refine the abstract model into the GraSSML architecture. An important step is to express the structure and semantics of the diagrams formally.

The three levels of GraSSML (semantic, structure and presentation level) and the transformations between these levels are described in the following sections.

4.4 Semantic level

Referring to the example illustrated in Chapter 2 (section 3.2.4A), at this level the question how to capture “What I mean?” arises.

The description of a diagram at the semantic level is captured by a data model which makes use of domain specific ontologies.

4.4.1 Ontology

For each application domain, information concerning the concepts used, the relationships between them, and the meaning behind each is required. The ontology is created by the domain expert. A domain expert needs to study the domain and the class of diagrams to identify what concepts, properties and constraints are needed to

represent the diagrams. The domain expert is in the best position to determine what fundamental information is needed in order to have a full understanding of the information to be conveyed. As was pointed by Herman and Dardailler (HERMAN and DARDAILLER, 2002), the specification of the right vocabulary is fundamental.

The process of constructing an ontology is well explained in (NOY and McGUINNESS, 2000). It is important to keep in mind that there is more than one way to represent a domain. The best representation depends on the application anticipated. The formulation of the ontology is an iterative process as the ontology is modified until the required result is obtained. The predefined set of queries defined at the initiation phase are useful in determining if the ontology is complete enough to resolve them.

The ontology could either be created, automatically generated from some source or be a pre-existing ontology. The possible use of pre-existing ontologies is an important aspect of the GraSSML approach as the creation of the ontology describing the background domain could be seen as complex and costly in term of time and effort. Furthermore, there already exist a large number of ontologies that have been created for various purposes. The ability to visualize, enquire and reason over heavy data conforming to these existing ontologies is one of the main benefits of GraSSML. This point is further discussed in Chapter 8 (section 8.3).

Along with the specification of a particular ontology, the domain expert defines the notational conventions that would govern the graphical representation of the information captured in the data model. The information might, for example, be represented in a diagram (graphical or tactile), or as written or spoken text. These transformations are captured in sets of rules (e.g. Data Model to ZineML) governing the generation of ZineML at the structure level from the data model. These transformation rules are described in detail in section 4.7.1.

4.4.2 Data Model

A Data model, an instance of a diagram which conforms to an ontology, lies at the heart of the GraSSML approach. GraSSML is agnostic to the source of the data model. GraSSML can be presented with any data model that conforms to an existing ontology. The specification of the ontology is important for the approach to successfully process the data model provided. Indeed, it is important to remember that

one possible data model representation with many semantics is possible as it depends on the ontology attached.

The ontology has a key role in the approach as having access to the data model and the ontology to which it conforms, allows implicit information to be made explicit. This offers the ability to enquire and reason about the information on which the diagram is based.

The data model could be hand authored, created using a tool, or generated by an application (Chapter 6 section 6.3.3B).

4.5 Structure level: ZineML

An intermediate structure layer has been introduced in the transformation from the semantic level to the presentation level. One could argue that it is possible to map directly from the semantic representation to the diagram presentation making the structure layer redundant. This mapping is possible but adaptability would be lost in the process. Indeed, as explained earlier section 4.2 (Figure 11), this layer has been added for specific reasons.

This level is intended to distinguish between the notational conventions followed by a presentation and the marks on the screen. This is necessary to obtain a presentation of the diagram that takes into account user or device requirements while respecting the notational conventions imposed at the semantic level. There is an interesting parallel between the role of the intermediate level in this model and the role of intermediate layers in graphics standards to promote device independence (ARNOLD and DUCE, 1990).

At this level a generic language that captures the structure of a diagram has been defined, called ZineML. Figure 12 shows the position of ZineML in the GraSSML system.

ZineML is at a higher level than SVG in representing the structure of the diagram and facilitates the creation and modification of diagrams. ZineML documents aim to be readable by humans and to give good overviews of diagram structures. The language seeks to be rich enough to allow accessible alternatives of the structural representation of a diagram to be created. ZineML provides the diagram components required by the diagram domain being presented. The set of primitive shapes differs from one diagram class to another and the constraints on positioning and connectivity differ. For example, when representing a UML activity diagram, the horizontal and

vertical layout of the diagrams is quite flexible but the shapes, of the concepts represented, (e.g. actions, decisions, initial activity, etc.) are precisely defined as are the types of connectors to be used. For organisation charts, the positioning in at least one direction is defined by the diagram class. This intermediate level binds the notational conventions in a coordinate free environment.

In the conversion from ZineML to its graphical representation there may be flexibility allowed in primitive placement, where connectors are attached to boxes, what rendering styles to use that are mainly for aesthetics or adaptability of a specific diagram. The constraint based layout engine (Figure 14) uses this flexibility in achieving the conversion to the appropriate graphical representation.

The transformation from ZineML to a presentation determines and adjusts positions and sizes using predefined presentation algorithms. The derivation of graphical or other presentations from ZineML is done in accordance with predefined rule sets (ZineML to Representation). These rule sets combine two levels of rules:

1. Notational conventions defined at the semantic level and passed on to ZineML (Data Model to ZineML), for example the geometric shape used to denote a particular semantic category.
2. Preferential presentation rules: these rules express preferences concerning the representation of the diagram in term of aesthetic preferences or practical preferences which depends on users' preferences and/or device requirements. These rules are considered important for implementation of "adaptability" features.

In consequence, ZineML needs to be more flexible than most intermediate languages. It needs to separate the primitives and attributes required by the diagram notational conventions from those required by the layout engine. This allows inquiries about the diagram to respond with information relevant to the diagram domain and not its aesthetic layout.

ZineML has a wider range of options than similar intermediate languages with the ability to divide the functionality into that required by the diagram domain and that required by the layout engine.

As some domain specific languages already exist, ZineML offers the possibilities to combine GraSSML with Standard notations or exchange formats for different domains, like for example XMI (OMG, 2007) for UML. Indeed, it is possible to generate ZineML from another domain specific language as well as the

other way round. The UML Activity Diagram use case (Chapter 5 section 5.3) demonstrates how a standard like XMI could be integrated within GraSSML.

As noted earlier, ZineML caters for diagrams such as process diagrams and hierarchical diagrams. ZineML defines a set of basic shapes (Figure 13) that cover a wide range of possibilities for such diagrams common in the domains considered. A predefined library of diagrams was used as a starting point (Figure 1 and Figure 2) and a set of basic shapes used in each diagram was extracted. It is important to keep in mind that it is impossible, a priori, to find all possible shapes since the diagrammatical notation has no limit. Indeed, users often imagine and create their own specific shapes. So ZineML needs to be extendable, and allow the addition of new shapes defined by users. To define a new shape, the user first needs to attribute a name identifying the new shape. This shape would be linked in the notational convention to its appropriate concepts it is intended to represent. The graphical representation of the new shape needs to be defined at the presentation level where the renderer will generate it using its user defined presentation aspect and integrate it within the diagram graphical representation.

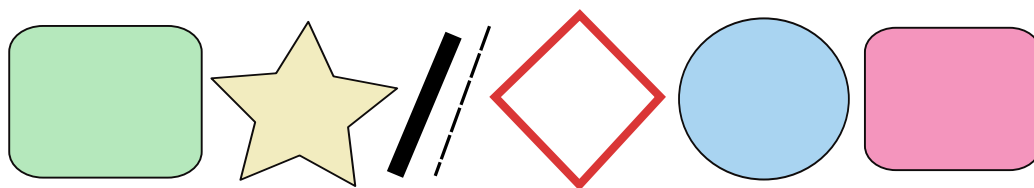


Figure 13: Examples of basic shapes defined by ZineML

ZineML is not domain dependent. It proposes options that express and apply rule sets when creating the diagrams. These rule sets can be used to tailor the diagram specifically to a domain. ZineML can determine and adjust positions and sizes semi-automatically, with a minimum of effort from the author for creation and modification of a diagram.

The graphical presentation of ZineML depends on the user predefined Structure to Presentation rule sets. The following options have been taken into account in the design of ZineML.

- The author wants to create a diagram without giving any explicit size and positional information. In this case a default algorithm defines default size and positions for each element of the diagram. For example, each shape may adjust itself to its content, which can be a text string or any other

shape. If the default diagram obtained does not suit the user, he still has the possibility of changing the rules applied by selecting another presentation algorithm or by using one of the following options that requires the user to give more explicit information on how he would like the structure to be presented (e.g. drawing direction).

- The author specifies positions and sizes of the elements. For example, the attributes “bottom”, “top”, “left”, “right” for the alignment, or “h” and “w” for the height and width of a shape, the attribute direction (down | up | right | left) to specify the drawing direction.
- The author creates the diagram with a specific set of global constraints. The specification of these constraints allows an author to define the syntax that all valid diagrams of this domain have to conform to. These constraints are defined in the set of rules used to represent the diagram: size of graphical element, relative positions, valid relations, etc.

Whatever the option used, if any modification is made, the presentation algorithm maintains the structure of the diagram by reapplying itself to the modified ZineML file, respecting the rule sets defined.

4.6 Presentation level

At the presentation level, diagrams are represented by modality specific languages.

Both the structure and semantic of the diagram are made available and both types of information can be conveyed.

To make the most of this valuable information available within the diagram, careful attention needs to be taken in the design and implementation of presentation and navigation tools, taking into account both user and device requirements. To present and explore this information in the best possible way and to provide presentations (visual or non-visual) as perceivable, operable and understandable as possible, specific studies are referred to.

Results from Chapter 3 are referred to in order to determine how to best explore the availability of the “information behind the diagram”, how best to present this information visually (SHNEIDERMAN, 1996) and non-visually (BROWN et al., 2004).

As mentioned in Chapter 3, a person can perform different tasks when faced with a diagrammatic representation. Shneiderman (SHNEIDERMAN, 1996) identified a list of tasks (overview, zoom, filter, details-on-demand, relate, history, extract) a system should support for the information of a diagram to be successfully presented and explored by a user. These issues concerning the non-visual presentation of diagram have been identified by Brown (BROWN et al., 2004) and needs to be kept in mind (recognition, task dependence, overview, search, representational constraints).

Thought needs to be given to how to allow these tasks to be performed on perceivable, understandable and operable diagrams or more specifically how to design perceivable, understandable and operable diagram presentations to allow any users or machines to perform these tasks and enquire and reason about the information.

In this area the GraSSML approach builds on knowledge acquired in the “bottom-up” projects described in Chapter 3. These projects have developed and successfully demonstrated the effectiveness of certain techniques in presenting graphical information to visually impaired users by successfully addressing some of the issues identified by Brown in the presentation of diagrams non-visually.

It is important to specify that the aim of this research project is not to develop new techniques in presenting graphical information effectively but to extrapolate from the existing techniques which have been proved to work.

The difference in the GraSSML approach is that these techniques are explored using the original information preserved from the creation stage rather than information inferred by some means.

The availability of the information “behind the diagram” allows alternative presentations to be generated such as static or interactive presentations. Static presentations include graphical and textual presentations. One of the original benefits of the GraSSML approach is that it allows the creation of query systems (graphical or textual interactive presentations) which allow interactive exploration of the information. Having access to the data model and its ontology allows implicit information to be made explicit thus one of the main advantages of diagrammatic representation “recognition” becomes possible.

4.6.1 Graphical representation

A graphical representation of the diagram is generated taking into account the information provided by the data model, the notational conventions rules (transformation from Data Model to ZineML) and the user and/or device requirements (transformation from ZineML to Representation).

SVG is used as the graphical output renderer. Good quality SVG is ensured by the existence of the structure level populated by the ZineML language.

ZineML captures the structure behind the diagram and the notational conventions governing its graphical representation. That information is organised and represented in SVG using the accessibility features (MCCATHIENEVILE and KOIVUNEN, 2000) SVG offers (appropriate use of grouping, title, desc, id, class, etc.).

For complex diagrams, interactive exploration is possible using SVG's facilities (e.g. Chapter 5 scripting is used) and the available information concerning the structure and semantics of the diagram (e.g. hiding and exposing details).

4.6.2 Textual representation

Verbalisation Model templates that generate a textual representation of the structure and the semantics of the diagram have been implemented. The verbalisation model consists of a template, defined by the domain expert, gathering essential information at the appropriate level of abstraction, organizing and presenting it in a way that is easy to understand textually. The information needed for such a verbalisation model is collected from the data model at the semantic level and from the ZineML representation of the diagram at the structure level.

Pre-selected ornamental keywords such as connection words (e.g. "labelled", "named") are used to make the resulting textual description more human readable, natural and easy to understand.

Construction of the templates is a manual process. The necessary information from the data model is extracted and the template is then applied to the retrieved information in order to generate the verbalized representation.

Techniques which have been successfully evaluated and accepted as efficient by research teams have been used to design such verbalisation models. The aim was to demonstrate that such representations could be generated using GraSSML.

This follows guidelines described in (NCAM, 2008, CORNELIS and KRIKHAAR, 2001, WEB, 2002, NBA, 2000, OU, 2009, NCAM, 2006).

Verbalisation models can also be used by the author as a tool to check the efficiency of the graphical presentation of his diagram. Indeed, the author can check that the textual representation obtained by applying the appropriate verbalisation model templates cover all the aspects he wants the user to perceive when accessing his diagram.

4.6.3 Query systems

With the structure and semantics of the diagram available, it is possible to express queries concerning specific parts of a diagram in novel ways: this provides a nascent “smart diagrams” capability. Because of the reversibility properties of the GraSSML architecture, information at any layer can in principle be queried from a representation at another layer.

Having access to the data model and its ontology, it is possible to derive additional information that the data model holds but does not express explicitly. The ontology holds some inference rules which are used by a reasoner to make inferences on the data model. This mechanism allows “recognition” by making implicit information explicit (BROWN et al., 2004).

The query system is a means for users to access this information in a way they require it. As it is not possible to predict all the different queries, it is essential to provide an endpoint which allows the user to query the information behind the diagram by formulating their own queries. The author can provide some queries, but can easily add more in response to demand and/or the user can add their own. This point is discussed in the final chapter on future work (Chapter 8).

Although, it would be possible to implement such a system allowing users to formulate their own queries, due to time constraints, it was decided to implement a defined standard set of queries which are believed to get used a lot in the respective domains of the selected formal diagrams. The idea is to demonstrate the feasibility and applicability of the approach.

Definition of the queries that would be used by the textual query system and/or the graphical query system should be defined with the aid of the domain expert who is in the best position to identify what queries users are likely to expect and which queries would allow the user to explore the diagram efficiently. It is the role of the

domain expert to determine meaningful and useful queries that would allow the user of the system to acquire comprehensive information about the diagram.

A. Graphical smart diagrams

The graphical query system allows the user to interact with the diagram by selecting some part of it and choosing the query to apply from a menu. The result is shown graphically by highlighting, hiding or changing font size or colour.

B. Textual query system

A textual query system allows the user to query the graphical information through a graphical interface. The results are shown textually.

4.6.4 Other representations

Other alternatives representations are possible at this level, such as Braille, sound and non speech, Audio/Tactile. Other researchers (BULATOV and GARDNER, 2004, HUYNH et al., 2005, KURZE et al., 1995, MIKOVEC and SLAVIK, 1999, ROTARD and ERTL, 2004) have studied which options are most appropriate in particular situations, supported by evidence from evaluations. These projects have developed and successfully demonstrated the effectiveness of certain techniques in presenting graphical information non-visually. One challenge for GraSSML is to generate such representations by transformation from one of its information representation level (structure “ZineML” or semantics (Data Model)). One approach would be to have “access” to the format used in these methods and from there generate their proposed representations using the GraSSML approach.

4.7 Transformations

An important motivation behind the GraSSML layered conceptual model (Figure 12) is to reduce a task to a sequence of transformations between inputs and outputs expressed in different “Little Languages” (BENTLEY, 1986). The worthiness, flexibility and efficacy of doing multiple transformations have been explored at various levels of the GraSSML conceptual architecture.

Many transformations could be considered between the different levels. The whole system is intended to be reversible, meaning that it is possible from one level to reach information available at another higher or lower level. The two main transformations are presented in the following sections.

4.7.1 Data model to ZineML

This transformation concerns the definition of the notational conventions (Data Model to ZineML) which are analogies between the concepts defined in the ontology and the notations used to represent them when the diagram is to be represented graphically. These sets of rules govern the generation of ZineML from the data model expressed at the semantic level. Information from the data model is extracted and the notational conventions which are defined by the domain expert are applied to the extracted information to generate ZineML.

4.7.2 ZineML to Representation

The information gathered during the authoring process is presented to the user who wants to consult the diagram in order to extract and explore the information contained therein. To present the information in an appropriate format, user and/or device requirements should be taken into account. These requirements allow the adaptation of the information into the appropriate presentation medium (e.g. graphical presentation, textual presentation, etc) and corresponding representation. If the user is visually impaired, a textual presentation might be more appropriate. The textual query system could also be used to browse the information and get a better understanding of the graphical information or of a large textual presentation. Many possibilities are considered; depending on the user/device requirements one or more of these presentations could be generated and presented to the user.

The Composite Capabilities/Preferences Profile framework (CC/PP) (W3C, 2004b) could be used at this level. It allows device capabilities and user preferences to be expressed using RDF and can be used to guide the adaptation of content presented to that device. CC/PP could improve the way different implementation versions of the same information are distributed to an end user.

For the graphical presentation, a transformation is applied to the ZineML. This generates the graphical representation of the diagram taking into account the notational conventions defined within ZineML passed by the semantic level and the user and/or device requirements.

When a textual presentation is required, the appropriate verbalisation model template is applied to generate it. The appropriate template is applied to the structure level if the textual description of the structure of the diagram is required and to the semantic level if the textual description of the semantics of the diagram is required.

4.8 GraSSML in practice

The methodology to apply and author GraSSML involves seven stages and a number of actors. Figure 14 illustrates the concrete GraSSML system architecture developed. The seven numbered areas of the figure illustrate the seven stages involved in the application of the GraSSML approach. These are described in section (section 4.8.2). The following sections summarize this by outlining the actors involved in the process and the process itself. Three different actors may be identified:

- **A domain expert**, to share his knowledge of the domain to define the ontology of the domain, the rules for notational conventions, the presentation rules to reflect user or/and device requirements, the queries definition, and the verbalisation model templates allowing the textual description of the structure and the semantics of the diagram.
- **The author**, the person or system who creates a diagram. The author creates the diagram by defining the data model of the information to be portrayed. The data model is linked to the ontology that formulates the conceptual schema for this kind of diagram in the domain selected.
- **The user**, the person who wants to access, explore, query, or browse the diagram content.

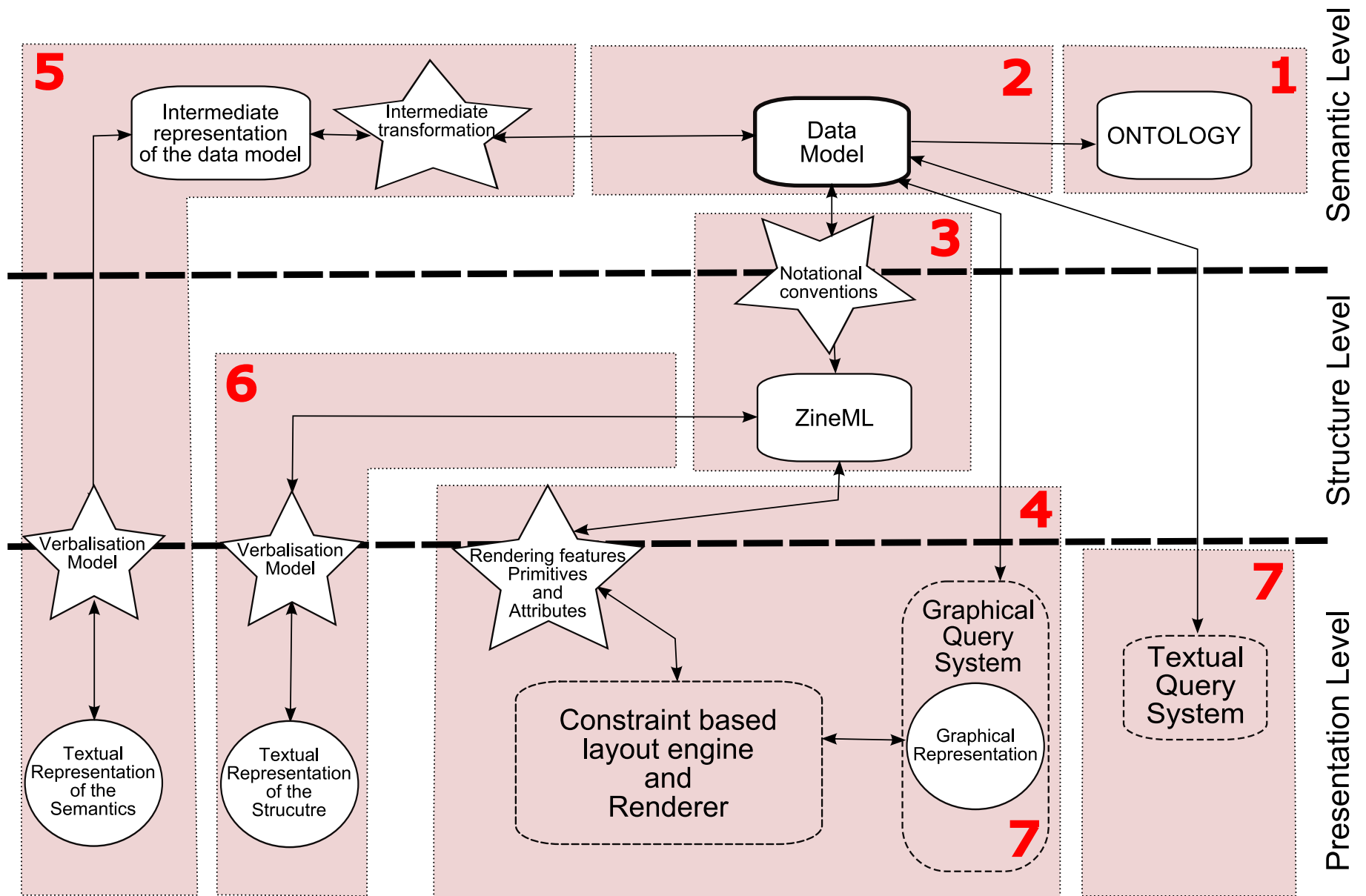


Figure 14: GraSSML System architecture

4.8.1 Initiation

Initiation is the process that collects (generates) the information required to handle a new type of diagram. There are four aspects, each carried out by the domain expert:

1. Formulation of the ontology
2. Notational conventions
3. Verbalisation model Templates
4. Basic Query definitions

The approach relies heavily on the presence of a domain expert to generate the original content in an appropriate form. Such domain knowledge is essential in the initial annotation representation of the diagram. This process provides a way to share that knowledge across a wide user base.

As discussed in section 4.4.1, the creation of the ontology describing the background domain could be seen as complex and costly in term of time and effort. But the ontology could either be created, automatically generated from some source or be a pre-existing ontology. It is worth noting that the cost of creating the ontology is not insurmountable as illustrated in Chapter 6 where an ontology has been generated from an XBRL taxonomy and then extended to meet the requirements of a new domain (financial reporting). This ontology was created by a third party who had never previously created an ontology.

With this type of knowledge in place, the GraSSML approach is sound.

The domain expert needs to indicate how much information needs to be provided by the author to produce a reliable data model and thus diagram.

The domain expert is also the person in the best position to choose the basic queries needed for the type of diagram. It is his role to know what is a relevant or not as a query. Hence basic queries are defined and put in place. Such tasks involve manual intervention from the domain expert. But it could be possible to generate these queries automatically. This point is further discussed in Chapter 8.

The time needed at the initiation stage may be seen as requiring too much effort. But it could be argued that this time is a small price for the domain expert to pay in order to allow potential authors to create diagrams that would be more perceivable, operable and understandable.

It is not expected that the author engage in the work of creating ontologies and graphical rules, though the framework would and should not preclude them from doing so. Of course a question remains: will the domain expert do a good job? This would lead us to questions outside the scope of the thesis.

4.8.2 Applying GraSSML

The seven numbered area of the Figure 14 illustrate the seven stages involved in the application of the GraSSML approach. These are described as follow:

- **Stage 1- Ontology:** Using information gathered at the initiation stage, an ontology describing the background domain is either created or generated from some source. A pre-existing ontology could also be used.
- **Stage 2- Data model:** The diagram is defined via a data model conforming to the ontology. The conformance to the ontology is important for the approach to successfully process the data model provided.
- **Stage 3- Notational conventions:** A set of notational conventions, which are analogies between the concepts defined in the ontology and the notations used to represent them when the diagram is to be represented graphically, are defined. These notational conventions govern the generation of ZineML from the data model.

The following stages emphasize the advantages of having access to the information “behind the diagram” as it allows alternative presentations to be generated and enquiry and reasoning to be achieved:

- **Stage 4- Graphical representation:** A graphical representation of the diagram is generated taking into account the information provided by the data model, the notational conventions and some user and/or device requirements.
- **Stage 5- Verbalisation model (Semantics) and Stage 6- Verbalisation model (Structure):** Verbalisation Model templates are created to generate a textual representation of the structure and/or the semantics of the diagram. The verbalisation model consists of a template gathering essential information at the appropriate level of abstraction, organizing and presenting it in a way that is easy to understand textually.
- **Stage 7- Query systems:** One of the original benefits of the GraSSML approach is that it allows the creation of query systems (graphical or textual

query systems) making it possible for users (and machines) to enquire and reason over the information. The ontology has an important role in the approach as having access to the data model and its ontology allows implicit information to be made explicit. This stage demonstrates the ability to enquire and reason about the information on which the diagram is based.

All of the stages are described in more detail in subsequent sections.

4.8.3 The authoring process

The GraSSML approach involves significant changes in the authoring process and the awareness of graphical information for the author and the user. Its different stages may be seen as expensive in time and effort.

The GraSSML authoring process starts by creating the data model of the diagram. The author needs to select the ontology for the domain of interest. This phase is semi-automatic. The author also needs to specify the type of the diagram at the structure level and to select the notational conventions from amongst those already defined by the domain expert.

The ultimate aim is to define an authoring process that is easy to learn and understand. The author should feel that he is in control and that he is able to create the intended diagram the way he wants it to be.

The author should also understand the aims of the system: contributing to the representation of diagrams in a way that could improve the ability to enquire and reason about the information on which the diagram is based. Graphical presentation of the diagram is not the sole intent of the GraSSML system, but only one of the possible media in which to present the information.

The focus of the prototype is to show the power of the transformational approach, and for this reason a more general approach has been taken to authoring, not least because the authoring process might itself involve transformation or extraction from some other sources.

As mentioned in section 4.4.2, the data model can originate from different sources, opening doors to new opportunities for the creation of diagrams from different sources and by different people with various abilities. A number of existing tools which people know about and feel comfortable with could be considered for the authoring process. Options involving WYSIWYG tools, that people feel comfortable with, and textual creation of RDF graph opens new horizons for the authoring of

diagrams aiming at preserving the information behind them and allowing enquiry and reasoning over that information. This idea is illustrated in an example later in the thesis (Chapter 5 section 5.3.2H).

4.9 Conclusion

This chapter has presented the main ideas behind the GraSSML approach aimed at fulfilling the defined requirements in Chapter 3 (section 3.4) and supporting the hypothesis. The underlying three-level conceptual architecture of GraSSML have been described offering details of its levels (semantics, structure and presentation levels) and the transformations between these levels.

Table 6 shows how GraSSML addresses the requirements and provides a comparison view of GraSSML with existing approaches reviewed in (Chapter 2 and Chapter 3). The seven stages previously described are referred to, to specify which feature of the architecture relate to addressing which requirement.

The overheads on diagram authoring required by the adoption of the proposed approach have been discussed (see section 4.8). It is strongly believed that the benefit of providing access to the information behind the diagram will outweigh the effort involved in using the GraSSML approach in its current state of development. Many developments could be carried out in order to make the process of using GraSSML a pleasant experience, making authors confident in its viability. This will be discussed in Chapter 8.

	Requirements	TeDUB	SIGHT	iGraph Lite	OntoDiagram	SemViZ	GraSSML
Information	<ul style="list-style-type: none"> • Capture, encode and preserve the original information behind the diagram at the creation stage. 				✓	✓	✓ 1, 2
	<ul style="list-style-type: none"> • Present the diagram from it 				✓	✓	✓ 3, 4, 5, 6,7
Graphical format	<ul style="list-style-type: none"> • Use an appropriate format to store the information of the diagram. 	Vector Image	Vector image	Vector image	Vector image	Vector image	Vector Graphics
Presentation	<ul style="list-style-type: none"> • Provide a mechanism to provide access to implicit information present in the diagram information (Recognition), by making it explicit 						✓ 1, 2, 7
	<ul style="list-style-type: none"> • Provide mechanisms of exploration/ navigation of the information 	✓		✓	✓		✓ 5, 6, 7
	<ul style="list-style-type: none"> • Provide an overview of the information (Overviews). 	✓	✓	✓	✓		✓ 5, 6
	<ul style="list-style-type: none"> • Provide a mechanism to process and search the information presented (Search) 			✓	✓		✓ 4, 5, 6, 7
	<ul style="list-style-type: none"> • Provide the user with the power to decide which information he wants to obtain depending on the nature of the task he wants to achieve (Task dependence) 			✓	✓		✓ 4, 5, 6, 7

Table 6: Overview of existing approaches and GraSSML against the defined requirements

Three different evaluations are considered, aiming to assess the GraSSML approach from three different perspectives and gather evidences to support the hypothesis:

- **Technical perspective:** Aims at demonstrating the feasibility and applicability of the GraSSML approach. The development and implementation of a full functional proof-of-concept prototype

demonstrates GraSSML for two different types of diagrams selected from two different application domains presented as use cases: the hierarchical class of diagram, more specifically Organisation charts and the process class of diagram, more specifically UML activity diagrams

- **Author's perspective:** Aims at demonstrating the viability and applicability of the proposed approach. A third party has extended the GraSSML system (GraSSML FCP) for a different class of diagrams. He has developed the implementation of a different class of diagram "Charts" (financial charts).
- **User's perspective:** A user evaluation with three sighted users and two blind users has been carried out with the aim of providing an insight on whether GraSSML supports the perceivability, operability and understandability principles of the WCAG 2.0 (see Chapter 3 section 3.1.2) for diagrams and as a result to demonstrate the hypothesis.

These evaluations are presented in more details in the following chapters.

Chapter 5

Evaluation: A Technical Perspective

This chapter describes the system architecture of the proof-of-concept tool called the GraSSML prototype developed to demonstrate the feasibility and applicability of the GraSSML approach described in the previous chapter.

The implementation is seen as a means to an end. The aim was not to produce a production quality piece of software but more a fully functional proof-of-concept prototype reflecting the conceptual architecture of GraSSML. So, the different technologies used have been selected accordingly in an attempt to build a system “working prototype” where experiments can be done, within the timescale of the thesis.

The development and implementation of the full functional proof-of-concept prototype demonstrating GraSSML for two different types of diagrams, selected from two different application domains: Process diagrams (UML Activity Diagram) and Hierarchical diagrams (Organizational Charts), is then presented as two different use cases.

5.1 GraSSML Prototype Architecture

The GraSSML prototype architecture is illustrated in Figure 15. This is a reification of the system architecture presented in Figure 14. The seven stages involved in the application of the GraSSML approach have been described in Chapter 4 section 4.8.2.

GraSSML is an application of semantic web techniques to provide machine-readable metadata for diagrams. Semantic web technologies have been used to express the semantic intent behind a diagram required by the GraSSML approach. As the research progressed, semantic web technologies ((Resource Description Framework (RDF) / RDF Schema (RDFS) / Web Ontology Language (OWL)) became more prominent technologies and tool support improved. Therefore a general approach starting from a data model (RDF graph) and associated ontologies was developed.

All the parts of the GraSSML conceptual architecture have been implemented in the GraSSML prototype. Extensible Markup Language (XML), Extensible Stylesheet Language Transformations (XSLT) and Scalable Vector Graphics (SVG) have all been used to populate the GraSSML architecture.

The open source development platform Eclipse has been used as a Java integrated development environment (IDE). The key transformations have been implemented using an XSLT transformation engine (Saxon).

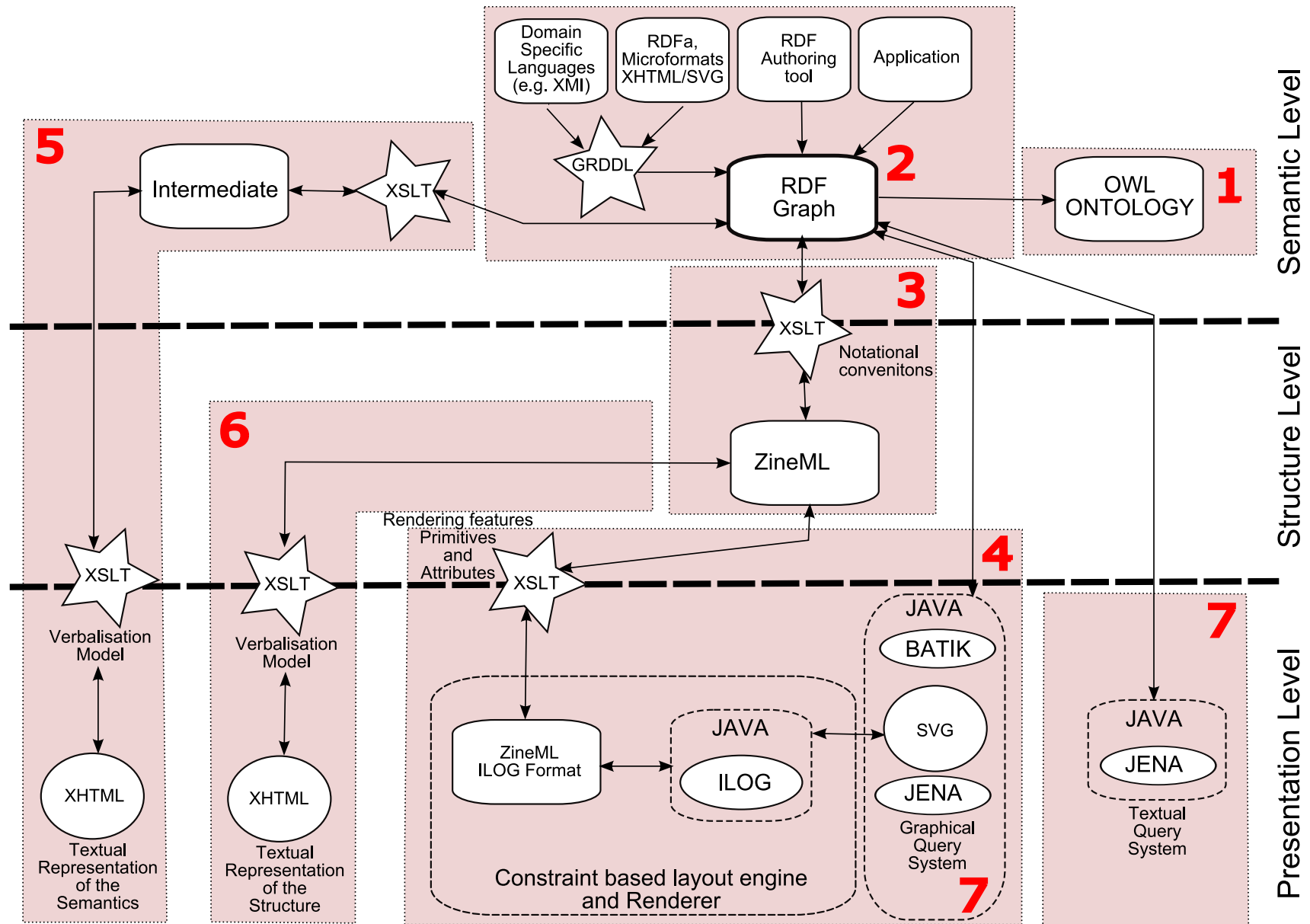


Figure 15: GraSSML Prototype Architecture

5.1.1 Stage 1: Ontology

Protégé (PROTEGE, 2009), the extensible platform-independent environment for creating and editing ontologies and knowledge bases has been used to author this ontology.

5.1.2 Stage 2: Data model

The RDF data model lies at the heart of the proposed GraSSML approach. Its conformance to the OWL ontology is important for the GraSSML approach to successfully process the RDF graph and make implicit information explicit. So the relationship between the RDF graph and the OWL ontology it conforms to, is key to the ability to enquire and reason about the information on which the diagram is based.

The RDF data model may be produced in several different ways (Figure 16).

The data model might be:

- created through an authoring tool (PROTEGE, 2009)
- extracted from SVG, XHTML or HTML5 documents using technology such as GRDDL (W3C, 2007), Microformats (MICROFORMATS, 2005) and RDFa (W3C, 2008a)
- scraped from Web pages using technology such as Piggy Bank (HUYNH et al., 2005)
- extracted from domain specific markup languages (e.g. XMI (OMG, 2007) for UML diagrams (see section 5.3), XBRL (XBRL, 2007) for Business Reporting).

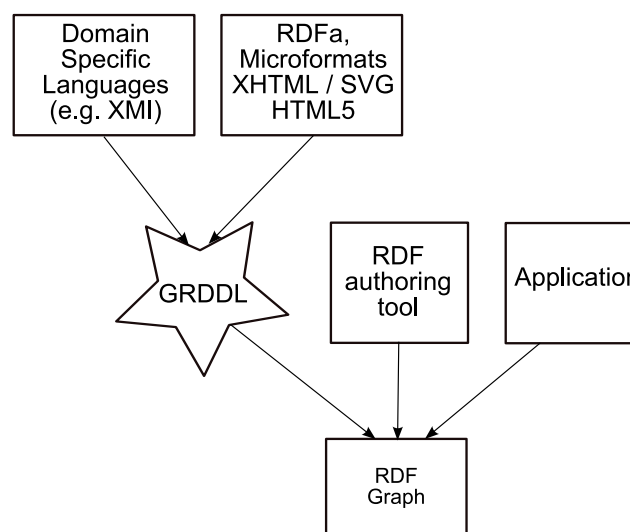


Figure 16: Possible sources of information of the RDF data model

The Protégé OWL editor has been used to create individuals (RDF graphs) validated against the appropriate ontology. Protégé offers two different options to create the data model, either using only forms (Figure 17) or alternatively using the graph widget to create the individuals (Figure 18) and then forms to enter the details concerning these individuals.

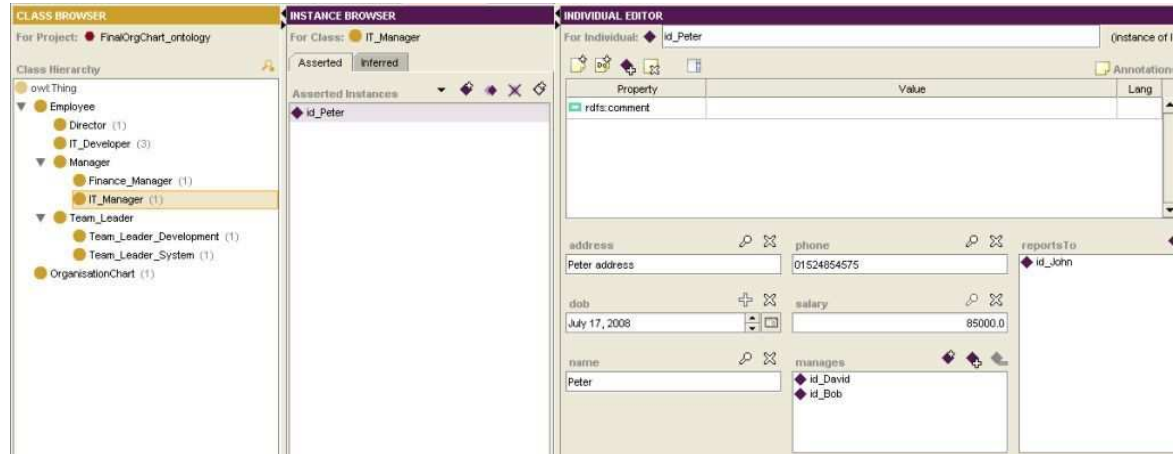


Figure 17: Creation of the RDF Graph using forms in Protégé

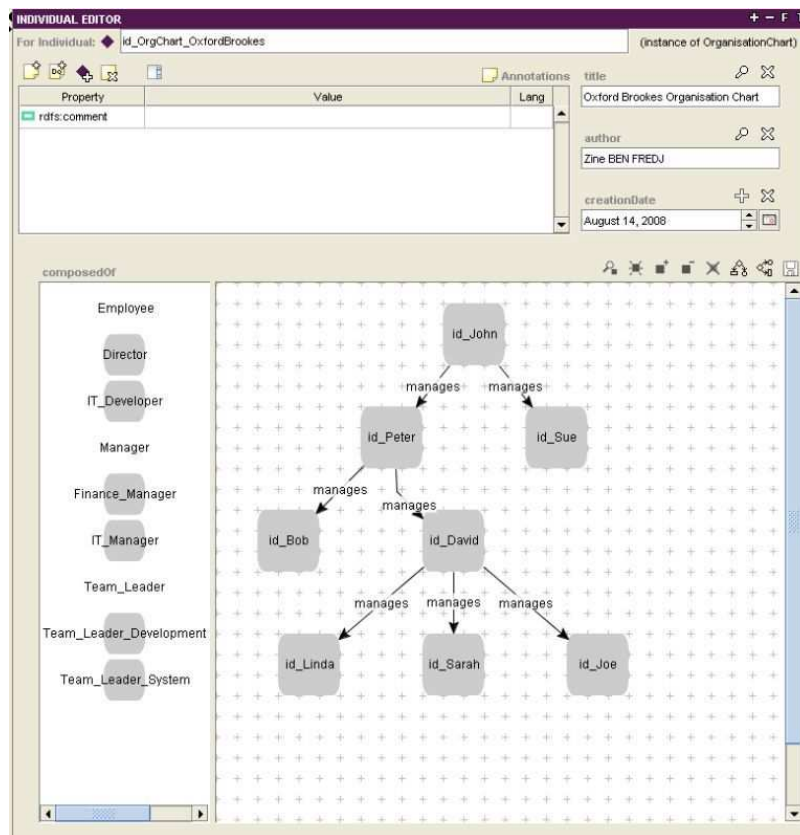


Figure 18: Creation of the RDF Graph using Graph Widget in Protégé

5.1.3 Stage 3: Notational conventions

A. XSLT Transformation

The rules which relate concepts expressed in the data model with the notations needed to represent them in the diagram to be represented graphically are expressed by an XSLT (Extensible Stylesheet Language Transformation) transformation.

The XSLT transformation generates ZineML from the information retrieved from the RDF graph. The XSLT transformation embodies the notational conventions which express analogies between the concepts defined in the ontology and the notation used to represent them when the diagram is to be represented graphically.

B. ZineML

This section provides a description of the syntax of ZineML.

❖ ZineML Syntax

ZineML is an application of XML with ZineML as the root element. It has an attribute “type” that defines the class of diagrams being represented.

Some of the features of the language are:

- By default, elements are drawn from left to right and connected at the default positions.
- Each element has a well defined bounding box which can be named for referencing.
- Objects have a default size but this is expanded to ensure text or other shapes inside it remain inside the shape of the object.
- Each object has a set of defined connection points named by default n | s | e | w | nw | ne | sw | se | c | left | right | top | bottom.
- An XML Schema for a specific class of ZineML diagrams ensures the markup is constrained to that specific class’s requirements.
- Coordinates in ZineML have the X-direction from left to right and the Y-direction from top to bottom. This is the same as SVG and PDF.
- The default unit is the point (72 points to an inch). Units can also be defined in px, mm, cm and inches. The units to be used are defined as an attribute “units” on the root element.
- Units can be defined as absolute or relative to the current position.


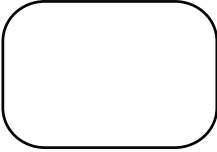

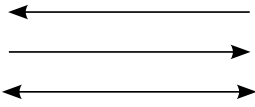
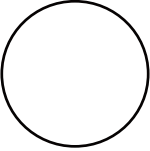
❖ ZineML Basic Shapes

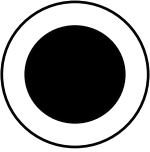
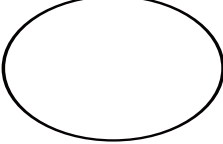
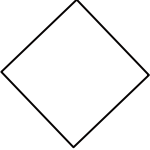

The set of shapes provided by ZineML is open-ended and user-defined. A set of basic shapes, presented in the following table, is provided initially. Most elements can have textual content. For example `<box>some text</box>`.

Each shape has a set of attributes. Some attributes apply only to specific shapes and others to all shapes. The attributes presented in the table are specific to that shape whereas the following sets of attributes apply to all shapes:

- id: unique id
- visibility: true | false
- color: fill color of the shape
- strokeColor: stroke color of the shape
- x, y, width, height
- halign, valign
- att-point: n | ne | e | se | s | sw | w | nw | c | left | right | top | bottom

The basic set of shapes is illustrated below.

ZineML element	Shape (ILOG Symbol)	Specific attributes
Box		labelColor labelVisibility
Rbox		labelColor labelVisibility
Line		x1, y1 x2, y2
arrow		from, to headType= start end both none
circle		R

Dcircle		colorIn colorOut strokeColorIn strokeColorOut
ellipse		rx, ry labelColor labelVisibility
diamond		
thickBar		Direction

5.1.4 Stage 4: Graphical Representation

The commercial ILOG JViews Diagrammer tools (Figure 19) and SDK in Java have been used to generate the graphical representation of diagrams in SVG from the ZineML description at the structure level.

At this level due to the nature of the topology of process diagrams and different possible layouts for hierarchical diagrams, it was quickly realized that a constraint based algorithm was needed. At this point it was interesting to explore existing tools and find out how they would integrate with the GraSSML approach. ILOG JViews Diagrammer was found to be an excellent choice for a decent constraint based layout algorithm. It has demonstrated its ability internationally and commercially. As ILOG were interested in the project, it was possible to obtain a copy of this powerful tool for this research project.

While generating diagrams using JViews Diagrammer different aspects have to be understood. ILOG JViews Diagrammer provides several design tools (Figure 19) to automate the production of diagrams. This research makes use of two of them: the Symbol Editor and the Designer.

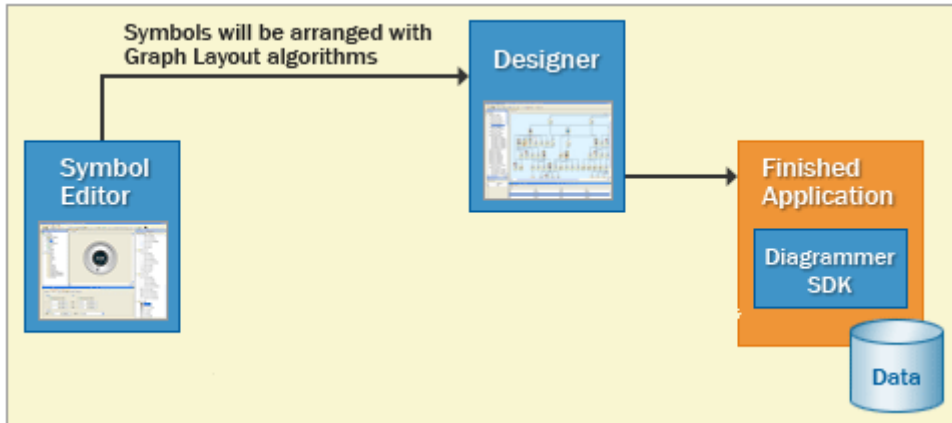


Figure 19: Based on ILOG JViews Diagrammer Tool Chain

The JViews Diagrammer architecture (Figure 20) provides a clear separation between the data and its presentation. The JViews Diagrammer data model is part of the Styling and Data Mapping (SDM) engine, and forms the connection between the data source and its views on display. SDM relies on the Framework SDK and the Graph Layout SDK which are also part of JViews Diagrammer. The SDM engine is one of the most important pieces of JViews Diagrammer as it controls the data-to-graphics mapping.

The Framework SDK provides a comprehensive structured 2D graphics API that includes graphic objects, interactors, views, transformations, graphs, sub graphs, etc. The Graph Layout SDK provides a set of graph layout algorithms.

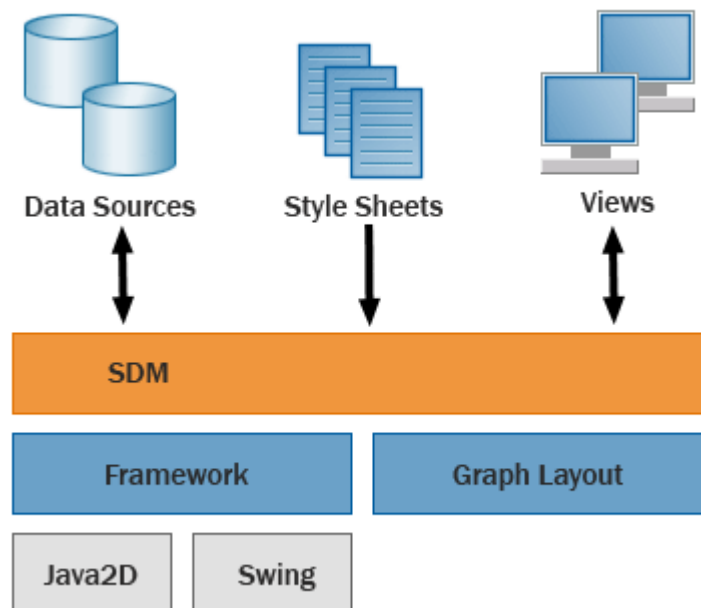


Figure 20: Architecture of JViews Diagrammer

Symbols are the starting point of a diagram. A symbol is a collection of graphic objects, parameters and conditions used to represent a concept. A CSS file is used to store the information needed to represent a symbol (type and position of shapes, text, images, etc.). The CSS file contains a description of the JavaBeans to use, their settings and logic needed at the instantiation of the symbol. Symbols data is stored and organized inside palette files structures which are stored in Java archive files (.jar). A tool called the Symbol Editor (Figure 21) is used to create the symbols and organize them within palettes. Different parameters could be assigned to the symbol; these would be linked later on to specific values from a data model containing the information to be represented.

The diagram generated is created from a data source which is expressed in our case in the required XML format called “diagram format” defined by ILOG. This data source is in fact an ILOG valid representation of the information represented in ZineML (notational conventions). It is generated from ZineML using an XSLT transformation.

In JViews Diagrammer, the Designer creates a display by binding the data source elements to the symbols created with the Symbol Editor and saved in a palette. Using the Designer it is possible to specify that a given type of data instance from the data source (e.g. Diamond) be represented by a given symbol (e.g. Lozenge), and that a particular value of a field defined in the data source to be bound to some symbol parameters to define the aspects of the symbol.

The Designer defines the link between the symbols and the elements defined in the data source; this mapping is described in a CSS format and allows the Designer to create a display of the diagram. The output is a project file combining the CSS part and the XML part (data source) and can be loaded into an application at run time.

It is also possible using the SDK of ILOG JViews Diagrammer to customize the data model and refine the graphic representation of the data. Further Java classes provided within the JViews Diagrammer Advanced Framework allows the generation of the output SVG graphical representation of the diagram. This is what has been used in order to generate a tailored SVG representation of the diagram. ILOG generates SVG as an output format. Some modifications were needed to enrich and reorganize the SVG output to provide a link to the semantic information.

ILOG JViews Diagrammer and ZineML have produced a flexible system where significant changes in rendering can be divorced from the logical description of the

diagram. ILOG proved to be an excellent choice in putting in place generating the graphical representation of the information taking into account the information collected at the structure level through ZineML. When using the GraSSML approach having one set of symbols rather than another set of symbol is not important, what matters is the definition of a set of symbols and connectors. The tools provided with ILOG (symbol editor and the designer) are perfect candidates to implement such aspects of the approach. ILOG JViews Diagrammer provides most of what the GraSSML prototype requires although some features required by ZineML in the generation of SVG could not be generated correctly but this did not affect the overall correctness of the GraSSML approach. It was not possible within the scope of the project to correct these.

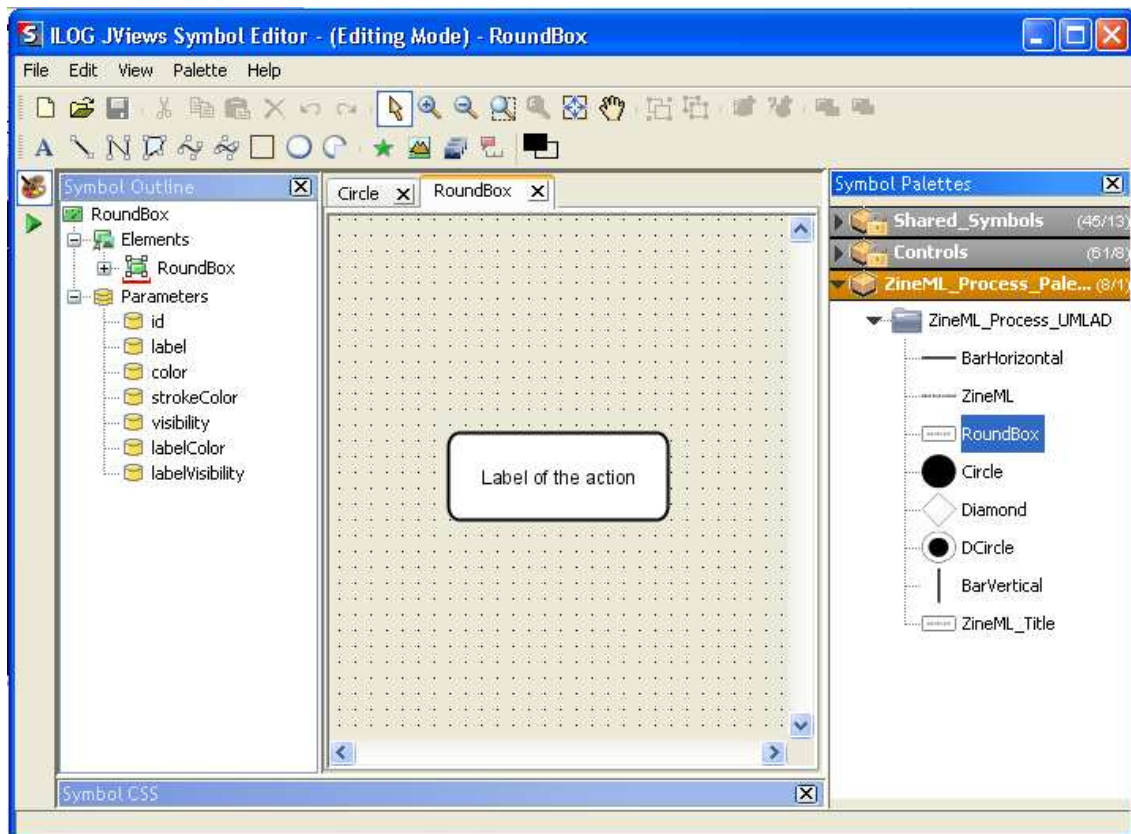


Figure 21: Symbol Editor for UML Activity Diagram Example

Of course, it would be possible to use any other system to replace ILOG (e.g. VGJ, Dia, etc.). The main points involve the creation of the symbols, naming them, identifying their anchor points and using a constraint based layout engine and renderer to deal with the generation of the appropriate graphical representation.

5.1.5 Stage 5: Verbalisation model (Semantics)

The verbalisation model templates defined in XSLT, generate textual representations of the diagram semantics. In the GraSSML prototype, this is done in two stages using an XML intermediate format.

Information is extracted from the RDF graph and transformed into the syntax of an XML intermediate document. A second XSLT Transformation is applied to the intermediate document using the information extracted from the RDF graph and the research into appropriate presentation of diagrams non-visually (NCAM, 2008, CORNELIS and KRIKHAAR, 2001, WEB, 2002, NBA, 2000, OU, 2009, NCAM, 2006).

This research suggests information should be provided in layers of increasing detail. Initially the user is given an accessible XHTML overview targeted at a screen reader (title, type of diagram, complexity, intent, etc.) ideally defined by a domain expert using appropriate techniques and enriched with appropriate ornamentations (e.g. “labelled”, “followed by”, “if”, etc.).

For information aimed at screen readers, special attention should be paid to with links, headings, paragraphs and page elements. The screen reader Jaws (FREEDOMSCIENTIFIC., 2009) has been used to evaluate how it would read the description generated. If needed, the presentation of the end result can also be styled using CSS.

5.1.6 Stage 6: Verbalisation model (Structure)

Some diagrams need to be described in terms of their structure (e.g. modelling languages such as UML).

The implemented verbalisation model templates generate the diagram structure in XHTML based on the syntax of the corresponding ZineML representation.

5.1.7 Stage 7: Query systems

The Jena framework (JENA, 2009) for building Semantic Web applications (written in Java) and the SPARQL query language for RDF (W3C, 2008b) have been used to retrieve information from the RDF graph and store it in an XML document.

The graphical query system and the textual query system have been implemented using Jena and SPARQL.

For both query systems a SPARQL end point would have been the best option for authors and users to define their own queries. Such an end point could allow the automatic generation of queries from some formal specifications and so being more author-friendly and user-friendly. But unfortunately due to the timescale of the thesis it was not considered as essential to implement.

On the other hand to prove the feasibility of such a query system a “hand-coded” version with a predefined sample of possible queries has been developed. The hand-coding involved includes the SPARQL queries themselves as well as the Java or Java/JavaScript interfaces to collect the information required by the selected query (e.g. “who manages directly...?” query requires the “name of the person concerned by the query” e.g. “Linda”) and the Java or Java/JavaScript interface displaying the result of the query.

The “Textual Query System”, queries the RDF graph of the diagram using a predefined user interface implemented in Java. The results are presented textually within the same interface.

The Graphical query system allows the SVG diagram to be queried directly by selecting an object on the diagram and selecting the query to apply to it from a predefined graphical menu expressed in SVG. The results are presented graphically (e.g. using highlighting or hiding). This graphical query system has been written in Java and JavaScript. It currently runs as a Java component using Batik SVG in a Java environment which offers JavaScript support. An important aspect at this level is the Java which is called from JavaScript to provide results of queries selected on the SVG document. The queries are executed by the Java program and returned to JavaScript which displays them. The results are then used to modify the SVG document. The query could also be run within a Web browser using techniques described in (COOPER et al., 2005).

5.1.8 Graphical User Interface

The graphical User interface has been implemented with accessibility in mind. Effort has been put in place to ensure keyboard and screen reader accessibility of the different elements of the interface.

Java Access Bridge (JAVA, 2009) for windows has been used to ensure that Assistive technologies such as screen readers are able to interact with the Java based graphical interface objects under windows. Mnemonics as well as keyboard short cuts

have been used to ensure accessible navigation around the interface. The interface has been tested using Jaws to ensure the focus is not lost while navigating the interface.

5.2 Use Case: Hierarchical Diagram “Organisation Charts”

Hierarchical diagrams, as exemplified by “Organisation charts” will be explained in this section. A full description of one organisation chart will be followed by more specific examples from the same domain. Finally some examples from the same class of diagram but in a different domain will be presented.

5.2.1 Organisation Charts

Organisation charts are found in most corporate, government and academic institution web sites. They show the formal structure of an organisation in terms of the relationships between its departments and/or employees. They show the lines of command and responsibilities within the organisation.

Many variants of the basic chart exist often due to the structure and the size of the organisation chart. A good starting point for finding examples of different types of organisation charts is (<http://www.orgplus.com/>).

Organisation charts consist of a set of connected shapes (e.g. circle, rectangle, square, triangle, etc.) where each shape represents an employee or a department. The names of the department, employee or position are often the textual information provided with the shape but this varies depending on the company’s need. The lines connecting the shapes represent direct “line relationships” between superior/subordinate employees or departments. “Lateral relationships” between different employees or departments working at the same hierarchical level, can be shown. These charts can be represented as horizontal or vertical trees. They need to be frequently revised and updated if the workforce is regularly changed or if the structure of the institution changes. Sometimes to avoid frequent updating of an organisation chart, only position titles are used rather than the names of employees. They become very large and complex for large organisations and are then represented as a set of smaller organisation charts representing individual departments. Alternatively nested organisation charts can be represented. Different layout conventions can be found within the same chart to accommodate the size of the chart (Figure 22). The layout used is either level layout or tip-over layout or a combination of both.

- Level layout: the nodes are organised into levels and these levels are arranged either horizontally or vertically
- Tip-over layout: in order to balance the organisation chart, certain tip-over alignments are applied to a selected set of nodes (e.g. leaf nodes, or root nodes, etc.).

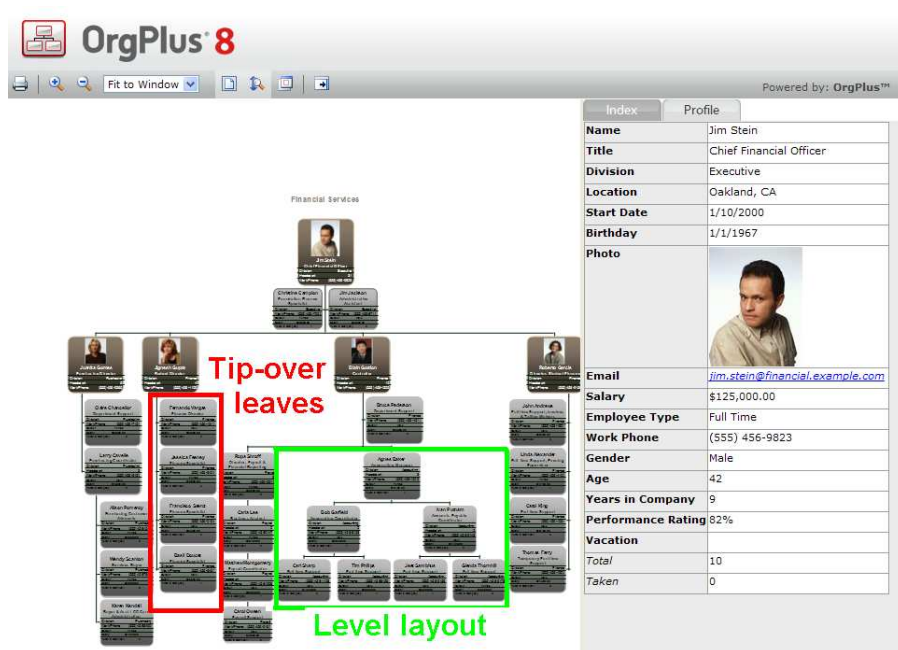


Figure 22: Organisation chart example

Different notational conventions can be found, use of different colour schemes, different information displayed (e.g. photos, title, division, name, etc.), different interactive facilities such as links or buttons to expand or hide a sub-tree or zoom on/click on an employee to get access to more details about the person.

Organisation charts are mostly represented on the current web as vector images. When designed with accessibility in mind other representations are provided to make them accessible to screen readers. A text alternative describing the vector image (e.g. Organisation chart of...) as well as a link to ("Organization chart Text only version"), a detailed description of the organisation chart, are the most common non-visual alternatives for this type of diagram found on the web. Either detailed natural language descriptions or bullet lists describing the different levels of the organisation chart can be found.

5.2.2 Main Example

Figure 24 presents the organisation chart of VISIONS (VISIONS, 2009), an incorporated non-profit social service and vision rehabilitation agency. This example will be used to describe the organisation chart use case.

A. Stage 1: Ontology

An ontology has been created. As the ontology is too big to be shown, Figure 23 shows the class hierarchy of the extended technology built using Protégé.



Figure 23: Class hierarchy for Visions Organisation Chart Ontology

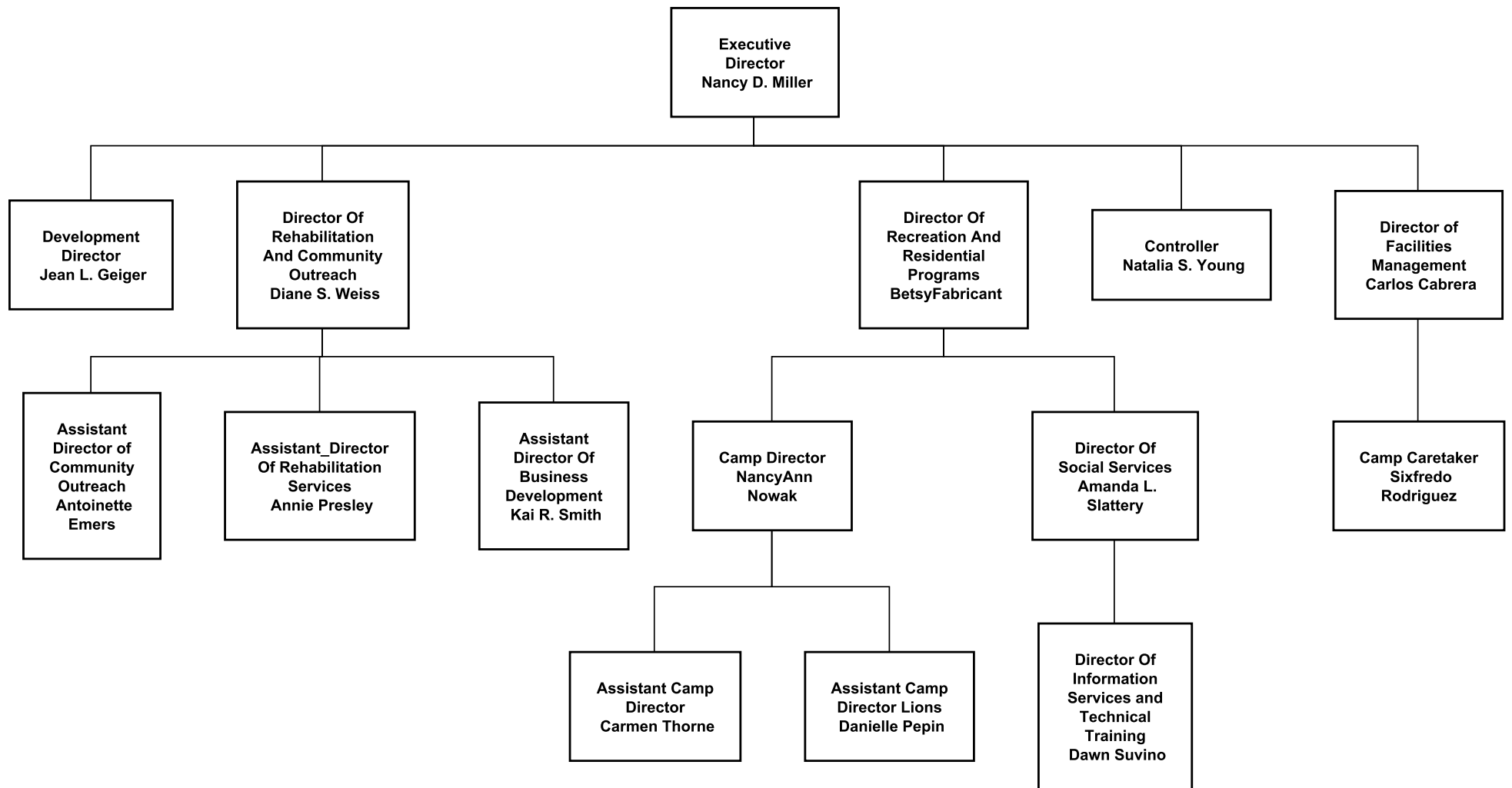


Figure 24: VISIONS Organisation Chart

B. Stage 2: Data model

The data model has been created using Protégé but could be accessed, previewed and modified using the graph widget (Figure 25) of the GraSSML prototype interface..

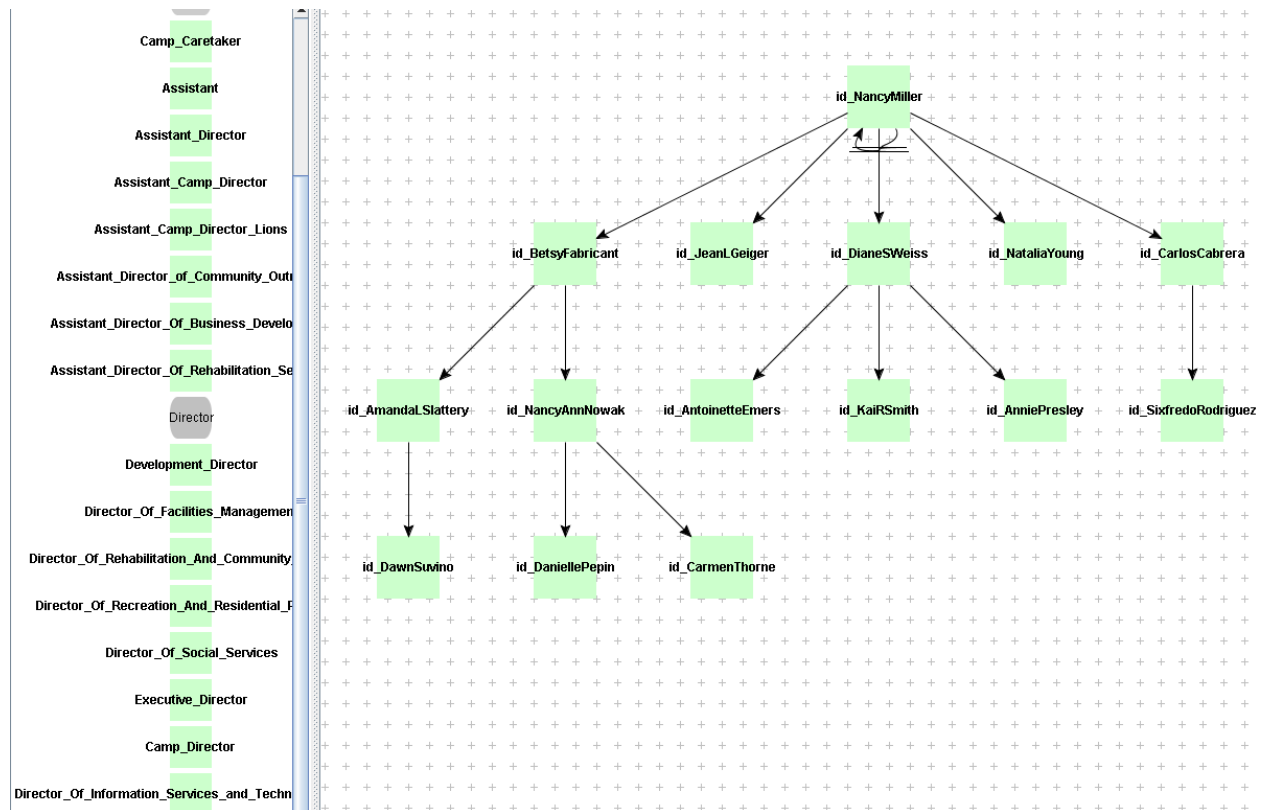


Figure 25: Creation of the Individuals Using Graph Widget

C. Stage 3: Notational conventions

The following conventions have been defined for the VISIONS example:

- Each employee is represented by a black box filled with white and a black centred label representing the position and name of the employee.
- Each “manages” and “reportTo” relationship is represented by orthogonal black lines connecting the employees of the relationships.
- The organisational chart is represented as a hierarchical diagram from top to bottom.

Information is extracted from the RDF graph and transformed into the syntax of an XML intermediate document. A second XSLT Transformation generating the ZineML is then applied to the intermediate document using the information extracted from the RDF graph and the appropriate notational conventions.

The ZineML for the Visions Organisation Chart example is as follows (Figure 26):

```

<?xml version="1.0" encoding="utf-8"?>
<ZineML type="Hierarchy" id="Zine BEN FREDJ" direction="Bottom" creationDate="2009-09-03 " author="Zine
BEN FREDJ" title="Visions Organisation Chart" visibility="false" layoutMode="LEVEL">
<box color="white" visibility="true" strokeColor="black" labelColor="black" labelVisibility="true"
id="id_NancyMiller">Executive Director Nancy D. Miller
<box color="white" visibility="true" strokeColor="black" labelColor="black" labelVisibility="true"
id="id_NataliaYoung">Controller Natalia S. Young</box>
<box color="white" visibility="true" strokeColor="black" labelColor="black" labelVisibility="true"
id="id_BetsyFabricant">Director Of Recreation And Residential Programs Betsy Fabricant
<box color="white" visibility="true" strokeColor="black" labelColor="black" labelVisibility="true"
id="id_NancyAnnNowak">Camp Director NancyAnn Nowak
<box color="white" visibility="true" strokeColor="black" labelColor="black" labelVisibility="true"
id="id_DaniellePepin">Assistant Camp Director Lions Danielle Pepin</box>
<box color="white" visibility="true" strokeColor="black" labelColor="black" labelVisibility="true"
id="id_CarmenThorne">Assistant Camp Director Carmen Thorne</box>
</box>
<box color="white" visibility="true" strokeColor="black" labelColor="black" labelVisibility="true"
id="id_AmandaLSlatery">Director Of Social Services Amanda L. Slatery
<box color="white" visibility="true" strokeColor="black" labelColor="black" labelVisibility="true"
id="id_DawnSuvino">Director Of Information Services and Technical Training Dawn
Suvino</box>
</box>
</box>
<box color="white" visibility="true" strokeColor="black" labelColor="black" labelVisibility="true"
id="id_JeanLGeiger">Development Director Jean L. Geiger</box>
<box color="white" visibility="true" strokeColor="black" labelColor="black" labelVisibility="true"
id="id_DianeSWeiss">Director Of Rehabilitation And Community Outreach Diane S. Weiss
<box color="white" visibility="true" strokeColor="black" labelColor="black" labelVisibility="true"
id="id_KaiRSmith">Assistant Director Of Business Development Kai R. Smith</box>
<box color="white" visibility="true" strokeColor="black" labelColor="black" labelVisibility="true"
id="id_AnniePresley">Assistant_Director Of Rehabilitation Services Annie Presley</box>
<box color="white" visibility="true" strokeColor="black" labelColor="black" labelVisibility="true"
id="id_AntoinetteEmers">Assistant Director of Community Outreach Antoinette Emers</box>
</box>
<box color="white" visibility="true" strokeColor="black" labelColor="black" labelVisibility="true"
id="id_CarlosCabrera">Director of Facilities Management Carlos Cabrera
<box color="white" visibility="true" strokeColor="black" labelColor="black" labelVisibility="true"
id="id_SixfredoRodriguez">Camp Caretaker Sixfredo Rodriguez</box>
</box>
</box>
</ZineML>

```

Figure 26: ZineML code for the Visions Organisation Chart Example

Changing the direction of the Organisation chart only requires changing the value of the attribute “direction” in the ZineML element. Figure 27 illustrates the result for direction=”Right”.

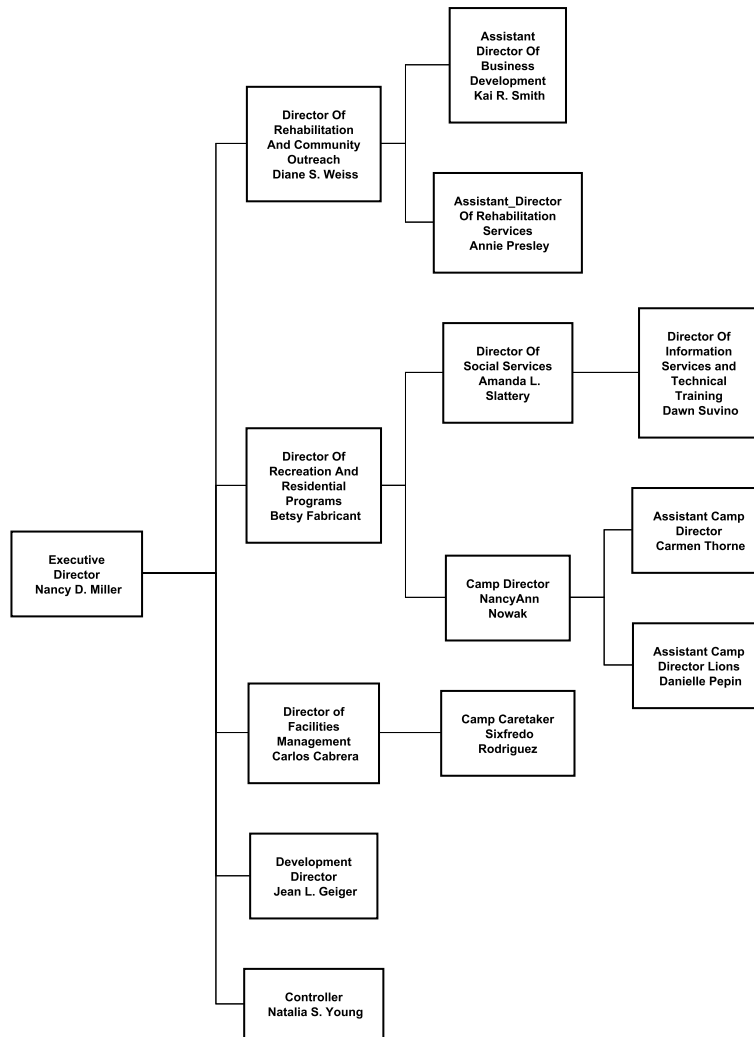


Figure 27: VISIONS Organisation Chart with drawing direction “Right”

D. Stage 4: Graphical representation

In the current prototype, the SVG code has been generated by the ILOG JViews Diagrammer. The actual SVG code generated is not relevant to the use case. This is rendered correctly but is large and complex.

E. Stage 5: Verbalisation model (Semantics)

A verbalisation model allowing the generation of a textual representation of the semantics of the diagram has been implemented.

The definition of the template has been based on guidelines on how to describe this kind of hierarchical diagrams. They recommend that these diagrams be described using well structured prose or using nested lists reflecting the levels of the hierarchical structure of the organization chart.

The verbalisation model template shown in Figure 28 has been prepared to describe the semantics of an organisation chart, parameterized items are shown in italics. This follows guidelines described in (NCAM, 2008, CORNELIS and KRIKHAAR, 2001, WEB, 2002, NBA, 2000, OU, 2009, NCAM, 2006). Two different verbalisation models have been defined for this use case:

- Presented as a more “natural language” description using some “ornament text” (version often found on the web as a text alternative to this kind of diagram). This version might present some drawbacks in certain situation due to the limitation of “Text Alternative”.
- Presented as a list reflecting the hierarchical structure of the data. This method has proved effective by many researchers aiming at presenting information non-visually (BENNETT, 2002, BROWN et al., 2004, METATLA et al., 2008)

<p>Semantics of the diagram</p> <p>General description</p> <p>The Organisation chart is entitled title</p> <p>It was created on the creation date</p> <p>The author is author information</p> <p>The chart is organized into depth of the organization chart levels and is composed of number of employee employee(s)</p> <p>Detailed prose description</p> <p>(Description of each individual level)</p> <p>At level current level in the organization chart</p> <p>There is (are) number of employee employee(s)</p> <p>(Description of each employee at this level)</p> <p>The position of employee named name of employee directly managed by position of employee named name of employee</p> <p>Detailed tabular description</p> <p>Hierarchical data structure representing the organization chart</p>

Figure 28: Verbalisation Model Template for Organisation Charts

The template (Figure 28) has been applied to the XML intermediate document extracted from the RDF data model to generate the textual representation of the semantic of the Visions organization chart.


The verbalisation model has been developed as an XSLT Transformation on the XML intermediate document retrieved from the data model.

The result obtained is presented in four different sections in the web page:

1. Menu (Figure 29): a menu allowing the navigation around the different presentations of the information
2. General description (Figure 29): this provides general information concerning the organization chart diagram: title, creation date, name of the author, an indication of the number of levels and the total number of

employees composing the diagram. This last piece of information provides an overview of the complexity of the diagram.

3. Detailed prose description (Figure 30): the detailed prose description provides a complete human-like description of the diagram as it would be described by a third party in audio.
4. Detailed list description (Figure 31): the detailed list description allows navigation around the information by making use of nested lists.



Semantics of the diagram

[See the graphical version of the Organisation chart](#)
[General description](#)
[Detailed prose description](#)
[Detailed list description](#)

General description

- The Organisation chart is entitled Visions Organisation Chart
- It was created on the 2009-09-03
- The author is Zine BEN FREDJ
- The chart is organised into 4 levels and is composed of 15 employees

Figure 29: "Menu" and "General description" sections

Detailed prose description

- At level 1
 - There is 1 employee
 - The Executive Director named Nancy D. Miller
- At level 2
 - There are 5 employees
 - The Controller named Natalia S. Young directly managed by the Executive Director named Nancy D. Miller
 - The Director Of Recreation And Residential Programs named Betsy Fabricant directly managed by the Executive Director named Nancy D. Miller
 - The Development Director named Jean L. Geiger directly managed by the Executive Director named Nancy D. Miller
 - The Director Of Rehabilitation And Community Outreach named Diane S. Weiss directly managed by the Executive Director named Nancy D. Miller
 - The Director of Facilities Management named Carlos Cabrera directly managed by the Executive Director named Nancy D. Miller
- At level 3
 - There are 6 employees
 - The Camp Director named NancyAnn Nowak directly managed by the Director Of Recreation And Residential Programs named Betsy Fabricant
 - The Director Of Social Services named Amanda L. Slattery directly managed by the Director Of Recreation And Residential Programs named Betsy Fabricant
 - The Assistant Director Of Business Development named Kai R. Smith directly managed by the Director Of Rehabilitation And Community Outreach named Diane S. Weiss
 - The Assistant_Director Of Rehabilitation Services named Annie Presley directly managed by the Director Of Rehabilitation And Community Outreach named Diane S. Weiss
 - The Assistant Director of Community Outreach named Antoinette Emers directly managed by the Director Of Rehabilitation And Community Outreach named Diane S. Weiss
 - The Camp Caretaker named Sixfredo Rodriguez directly managed by the Director of Facilities Management named Carlos Cabrera
- At level 4
 - There are 3 employees
 - The Assistant Camp Director Lions named Danielle Pepin directly managed by the Camp Director named NancyAnn Nowak
 - The Assistant Camp Director named Carmen Thorne directly managed by the Camp Director named NancyAnn Nowak
 - The Director Of Information Services and Technical Training named Dawn Suvino directly managed by the Director Of Social Services named Amanda L. Slattery

Figure 30: "Detailed prose description" section

Detailed list description

- Executive Director Nancy D. Miller
 - Controller Natalia S. Young
 - Director Of Recreation And Residential Programs Betsy Fabricant
 - Camp Director NancyAnn Nowak
 - Assistant Camp Director Lions Danielle Pepin
 - Assistant Camp Director Carmen Thorne
 - Director Of Social Services Amanda L. Slattery
 - Director Of Information Services and Technical Training Dawn Suvino
 - Development Director Jean L. Geiger
 - Director Of Rehabilitation And Community Outreach Diane S. Weiss
 - Assistant Director Of Business Development Kai R. Smith
 - Assistant_Director Of Rehabilitation Services Annie Presley
 - Assistant Director of Community Outreach Antoinette Emers
 - Director of Facilities Management Carlos Cabrera
 - Camp Caretaker Sixfredo Rodriguez

Figure 31: "Detailed list description" section

F. Stage 6: Verbalisation model (Structure)

This stage respects the same template created at stage 5 (Verbalisation model (Semantics)), but expresses the elements of the diagram in terms of objects rather than concepts. The resulting sections are shown in (Figure 32, Figure 33 and Figure 34).

Structure of the diagram

[See the graphical version of the Organisation chart](#)
[General description](#)
[Detailed prose description](#)
[Detailed list description](#)

General description

- The diagram is entitled Visions Organisation Chart
- It was created on the 2009-09-03
- The author is Zine BEN FREDJ
- This diagram is of type Hierarchy
- It is organised into 4 levels and is composed of 15 elements

List of the different elements

- There are 14 boxes.

Figure 32: Textual representation of the structure (Menu and General description sections)

Detailed prose description

- At level 1
 - There is 1 element
 - The box labelled Executive Director Nancy D. Miller
- At level 2
 - There are 5 elements
 - The box labelled Controller Natalia S. Young directly linked to the box labelled Executive Director Nancy D. Miller
 - The box labelled Director Of Recreation And Residential Programs Betsy Fabricant directly linked to the box labelled Executive Director Nancy D. Miller
 - The box labelled Development Director Jean L. Geiger directly linked to the box labelled Executive Director Nancy D. Miller
 - The box labelled Director Of Rehabilitation And Community Outreach Diane S. Weiss directly linked to the box labelled Executive Director Nancy D. Miller
 - The box labelled Director of Facilities Management Carlos Cabrera directly linked to the box labelled Executive Director Nancy D. Miller
- At level 3
 - There are 6 elements
 - The box labelled Camp Director NancyAnn Nowak directly linked to the box labelled Director Of Recreation And Residential Programs Betsy Fabricant
 - The box labelled Director Of Social Services Amanda L. Slattery directly linked to the box labelled Director Of Recreation And Residential Programs Betsy Fabricant
 - The box labelled Assistant Director Of Business Development Kai R. Smith directly linked to the box labelled Director Of Rehabilitation And Community Outreach Diane S. Weiss
 - The box labelled Assistant_Director Of Rehabilitation Services Annie Presley directly linked to the box labelled Director Of Rehabilitation And Community Outreach Diane S. Weiss
 - The rbox labelled Assistant Director of Community Outreach Antoinette Emers directly linked to the box labelled Director Of Rehabilitation And Community Outreach Diane S. Weiss
 - The box labelled Camp Caretaker Sixfredo Rodriguez directly linked to the box labelled Director of Facilities Management Carlos Cabrera
- At level 4
 - There are 3 elements
 - The box labelled Assistant Camp Director Lions Danielle Pepin directly linked to the box labelled Camp Director NancyAnn Nowak
 - The box labelled Assistant Camp Director Carmen Thorne directly linked to the box labelled Camp Director NancyAnn Nowak
 - The box labelled Director Of Information Services and Technical Training Dawn Suvino directly linked to the box labelled Director Of Social Services Amanda L. Slattery

Figure 33: Textual representation of the structure (Detailed prose description section)

Detailed list description

- box labelled Executive Director Nancy D. Miller
 - box labelled Controller Natalia S. Young
 - box labelled Director Of Recreation And Residential Programs Betsy Fabricant
 - box labelled Camp Director NancyAnn Nowak
 - box labelled Assistant Camp Director Lions Danielle Pepin
 - box labelled Assistant Camp Director Carmen Thorne
 - box labelled Director Of Social Services Amanda L. Slattery
 - box labelled Director Of Information Services and Technical Training Dawn Suvino
 - box labelled Development Director Jean L. Geiger
 - box labelled Director Of Rehabilitation And Community Outreach Diane S. Weiss
 - box labelled Assistant Director Of Business Development Kai R. Smith
 - box labelled Assistant_Director Of Rehabilitation Services Annie Presley
 - rbox labelled Assistant Director of Community Outreach Antoinette Emers
 - box labelled Director of Facilities Management Carlos Cabrera
 - box labelled Camp Caretaker Sixfredo Rodriguez

Figure 34: Textual representation of the structure (Detailed list description section)

G. Stage 7: Query systems

To illustrate the approach in the context of the organisation chart use case example (Figure 24), the following queries have been defined. Note these were derived using anecdotal skill and are not the result of a formal study into how users might want to query organisation chart diagrams.

- General Information Queries about the diagram
 - What is it? Give me more information (Graphical query system).
 - Who is the author of the diagram?
 - When was it created?
 - What is the title of the diagram?
 - How many employees have the same first name?
- Queries about the employees and the relationships between them
 - Who is this employee?
 - Who manages directly this employee?
 - Who manages indirectly this employee? (Recognition using the ontology)
 - Who reports directly to this employee?
 - Who reports indirectly to this employee? (Recognition using the ontology)
 - How many managers have more than 5 staff?

The following code (Figure 35) shows the SPARQL query “Who Manages directly...?” as it is used by the query system. First a model which reads the appropriate information from the data model is defined. The defined query is executed to find all the triples in the data model matching the query. The result is captured and then output on a Java interface.

```
String QueryPrefix = "PREFIX mydomain:<http://www.owl-ontologies.com/OrgChart_Ontology.owl#>";
String QueryString = QueryPrefix
+ "SELECT ?IDResult ?NameResult ?IDEmployee "
+ "WHERE { ?IDResult mydomain:manages ?IDEmployee . "
+ "?IDResult mydomain:name ?NameResult . "
+ "?IDEmployee mydomain:name \"\" + EmployeeName + \"\" .}";
```

Figure 35: SPARQL query "Who Manages directly...?"

As described in Chapter 4, having access to the data model and its ontology, it is possible to derive additional information that the data model holds but does not express explicitly. The ontology holds some inference rules which are used by a reasoner to make inferences on the data model. This mechanism allows “recognition” by making implicit information explicit (BROWN et al., 2004).

This is illustrated with the SPARQL query “Who Manages Indirectly...?” In this case an inference model is defined making use of the OWL reasoner to collect the appropriate result from the data model as well as the ontology it conforms to. The “manages” property is defined in the ontology as transitive, which allows the OWL reasoner to infer “who directly manages” but also who indirectly manages a given employee. Both results are collected and sent to the interface to be displayed.

❖ **Textual query system**

The following examples provide an overview of the text-based query system. Figure 36 shows the result of the query “Who Manages Directly Danielle Pepin” when indirect superiors are excluded. Figure 37 shows the result of the query “Who Manage Indirectly Danielle Pepin” when indirect superiors are included.

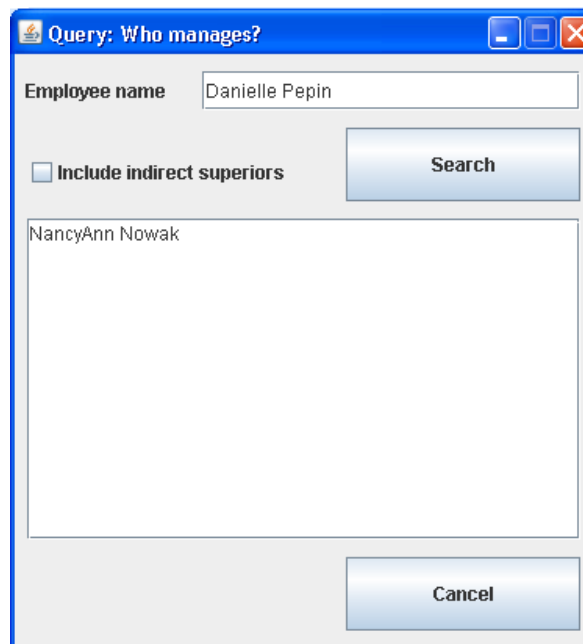


Figure 36: Result of query “Who Manages Directly Danielle Pepin”

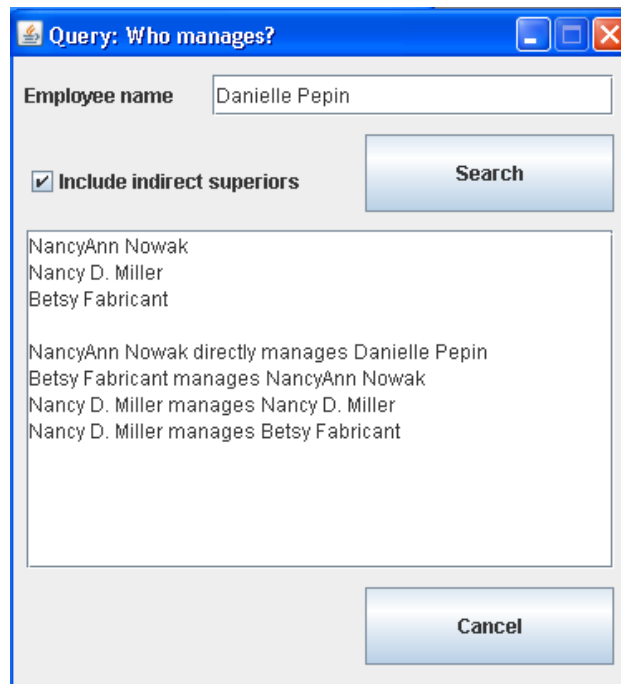


Figure 37: Result of query “Who Manages Indirectly Danielle Pepin”

❖ **Graphical smart diagram**

The two following examples illustrate the process of querying the smart SVG diagram directly. In Figure 38, the box representing the “Who Manages Directly Danielle Pepin” has been selected, and a menu offering different possible queries appears. In this example the query “Who Manages Directly” has been selected. The result is highlighted in pale red and the element queried is highlighted in pale yellow.

In the example presented in Figure 39, the query “Who Manages Indirectly” has been selected. The result is highlighted in similar ways to the previous example. This example illustrates the advantage of the “recognition” mechanism offered by GraSSML making implicit information explicit using the inference rules expressed in the ontology linked to the data model representing the information behind the diagram. The property “manages” being defined as transitive, it is possible for a reasoner to deduce who manage indirectly a given employee. The “Who Manages Indirectly” query result also includes the “Who Manages Directly” results.

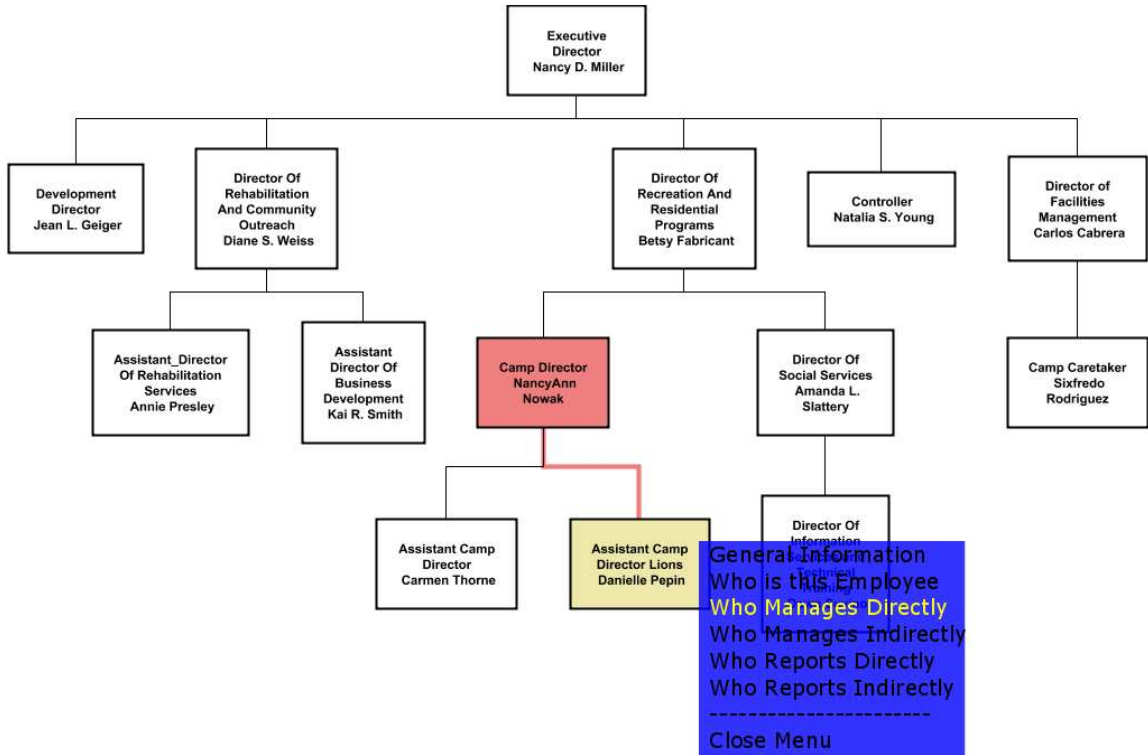


Figure 38: Result of query “Who Manages Directly Danielle Pepin”

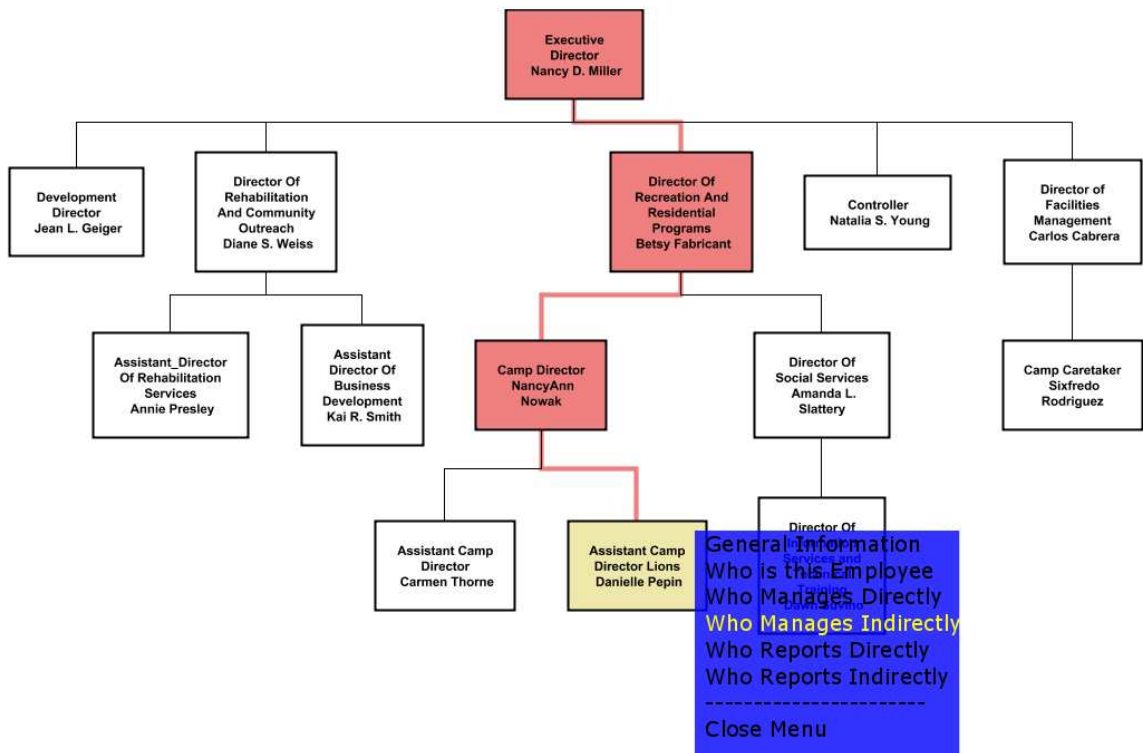


Figure 39: Result of query “Who Manage Indirectly Danielle Pepin”

5.2.3 More examples

This section explores other examples of the same class of diagrams in the same domain. A full description of each will not be provided but specific aspects which need to be tailored are described.

A. Different ontology and notational conventions

This section considers the Oxford Brookes Organisation Chart example presented in Figure 40. In this case both the ontology and the notational conventions are different. The ontology was extended to add the new concepts identified such as Dean, Assistant Dean, Director of research, etc.

The notational conventions needed to be changed. This involved changing both the stroke color of the symbol representing employee to white and changing the layout algorithm to TIP_LEAVES_OVER.

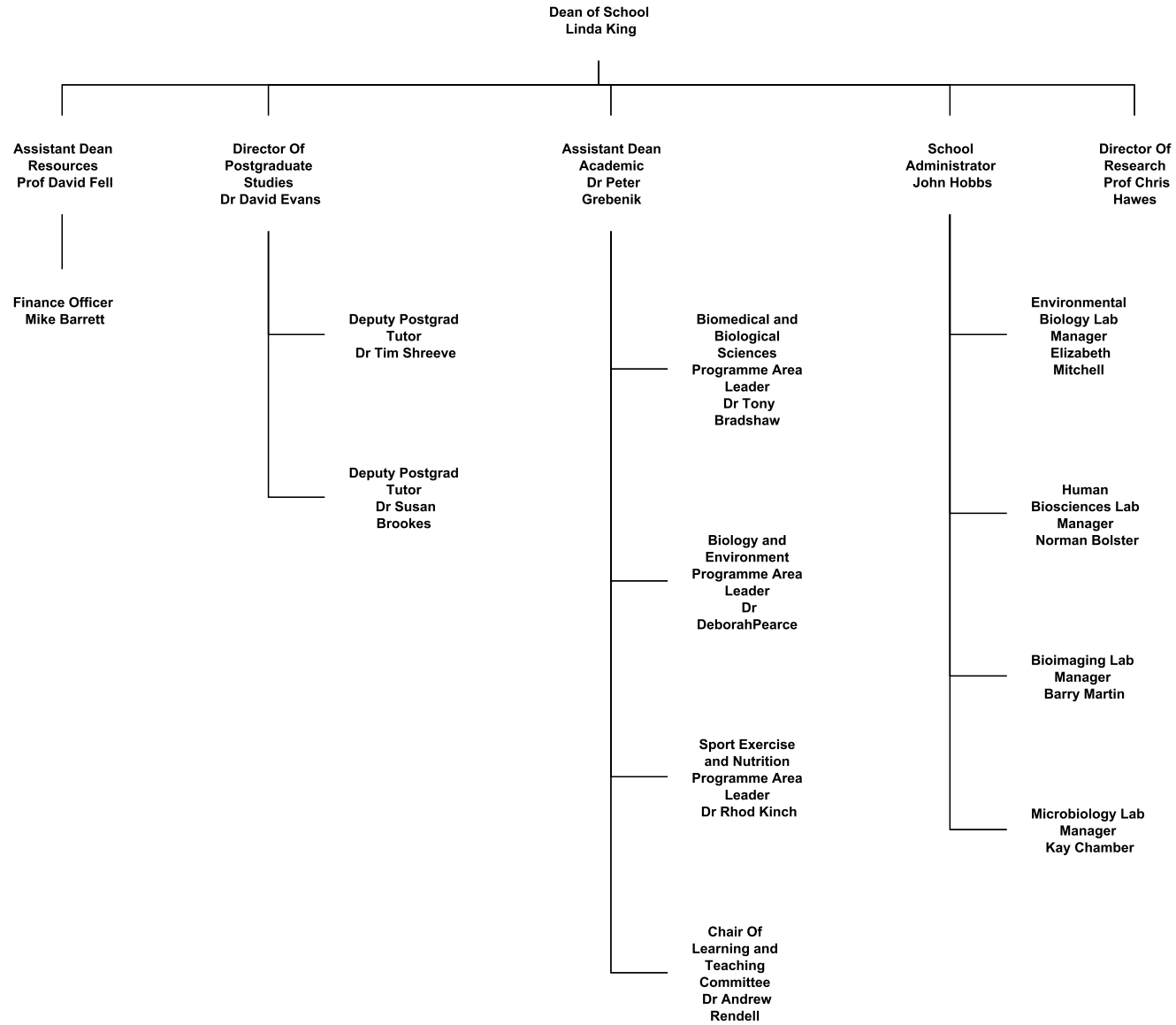


Figure 40: Oxford Brookes University Organisation Chart Example

B. Different notational conventions

Changing the direction of the Organisation chart (Figure 41) only requires changing the value of the direction attribute in the ZineML element.

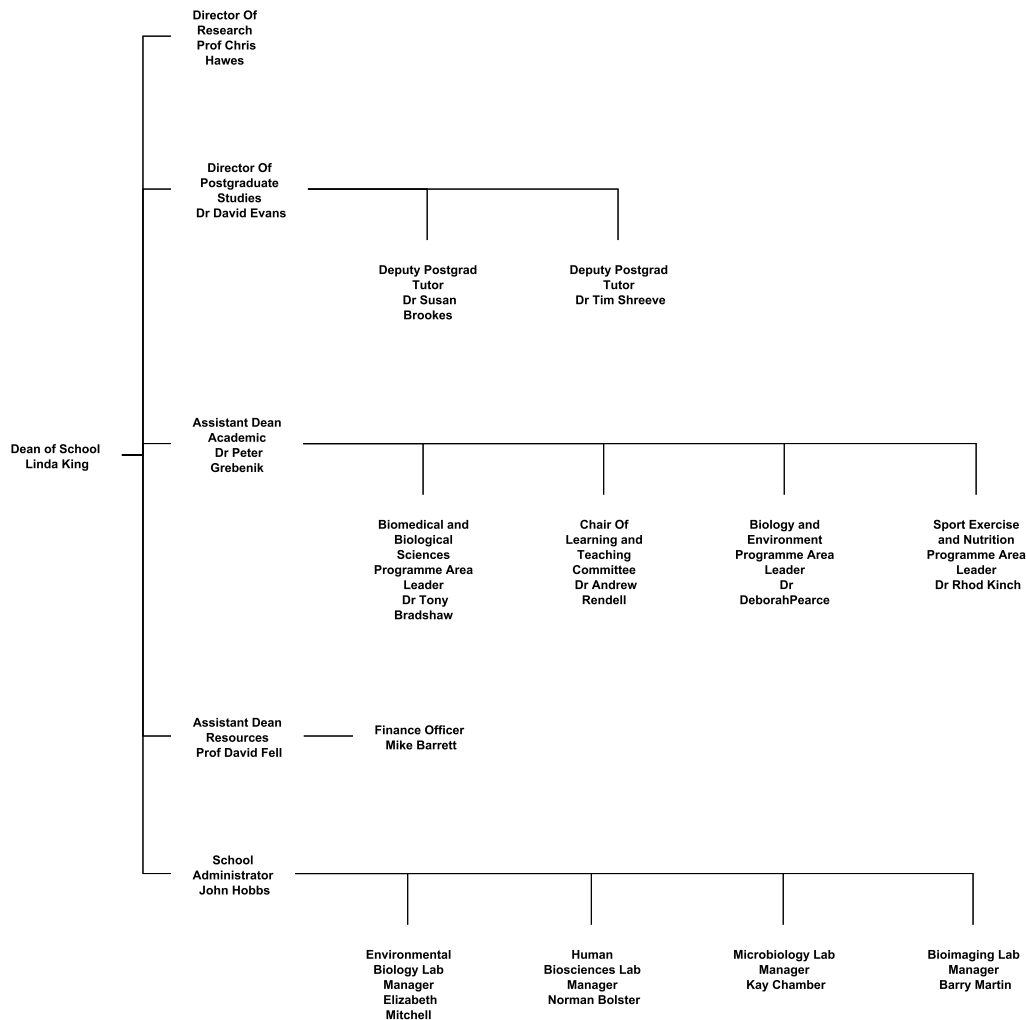


Figure 41: Oxford Brookes University Example (Direction="Right")

5.2.4 Hierarchical diagrams in different domains

Hierarchical diagrams are widely used in different domains: in mathematics (graphs, probability tree diagrams (Figure 42)), in computing (Website structure (Figure 43), Questions, Options and Criteria (QOC)), in biology (genetic family trees (Figure 44)), etc. Hierarchical diagrams are often used in order to provide an overview of a hierarchical structure. They all vary in term of information represented and notational conventions. Different symbols are employed, labelled in different ways and using different colours to represents different concepts. The links between the

different hierarchical levels can be undirected represented by lines or directed in one or both directions represented by arrows.

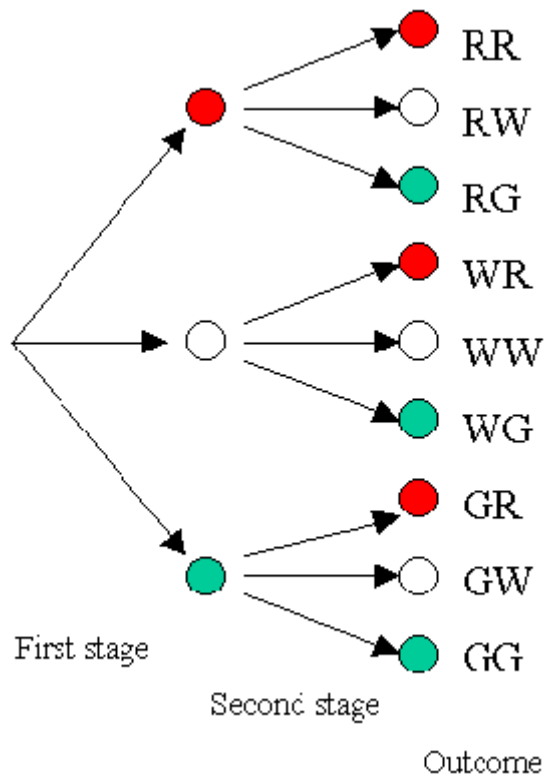


Figure 42: Probability tree diagram example

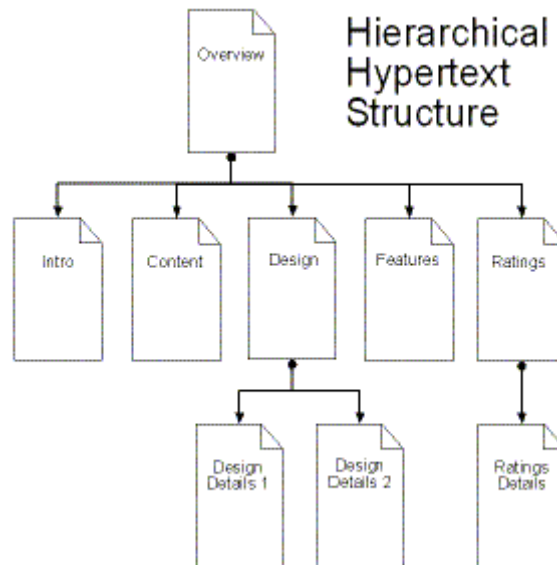


Figure 43: Website structure example

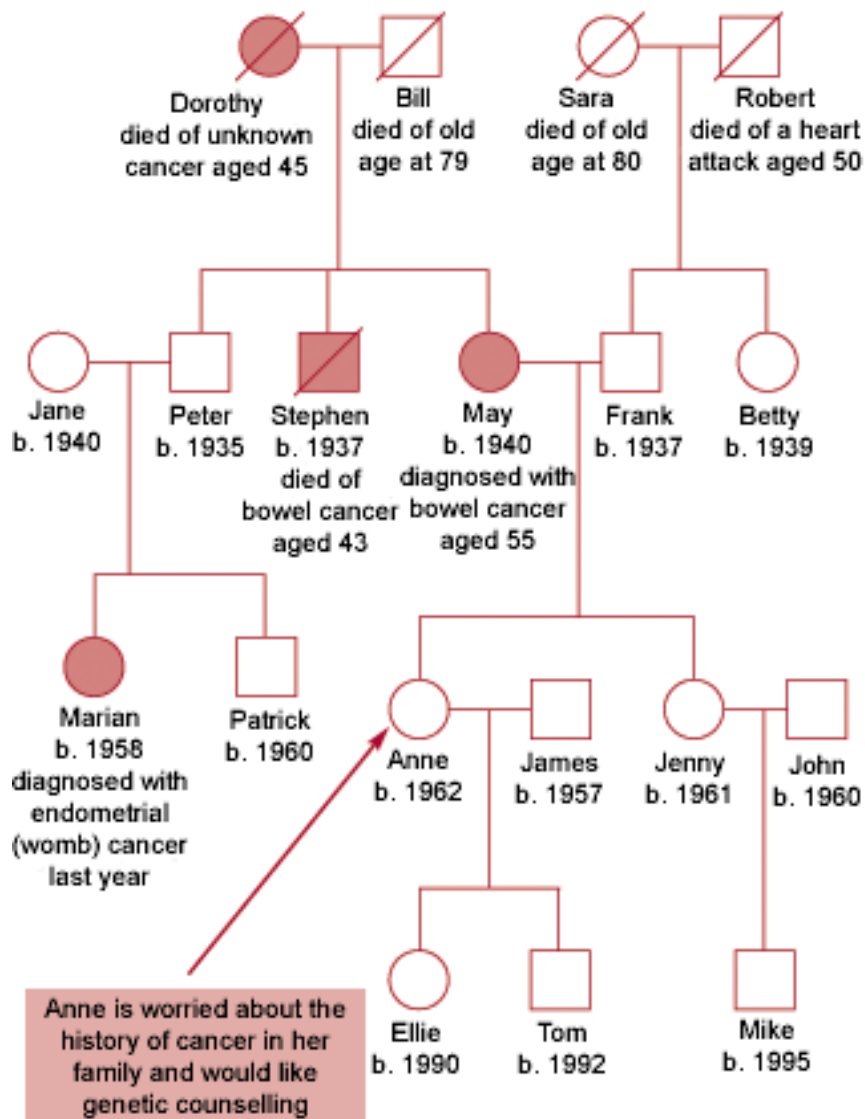


Figure 44: Genetic family tree example

GraSSML could be applied to these types of diagrams as formal conventions govern their representation and interpretation. But as mentioned in previous chapters presenting the GraSSML approach, this would require a formal study of the different domains and their notational conventions to create their appropriate ontology and necessary transformations required at the different levels of GraSSML. No changes are involved in term of the conceptual architecture of GraSSML, only the type of information required at the different levels needs updating.

5.3 Use Case: Process Diagram “UML Activity Diagram”

Process diagrams, as exemplified by “UML Activity Diagrams” will be explained in this chapter. A full description of one UML Activity Diagram will be given followed by more specific examples from the same domain. Finally some examples from the same class of diagram but in a different domain will be presented.

5.3.1 UML Activity Diagram

Process diagrams represent stages in a process using different basic shapes linked by arrows indicating process flow.

The UML activity diagram (UMLAD) (AMBLER, 2001) models high-level business processes or the transitions between states of a class. Activity diagrams show the flow of activities through the system. They are typically used for modelling a single business process. Activity diagrams show a sequence of actions, processes, activities.

5.3.2 Main Example

Figure 45 (AMBLER, 2001) presents the process of enrolling at the university for the first time. This example will be used to describe and illustrate the application of the GraSSML prototype to UML activity diagrams.

This activity diagram shows how someone new to the university would enrol for the first time. In summary, if your forms are correctly filled out, then you can proceed to enrol in the university. If your forms are not correct, then you need to obtain help, perhaps from a registrar, to fill them out correctly. After you are successfully enrolled in the university, you must attend the mandatory overview presentation, as well as enrol in at least one seminar and pay in at least some of your tuition.

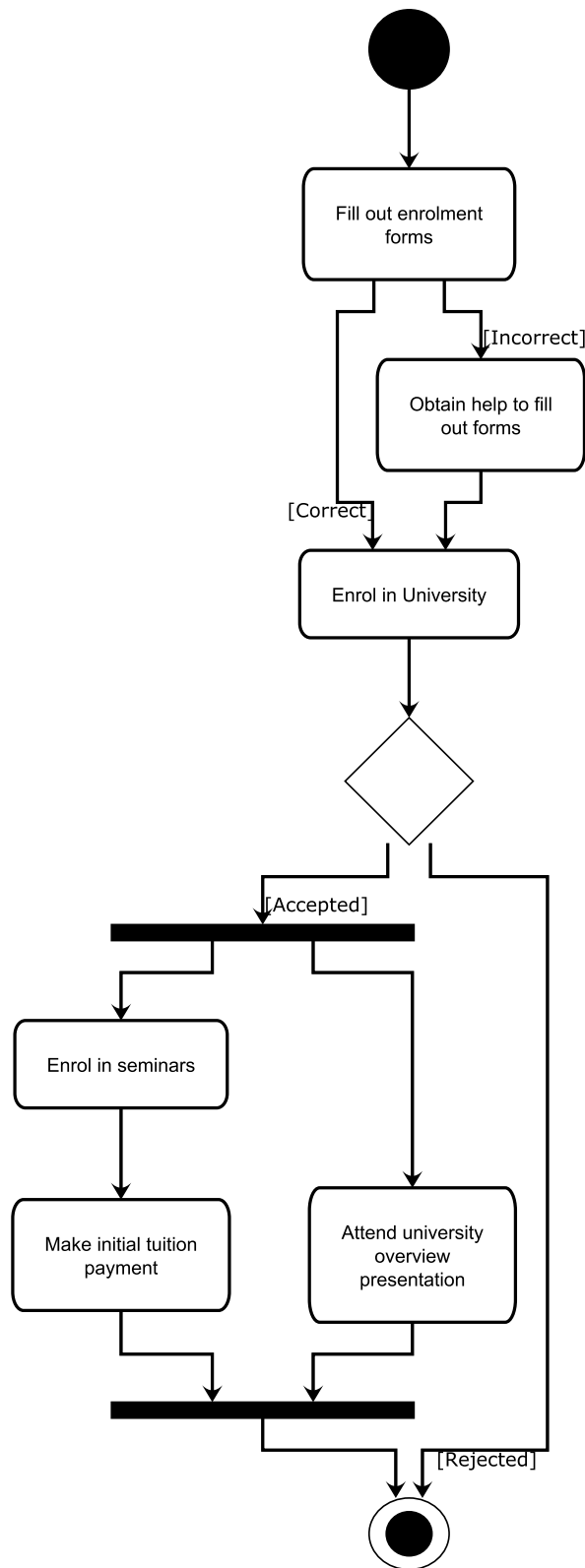


Figure 45: UML AD enrolling in the university for the first time

A. Stage 1: Ontology

Figure 46, Figure 47 and Figure 48 show respectively the class hierarchy, object properties and datatype properties for the UML Activity Diagram (UMLAD) ontology.

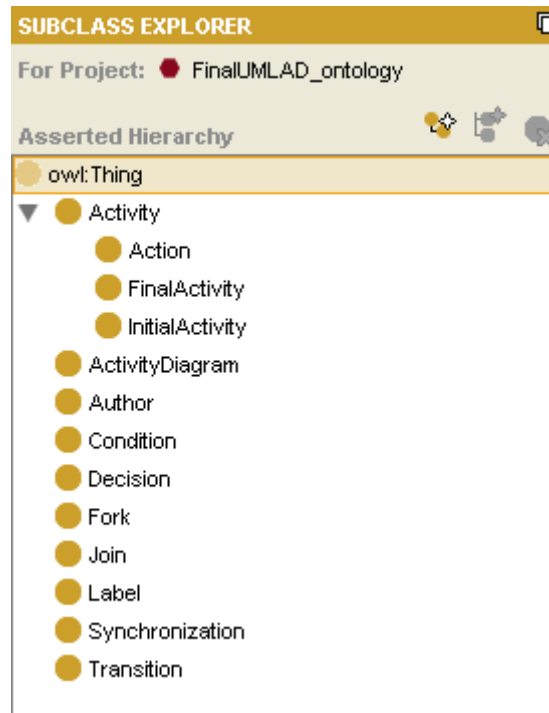


Figure 46: Hierarchy of the Classes of the UMLAD Ontology

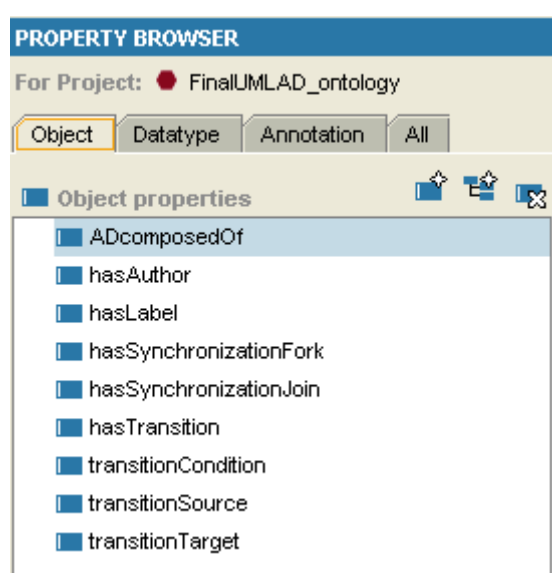


Figure 47: Object properties of the UMLAD Ontology

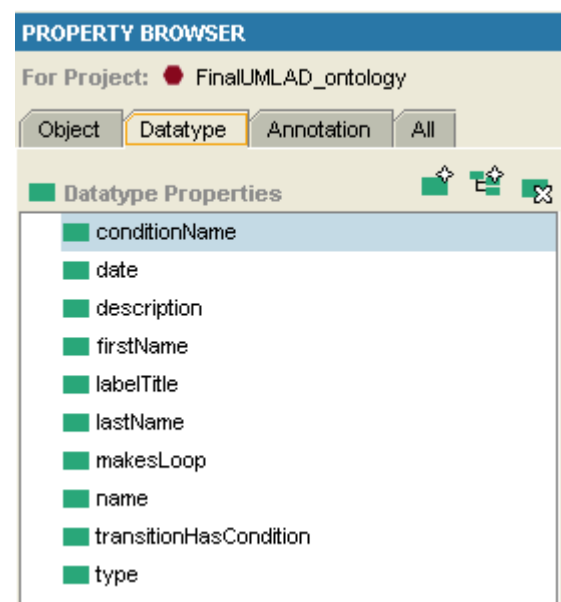


Figure 48: Datatype properties of the UMLAD Ontology

B. Stage 2: Data model

The data model was created and modified using Protégé within the GraSSML prototype.

C. Stage 3: Notational conventions

The following table (Table 7) describes the different symbols in the UML activity diagrams.



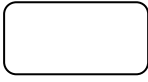

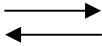

	<p>Initial Activity. This shows the starting point or first activity of the flow. It is denoted by a solid circle. It is the point at which you begin reading the action sequence.</p> <p>The filled circle represents the starting point of the activity diagram.</p>
	<p>Final Activity: The end of the Activity diagram is shown by a bull's eye symbol, also called a final activity.</p> <p>The filled circle with a border represents the ending point. A circle surrounding a smaller solid circle.</p>
	<p>The rounded rectangles represent processes or activities that are performed.</p> <p>The rounded rectangle represents an action</p>
	<p>Decisions: if a decision is to be made, it is depicted by a diamond, with the options written on either side of the arrows emerging from the diamond, within box brackets (see the [condition] symbol).</p> <p>The diamond represents decision points. It could have many possible outcomes depending on the condition.</p>
	<p>Transitions: The arrows model the flow order between the various activities.</p> <p>The arrows represent transitions between activities.</p>
<p>[Condition]</p>	<p>Conditions: The text on the arrows represent conditions that must be fulfilled to proceed along the transition and are always described using the format “[condition]”</p>
	<p>Concurrent Activities: Some activities occur simultaneously or in parallel. Two activities can occur in parallel when no direct relationship exists between them and they must both occur before a third activity can.</p> <p>Such activities are called concurrent activities. For example, listening to the lecturer and looking at the blackboard is a parallel activity. This is represented by a horizontal split (thick dark line) and the two concurrent activities next to each other, and the horizontal line again to show the end of the parallel activity.</p> <p>The thick bars represent the start and end of potentially parallel processes.</p>
<p>Title+ ID</p>	<p>The label at the top is composed of an appropriate title for the diagram and an identifier.</p> <p>It may also contain a date and the names of the authors.</p>

Table 7: Symbols in the UML activity diagrams

The following key aspects of the conventions are defined for this use case:

- There can be only one initial activity symbol on an activity diagram and only one transition line connecting the initial activity to an action. The location of the initial activity can be anywhere; but it is usual to place it at the top left corner of the diagram.
- Every activity diagram should have at least one final activity symbol to show where the action sequence ends. Several final activity symbols are allowed indicating different terminations in the logic.
- Each decision point symbol will have multiple transition lines connecting it to different actions. Each transition line involved in a decision point must be labelled with a guard condition text which is always placed in brackets. A guard condition explicitly says when to follow a transition line to the next action. A second approach to modelling decisions is to have multiple transition lines coming out of an action. As with decisions points, each transition line involved must be labelled with a guard condition text.
- Activities which occur simultaneously or in parallel are called concurrent activities. These are represented by a fork which is a thick solid bar with one transition entering it and several leaving it and a join which is also a thick solid bar but with several transitions entering it and only one leaving it. For every start (fork) there is an end (join).

UML activity diagrams are usually documented with a brief description of the activity and an indication of any action taken during a process. The ZineML elements defined are presented in Table 8.





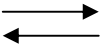

Symbols	Concept	ZineML element	Attributes (default value)
	Initial Activity	<circle>	id (no default value) color (black) visibility (true) strokeColor (black)
	Final Activity	<Dcircle>	id (no default value) visibility (true) colorIn (black) colorOut (white) strokeColorIn(black) strokeColorOut(black)
	Action	<Rbox>	id (no default value) color (white) visibility (true) strokeColor (black) labelColor (black) labelVisibility (true)
	Decision	<diamond>	id (no default value) color (white) visibility (true) strokeColor (black)
	Transition	<arrow>	id (no default value) makesLoop (false) from (no default value) to (no default value)
[Condition]	Condition	Content of the <arrow> element if condition exist	
	Fork or Join	<thickBar>	id (no default value) color (black) visibility (true) strokeColor (black)

Table 8: ZineML elements for the UML activity diagrams

D. Stage 4: Graphical representation

In the current GraSSML prototype the SVG code has been generated by the ILOG JViews Diagrammer.

E. Stage 5: Verbalisation model (Semantics)

This verbalisation template has been designed taking into account expert advice on how to describe this type of diagram textually and in audio. The following template (Figure 49) extracted from (TAYLOR, 2008b) served as a basis to express the template for process diagrams.

This flow chart shows the [process/flow/...] [Optional list of symbols or explain symbols used] [Describe overall process from start to end points] [Describe simplest path first, if necessary define subsections and relation to each other, then details of sub-sections.]

Figure 49: Template extracted from (TAYLOR, 2008b)

The following template (Figure 50) has been applied to an XML intermediate document extracted from the RDF data model to generate the textual representation of the semantic of the UML Activity Diagram:

Semantic of the diagram <i>Link to the graphical representation of the diagram</i> General description This UML Activity Diagram is entitled <i>title</i> The author is <i>Author information</i> The creation date is <i>date</i> Description: <i>Extra description provided by the author if exist</i> The activity diagram is composed of <i>List of objects composing the diagram (e.g. 6 Actions)</i> Detailed prose description Description of the overall process represented in the diagram from start to end points Detailed tabular description Hierarchical data structure representing the UML activity diagram from start to end points

Figure 50: Verbalisation Model Template for UML Activity Diagrams

The verbalisation model has been developed as an XSLT Transformation on the XML intermediate document retrieved from the data model.

The resulting textual representation of the semantics of the diagram is shown below (Figure 51, Figure 52 and Figure 53).

Semantics of the diagram

[See the graphical version of the Organisation chart](#)

[General description](#)

[Detailed prose description](#)

[Detailed list description](#)

General description

- This UML Activity Diagram is entitled Enrol at the University for the first time
- The author is Zine BEN FREDJ
- The creation date is 2008-11-29
- Description: This activity diagram depicts the business logic for how someone new to the university would enrol for the first time.
- The activity diagram is composed of:
 - 1 Label
 - 1 Starting Point
 - 6 Actions
 - 1 Decision Point
 - 2 Parallel Processes Bars
 - 1 Ending Point
 - 13 Transitions

Figure 51: Textual representation of the semantics (General description section)

Detailed prose description

The process starts with the Action Fill out enrolment forms.

At this point a decision has to be made. Either decision 1 meaning the Action called Fill out enrolment forms is Incorrect or decision 2 meaning the Action called Fill out enrolment forms is Correct.

1. The Action Fill out enrolment forms is Incorrect.
If the Action Fill out enrolment forms is Incorrect then process the Action called Obtain help to fill out forms. After the Action called Obtain help to fill out forms process the Action called Enrol in University. At this point a decision has to be made. Either decision 1 meaning the Action called Enrol in University is Rejected or decision 2 meaning the Action called Enrol in University is Accepted.
 1. The Action Enrol in University is Rejected. If the Action Enrol in University is Rejected then process to the Final activity, you have reached the End of the activity.
 2. The Action Enrol in University is Accepted. If the Action Enrol in University is Accepted then process the start of the concurrent activities. There are 2 sets of concurrent activities
 - A. Set of activities 1 process the Action called Attend university overview presentation.
 - B. Set of activities 2 process the Action called Enrol in seminars. After the Action called Enrol in seminars process the Action called Make initial tuition payment.Process the end of the concurrent activities. After the end of the concurrent activities process to the Final activity, you have reached the End of the activity.
2. The Action Fill out enrolment forms is Correct.
If the Action Fill out enrolment forms is Correct then process the Action called Enrol in University. At this point a decision has to be made. Either decision 1 meaning the Action called Enrol in University is Rejected or decision 2 meaning the Action called Enrol in University is Accepted.
 1. The Action Enrol in University is Rejected. If the Action Enrol in University is Rejected then process to the Final activity, you have reached the End of the activity.
 2. The Action Enrol in University is Accepted. If the Action Enrol in University is Accepted then process the start of the concurrent activities. There are 2 sets of concurrent activities
 - A. Set of activities 1 process the Action called Attend university overview presentation.
 - B. Set of activities 2 process the Action called Enrol in seminars. After the Action called Enrol in seminars process the Action called Make initial tuition payment.Process the end of the concurrent activities. After the end of the concurrent activities process to the Final activity, you have reached the End of the activity.

Figure 52: Textual representation of the semantics (Detailed prose description section)

Detailed list description

- START initial activity
- Action Fill out enrolment forms. At this point a decision has to be made, select one of the 2 following options

[Fill out enrolment forms Incorrect](#) or [Fill out enrolment forms Correct](#)

1. Action Fill out enrolment forms Incorrect
 - Action Obtain help to fill out forms
 - Action Enrol in University
 - Decision Point. At this point a decision has to be made, select one of the 2 following options [Enrol in University Rejected](#) or [Enrol in University Accepted](#)
 1. Action Enrol in University Rejected
 - Final activity, you have reached the End of the activity.
 2. Action Enrol in University Accepted
 - Start of the concurrent activities. There are 2 sets of concurrent activities
 - A. Set of activities 1
 - Action Attend university overview presentation
 - B. Set of activities 2
 - Action Enrol in seminars
 - Action Make initial tuition payment
 - End of the concurrent activities
 - Final activity, you have reached the End of the activity.
2. Action Fill out enrolment forms Correct
 - Action Enrol in University
 - Decision Point. At this point a decision has to be made, select one of the 2 following options [Enrol in University Rejected](#) or [Enrol in University Accepted](#)
 1. Action Enrol in University Rejected
 - Final activity, you have reached the End of the activity.
 2. Action Enrol in University Accepted
 - Start of the concurrent activities. There are 2 sets of concurrent activities
 - A. Set of activities 1
 - Action Attend university overview presentation
 - B. Set of activities 2
 - Action Enrol in seminars
 - Action Make initial tuition payment
 - End of the concurrent activities
 - Final activity, you have reached the End of the activity.

Figure 53: Textual representation of the semantics (Detailed list description section)

F. Stage 6: Verbalisation model (Structure)

The resulting textual representation of the structure of the diagram is shown below (Figure 54, Figure 55 and Figure 56).

Structure of the diagram

[See the graphical version of the Organisation chart](#)

[General description](#)

[Detailed prose description](#)

[Detailed list description](#)

General description

- This UML Activity Diagram is entitled Enrol at the University for the first time
- The author is Zine BEN FREDJ
- The creation date is 2008-11-29
- Description: This activity diagram depicts the business logic for how someone new to the university would enrol for the first time.
- The activity diagram is composed of:
 - 1 circle
 - 1 double circle (circle surrounding a smaller black solid circle)
 - 6 rounded boxes (box with rounded corner)
 - 1 diamond
 - 2 thick bars
 - 13 arrows

Figure 54: Textual representation of the structure (General description section)

Detailed prose description

START. This activity diagram starts with a circle .

The circle is connected to a rounded box labelled Fill out enrolment forms by an arrow. The rounded box labelled Fill out enrolment forms is connected to 2 shapes

1. The rounded box labelled Fill out enrolment forms is connected to the rounded box labelled Obtain help to fill out forms by an arrow labelled [Incorrect] . The rounded box labelled Obtain help to fill out forms is connected to the rounded box labelled Enrol in University by an arrow. The rounded box labelled Enrol in University is connected to the diamond (Decision_Point1) by an arrow. The diamond (Decision_Point1) is connected to 2 shapes
 1. The diamond (Decision_Point1) is connected to the double circle END by an arrow labelled [Rejected] .
 2. The diamond (Decision_Point1) is connected to the thick bar (S1Fork) by an arrow labelled [Accepted] . The thick bar (S1Fork) is connected to 2 shapes
 1. The thick bar (S1Fork) is connected to the rounded box labelled Attend university overview presentation by an arrow. The rounded box labelled Attend university overview presentation is connected to the thick bar (S1Join) by an arrow. The thick bar (S1Join) is connected to the double circle END by an arrow.
 2. The thick bar (S1Fork) is connected to the rounded box labelled Enrol in seminars by an arrow. The rounded box labelled Enrol in seminars is connected to the rounded box labelled Make initial tuition payment by an arrow. The rounded box labelled Make initial tuition payment is connected to the thick bar (S1Join) by an arrow. The thick bar (S1Join) is connected to the double circle END by an arrow.
2. The rounded box labelled Fill out enrolment forms is connected to the rounded box labelled Enrol in University by an arrow labelled [Correct] . The rounded box labelled Enrol in University is connected to the diamond (Decision_Point1) by an arrow. The diamond (Decision_Point1) is connected to 2 shapes
 1. The diamond (Decision_Point1) is connected to the double circle END by an arrow labelled [Rejected] .
 2. The diamond (Decision_Point1) is connected to the thick bar (S1Fork) by an arrow labelled [Accepted] . The thick bar (S1Fork) is connected to 2 shapes
 1. The thick bar (S1Fork) is connected to the rounded box labelled Attend university overview presentation by an arrow. The rounded box labelled Attend university overview presentation is connected to the thick bar (S1Join) by an arrow. The thick bar (S1Join) is connected to the double circle END by an arrow.
 2. The thick bar (S1Fork) is connected to the rounded box labelled Enrol in seminars by an arrow. The rounded box labelled Enrol in seminars is connected to the rounded box labelled Make initial tuition payment by an arrow. The rounded box labelled Make initial tuition payment is connected to the thick bar (S1Join) by an arrow. The thick bar (S1Join) is connected to the double circle END by an arrow.

Figure 55: Textual representation of the structure (Detailed prose description section)

Detailed list description

- START circle
- rounded box Fill out enrolment forms, 2 arrows.

[arrow \[Incorrect\]](#) or [arrow \[Correct\]](#)

1. arrow [Incorrect]
 - rounded box Obtain help to fill out forms
 - rounded box Enrol in University
 - diamond (Decision_Point1) , 2 arrows. [arrow \[Rejected\]](#) or [arrow \[Accepted\]](#)
 1. arrow [Rejected]
 - double circle END
 2. arrow [Accepted]
 - thick bar (S1Fork) , 2 arrows. [arrow 1](#) or [arrow 2](#)
 1. arrow 1
 - rounded box Attend university overview presentation
 - thick bar (S1Join)
 - double circle END
 2. arrow 2
 - rounded box Enrol in seminars
 - rounded box Make initial tuition payment
 - thick bar (S1Join)
 - double circle END
2. arrow [Correct]
 - rounded box Enrol in University
 - diamond (Decision_Point1) , 2 arrows. [arrow \[Rejected\]](#) or [arrow \[Accepted\]](#)
 1. arrow [Rejected]
 - double circle END
 2. arrow [Accepted]
 - thick bar (S1Fork) , 2 arrows. [arrow 1](#) or [arrow 2](#)
 1. arrow 1
 - rounded box Attend university overview presentation
 - thick bar (S1Join)
 - double circle END
 2. arrow 2
 - rounded box Enrol in seminars
 - rounded box Make initial tuition payment
 - thick bar (S1Join)
 - double circle END

Figure 56: Textual representation of the structure (Detailed list description section)

G. Stage 7: Query systems

❖ Textual query system

Figure 57 shows the selection of the query “Traverse Diagram From ... To...” from the view mode interface.

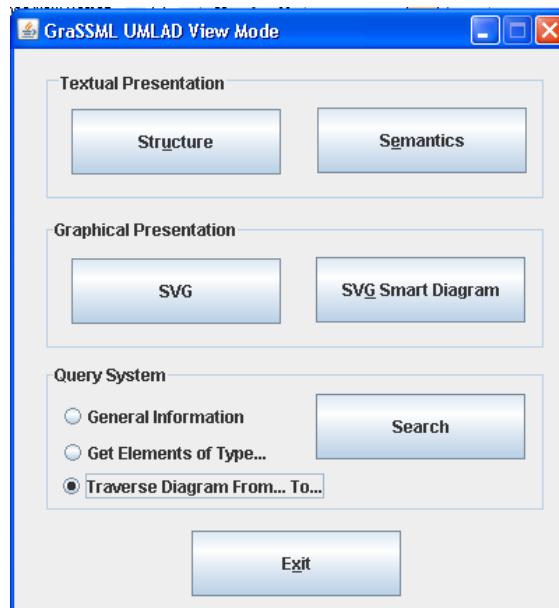


Figure 57: Textual query system from View mode interface

Once the query is selected, the user is required to specify the “from” and “to” options from the appropriate combo boxes (Figure 58). The choices presented in the combo boxes are extracted from the data model. This technique is used to avoid possible mistakes while entering the values. The user can select the check box “Step by step” if he wants to follow the process of traversing the diagram step by step. While traversing the graph, if different paths are possible, the user would be requested to select one possible option via a dialogue box. In this example the option “Fill out enrolment forms is Correct” has been selected (Figure 58) and the option “Enrol in University” is Accepted has been selected (Figure 59). The final result is presented in Figure 60. The whole process has been implemented using SPARQL to query the data model containing the information of the diagram.

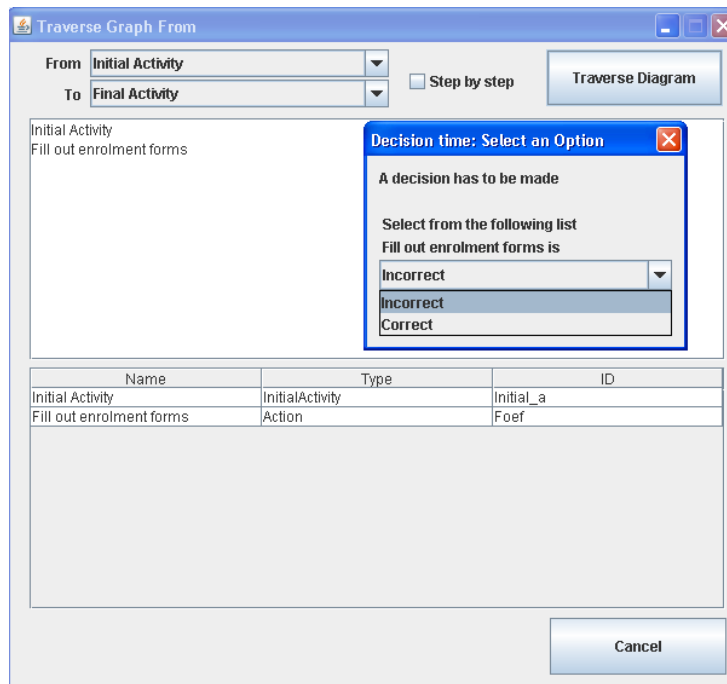


Figure 58: Completing selection of an option

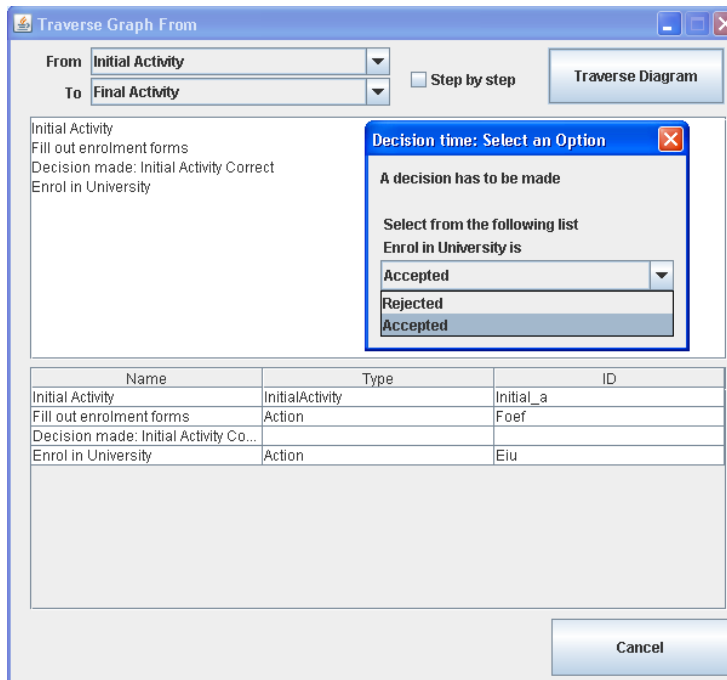


Figure 59: Selecting option for “Enrol in University”

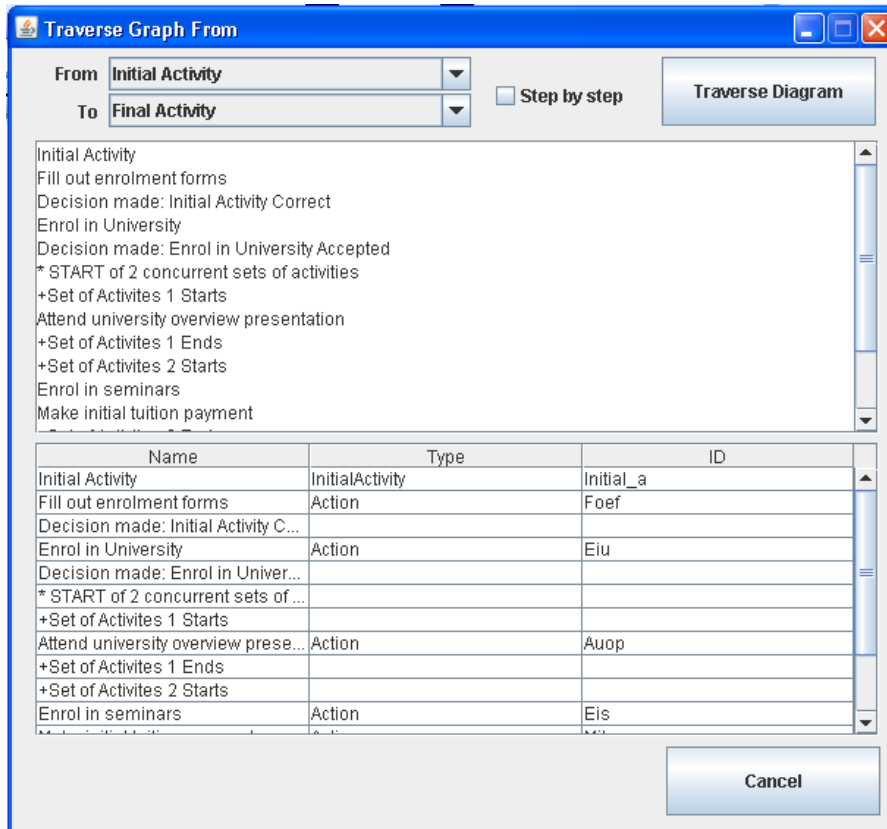


Figure 60: Result of query “Traverse Graph from ... To ...”

❖ Graphical Smart Diagrams

Two examples are illustrated. In the first example the query “Show me all elements of type...” is selected. A dialogue box is displayed waiting for the type to be selected from a list (Figure 61). The type “Activity” is selected. The results are shown in Figure 62. In this example, access to the data model as well as access to its corresponding ontology plays an important role. The ontology holds some inference rules which are used by a reasoner to make inferences on the data model. This mechanism allows “recognition” by making implicit information explicit (BROWN et al., 2004). Using the Jena inference model and the OWL ontology reasoner, in this specific query, since action, initial activity and final activity are defined as sub-classes of activity, they are inferred as being activities. The result presented in Figure 62 reflects this result as the correct elements are highlighted.

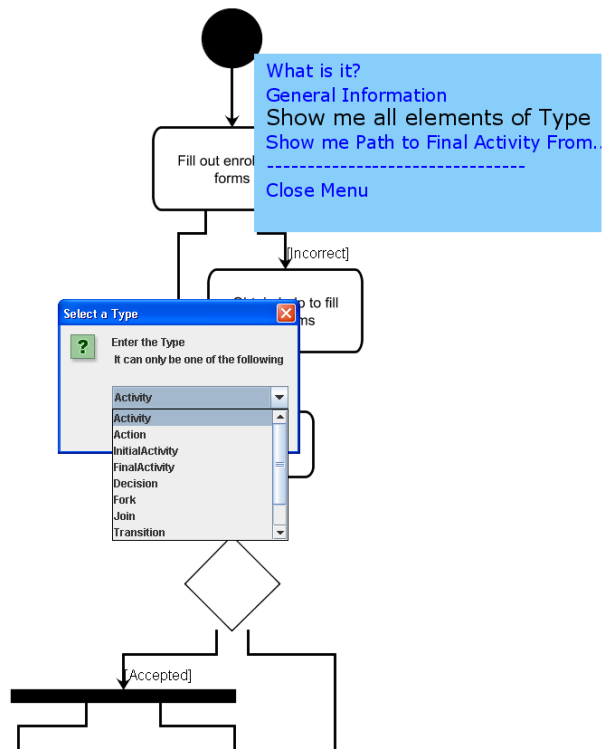


Figure 61: Selection of Query “Show me all elements of Type...” with option selected “Activity”

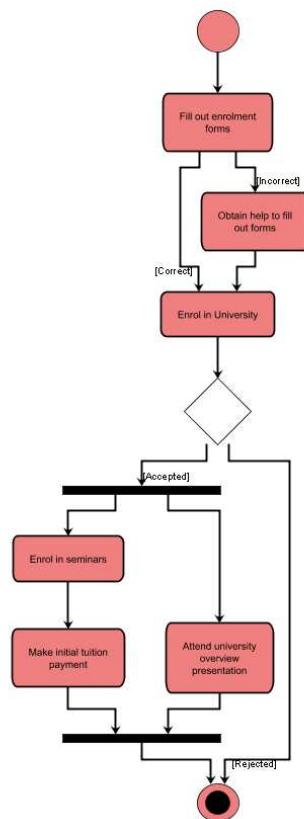


Figure 62: Results of the query “show me all elements of type Activity”

The second example is the same query described for the textual query system. From the menu presented in Figure 63, the query “Show me path to final activity from...” is selected with the option “Initial activity”. The path from the initial activity to the final activity is traversed interactively using information provided by the user when needed. The same decisions are selected as the previously presented example. Figure 64 shows the results of the query executed.

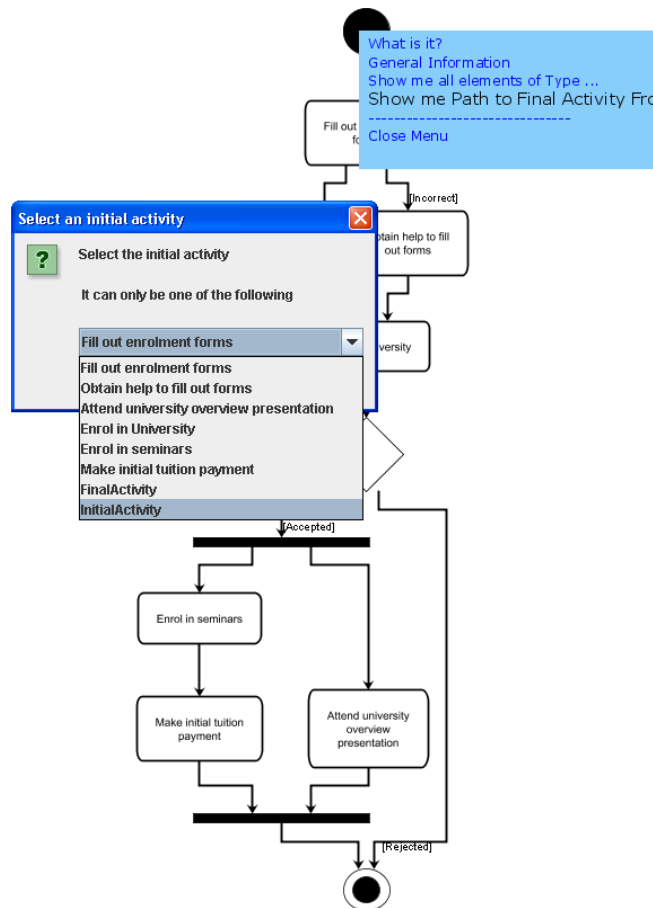


Figure 63: Selection of Query “Show me Path to Final Activity From...” with option selected “Initial Activity”

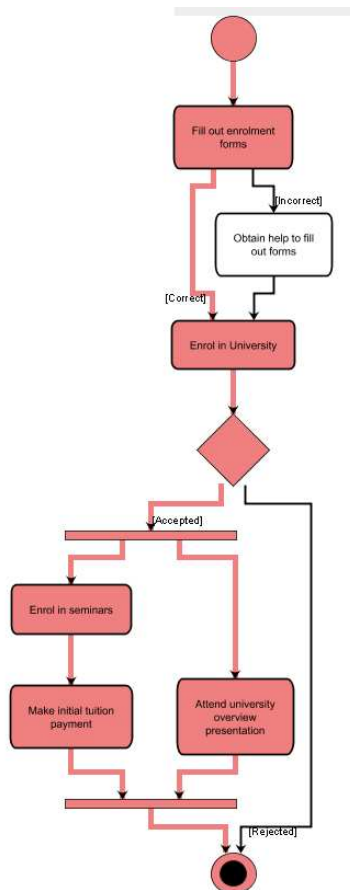


Figure 64: Results of query “Show me Path to Final Activity From ...”

H. More examples

This section explores other examples of the same type of diagrams in the same domain. A full description of each will not be provided but specific aspects which need to be tailored are described. The UML Activity diagram ontology could be extended to include other concepts such as sub-action, swimlanes, other resources, etc. All of these features can be implemented however; there is much work ahead before eventually reaching a full implementation compliant with the UML Standard.

❖ UMLAD with Different notational conventions

Figure 65 describes the bowling activity. From this Activity diagram, you can get the information needed by an animation shown when someone made a strike.

In this case only the notational conventions at the structure level needs to be changed. This involves changing the colour of the symbol representing an action from white to green and changing the colour of symbol representing a decision point from white to red.

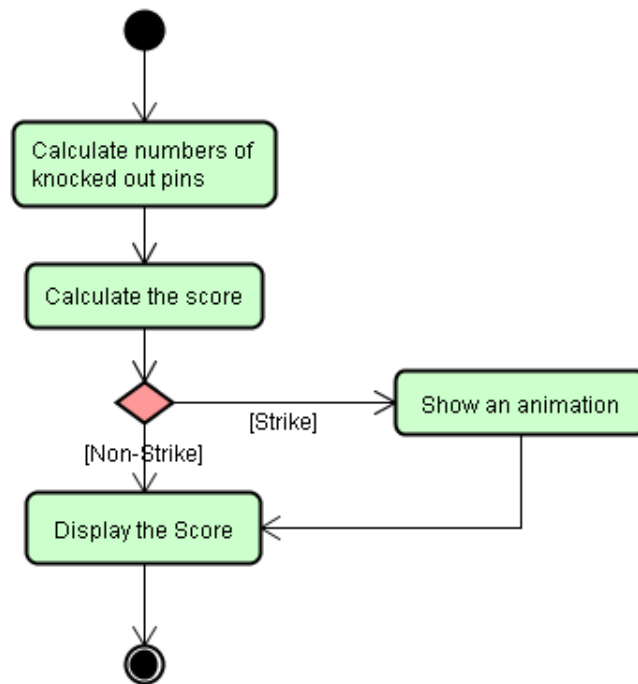


Figure 65: Activity Diagram example “bowling”

Because of time constraints, in the current GraSSML prototype the XSLT transformations needed to apply the notational conventions are hand coded, so such modifications require the manual modification of the appropriate information.

One could easily create a tool to automatically generate the transformations from a set of defined requirements. Such a tool would allow the user to map the concepts and relationships in the ontology to their corresponding notational conventions. Such a tool would essentially be a table where the user would enter the notational conventions expressed in ZineML for all the concepts and relationships.

❖ **Sub-action extensions**

In an activity diagram, activities can be nested. This involves some actions referring to another activity diagram showing the internal structure of that action. Two notational conventions are possible: represent the nested activity diagram inside the action (Figure 66) or represent an external activity diagram inside the action (Figure 67).

This involves changes in both the semantic level and the structure level. At the semantic level the ontology should reflect this kind of construct. This could be done by defining an action as either simple or composed. If defined as composed the action could itself refer to another activity diagram describing it.

At the structure level, some modifications would be required depending on how the nested activity diagram is to be represented, as nested or external representations. For both cases, the notational conventions would need to be updated.

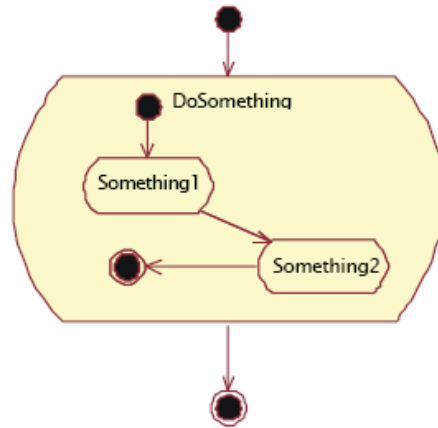


Figure 66: Nested activity diagram example

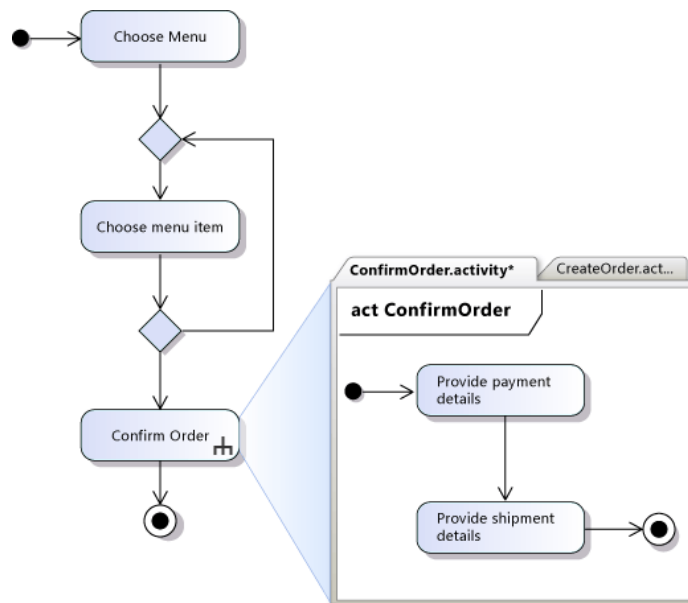


Figure 67: Example of sub-activities

❖ **Importing Activity Diagrams from StarUML using XMI**

This example illustrates the generation of a data model of a UML Activity diagram created using a software modelling tool and exported in XMI to be integrated within GraSSML.

The XML Metadata Interchange (XMI) is an Object Management Group (OMG) standard for exchanging metadata information via Extensible Markup Language (XML). Its most common use is as an interchange format for UML models.

The StarUML project is an open source UML project to develop a fast, freely-available software modelling tool and platform. With StarUML, it is possible to import as well as export XMI 1.1. UML 1.3.

StarUML (Figure 68) was used to create a UML Activity diagram named “Jukebox”. Once created, the diagram was exported in XMI. Based on the information in the XMI exported, a data model conforming to the UMLAD ontology defined within GraSSML has been generated.

The transformation uses a set of templates, matching the appropriate XMI tags to create the necessary RDF resources for each of the diagram concepts. The stylesheet concatenates the UMLAD ontology directly into the RDF and the instances created conform to the previously created ontology. The generated OWL file can be opened in Protégé successfully.

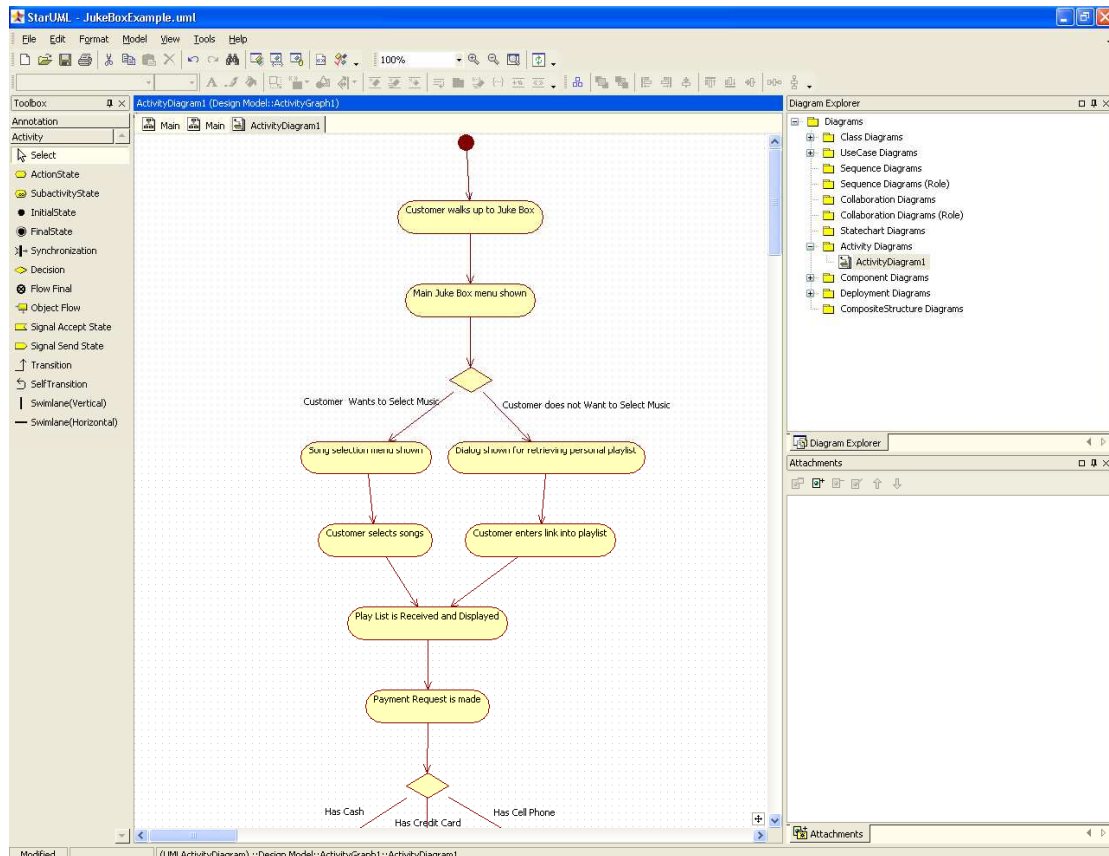


Figure 68: StarUML example “Jukebox”

5.3.3 Process diagrams in different domains

Activity diagrams are very similar to Flowchart diagrams (Figure 69): system flowcharts, program flowcharts, document flowcharts, logic flowcharts and process flowcharts. They represent processes or algorithms using shapes linked by arrows which, as for UML activity diagram, indicate flows. They are used in many different fields (e.g. computing, business), varying in terms of notational conventions (different shape in different colours) to analyse, design and document processes.

The three following figures present three different examples of flowcharts (Figure 69, Figure 70 and Figure 71).

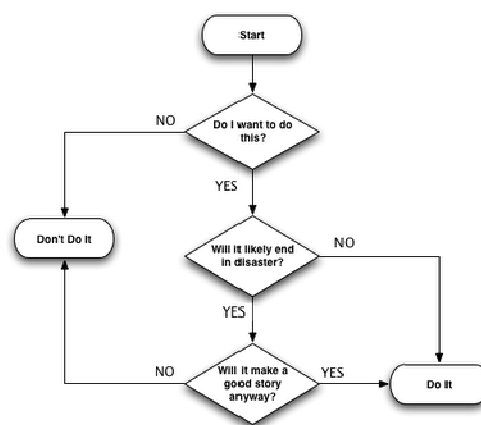


Figure 69: A simple flowchart example

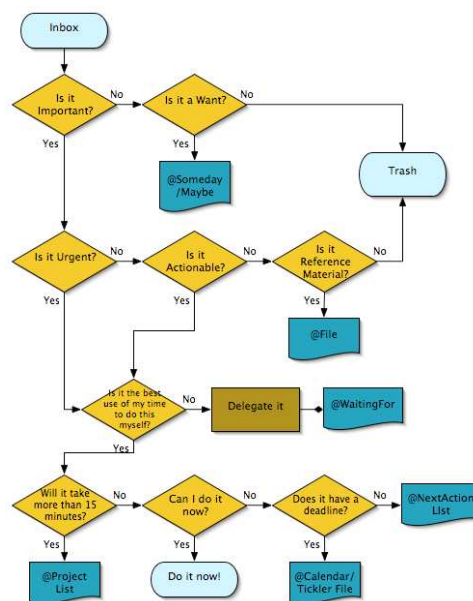


Figure 70: Another example of a flowchart with different notational conventions

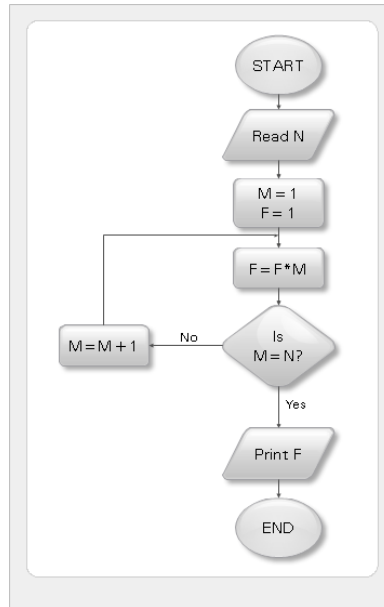


Figure 71: Example of a flowchart for computing factorial N (N!)

As can be observed, different symbols are employed. Some are similar to those explored in the UML Activity diagram use case and some depend on the type of flowchart represented.

Start and end symbols can be represented by either circles or rounded rectangles labelled “Start”, “End” or using a short sentence indicating either the start or end of the process. The flow of control is represented as an arrow between the symbols and decisions are also represented as diamond symbols. Different symbols such as cylinders, parallelograms or rectangles with wavy base are also introduced.

GraSSML could be applied to such diagrams but it would require a formal study of the different conventions to create a suitable ontology.

The main differences are:

- Ontology definition
- Notational conventions, nothing major
- The ZineML language will need to be extended with the new shapes
- Verbalisation model: Depending on the type of flow chart selected, the implementation of the verbalisation template will need some modification in the traversal of the graph defined by the data model. But the structure of the verbalisation model itself does not have to be modified as it applies to all process diagrams.

5.4 Conclusion

The GraSSML prototype has demonstrated that the GraSSML conceptual approach is sound.

This chapter has described how the GraSSML conceptual approach has been concretely achieved using web technologies and tools available. It gave a concrete view of one possible realisation of the approach but it would be relatively easy to replace the current set of technologies with alternatives of similar functionality. It has been shown how the approach could integrate with some existing powerful tools to generate diagrams. ILOG is a good example of such tool.

This chapter has demonstrated the applicability of the GraSSML conceptual architecture through the GraSSML prototype for:

- the hierarchical class of diagram, more specifically Organisation charts.
- the process class of diagram, more specifically UML activity diagrams.

It has also demonstrated what would be required to apply the approach to other hierarchical diagrams or process diagrams in different application domains.

All the use cases presented in this chapter involved the creation of an ontology or minor amendments of a pre-existing ontology which involved a certain amount of time and effort. As mentioned in Chapter 4 section 4.4.1, the possible use of pre-existing ontologies is an important aspect and one of the key benefit of using the GraSSML approach.

The following chapter presents a different use case which takes advantage of this benefit. The implementation of a different class of diagram “Charts” (financial charts) have also been developed but by a third party. Indeed an external user of the system has assessed the viability of the approach by extending the system for a different class of diagrams.

Chapter 6

Evaluation: An Author's Perspective

Olivier BRAECKMAN, an MSc Student, independently, applied the GraSSML approach to a different class of diagrams “charts”, in the finance application-domain (“financial charts”). Examples of this class of diagram are Column charts, Bar charts, Pie charts, Line charts and Scatter charts.

Having a computing and business background, he defined a research project aimed at developing accessible financial reports.

The first step of his project involved some research on both financial reporting practices and existing graphic accessibility approaches. It was decided at this stage to introduce him to the GraSSML conceptual architecture by explaining to him the ideas behind the approach and demonstrating the working prototype developed for process and hierarchical diagrams.

Even though, Olivier BRAECKMAN's research project aimed at exploring the possible enhanced accessibility features GraSSML could offer in order to produce accessible charts, this evaluation is intended to assess the viability and applicability of the GraSSML approach rather than evaluating the accessibility of the charts obtained.

The principle aim of this evaluation was to assess the understanding, viability and the applicability of the GraSSML approach to a different class of diagrams (in this case “charts”) in a different domain (“financial reporting”) by a third party. It assesses whether the methodology is sound and reliable enough and well enough defined for somebody else to apply it with minimum intervention and it also shows what is involved for somebody else applying it.

The application of the methodology has been tested by Olivier BRAECKMAN during a double blind trial in which he, alone, performed the data gathering, processing and recording as well as an initial analysis. After which the results and data obtained were analysed, interpreted and evaluated for the purpose of the thesis.

After two initial sessions, of four hours each, explaining and demonstrating the approach, Olivier was left to reflect on his project and the possible approaches which could be applied to address them. Olivier decided to use GraSSML to create accessible financial reports especially accessible charts. This chapter describes the

evaluation that was carried out and the main points concerning the GraSSML Financial Chart Project (GraSSML FCP) authoring process.

6.1 Introduction

6.1.1 The GraSSML FCP project

For a full description of the project, see (BRAECKMAN, 2008).

The initial analysis has been carried out by Olivier.

A. Financial reporting practices and standardisation

Financial reporting, which is a formal report of a business financial activity provides essential information to assess the health of a business.

It is used by various users for different purposes such as investors who want to make sure of the security of the investment, banks who want to insure loans could be repaid, tax authorities to check if declarations are truthful, employees to check the situation of the business they work for, etc. The information is presented in a structured manner using four basic elements: Balance sheet, Income statement, Statement of retained earnings and Statement of cash flows.

Over time each country has developed their own practices, making the comparison of international companies a tricky task as it involved a certain amount of guidelines and rules (Generally Accepted Accounting Principles (GAAP)) to make it possible. It is in that context that the International Accounting Standards Board (IASB) triggered the development of the International Financial Reporting Standards (IFRS) which have been adopted by many countries (e.g. USA, EU, and Canada).

Such standards are seen as an opportunity for the development of generic tools to generate and communicate financial information. It is in that perspective that since 1998 XBRL (eXtensible Business Reporting Language) (XBRL, 2007) which is an open data standard for financial reporting, has been developed by an international consortium. Its aim is to create financial reports which are computer-understandable and in which the information can be accessed and used 'intelligently' (recognised, selected, analysed, stored and exchanged).

Financial charts are an integral part of these reports and are an important aspect of such reports. They are usually generated using one of the following mostly used software: SPSS, Microsoft Excel or OpenCalc.

These charts are seen as an important part of such reports for reasons that have already been acknowledged in Chapter 2. Even though it is acknowledged that these charts cannot contain all the information available in financial reports, they play an important role in conveying important information in an understandable way, allowing non specialised persons to gain a minimum access to the information.

These financial reports aim at being an important source of information for various persons in various situations and for various reasons. Therefore the information they present should be relevant, reliable, understandable and most importantly accessible as it can be accessed by users having different needs and preferences (e.g. language, device, situation, disability, etc.).

B. Accessibility issues

Currently financial reports are mostly presented using two formats: HTML and PDF. Although these formats offer accessibility features, they can be accessible only if authored with accessibility in mind. In financial reporting different accessibility issues can be identified, but these are not proper to financial reporting but are linked to the authoring of documents in different formats such as PDF or HTML, the acknowledgment and respect of the available accessibility guidelines and the considerations given to user's preferences and needs (language, disability, devices used, etc.). The accessibility of financial charts which is a type of diagram has been presented and discussed at length in the Chapter 3.

6.1.2 Objectives of the GraSSML FCP project

The first stage of this project involved some research in the exploration and analysis of different existing approaches which could improve the accessibility of financial reports. Following this analysis, Olivier has decided to apply the GraSSML approach to the identified problem that is the accessibility of well-defined financial charts in financial reporting.

Olivier choose to apply GraSSML to financial reports in order to represent “financial charts” in a way that it is possible to reason and enquire over the information they carry thus making them more perceivable, operable and understandable and, as a result, enhancing their accessibility.

The objective of the project was an attempt to improve the financial reporting practises by creating automated ways of generating and processing the financial information available in such reports respecting the existing standards. A “proof-of-

concept” prototype applying the GraSSML conceptual architecture into the generation of the most commonly used chart in financial reporting was achieved. The methodology followed is presented in the following section.

Due to the complexity of the IFRS which requires a vast amount of knowledge to make fully accessible financial reports, it was decided by Olivier to focus on three commonly used types of charts: pie charts, columns chart and bar charts, all generated based on a single sample balance sheet expressed in XBRL.

6.2 Evaluation methodology

Different steps were involved in the evaluation process for the applicability of the GraSSML approach to financial charts. Regular meetings were set up with a view to observing Olivier’s understanding and applicability of the GraSSML approach.

Before starting the evaluation, Olivier researched existing approaches into making graphics accessible (iGraph, TeDUB, SVG Linearization and GraSSML, all of these have been presented in Chapter 3 section 3.2.2).

It was important at this stage, for the success of the evaluation, to make sure that Olivier had an understanding of the main ideas behind the GraSSML approach, its applicability context and its limitations. To achieve this, an interview was carried out. Olivier was asked about the different approaches researched and their differences with GraSSML. It was noticed that some clarification was needed during the discussion.

From the interview it was judged that Olivier had gained an understanding of the main ideas behind GraSSML. To illustrate his understanding of GraSSML, he made a remark related on his understanding of XBRL: “The idea behind GraSSML is similar to the one behind XBRL, instead of treating graphical information as a set of pixels or geometric objects; GraSSML identifies each of the concepts represented and their relationships to each other. This information is made available so that a computer can make sense of it and perform different actions such as search, inferences, etc. and then generate an appropriate representation of it in a certain modality”.

Even though the main ideas were grasped by Olivier, he was still not very sure about the roles of all the different levels of abstractions (semantic, structure and presentation), especially the structure level. He was also worried about the applicability of GraSSML to financial chart accessibility and the selection of appropriate web technologies to implement these levels. But he was reassured and

introduced to the next step of the evaluation which involved the applicability of GraSSML in different stages: the GraSSML conceptual architecture (Figure 12) and as well as the GraSSML system architecture (Figure 14).

A. “Initiation” Study of the field of interest

The first stage involved a study of the field the GraSSML approach applied to, in this case financial reporting. Indeed, a certain amount of information is needed in order for GraSSML to handle a new class of diagram in this specific field.

Having identified the class of diagram “charts”, this “initiation” phase aims at collecting essential information to be used to formulate the four aspects needed to apply GraSSML:

1. The Ontology
2. Notational conventions
3. Verbalisation model Templates
4. Query definitions

A certain amount of domain knowledge is essential, in particular for the choice of minimal information needed (“primary resources”), choice of the appropriate vocabulary, choice of the most commonly needed queries (not exhaustive just needed to check the reliability of the ontology formulation).

B. Applying GraSSML

After the initiation stage, the process of applying the GraSSML approach to financial reporting took place. This stage involved the different stages presented in Chapter 4 which were presented to Olivier in the form of a document accompanied by its graphical representation. It was mentioned at this stage that the basic model of GraSSML does not depend on specific technologies and that this choice is open to any modification. The seven stages that have already been described Chapter 5 are summarized as follows:

- Stage 1: Ontology
- Stage 2: Data model
- Stage 3: Notational conventions
- Stage 4: Graphical representation
- Stage 5: Verbalisation model (Semantics)
- Stage 6: Verbalisation model (Structure)
- Stage 7: Query systems

6.3 GraSSML FCP

6.3.1 Class of diagrams: “Charts”

This class of diagram mainly encodes quantitative information. Charts describe data using symbols (bars, slices, lines, points, etc.) and text (symbols label, title, axis labels, legends, etc.). Different types of charts exist, each providing a different way of representing a set of data (chart: bar, pie, column, line, scatter chart). Depending on the type of chart used according to the type of data to be represented, a certain amount of meaningful information can be extracted as some kinds of charts are more useful for presenting a given set of data than others. For the context of this project the three most commonly used charts used in financial reporting were selected: bar (Figure 72), column (Figure 73) and pie (Figure 74) charts.

There exist different variants of bar/column chart and pie chart such as clustered column chart, stacked bar chart, exploded 3D pie chart, Pie of pie chart, etc. Many other type of charts are part of this class of diagram such a line charts, scatter charts, bubble charts (which are a type a scatter chart), etc.

A. Bar / Column charts

Bar charts and column charts are charts in which data values are represented using bars lengths which are proportional to the represented value. They differ in terms of notational conventions, in a bar chart, the bars are presented horizontally and in a column chart the bars are presented vertically. They are often used to compare relative quantities over a certain period of time or representing different categories. Text and colour can be used to add information to them.

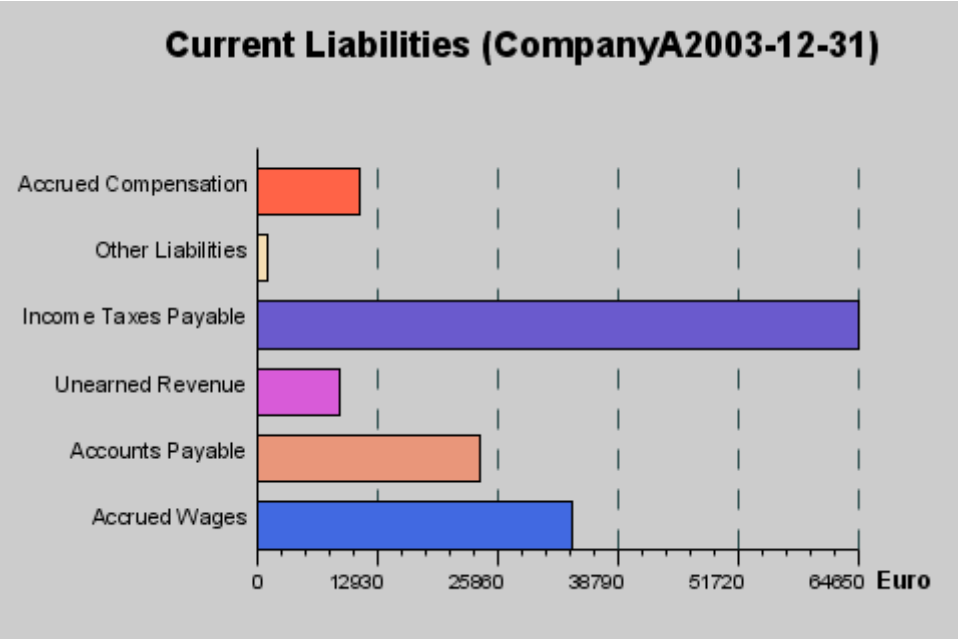


Figure 72: Bar Chart Example

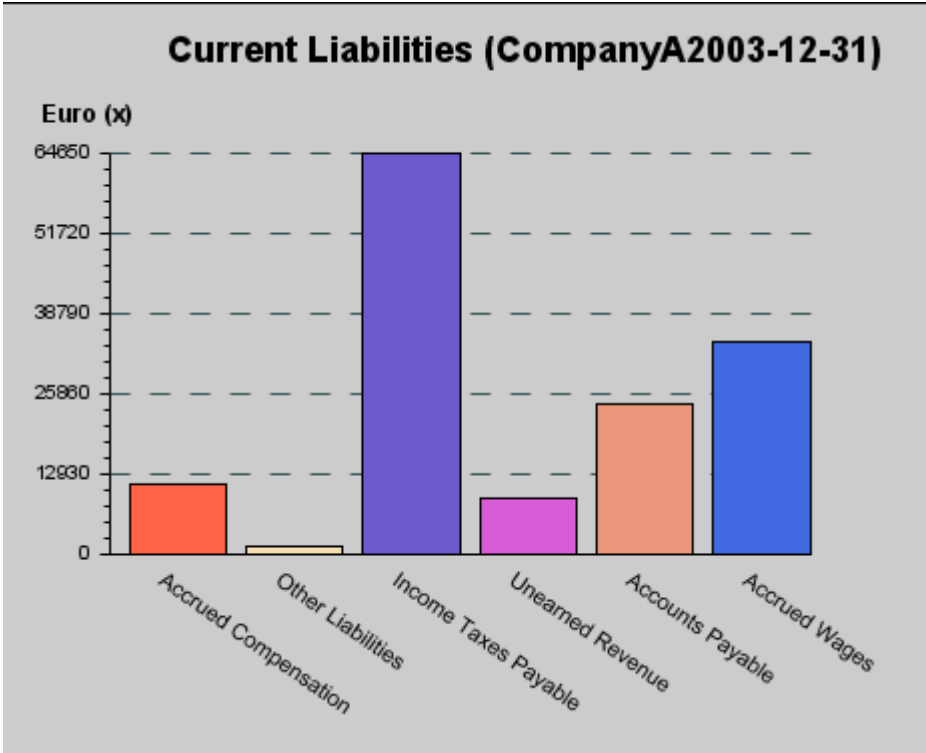


Figure 73: Column Chart Example

B. Pie charts

Pie charts are used to illustrate percentages represented as sectors. Each sector is proportional to the quantity it represents out of the total quantity represented by the

entire pie. Text can be used to label the sections, provide legends and a title. Different colours can be used to differentiate the sections.

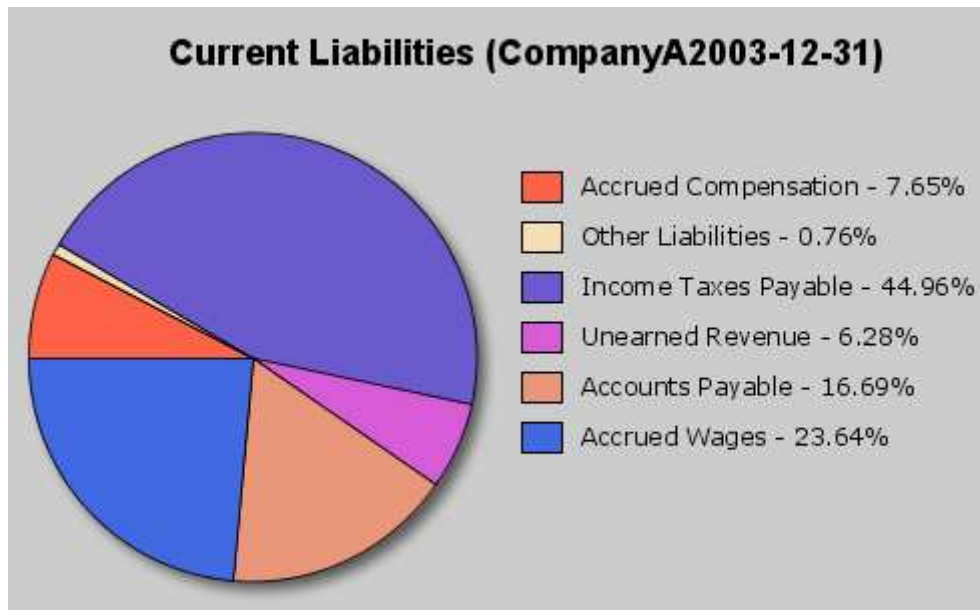


Figure 74: Pie Chart Example

6.3.2 The GraSSML FCP approach

Figure 75 shows the GraSSML FCP conceptual architecture derived from the GraSSML conceptual architecture (Figure 12). At the structure level FGML (Financial Graphics Markup Language), based on ZineML, has been introduced. This XML based language is based on the ideas behind the ZineML language presented in Chapter 4 aiming at capturing the structure behind the diagrams.

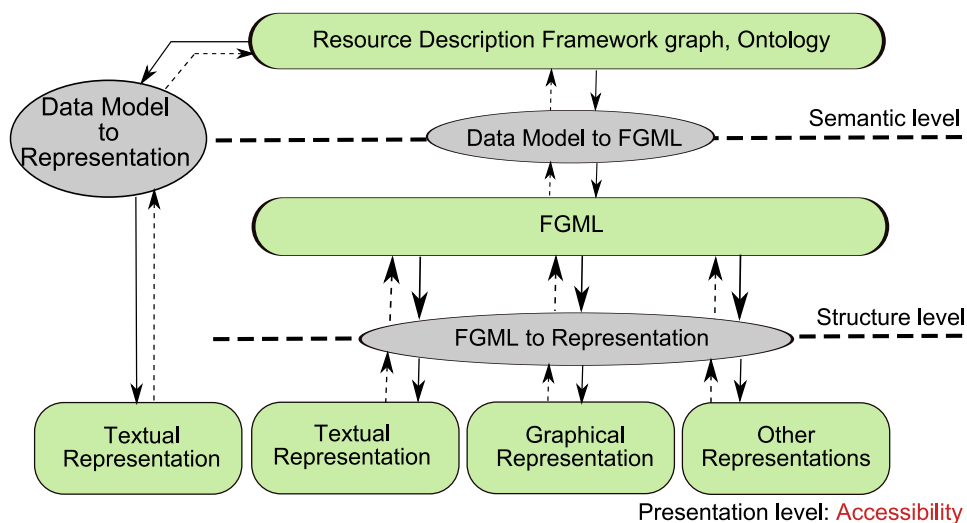


Figure 75: The GraSSML FCP Conceptual Architecture

Accessible financial charts were generated based on information retrieved from a created sample of a Balance sheet (Figure 76) expressed using the XBRL open standard. The decision to use XBRL was based on its anticipated success in financial reporting but the source of the information could have also been extracted from any other properly defined language.

A study and an understanding of the XBRL report taxonomy were essential for the creation of this balance sheet sample. The possibilities of using such a standard within the GraSSML approach demonstrates the interoperability advantages of XBRL as well as the flexibility GraSSML offers. From this study of XBRL a basic XBRL taxonomy tailored to the needs of the project has been created using XML Schema. The sample balance sheet instance document was then created using this taxonomy.

Balance Sheet - Company A Unit Ref: EUR	2002	2003	2004
Assets	376 384	337 384	261 834
Current Assets	78 784	72 784	52 500
Cash	30 000	29 000	20 000
Investments	34 000	33 000	28 000
Inventories	8 000	7 000	8 000
Accounts Receivable	3 100	2 100	2 000
Pre-Paid Expenses	1 684	684	1 000
Other	2 000	1 000	500
Fixed Assets	296 500	264 500	201 684
Property And Equipment	254 000	225 000	165 000
Leasehold Improvements	3 500	2 500	3 000
Equity And Other Improvements	34 500	33 500	30 500
Accumulated Depreciation	4 500	3 500	3 184
Other Assets	1 100	100	650
Goodwill	1 100	100	650
Liabilities And Stockholders Equity	376 384	337 384	261 834
Current Liabilities	149 780	143 780	120 900
Accounts Payable	25 000	24 000	20 000
Accrued Wages	35 000	34 000	30 000
Accrued Compensation	12 000	11 000	10 000
Income Taxes Payable	65 650	64 650	50 000
Unearned Revenue	8 030	9 030	10 000
Other	4 100	1 100	900
Long Term Liabilities	51 604	29 604	29 500
Mortgage Payable	51 604	29 604	29 500
Owners Equity	175 000	164 000	111 434
Investment Capital	165 000	163 000	100 000
Accumulated Retained Earnings	10 000	1 000	11 434

Figure 76: Balance Sheet Example Used

Figure 77 shows the GraSSML FCP system architecture presenting the specific web technologies selected by Olivier at the different stages of the application. Note that stage 6 (section 6.2B) which relates the verbalisation model of the structure of the diagram was not achieved during the evaluation.

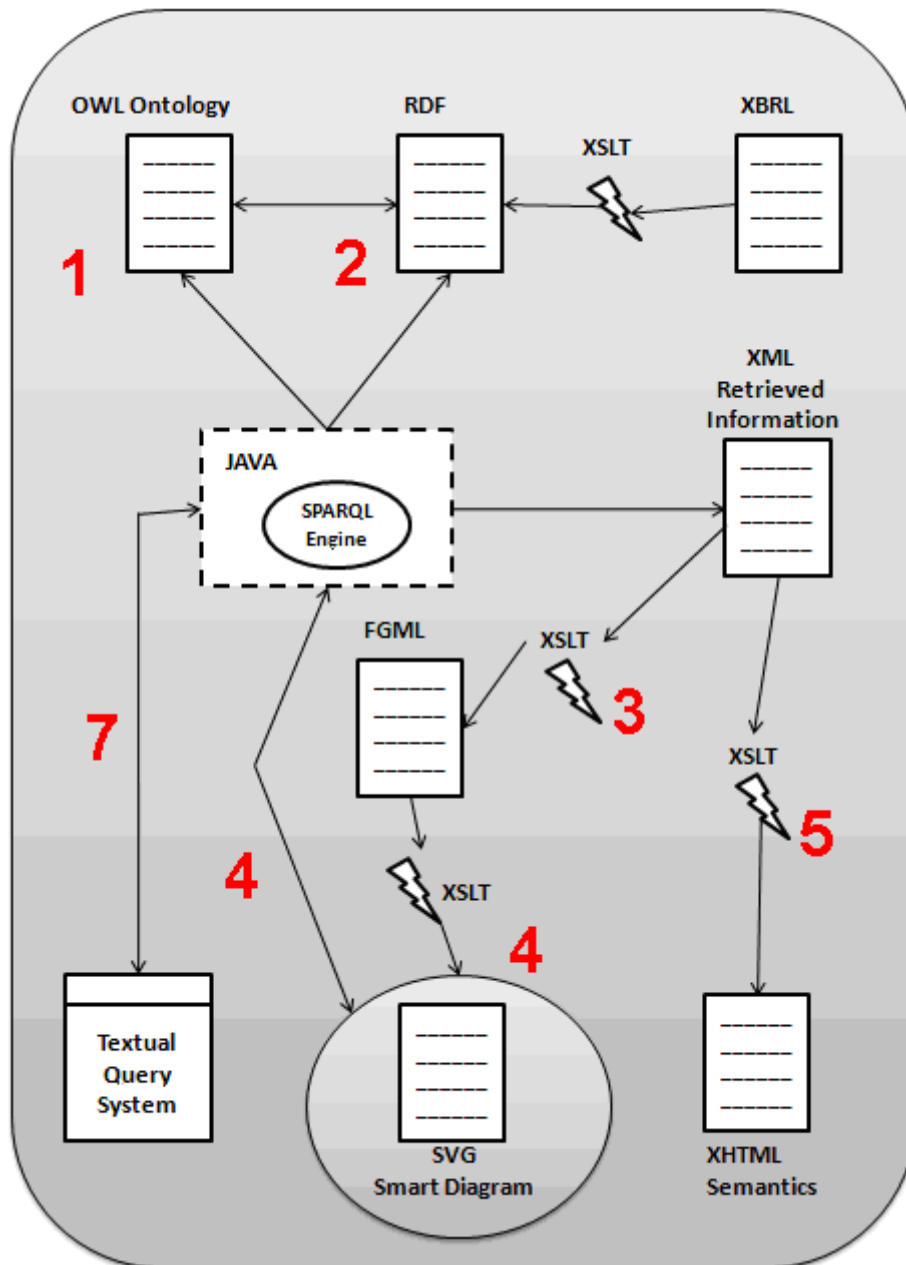


Figure 77: The GraSSML FCP System Architecture

6.3.3 Semantic Level

A. Stage 1: Ontology

This stage was a challenge for Olivier as it was the first time he had to create an ontology. The paper entitled “Ontology Development 101: A Guide to Creating Your First Ontology” (NOY and McGUINNESS, 2000) was given to Olivier for guidance. He found it very useful in the process of creating his ontology for financial reporting. The study of the XBRL taxonomy as well as the understanding of the paper provided allowed Olivier to proceed with the creation of its ontology using the ontology editor Protégé.

The concepts and properties of these concepts required for the creation of the ontology were identified and organised appropriately into classes and subclasses (Figure 80). Apart from the concepts expressed on the previously defined taxonomy extra concepts (context, unit and creation) were added to the ontology.

Figure 78 shows the different datatype properties defined and Figure 79 the different Object properties.

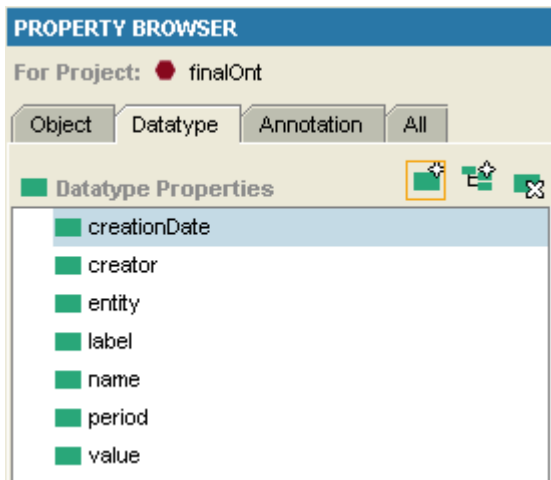


Figure 78: Datatype properties of GraSSML FCP Ontology

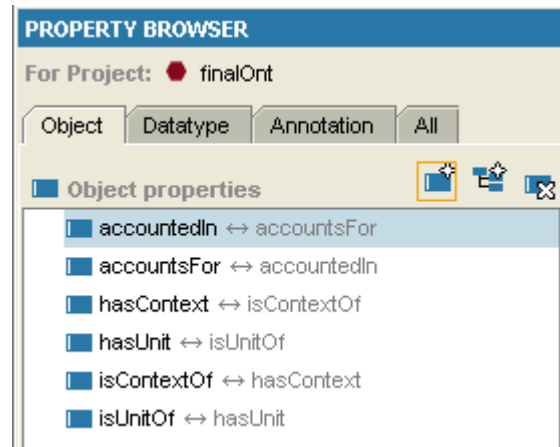


Figure 79: Object properties of the GraSSML FCP Ontology

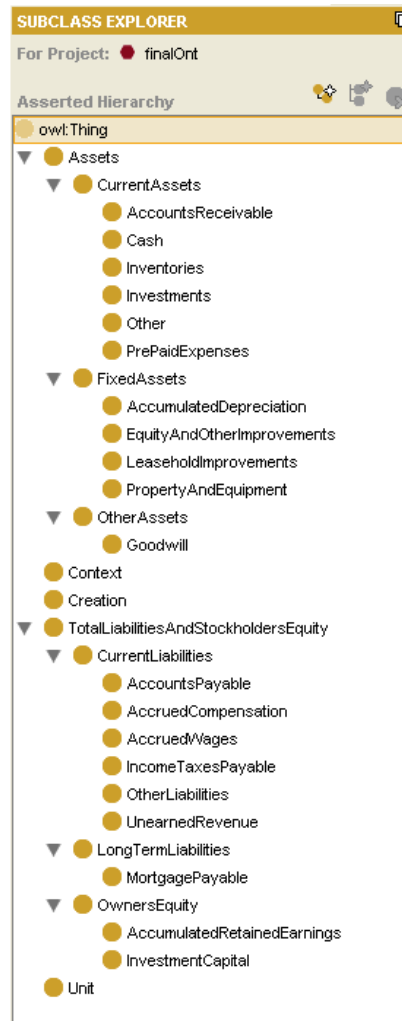


Figure 80: Hierarchy of the Classes of the GraSSML FCP Ontology

B. Stage 2: Data model

At this stage the whole financial report was encoded into the data model making the semantics of the whole report, and not only the charts' semantics, accessible. The information contained in the financial report was extracted from its XBRL representation using an XSLT transformation. The data model obtained respected the defined ontology. A data model holding only the semantics of the chart to be generated was created. It is a sub-data model of the data model. This sub-data model was very important in the GraSSML approach. Its content is relied upon at the structure and presentation level of the GraSSML conceptual architecture for the generation of one of its possible representations.

The author is required to provide information on what concepts in which context he wants to represent in a specified type of chart (pie chart, bar chart, etc.). The author is assisted in the process as all the possible options extracted from the data

model are presented to him, avoiding in this way any mistakes. Using the Jena API and the SPARQL engine, all the appropriate information is retrieved from the data model into the sub-data model which now represents the semantics of the information to be contained in the selected type of chart.

6.3.4 Structure level

A. Financial Graphics Markup Language (FGML)

At this stage, Olivier found it hard to understand the role of this level, arguing that it would be possible to generate the presentation level from the semantic level.

With the existence of this level clarified and understood, the creation of an intermediate ZineML-like structure level language, FGML, was easy. In order to define FGML, Olivier researched and analysed the core structure components of the charts to be studied (column/bar chart and pie charts in this case). Using the column chart as an example, the process of creating FGML was as follows:

First the column chart was decomposed and its basic elements identified. The decomposition is illustrated in Figure 81.

- The axis: horizontal (1) and vertical (2)
- Label: axis can be labelled (3)
- Scale to the vertical axis (5)
- Segments or bars (4)

More complex elements such as segments composed of sub-segments are possible in a stacked column chart for example. But even though FGML could be extended to express these alternatives this was not treated in the project.

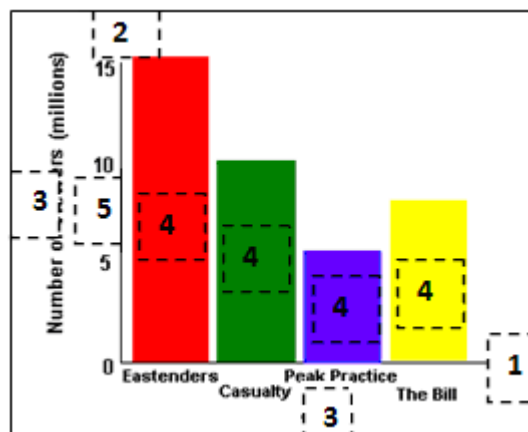


Figure 81: Column chart structure element

B. Stage 3: Notational conventions

The notational conventions allow the generation of FGML from the sub-data model. An XSLT Transformation for each type of chart has been used to apply these notational conventions to the extracted sub-data model.

At this level Olivier started realizing the different options offered by the GraSSML conceptual architecture. Having read about XML languages, he suggested the possible integration of other existing structure languages such as GraphXML. This aspect has been discussed in Chapter 4 section 4.5.

The following code sample (Figure 82) shows the core components express in FGML of the example presented in Figure 83. This example shows the evolution of the cash from 2002 till 2004. The attribute “segmentSeparator” indicates that each segment will be separated by 10 pixels. The value of the segment width is automatically calculated taking into account the total number of segments in the chart.

```
<chart type="Column" title="Evolution of the Cash">
  <axisX segmentSeparator="10" length="350" />
  <axisY minVal="0" maxVal="30000" unit="Euro" length="200" />
  <segments>
    <segment id="Cash_CompanyA2002-12-31" stroke="black">
      <name>Cash</name>
      <value>30000</value>
    </segment>
    <segment id="Cash_CompanyA2003-12-31" stroke="black">
      <name>Cash</name>
      <value>29000</value>
    </segment>
    <segment id="Cash_CompanyA2004-12-31" stroke="black">
      <name>Cash</name>
      <value>20000</value>
    </segment>
  </segments>
</chart>
```

Figure 82: FGML code sample

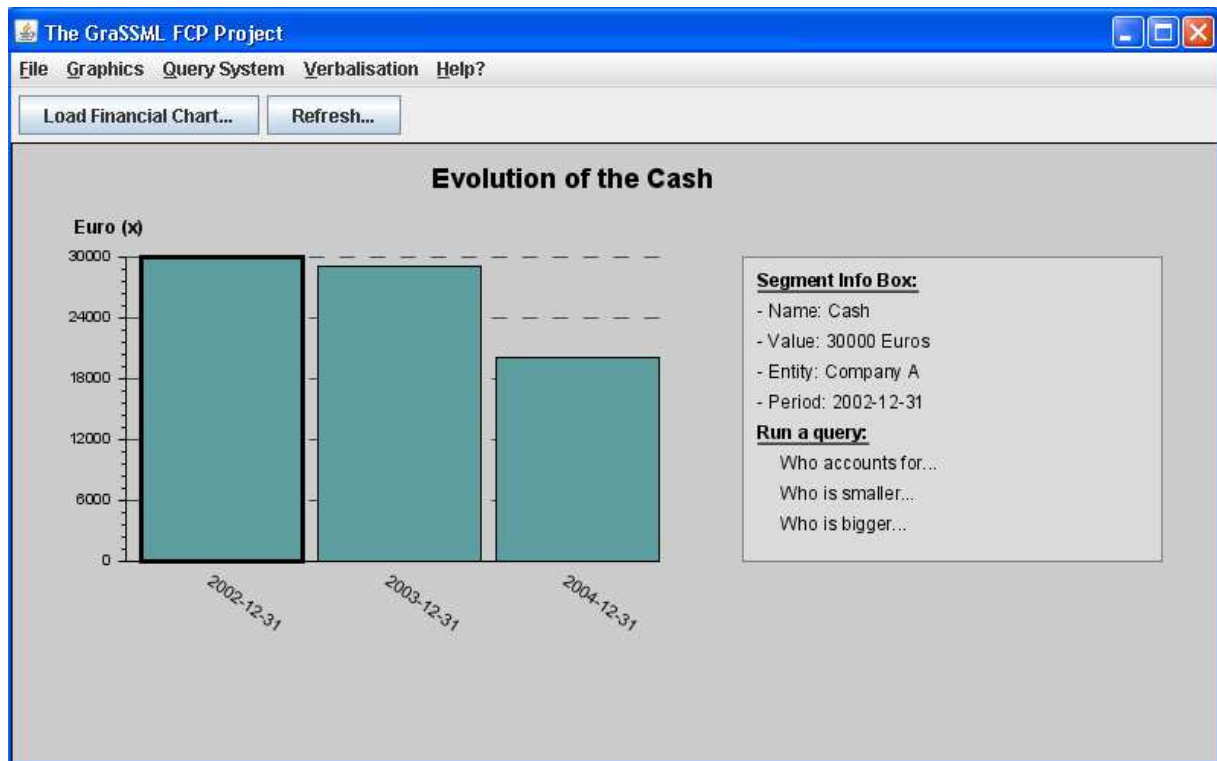


Figure 83: FGML “Evolution of the Cash” example

6.3.5 Presentation level

It is at this level that Olivier realised the various possibilities offered by GraSSML in terms of presentations. Three possible presentations were implemented: the textual presentation of the semantics of the chart, a textual query system and an interactive graphical presentation “smart diagram”. Before any of these presentations could be generated, the data model of the financial report and the sub-data model of the wanted chart were generated using the main graphical interface. All the transformations leading to the appropriate presentation depend on the information contained in these data models.

A. Stage 5: Verbalisation model (Semantics)

Verbalisation model templates allowing the generation of such description for each type of chart were developed using an XSLT Transformation. The generic template allowed the description of a specific chart using commonly used sentences to be built dynamically. After researching the literature on how to read, understand and describe the selected charts, Olivier came up with a set of conventions on how to describe them. Based on description guidelines from the Open University (TAYLOR,

2008a) different steps were identified for the description of column charts and pie charts.

A column chart description should involve the following steps:

- Identify the context of the information (title, scale, unit, years, etc.)
- Identify the number of segments in the graphic
- Identify each column and describe it as much as possible (label, value, etc.)
- Analyse and compare the segments (i.e. identify the highest and lowest segments)

A pie chart description should involve the following steps:

- Read the title to find what the proportions are about
- Identify the number of slices in the pie
- Identify each segment and describe it as much as possible (slice name, value, etc.)
- Analyse and compare the slices (i.e. identify the highest and lowest slices)

Having accessibility in mind the XHTML textual description generated took into account accessibility guidelines on their authoring.

As well as a sequential textual description (Figure 85) of the information contained in these charts, a tabular representation (Figure 86) of the information they contain as been achieved. This involves the creation of a table containing the name and value of each segment or slice. Appropriate accessibility guidelines have been followed by Olivier.

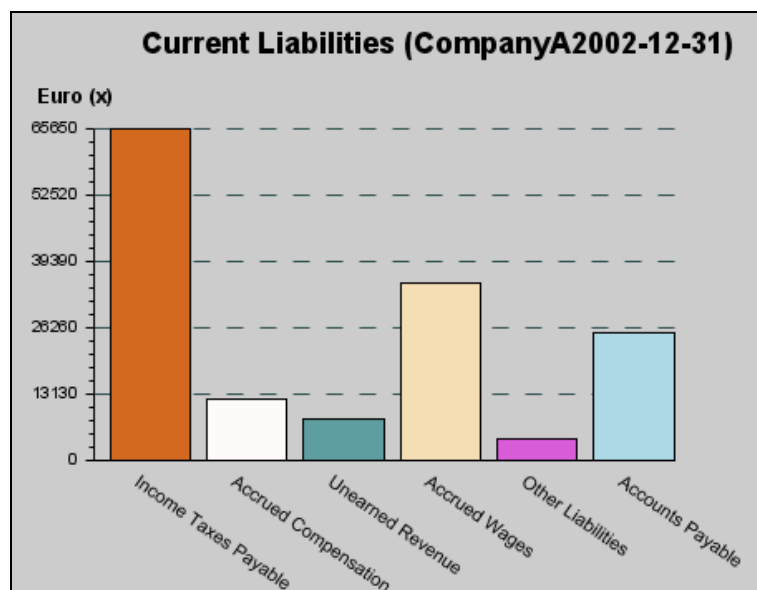


Figure 84: Column Chart Generated using GraSSML FCP

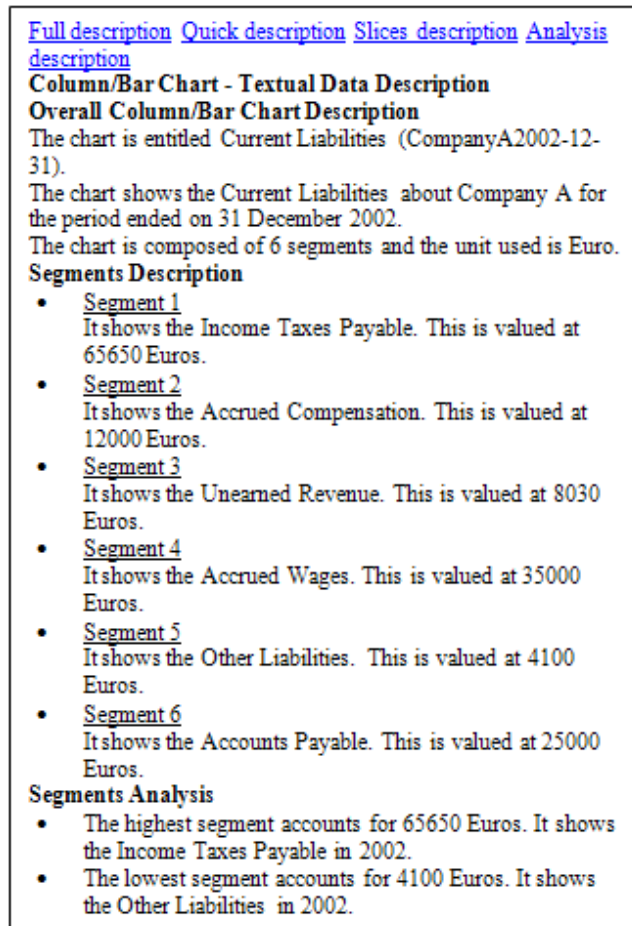


Figure 85: Textual presentation of Figure 84

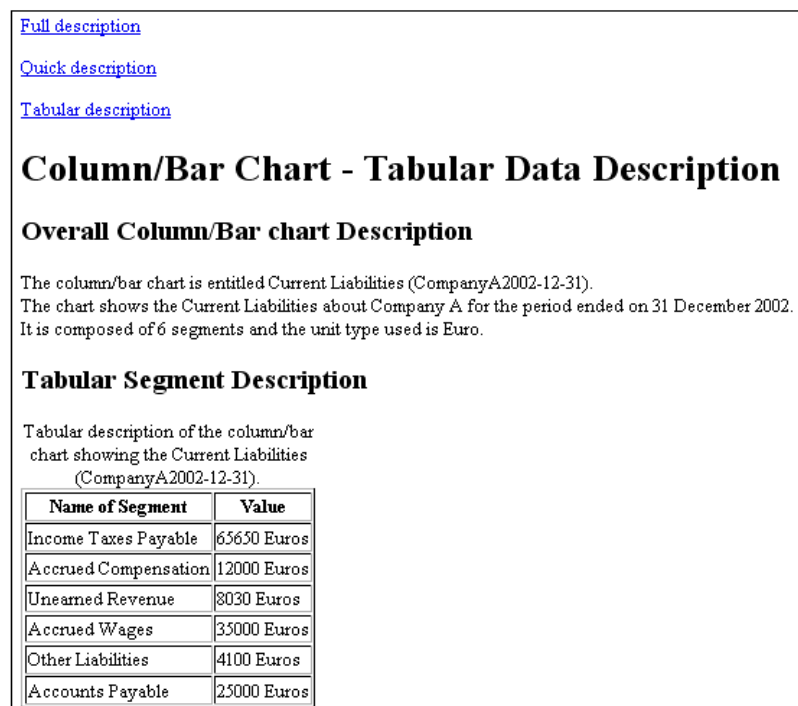


Figure 86: Textual presentation of Figure 84 (Tabular version)

B. Stage 7: Textual Query System

Having access to the ontology and the data model representing the financial report, it was possible to develop a textual query system allowing the formulation of general or specific queries. Dependent on the limits of the ontology reasoner and the presence of the appropriate information a certain amount of queries could be implemented. In an ideal situation the development of a SPARQL end point would allow any user of the financial report to formulate their own queries specific to their needs but due to time limits and for the purposes of the research project a limited set of queries were implemented:

Queries aiming at providing general information about the chart:

- Find the Balance Sheet Creation Details
- Find the Balance Sheet Unit
- Find the different Balance Sheet contexts

An accessible Java graphical interface was implemented to access all available queries. With the use of the AccessBridge, mnemonics and keyboard shortcut, the graphical interface became accessible to most users using screen readers.

Other, more specific, queries were implemented. Some made use of the advantages the ontology reasoner provided in making “inferences”.

The example presented in Figure 87 shows that “the Assets accounts for Current Assets which itself accounts for the Cash concept”. A transitive “accountFor” property has been defined in the data model. An ontology reasoner using this transitive property definition would be able to infer that Current Assets accounts for the Cash concept but also that Assets indirectly accounts for the Cash concept.

Assets	376 384
Current Assets	78 784
Cash	30 000

Figure 87: Balance Sheet Assets Subset Example

Olivier, not being a domain expert, found it quite difficult to select an initial set of queries to implement. But once selected, he found the process of implementing them quite easy.

The implementation of the accessible graphical interface was a challenge for him but with some research and with the help of a screen reader (Jaws evaluation version) to check the result produced, he succeeded in creating it.

C. Stage 4 and 7: Graphical Query system “Smart Diagram”

The “Smart diagram” feature was the most exciting one for Olivier. Being able to query the graphical presentation of the chart and gaining graphical or textual feedback from it was for him an advanced feature that he had never encountered in previous studies.

At this stage, due to the time limit involved, some propositions on how this stage could be implemented have been presented to Olivier. However, he was still asked to implement it in the way he thought would be appropriate.

The first step (**Stage 4**) involved the generation of the graphical representation using SVG. Olivier was aware of the different possibilities for selecting the information needed to generate an appropriate chart from the data model of the financial report. The solution adopted might not be the best one, as it does not take into account the best graphical representation for the selected data and the tasks planned, but rather the user selected chart for this selected data. But at this stage the aim was to demonstrate the idea and not to create “effective” charts (GURR, 1999) as addressed by previously presented researched such as APT (MACKINLAY, 1986), BOZ (CASNER, 1991) or AVE (GOLOVCHINSKY et al., 1995).

Using the implemented accessible graphical user interface, the generation of a bar/ column chart or pie chart is described as follows by Olivier.

1. Choose a graphic type to generate
2. Choose the purpose of the graphic. This can either be
 - a. Trends of a given concept in the available contexts, or
 - b. Comparison of several concepts in a given context.
3. Generate the sub-data model
4. Apply the Notational conventions from sub data model to FGML
5. Apply the transformation from FGML to SVG
6. Load the graphic in the interface

The transformation from FGML to SVG was done using a generic XSLT transformation taking into account the data present in the FGML document and the nature of the chart to be generated (pie, column or bar chart).

Once the graphical representation generated, different technologies such as Java, JavaScript, Jena SPARQL and Batik are used to allow the different elements of the graphical presentation expressed in SVG to be queried directly, interactively, making reference to the semantic of the chart expressed in the data model.

The example illustrated in Figure 88 shows an information area, on the right of the smart pie chart containing a list of predefined queries, which is generated once an element is clicked on.

Depending on the selected element of the pie chart, a set of information corresponding to this element is provided. This information results from basic queries on the element selected such as its name, value, entity, and period. All this information is automatically extracted from the data model. This provides an overview of the information concerning the selected element.

The information behind the diagram being totally accessible it is possible to perform various queries to provide information or make comparisons.

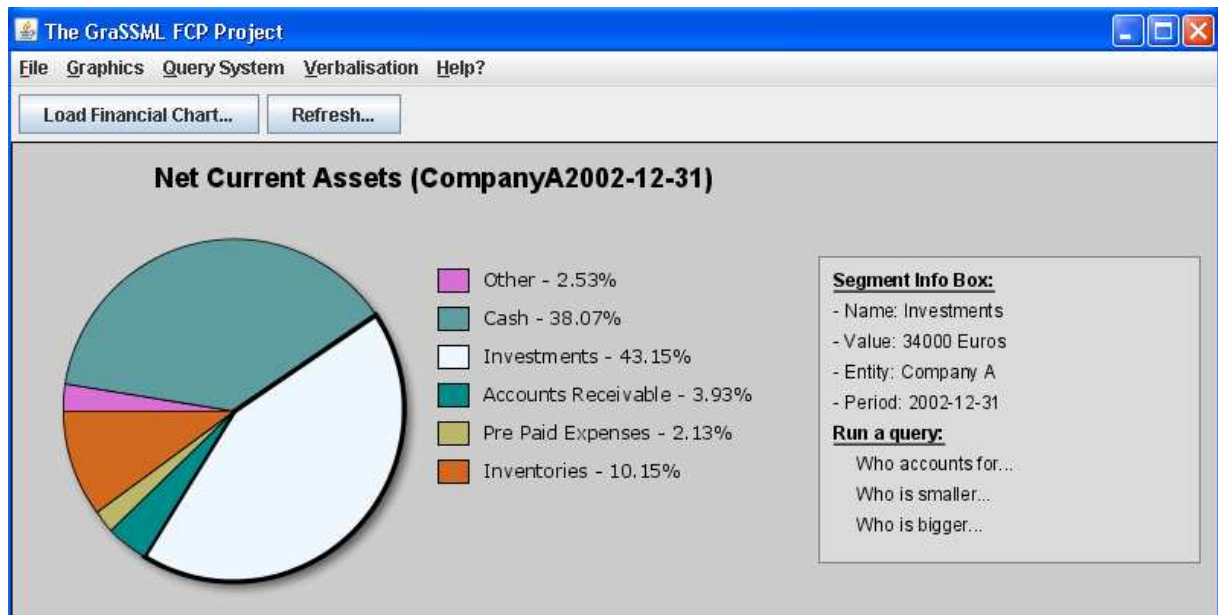


Figure 88: smart Pie chart “Net Current Assets for Company A in 2002” example

The query “Who is smaller” than the selected slice in the Pie chart of Figure 89, returns two results. Both slices have been highlighted by changing their colour.

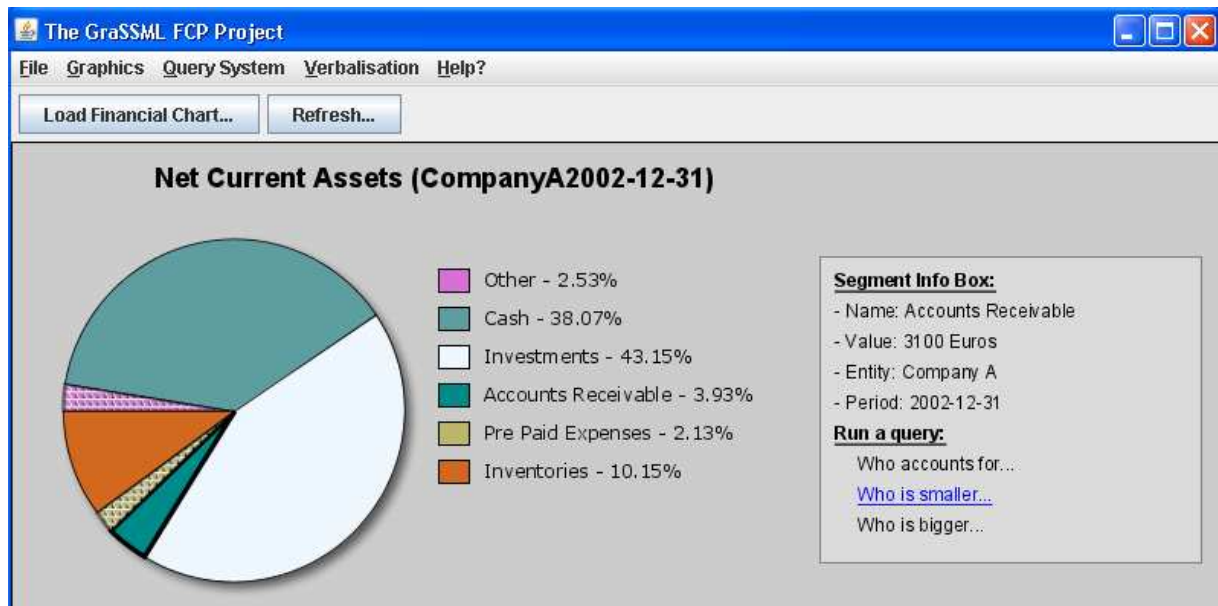


Figure 89: Smart Pie Chart “Who is smaller...” query

The inference query discussed previously “Who accounts for...” could also be run by selecting a slice and clicking on the appropriate query on the information area.

The result showing the concepts accounting for the concept selected (in this case “Account Receivable”) is returned in an accessible messageBox Figure 90.



Figure 90: Smart Pie Chart “Who accounts for...”

6.4 Results

Olivier carried out different technical testing to evaluate the implementation side of the project. A set of specific test cases carefully selected to evaluate the different part of the prototype have been successfully carried out.

Olivier thought about carrying out a user evaluation but due to limited resources and the time at hand he decided that it should be considered in future work.

The original aim of this evaluation was to evaluate the applicability of the GraSSML conceptual approach to a different class of diagram in a different field by a

third party and to demonstrate the ability GraSSML offers in inquiring and reasoning about the information on which a diagram is based.

The evaluation was considered a success as the result obtained was as expected. It was interesting to evaluate the degree of difficulty felt by Olivier during the different stages of the application. Different feedbacks were collected from Olivier, each concerning different aspects of the project. The interviews conducted at the different stages included some of the following questions: Was it difficult? How long did it take you to understand and apply GraSSML? Which part was the most difficult? What would you have changed? Do you think it could be used? Do you think you would be ok creating Graphs this way? How did you think about the instructions on the application of the methodology? Do you have any further comment?

Olivier realised that financial expert knowledge was essential in the creation of such an approach. Such knowledge would greatly influence the creation of the ontology, the type of charts generated depending on the data to be represented and the tasks intended for, the way the information is presented, and also the design of an appropriate end-point allowing a potential user to express queries.

The complexity of the IFRS taxonomy was such that a decision has to be made in simplifying it into a subset based on a simple example. At this level, expert knowledge is essential.

At the beginning of the evaluation project Olivier expressed his apprehension on the application of GraSSML. Some of the levels expressed in GraSSML were not understood and some features expected such as the “smart diagrams” seemed too complex to be achieved in the time at hand. It was difficult for Olivier to see “charts” from a different perspective, thinking of its creation in a completely different way he is used to.

The following suggestions have been made by Olivier:

- ❖ **Implementation of new type of charts:** line charts, scatter chart or different possible variant such as 3d charts, split pies, etc.
- ❖ **Involve financial experts:** to allow a better understanding of the requirements and/or needs of the potential system users.
- ❖ **Ontology generation:** In a future development a transformation from XBRL to automatically generate the ontology could be considered.
- ❖ **Built-in speech interface:** accessibility could be improved by integrating a “built-in” speech interface removing the need of a screen reader.

- ❖ **Graphical presentation improvement:**
 - To reduce the amount of visual processing required understanding and analysing a financial chart.
 - In terms of implementation of the graphical presentation some improvement could be achieved in term of compatibility as the actual solution provided was more to prove the feasibility of the approach rather than taking into account such aspects.
- ❖ **Smart diagram capabilities:** Taking into account results on studies to reduce the amount of visual processing required understanding and analysing a financial chart, results of the “smart diagram” could be improved (e.g. reordering resulting columns).

6.5 Conclusion

This evaluation provided information on the applicability of the GraSSML conceptual architecture. The viability of the GraSSML approach was successfully assessed by Olivier who extended the system for a different class of diagrams.

Even though Olivier started off the project with reservations, by the end he fully understood the system enabling him to successfully apply GraSSML in the financial reporting domain. The result obtained was quite impressive as Olivier experimented. The number of propositions he made to improve the implemented prototype were various, demonstrating a full understanding and enthusiasm for the approach. Olivier set up the approach in another domain by following a set of stages presented to him step by step. Even though Olivier applied the approach successfully an initiation to the project was needed. The methodology should be reviewed to make it well defined enough for someone to apply it with minimum intervention. The analysis of this evaluation could be used to compile a formal document into the application of GraSSML into another domain for a different class of diagram. Olivier expressed a willingness to develop the system further especially at the presentation level where different flexible and extensible presentation modalities are possible.

This project has also demonstrated the flexibility of the conceptual architecture as argued in Chapter 4. The origin of the information populating the Ontology and the data model (on which GraSSML approach relies) can be multiple. In this specific case the information came from the “XBRL taxonomy” and the “XBRL document” expressing the financial report.

Chapter 7

Evaluation: A User's Perspective

GraSSML aims to contribute to the representation of diagrams in a way that could improve the ability to enquire and reason about the information on which the diagram is based and thus making diagrams more perceivable, operable and understandable and, as a result, suggesting enhanced accessibility benefits for such diagrams.

The technical perspective evaluation and the author's perspective evaluation demonstrated the feasibility, viability and applicability of the GraSSML approach but did not provide any insight on whether GraSSML supports the POU(R) principles of the WCAG 2.0 (see Chapter 3 section 3.1.2) for diagrams.

This chapter describes an evaluation which was undertaken with two blind participants. This study was conducted to assess the perceivability, operability and understandability of diagrams presented using GraSSML. The Robust (R) principle of the WCAG 2.0 was not addressed nor assessed in this research.

The main objective was to get an initial idea of the POU(R) support of GraSSML used by blind users against a set of predefined tasks.

Of course, it is acknowledged that every blind user is different (Appendix B) and has different preferences and ways of working, so evaluating the system with only two blind users is not representative of the blind community. It is also important to keep in mind that the primary aim of this research project does not lie in the evaluation of the modalities employed at the presentation level. It was never intended to create a new presentation level modality or to develop the best accessible interface taking advantage of the proposed approach. It is recognized that other projects, involving specialised teams of researchers and adapted material, have invested much effort in generating accessible presentations (Chapter 3). The presentation level of GraSSML relies on and refers to these existing established approaches (see Chapter 3 section 3.2) (e.g. verbalisation models guidelines, TeDUB approach to presenting information, T3 or ViewPlus use of audio/tactile modalities, non speech sound, etc.).

The evaluation here gives an insight into the kind of results and reactions to be expected from the main benefits of the GraSSML approach which is the ability to enquire and reason about the information on which the diagram is based.

This phase of the project was carefully prepared and much thought was given into the methods to use. Aspects to be taken into account included the search for blind users willing to carry out the evaluation, the approach to be evaluated, the type of evaluation to use, the set up of the evaluation and the evaluation itself as well as the analysis of its results. “Welcome to Just Ask: Integrating Accessibility throughout design” by Shawn Henry (HENRY, 2007) was a useful resource that provided awareness of different aspects of accessibility implementation and interaction with disabled users.

Before the evaluation with blind participants, an informal “heuristic evaluation” (NIELSEN and MOLICH, 1990) was carried out with three sighted users, one being familiar with screen readers (Jaws in particular). The aim of the heuristic evaluation was to identify obvious faults in the design of the overall interface used by blind or sighted users but it would also provide an initial idea of the POU support of the GraSSML system by sighted users against a set of predefined tasks.

As finding keen blind participants with the right profile is difficult, it was important not to waste them. To maximize the chances of success, it was important to remove any interface errors early on. The heuristic evaluation allowed the correction of identified interface problems and the evaluation with blind participants was carried out with the revised version of the prototype. Both evaluations and their respective results are presented in the following sections.

7.1 Heuristic evaluation

Heuristic evaluation (BENYON et al., 2004) involves usability experts checking the GraSSML prototype against a list of heuristics for good design.

The heuristics were used as a basis for the evaluation. These heuristics are the WCAG 2.0 principles (perceivable, understandable and operable).

- **Perceivable:** Information and interface components must be perceivable. Users must be able to perceive the information presented using at least one of its senses.
- **Operable:** Interface components and navigation must be operable. Users must be able to operate the interface.

- Understandable: Information and the operation of the interface must be understandable. Users must be able to understand the information presented as well as the operations the system offers.

7.1.1 Participants

This evaluation involved three participants who provided feedback on the degree to which the user interface was usable. All three participants were familiar with graphical user interface design having studied these at university. Only one of the participants was familiar with the use of screen readers.

7.1.2 Method

The evaluators were first informed about the aim of the prototype and its intended use. The evaluators were asked to explore the GraSSML system, considering each heuristic in turn in an effort to identify potential usability issues with respect to the three principles presented earlier (perceivable, understandable and operable). They were asked to bear in mind that the GraSSML system evaluated was only a prototype aiming at evaluating the applicability of the proposed approach. It was important not to go too deep into the evaluation for example concerning the aesthetics of the interface.

The evaluation was conducted with use cases where the participants had to carry out typical user tasks (section 7.2.4). The walkthrough involved the same tasks to be used in the evaluation with the blind participants. These tasks could be achieved in different ways using different modalities offered by GraSSML:

- Textual presentation: the verbalisation model of the structure and the semantics of the diagram
- Graphical presentation: the SVG presentation
- Query systems: the textual query system and the graphical query system.

Even though the aim was to prepare for the evaluation with the blind participants, the opportunity was taken to evaluate the whole interface, including the parts the blind users would not access but would appreciate if working in a collaborative environment with sighted users.

Carrying the heuristic evaluation this way would also provide an initial idea of the perceivability, operability and understandability support of the GraSSML system by sighted users against a set of predefined tasks.

The evaluators were expected to evaluate the different modalities following the tasks provided, first using the mouse and then using only the keyboard and screen reader. Only the evaluator experienced with the screen reader Jaws was asked to switch off her screen. Each evaluator worked independently, and was asked to report any problems, identify the main principle it corresponded to and to make a recommendation when possible.

7.1.3 Results

The process took between 3 and 4 hours for each participant taking into account the interaction needed when they were unsure about some aspect of the prototype.

It was interesting to see how the participants could perform the tasks provided.

They were able to perceive, operate and understand the diagrams presented to them using the GraSSML approach.

Some problems with the prototype interface were identified. It gave a good overview of what might be incompatible with the intended blind users' needs and preferences.

The main problems reported concerned operability, in particular keyboard accessibility where some parts of the interface were not keyboard accessible. The screen reader user reported problems concerning "lost focus" which confused the screen reader. Also when the focus was correct the logical order of the information explored with the keyboard was not correct. Other problems reported were consistency of layout and button text.

It was suggested that confirmation dialog boxes should be added in appropriate locations to avoid the user going through a whole lot of stages by mistake or simply to exit the system by mistake. The lack of accelerators was also mentioned.

The problems identified have been categorised in terms of perceivability, operability and understandability principles.

The following problems and suggested solutions were obtained:

❖ **Operability:**

- **Problem:** The main interfaces for both types of diagrams were judged confusing. Evaluators became lost and did not know what to do next. This was primarily due to features of the interface related to the authoring process not required by their set of tasks.

Recommendation (implemented): creation of two views: “authoring mode” and “view mode”.

- **Problem:** One interface presented problems with tabulation order when it was used with a screen reader.

Recommendation (implemented): review the implementation of the tabulation order.

- **Problem:** The user was unable to display previously searched information to recall the results. He had to redo the search.

Recommendation (not implemented): provide a way to keep track of previously explored information.

- **Problem:** The prototype only provides a solution to a limited set of predefined queries.

Recommendation (not implemented): provide an end point to allow users to create their own queries.

- **Problem:** After using the same interface for a while, the evaluators noticed the lack of alternative ways to perform some functionality. An example was the “who manages...” interface from the textual query system. For the query to be executed, the interface requires the name of the employee to be entered and the “search” button to be pressed. Once experienced with the interface, the evaluator would have loved to simply press enter after entering the name of the employee instead of having to tab to the “search” button or click on it using the mouse. The same remark applied to other parts of the prototype.

Recommendation (implemented): Provide multiple ways of doing things in order to accommodate different level of user experience and habits developed from other similar interfaces.

❖ **Understandability:**

- **Problem:** The evaluators expressed a concern that the amount of information presented in the menu was confusing and that it was not needed to access the information of the diagram. One evaluator suggested the creation of two modes, an “authoring mode” presenting all the options to create/amend/explore a diagram and a “view mode” to access the information of the diagram.

Recommendation (implemented): creation of two views: “authoring mode” and “view mode”.

- **Problem:** The steps needed to “Traverse” a UML activity diagram were not obvious and needed clarification. This was mainly due to the names given to the buttons and the different ways the information was presented (using a text editor view and then using a table view). The table view confused the users and its presence was not understood.

Recommendation (implemented): change the labels of the buttons and provide clear information about the role of the different views presented. Alternatively, provide a help facility describing the role of each element of the interface.

- **Problem:** Different names were used for buttons having the same function. Their positions on different interfaces varied, confusing and triggering wrong operations from the evaluator.

Recommendation (implemented): make sure all features have been used, named and positioned consistently.

- **Problem:** The names given to the windows were unclear leading to users not knowing which window he was looking at. This applied to the evaluator using his keyboard, screen reader and with the screen switched off to access the interface.

Recommendation (implemented): review the names given to the windows.

- **Problem:** The prototype did not allow the user to undo errors made when asking for some options He had to either exit the task and go through the process again or wait until the end of the process and restart a new one. The same problem appeared when the user unintentionally clicked on exit.

Recommendation (implemented): include a confirmation dialog box before executing a task.

- **Problem:** Some messages were judged as not convivial and informative enough, mainly the error messages.

Recommendation (implemented): review these messages and amend them.

7.1.4 Conclusion

This evaluation has demonstrated how the diagrams expressed in GraSSML could be perceivable, operable and understandable for main stream users.

No problem concerning the perceivability of the information has been reported by the participants. The problems reported concerned exclusively the operability and understandability of the system. Most of them could be addressed, within the time frame of the thesis, by following the recommendations.

This evaluation proved very useful in identifying usability problems concerning the graphical interface of GraSSML. If not corrected, the identified problems would have created problems during the blind user evaluation and wasted very precious time with both blind participants. The version of the prototype used by the blind users included the amendments resulting from the heuristic evaluation.

7.2 Blind users Evaluation

This section describes how an evaluation of the GraSSML prototype, carried out with two blind users, allowed us to assess the perceivability, operability and understandability of GraSSML.

This evaluation aims at demonstrating the hypothesis that “if information on the structure and the semantics of formal diagrams were preserved, made ‘part of the diagram’ by willing authors at the creation stage, these diagrams would be more perceivable, operable and understandable and, as a result, suggest enhanced accessibility benefits for such diagrams.”

Will the user be able to perceive, operate and understand diagrams using the GraSSML system? Will he be able to enquire and reason over the information on which the presented diagram is based? How would the users react to the GraSSML system? Will they consider using this kind of system to access diagrams? To find answers at these questions, two methods have been used: a user evaluation and an interview.

7.2.1 Participants

Both participants were male and aged between 25 and 30 and blind from birth. They volunteered when a call for participants was given through different discussion groups, forums, events, etc. The first participant graduated from Loughborough University at the Department of Computer Science. The second participant held a

GNVQ in Information Technology. Both were computer literate and taught their knowledge to other people (visually impaired as well as sighted). Both considered themselves as screen reader experts with Jaws (they also had experience with other screen readers: windows eye, NVDA). Both were familiar with tactile diagrams and diagrams described in audio or by a third party.

Both were familiar with the type of diagrams used in the experiment. They recalled the frustration at not being able to access such diagrams and needing the constant assistance of a sighted person in attempting to describe them. They considered their knowledge of the domain UML Activity diagram as basic. In the sense that they had learned about the concepts but because of the unavailability of appropriate application compatible with screen readers, they did not go too deep into the exploration or creation of these diagrams. Both participants expressed a very good interest and excitement about the ideas behind the GraSSML project. They were both used to carrying out similar kinds of evaluations for different research projects at universities and companies.

7.2.2 Equipment

A laptop with a standard keyboard was used for the evaluation. The GraSSML system was installed on the computer and accessed using the mouse, keyboard and screen reader. The screen reader Jaws was installed. An audio and video recording was made for the entire evaluation and later transcribed for analysis. The participants were encouraged to describe verbally any problem they encountered.

7.2.3 Materials

Six diagrams, three from each class were selected for the evaluation. These diagrams varied in size (number of objects) and in complexity (specific aspect such as synchronisation, loops).

These variations were aimed at encouraging the participants to explore different modalities in completing the tasks. The diagrams are shown in Figure 91 to Figure 96.

Diagram 1 (Figure 91) presents a simple activity diagram for attending a course lecture. This diagram is composed of just 4 actions and a decision point.

The first activity is to get dressed before leaving for the lecture. A decision then has to be made, depending on the time available before the lecture starts. If there is sufficient time to catch the train, take the train; otherwise, take a cab to the University.

The final activity is to actually attend the lecture, after which the activity diagram terminates.

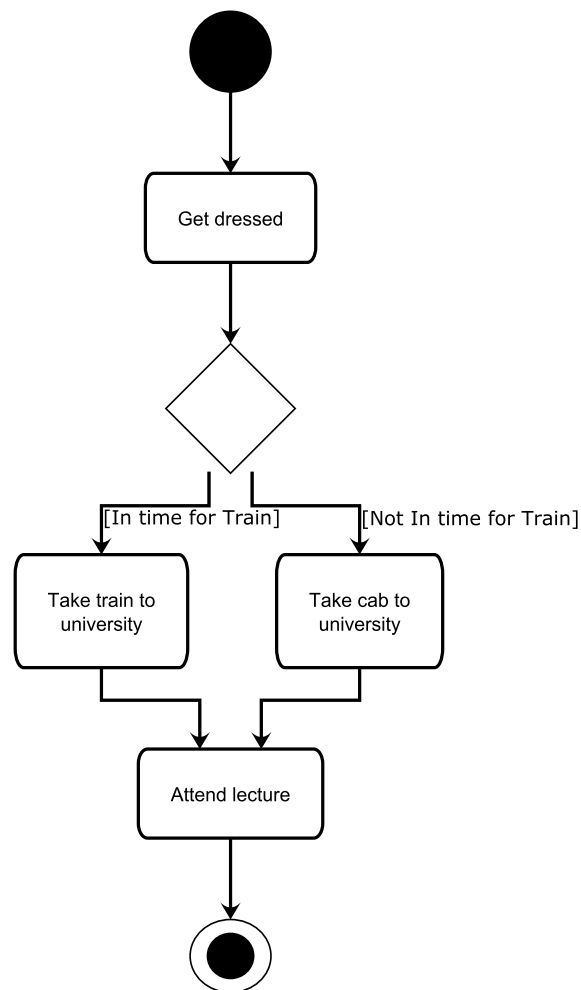


Figure 91: UMLAD for attending a course lecture (Diagram 1)

Diagram 2 (Figure 92) presents a simple activity diagram composed of 3 actions, a decision point and two Parallel Processes Bars. The loop formed from the decision point back to the first action “Eat something good from the kitchen” as well as the parallel process bars are the main distinctiveness of this example.

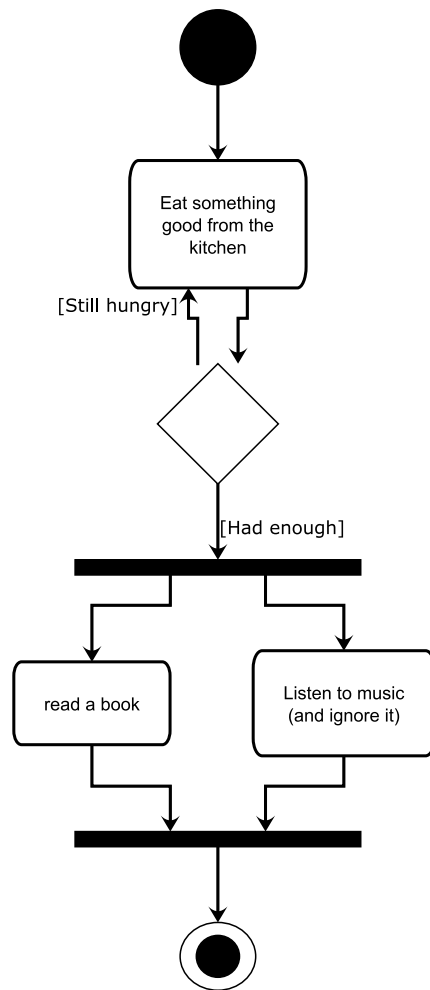


Figure 92: UMLAD for eating when hungry (Diagram 2)

The final UML activity diagram example presented in diagram 3 (Figure 93) is the UML activity diagram used previously (Chapter 5 section 5.3). Its complexity compared to the previous diagrams is of interest as it presents more elements including one decision point; two decisions modelled out of an action (“fill out enrolment forms”), concurrent activities which include a set of sequential activities.

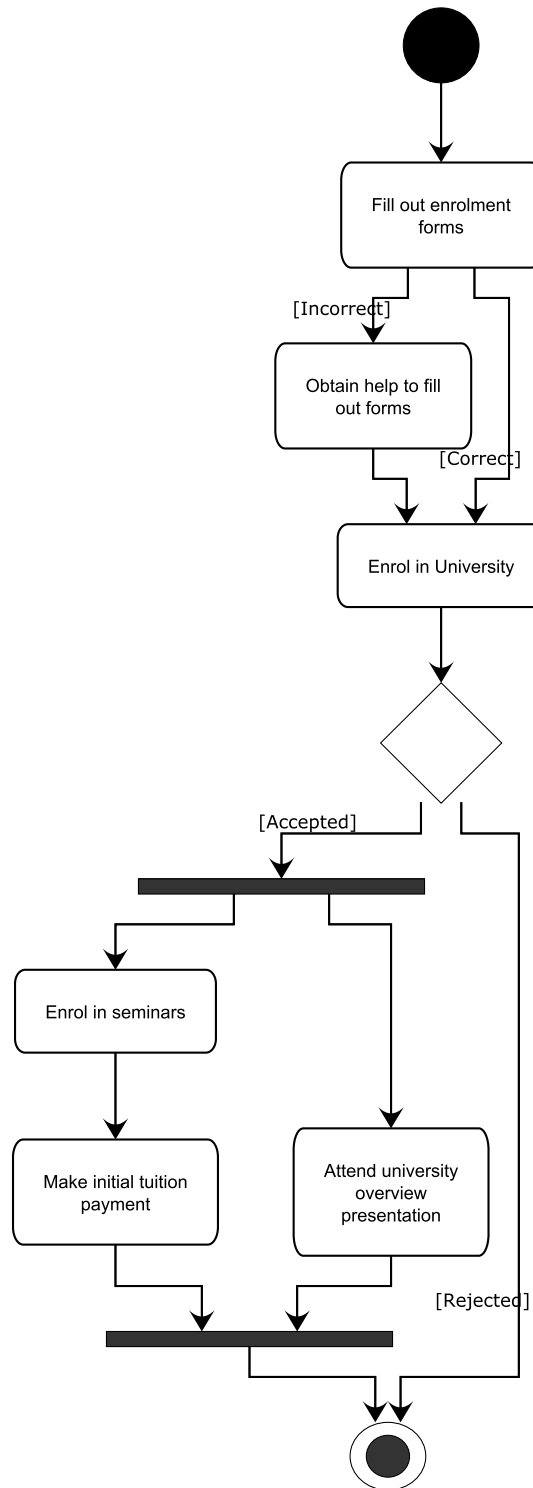


Figure 93: UMLAD Enrolling in the university for the first time (Diagram 3)

The three diagrams (Figure 94, Figure 95 and Figure 96) are organisation charts which differ in terms of complexity, meaning the number of employees involved and the number of levels they are organised into. All of these examples have already been described in Chapter 5 section 5.2.

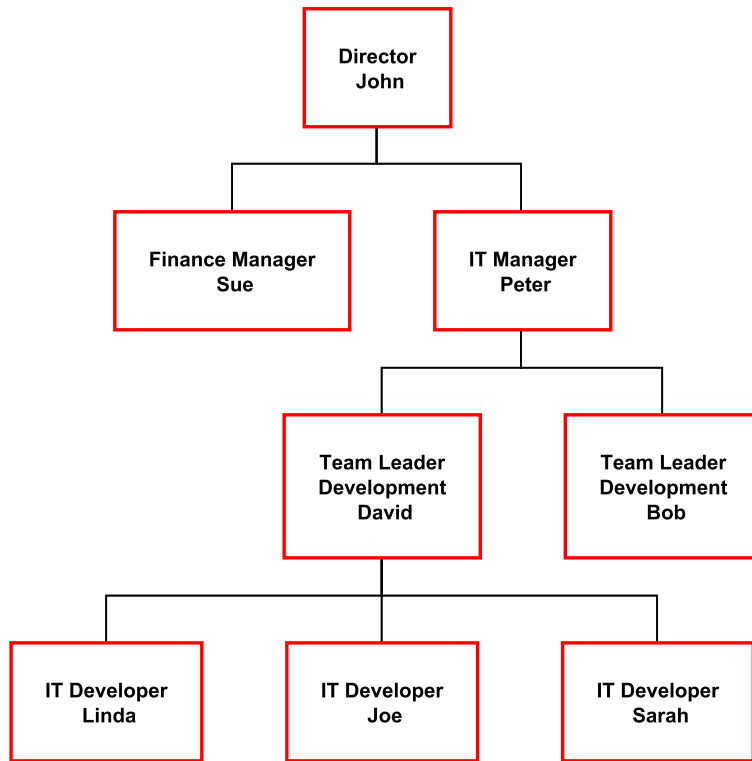


Figure 94: My Simple Organisation Chart (Diagram 4)

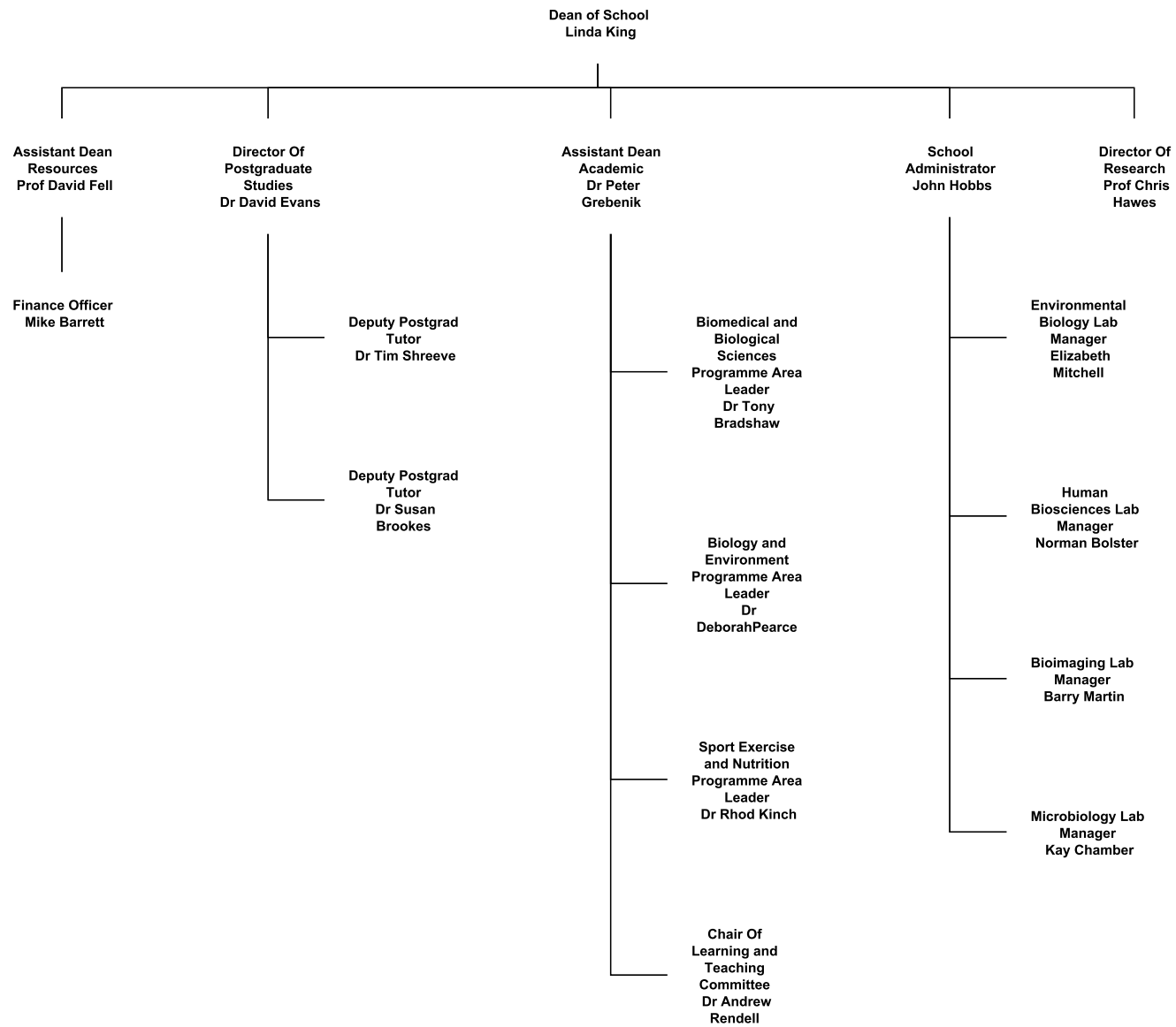


Figure 95: Oxford Brookes University Organisation Chart Example (Diagram 5)

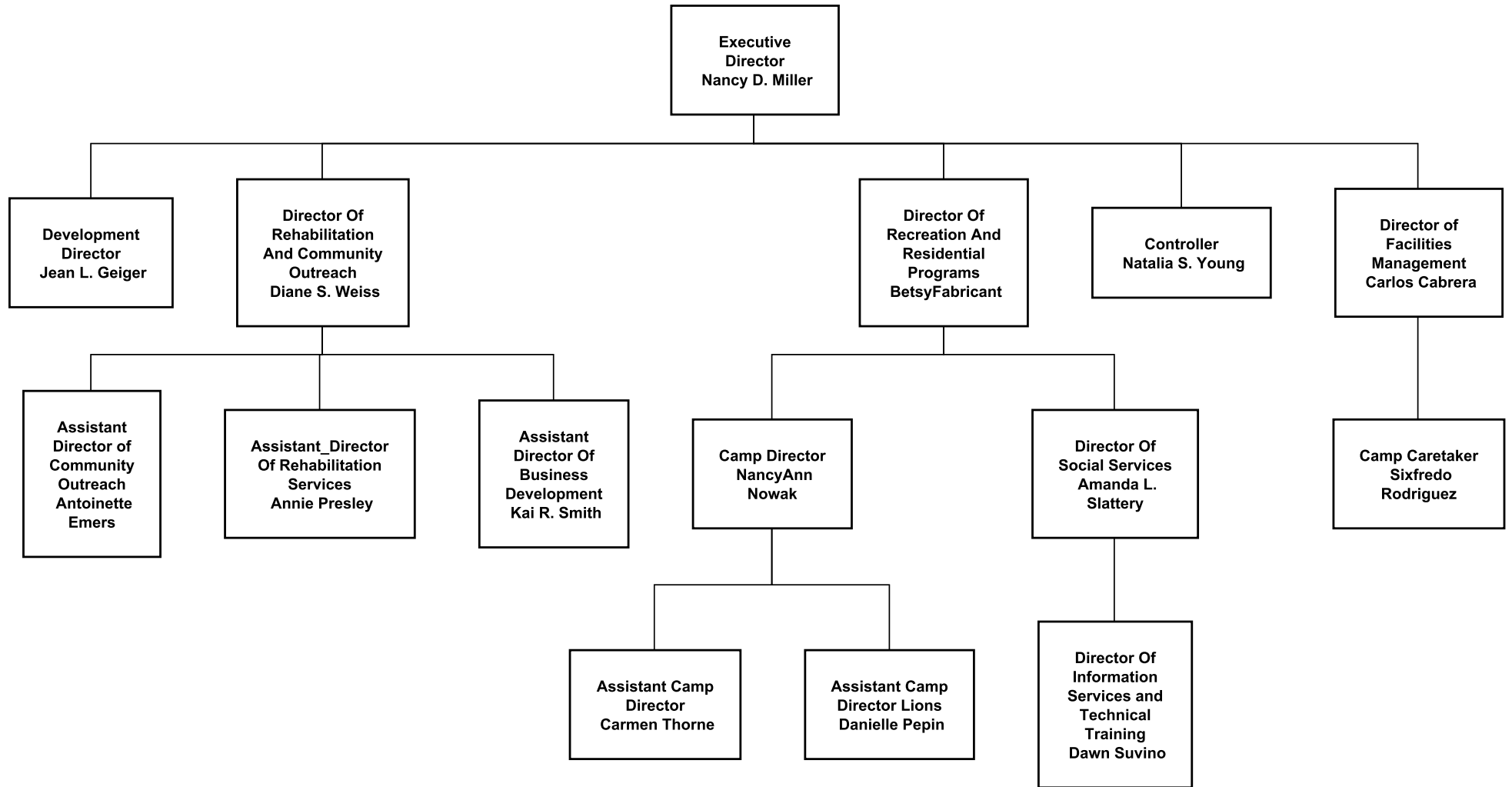


Figure 96: VISION Organisation Chart (Diagram 6)

7.2.4 Tasks

The tasks were aimed at exploring different parts of the GraSSML prototype. The tasks were deemed realistic, feasible with the prototype and explored the prototype thoroughly. They were mainly aimed at finding out if the user could gain an overview of the diagram and then answer some specific questions about the diagram to assess the understanding of the information accessed. The questions were typical for these types of diagram. The user was not expected to achieve all the tasks perfectly at the first attempt using GraSSML. Learnability is considered an important aspect to observe when performing the tasks. It was hoped the repetition would allow assessment of the learnability of the user in using the prototype. Would the familiar user develop different strategies in using the prototype?

For each diagram a task sheet was prepared. The following section presents the different tasks for the different diagrams:

❖ **Diagram 1: UML Activity Diagram “Getting dressed”**

- How many actions? And what are these actions?
- What can you tell me about this diagram?
- What can you tell me about the complexity of this diagram?
- Is there any decision that needs to be taken? If yes how many? And what are they? Answer the question using first the textual description and then the query system.

❖ **Diagram 2: UML Activity Diagram “eating when hungry”**

- What can you tell me about the complexity of this diagram compared to the previous one?
- How many actions? And what are these actions?
- Is there any decision that needs to be taken? If yes how many? And what are they?
- Can you describe what happens if “still hungry”? And also if “had enough”?

❖ **Diagram 3: UML Activity Diagram “Enrol at University for the first time”**

- What can you tell me about the complexity of this diagram?
- What is the sequence of actions happening before “Enrol at university”?
- Is there any decision which needs to be taken?

- What action needs to be processed before the action “Make initial tuition payment”
- Are there any actions executed in parallel? If yes, what are they?

❖ **Diagram 4: Simple Organisation Chart**

- How many employees are there?
- Who manages Sarah directly?
- Who manages Sarah indirectly?
- Who reports to Bob?
- Who reports to Peter directly?
- Who reports to Peter indirectly?

❖ **Diagram 5: Vision Organisation Chart**

- How complex is the diagram compared to the previous one? How many levels and employees are there?
- Who manages Nancy D. Miller directly?
- Who manages Nancy D. Miller indirectly?
- Who reports to Annie Presley?
- Using the list description of the textual description of the semantic of the diagram can you tell me who reports directly to Betsy Fabricant?
- Can you confirm your answer using the query system?

❖ **Diagram 6: Oxford Brookes University**

- How complex is the diagram compared to the previous one? How many levels and employees are there?
- Who is the director?
- Who reports directly to the director?
- How many persons in total report directly and indirectly to the director?
- Who manages Barry Martin directly? Who manages Barry Martin indirectly? First using the list description, then using the prose description and then using the query system.
- How many employees are there in level 2?

7.2.5 Methods

The blind users were given different dates for the evaluation which took place in isolation at the university. Both participants journey was planned with them and arranged in a way for them to feel safe. They were greeted and collected directly from the train station to the evaluation location. It was ensured that recording facilities were available and functioning. There was no time limit for the evaluation. The users were thanked for their time and work. They were then accompanied back to the train station. They were also reimbursed for their travelling expenses.

During the whole process notes of observations were taken on paper and audio as well as video recordings were made.

The whole evaluation process involved five different phases which are presented in the following section.

A. Pre-experiment questionnaire

An initial questionnaire was answered by participants to gain an insight into their visual ability, the duration of their visual impairment, their computer literacy, their educational background, their experience with diagrams and the methods used to access them, their knowledge of UML Activity diagrams and Organisational charts and finally their experience of screen readers and preferences.

Additionally they were encouraged to share any previous experience with this type of diagrams, specifying where, how, the type of modality and interaction involved.

B. Presentation

First the participants were thanked for participating at this evaluation.

Before starting the evaluation the participants were told what the project was about and the nature of the evaluation. They were then explained the procedure that was going to be followed. They were also informed of their right to stop at any time, to ask for a break, or to ask any questions if needed.

It was specified to the participants that they were not the ones being tested but that they were testing the system, so they should not worry about making mistakes. They were made aware that the evaluation sessions would be videoed and tape recorded and that they were free to refuse.

C. Training

After the introduction of the project and the description of the evaluation process, each participant started with some training to familiarise themselves with the interface of the prototype.

This involved familiarisation with the interface and exploration of the different modalities offered by GraSSML. The first diagrams of both domain (Figure 91 and Figure 94) were used here. At this stage of the evaluation, the participant was allowed to ask questions related to the interface and the interpretation of the information. Remarks and questions were recorded for later analysis.

D. The evaluation

The evaluation was to determine whether the blind users could extract useful information and perform content-related tasks based on information, extracted from the data model and ontology, and presented using GraSSML alternative presentations through enquiry and reasoning.

Each participant was asked to complete a series of tasks on 6 different diagrams. Task success as well as the process followed by the participants to achieve the task was observed. The tasks were read out by the evaluator. The tasks were designed to test the participant's perception and understanding of the information presented by the diagram. The answers were recorded and the user did not receive a feedback on the correctness of the answer. When the participant seemed confused and frustrated, an attempt to clarify the question was allowed by asking further questions or the decision not to perform the particular tasks were allowed. A pause was taken between the two type of diagrams or when the participant required a break.

The main aim was to judge if the tasks were achieved successfully or not. At this stage of the project, the evaluation was more concerned with effectiveness than efficiency of the prototype in term of correctness and success. This involves the access of the appropriate information by the user as well as how well this information was conveyed and how well the user made use of it by converting it into knowledge for action.

The ease of use was also looked at, as this involved investigating aspect such as ease of learning, understandability and subjective satisfaction. So as well as observations made, measurements such as the number of correct answers were taken during the evaluation.

The participants were asked whenever possible to use “Think aloud protocols” which involves thinking aloud as they perform the given tasks. This allowed getting information on what they were thinking of and/or feeling while completing the tasks. The users were encouraged to give comments on what they were doing, why they were doing it and specify uncertainties they encounter. Notes were taken on when users found problems or did something unexpected. Help was provided if users found themselves stuck in the task and helped to move to the next task. It was considered important to keep users talking to understand what was going through their mind while doing the evaluation so questions such as “What do you want to do?”, “What information is presented to you?”, “Is there a different way of doing it?”, “Why did you choose this one?”, “What were you expecting?”, etc. So the evaluator (the author of this thesis) sat with each blind user and engaged with them as they carried out the tasks, providing some help if the user seems frustrated and lost.

The users were told that if they required information from the system that was not yet implemented but that was possible to implement if time was given, then the “Wizard of Oz” methodology (KELLEY, 1984) would be used. In such a case, the evaluator would take the place of the system and provide an answer to the query.

E. Interview

A short structured interview involving the following questions was carried out at the end of the evaluation. This stage of the evaluation is a more structured approach to obtain feedback from the users at the end of the evaluation.

- Which representation do you think allowed you to gain an overview of the information of the diagram?
- Did it give you an idea of the complexity of the diagram presented?
- What did you think of the amount of information presented?
- How easy was it to manage this information to complete the tasks?
- How easy was it for you to select the right representation to complete the tasks?
- At any point did you think you were lost and/or confused? Why?
- What do you think of the proposed approach for accessing diagrams?
- Would you consider using this approach to access diagrams?
- Was there anything particularly good or bad about this approach?
- What would you suggest to improve the whole experience?

7.2.6 Results and discussion

The results of the evaluation demonstrated that the participants were able to perceive, operate and understand diagrams presented using GraSSML.

It was observed that the participants did not rely on a specific modality but on a set of modalities. Indeed, once the participants felt confident in navigating between the different modalities, they started using them to complement and validate each other. However, users were observed to have a preference toward the textual representation of the semantics to gain an overview of the diagram and to refer to the diagram when wanting to verify information they needed.

One user enjoyed the different presentations of the same information by using different formats. Being able to navigate from the textual description (that he considered as a complete reassuring description of the diagram to have) and the more specific textual query system was reassuring as he could find the wanted answer very fast and check his answers if he was not sure.

Both users expressed their appreciation of the possibilities offered to extract information from part of the diagram while maintaining access to the entire diagram in different way. They felt secure having access to this external memory support.

Both users liked the fact that the verbalisation allowed them to get an overview of the diagram whereas the query system allowed them to get a specific detail from the information.

A strong preference for the textual query system was also expressed, both users seemed to strongly appreciate this feature but they also made the remark that they would rather love having a facility to allow them to decide on the queries formulation to complement the basic queries available. For both users, having such search facilities would be considered as ideal. Based on their past experience they compared the query system as constantly having a third party answering your questions but without being uncertain of the answer or tired of repeating the information.

Once familiar with the interface, navigation tools and format employed to present the information, the users developed their own strategy at resolving the tasks given. Both mentioned that they felt reassured of having constantly access to the semantic textual representation of the diagram as they could refer to it, search it and explore part of it when wanting to remember particular information about the diagram.

When asked about the quantity of information presented, both replied that they felt confident in having access to the whole information and being able to choose how to navigate it using different views. As well as the query system, the detailed list description, once the way the representation works was understood, was a real success in browsing the information quickly. However, for one user it took some time to understand this representation as he made consecutive mistake when asked to use this representation to provide the answer. He said that he was not used to nested lists. One interesting point he made was the confusion in the organization chart presentation between the name of the level provided by GraSSML and the way Jaws numbered the levels in the nested list. For example, the root element is considered as level 1 in GraSSML and identified a level 0 using the nested list navigation with Jaws.

One user felt it would be useful to have the visual equivalent of the spoken page available at the same time so that a sighted co-worker could be called in for clarification or work co-operatively with the blind worker. It was explained to him that the graphical representation of the diagram could also be provided and implementing a parallel view of it is possible.

In the case of complex diagrams, exploring the diagram non-visually can place considerable strain on short term memory. It was suggested that the interface should provide a way of marking points of interest in the information provided textually for the user to be able to return to if needed. This would be of benefit to both sighted and visually impaired users.

When exploring UML activity diagrams both users did not understand the role of the “table representation” presenting the full information of elements when traversing the diagram from one activity to another. Once explained, they strongly advised on the creation of a help file providing this kind of information and other information such as a summary of all the commands and their corresponding shortcuts.

One of the users while using the textual query system to find out in diagram 1, “who reports indirect” or “who manages indirect” found that the results were confusing as the order provided did not reflect the chain of responsibility. The evaluators in the heuristic evaluation provided this remarks but the solution used to resolve the problem was apparently not enough (description of the details relationships between the result employees).

One point that both users found frustrating was the opening of some functionalities which needs an XSLT transformation as the screen reader would read the whole information present on the command prompt while the transformation was running. Once used to it, they used “Ctrl” to make the screen reader stop reading it.

It was overwhelming how both blind users expressed a strong interest in the developed prototype and required a copy to experiment further with it. One participant asked of the possibilities offered to create diagrams using GraSSML. Once explained he asked to make an attempt at using ZineML to create a simple activity diagram. This was successfully achieved. It was explained that this aspect of the project needed more work as using GraSSML for the creation of a more complex diagram might cause some problems and that ZineML was not really aimed to be authored this way. An attempt was made to demonstrate the use of protégé, but it was noticed at this stage that the interface provided by protégé was completely inaccessible for screen readers.

7.3 Conclusion

Both evaluations supported the initial hypothesis “if information on the structure and the semantics of formal diagrams were preserved, made ‘part of the diagram’ by willing authors at the creation stage, these diagrams would be more perceivable, operable and understandable and, as a result, suggest enhanced accessibility benefits for such diagrams”.

❖ GraSSML supports perceivability, operability and understandability

These evaluations demonstrated that GraSSML supports three of the principles of the WCAG 2.0: perceivable, operable and understandable.

Both sighted and blind users were able to correctly answer all the questions. They performed content-related tasks based on the information by enquiring and reasoning about the information on which the diagram is based.

The information of the diagram presented through the verbalisation models and the query system were found helpful for the enquiry of the diagrammatical content.

This demonstrated the perceivability, operability and understandability support of GraSSML for both types of users against a set of predefined tasks.

All participants expressed a strong interest in the GraSSML system asking if it would be available soon.

These evaluations support the ability for further studies, for greater work. Based on suggestions made by the participants, it can also be concluded that the future of GraSSML is very promising as all users were able to enquire diagrams.

❖ **GraSSML suggest enhanced accessibility benefits**

All participants were able to perceive, operate and understand the diagrams GraSSML presented. Perceivability is the first step to accessibility upon which all others are based, and without which accessibility cannot happen (WebAIM, 2009b).

With GraSSML, the information about the diagram is preserved “made part of the diagram” in a way that improved the ability to enquire and reason about this information making it perceivable, operable and understandable as demonstrated in both evaluation.

It is acknowledged that both evaluations could not be used to make claims about accessibility of the diagrams presented with the GraSSML prototype. The results won't be conclusive taking into account that the two blind users are not representative of the whole blind population. A much larger set of users should be recruited to take part in a more controlled evaluation of GraSSML; more thought needs to be put in into the different way of testing the different aspects GraSSML has to offer, adapted presentation modalities included.

❖ **Implicit accessibility benefits results**

The representation of the information at the presentation level in GraSSML can be various and it was never the aim of GraSSML to create a new representation for this level but more to draw inspiration from existing projects previously presented which are specialized at that level.

It is in that context that for example, the best reliable verbalisation models that are around have been sought. They have been demonstrated to work and the implementation of the GraSSML approach demonstrated that they can be generated automatically from the information captured at the creation of the diagram. So it has been demonstrated that GraSSML can generate verbalisations that others have subjected to more vigorous evaluation and they demonstrated to be working.

❖ **Proposed approach for future accessibility evaluations**

Currently, there is no equivalent system to compare GraSSML against.

Projects reviewed in Chapter 3 such as TeDUB, T3 and ViewPlus have researched, produced and obtained some very good results in terms of presentation of diagram non-visually. It is acknowledged that if all of these approaches had clean available and accessible formats of information, that they are using to generate their elaborated presentations, then GraSSML will simply generate them and will feed them back in their system that they have proved to work. Redoing the same evaluations as they did but using the GraSSML approach to capture the information behind the diagram could be done. This could bring some answers to the problems (Chapter 3) they have identified at the creation level of the information to be used (automatic and semi-automatic recovery of the information).

So a proposed approach for future evaluations of GraSSML would involve the performance of a comparative study of the usability and accessibility of output obtained using GraSSML against results obtained from a standard non GraSSML diagram. It is suggested to use the TeDUB (HORSTMANN et al., 2004a) system which has already produced good results when rendering standard diagrams in accessible form for visually impaired user. In other words, to explore the extent to which GraSSML can automatically generate the kinds of best-practice presentations that other approaches generate manually or semi-automatically. The system would be used to render GraSSML versions of a diagram and also render a standard graphical version of the same diagram and compare the results for usability and accessibility purposes. In this way it can establish whether the inclusion of the information behind the diagram contributes to the usability and accessibility of the rendering of the diagram.

Chapter 8

Conclusion and Future Work

This thesis contributes to the representation of diagrams in a way that improves the ability to enquire and reason about the information on which the diagram is based and thus making diagrams more perceivable, operable and understandable and, as a result, suggesting enhanced accessibility benefits for such diagrams.

This chapter summarizes the main findings of the research, presents the research contributions made and discusses open issues and potential future work.

8.1 Main findings of the research

Diagrams offer powerful advantages (Chapter 2) in presenting, accessing and processing information. They are created in order to communicate a specific intent which needs to be accessed to discover the knowledge the diagram carries.

Different approaches aiming at making diagrams accessible have been reviewed and analysed (Chapter 3). This analysis allowed the identification of a number of issues current approaches present for the problem of diagrams accessibility on the web. The absence of information “behind” the diagram which is lost at the creation stage has been identified as the main problem. This led to the identification of a set of requirements (Chapter 3 section 3.5) a new approach should satisfy in order to overcome them. This set of requirements would be incorporated into a system aimed at creating perceivable, understandable and operable diagrams.

This research explored the concepts underlying a complete system based on these requirements and defined a solution for some specific diagram classes (Chapter 2 section 2.4.5). The proposed approach is directed towards the acquisition of the information behind carefully selected “formal diagrams”.

8.2 Contributions

The contributions of the thesis are summarized as follow:

- The GraSSML Conceptual Architecture
- A fully working prototype demonstrating the feasibility and applicability of the approach for three different application domains

- A methodology to apply the GraSSML conceptual architecture to a given application domain
- Credibility of GraSSML: evidence of the benefits of GraSSML through instances of evaluation of the working prototype with users.

8.2.1 The GraSSML Conceptual Architecture

The main contribution of this research is the introduction of a novel approach called Graphical Structure Semantic Markup Languages (GraSSML). The hypothesis underlying the GraSSML approach is that: “if information on the structure and the semantics of formal diagrams were preserved, made ‘part of the diagram’ by willing authors at the creation stage, these diagrams would be more perceivable, operable and understandable and, as a result, suggest enhanced accessibility benefits for such diagrams”

GraSSML makes the information on the structure and the semantics of the diagrams ‘part of the diagram’. Graphical content is no longer thought of as ink on the paper or pixels on the screen but as the intrinsic structure and semantics of the information. The aim is to start with the information behind the diagram content instead of its graphical (visual) presentation and then generate one of its possible presentations without losing access to the initial meaning. Thus, rather than starting from a filtered view (i.e. low level, low information image) and trying to infer the information, the initial information is used as a starting point.

GraSSML contributes to the representation of diagrams in a way that improves the ability to enquire and reason about the information on which the diagram is based and thus making diagrams more perceivable, operable and understandable and, as a result, suggesting enhanced accessibility benefits for such diagrams.

This framework relies on the presence of essential selected information provided by domain experts and the willingness of authors to allow the capture and access to such information while creating their diagrams upon which the approach relies. This exposes a new way of thinking about the authoring of diagrams.

In order to evaluate the GraSSML approach, three different evaluations from three different perspectives have been carried out: a technical perspective evaluation, an author’s perspective evaluation and a user’s perspective evaluation. These are discussed in section 8.2.2, section 8.2.3 and section 8.2.4 respectively.

8.2.2 Fully working prototype

The viability of the approach and its applicability for three different application domains has been demonstrated through a fully working prototype.

A fully functional implementation of the GraSSML conceptual architecture has been implemented in a proof-of-concept tool (GraSSML prototype). Three use cases have been considered, each for different classes of diagrams from different application domains: Process diagram “UML Activity Diagram” (Chapter 5 section 5.3), Hierarchical diagram “Organisational Charts” (Chapter 5 section 5.2) and Charts “Financial charts” (Chapter 6).

The system uses a combination of web technologies, including OWL, RDF, XSLT, GRDDL, along with some new XML based languages (ZineML), in an attempt to enable the authoring of diagrams based on their meaning (semantics) and not their visual rendering/ presentation. This exposes a new way of thinking about the authoring of diagrams and is an interesting use of the semantic web technologies to assist this authoring process.

The GraSSML prototype has demonstrated that the GraSSML conceptual architecture is sound.

8.2.3 Methodology to apply the GraSSML approach

A methodology to apply the GraSSML conceptual architecture to a ‘formal diagram’ in a given application domain has been developed. An author’s perspective evaluation provided information on the viability and applicability of the methodology developed for the application of GraSSML conceptual architecture.

Olivier BRAECKMAN, an MSc student, independently, applied GraSSML to financial reports in order to represent “financial charts” in a way that it is possible to reason and enquire over the information they carry thus making them more perceivable, operable and understandable and, as a result, enhancing their accessibility. The application of the methodology has been tested by Olivier during a double blind trial in which he, alone, performed the data gathering, processing and recording as well as an initial analysis. After which the results and data obtained were analysed, interpreted and evaluated. The viability and applicability of the GraSSML approach was successfully assessed by Olivier who extended the system for a different class of diagrams. The evaluation of the process is presented in the evaluation chapter from an author’s perspective (Chapter 6).

8.2.4 Credibility of GraSSML

The credibility of GraSSML has been demonstrated through the ability to reason and enquire about the information behind the diagram. This ability to reason and enquire has been demonstrated through the proof-of-concept prototype.

The three use cases implemented within the GraSSML prototype demonstrated the feasibility, viability and applicability of the GraSSML approach but did not provide any insight on whether GraSSML support the hypothesis.

An evaluation of the GraSSML prototype (Chapter 7), carried out with two blind users and three sighted users against a set of predefined tasks, demonstrated the perceivability, operability and understandability of diagrams using GraSSML and as a result demonstrated the hypothesis.

Accessibility of diagrams is one possible application of the proposed approach. Using GraSSML, the information content of the diagram (structure and semantics) is captured at the creation stage. The availability of this information is a key aspect of the approach. It allows GraSSML to provide, users and computers, the ability to enquire and reason over this information. By making the meaning explicit then GraSSML assist perceivability, operability, understandability and so, even though it is not directly tested, there is like an inductive reason for having these three principles. So by making diagrams perceivable, operable and understandable then by definition GraSSML can make them more accessible. Due to time limitations and lack of resources we cannot test this directly by carrying major user studies.

GraSSML could also be used in other type of systems different application

8.3 Open issues and future work

The work so far has demonstrated the feasibility and viability of the GraSSML approach but further work is needed to fully elaborate the architecture and test its general applicability. Work of value would be:

❖ **Authoring process**

- The GraSSML approach involves significant changes in the authoring process and the awareness of graphical information for the author and the user. It involves different phases which may be seen as expensive in time and effort. The time spent in the creation of subject ontologies, the creation of the notational conventions, the choice of useful queries, and the creation of the verbalisation model templates may be all seen as requiring too much

effort. Creating accessible diagrams is time consuming and any facilities to shorten the timescale and reduce the effort would be advantageous. The development of authoring tools to make the approach viable for a range of authors is to be considered. It is believed many developments could be carried out in order to make the process of using GraSSML a pleasant experience, making a range of authors confident in its viability as a solution to the problem of diagrams accessibility.

- This could be made in different ways considering the expertise and experience of the author and the user. The authoring tools could allow the creation of accessible diagrams by providing predefined libraries for data models, predefined verbalisation templates and ZineML definitions for a number of often used diagram classes for example. This could also involve the inclusion of editing tool for the creation of specific ZineML markups, templates, query systems, notational conventions. At this stage different options could be envisaged. The ultimate goal being to assist authors in learning and applying the GraSSML framework, developing specific ontologies, various representation levels of graphics, etc.
- A certain amount of domain knowledge is essential, in particular for the choice of minimal information needed (“primary resources”) and the choice of the appropriate vocabulary. A method to measure what a good data model is should also be considered and be created. This should allow knowing if there is enough information to produce an effective alternative.

❖ **Improve the GraSSML approach**

- At the moment the GraSSML prototype includes a SPARQL endpoint, though not visible to user. It essentially provides a user interface with specific queries to illustrate the approach. Other future options include
 - providing an explicit SPARQL endpoint to users,
 - a richer set of queries for situations where user tasks are well-defined,
 - a natural language interface to either pre-defined queries or a “full” SPARQL endpoint.
- Ontology at the structure level could be defined allowing the manipulation of the structural information of the diagram (e.g. queries on the structure).

- As not all users and situations are the same, adaptability is needed. Due to the diversity of users' ability and knowledge, diversity of devices and taking into account new regulations and guidelines on accessibility, there is a real need to make diagrams accessible and adaptable to the context in which they are used. Adaptability has been considered and taken into consideration during the design process of GraSSML but although it is possible to implement it has not been done for time reasons. So further development are needed at the presentation level, other modalities should be explored by using successful existing technologies such as the T3 (RNCB, 2009) or the IVEO (VIEWPLUS, 2009) project to presenting the information collected using multimodal modalities (e.g. Audio/Tactile)
- Possibilities of inserting an Explanation level which would justify the different answers obtain from querying the data model based on the information contained in the ontology should be considered. At the structure level, ZineML allows you to know information about the diagram that does not require you to know about the SVG. The same applies at the semantic level which allows you to know about the semantic of the diagram which does not require you to know about its ZineML representation. If an explanation layer exists, it will be possible to explain the reason behind the different answers from the query system. Not only will GraSSML be able to provide you with answers but also with explanations of why this information is provided.

❖ **Applying GraSSML to other existing systems**

- Investigating the application of GraSSML to other existing systems could be the subject of further research. Indeed, GraSSML could be the starting point for many projects currently aiming to access, present, explore and adapt graphical information (DUKE, 2004, MARRIOTT et al., 2004).
- Other researchers would be the main beneficiaries being exposed to the proposed GraSSML idea. In the long term all users including visually impaired people do stand to benefit.

❖ **Open source development: Releasing the GraSSML framework on the Web**

- A release of the GraSSML framework code to the general public on the web would be of benefit to both the general public and the development and evaluation of GraSSML itself.

❖ **Evaluation of the benefit of the use of GraSSML**

- The evaluations carried out have demonstrated the viability and the worthiness of the GraSSML approach but it is acknowledged that the results are not conclusive. Further evaluations and experiments are required to reach a definitive conclusion on the benefits GraSSML offers in accessing diagrammatic information for all (blind included). A much larger set of users should be recruited to take part in a more controlled evaluation of GraSSML; more thought needs to be put into the different way of testing the different aspects GraSSML has to offer, adapted presentation modalities included.
- Although GraSSML has been evaluated only with blind users accessing diagrams, it is strongly assumed that the approach could benefit a wider range of users in diverse situations as for example people who learn by hearing information or people with learning disabilities that would benefit from alternative presentations of graphical information, mobile users, etc. GraSSML could also find a good use for education: e-learning. Indeed diagrams which are known to have great importance in education could be made to “reveal” information about them that might not be obvious for some readers. Evaluations along these lines should also be considered.

❖ **Possible Standardisation**

This research could be the starting point to an accessibility guideline for vector graphics similar to WAI. WAI Support for the approach would be needed and system support (browsers and assistive technologies (e.g.: screen readers)) to allow users to access the graphics at the logical or semantic view of the graphic would be needed and could occur as a result.

8.4 Conclusion

The GraSSML approach contributes to the representation of diagrams in a way that improves the ability to enquire and reason about the information on which the diagram is based and thus making diagrams more perceivable, operable and

understandable and, as a result, suggesting enhanced accessibility benefits for such diagrams. The technical perspective evaluation and the author's perspective evaluation demonstrated the feasibility, viability and applicability of the GraSSML approach. And finally the user's perspective evaluation demonstrated the perceivability, operability and understandability of diagrams using GraSSML and as a result validated the hypothesis.

References

- ALDRICH, F. & SHEPPARD, L. (2000) 'Graphicacy': the fourth 'R'? *Primary Science Review*, 64,8.
- AMBLER, S. W. (2001) *The Object Primer: The Application Developer's Guide to Object-Oriented and the UML*, Cambridge University Press, 0521785197.
- ARNOLD, D. & DUCE, D. (1990) *ISO Standards for Computer Graphics*, Butterworth-Heinemann Ltd, 0408040173.
- AULT, H. K., DELOGE, J. W., LAPP, R. W., MORGAN, M. J. & BARNETT, J. R. (2002) Evaluation of Long Descriptions of Statistical Graphics for Blind and Low Vision Web Users. *Computers Helping People with Special Needs*. Springer Berlin / Heidelberg.
- BADROS, G. J., TIRTOWIDJOJO, J. J., MARRIOTT, K., MEYER, B., PORTNOY, W. & BORNING, A. (2001) A constraint extension to scalable vector graphics, Proceedings of the 10th international conference on World Wide Web. Hong Kong.
- BAILLIE, C., BURMEISTER, O. K. & HAMLYN-HARRIS, J. (2003) Web-based teaching: communicating technical diagrams with the vision impaired, Proceedings of the Multi-Modal Content: Flexible, Re-useable and Accessible: the 2003 Australian Web Adaptability Initiative (OZeWAI) Conference. Bundoora, Victoria, Australia.
- BALCHIN, W. G. V. & COLEMAN, A. M. (1966) Graphicacy should be the fourth ace in the pack. *Cartographica: The International Journal for Geographic Information and Geovisualization*, 3,1, 23-28.
- BENNETT, D. J. (2002) Effects of Navigation and Position on Task When Presenting Diagrams to Blind People Using Sound *Diagrammatic Representation and Inference*. Springer Berlin / Heidelberg.
- BENTLEY, J. (1986) Programming pearls: little languages. *Communications of the ACM*, 29,8, 711-721.
- BENTLEY, J. L., JELINSKI, L. W. & KERNIGHAN, B. W. (1986) CHEM - A Program for Typesetting Chemical Structure Diagrams. *Computers and Chemistry*. Bell Labs.
- BENTLEY, J. L. & KERNIGHAN, B. W. (1986) GRAP- A language for typesetting graphs. *Commun. ACM*, 29,8, 782-792.
- BENYON, D., TURNER, P. & TURNER, S. (2004) *Designing Interactive Systems - People, Activities, Contexts, Technologies*,

- BERTIN, J. (1983) *Semiology of graphics*, Madison, University of Wisconsin Press, 0299090604.
- BIGHAM, J. P., KAMINSKY, R. S., LADNER, R. E., DANIELSSON, O. M. & HEMPTON, G. L. (2006a) WebInSight: making web images accessible, Proceedings of the 8th international ACM SIGACCESS conference on Computers and accessibility. Portland, Oregon, USA.
- BIGHAM, J. P., MKAMINSKY, R. S., LADNER, R. E., DANIELSSON, O. M. & HEMPTON, G. L. (2006b) WebInSight:: making web images accessible, Proceedings of the Proceedings of the 8th international ACM SIGACCESS conference on Computers and accessibility. Portland, Oregon, USA.
- BLACKWELL, A. F. & ENGELHARDT, Y. (2002) A meta-taxonomy for diagram research. IN OLIVIER, P., ANDERSON, M. & MEYER, B. (Eds.) *Diagrammatic Representation and Reasoning*. London, Springer Verlag.
- BLENKHORN, P. & EVANS, D. G. (1998) Using speech and touch to enable blind people to access schematic diagrams. *Journal Of Network and Computer Applications*, 21,1, 17-29.
- BRAECKMAN, O. (2008) GraSSML financial charts project. *MSc Dissertation, School of Technology*. Oxford, Oxford Brookes University.
- BROWN, A., STEVENS, R. & PETTIFER, S. (2004) Issues in the non-visual presentation of graph based diagrams. *Proceedings of the Eighth International Conference on Information Visualisation*.
- BULATOV, V. & GARDNER, J. (2004) Making Graphics Accessible, Proceedings of the SVG Open 2004. Tokyo, Japan.
- CARD, S. K., MACKINLAY, J. & SHNEIDERMAN, B. (1999) Using vision to think. *Readings in information visualization: using vision to think*. Morgan Kaufmann Publishers Inc.
- CASNER, S. M. (1991) Task-analytic approach to the automated design of graphic presentations. *ACM Transactions on Graphics*, 10,2, 111-151.
- CLEVELAND, W. C. & MCGILL, R. (1984) Graphical Perception: Theory, Experimentation, and Application to the Development of Graphical Methods *American Statistical Association*, 79,387, 531-554.
- COOPER, C., DUCE, D. A., LI, W., SAGAR, M., BLAIR, G., COULSON, G. & GRACE, P. (2005) The Open Overlays Collaborative Workspace, Proceedings of the SVG Open 2005.
- CORNELIS, M. & KRIKHAAR, K. (2001) Guidelines for Describing Study Literature, Available at <http://projects.dedicon.nl/tedub/#Guidelines>. Last accessed 2008.
- DOUG, S. & CULLER, M. (2009) A City Upon a Screen

- Exposing Civic Data Through Accessible Interactive Data Visualizations, Proceedings of the SVG Open 2009. Mountain View, California.
- DRC, D. R. C. (2004a) Formal Investigation report: web accessibility, Available at <http://www.drc-gb.org/publicationsandreports/report.asp>. Last accessed 2009.
- DRC, D. R. C. (2004b) The Web: Access and inclusion for disabled people, Available at <http://joeclark.org/dossiers/DRC-GB.html>. Last accessed 2009.
- DUCE, D. A., HERMAN, I. & HOPGOOD, F. (2002) Web 2D Graphics File Formats. *Computer Graphics Forum*, 21,1, 43-64.
- DUKE, D. J. (2004) Drawing Attention to Meaning. *CyberPsychology & Behavior*, 7,6, 673-682.
- DUKE, D. J., BRODLIE, K. W., DUCE, D. A. & HERMAN, I. (2005) Do You See What I Mean? *IEEE Computer Graphics and Applications*, 25,3, 6-9.
- ELZER, S. & SCHWARTZ, E. (2007) A Browser Extension for Providing Visually Impaired Users Access to the Content of Bar Charts on the Web. , Proceedings of the 3rd WEBIST Conf. on Web Information Systems and Technologies. Barcelona, Spain.
- EUROPEAN, I. S. (2009) e-Inclusion, Available at http://ec.europa.eu/information_society/activities/einclusion/index_en.htm. Last accessed 2009.
- FATHULLA, K. & BASDEN, A. (2007) What is a diagram?, Proceedings of the 11th International Conference Information Visualization.
- FERRAILOLO, J. (2008) How Ajax Changes the Game for SVG, Proceedings of the SVG Open 2008. Nuremberg, Bavaria.
- FERRES, L., VERKHOGLIAD, P., LINDGAARD, G., BOUCHER, L., CHRETIEN, A. & LACHANCE, M. (2007) Improving accessibility to statistical graphs: the iGraph-Lite system, Proceedings of the 9th international ACM SIGACCESS conference on Computers and accessibility. Tempe, Arizona, USA.
- FERRES, L., VERKHOGLIAD, P., SUMEGI, L., BOUCHER, L., LACHANCE, M. & LINDGAARD, G. (2008) A syntactic analysis of accessibility to a corpus of statistical graphs, Proceedings of the 2008 international cross-disciplinary conference on Web accessibility (W4A). Beijing, China.
- FIELDING, R. T. (2000) Architectural Styles and the Design of Network-based Software Architectures. *Information and Computer Science*. IRVINE, USA., UNIVERSITY OF CALIFORNIA.
- FRANKLIN, K. M. & ROBERTS, J. C. (2004) A Path Based Model for Sonification, Proceedings of the Information Visualisation, Eighth International Conference.

- FREEDOMSCIENTIFIC. (2009) JAWS Screen Reading Software, Available at <http://www.freedomscientific.com/products/fs/jaws-product-page.asp>. Last accessed 2009.
- GARDNER, J. & BULATOV, V. (2001) Smart Figures, SVG, and Accessible Web Graphics, Proceedings of the CSUN International Conference on Technology & Persons with Disabilities Conference. Los Angeles, CA.
- GARDNER, J., RANGIN, H. B., BULATOV, V., KOWALLIK, H. & LUNDQUIST, R. (1997) The Problem of Accessing Non-Textual Information On The Web, Proceedings of the International World Wide Web Conference. Santa Clara, CA.
- GILLIES, J. & CAILLIAU, R. (2000) *How the Web Was Born: The Story of the World Wide Web*, Oxford Paperbacks, 0192862073.
- GILSON, O., SILVA, N., GRANT, P. W. & CHEN, M. (2008) From Web Data to Visualization via Ontology Mapping, Proceedings of the Eurographics / IEEE VGTC Symposium on Visualization (EuroVis '08). Eindhoven, Netherlands.
- GOLOVCHINSKY, G., REICHENBERGER, K. & KAMPS, T. (1995) Subverting Structure: Data-Driven Diagram Generation, Proceedings of the 6th conference on Visualization '95.
- GONCU, C. (2009) Generation of accessible diagrams by semantics preserving adaptation. *ACM SIGACCESS Accessibility and Computing*, **93**, 49-74.
- GURR, C. A. (1999) Effective Diagrammatic Communication: Syntactic, Semantic and Pragmatic Issues. *Journal of Visual Languages and Computing*, 10 317-342.
- GWMICRO (2010) Window-Eyes, Available at <http://www.gwmicro.com/Window-Eyes/>. Last accessed 2010.
- HARPER, S., YESILADA, Y. & GOBLE, C. (2006) Proceedings of the 2006 international cross-disciplinary workshop on Web accessibility (W4A). Edinburgh, U.K., ACM.
- HELLER, M. A. (1989) Picture and pattern perception in the sighted and the blind: the advantage of the late blind. *Perception*, 18,**3**, 379-389.
- HENRY, S. L. (2007) *Just ask: integrating accessibility throughout design*, Lulu, 2007, 1430319526
- HERMAN, I. & DARDAILLER, D. (2002) SVG Linearization and Accessibility *Computer Graphics Forum*, 21,**4**, 777-786.
- HORN, R. (2001) Visual language and converging technologies in the next 10--15 years (and beyond), Proceedings of the National Science Foundation Conference on Converging Technologies (Nano-Bio-Info-Cogno) for Improving Human Performance.

- HORSTMANN, M., LORENZ, M., WATKOWSKI, A., IOANNIDIS, G., HERZOG, O., KING, A., EVANS, D. G., HAGEN, C., SCHLIEDER, C., BURN, A. M., KING, N., PETRIE, H., DIJKSTRA, S. & CROMBIE, D. (2004a) Automated interpretation and accessible presentation of technical diagrams for blind people. *The New Review of Hypermedia and Multimedia, Special issue: Accessible hypermedia and multimedia* 10,2, 141-163.
- HORSTMANN, M., MAGEN, C., KING, A., DIJKSTRA, S., CROMBIE, D., EVANS, G., IOANNIDIS, G., BLENKHORN, P., HERZOG, O. & SCHLIEDER, C. (2004b) TEDUB: Automatic Interpretation and Presentation of Technical Diagrams for Blind People, Proceedings of the Conference and Workshop on Assistive Technologies for Vision and Hearing Impairment: State of the Art and New Challenges. Granada, Spain.
- HUYNH, D., MAZZOCCHI, S. & KARGER, D. (2005) Piggy Bank: Experience The Semantic Web Within Your Web Browser, Proceedings of the International Semantic Web Conference (ISWC).
- JAVA (2009) Java Access Bridge, Available at <http://java.sun.com/javase/technologies/accessibility/accessbridge/index.jsp>. Last accessed 2009.
- JENA (2009) JENA: A Semantic Web Framework for Java, Available at <http://jena.sourceforge.net/>. Last accessed 2009.
- KELLEY, J. F. (1984) An iterative design methodology for user-friendly natural language office information applications. 2,1, 26-41.
- KELLY, B., PHIPPS, L. & HOWELL, C. (2005a) Implementing a holistic approach to e-learning accessibility, Proceedings of the ALT-C 2005 12th International Conference Research. University of Manchester, England.
- KELLY, B., SLOAN, D., PHIPPS, L., PETRIE, H. & HAMILTON, F. (2005b) Forcing standardization or accommodating diversity?: a framework for applying the WCAG in the real world, Proceedings of the 2005 International Cross-Disciplinary Workshop on Web Accessibility (W4A). Chiba, Japan.
- KENNEL, A. R. (1996) Audiograf: a diagram-reader for the blind, Proceedings of the second annual ACM conference on Assistive technologies. Vancouver, British Columbia, Canada.
- KERNIGHAN, B. W. (1982) PIC - A Language for Typesetting Graphics. *Software Practice Experience*, 12 1-21.
- KERNIGHAN, B. W. (1991) PIC - A Graphics Language for Typesetting User Manual. IN DOCUMENTS, I. U. P. S. M. S. (Ed.).
- KULPA, Z. (1994) Diagrammatic representation and reasoning. *Machine Graphics & Vision*, 3,1/2, 77-103.

- KURZE, M., PETRIE, H., MORLEY, S., DECONINCK, F. & STROTHOTTE, T. (1995) New approaches for accessing different classes of graphics by blind people, Proceedings of the 2nd TIDE Congress. Paris, Amsterdam.
- LARKIN, J. H. & SIMON, H. A. (1987) Why a Diagram is (Sometimes) Worth Ten Thousand Words *Cognitive Science*, 11 65-100.
- LEWIS, R. (2006) The meaning of 'life': capturing intent from web authors, Proceedings of the 2006 international cross-disciplinary workshop on Web accessibility (W4A): Building the mobile web: rediscovering accessibility? Edinburgh, U.K.
- LOHSE, G. L., BIOLSI, K., WALKER, N. & RUETER, H. (1994) A classification of visual representations. *Communications of the ACM*, 37,12, 36-49.
- LOHSE, J., RUETER, H., BIOLSI, K. & WALKER, N. (1990) Classifying Visual Knowledge Representations: A Foundation for Visualization Research, Proceedings of the Visualization '90. San Francisco, CA, USA.
- MACKINLAY, J. (1986) Automating the design of graphical presentations of relational information. *ACM Transactions on Graphics*, 5,2, 110-141.
- MARRIOTT, K., MEYER, B. & STUCKEY, P. (2004) Towards Flexible Graphical Communication Using Adaptive Diagrams, Proceedings of the Advances in Computer Science - ASIAN 2004.
- MCCATHIENEVILE, C. & KOIVUNEN, M.-R. (2000) Accessibility Features of SVG, Available at <http://www.w3.org/TR/SVG-access/>. Last accessed 2009.
- METATLA, O., BRIAN-KINNS, N. & STOCKMAN, T. (2008) Constructing relational diagrams in audio: the multiple perspective hierarchical approach, Proceedings of the Proceedings of the 10th international ACM SIGACCESS conference on Computers and accessibility. Halifax, Nova Scotia, Canada.
- MICROFORMATS (2005) MICROFORMATS, Available at <http://microformats.org/>. Last accessed 2009.
- MIKOVEC, Z. & SLAVIK, P. (1999) System for Picture Interpretation for Blind, Available at <http://cs.felk.cvut.cz/~xmikovec/bis/interact99/index.html>. Last accessed 2009.
- MILLER, G. (1956) The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information. *Psychological Review*, 63 81--97.
- NARAYANAN, N. H. (1997) Diagrammatic communication: A taxonomic overview. IN KOKINOV, B. (Ed.) *Perspectives on Cognitive Science*. Sofia: New Bulgarian University Press.
- NBA (2000) Excerpts from the NBA Tape Recording Manual, Third Edition, Available at <http://www.w3.org/2000/08/nba-manual/>. Last accessed 2009.

- NCAM (2006) Beyond the Text project: Design Guidelines for Electronic Publications, Multimedia and the Web, Available at <http://ncam.wgbh.org/publications/adm/>. Last accessed 2009.
- NCAM (2008) Effective Practices for Description of Science Content within Digital Talking Books, Available at <http://ncam.wgbh.org/publications/stemdx/index.html>. Last accessed 2009.
- NCTD (2008) RNIB National Centre for Tactile Diagrams, Available at <http://www.nctd.org.uk/index.asp>. Last accessed 2009.
- NEUBERG, B. (2009) SVG in Internet Explorer and at Google, Available at <http://www.svgopen.org/2009/keynotes.shtml>. Last accessed 2009.
- NIELSEN, J. & MOLICH, R. (1990) Heuristic evaluation of user interfaces, Proceedings of the Proceedings of the SIGCHI conference on Human factors in computing systems: Empowering people. Seattle, Washington, United States.
- NOY, N. F. & MCGUINNESS, D. (2000) Ontology Development 101: A Guide to creating your first Ontology. *Stanford KSL Technical Report KSL-01-05*.
- NVDA (2010) NonVisual Desktop Access, Available at <http://www.nvda-project.org/>. Last accessed 2010.
- OMG (2007) XML Metadata Interchange (XMI), Available at <http://www.omg.org/technology/documents/formal/xmi.htm>. Last accessed 2009.
- OU, K. N. (2009) Guidelines for describing visual teaching material, Available at <http://kn.open.ac.uk/public/workspace.cfm?wpid=2709>. Last accessed 2009.
- PATIL, S. R. (2007) Accessible image file formats: the need and the way (position paper), Proceedings of the Proceedings of the 2007 international cross-disciplinary conference on Web accessibility (W4A). Banff, Canada.
- PETRIE, H., HARRISON, C. & DEV, S. (2005) Describing images on the Web: a survey of current practice and prospects for the future, Proceedings of the 3rd International Conference on Universal Access in Human-Computer Interaction International (HCII). Las Vegas, NEVADA.
- PILGRIM, M. (2003) The Vanishing Image: XHTML 2 Migration Issues, Available at <http://www.xml.com/lpt/a/1240>. Last accessed 2010.
- PROTEGE (2009) Open source ontology editor and knowledge-base framework, Available at <http://protege.stanford.edu/>. Last accessed 2009.
- RESKINOFF, S., PASCOLINI, D., ETYA'ALE, D., KOCUR, I., PARARAJASEGARAM, R., POKHAREL, G. P. & MARIOTTI, S. P. (2004) Global data on visual impairment in the year 2002. *scielosp*.

- RIBERA, T. M. (2008) Is the PDF format accessible? *Information Technology and Libraries* 27,3, 25-43.
- RNCB (2009) T3 Talking Tactile Technology, Available at <http://www.mcb.ac.uk/t3/>. Last accessed 2009.
- RNIB (2009) Royal National Institute of Blind People, Available at <http://www.rnib.org.uk/>. Last accessed 2009.
- ROTARD, M. & ERTL, T. (2004) Tactile Access to Scalable Vector Graphics for People with Visual Impairment, Proceedings of the SVG Open 2004. Tokyo, Japan.
- ROTH, P. & PUN, T. (2003) Design and Evaluation of Multimodal System for the Non-visual Exploration of Digital Pictures, Proceedings of the Interact 2003, 9th ICIP TC13 Int. Conf. on Human-Computer Interaction.
- SAP (2005) Accessing Maps, Diagrams, and similar Object-Oriented Graphics, The Science Access Project, Available at <http://dots.physics.orst.edu/graphics.html>. Last accessed 2009.
- SECTION508 (2009) Section 508 of the Rehabilitation Act, Available at <http://www.section508.gov/>. Last accessed 2009.
- SHNEIDERMAN, B. (1996) The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations, Proceedings of the 1996 IEEE Symposium on Visual Languages.
- SLOAN, D., HEATH, A., HAMILTON, F., KELLY, B., PETRIE, H. & PHIPPS, L. (2006) Contextual web accessibility - maximizing the benefit of accessibility guidelines, Proceedings of the 2006 international cross-disciplinary workshop on Web accessibility (W4A): Building the mobile web: rediscovering accessibility? Edinburgh, U.K.
- STROBBE, C. (2008) SVG Accessibility Issues, Proceedings of the SVG Open 2008. Nuremberg, Bavaria.
- SUWA, M. & TVERSKY, B. (2002) External Representations Contribute to the Dynamic Construction of Ideas. *Diagrammatic Representation and Inference*. Springer Berlin / Heidelberg.
- SWAN, H. (2009) Just how accessible is SVG?, Available at <http://www.iheni.com/just-how-accessible-is-svg/>. Last accessed 2009.
- T2RERC (2003) Proceedings from the Stakeholder Forum on Visual Impairment. IN STROBEL, W. & BAUER, S. M. (Eds.).
- TAKAGI, H. & TATSUYA, I. (2007) Technology Advances and Standardization Toward Accessible Business Graphics. *Universal Access in Human-Computer Interaction. Applications and Services*.

- TAKAHASHI, S., MATSUOKA, S., YONEZAWA, A. & KAMADA, T. (1991) A general framework for Bi-directional translation between abstract and pictorial data, Proceedings of the 4th annual ACM symposium on User interface software and technology. Hilton Head, South Carolina, United States.
- TAYLOR, M. (2008a) Describing graphs, charts and diagrams, Available at <http://kn.open.ac.uk/public/workspace.cfm?wpid=2752>. Last accessed 2009.
- TAYLOR, M. (2008b) Flow chart template, Available at <http://kn.open.ac.uk/public/workspace.cfm?wpid=3041>. Last accessed 2009.
- TENNISON, J. (2005) Managing Complex Document Generation through Pipelining, Proceedings of the XTech
- TUFTE, E. (1990) *Envisioning Information*, Graphics Press USA (1990), 0961392118.
- TUFTE, E. (1997) *Visual Explanations: Images and Quantities, Evidence and Narrative* Graphics Press USA (1997), 0961392126.
- TUFTE, E. (2001) *The Visual Display of Quantitative Information* Graphics Press USA; 2nd Ed edition (Jan 2001), 0961392142.
- VIEWPLUS (2009) IVEO Tactile Touch and Audio Learning System, Available at <http://www.viewplus.com/products/touch-audio-learning/IVEO/>. Last accessed 2009.
- VISHWANATH, K., VISWANATH, V., DRAKE, W. & LEE, Y. (2005) OntoDiagram: automatic diagram generation for congenital heart defects in pediatric cardiology. *AMIA Symposium*, 754-758.
- VISIONS (2009) Services for the Blind and Visually Impaired, Available at http://www.visionsvc.org/org_chart_graphical.html. Last accessed 2009.
- VISWANATH, V., TONG, T., DINAKARPANDIAN, D. & LEE, Y. (2006) Ontological modeling of transformation in heart defect diagrams. 799–803.
- VORBURGER, M. (1999) Altifier: Web Accessibility enhancement tool.
- W3C (1999) Web Content Accessibility Guidelines 1.0, Available at <http://www.w3.org/TR/WAI-WEBCONTENT/#content-structure>. Last accessed 2009.
- W3C (2000) Authoring Tool Accessibility Guidelines 1.0, Available at <http://www.w3.org/TR/ATAG10/>. Last accessed 2009.
- W3C (2001) Scalable Vector Graphics (SVG) 1.0 Specification, Available at <http://www.w3.org/Graphics/SVG/>. Last accessed 2009.
- W3C (2002) User Agent Accessibility Guidelines 1.0, Available at <http://www.w3.org/TR/UAAG10/>. Last accessed 2009.

- W3C (2004a) Architecture of the World Wide Web, Volume One, Available at <http://www.w3.org/TR/2004/REC-webarch-20041215/>. Last accessed 2009.
- W3C (2004b) Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies 1.0, Available at <http://www.w3.org/TR/CCPP-struct-vocab/>. Last accessed 2009.
- W3C (2007) Gleaning Resource Descriptions from Dialects of Languages (GRDDL), Available at <http://www.w3.org/TR/grddl/>. Last accessed 2009.
- W3C (2008a) RDFa Primer: Bridging the Human and Data Webs, Available at <http://www.w3.org/TR/xhtml-rdfa-primer/>. Last accessed 2009.
- W3C (2008b) SPARQL Query Language for RDF, Available at <http://www.w3.org/TR/rdf-sparql-query/>. Last accessed 2009.
- W3C (2008c) Web Content Accessibility Guidelines 2.0, Available at <http://www.w3.org/TR/WCAG20/>. Last accessed 2009.
- W3C (2008d) The World Wide Web Consortium (W3C) Available at <http://www.w3.org/>. Last accessed 2008.
- W3C (2009) HTML5 A vocabulary and associated APIs for HTML and XHTML, Available at <http://dev.w3.org/html5/spec/Overview.html>. Last accessed 2009.
- W3C (2010) Accessible Rich Internet Applications (WAI-ARIA) 1.0, Available at <http://www.w3.org/TR/2010/WD-wai-aria-20100916/>. Last accessed 2010.
- WAI (2009) Web Accessibility Initiative (WAI), W3C, Available at <http://www.w3.org/WAI/>. Last accessed 2009.
- WALL, S. A. & BREWSTER, S. A. (2006) Tac-tiles: multimodal pie charts for visually impaired users, Proceedings of the Proceedings of the 4th Nordic conference on Human-computer interaction: changing roles. Oslo, Norway.
- WEB, A. F. A. (2002) How to Create Descriptive Text for Graphs, Charts & other Diagrams, Available at <http://www.cew.wisc.edu/accessibility/tutorials/descriptionTutorial.htm>. Last accessed 2009.
- WEBAIM (2009a) Survey of Preferences of Screen Readers Users, Available at <http://webaim.org/projects/screenreadersurvey/>. Last accessed 2010.
- WEBAIM (2009b) Web Accessibility in Mind, Available at <http://www.webaim.org/>. Last accessed 2009.
- WEBAIM (2010) Creating Accessible Flash Content, Available at <http://webaim.org/techniques/flash/>. Last accessed 2010.
- WHITNEY, G. & KEITH, S. (2008) European Developments in the Design and Implementation of Training for eInclusion Springer Berlin / Heidelberg.

- WILMOT, P. D. (1999) Graphicacy as a form of communication. *South African Geographical Journal*, 2,**81**.
- WILSON, B. (2008) MAMA - the "Metadata Analysis and Mining Application", Available at <http://dev.opera.com/articles/view/mama/>. Last accessed 2009.
- XBRL, I. C. (2007) eXtensible Business Reporting Language, Available at <http://www.xbrl.org>. Last accessed 2009.
- YU, W., REID, D. & BREWSTER, S. A. (2002) Web-based multimodal graphs for visually impaired people, Proceedings of the 1st Cambridge Workshop on Universal Access and Assistive Technology (CWUAAT). Cambridge, England.

Appendix A

Graphical Formats on the Web

There exist many file formats for diagrams. Those file formats can be grouped into two main types: raster and vector. The choice on either depends on its application; some file formats are more appropriate than others depending on the context they are used in. A review of the most commonly found formats on the web is presented as well as their main characteristics and their limitations. A review of Web file formats is presented in (DUCE et al., 2002).

The following raster vs. vector format discussion is intentionally nuanced as to the context intended in this thesis which focuses on diagrams.

Most of today's web graphics are raster-based formats (e.g., GIF, PNG, and JPEG). Despite its advantages for photographic images, this earliest support for web graphics presents many limitations for other types of graphics such as diagrams. Vector graphics formats (e.g. SVG, WebCGM and SWF), on the other hand, present many advantages over raster-image formats for representing graphics such as diagrams.

A.1 Raster formats

Raster-based formats consist of a rectangular array of pixels. The information describing every individual pixel (colour, opacity, etc.) is needed to render the graphic.

These are appropriate for some applications, for example sharing photographs on the Web, but they also have several limitations for other applications such as rendering diagrams. Their limitations can be summarized as follow:

- ❖ **Time consumed in downloading large files:** The size of raster formats is important which can be an issue for their transmission. To tackle this issue, two kinds of compression technique are implemented depending on the file format: lossless and lossy. With lossless compression, the file size is reduced without losing image quality whereas lossy compression involves losing some information depending on the level of compression.

- ❖ **Resolution dependent:** Raster graphics cannot be scaled without losing quality. Indeed, being a resolution dependent format, expanding the size will impact on image quality.
- ❖ **Not searchable:** Raster graphics are not searchable as they do not carry the information represented in the image but only the information concerning the pixels to be rendered. The information concerning the diagram represented is lost. To overcome this limitation, optical character recognition (OCR) techniques are used to recover the lost information, but these techniques present some limitations themselves (i.e. noise might engender some mistakes from the OCR).

Raster graphics are stored as a rectangular array of pixels. The structural information of the diagram is lost. Although metadata can be injected in some raster formats, this facility imposes some constraints as the content can only be described as a separate part of the file format and not at the individual level of objects.

- ❖ **Inability to style or make changes to the image:** Very often the modification of a diagram represented as a raster diagram requires recreating the diagram as it is either not possible to alter it or simply too time consuming. This is because it is not possible to edit individual part of such raster graphics as separate objects. Raster graphics do not have the ability to be styled as no separation between content and presentation is made.
- ❖ **Inability to link or interact with the image:** Although, linkage to some part of the raster graphic is possible in HTML using image maps which allows areas of the image to be hyperlinked to further information. It is difficult to link and interact with raster graphics as only an access to a rendered version of the information is available.

The three main raster formats used on the web are GIF, PNG and JPEG

A.1.1 Graphic Interchange format (GIF)

This format is suitable for simple computer graphics such as simple illustrations, shapes, logos or diagrams with at most 256 colours. This format is widely supported and used on the web as it can be efficient with low numbers of colours. It also offers transparency, interlacing and limited animation support. It compresses the graphic without losing data or distorting it.

A.1.2 Portable Network Graphics (PNG)

PNG is a free open source lossless file format. Images saved as PNG are efficiently compressed. PNG was developed as an alternative to GIF and provides full colour (up to 24 bit in colour) support and is acceptable for real world images. PNG, which is a recommendation of the W3C and an ISO/IEC Standard, was designed to be used on the web.

PNG allows a short text description of the graphic to be embedded. This information can be accessed by search engines.

A.1.3 Joint Photographic Experts Group (JPEG)

JPEG is the standard format for photographs on the web as it supports a full-colour image (16.7 million colours) which produces good results for this type of graphics. JPEG uses the lossy compression technique which offers a good compression ratio but unlike GIF/PNG, JPEG loses some of its information when compressed. Progressive JPEG allows a foggy view of the entire image while the rest of the image is downloaded.

This format allows metadata to be embedded within the file header offering additional information about the image.

A.2 Vector formats

Vector formats define a rich set of geometric objects, which can be rendered to produce a visual image. The vector objects are interpreted and then drawn by translating them into a raster image if displayed on a raster display. Vector formats present many advantages over image format:

- ❖ **Resolution independent:** Vector formats are fully scalable without loss of resolution. It is possible to alter the size without any impact on the quality as the geometry adjusts to the changes. So, they can be zoomed and resized as needed as they adapt well to arbitrary resolution.
- ❖ **Smaller size:** Files are “generally” smaller and the size is independent of resolution, so they can often be downloaded and viewed faster than raster images.
- ❖ **Searchable:** Additionally, the text of the graphics can be searched when text is represented as text strings (it is sometimes not the case). Indeed, text is also a vector object, so it can be accessed by search engines. The text can also be

selected, copied, edited and modified. Additional information (metadata) can be added and made an integral part of the structure of the graphics.

Both proprietary standards such as SWF and open standards such as WebCGM and SVG can be found.

A.2.1 Computer Graphics Metafile (CGM or WebCGM)

CGM, an ISO/IEC standard, is a file format for 2D vector graphics, cell-array graphics and text. CGM provides a rich set of primitives and attributes to describe graphics.

It allows 2D graphics data interchange. CGM uses profiles to enable it to be tailored to the needs of specific application areas in well-controlled ways.

WebCGM is a profile of CGM and an ISO/IEC standard for 2D vector graphics on the Web. It was developed by CGM Open in collaboration with the W3C. It is a recommendation of the W3C. WebCGM is mainly used in technical documents.

A set of metadata in WebCGM allows the support of hyper linking and links to the application data, layering, document navigation, picture structuring, search and query of the graphical information.

WebCGM does not have an XML representation and does not make use of CSS, which can be seen as a limitation for some applications.

With WebCGM 2.0 which became a W3C Recommendation and an Oasis Standard in 2007, metadata are XML-based using the XML Companion File (XCF) and WebCGM objects can be accessed using the DOM.

A.2.2 Scalable Vector Graphics (SVG)

SVG is a vector graphic open standard developed by the W3C and heavily influenced by the PostScript/PDF rendering model and primitive/attributes (FERRAILOLO, 2008).

The emergence of this W3C Recommendation (W3C, 2001) has changed the way 2D graphics are created on the web. SVG started off as a popular web format for desktop computers and aimed at being a major vector graphics format for the web. It has been widely supported by mobiles, authoring tools and browsers. Adobe, which provided the Adobe SVG viewer (ASV) for IE, terminated its development and support. The lack of support of SVG in IE (SWAN, 2009) has been seen as the biggest hold-up in its popularity. However, this has been addressed by Google

(NEUBERG, 2009) who provide SVG support in all browsers (e.g. Safari, Opera, Firefox, IE) using either the native SVG support or “SVG Web” which is an advanced JavaScript library that provides support for SVG.

Unlike WebCGM, SVG is an XML language and makes use of the CSS mechanism supported on the Web.

A. Advantages and Accessibility features of SVG

SVG has introduced some accessibility features (MCCATHIENEVILE and KOIVUNEN, 2000) that raster formats do not offer (STROBBE, 2008). When using SVG, information about the graphic is available to the browser in terms of the objects it is composed of.

SVG presents many advantages and accessibility features, some originating from the vector graphic model, some inherited because SVG is an Extensible Markup Language (XML) based language and some are part of the design of SVG itself.

- Origination from the vector graphic model
 - SVG inherits the advantages previously mentioned, such as resolution independence, small file size, scalable, zoomable, plain text format and maintainable.
 - The structural information of the diagram could be an integral part of the graphic.
 - Created graphical objects can be reused, making the structure of complex graphics easier to manage and understand.
- Being an XML based language.
 - SVG is encoded as plain text offering new possibilities for assistive technology (text contained in text elements are kept intact as such).
 - SVG offers all the advantages of XML: interoperability, Unicode support, wide tool support, textual nature of XML (compression).
 - SVG can be used as an XML namespace, so can be used in conjunction with other XML languages such as MathML, SMILE or XHTML for example. It also integrates well with other web technologies (DOM, CSS, XSLT, XPath, etc.). This allows appropriate language markups to be used suitably by authors, increasing accessibility.
 - Content can be adapted to the characteristics of different clients through the use of styling and scripting. CSS2 and XSLT can be used to modify

the presentation of an SVG graphic which is important for accessibility. The rendering can be adapted to the needs of the user. This can be done in different ways: style definition, using classes and grouping or by providing style definitions for different media. CSS2 and XSLT allow the user to modify the presentation of the graphics (e.g. colour, visibility). SVG supports dynamic interactivity having Scripting potential, (EcmaScript).

- The information can be enriched by metadata which help provide more information about the graphic and can help the search for information.
- Interaction becomes possible as an access to the DOM allows assistive technology to access the underlying structure of the graphic.
- Part of the SVG design itself
 - Alternative equivalent descriptions at the individual graphical element level allow authors to include a textual description and a title for each logical component of graphic in order to convey its role in the graphic.
 - It is possible to provide alternatives based on whether a feature is supported by using the ‘*switch*’ element and its defined attributes (requiredFeatures or systemLanguage).
 - SVG provides a number of style features beyond the properties defined in CSS. These provide the ability for author to create content that can be adapted to users needs; examples of such features include masking, filters, etc.

B. Limitations of SVG

Nevertheless, a number of limitations remain, as SVG graphics are not automatically accessible.

- **Structure might not be suitably authored:** Even though SVG stores structural information about the graphic as an integral part of the graphic, the amount of structure is mainly author dependent. The author has to provide a well constructed structure. Some SVG documents are very badly structured and therefore less informative. So occasionally, re-ordering the hierarchical structure of the SVG to organize objects into proper groups may be necessary. The structure of the SVG may reflect the sequence of operations used to create the diagram, rather than the intrinsic object

structure within the diagram itself. This is likely to be the case for diagrams created with a general-purpose drawing tool. If not created using such a tool, the final structure is dependent on the author. Some drawing tools embed additional information in an SVG document so that higher level structure (e.g. connectivity information) can be recovered by the drawing tool when the document is subsequently modified. Of course one could argue that SVG being an XML language, XSLT could be used to achieve the transformation.

- **Presents information at a low level of abstraction:** SVG does not capture diagrams at a high enough level of abstraction. It is more a “final form” presentation, which has some drawbacks in the direct creation of complex, highly structured, diagrams. It may be difficult at this level to handle resizing (which in general may involve a change to the layout of the diagram, not just a change of scale) and re-positioning of different shapes in complex diagrams. A simple modification such as changing the layout of a tree from vertical to horizontal can be awkward since the coordinates of graphical elements have to be changed other than by application of an affine transformation (Figure 97).

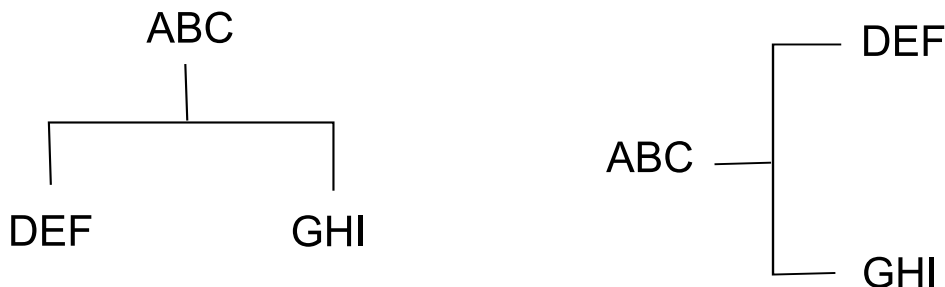


Figure 97: Example of simple modification in SVG

- **No adaptability possible:** SVG does not allow ‘flexible’ readjustment of layout in response to viewer requirements and the viewing environment such as different screen formats (PDA, mobile phone, etc.). The issue is that the intentions of the author are not totally captured. This issue is partially addressed with the “switch” statement which along with some specific attributes provides the ability to specify alternative viewing depending on user’s capabilities or language. So an alternative can be provided depending on whether or not a feature is supported.

- **Limit in generating alternative presentations:** Although SVG stores structural information about graphical shapes as an integral part of the SVG document and allows metadata to be attached to primitives, there is little real scope for generating alternative presentations from a description at this level. If a document is not properly structured (e.g., careful use of <g>elements), it becomes very difficult to provide alternative presentations even if alternative descriptions are provided for individual elements. Most editors do not encourage authoring accessible SVG by correctly grouping elements or prompting for “title” or “desc” attributes for example.
- **Implicit semantic of the information:** An additional issue is that an SVG document contains the semantics of the diagram only implicitly. Authors can include a text description for each logical component and a text title to explain the component’s role in the diagram, but this approach of “alternative equivalents” could become tedious for the creation of complex diagrams and also if the metadata added by the author are not accurate enough, the semantics of the diagram could differ from the description obtained from this metadata.
- **Problems with keyboard accessibility:** SVG does not provide keyboard access support.

A.2.3 Small Web Format (SWF)

SWF is a binary file format intended to deliver vector graphics and animations on the web. SWF files can be generated from different Adobe products, Flash being an example.

It delivers vector graphics, text, audio (version 3), video (version 6) over the internet. The file format SWF was initially designed by Future Wave Software; it was then purchased by Macromedia and finally acquired by Adobe in 2005.

Adobe made the SWF format a partially open format in 2008. The Flash technology is widely used on the web. SWF files can be played by the Adobe flash player (either as a browser plugin or standalone player).

Being of multimedia nature Flash has the ability to create highly accessible content: multiple ways of presentation (audio, text, graphic, video, etc.), scalable, keyboard accessible, extracting audio from the animation is possible replacing the need of a screen reader.

Flash can be made accessible but to take advantage of these abilities, many issues have to be taken into account while creating the files. For example, to address blindness issues, the creator should ensure screen reader accessibility or provide an accessible alternative, ensure keyboard accessibility, do not interfere with screen reader audio or keyboard commands, provide textual equivalent for all non-text elements.

Flash can be made accessible if the Flash developer develops it with accessibility issues in mind, such issues are exposed in many tutorials aiming at directing such development (e.g. WebAIM article (WebAIM, 2010)).

SWF presents various drawbacks. It is not supported by any browser and requires the use of a plugin. It is in a binary format which is not human readable. It does not support cubic Bezier curves, font definition, has limited zooming and limited filtering.

SUMMARY OF GRAPHICAL FORMATS and ACCESSIBILITY of DIAGRAMS on the WEB

	Vector Formats: “set of geometric objects”			Raster Formats: “rectangular array of pixels”		
Advantages	Appropriate for charts, technical drawings, logos, etc. Resolution independent: fully scalable without loss of resolution Generally small files size independent of resolution Searchable when text is represented as text strings Metadata can be added and made an integral part of the structure of the graphic			Appropriate for photographs and images using complex shading		
Limitations	Not appropriate for a photograph			File size, slow downloading speed Resolution dependent: expanding will change alter the quality Structural information of the diagram is lost as only data describing individual pixels of the image is available, inability to search for the content of the image Linkage to some part of raster graphic possible but not easy Not possible to edit individual part of such vector images as separate objects Appearance of the graphics cannot be styled		
	SVG	WebCGM	SWF	PNG	GIF	JPEG
Characteristics	W3C Recommendation open format XML Based language Uses CSS and DOM Native support in most browsers	A profile of CGM W3C recommendation and ISO standard Mainly used In technical documents Metadata which can be added are XML based using the XML companion File (XCF) and support the DOM There is no XML representation of WebCGM WebCGM does not use CSS	Partially open Binary file format Most used format on the web for displaying simple animated vector graphics Browser plugin required	Open file format W3C recommendation and ISO/IEC standard Efficient lossless compression technique Supports 24-bit true-colour images Acceptable for real world images Offers transparency and better interlacing	Best suited for simple illustrations, logos or diagrams with only solid colours or area of uniform colours Fixed 256 colours palette maximum Lossless compression technique Offers transparency interlacing and limited animation	Best suited for Photography Support full colour images (16.7 million colours) Lossy compression technique JPEG2000 also offers lossless and lossy compression.
Accessibility features	Alternative descriptions Title Desc g	Metadata can be added Content Layerdesc gnode	Keyboard accessible Multiple way of presentation Text equivalent	Mechanism for adding descriptive text to the image		Supports metadata which can be embedded in the file header

Appendix B

Accessibility

B.1 Definition

The World Wide Web (WWW) is an important resource of information which has an impact on our every day life: education, commerce, finance, politics, etc. The World Wide Web Consortium (W3C), which is the international consortium developing Web standards, aims at “leading the World Wide Web to its full potential”. It has defined four main goals with the first one being a “Web for Everyone”, which involves making the benefits the WWW offers “*available to all people, whatever their hardware, software, network infrastructure, native language, culture, geographical location, or physical or mental ability*” (W3C, 2008d).

In other words, accessibility involves providing access to this information for ALL, disregarding their disability, environment and technology.

The word “accessibility” is used here in a sense that is illustrated by the following quotation (WAI, 2009): “*Web accessibility means that people with disabilities can perceive, understand, navigate, and interact with the Web, and that they can contribute to the Web. Web accessibility also benefits others, including older people with changing abilities due to ageing. Web accessibility encompasses all disabilities that affect access to the Web, including visual, auditory, physical, speech, cognitive, and neurological disabilities ... a key principle of Web accessibility is designing Web sites and software that are flexible to meet different user needs, preferences, and situations.*”

Web accessibility has to be seen as a collective effort to improve the web, not only by web developers who create the web content but also by user agents (web browsers, assistive technologies (e.g. screen readers) used to access web content (W3C, 2002) and by the web software producers who provide tools (W3C, 2000) that support and encourage accessible web design by supporting and propagating guidelines developed by specific accessibility initiatives within the W3C, such as the WAI (WAI, 2009). All over the world, legislation has been put in place to ensure that efforts are made by all.

B.2 Legal Frameworks

Many laws and policies have been passed regarding web accessibility. Web accessibility has been long seen as a social responsibility but the development of new laws makes it a legal requirement. Many countries around the world (France, UK, United State, New Zealand, etc.) have enforced web accessibility by setting up specific legislation on the matter.

In the U.S, the Rehabilitation Act and the Americans with Disabilities Act (ADA) are civil rights laws protecting disabled people from discrimination. The ADA is more general, it does not deal directly with accessibility of the internet. It is more a law to make sure disabled people have equal opportunities for education, employment, transportation and inclusion. In 1998, an amendment to section 508 (SECTION508, 2009) of the Rehabilitation Act instructs the federal agencies to make their Electronic and Information Technology (EIT) accessible to people with disabilities, which includes web sites.

In the UK, in 1995, the Disability Discrimination Act (DDA), part III aimed to tackle the discrimination many disabled faced. All providers of goods, facilities and services to the general public are required to take into account disable people. In 1999, amendment to this Part III mentions explicitly Web Accessibility requiring web sites to be accessible.

More recently, accessibility follows a path from a legal obligation into an opportunity for society inclusion. The European Union looks at accessibility in a different perspective through the “e-inclusion” initiative (EUROPEAN, 2009): an opportunity rather than a solution to be imposed by legislation.

This initiative aims at guiding society towards the establishment of an “Inclusive Information Society” taking into account the whole range of population differences: age (e-Ageing), disabilities (e-Accessibility), education (e-Competences), geographical location. It does so by investing in research for the development and deployment of technologies, by developing policies, and by promoting best practices. ICT is seen as essential (WHITNEY and KEITH, 2008), an opportunity for both individuals and organisations to overcome traditional barriers to acquire information much needed by all to be fully involved and play an active role in society (e.g. screen readers allowing blind users to access current news, vote, pay taxes, live

independently, etc.). Europe wants to be an “Accessible Information Society”, e-accessibility, by promoting ICT accessibility, especially web accessibility.

B.3 Accessibility Guidelines

Accessibility has become a very important issue for the World Wide Web (WWW), and even more so, since the report on Web Accessibility from the UK’s Disability Rights Commission in 2004 (DRC, 2004b).

The World Wide Web Consortium (W3C) is one of the leading organisations in promoting Web Accessibility and in developing guidelines and techniques to encourage the development of accessible web content through work from the W3C Web Accessibility Initiative (WAI).

Web Accessibility guidelines have been defined to support and promote web accessibility. These guidelines have been set by Web accessibility standards. The W3C Web Accessibility Initiative (WAI) (WAI, 2009) developed the Web Content Accessibility Guidelines (WCAG) (W3C, 2008c, W3C, 1999). The WCAG have been developed over several years involving many participants including experts and individuals sharing the same interest. They are the most widely accepted recommendations to ensure web content accessibility. The first version of the WCAG, WCAG 1.0 (W3C, 1999) provided techniques to achieve accessibility which were mainly related to HTML. The second version WCAG 2.0 (W3C, 2008c), which became a recommendation in December 2008 aimed at being more general by taking into account different advanced technologies that have made their place in today’s web content and more precise so that compliance could be tested automatically. WCAG 2.0 focuses on four principles of accessibility: Perceivable, Operable, Understandable and Robust (POUR) aimed at providing a flexible framework to encourage conceptual thinking throughout the process of designing accessible web content. Some techniques are presented as separate documents.

Many other organisations have developed guidelines based on the WCAG. Another well known guideline is the one aiming at ensuring the implementation of the section 508 (SECTION508, 2009) of the U.S Rehabilitation Act of 1973 which was reinforced in 1998. Both the WCAG and section 508 have developed a set of guidelines and techniques to address issues on web content accessibility.

There are different kinds of disabilities. The four main types are:

- Sight: blindness, low vision, colour-blindness

- Hearing: deafness
- Motor: inability to type on a keyboard or/and manipulate, control a mouse, inability to click on small links
- Cognitive: clinical disability (includes autism, dementia, Down syndrome or traumatic brain injury) and functional disability (difficulties with memory, problem solving, attention, reading, linguistic, verbal comprehension, visual comprehension, mathematical comprehension).

Each type requires a certain amount of consideration when designing web content. It is important to understand that such considerations benefit not only disabled people but everyone finding themselves in a situation where one or many of their senses might be impaired (e.g. broken arm (motor), screen too small to display graphic (sight), noisy environment (hearing), working in parallel on another task (cognitive), etc.).

B.4 Blindness

The term blind is used in the following sense: blindness is defined as a total absence of useful vision. It involves loss of useful sight caused by various origins (inherited, disease, diet, etc.).

Blind cannot make use of external visual representations. Some visually impaired who are not clinically and socially categorized as blind but do not make use of external visual representation are also considered on this research, as they depend on other means of access than visual.

According to (RESKINOFF et al., 2004) in 2002, there were 161 million visually impaired people in the world, of whom about 37 million were blind. The RNIB (RNIB, 2009) states that there are two million people with sight problems in the UK of which 183,693 are registered as blind. The RNIB also specify that this number is an under estimate as many blind people do not register themselves as blind either because they are not aware of the services they are entitled to or they simply do not want to.

Blind users are not able to see the screen so they do not rely on the computer monitor and mouse that others use everyday to interact with computers. Blind users nearly always rely on screen readers to access, navigate and interact with the information presented to them on computers. Screen reader users mainly use their keyboards as a way of navigating the web. As well as converting text into speech,

screen readers can also convert text into Braille characters on Braille devices for users who might prefer Braille access or users who are deaf-blind.

Screen readers provide access to speech or Braille devices. It is important to mention the difference between blind people that were born blind and those that became blind later in life. People with congenital blindness and later in life blindness have different ways of treating information (HELLER, 1989, KENNEL, 1996) as they have different personal knowledge. Blind people who were not born blind recognise and interpret graphical information much more easily.

B.5 Screen Readers

A screen reader is an assistive technology that makes information on the computer screen accessible to blind users. It can present this information either in speech or in Braille on a refreshable Braille device. A screen reader often offers functionalities through keyboard keys to interact with most of the information presented (e.g. manage the flow of text, direct access to lists, navigation of tables, etc.). The most popular screen readers are Jaws (FREEDOMSCIENTIFIC., 2009), Window Eyes (GWMicro, 2010) and NVDA (NVDA, 2010) (Figure 98).

In the UK, agencies recommend NVDA and once the person is proficient he can judge if paying for another one is worthwhile. In December 2008- January 2009, Web Accessibility in Mind (WebAIM, 2009a) conducted a survey of preferences of screen reader users. WebAIM specifies that the sample of screen reader users was not controlled and so not representative of all screen reader users. Out of the 1121 persons who took part at the survey, 74% use Jaws, 23% use Window Eyes and 8% use NVDA.

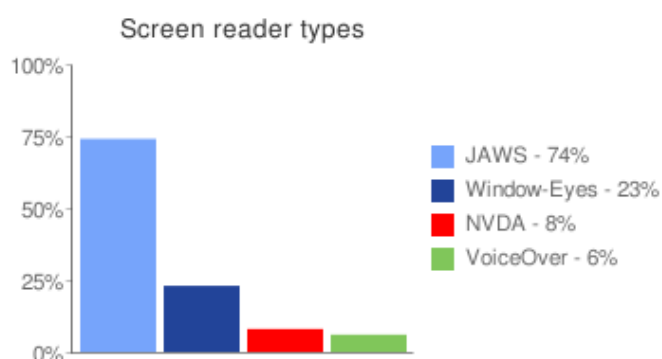


Figure 98: Screen Reader Usage (WebAIM, 2009a)

When using a screen reader to access information on a web page, the interaction experienced by the blind user, will depend on how well the web page is encoded.

Users of screen readers are presented the information in a linear way, generally from top left to bottom right, one line at a time, or using other navigation methods for example through the links, the frames, the headings, etc.

Using these assistive technologies, blind users gain independence. They do not depend any more on others to access information. It is true for textual information but is it true for other kinds of information: what about access to graphical information? What about diagrams?

Appendix C

Accessibility of Diagrams

C.1 Diagrams on the Web

Currently it is possible to insert both vector graphics and vector images (raster graphics) within a web page.

C.1.1 Diagram inserted as vector images

Vector images appear in web pages via the `` element in HTML. The location where the image is stored is specified using the `src` attribute. The author could add an alternative text for the image used to provide an equivalent alternative in case the image cannot be seen.

```
e.g. 
```

Depending on the context, HTML offers different possibilities to add such alternative descriptions using the *alt-text* and *longdesc* attributes within the `` element. If the description cannot be done briefly then an *alt text* is not enough, so it is necessary to either:

- Use the `longdesc` attribute to link the vector image to a long description.
- For user agents who do not support `longdesc`, the use of a d-link, which is a link to a long description of the vector image, should be added after or next to the image.

C.1.2 Diagram inserted as vector graphics

Vector graphics (e.g. SVG, SWF) can be embedded in web pages via either the `<embed>` element, the `<iframe>` element or the `<object>` element. However each of these are supported differently by browsers (STROBBE, 2008). Introduced by Netscape, the `<embed>` element, even though supported by most browsers, has never been accepted in HTML standard until it appeared in the HTML 5 working draft (W3C, 2009). The `<iframe>` element is not supported by XHTML 1.1 (Strict) and is slowly being replaced by the `<object>` element.

For the `<object>` element to be accessible to screen readers, descriptive text, HTML or other information should be enclosed within the object body. A text

equivalent can be provided by inserting it between the opening and closing tag as seen in the following example:

```
e.g. <object data="MyImage.svg" type=" image/svg+xml ">  
Text equivalent for my vector graphic...  
</object >
```

In the event the <object> element cannot be displayed the code between its opening tag (<object>) and closing tag (</object>) is executed. This is called the fall-back mechanism and should be used to provide an alternative way of presenting the information.

When using <object> it is also possible to provide longer descriptions within the element's content. Unlike the alt-text attribute of the element, alternative descriptions provided within the content of the object element are not limited in size and can include markup, adding more flexibility to the type of information provided (e.g. links). One extra advantage with the <object> element is to provide alternative representations of information by embedding other object elements within the content of the current object element.

C.2 Guidelines for diagrams accessibility

At the time of writing this thesis, it is believed that there are no specific guidelines for making vector graphics accessible on the web. Guideline 1.1 from the WCAG 2.0 (W3C, 2008c) attempts to address the accessibility problem of diagrams represented as vector graphics or as vector images ("any non-text" content) by recommending the use of text alternatives. Such recommendations present some limitations.

C.2.1 WCAG 2.0 – Guideline 1.1 Text Alternatives

In the context of web graphics accessibility, the main guideline (Guideline 1.1 Text Alternatives in WCAG 2.0) states that it is necessary to ***“Provide text alternatives for any non-text content (includes for example pictures, charts, applets, audio files...) so that it can be changed into other forms people need, such as large print, Braille, speech, symbols or simpler language.”***, which involves providing an alternative textual description of the information represented graphically. When possible, it is also recommended to provide a description of the image within the general text.

As mentioned earlier if a diagram has been represented as a vector graphic and inserted using the <object> element, a text alternative should be provided using the

fall back mechanism that the <object> element allows. If the diagram is represented as a vector image and inserted within a web page using the element, a text alternative should be provided.

C.2.2 Limitations of Text Alternatives

Despite the efforts in developing Web accessibility standards, still many designers either ignore them or simply do not follow them properly (DRC, 2004b).

The two main problems concerning the description of vector images on the web (DRC, 2004a) are stated as:

- The omission of alt text attribute on many images
- The lack of useful information in the case of images which include a description as an alt-text and/or longdesc attributes.

MAMA (WILSON, 2008), the “Metadata Analysis and Mining Application”, is a structural Web-page search engine developed by Opera. MAMA examined 3,509,180 URLs in 3,011,668 domains. Out of the set analysed, 91.74% used the IMG element and out of this 78.30% included an Alt attribute. This research does not specify the validity or nature of the Alt attribute (e.g. valid and usable, empty) but its frequency. Studies by Jeffrey P. Bigam (BIGHAM et al., 2006b) found that out of 500 most high traffic websites, only 39.6% of significant images found on their homepages were assigned alternative text.

Some tools have been developed to facilitate the addition of the alt text by formulating, when possible using the context, and providing alt text automatically to images (WebInSight (BIGHAM et al., 2006a) and Altifier (VORBURGER, 1999)), but these tools work only on some images, for the remainder human labelling is suggested and encouraged.

Web Accessibility standards have provided an important core in ensuring a more accessible web but as stated by Brian Kelly et al. (KELLY et al., 2005b) they prove challenging in implementation in certain situations. As an example an alt text attribute can be present but how do you validate that the alternative text describing the image is an appropriate alternative?

Sometimes even if they are used they are not usable or carry the wrong alternative text: as Kelly et al. (KELLY et al., 2005a) point out the “*ALT tab merely names, not explains an image.*” So, if it is not the case it will still conform to the

WCAG but won't be usable (i.e. the alt-text description is not useful). In this case technical accessibility would be satisfied but usable accessibility will not be achieved.

In an attempt to deal with all these issues, which they support in a survey of 100 homepages, Helen Petrie et al. (PETRIE et al., 2005) presented the initial stage of a program to encourage and improve the ALT-text descriptions by producing a set of standard guidelines to produce useful descriptions of images. There are different kinds of images used on the web, decorative images to navigate and structure a web page and there are images that convey content information to understand a web page. Vector images of diagrams are images which convey content information. Such images can be classified in two group "simple" and "complex" vector images.

The proposed guidelines allows decorative and simple images to be made accessible using appropriate alt tags for short descriptions or longdesc attributes for long description. Some (non-standard, non-official) guidelines have been produced in an attempt to guide the creation of alternative descriptions of diagrams (CORNELIS and KRIKHAAR, 2001, OU, 2009, NCAM, 2008, NBA, 2000, NCAM, 2006, AULT et al., 2002).

Most of the issues ALT-text presents have been recognised and standardised. Indeed, the HTML5 specification (W3C, 2009) provides a set of requirements (part 4.8.2.1 of the HTML5 specification) for providing text to act as an alternative for images depending on what the image is intended to represent. But all these are not enough (PATIL, 2007) for complex vector images of diagrams which are sometimes "worth a thousand words" (LARKIN and SIMON, 1987). Indeed alternative descriptions of complex diagrams are not trivial to produce and their quality depends on the expertise of the person authoring it. Depending on the author of the description the process can be complex and costly in term of time and effort.

Descriptions which are too long and too hard to make sense from can become confusing, tedious to read and hard to follow due to limited working memory (MILLER, 1956). The distinction between overview and details is hard. Such description requires a reader to listen to a whole description even if only some specific information from the diagram is needed and vice versa. This makes access to complex diagrams an issue. Furthermore, some diagrams are really difficult to describe or simply are inappropriate for textual representations as they provide complex information which is not always accessible sequentially (LARKIN and SIMON, 1987, MILLER, 1956).

C.2.3 Screen Readers and diagram accessibility

Blind users are known to face a certain number of barriers while accessing the WWW (WAI, 2009) inaccessible graphics and poorly described images with inadequate description are some of them.

A screen reader is an assistive technology that makes information on the computer screen accessible to blind users. A much detailed description of screen readers can be found in (Section B.5). Screen readers have some limitations, they cannot describe diagrams by themselves and they rely on alternative description information attached to the vector image. Diagrams represented as vector graphics also present some challenges for screen readers.

A. Diagrams as vector images on the web

When the screen reader finds a vector image on a web page inserted using an `` tag in HTML, the screen reader looks for the *alt* attributes, three situations can happen:

- If the alt-text attribute is present, the screen reader will read the attribute which should in ideal cases provide a useful description of the vector image of the diagram
- If the alt-text attribute is an empty string, the screen reader ignores the image
- If the alt-text is missing, the screen reader informs the user that an image is present but that no description is provided. Some screen readers in that case convey the “image file path” (e.g. “file11022545685245.jpeg”) or simply the word “graphic” with no extra information. So the user is simply denied access to information that could be important which usually leads to the feeling of frustration.

Most of the older screen readers did not recognise the *longdesc* attribute, which is also poorly supported by browsers. Jaws (FREEDOMSCIENTIFIC., 2009) for Windows was the first product (from version 4.01) to support the *longdesc* attribute. Although now available in latest screen readers, because it is used so infrequently, many users are unfamiliar with the *longdesc* attribute. A link to access more information about the image is recommended for that situation.

Even if an alternative description of a vector image is read out by the screen reader it does not mean that the screen reader can convey the meaning of all images.

This is for reasons explained previously (i.e. adequacy and accuracy of the alternative description or very long descriptions which are tedious to listen to and follow due to working memory limitations) and which are independent of the screen reader's capabilities.

B. Diagrams as vector graphics on the web

Despite its many accessibility features (MCCATHIENEVILE and KOIVUNEN, 2000) (see Appendix A section A.2.2), SVG is currently not yet well handled by screen readers (SWAN, 2009). The reasons why are unclear and various (browser support and compatibility, not easy keeping up with fast emergence of new technologies, etc). Although a load of work is needed to get screen readers to support SVG and its accessibility features, the problem is acknowledged and progress is under way.

The Accessible Rich Internet Applications documents (WAI-ARIA) (W3C, 2010) could become a future solution. It defines a way to make Web content and Web applications more accessible to people with disabilities by defining new ways for adding information to be provided to assistive technology. WAI-ARIA is only a step before becoming a recommendation and several browsers and assistive technologies already support it. ARIA 1.0 does not focus much on SVG but will apparently do in ARIA 2.0.

For Vector graphics inserted into an HTML page using the <object> element, to be accessible to screen readers, descriptive text or alternative representation should be enclosed within the object content. But the fall back mechanism of the object element is not well supported by screen readers (PILGRIM, 2003).

Appendix D

Accessibility of Diagrams and the WCAG 2.0

The four main principles of WCAG 2.0 (W3C, 2008c) have been used in this thesis for specific visual web content: “diagrams”. The focus is particularly on perceivability, operability and understandability of diagrams represented as “vector graphics”.

In the context of this research project, the most important principle considered is “Perceivable” as it is the principle for which access to diagrams differ the most.

D.1 The POUR Principles of the WCAG 2.0

D.1.1 Perceivable

Users must be able to perceive the information presented using at least one of their senses, if the content cannot be perceived then it cannot be accessed “...*the information must be perceived...that is the first step to accessibility upon which all others are based, and without which accessibility cannot happen*” (WebAIM, 2009b).

Diagrammatical communication is only possible if people can perceive the diagram content. They need to have access to the content presented to be able to process its information. For example, blind people are unable to use their sense of vision. They mainly rely on their sense of touch and hearing to gain access to information.

A. Availability of the information behind the diagram

The user needs to have access to the information behind the diagram in order to understand, think, reason, judge, analyse, navigate and memorize the information presented in the diagram.

The availability of the information relies on two important considerations: the author’s willingness to make it accessible and the format used to store it.

1. Origin of the information

❖ **Author’s willingness**

If the author is amenable, it would be useful if the original information from which their diagram was constructed was made available to the user.

At this stage what the author wants should be taken into account. It is believed that for various reasons most authors would not agree for that to happen. One possible reason could be the unwillingness to invest the effort. But perhaps the author might not want the real data to be showed as the diagram might have been deliberately simplified or distorted to convey a misleading message. It is his right as accessibility guidelines only expect “equivalent” content of visual representation of diagrams rather than deep description of the original underlying data which might expose potential misleading messages enabling the intent of the author.

If the information presented diagrammatically was linked to the presentation of the diagram then this information could be adapted and accessed according to the user’s needs.

It is believed that it would be more useful if the information represented diagrammatically is captured at the creation stage. This would allow a “direct access” as opposed as an “indirect access” to the information represented in the diagram. In a “direct access” a user obtains the information directly from the graphic and with an “indirect access” the information is first inferred and interpreted by a third party (human or computer) and then conveyed to the user using an alternative format.

The exploration of that information should then allow the user to understand and process the diagram.

❖ **Format used to store the information behind the diagram**

The format to be used to make it happen is very important.

Given the vector graphic format, it will be good if the information behind the diagram could be preserved to be made accessible.

For vector graphics to be made perceivable they would need to have all the necessary information of the diagram encoded within its format:

- the information it represents should be well structured (use of groups and ids) and appropriately coded (e.g. use text for true text rather than paths)
- facilities offered by SVG should be used properly (title, desc, comments)
- appropriate metadata should be added when needed
- separation between presentation and content: the content should not depend on the way it is presented (use of class attributes for styling).

2. **Graphic format used to store the diagram:** What format is used to store the information of the diagram

B. Type of information stored: structure and semantics

It would be useful to have access to the structure and the semantics of the diagram in a machine understandable way.

SVG allows both to be achieved. SVG stores structural information about the diagram as an integral part but the author has to provide a well constructed structure using the appropriate facilities (e.g. grouping). Storage of the well-defined data-model behind the diagram and the knowledge represented within the diagram represented as ontologies would be of benefit in making the semantic of diagrams accessible.

This information should be “programmatically determined”. This means that the information is authored in such a way that it is machine-understandable. User agents, including assistive technologies, could then have access to the information.

As previously explained the benefit of presenting co-ordinate positions of the objects composing the diagram has been investigated by David Bennett (BENNETT, 2002) and has proved to provide no benefit in the non-visual presentation of diagrams in audio for example. The information concerning the layout of the diagram is not important information whereas the concepts and relationships between these concepts represented in the diagram are important information.

But requirements concerning the information needed to be accessed vary depending on intended tasks, user’s capabilities, preferences and objectives.

For example the individual requirements of blind users are different. Congenital blind and later in life blind have different ways of treating information (KENNEL, 1996) as they have different personal knowledge. Blind people who were not born blind recognise and interpret graphical information much more easily. Depending on the information researched, the blind user might be interested in either the structure or the semantics of the diagram.

3. *Type of information available*: What kind of information is available to the blind user? The structure or the semantic of the diagram? Is it the implicit semantic or the explicit semantic?

C. Multimodal presentation of the information behind the diagram

When the visual channel cannot be used to perceive the diagrammatical information, this information has to be converted into alternative presentation modalities that can be perceived such as audio or tactile. This information needs to be presented in a way which can be easily perceived by the blind person using their

available senses “*Since not everyone has the same abilities or equal use of the same senses, one of the main keys to accessibility is ensuring that information is transformable from one form into another, so that it can be perceived in multiple ways.*”... “*Overall, text is the most easily and most universally transformable format*” (WebAIM, 2009b).

4. **Output modalities:** what are the output modalities supported and/or possible? Textual, graphical, Tactile, Audio, Tactile/Audio, Haptic, Haptic/Audio?

The alternative information presented should not restrict the blind person. The blind person should be allowed to gain access and explore the information or a subset of this information according to its need.

D. Access and exploration of the information

❖ Provide ways to access and navigate the information of the diagram:

- Being able to adapt the level of information provided: general overview, detailed, etc.
- Being able to access and navigate only relevant part of information hiding unnecessary details (“viewing options”)
- Possibility to navigate between different presentation modalities (e.g. from textual presentation to visual presentation)
- History mechanism, annotation mechanism “bookmarks”: would be useful if possible in order to come back to past exploration or as memory cues

❖ Support active interactive exploration

Query and search the information of the diagram and present the result in an accessible adaptable way. It should be possible to query the structure and the semantics of the diagram.

The query system gives some power to the user, who decides what information he wants from the diagram, what information is relevant for him to be communicated? Giving this power to the user can solve the problem of deciding what information is relevant, allowing the user to decide for him-self depending on the task he wants to achieve.

5. **View of information provided:** Power given to the user on the view of information provided (overview, detailed).

6. **Type of exploration:** Active or passive exploration? In a “passive” exploration the user cannot interact with the information provided whereas in an “active” exploration the user interacts with the information by querying it, annotating it, adapting it to his needs while exploring it (pause, hide, and stop). If “active” exploration then specify facilities offered to interact with and navigate the information (ability to annotate, to query, details-on-demand enables getting detailed information from specific elements of interest, ability to hide, stop, repeat, pause, travel forward and backwards when needed).

E. Adaptability

This concerns the facility provided to adapt the information to user and /or device requirements. Create content that can adapt to the specific needs and preferences of the user without losing information and in a location convenient to them (preserve meaning, relationships, reading order). The aim is to obtain diagrams which adapts to individual needs and preferences.

Adaptation facilitates access to information by adapting/ reconfiguring this information to specific input/output devices, modifying, filtering, transforming or removing/adding aspect of the current presentation into a different modality depending on user requirements.

7. **Adaptability:** Is adaptability considered? Does the output take into account device and/or user requirements?

D.1.2 Operable

Users must be able to operate the access to the information of the diagram. For example operability for both sighted and blind users to permit possible collaborations. Guidance should be provided to explore the diagram: instructions, error alert, warnings, etc.

Multimodal access should be considered. Keyboard accessibility allows many technologies to be operable providing alternative mode in allowing search, exploration, navigation and interaction with the information, allowing access to different disability types. Blind users use their keyboard as their primary means for navigating the web.

So keyboard accessible or other accessible input modality adapted to blind users (Braille, voice) should be provided to access the information behind the diagram.

D.1.3 Understandable

The usability of the information accessed relies on its Understandability “...providing alternative or supplemental representations of information can often increase understandability” (WebAIM, 2009b).

Users must be able to understand the information presented as well as the operations the system offers. Functionalities offered to explore the information in its alternative forms should be understandable (e.g. navigation and interaction functionalities).

The language employed should take into account the intended audience taking into consideration educational background, knowledge familiarity with the domain content, etc. The knowledge such as the type of diagram would allow a better understanding of the information provided.

D.1.4 Robust

Careful thought needs to be given into the choice of technologies used with regards to compatibility with existing technologies such as web browsers, assistive technologies (e.g. screen readers), etc. The diagram should remain accessible, no matter how the technologies evolve.

Extensibility, interoperability and deployment should be taken into account.

SVG support in all browsers (native and/or using SVG Web) and continuous improvement makes it a good format to use for interoperability.

A standard should also be provided for capturing graphic information at a higher level.

D.2 Existing approaches in the context of the defined requirements

The following tables (Table 9 and Table 10) provide an overview of all the existing approaches reviewed. The following information is presented.

Aim result of the project: Describe the aim result of the project in terms of general outcome (methodology, system, guidelines, etc.).

Advantages: Present the advantages of the approach

Limitations: Present the limitations of the approach

1. Origin of the information

2. **Graphic format used to store the diagram:** What format is used to store the information of the diagram
3. **Type of information available:** What kind of information is available to the blind user? The structure or the semantic of the diagram? Is it the implicit semantic or the explicit semantic?
4. **Output modalities:** what are the output modalities supported and/or possible? Textual, graphical, Tactile, Audio, Tactile/Audio, Haptic, Haptic/Audio?
5. **View of information provided:** Power given to the user on the view of information provided (overview, detailed).
6. **Type of exploration:** Active or passive exploration? In a “passive” exploration the user cannot interact with the information provided whereas in an “active” exploration the user interacts with the information by querying it, annotating it, adapting it to his needs while exploring it (pause, hide, and stop). If “active” exploration then specify facilities offered to interact with and navigate the information (ability to annotate, to query, details-on-demand enables getting detailed information from specific elements of interest, ability to hide, stop, repeat, pause, travel forward and backwards when needed).
7. **Adaptability:** Is adaptability considered? Does the output take into account device and/or user requirements?

REVIEW OF BOTTOM-UP APPROACHES FOR VECTOR IMAGES and Vector Graphics ACCESSIBILITY

	Text Alternatives (W3C Guideline, WebInSight, Altifier, Ault project, etc.)	AudioGRAF	T3	BIS	GUIB
Aim result of the project	Guidelines, Corrective tools	System	System	System	Methodology
Advantages	Might make some simple vector images accessible	if comprehensible auditory display then allow user to build a mental map of the information	provide effective access for simple diagrams multiple levels of details which organize and allow a large quantity of information Interactive experience	Allows the exploration of the structure and implicit semantic of a vector image.	for frequently used vector images allows the work of one sighted person to be used by many blind people
Limitations	Quality depends on willingness and expertise of its author Might be incorrect, unusable, incomplete or inappropriate automatic generation not 100% efficient not trivial and not enough or inappropriate for description of complex diagrams can be costly in terms of time and effort for complex diagram	preparation work to be done by sighted person special equipment a bad auditory system could deep the user into confusion and so frustration and giving up limited by the limitations of tactile graphics involves special equipment Tablet plus tactile graphic	preparation work to be done by sighted person, which can be time consuming and require a certain level of expertise which can be difficult to carry around diagram need preparation limited by the limitations of tactile graphics involves special equipment Tablet plus tactile graphic	preparation work to be done by sighted person the description obtained depends on the person creating it, who will probably not be the author of the diagram and might omit important information from the description	preparation work to be done by sighted person who has an important responsibility, who decides what information to convey and imposes his view involves special equipment blind user depends on sighted for rarely used diagrams involves special equipment Audio accessories

	Text Alternatives (W3C Guideline, WebInSight, Altifier, Ault project, etc.)	AudioGRAF	T3	BIS	GUIB
1. Origin of the Information	Inferred by Author or third party	Inferred by Author or third party	Inferred by Author or third party	Inferred by Author or third party	Inferred by Author or third party
2. Graphic format used to store the diagram (Raster or vector)	vector image	vector image	vector image	vector image	vector image
3.Type of information presented (structure or semantic)	Information that the graphics intend to convey visually (implicit semantic) and/or structure of the graphic (depends on the author of the description)	Structure, implicit semantic	Structure, implicit semantic	Structure and implicit semantic	Structure and or semantic depending on the sighted person describing it
4. Output modalities	Textual description accessed using screen reader	Tactile/Audio	Tactile/Audio	Textual, audio using screen reader	Audio through text-to-speech device
5.View of information provided (overview, detailed)	Depend on how well the guidelines are followed	Multiple levels of details, the choice is given to the user	Multiple levels of details, the choice is given to the user	Filtering information to filter out unnecessary information	Depends on the description provided by the person describing it, , but user has no choice
6. Type of exploration	Passive Possible Navigation depending on the encoding of the textual information and the screen reader used (accessible XHTML)	Active Interaction using the touch panel	Active Interaction using the touch panel	Active Exploration of the semantics or the structural using the browser to navigate around the tree of objects	Passive
7. Adaptability	Not addressed	Not addressed	Not addressed	Not addressed	Not addressed

Table 9: Review of Bottom-up approaches (part 1)

	SIGHT	TeDUB	iGraph Lite	ViewPlus	SVG Linearization
Aim result of the project	System, methodology	System	System	System composed of set of tools	Methodology
Advantages	sometimes provides intended message successfully does not require intervention of author does not involve the use of specialized hardware	allows the navigation and annotation of diagrams through a number of input and output devices Presents overview of the information by providing access to hierarchical data structure for the information of the diagram	Makes some information carried by the vector image accessible attempt to make some implicit features explicit (e.g. max, min) Provides facilities to access and explore information (e.g. sequential query system) Open source, free	provide effective access for simple vector image and various vector graphics multiple levels of details which organize and allow a large quantity of information Interactive experience	Generate accessible description of the diagram if the SVG is appropriate will well organised information.
Limitations	Not 100% effective: message inferred not always the one intended Based on assumptions Some technical issues concerning the amount and quality of information presented	Rely on a semi-automatic process to recover the information from the vector image which might generate some errors due to some noises in the vector image Overcoming this issue might require human intervention and time involves special equipment computer game joystick	Depends on plug-in which might not be supported by graphical applications Some information are not inferred and so not conveyed the present query system is not flexible and depends on the sequential exploration of the textual description	preparation work to be done by sighted person, which can be time consuming and require a certain level of expertise information in SVG might need to be reorganised or enriched with extra information if missing limited by the limitations of tactile graphics involves special equipment Tablet plus tactile graphic and embosser, which can be difficult to carry around	Rely on author's patience and willingness adding RDF is an onerous task for the author might involves tedious efforts

	SIGHT	TeDUB	iGraph Lite	ViewPlus	SVG Linearization
1. Origin of the Information	Inferred by automatic analysis	Inferred by Semi-automatic analysis	Inferred by automatic analysis	Inferred by Author or third party	Inferred by Author or third party
2. Graphic format used to store the diagram (Raster or vector)	vector image	vector image	vector image	vector image and vector graphics	Vector graphics
3.Type of information presented (structure or semantic)	Hypothesized Intended message of the diagram (implicit semantic)	Structure and semantic (implicit)	Implicit semantic	Structure and implicit semantic	Structure and implicit semantic
4. Output modalities	Textual description accessed using screen reader	Textual, Haptic/audio	Textual, Audio using screen reader	Audio and tactile	Textual, audio using screen reader
5.View of information provided (overview, detailed)	General and detailed, but user has no choice	Presents overview of the information using hierarchical data structure Detailed as possible to navigate diagrams	Extract, explore and navigate information using Flexible and scalable NL dialogue, Sequential query system iGraph-Lite navigation tool	Multiple levels of details, the choice is given to the user	No
6. Type of exploration	Passive but plan to provide facilities for interaction	Active Diagram exploration and information searching, annotations facilities	Active and/or passive Textual description using screen reader or query system	Active Interaction using the touch panel	Passive
7. Adaptability	Not addressed	Not addressed	Not addressed	Not addressed	Not addressed

Table 10: Review of Bottom-up approaches (part 2)

