# Visual Language and Environment for Statistical Surveys

Chul Hwee Kim

This thesis is submitted in fulfilment of the requirements for the degree of Master of Science in Computer Science, completed at The University of Auckland.

February 2006

## Abstract

Statistical surveys are a major analytic tool in inferring information about a set of objects comprising a target population. The process of designing and managing statistical surveys is a complex process with multi-faceted operational and technical issues. The development of supporting tools in the area of statistical surveys is very topical in its scope and nature. Existing tools exhibit the lack of conceptual modelling capabilities. This means that they cannot provide a high-level survey specification to overcome the problems of process heterogeneity.

Our research is concerned with the practical issues in supporting the survey process, particularly from a visual approach. We offer conceptual modelling functionality for statistical surveys using a suite of visual languages (Survey Design Language - SDL) based on our previous work in this area. The development and wide adoption of UML by the software engineering community shows how the types of the problems, which are paralleled in the survey process, can be resolved by providing a single place to form a unified view on the particulars of the system using a set of independent views. Multiple diagrammatic representations of the SDL are influenced by UML in this aspect. The approach is supported further by the necessity of multiple modelling spaces and the benefits of heterogeneous reasoning.

We also present a proof-of-concept visual environment that enables the development of executable models in a practical setting. Our prototype implementation allowed us to visually model various aspects of surveys, build executable analytical exploration models and generate web services from diagrammatic specifications.

We tested the use of the prototype tool in the context of a real-life survey by building a walkthrough designed around the broad theme of the National New Zealand Survey of Crime Victims 2001.

Our prototype tool is also presented together with a user evaluation of the visual language and the tool in the form of a cognitive walkthrough. Usability issues were analyzed from the perspective of the cognitive framework.

SDL diagrams were useful both as a conceptual modelling tool and an executable model development environment. The multiple usage of SDL diagrams are made possible through the use of three layers: visual, metamodel, and semantic. The layered approach maintains a high-level of visual awareness, while allowing concrete executable specifications and diagram semantics to coexist. This enabled the back-end integration into heterogeneous external services such as a computation engine and inter-diagram integration.

We put forward the future directions for research to refine the layered diagram structure on a more solid ontological foundation and improve integration with the survey process while creating better user experiences.

## Acknowledgements

My heartfelt thanks to John Hosking and John Grundy. It would not be an overstatement to say that they laid the foundation for what I have here.

Thanks also to James Reilly who contributed his valuable time freely to guide me in this journey.

I would like to express my gratitude to all those who participated without payment in my evaluation study.

And finally special thanks to my family for their unceasing support and patience.

# Table of Contents

# *List of Figures and Tables*

# Chapter 1 Introduction

Surveys have extensive roots that run back to ancient times (Mandansky 1986) and the subsequent development of probability theory and mathematical statistics provide a scientific foundation for statistical surveys (Biemer and Lyberg 2003). Statistical surveys have become a valuable ubiquitous tool for obtaining trustworthy information about a target population in many areas.

Our research is concerned with practical issues in supporting the survey process, particularly from a visual approach. From a survey practitioner's point of view, statistical computing is well supported by high quality software package like R (Narasimhan 2005). That is, the low-level implementation of a statistical technique is not a significant operational concern in the survey process. However there are other important aspects such as statistical metadata (MetaNet 2003), Triple-S (Jenkins 1996) and heterogeneity in statistical data semantics (Olenski 2003) that are not well addressed. Non-mathematical activities are as much a part of the survey process as mathematical ones. To date, there has been no explicit tool and modelling support for the survey process which can cover all the multiple facets of statistical surveys. This research investigates a visual development environment for statistical surveys and presents our strategies in developing solutions for the eclectic survey issues.

## *1.1 Our Motivation*

Our research finds its motivation in several areas. The success and wide adoption of UML and model-based approach in software engineering inspired us to design a set of notations for statistical survey specifications. Our research hypothesis is that when the semantics of survey designs are visually specified and modelled in a visual language then the following benefits are expected:

- Mitigation of communication overheads for both non-experts and experts. A difficult survey concept can be represented with a graphical metaphor, the graphical representation provides an abstraction layer which isolates low-level details from high-level concepts.
- Harmonisation of disparate operational semantics present at the tool level.
- Model-driven management and execution of the survey process.

Our foundational research work was presented at ***IEEE VL/HCC 2005*** (Kim, Hosking, and Grundy 2005) and this research updates the notational elements of the visual language and incorporate the development of a proof-of-concept tool to demonstrate the benefits in practice.

## 1.2 Our Goal

We attempt to answer a question, "How well can we support the survey process from the perspective of a visual language?" The answers to this question can be from several angles. Firstly, we aim to present a case for how the survey process should be modelled using a suite of visual languages and this is a central topic of Chapter 2. Secondly, the development of a software environment for statistical surveys will be based on visual language to give us practical perspectives on tool support for visual language in the context of the survey process. Finally we aim to evaluate whether the two research outcomes, which are diagrammatic notations and the proof-of-concept tool, are congruent with the concerns and needs of survey designers.

## 1.3 Our Approach and Thesis Overview

Our approach to the research problems follows the structural progression of this thesis. Thus we will present each thesis chapter along with relevant contextual information to our approach.

In ***Chapter 2***, we begin with a broad review in the areas of statistical surveys, UML, statistical packages and a semantic layer support provided by TopicMaps to provide sufficient background information on our research. The concepts and technologies presented in the review are building blocks in the development of both the visual language suite and tool support. It must be noted that related research topics presented in the next chapter might at first seem to be disparate. The domain specificity of our research and the lack of strongly related research in our subject meant that we had to look for research works that only cover the partial facets of our work.

***Chapter 3*** begins the exploration of visual language development and introduces the concepts underlying the SDL design. We support multiple diagrammatic representations for statistical surveys from two orthogonal angles: modelling spaces (Unhelkar and Henderson-Sellers 2004) and heterogeneous reasoning (Barwise and tchemendy1995).

We then lay down theoretical concepts and the foundation for our visual language suite ***(Chapter 4)*** and tool so that we form a solid basis for the evaluation and extension of our current work. Once the basis for the theoretical foundation is established, the requirements (***Chapter 5***) and implementation of the proof-of-concept tool is discussed (***Chapter 6***) and some of significant design decisions are elaborated in ***Chapter*** *7*.

The remaining chapters (***Chapter 8 and 9***) focus on presenting research outcomes in the context of the survey process. ***Chapter 8*** covers the usage of a sub-set of the tool features and conceptual modelling that are relevant to a survey scenario which is derived from the New Zealand National Survey of Crime Victims in 2001 (NZCS 2003). ***Chapter 9*** discusses formal user testing of our visual language suite and the proof-of-concept tool to view out research work from the perspective of real-life users.

We conclude this thesis with research outcomes, our contributions to the field of statistical surveys and a broad overview of paths for future research.

## 1.4 Our Contributions

In this theis, we develop and evaluate a novel approach to facilitate the survey process in the areas that have been neglected by existing survey software tools. We implement a suite of visual languages for statistical surveys and a visual environment to develop and execute them.

We envisage that both conceptual and executable models are necessary for improved survey support. We put forward a multi-layered approach that enables purely visual representations to coexist with executable models to minimise the division between conceptual modelling and low-level implementation stages.

In addition, we explore the survey infrastructure and collaborations models to integrate heterogeneous components within the survey process.

The scope of this theis is very eclectic, which ranges from statistical metadata to visual language. We feel that the depth of our research has been limited by imposed time limits but hope to provide a small piece of insight into future development in the area of visual language and tool support for statistical surveys.

## 1.5 Summary

In this chapter we have introduced the motivating forces, key research areas and our approach to this research. The next chapter presents some of the building blocks for our research and critiques related research.

# Chapter 2 Background and Related Research

## 2.1 Introduction

In this chapter, we will introduce concepts and background information that will be helpful in understanding latter chapters. We will also present related research in the area of statistical computing to acquaint ourselves with the relevant research activities from both academic and commercial sectors. The research scope of this thesis is eclectic: the academic subjects ranging from visual language to the statistical survey process. Therefore the decision to opt for prerequisite requirements which are biased toward software engineering practitioners arises to limit the scope and length of this thesis otherwise the bulk of thesis would be assumed be primers for disparate disciplines. The prerequisites assumed of the reader are:

- Object oriented programming and design techniques
- Basic knowledge of metadata solutions
- Basic knowledge of pervasive and distributed computing
- Elementary exposure to statistical concepts and techniques

Some of the tool specific background information is discussed in Chapter 5 in more detail.

## 2.2 Background

### 2.2.1 Statistical Survey

In this section the subject of statistical survey is presented. Statistical survey is a vast discipline with a variety of topics. Thus this section focuses only on the fundamental ideas of survey and the survey process that are most relevant to the development of SDL. The first part of this section is largely adapted from Moore and McCabe (1993) , and Biemer and Lyberg (2003) and (Kim, Hosking, and Grundy 2005) and the second part summarises the outcomes from the discussions with our statistics consultant James Reilly.

The definitions of a survey comprise a number of prerequisites as follows:

1. A survey concerns a set of objects comprising a population.
2. The goal of a survey is to describe the population by one or more parameters defined in terms of measurable properties.
3. To obtain observational access to the target population, a frame is needed. An example of a commonly used frame is a phone book or electoral roll.
4. A sample of objects is selected from the frame in accordance with a sampling design.

***Survey Process***

Having reviewed the definition of a survey, we now overview of the process for planning and executing surveys. Figure 2.1 shows the survey process, which consists of multiple stages.

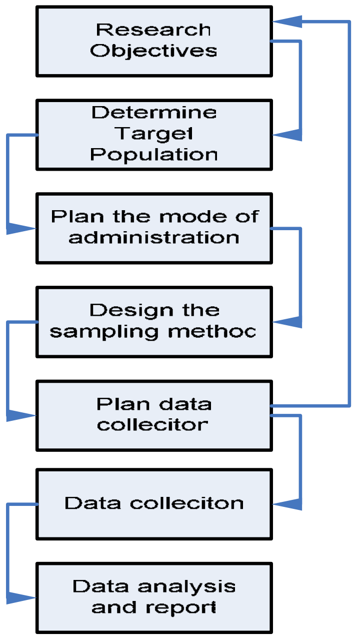

Research Objectives
Determine Target Population
Plan the mode of administration
Design the sampling methoc
Plan data collecitor
Data colleciton
Data analysis and report

**Figure 2. 1 Survey process (Biemer and Lyberg 2003)**

The survey process is composed of a number of steps that are usually executed sequentially, from determining the research objectives to analysing the data. The first step in the survey process is to determine the research objectives. The next step is to define the population to be studied. For example the target population in a university student survey is students who are enrolled at the university. The next step is to determine the mode of administrations for the survey. Here we consider the mode of collecting data. E.g. telephone, face-to-face interviewing, or online. Once the target population and the research objectives and the mode of administration are defined then the next stage of the process, which specifies the sampling design, can begin. The sampling design sets out specifications to describe the sampling frame (the list of population members) and the methods used for randomly selecting the sample. The next step of involves implementing data collection plans developed in the previous steps, e.g. sending mail questionnaires or conducting phone interviews. Once the data are collected they can be cleaned and finally prepared for data analysis. Data analysis draws inferences about a population. Data analysis employs a variety of statistical techniques tools for

***Statistical Computing and Sources of Complexity***

The inception of the large-scale surveys such as the decennial census (The U.S. Bureau of the Census) brought enormous managerial complexity that prompted the development of statistical computing long before the birth of personal computers (Chambers and Ryan 1990). Various software packages have been developed to assist in studying data from complex surveys and the availability of statistical software tools are not the part of the problem that we attempt to address as mentioned in Chapter 1. But the lack of integration and cross tool support is often the sources of in the survey process. Whenever a popular statistical methodology emerges, software vendors are keen to come up with another tool or upgrade. The existence of a myriad of tools has had an impact on organizational efficiency since the distinctiveness of each tool, in the absence of mature and widely accepted inter-operable standards, requires an associated set of distinct formats to describe data and communicate application events. The following example shows the problem in practice. Let's say two particular organisations A and B use three software tools for surveys.

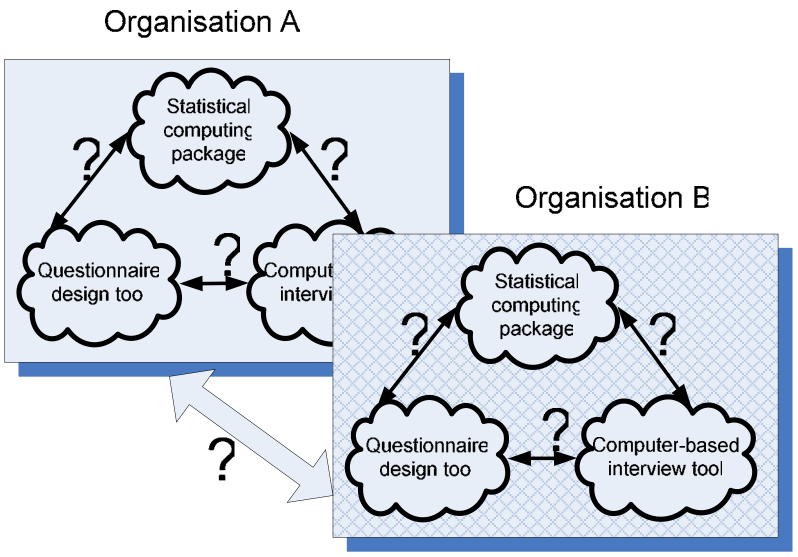


**Figure 2. 2 Lost semantics between different tools and organisations**

Many popular statistical software tools are good for the most part. However they all tend to address very narrow goals and specific types of tasks. Thus the tools may provide an immediate solution in the target problem domains, yet they add significant secondary costs by creating problems such as data incompatibility and loss of semantics. All three tools do not share inter-operable data formats that can annotate the data with semantics (e.g. how data is created and what can be done with data). Therefore the only way to pass one form of data to another would be exporting the native data forms to generic forms such as comma separated values then the lost semantics are added either by manually or programmatically. Suppose we would like to make the three applications to be used in sequence. Without equal cross-tool communication standards support by all three tools, orchestration would be a labour intensive process. In addition to the inter-organisation communication difficulties.

### 2.2.2 The UML

Growing out of the Object Management Group's (*OMG*) call for a common approach in object oriented modelling, no visual language has achieved wide acceptance across so many areas raging from software development to business process modelling in the last decade like UML. Whether the UML notation is used as a mean to capture business rules, build conceptual models, or design system architectures, one of strengths of UML has been its ability to describe a complex system from various perspectives in a unified fashion. The impact of UML's approach in representing complex systems can be seen in our diagrams (SDL) in Chapter 3 though it is not the only factor contributing to SDL's design principles; a survey is modelled in a set of views to represent multiple perspectives of a survey. Likewise SDL offers five views, which are expressed diagrammatically in survey diagram, survey data diagram, survey analysis diagram, survey task diagram and survey process diagram, to depict the multiple views. In the rest of this section, we will present the essential nature of the UML diagrams that parallels the development of SDL rather than rehashing the syntax, usage and semantics of UML diagrams.

UML 2.0 consists of 13 diagrams and each diagram represents a different aspect of software projects, addresses the needs of various stakeholders and plays a specialised role in the software modelling.

**Nature of UML Diagrams**

The nature of UML diagrams can be categorised into two groups: structural and behavioural (OMG 2005) as shown in Figure 2.3.

A structural diagram's main concern is to model how things exist in relationships and to provide the abstraction of how objects of interest are composed. A behavioural diagram describes what they should do and how they function.

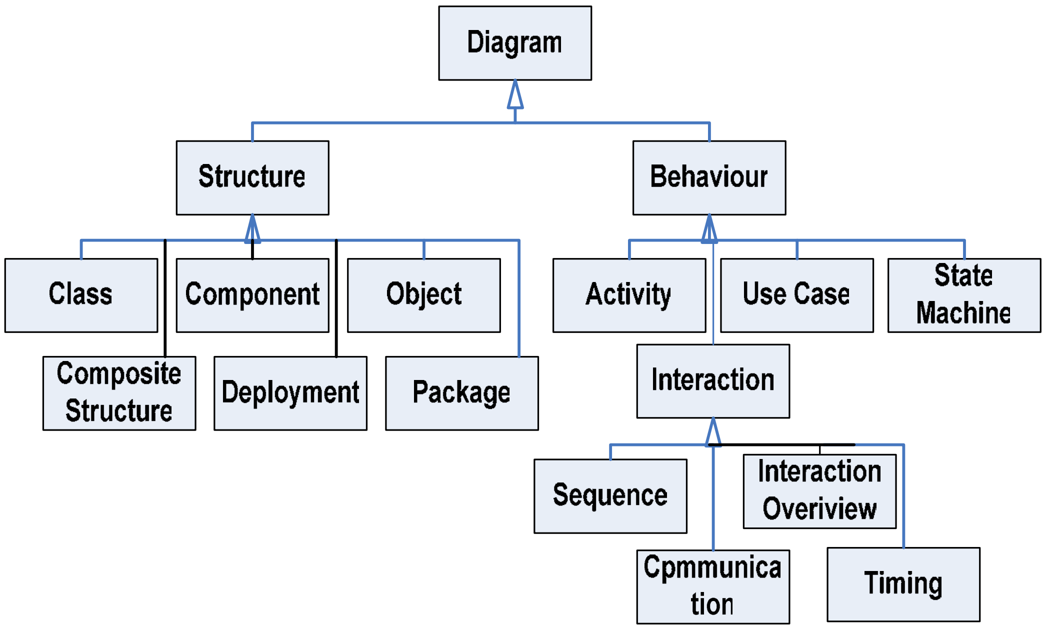


**Figure 2. 3 Hierarchy of UML diagrams by nature (Keller 2004)**

| ***Diagrams*** | ***Modelling Goal*** |
|---|---|
| Object | Model objects and the relationships between them |
| Class | Model classes, attributes, and interrelationships in the system |
| Composite structure | Run-time behaviour of component or object |
| Component | Model software systems at architecture-level |
| Deployment | Model a static view of the run-time configuration of processing nodes and the components |
| Package | Model how subsystems are organised |
| Use case | High-level functional user requirement |
| Activity | Model the logic of a business rule or the logic expressed in a use case |
| State machine | Model object states and the transitions between those states |
| Interaction overview | Variant of activity diagram, models control flow |
| Sequence | Model the flow of logic in the system |
| Communication | Model message flow between objects |
| Timing | Model behaviours of objects throughout a given period of time |

**Table 2. 1 UML diagrams and their modelling goals**

When a problem is viewed from various perspectives, the multi-faceted analysis can increase the richness and depth of communication. However UML's elasticity (Unhelkar 1999 p28) is not only derived from having multiple types of diagrams that can support the entire software lifecycle process but is firmly rooted in OMG's metamodel which provides a liberal extension of the UML language. UML not only offers visual modelling toolsets but has built-in extensions schemes so that users may expand or reduce UML considering their unique needs. The discussion of the UML metamodel alone is a vast subject thus we limit our discussion around core metamodel attributes that are most relevant to our research.

The UML metamodel is based on the four-layer architecture:
(UML Semantics, OMG 1998 Ch2)

- User objects: An instance of a model. Defines a specific information domain.
- Model: An instance of a metamodel. Defines a language to describe an information domain.
- Metamodel: An instance of a meta-metamodel. Defines the language for specifying a model.
- Meta-metamodel: The infrastructure for metamodeling architecture. Defines the language for specifying metamodels.

The layered architecture is illustrated in the following Figure 2.4 which depicts the process to abstract, generalize and classify on the problem domain of the modelling language (Mellor et al 2004 Ch4 MDA Distilled: Principles of Model-Driven Architecture):

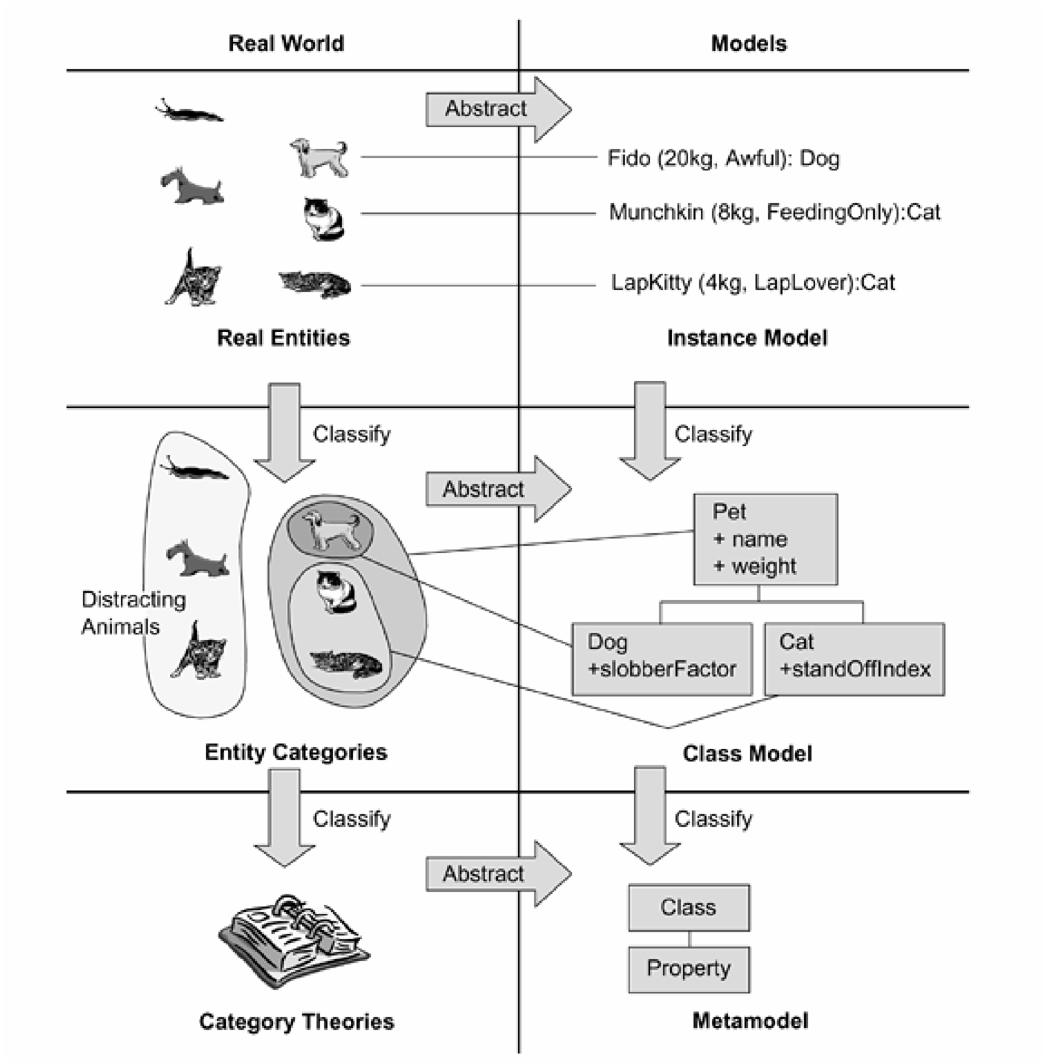


**Figure 2. 4 Metamodel in relation to developer models (Meller et al. 2004 Ch4)**

The layered approach facilitates precise communication of what a model means without exhaustive annotations. By allowing regression to more abstract layers, we can delegate the tasks of providing more precise semantics out of the UML model (what users can see) thus simplify the communication process. A parallel analogy can be made in the area of object-oriented analysis (OOA). For example, an 'Account' object may have attributes and operations and is persisted by serialisation. When communicating the Account object we can describe the object exhaustively or simply define as an instance of an Account class type and hand over the task of elaborating its precise meanings to the appropriate metamodel of the instance.

## Application of UML Approach: Relevance to Our Research

The categorisation of the diagrams suggests that there are multiple conceptual modelling spaces that should be addressed by visual diagrams to build successful models for a complex problem. The application of the layered architecture to the design of visual language could provide flexibility in extending and enriching SDL as it does for the UML. This discussion of the elasticity of visual language will be expanded in Chapter 4.

In practice although UML is a widely regarded as a de facto standard in designing and documenting information systems in various domains, it is criticized for imprecise semantics leading to subjective interpretation and difficulties in tool support and machine processing. This is especially true when the diagrams are not constrained by the use of OCL and lack secondary diagram notes. This type of shortcoming has been remedied in the SDL language design by creating a layer that sets out relationships between the model layer and the modelled ontology for SDL using a topic map. Our development direction differs significantly from the constraints-based approach. One principle reason behind our decision was due to the highly technical nature of the OCL that alienates much of our target audiences. In the next section, the main ingredient for the modelling ontology the topic map is presented and its relevance to the layered architecture is one of major topics of Chapter 4.

### 2.2.3 Topic Maps

***Purpose***
Topic Maps were originally created to handle the construction of lists, glossaries, thesauri, and tables of contents, or more simply automatic indexing systems. (Park and Hunting 2002) Topic Maps grew out of the original domain of application and found their uses in diverse areas, ranging from library and information science (LIS) to astrophysics (Mahabal et al. 2002). The maturity of Topic Maps can be seen in the standardisation efforts that resulted in the international standards for Topic Maps (ISO/IEC 13250) and on-going research into myriads of related issues such as their complementary roles in the Semantic Web (Garshol 2003, Pepper 2002) and ontology-driven topic maps (Vatant 2004).

***What is a Topic Map***
The advent of XML facilitated the creation of interoperable standards. The interoperability capabilities do not automatically organise interoperable resources into useful information systems which show how the individual resources exist in relationship to other resources.

A high-level description of Topic Maps can be summarised as an abstraction of key concepts within the context of their relationships with each other. The resulting indexing structure may contains semantic links that span over multiple information hierarchies thus permitting higher degree of freedom in finding relevant information contrary to designer-driven rigid indexing schemes that often require users to follow a pre-defined knowledge navigation.

Elements of a Topic Map show how the above mentioned premises are supported in its implementation. A Topic Map has only a few key conceptual elements as follows:

*Topics*
Topics are the most central concept in topic maps. A topic can be anything that we want to represent in topic maps. For example, in a topic map about computer languages we may find topics representing C++, Java, and compiler.

*Types*
Each topic can have one or more types. Types are themselves also types. Topic map types can be seen as the indication of topic's membership to a set. E.g. the sun belongs to a star type.

*Associations*
The relationships between the topics are denoted by associations and likewise the topics they can also be typed. For example, the relationship between Auckland and New Zealand can be "is-city-of" association.

*Roles*
A topic map role is one of attributes of the associations and represents a role played by an association.
For example,

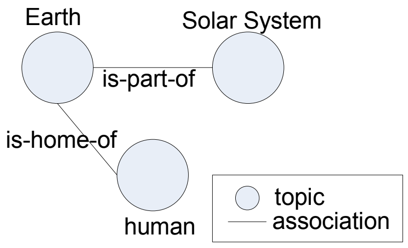


**Figure 2. 5 A simple topic map**

The topic 'Earth' plays distinctive roles in both of the associations. In the "is-part-of" association the Earth plays the role of 'member' while in the "is-home-of" associations plays the role of 'home'.

*Occurrences*
Occurrences are information resources that are relevant to a topic. An occurrence can point to external resources via various addressing mechanisms such as XRef (XRef 2001). For instance the topic 'Earth' may have occurrence in forms of web pages, media files, etc.

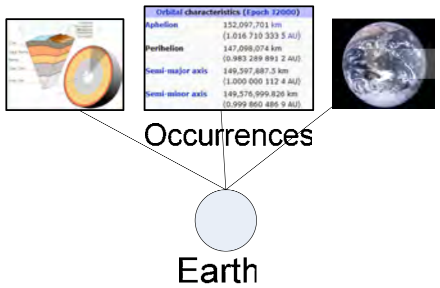
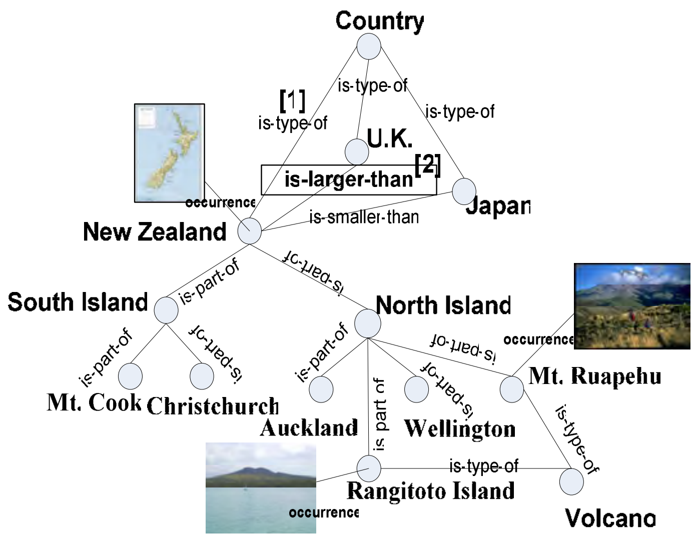


**Figure 2. 6 Multiple occurrences of the topic 'Earth'**

**Example**

To demonstrate how a topic map is created in practice, we will present a simple case involving the modelling of an information space for the geography of New Zealand. We will start by defining topic types first then list types for our topic map. Both visual and XML Topic Maps (XTM 2001) representations (partial) will be presented side-by-side.

- Topic Types: country and volcano.
- Topics: New Zealand, U.K., Japan, North Island, South Island, Mt. Ruapehu, Auckland, Christchurch, Wellington, Rangitoto Island, and Mt. Cook



[3]
```
<topic id="country">
        <baseName>
            <baseNameString>Country</
baseNameString>
        </baseName>
    </topic>
    <topic id="NZ">
        <instaceOf>
            <topicRef xlink href="#country"/>
        </instaceOf>
        <baseName>
            <baseNameString>New Zealand</
baseNameString>
        </baseName>
    </topic>
    <topic id="UK">
        <instaceOf>
            <topicRef xlink href="#country"/>
        </instaceOf>
        <baseName>
            <baseNameString>UK</baseNameString>
        </baseName>
</topic>
```

[4]
```
<topic id="#is-lager-than">
        <baseName>
            <baseNameString>
            is-larger- than
            </baseNameString>
        </baseName>
    </topic>

<topic id="#smaller">
        <baseName>
            <baseNameString>
            Smaller entity
            </baseNameString>
        </baseName>
    </topic>
    <topic id="#larger">
        <baseName>
            <baseNameString>
            Larger entity
            </baseNameString>
        </baseName>
    </topic>
```

[5]
```
<association>
        <instaceOf>
            <topicRef xlink href="#is-lager-than"/>
        </instaceOf>
        <member>
            <roleSpec>
                <topicRef xlink href="#smaller"/>
            </roleSpec>
            <topicRef xlink href="#UK"/>
        </member>
        <member>
            <roleSpec>
                <topicRef xlink href="#larger"/>
            </roleSpec>
            <topicRef xlink href="#NZ"/>
        </member>
    </association>
```

**Figure 2. 7 Modelling the geography of NZ using a topic map**

Topics are modelled with associations [1 and 2]. The ‘is-larger-than’ association [2] is represented by XTM documents [4 and 5]. The topic ‘UK’ is related to the topic ‘New Zealand’ by the association [2 and 4] which the two topics play the roles of the ‘smaller’ and the ‘larger’. This means that the association is not just a link between the two topics but express the underlying semantics of the association.

**Applications**
Topic maps have found their primary usages in the area of knowledge organisation and are used in practice by various organisations such as The U.S. Internal Revenue Service (Biezunski 2003) and the U.S. Social Security Administration (Degler and Battle 2003). Some of the benefits of the applications (Mahabal 2002 and Ahmed 2001) are:

- Enhanced searching for knowledge base
- Navigation of large data repositories
- Ontology for organisational information
- Knowledge organisation in federated datasets
- Expressive data structure

Topic maps are potentially useful in expressing the semantics of inter-related concepts or in Topic map’s terminology ‘topics’. Since anything that can be discoursed can be put into relationships with other concepts by means of association and we can integrate them in well-specified ways using a standardised format (Passin 2005). The nature of Topic Maps can have a direct impact on our approach in handling statistical metadata and the organisation of information as it shapes how all foundational concepts of our visual language are integrated together to create a unified view of the visual language out of diversities at the visual level.

### 2.2.4 Meta-Tools

Our prototype tool is built on Pounamu and the development of the visually rich tool would have been almost unfeasible given our time and resource constrains. Therefore, we will briefly discuss on Pounamu and a broad category of tools encompassing Pounamu call meta-tools.

***Meta-Tool Approach***
A metamodel is simply a model of a model and it defines the structure, semantics, and constraints for a family of models (Meller et al.2004). A meta-tool is able to work with metamodels to build modelling environments. This means that when a visual language’s metamodel is defined then a meta-tool can work with the metamodel to produce modelling environments. If the modelling environment is visual, then the end outcome is the creation of a visual modelling tool for the visual language. From an end-user’s perspective, this is almost effortless access to modelling tools. There have been a few meta-tool offerings such as MetaEdit+ (Kelly and Rossi 1996), IPSEN (Klein and Schürr 1997), GME (Ledeczi et al. 2001) and Pounamu (Zhu, Grundy, and Hosking 2004*)* from diverse backgrounds.

***Pounamu (Zhu, Grundy, and Hosking 2004)***
Pounamu is an indispensable part of our prototype tool and we will briefly review the development model using Pounamu against an alternative development option. With Pounamu virtually everything is visual except behavioural specifications that still need to be hand-coded. Diagram authoring support can be implemented in an instant. A model for a visual language is composed using a built-in visual model element editor and then the models can be mapped to corresponding visual shapes. The last step is creating a view in which the models reside. In three steps, we have a fully working visual diagram editor. Pounamu is particularly suitable in the areas of software development when clear model definitions are mandatory. In an alternative scenario using the MVC framework, it has three components that are analogous to Pounamu's three sub components. However, it would be a tremendous task to build an actual working tool in an instant from the conceptual meta-models since the programmer(s) must take care of the model to code mappings manually.

## 2.3 Related Research

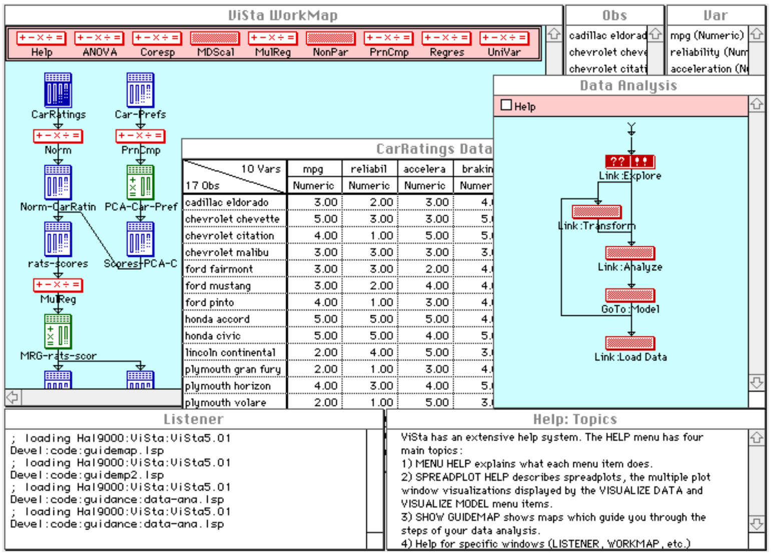
The Narrow domain specificity of our research means that there are only a few research activities that can be considered closely related to our research in their primary research directions. This can be one of the negative aspects of developing a novel solution in a highly specialised field such as statistical surveys. The following research or commercial projects do not match the scope of our research work in all areas however they all address the common goal of assisting a statistical process using software tools and statistical metadata management.

### 2.3.1 ViSta: A Visual Statistics System (Young and Bann 1997)

ViSta is an open and freely available visual statistics system which targets data analysts with varying abilities ranging from beginning statistics students to expert data analysts.

ViSta was developed originally by Forrest W. Young at the University of North Carolina at Chapel Hill in 1990 and has been under continuous development since then.

"The fundamental hypothesis underlying ViSta's design is that data analyses performed in an environment that visually guides and structures the analyses will be more productive, accurate, accessible and satisfying than data analyses performed in an environment without such visual aids." (Young and Bann 1997)



CarRatings Data

| 10 Vars / 17 Obs | mpg | reliabil | accelera | brakin |
|---|---|---|---|---|
| | Numeric | Numeric | Numeric | Numer |
| cadillac eldorado | 3.00 | 2.00 | 3.00 | 4. |
| chevrolet chevette | 5.00 | 3.00 | 3.00 | 5. |
| chevrolet citation | 4.00 | 1.00 | 5.00 | 5. |
| chevrolet malibu | 3.00 | 3.00 | 3.00 | 3. |
| ford fairmont | 3.00 | 3.00 | 2.00 | 4. |
| ford mustang | 3.00 | 2.00 | 4.00 | 4. |
| ford pinto | 4.00 | 1.00 | 3.00 | 4. |
| honda accord | 5.00 | 5.00 | 5.00 | 4. |
| honda civic | 5.00 | 5.00 | 4.00 | 5. |
| lincoln continental | 2.00 | 4.00 | 5.00 | 3. |
| plymouth gran fury | 2.00 | 1.00 | 3.00 | 4. |
| plymouth horizon | 4.00 | 3.00 | 4.00 | 5. |
| plymouth volare | 2.00 | 1.00 | 5.00 | 3. |

Listener

```
; loading Hal9000:ViSta:ViSta5.01
Devel:code:guidemap.lsp
; loading Hal9000:ViSta:ViSta5.01
Devel:code:guidemp2.lsp
; loading Hal9000:ViSta:ViSta5.01
Devel:code:guidance:data-ana.lsp
; loading Hal9000:ViSta:ViSta5.01
Devel:code:guidance:data-ana.lsp
```

Help: Topics

ViSta has an extensive help system. The HELP menu has four main topics:
1) MENU HELP explains what each menu item does.
2) SPREADPLOT HELP describes spreadplots, the multiple plot window visualizations displayed by the VISUALIZE DATA and VISUALIZE MODEL menu items.
3) SHOW GUIDEMAP shows maps which guide you through the steps of your data analysis.
4) Help for specific windows (LISTENER, WORKMAP, etc.)

**Figure 2. 8 Screenshot of ViSta**

As can be seen from Figure 2.8, ViSta is able to present multiple aspects of a data analysis side-by-side with many visual and non-visual options to customise a user's experience. ViSta's distinctive visual approach caters for the user's level of expertise by providing multiple modes of authoring statistical techniques. Novices are given an explicit step-by-step visual guide in the form of Guide Maps and a memory aid called WorkMaps. More experienced data analysts are free to use all five different types of analysis environment: Guide Maps, WorkMaps, and standard GUI based menus, command line and script. Amongst the five analysis environment, our discussion here will focus on the first two visually oriented modes.

## *GuideMaps*

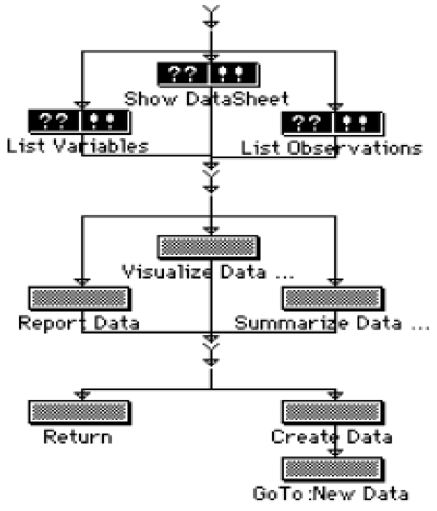


**Figure 2. 9 A GuideMap (Young and Bann 1997)**

A simple summarisation of GuideMaps can be a visually prescribed set of sequences for a data analysis. Figure 2.9 shows an example of a guide map for exploring data. Rectangular shaped buttons represent the steps that should be followed and arrow connectors indicate a flow of analysis. Users make use of visual cues to know the current status of a guide map. For example, Figure 2.9 shows that the user has three actions to choose from as indicated by the three highlighted buttons. Clicking on the !! side of a highlighted button either executes a data analysis step or navigates the user to another guide map. Context-sensitive help can be obtained by clicking ??. GuideMaps visualise an explicit analysis pattern that permits a novice an active role in exploring data and shows that when users are presented with appropriate contextual information and help, they may participate in a data analysis process more actively.

### *WorkMaps*

A guidemap has a static structure that does not change. In contrast, a workmap models a data analysis process from start to end and is built as the user carries out a data analysis.

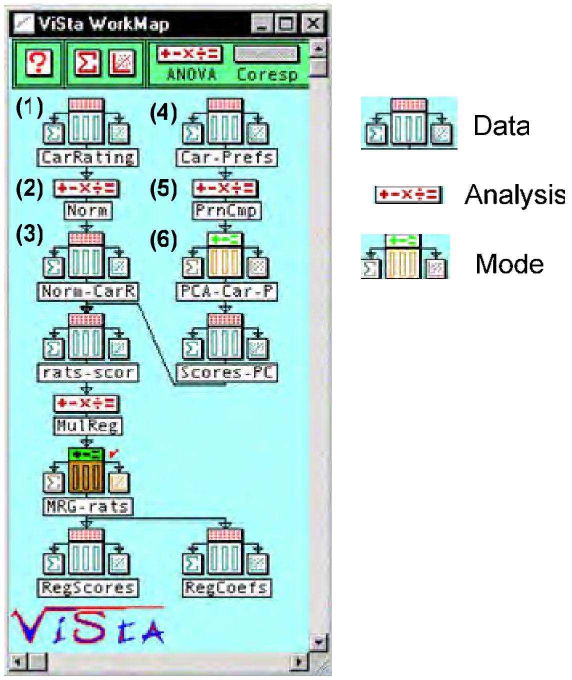


**Figure 2. 10** ***Example of a workmap (http://forrest.psych.unc.edu/research/vista-frames/workmaps.html)***

A workmap consists of three graphical icons: data, analysis and model. They represent statistical data, analytical technique and mathematical model respectively. The three visual icons are connected to represent an analytical flow in a data analysis process. Figure 2.10 illustrates the following analysis operation (partial):

1. Loaded the CarRatings data and it is represented by the data icon ‘CarRating’.
2. The data is normalised.
3. The normalised data is represented by the data icon ‘Norm-CarR’.
4. Loaded in data named and it is represented by the data icon ‘Car-Prefs’.
5. Perform a principal component analysis of these data

6. The mathematical model is produced as the result of the analysis and named PCA-Car-Pref

Once data is imported into ViSta, it is not just represented by a static visual icon but the user is able to view the data from multiple perspectives. Figure 2.11 shows how the data might be explored using ViSta's built in visualisation functionalities.

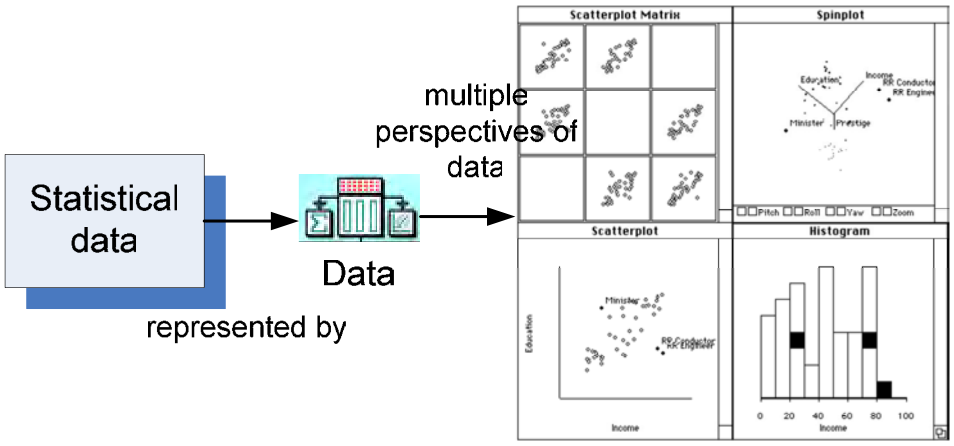


**Figure 2. 11 Various data visualisation techniques available in the ViSta statistics system**

***Developer's Perspective***

ViSta offers a rich set of extensions for users to customise and add new features. ViSta's underlying data analysis language ViDAL permits custom-made scripts to be used for various purposes such as Web Applet creation and data programs. ViSta itself is built on XLispStat extensions for statistical software development which is extensible by Lisp programmers. XLispStat is open and permits non-restrictive modifications. C++ and FORTAN routines are linkable with ViSta. Therefore as with its foundational technical components VisTa is extensible in virtually every aspect from incorporation of add-ins to extension of data analysis techniques.

***Evaluation***

We found many of the concepts and approach taken by the developers of ViSta to be highly innovative and it occupies a unique spot amongst statistical software packages. ViSta pioneered the use of multiple visual models to suit various user groups. A GuideMap helps user to carry out a data analysis even when they lack detailed technical knowledge of techniques involved in the analysis. A WorkMap not only visualises a data analysis but also captures valuable contextual information such as analytical steps taken by a data analysts. Additional dimensionalities afforded by a WorkMap facilitate the communication of a data analysis in a visual form. It can be seen later that SDL mirrors some of the concepts in ViSta diagrams closely.

During our evaluation of ViSta, the following aspects of ViSta emerged as ViSta's main drawback in using it in statistical surveys:

- Task based organisation is well expressed but provides little support in describing how tasks are merged together to support contexts of a survey such as objectives, data collection and data analysis.

- Task level organisation is well expressed but ViSta's visual models lack support for non-analytic activities that are not related to statistical data such as questionnaire design.

- Statistical metadata support is not provided natively. Statistical data can be viewed from various perspectives but associated metadata relationships are not expressed. For example, we may have sample data on student academic records but without statistical metadata such as population size and sampling method, the data may become useless for rigorous statistical tests.

- ViSta's Lisp-based extensions may not be well received by software developers and survey researchers who are content with dominant S-based statistical languages such as R in the statistics community (Narasimhan 2005)

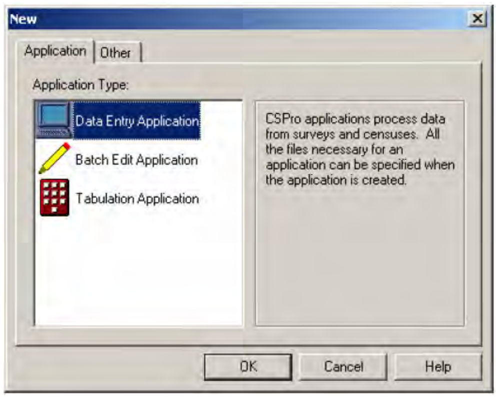
- ViSta is a statistical tool but it is not designed to integrate heterogeneous resources and back-ends architecturally.

The drawbacks are not unexpected as ViSta is not designed specifically for statistical surveys and its goal as stated in ViSta's research hypothesis is to provide a visually guided and structured environment for data analysis not to build an integration point for various survey activities. By contrast, our approach focuses first on building a suite of visual languages that are expressive enough to model survey specifications.

### 2.4.2 CSPro (Census and Survey Processing System, U.S. Census Bureau)

Recognised as the world's largest statistical organisation (ETC), the US census bureau develops and uses a public domain survey software called Census and Survey Processing System (CSPro) and has been used to administer statistical surveys not just in U.S. ,but also globally (CB). CSPro is a standard GUI software package and platform specific solution as it runs only on Windows systems. CSPro is neither extensible nor flexible as ViSta but it offers powerful logic programming capabilities. Even though CSPro's implementation approach is very different from ours and its capabilities are limited to survey data processing tasks, we considered the evaluation of CSPro to be a valuable exercise to form comparative insights into the survey process by observing how the survey process is supported by a software tool which must meet the needs of various large-scale survey types.

***Capabilities: Data Processing***
Capabilities of CSPro are strongly data-oriented that is to assist users from data entry stage to producing clean data which is free from inconsistencies, structural errors, or other errors.



CSPro applications process data from surveys and censuses. All the files necessary for an application can be specified when the application is created.

**Figure 2. 12 A start window for a new CSPro application showing three types of applications: data edit, batch edit and tabulation**

The starting window in Figure 2.12 shows three application types supported by CSPro and they are:

- Data Entry Application: to capture data in a questionnaire.
  This type of application enables users to create and design questionnaire forms which can be either a paper-based or computer assisted. Data entry assisting features such as data entry validation logic of unlimited complexity is supported. A created entry form can be run locally or deployed to other remote environments.

- Batch Edit Application: to produce error-free data file
  Utilising CSPro language, the user can build extensive and rich logical operations to produce set of files or reports. For software developers, this type of application can be analogous to XSLT mapping tool to convert one document into another to utilise the attributes of the newly created document. In practice batch edit applications are mainly used to remove errors in a data file so that the file can be tabulated or organised for publication.

- Tabulation Application: to produce publication ready data tables
  As the name tabulation implies, tabulation supporting features are used to create the relationship between data variables in an easy to use tabular form.

CSPro has not no data analysis functionalities so analytical tasks have to be carried out by other specialised statistical tools such as SAS and S-Plus.

### *Survey Process and CSPro*

In the background section we briefly discussed how the survey process is divided into multiple stages. Even the brief Overview of CSPro indicates it only covers some stages of the survey process selectively. It has no support at all for sampling design and data analysis stages. Thus it is not a solution platform for the user to integrate the multi-faceted aspects of statistical surveys however it is a very capable software tool in the areas it covers.

***Evaluation***

CSPro's vast applications in many survey types and free distribution of software make it a very attractive survey software package to use in practice. In addition to these positive factors, it has very powerful logic programming capabilities to facilitate efficient data file processing. However as stated in the previous paragraph, one of the main drawbacks of CSPro is its coverage and lack of support for survey planning and analytical stages. CSPro' main objective is to produce publication-ready data files, it has no explicit support to communicate the nature of a data operation to others. CSPro' approach is parallel to that of traditional GUI applications unlike ViSta it has no support designed for users of various backgrounds but has one mode of interaction for all. This implies rigid user interaction patterns that must be followed by the user. Extensibility of CSPro is rated low compared to ViSta. It is a largely closed GUI program that allows internal functionality extensions via its logic programming language but is closed to third party add-ins.

The Association for Survey Computing (UK) maintains a Register of Software for Statistical and Social Survey Analysis. For more information on software packages for statistical surveys refer to the ASC's register (ASC 2005).

### 2.4.3 Statistical Data and Metadata for Surveys

This section presents a brief review on the current state of the exchange of statistical data and metadata with a specific focus on statistical surveys. Then pragmatic approaches to bring about a seamless integration statistical data and metadata into our solution are discussed.

***Challenges in transferring surveys***

For most statistical organizations, moving information to and from word processors spreadsheets, PC statistical packages, databases, etc are routine (Tienhaara). However these common operational requirements can result in information loss. The loss can be contributed to a lack of means to interchange survey information other than at a data level (Lamb 2001). Statistical surveys are designed and executed to meet informational needs of end-users. The current options may suffice for day-to-day operations of survey practitioners in providing micro level data (e.g. tabulated datasets) across software package boundaries but metadata associates with various statistical activities is often lost and difficult to be reused in a heterogeneous environment.
In the development of software integration architectures, analogous problems instigated a trend toward an industry standards version of metamodels. Thus it is not surprising that a similar approach is gaining a momentum in statistical computing.

***Standards***

Before discussing some of the notable standards for statistical data and metadata it should be noted that there is no single established standard. The following standards or concerted standardisation efforts are geographically localised or discipline-specific. Understanding their approaches would give us requirements for incorporating statistical data and metadata into SDL.

*Triple-S*

Triple-S is an XML based standard which defines "a mean of transferring the key elements of the entire surveys between different software packages across various hardware and software platforms" (Tienhaara). Triple-S enables exchange of statistical data and metadata in the context of statistical surveys.
Triple-S has a structure which has been spearheaded by key developers from commercial survey software houses to encompass data and metadata and the structure is design to produce human readable and sufficiently compact documents in usual survey practices. Figure 2.13 shows the data structure of the triple-s standard in as a UML class diagram.

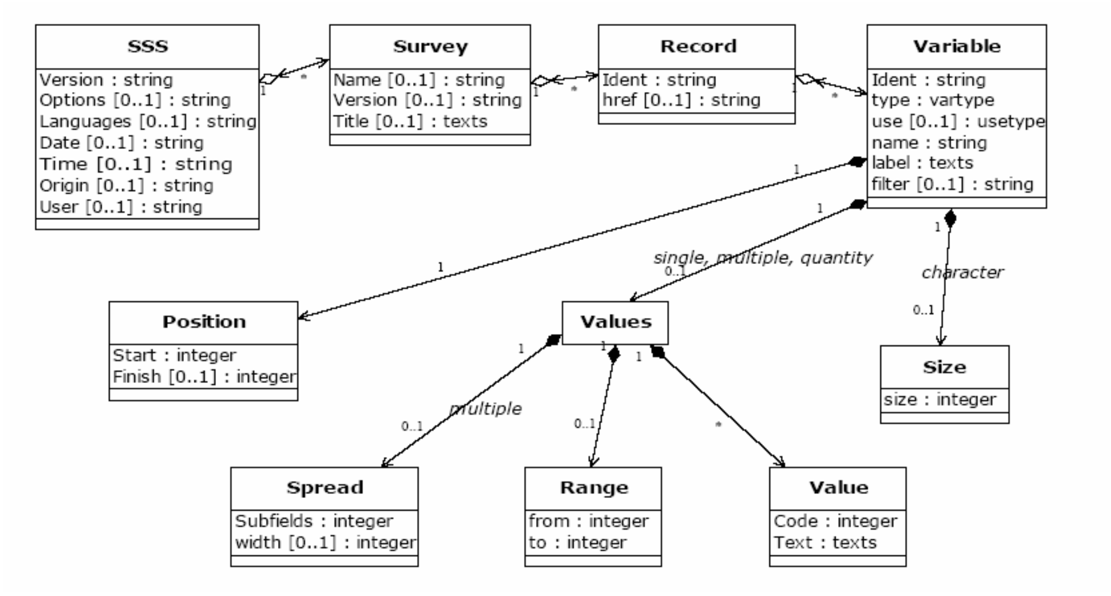


**Figure 2.13 UML class diagram for the Triple-S structure (MetaNet 2003 pp113)**

Survey data and related metadata are expressed within a 'Variable' element and general survey metadata precedes them. Triple-S is recognised as a survey interchange standard by OASIS. Triple-S offers the possibility of representing a part of the survey process without creating communication overheads between survey users and practitioners. However possible drawbacks would be these:

- Management issues for a more complex survey depends also on numerous process related data and metadata.
- Triple-S offers only the implicit semantics of structure that should rely on verbal agreements or by the use of textural comments.

*The Data Document Initiatives (DDI)*

The conceptual underpinning of DDI is a codebook for documenting datasets for social sciences. Since its inception, The DDI has gained a considerable body of collaborating organisations in the area of social sciences. The DDI is designed to be much broader in scope compared to Triple-S and requires extensive and detailed mandatory operations to archive meta-information. Despite some significant differences in their low-level implementations, both Triple-S and the DDI come across as offsprings of the early metadata repositories known as data dictionaries. As data dictionaries are data focused than information focused the two standards lack the level of abstraction above or beyond a physical statistical product. Thus in reality their purpose is biased towards supporting data archivists.

*MetaNet (MetaNet 2003)*
MetaNet is an EC supported project for harmonising and synthesising the development of statistical metadata. MetaNet covered numerous research topics that stemmed from a wide range of organisations intimately linked to statistical data and metadata and produced future directions in the following focus areas:

- Development of proposals for standards in the methodology used for describing statistical metadata and statistical information systems
- Development of proposals for recommendations on the metadata objects in a common conceptual model of statistical metadata
- To disseminate these proposed standards to the relevant user communities and standards bodies
- To interact with relevant projects on the development and agreement of these proposals, and to advise on methods of achieving coherence of approach in the field of metadata for statistical information systems
- To integrate the different views of metadata into one model and bring together these different perspectives.

Though MetaNet's final outcomes were mostly initiatory in nature, it serves as one of the most extensive bodies of work available in the area of statistical data and metadata research and provides valuable insights for the development of SDL and related tools.

### *Implications for Our Research*

Representing statistical data and metadata can be a difficult task. There are many factors to consider, such as to what types of metadata we need to convey, who is going to use it and how it functions in the survey process.

In this section, we will describe how the preceding researches may impact the task of incorporating statistical data and metadata. Triple-S is clearly a leading standardisation effort for surveys judging by its history and acceptance by OASIS. Therefore Triple-S is an obvious choice for our particular development environment and requirements for our prototype tool. Seamless bidirectional integration of Triple-S XML and the prototype should be a part of final achievements.

MetaNet's metadata reference model and proposals supersede DDI in both scope and depth in general. Throughout the integration process some of the valuable findings/proposals of MetaNet such as:

- Microdata and macrodata flows in statistical activities and
- Additional dimensions to describe statistical data.
    - Structure
    - View

- Form
- Function

C.f. Currently SDL only expresses the structure of statistical data using a data entity tree. The additional dimensions will be incorporated in the revision of SDL.

The review of literature related to the standards showed many positive notes on the overall direction of this thesis work. Especially some of the outcomes of MetaNet's workgroups have validated our core approaches in working toward the visual environment for surveys as follows:

1. UML as a modelling tool
The MetaNet project draws the conclusion that the UML or similar equivalent is the correct tool for building models of statistical processes and the associated metadata. However it concludes that two shortcomings of the UML regarding the domain specificity of the UML and end-user usability would work against the UML use in practice. The underlying design philosophy of SDL which closely follows that of the UML should be a big asset in offering survey practitioner usable expressiveness at low cost.

2. MetaNet recommendations for a conceptual framework for statistical metadata.
"To be valid, a conceptual framework has to overcome the hitherto predominating view that statistical metadata is to document existing datasets only, and that all other relevant statistical objects – like statistical populations, or classifications – are considered subordinate to these datasets implying, correspondingly, a "dataset-attached" mode of documenting these objects." (Froeschl, Grossmann, and Vecchio 2003)

SDL falls outside the predominating view as it focuses on modelling multiple aspects that make up a survey rather than centring on statistical datasets. Further development to collaborate with survey exchange standards such as triple-s and related efforts should provide a tangible opportunity for survey developers to move forward in building a unified platform that could support every stage of the survey lifecycle.

## *2.5 Summary*

In this chapter we have presented the background material and related research for our research. We discussed the survey process and the source of complexity in practice and how existing statistical packages cannot deal with the semantics loss. This aspect of the survey process was looked at from the perspective of statistical metadata management in the related research section.

To illustrate our visual approach to the problem, the nature and relevancy of the UML were presented. The imprecise semantics in interpreting UML diagrams when they are not augmented with OCL led us to investigate a means to give visual representations a solid ontological basis. TopicMaps were presented to explore their possible usage in integrating the diverse concepts which must be expressed by our visual language.

ViSta the visual statistics system was one of two software packages to review and critique its design and functional capabilities. ViSta has remarkable capabilities to put novice users in control of very technical tasks and offered superb flexibility via five types of tool interaction modes, two of which are visually oriented. However ViSta is not a tool specifically designed for surveys and lacked core non-analytical functionalities, which we identified as a must to support the survey process, to be useful in the overall survey process.

CSPro was evaluated and was excellent in what it does. But as we discussed in the beginning of this chapter, it can be grouped with many other survey software packages for being narrow focused and having little support for integration and collaboration in the heterogeneous environment of the survey process.

The broad theme of this chapter will be seen in the next chapter as the motivating forces behind the development of our visual language.

# Chapter 3 Survey Design Language (SDL)

## *3.1 Introduction*

This chapter introduces all the SDL diagrammatic notations and diagrams. It also discusses the design principles of these SDL diagrams and their relationships with organizational, activity-related and technical aspects of statistical surveys to clarify the motivational forces behind SDL diagrams. It then focuses on the general nature, features, and applicability of the SDL diagrams. Working examples, which are based on the national survey, accompany the language descriptions to view the diagrams from a practical perspective. The discussion of inter-diagram relationships and their mappings to a unifying model are further developed in the following chapter.

## *3.2 The General Overview of SDL*

### 3.2.1 Developing SDL

The suite of SDL diagrams aims to provide visual models for statistical surveys. The first step in achieving the primary goal of SDL was to establish a foundational base to set the right course for core design principles of SDL. A careful review of the body of literature, both academic and commercial, for software modelling and visual language was undertaken to build up the corpus of research findings that shaped many aspects of SDL at various levels of developmental stages. Particularly, the contributions of object-oriented modelling, UML and UML-based model driven development to the development of SDL were imperative in providing the crucial models for visual communication. Hence given the important role of UML in our research and in many disciplines of software engineering, the overview of SDL makes references to the syntax and semantics of the UML when appropriate to aid the communication of the research and development work behind SDL. In supporting the emerging object-oriented analysis and design methods (OOAD) in 1990's, the UML largely resolved the heterogeneity in visual OOAD and as result the UML amalgamated several key streams of research in OOAD. The UML has reached a stage of maturity where it is possible to model complex interactive systems using UML diagrams (Goldin, Keil, and Wegner 2001) in diverse aspects of OOAD. Thus the nature of the UML was a valuable template for what SDL should be.

As presented in the preceding chapter, the survey process has multi-faceted sources of complexity which are introduced through the wide array of the communication inconsistencies. In addition to the communication related problems the inception of large scale survey has brought about enormous managerial complexity that is akin to those commonly found in business modelling. By putting the motivations of object-oriented modelling in parallel with the issues of the survey process, the initial problem-solution mapping for SDL is formulated.

The refinement of the problem-solution mapping closely resembles the historic outcomes of the UML development.

### 3.2.2 SDL Representations for Statistical Surveys

In this section we will deal with the necessary nature of SDL diagrams, in the area of the cognitive underpinnings of SDL and inter-model consistencies.

Firstly we should understand what the required mappings from statistical surveys to the visual framework of SDL are. The necessary mappings can be conceptualized as the creation of metaphors for statistical surveys. The metaphorical description of visual modelling can be applicable to diagrams in general as theories of conceptual metaphor have been explicitly extended from language to diagrams (Blackwell 1998, Lakoff 1993). Thus from notable diagrams such as a flowchart we can demonstrate the metaphoric connections between source and target domains. For example a flowchart includes the following source-to-target mappings (non-exhaustive):

- Diamond shapes for decision points
- Rectangular boxes for actions or processes
- A directed arrow for the direction of flow

SDL diagrams visualize the cognitive structure of survey activities for the survey process in the way that two conceptual domains (target and source) are employed by a conceptual metaphor. Our view of visual diagrams' nature as cognitive tools (Blackwell 1998) was instrumental in setting up a firm starting point in the implementation of SDL diagrams.

Cognitive tool models for statistical surveys include:

- the survey requirement model, which describes the behavioural and functional aspects of the survey requirements in consideration.
- the survey data model, which organizes statistical data and associated statistical metadata and provides a metaphor for the manipulation of the data.
- the survey task model, which defines a generic task (Sutcliffe 2002) that is carried out by users.
- the survey procedural model, which integrates the task metaphors and orchestrates them with constraints.

Those cognitive tool models have been incorporated into a suite of SDL diagrams and the multiple perspectives, which are afforded by the multiple diagrams, parallel the development of the UML to convey a complex system by incorporating a set of independent views.

The second area of concern is of the problem of inter-model discontinuity which makes difficult to integrate models as a whole (Medvidovic et al 2001). Each visual model can be independent and yet it is only a partial view of the entirety, and so it is not a sufficient integration point for the survey process. The UML diagrams are bounded by OMG's meta-model (Unhelkar 2005) which provides a unifying integration model. Without the underlying meta-model many features of the UML that are driven from the meta-model specifications such as versatility, rich extension support and model consistency checking capability can be negated. Given the relevance of the UML and the multiple SDL models to be visualised, it is imperative to realise the necessity of an underlying meta-model that can lay down explicit specifications to maintain a unifying view from the diagrammatic diversity. This topic will be discussed further in the next chapter.

## 3.3 Purposes of SDL

The four main goals of UML (OMG 1999) reflect the industry wide consensus for a common language to aid the development of IT projects throughout the entire lifecycle. Historically UML stands in the mainstream of visual software modelling over the past four decades that started with the ever-present fountainhead flowcharts. UML's evolutionary path shows some of reactionary steps taken by its designers to provide solutions to the modelling difficulties which originated from the emergence of new logical frameworks (e.g. OOA, MDA) and technological developments (OMG 2004). The historical outline of UML provided a valuable insight into the essential purposes of the visual language that deals with a complex multidimensional problem. Thus it is no coincidence that the purposes of SDL closely parallel that of UML at high level. The following categorical purposes of SDL are strictly within the context of statistical surveys and are not exhaustive.

*Visualisation*
SDL notations and diagrams provide visual representations for the multiple aspects of a statistical survey. The scope of the visual modelling spans the entire survey process. Each diagram visualises a specific aspect of a statistical survey. In other words each diagram's visual model signifies an important dimension in communicating statistical surveys. This recurring theme of the multiple aspects of a statistical survey will be discussed in the next section.

The visual development of SDL diagrams is driven by the development process for statistical survey, which is presented in Chapter 2. The visualization process starts by conveying survey contexts and attributes based on requirements that survey designers and stakeholders may have. Due to the temporal position of the process it inevitably accompany consensus forming activities in the areas of requirements analysis, scope of functionality, problem investigation and the overall flow of the survey process. Hence the initial stage of visualisation centres on the high-level organisational visualisation of statistical surveys and becomes a starting point for other diagrams which are designed to provide more functional and structural refinements of visual elements that are part of the high-level visual description of statistical surveys.

The next step of the visualisation process maps the visual information acquired by in the initial stage of the visualisation into more concrete visual definitions that are closer to the language of statistics and software constraints. The mapping permits the transition of visual representations to working implementations in existing statistical computing software tools.

*Specification*
All SDL diagrams as a whole assemble specifications of various survey artefacts. Thus a statistical survey is conceptualised as the collective relationships between the survey artefacts. The visual representation alone can be inexact as in many instances the amount of visual information should be curbed to prevent the complexity of visual annotations from reducing diagram comprehension. Thus the additional diagram layers which permit users or software tools to delve into additional descriptions that are not available from visual diagrams, are incorporated into SDL. Chapter 4 discusses the relationship between the additional layers and the visual layer of SDL diagrams.

Specifications can be of the following types:

- Relationships between survey artefacts: Two or more SDL diagrams may be involved in inter-diagram relationships via a common survey artefact.
- Resource description: Mapping of survey artefacts to physical resources or services such as raw datasets and web services.
- Attributes: Properties associated with a survey artefact.

*Survey implementation and execution*
Just as MDA driven solutions utilise UML-based models to actualise software artefacts, SDL diagrams can also be used to construct supporting software solutions for a statistical survey. For example, diagrammatic specifications may be turned into corresponding web services which can be orchestrated according to the specifications generated from visual diagrams.

*Survey documentation*
Survey documentation is an intrinsic part of the survey process. Lack of common standards or operational difficulties in imposing an explicit formula over texturally verbose activities often present difficulties in resolving presentation heterogeneity (Olenski 2003). Considering many surveys heavily overlap in their topics and research methodologies, we may encounter many instances where a simple search to review existing surveys for a particular topic (e.g. National crimes survey) incurs unnecessary costs in terms of human efforts. Also a considerable portion of post-data collection activities do not require detailed documentation to fulfil operational goals (e.g. Coding SAS or R procedures, software package settings, metadata management) thus making documentation for those activities relatively scarce in reality.

SDL offers visual diagrams that can largely automate the documentation of the survey process – not only the activities where are traditionally well supported by documentation, but also the activities which manual documentation can be perceived as being of secondary importance. The nature of SDL diagrams (multi-aspect and layered) plays a big role in the richness of documentation that can be achieved across multiple survey aspects and present a novel perspective on survey documentation systems. SDL documentation system can convey a wide range of survey aspects from sampling to data analysis and a multi-layered approach can be taken to disseminate metainformation at varying abstraction levels.

Legacy documentation strategies resemble the reporting process which takes a complete form only after primary activities are accomplished. SDL's native support for its dual usage as an executable software tool and a communication medium means that documentation can be done in conjunction with main survey activities such as a sampling design without explicit human inputs. This mirrors some of outcomes of MDA based software solutions. When executable code is created from UML-based models, problem semantics expressed by the visual models are transposed to the generated code thus largely negate needs for manual documentation of the code. Users can always revert back to the visual models to comprehend standing views that manifest behind the generated code.
Likewise outcomes of SDL based survey developments always have corresponding visual models and specifications that can be utilised to mine textural documentation when it is required.

## *3.4 Aspects of Statistical Surveys and Multiple Diagrammatic Representations*

SDL consists of a suite of five diagrams to model statistical survey. In section 3.2.1, we stated that the reason for having multiple diagrammatic representations was due to the multi-faceted nature of the survey process. In this section, we will elaborate on the foundational design decision and the research influences in devising SDL diagrammatic representations. The use of multiple diagrams to form an integrated model is one of the central problems of our research.
Some of the foregoing questions in investigating the validity of using a suite of diagrams to represent statistical survey are:

- How the multi-faceted nature of the survey process should be addressed in the context of visual representations of statistical surveys?
- Can we design a universal diagrammatic system for the survey process without relying on the need for multiple diagrams? (One diagram type to model all aspects of the survey process)
- What are some of the relevant real-world examples to our design problems?

Our answers for the preceding list of questions base much of their core viewpoints on the necessity of multiple modelling spaces (Unhelkar and Henderson-Sellers 2004) and heterogeneous reasoning (Barwise and Etchemendy1995) in modelling complex multi-dimension problems.

*Modelling Spaces*
A modelling space can be briefly defined as the area which modelling needs to take place. Thus many real-world UML-based modelling activities happen within multiple modelling spaces since each diagram is designed to depict a specific perspective. It has been proposed by Unhelkar and Henderson-Sellers (2005) that in managing the problem in systems engineering distinctive but related modelling spaces (problem, solution and background) considered within the context of UML provide a much more robust model, as the problem will be analysed from the three points of view

- To understand what the problem (Model of problem space)
- To create a solution to the problem (Model of solution)
- To influence the two purposes from background based on constraints (Model of background)

A simple elementary problem can be broken down into the three modelling spaces as follows. Assuming that we need to come up with a file sharing scheme for an organisation, within the model of problem space all models to study the problem itself independent of any software and hardware platforms reside, within the model of solution space all models to provide real concrete software solutions reside (e.g. P2P Network server and client) and the model of background space apply constraints such as costs, maximum bandwidth and minimum throughput for the both model spaces. Due to the highly elastic nature of systems modelling the findings related to the systems modelling can be seen within the context of the survey process. Let us expound on the three modelling spaces and their implications to diagrammatic models for statistical surveys.

Two angles were taken in investigating the relationship between the three modelling spaces and the use of multiple diagrams. First our investigation focused on the realisation that the survey process spans across the three model spaces. The following Figure 3.1 illustrates how the previously discussed survey process structure falls within the three modelling spaces.

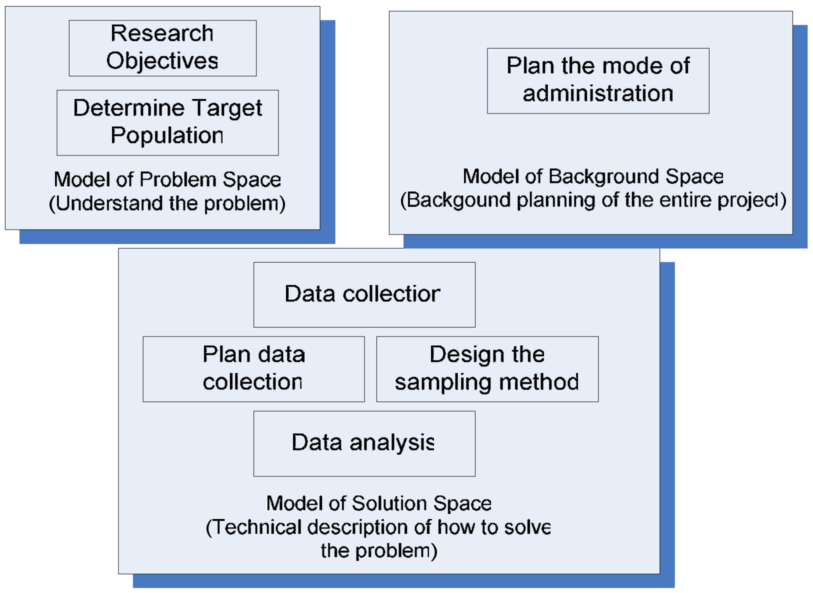

Research Objectives
Determine Target Population
Model of Problem Space (Understand the problem)
Plan the mode of administration
Model of Background Space (Backgound planning of the entire project)
Data collection
Plan data collection
Design the sampling method
Data analysis
Model of Solution Space (Technical description of how to solve the problem)

**Figure 3. 1 Three modelling spaces of SDL**

As each UML diagram has it own appropriateness to the modelling spaces (Unhelkar and Henderson-Sellers 2004), it is imperative we have the right set of diagrams to model the survey process within the context of the three modelling spaces.

Alternatively we could have just one universal diagram to cover all the three modelling spaces but it goes against the following important factor. According to

Barwise and Etchemendy (1991) diagrams are physical situations and a representation scheme should ensure a good match with the constraints on the descried situation. The apparent operational and conceptual divisions within the survey process create multiple situations to be modelled. For a single diagram it would be difficult to ensure the requirement for appropriate diagrammatic constraints as the contextual scope of the diagram has to be expanded to incorporate all those situations.

Secondly the concept of heterogeneous reasoning by Barwise and Etchemendy (1995) was instrumental in refining our approach to the survey modelling using a suite of diagrammatic representations. The survey process is a complex and multi-dimensional affair. Barwise and Etchemendy (1995) showed that in considering the design approach to model such a state, the clarity and utility of the diagrammatic system can be greatly improved if each of representational schemes is highly "homomorphic" (structurally closely mapped representation) to the aspect of the situation they represent while a universal diagrammatic system does the opposite in many real-world instances. Thus "heterogeneous" reasoning system is often an ideal choice to cope with multiple aspects of the problem to be modelled and this was a clear indication to steer SDL to the multi-diagrammatic survey representation system.

## *3.5 SDL Diagrams*

In this section, we introduce five types of SDL diagrams. Each SDL diagram is presented in terms of language syntax, semantics and applicability. The logical breakdown of each diagram into the syntactic and semantic parts is expanded in the next chapter. The syntactic part of SDL diagram lays down the rules for the manipulation of diagrammatic notations. The semantics of SDL diagram defines meanings for structural and conceptual constructs of a diagram. One example is a dataflow metaphor expressed with incoming and outgoing connectors.

The main purpose of this section is to introduce a diagrammatic notation for communication about statistical surveys. The expositional study on SDL diagrams should include what we do not see such as the metamodel behind the diagram but we limit our discussion in the section to the aspects of SDL that are visually expressed as a communication medium. All SDL diagram types that we present in this section are designed to be integrated together to form the unified view of the survey process.

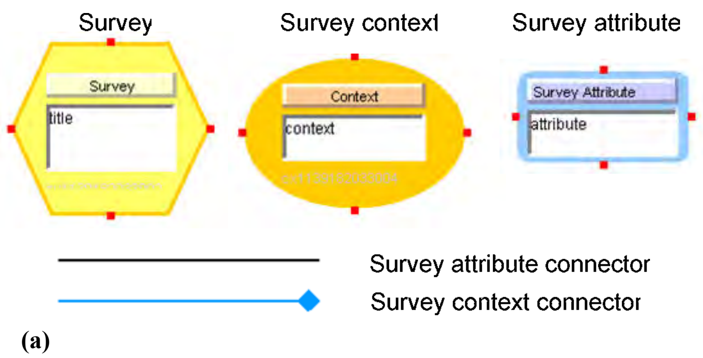
We will begin with the functional nature of SDL diagrams and elaborate on the main concepts that are conveyed via the diagrams briefly. Working examples are supplied for each of them to demonstrate how all of them can be put together in real practice.

### 3.5.1 Survey Diagrams

Survey diagrams are the first diagram to be produced during the initial planning stage of the survey process. A survey diagram's goal is to provide a visual model to analyse the survey process as a whole at a very high level thus it lacks any low-level technical descriptions but focuses on delivering overall contexts within the survey process and context attributes which are posited by them. Generally verbose survey design documentations can be visualised in the form of a survey diagram. Survey diagrams expose the overall survey process from various stakeholders' perspectives to facilitate brainstorming processes at the beginning then casual nature and minimal learning overheads reflect this nature of survey diagrams.

Composing a survey diagram begins with a visual icon which represents a statistical survey. A context is a generalised mental construct that constitutes the semantic meaning of a survey and it binds attributes of a survey together. Attributes can be survey operations or stem purely from data. Survey contexts, which usually can be identified from lexically frequent survey topics in survey design documentations, are attached to the survey icon. Usual contexts of a survey are objectives, subjects, and survey procedures. Those contexts are the usual building blocks used by survey researchers to plan a survey thus thinking in terms of survey contexts would not be a significant mental operation by key persons involved in a survey. Note that the process of composing survey diagrams is designed to be informal and can be very rapid. This comes at the price of losing rigour in specifying terms like "context" but the loss of rigour is expected to be recouped in the speed in facilitating group consensus.

The diagrammatic notions for survey diagrams are given as follows:



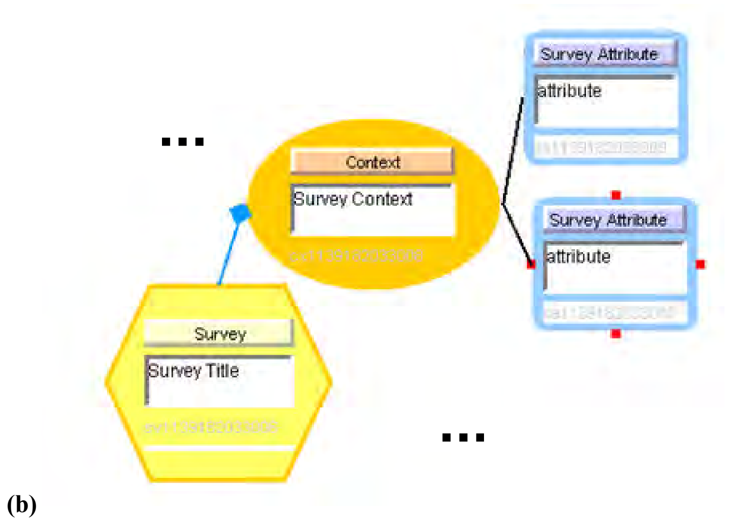


**Figure 3. 2 (a) survey diagram diagrammatic notations (b) partial snippet of a survey diagram**

A survey icon is depicted as a hexagon. Survey contexts are associated with the survey icon. Survey contexts may involve interactions between survey attributes. These interactions, depicted by directed arrows, show how a certain context of the survey process is shaped by multi-faceted attributes within the context.

A survey context consists of a collection of survey activities to support it and survey activities are conceptualized as survey tasks. This is one of the most significant features of a survey diagram as it opens a pathway to the task model which binds statistical data and techniques together. The relationship is one of the main discussions in the next chapter.

**Developing Survey Diagrams: Victimisation Survey**

Figure 3.3 shows a survey diagram for a survey to study victimisation data. There are five contexts – purpose, target population, implications, procedures and results - that make up the survey. The contexts are shown as oval shapes and each oval shape is relationally connected to attributes. At the centre of the diagram is a hexagonal shape, which represents the survey. Contexts are connected to the survey shape by circle-ended connectors and attributes are connected to contexts by arrow shaped connectors.

Figure 3.3 demonstrates well that how the entire survey can be summarized using a survey diagram with only handful of graphical components.

Since the victimisation survey will be used throughout this chapter as an example survey, let us present the overview of the survey. The survey follows the theme of the national survey, which was carried out by Statistics New Zealand. In 2001, samples of New Zealand residents were chosen to gather variety of crime victimisation information.

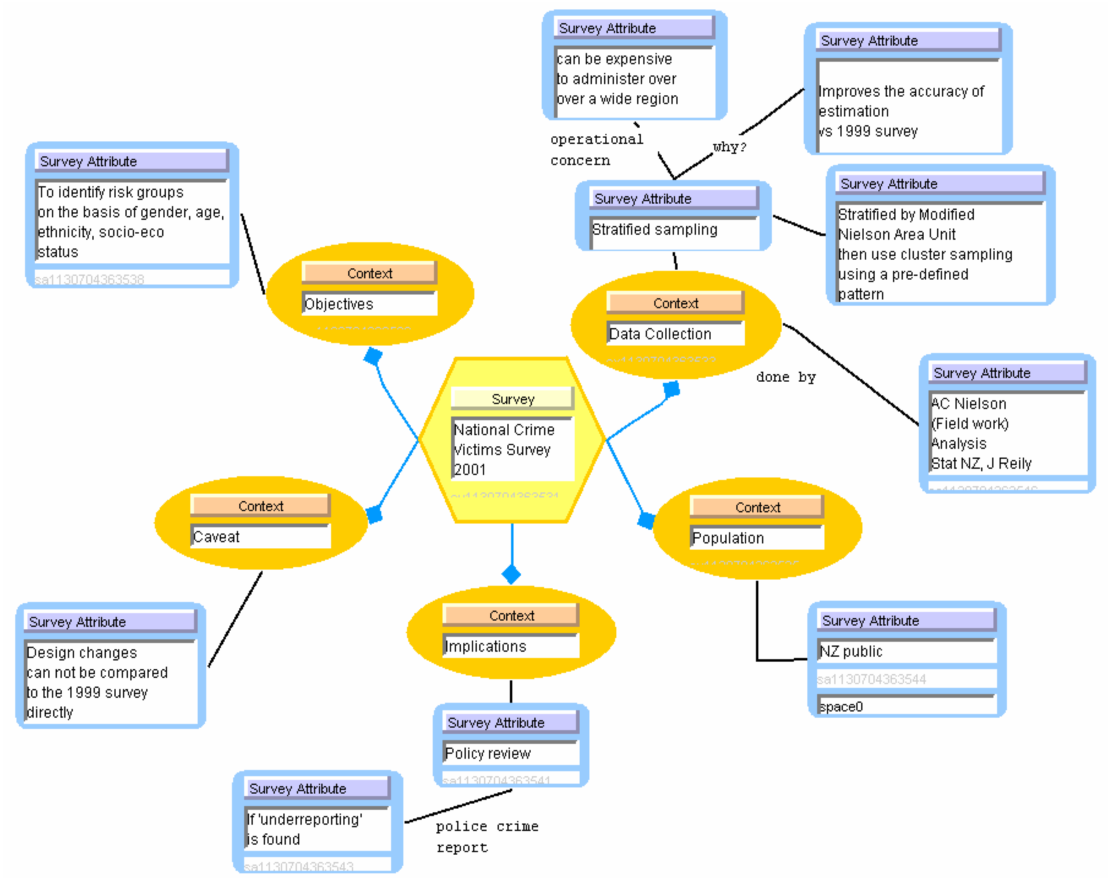

Survey Attribute
can be expensive to administer over over a wide region
Survey Attribute
Improves the accuracy of estimation vs 1999 survey
operational concern
why?
Survey Attribute
To identify risk groups on the basis of gender, age, ethnicity, socio-eco status
Survey Attribute
Stratified sampling
Survey Attribute
Stratified by Modified Nielson Area Unit then use cluster sampling using a pre-defined pattern
Context
Objectives
Context
Data Collection
done by
Survey Attribute
AC Nielson (Field work) Analysis Stat NZ, J Reily
Survey
National Crime Victims Survey 2001
Context
Caveat
Context
Population
Context
Implications
Survey Attribute
Design changes can not be compared to the 1999 survey directly
Survey Attribute
NZ public
space0
Survey Attribute
Policy review
Survey Attribute
If 'underreporting' is found
police crime report

**Figure 3. 3 Survey diagrams for the victimisation survey**

The identified core contexts of the survey are laid out around the survey shape then contexts are further expanded by adding context attributes.
We can describe the survey diagram in Figure 3.3 as follows:

- Objectives of the surveys are to find out risk groups.
- The target population of the survey is the NZ public.
- The survey could be used in future policy reviews.
- Stratified sampling was used to sample subjects.
- 2001 survey results must not be compared directly out of context to prior crime surveys as there were significant design changes.

The illustration of the interaction between multiple attributes is an important tool in conveying design decisions, which must consider possibly conflicting options side-by-side, in a very compact form.

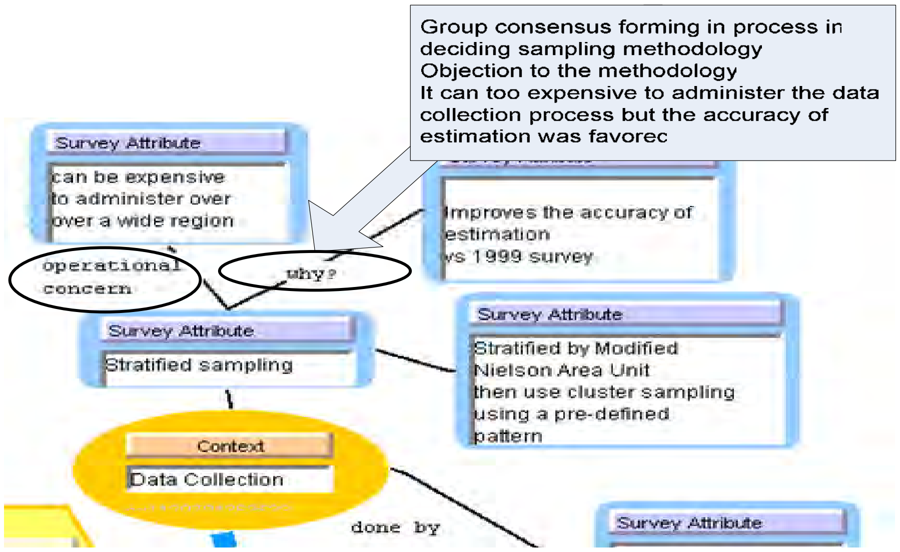

Group consensus forming in process in deciding sampling methodology
Objection to the methodology
It can too expensive to administer the data collection process but the accuracy of estimation was favorec
Survey Attribute
can be expensive to administer over over a wide region
operational concern
why?
Improves the accuracy of estimation vs 1999 survey
Survey Attribute
Stratified sampling
Survey Attribute
Stratified by Modified Nielson Area Unit then use cluster sampling using a pre-defined pattern
Context
Data Collection
done by
Survey Attribute

**Figure 3. 4 Using the survey diagram in consensus forming activities**

A survey diagram's scope covers the entirety of a survey and expressed at the very high-level. There are no rules to how abstract or explicit a context is expanded. The relaxed approach has both advantages and disadvantages due to the varying levels of abstraction across contexts. This will be discussed in detail in section.

### 3.5.2 Survey Data Diagrams

Survey data diagrams along with survey technique diagrams are arguably be the most immediately relevant form of SDL diagram to survey statisticians and data analysts alike. Survey data diagrams model statistical datasets, metadata and data operations (e.g. sampling). Major communication concerns addressed by survey data diagrams include the visualisation of data flows, data operations, statistical metadata and the population to statistical data relationships. Survey data diagrams show data-level details as well as data-level operations which are usually derived from sampling techniques. In addition to showing statistical datasets using just one visual layer, survey data diagrams allows the relationship between statistical datasets and their metadata to be described using multiple diagram layers. The main theme of a survey data diagram is set by datasets (rectangles) and data operations (ovals). The pentagon icon represents data probing activities such as getting descriptive statistics on a dataset. The data probe shape is not one of the main visual constructs of the survey data diagram in modelling the sampling process. However it plays an important auxiliary role as the outlet which can be used to obtain frequently requested statistical descriptions on a dataset. A dataset shape represents a statistical dataset. These shapes may be layered into three compartments as follows:

- Dataset: Real physical dataset in a common two dimensional table form.
- Dataset Metadata – Triple-S: Standards for preserving statistical metadata such as questionnaire details, response types, default empty, answer codes, default value, etc.

- Dataset to population relationship: Statistical samples are taken from the population and the relationship between them are visualised as tree structures with composite connectors indicating a sample's membership status.

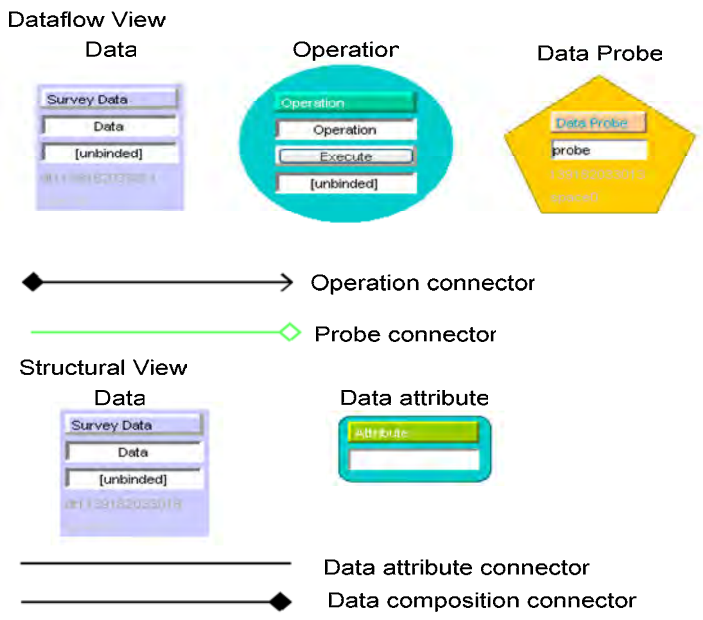
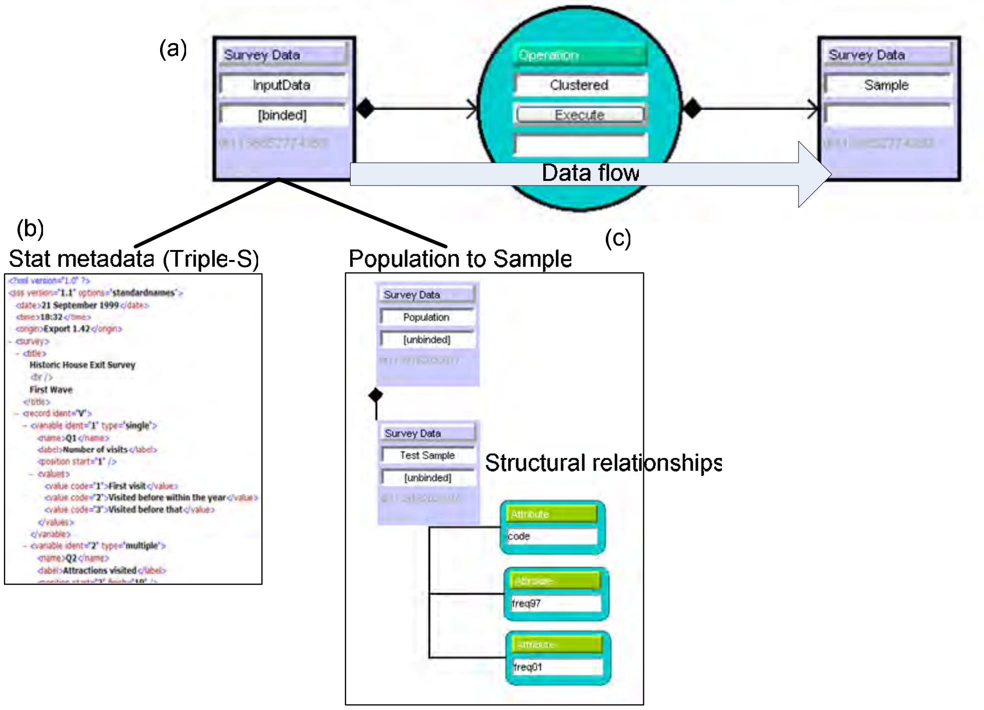


**Figure 3. 5 Diagrammatic notions for a survey data diagram**



```
<?xml version="1.0" ?>
<sss version="1.1" options="standardnames">
  <date>21 September 1999</date>
  <time>18:32</time>
  <origin>Export 1.42</origin>
- <survey>
  - <title>
      Historic House Exit Survey
      <br />
      First Wave
    </title>
  - <record ident="V">
    - <variable ident="1" type="single">
        <name>Q1</name>
        <label>Number of visits</label>
        <position start="1" />
      - <values>
          <value code="1">First visit</value>
          <value code="2">Visited before within the year</value>
          <value code="3">Visited before that</value>
        </values>
      </variable>
    - <variable ident="2" type="multiple">
        <name>Q2</name>
        <label>Attractions visited</label>
```

**Figure 3. 6 (a) a simple survey data diagram (b) some of extra dimensions that can be added to a dataset Triple-S survey metadata and population to sample relationship)**

Statistical datasets in the usual raw table forms have very little value for users who have no access to associated statistical metadata. Thus all three aspects should be integrated and complement together to form a complete view for the dissemination of datasets. The management of statistical metadata is a complex subject of its own and it is one of the core issues that large statistical organisations face as daily operational hindrances. SDL permits all relevant statistical data and metadata to coexist so that users can have a complete view of statistical data represented by a data icon. The mechanism of the integration is presented in the next chapter.

The survey data diagram's layers offer elision and information hiding functionality rather then permanently presenting all three aspects to be displayed side-by-side. As mentioned in the beginning of this chapter, we envisaged tool support for SDL and this impacted its language features especially in the area of diagram layering. The different degrees of visibility for the three aspects of statistical datasets are not easily supported using a static "pen-and-paper" mode since we have to allow seamless navigations between the underlying three aspects of statistical datasets. The investigation on this particular language feature is presented as part of the user testing phase in the cognitive framework.

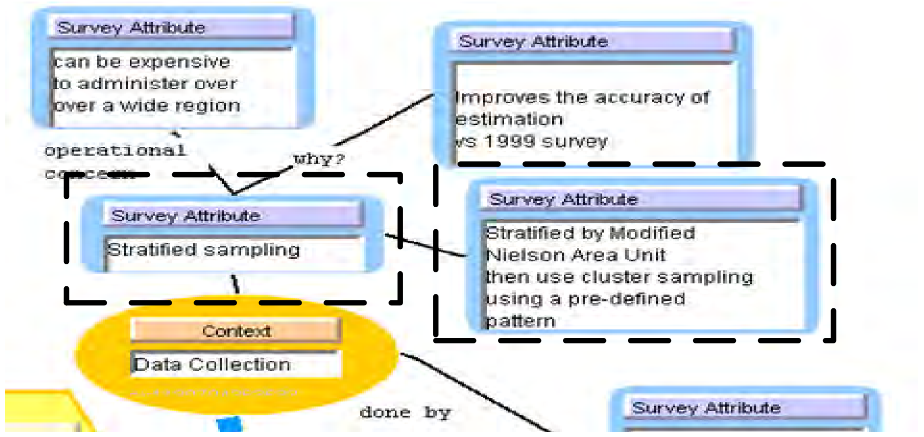
Dataflows are shown using directed connector as in Figure 3.6 (a) except auxiliary data probe connectors. Dataflows model the process of how one dataset is transformed to another by one or more data operations. Dataflows are the most general form of relationship in survey data diagrams.

We consider a survey data diagram's role in several stages of the survey process. Survey diagrams are used during the sampling design stage when survey designers need to determine appropriate sampling strategies. Survey diagrams are also used to put frequent sampling process patterns into a common repository and publish sampling designs to peers for both validation and reuse. As presented in Chapter 7, all SDL diagrams can operate as tools, this makes a survey data diagram to be an interactive tool to do work with live statistical datasets.

Survey data diagrams are related to survey technique diagrams since datasets modelled in survey diagrams can be reused as inputs to a statistical technique. The task model may have survey data diagram's datasets as its task outputs as shown in Figure 3.11.

**Developing Survey Data Diagrams: Victimisation Survey**
The relevant aspect of the survey can be easily seen (dotted survey attributes) in the survey diagram as in Figure 3.7(a) without going through a verbose textual description of the survey. The two survey attributes provide sufficient information to begin developing a survey data diagram. Our sampling design uses a stratified sampling (stratified by the area unit). A simple random sample is taken from every stratum.

(a)


Survey Attribute
can be expensive to administer over over a wide region
operational concern
why?
Survey Attribute
Improves the accuracy of estimation vs 1999 survey
Survey Attribute
Stratified sampling
Survey Attribute
Stratified by Modified Nielson Area Unit then use cluster sampling using a pre-defined pattern
Context
Data Collection
done by
Survey Attribute

(b)

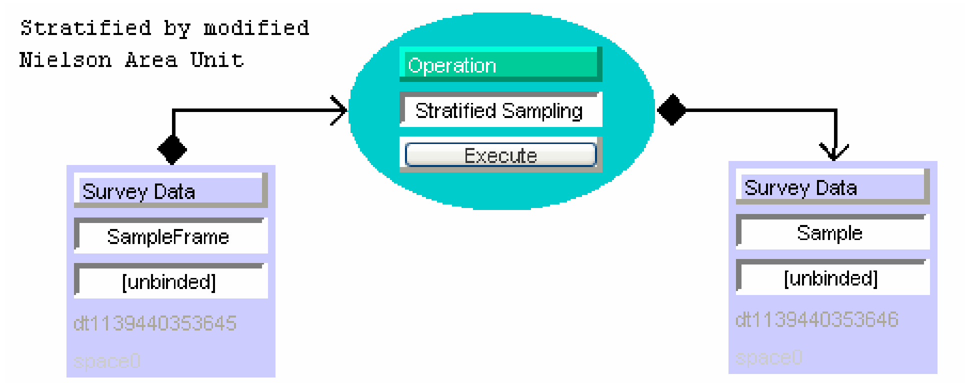


**Figure 3. 7(a) relevant survey attributes for the survey data diagram shown below (b) survey data diagram which depicts a stratified sampling process**

The next step in modelling the sampling process is to depict the sample to population relationship. This additional layer of information is associated with the survey data icon ‘Sample’ at the right in Figure 3.7 (b).

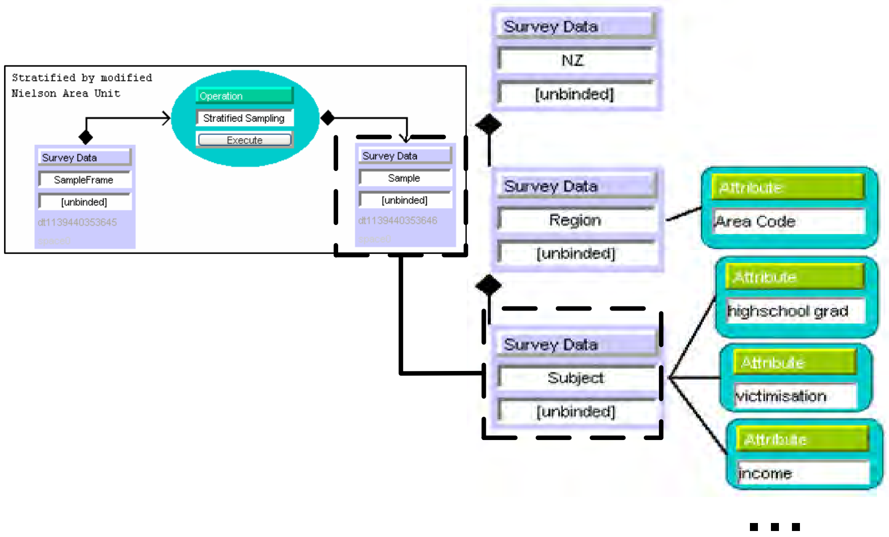


**Figure 3. 8 Sample to population relationship can be visualised as shown in the right hand side**

The diagram visualises how sampled subjects are related to the population and types of attributes that describes sampled subjects.

### 3.5.3 Survey Technique Diagrams

At a glance survey technique diagrams visually resemble survey data diagrams as shown in Figure 3.10. This is partly because a dataflow metaphor is used in both of the diagrams. However there are some significant differences between the two diagrams to separate them into two distinctive modelling spaces. In survey data diagrams the dataflow literally signifies the stream of datasets that are to be manipulated by sampling operations (e.g. selecting every nth data rows). This implies real physical changes in datasets. However for survey technique diagrams no such analogy can be made since generally statistical techniques imply only exploratory activities without explicit changes in datasets.  Some categorical examples of common statistical techniques are regression analysis, multivariate graphing, and multivariate data exploration.

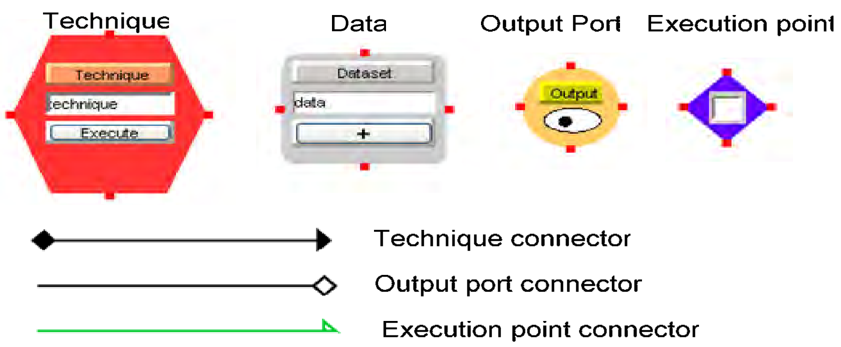


**Figure 3. 9 Diagrammatic notions for a survey technique diagram**

An important key characteristic of a survey technique diagram can be seen in Figure 3.10. We have data flows (directed connectors) in and out of the technique (hexagon) and the technique output port (circle). The hexagon to which the directed connector point represents the analytic technique and can be mapped to real-working version of the technique (e.g. R procedure). The concept of binding is presented in chapter 6 in detail. There is also one more additional visual entity that indicates the order of execution of two or more techniques in the modelling space. The main purpose for this feature is for removing the dataflow ambiguity in generating services out of survey technique diagrams.

The minimal working structure for a survey technique diagram serves as a compact presentation of the post data-collection analytic process. Furthermore, survey technique diagrams support visualisation of problem solving knowledge. Consider a survey researcher examining the collected survey data and making inferences about the data. This requires that the researcher applies an appropriate statistical technique on the data. The outcomes of the technique are of two kinds: visual and numerical. For instance visual outcomes include plots and graphical diagrams and numerical outcomes can be coefficient values and descriptive statistics. The output port has an additional layer that can present either form of technique outcome.

**Developing Survey Technique Diagrams: Victimisation Survey**
At the completion of the data collection and verification stage, we will have a statistical dataset with the survey data attributes in Figure 3.10 as data variables. Suppose we are interested in the level of education and crime victimisation in our crime survey example. To investigate the strength of association between two data variables, the use of the chi-squared approximation is chosen as modelled as shown in Figure 3.10.

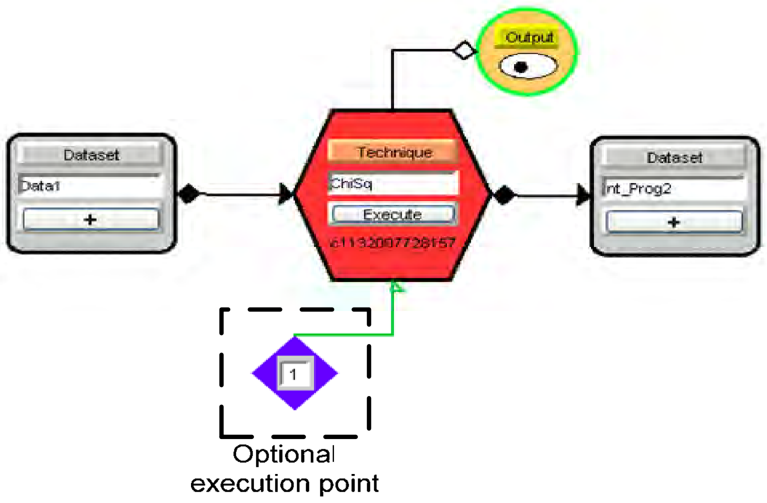


**Figure 3. 10 Survey data diagram which depicts a chi-square test on the victimisation data**

Figure 3.10 shows the data flow which has the sampled subjects as an input to the statistical technique 'ChiSq' and its output that will decide the validity of the association. The ChiSq technique's visual and textual outputs are directed to the output port and the data icon at the end of the dataflow is bound to the resulting data and metadata produced by the chi-square test.

The optional execution point (1-based index) in the dotted square shows that the connected technique 'ChiSq' is the first technique to execute. Execution points can be added to resolve the ambiguity in the order of execution of techniques.

### 3.5.4 Survey Task Diagram

Survey task diagrams can be conceptually less intuitive and difficult to use initially. Brief background on the design of a survey task diagram will be presented first to show our logical progression from modelling requirements. The concept of a survey is well defined and survey prerequisites or criteria are formally defined (Dalenius 1995 ). Thus statistical surveys have distinctive factors that set themselves apart from other research projects and possess ubiquitous components that will be conceptually stable over time. The ubiquitous components such as sampling have contributed to the unique aspects of SDL and SDL's explicit support of the particulars of surveys may enhance usability during a take-up period. However the particulars do not cause a complete chasm between surveys on one side and other knowledge based research. When a survey is abstracted as a set of tasks, it mirrors typical knowledge-based research activities. This generalisation is important at this stage of our research as this implies that to support survey activities we need to address the following:

- Ways to bind knowledge-based activities together
- Nature of tasks
- Mutual understanding (to make users' descriptions constitute reference to the same thoughts)

Our basic approaches to the issues are based on the following assumptions:

- Knowledge based activities can be explicit and tacit.
- Survey activities can be modelled by a number of tasks.
- A single task is not a sufficient integration point for itself.

A task model is a composite of tasks and the model shows how survey tasks exit together in relationships and survey task diagrams represent task models.

**Task Model**
A hierarchical tree consists of tasks, survey artefacts, sub task connectors and task operators (AND, OR, Iteration).

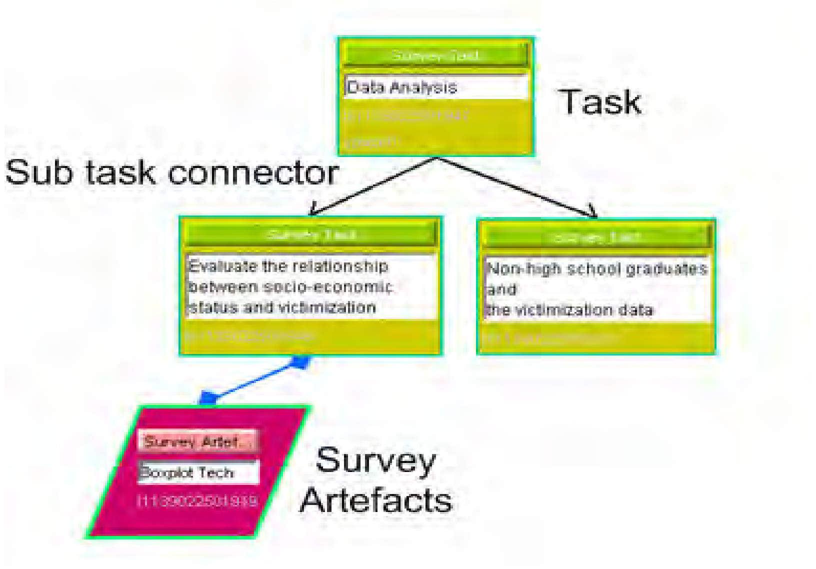

Data Analysis
Task
Sub task connector
Evaluate the relationship between socio-economic status and victimization
Non-high school graduates and the victimization data
Survey Artef.
Boxplot Tech
Survey Artefacts

**Figure 3. 11 A simple task model**

The task model is not a new invention. It can be best understood as a special case of the generalised task model developed by Sutcliffe (2002) as part of his research into patterns for knowledge and software reuse. The task model represents abstractions of solutions to commonly experienced problems and the task model is one of many visual representations of pattern solutions.

The representation of task models in task diagrams takes the form of a collection of rectangles (the survey tasks) directed connectors (the sub task connectors) and parallelograms (the survey artefacts). A survey task diagram may be derived from diagram in a simple manner to the way a class may be derived from a super class. The hierarchical order between tasks is explicitly annotated by the use of the task connectors. The default mode of processing task diagrams is from the left to right by convention. A task is considered fulfilled only if all sub tasks accomplished their goals. That is sibling tasks at the left side precede those at the right side sequentially.

The expressiveness of the task execution order is extended by the use of two types of task operators: OR and Iteration. If the OR operator is present, an upper-level task may be carried out by one of several variations. The Iteration operator signifies that a task is on-going process. The following example task models demonstrate the task operators in action.

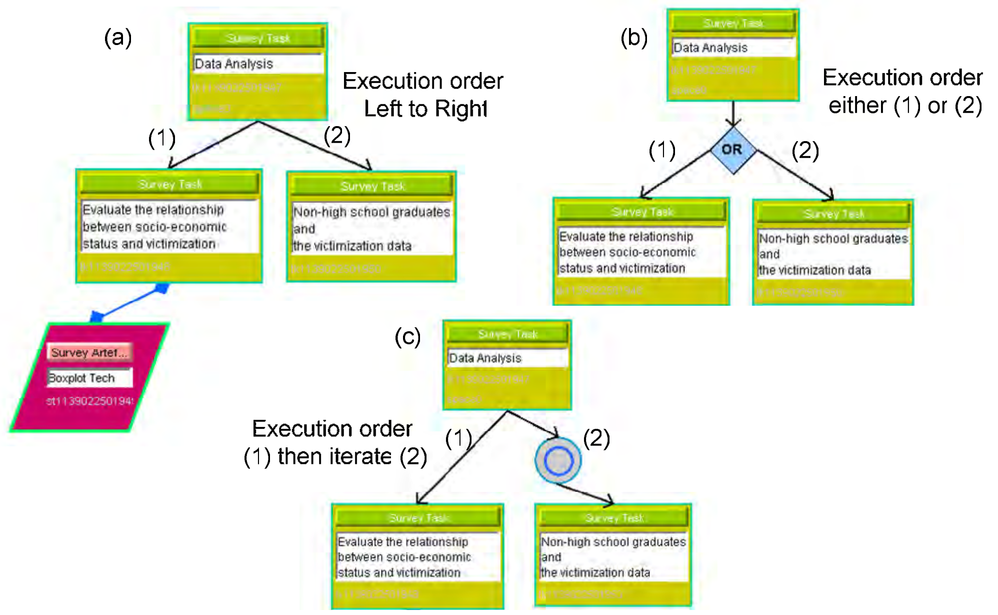

(a)
Survey Task
Data Analysis
Execution order
Left to Right
(1)
(2)
Survey Task
Evaluate the relationship between socio-economic status and victimization
Survey Task
Non-high school graduates and the victimization data
Survey Artef...
Boxplot Tech
(b)
Survey Task
Data Analysis
Execution order
either (1) or (2)
OR
(1)
(2)
Survey Task
Evaluate the relationship between socio-economic status and victimization
Survey Task
Non-high school graduates and the victimization data
(c)
Survey Task
Data Analysis
Execution order
(1) then iterate (2)
(1)
(2)
Survey Task
Evaluate the relationship between socio-economic status and victimization
Survey Task
Non-high school graduates and the victimization data

**Figure 3. 12 Task operators and their role in the execution of a task model**

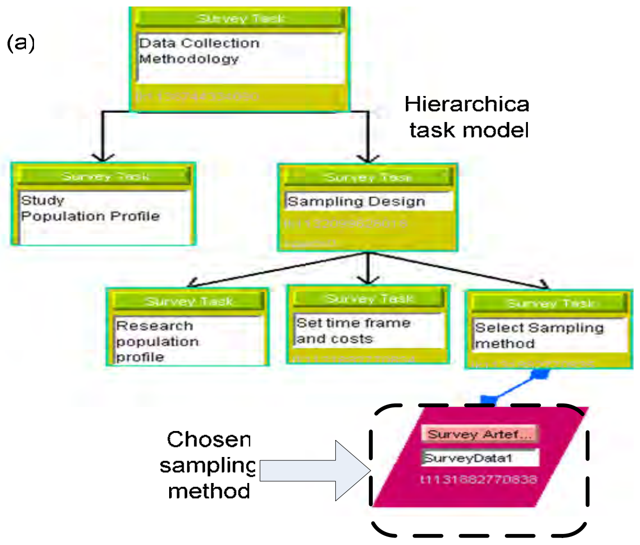
The survey process is broken down into a collection of survey tasks which are modelled in a task model. Each task model may be mapped into one of the survey context of the survey diagram. Implying a particular context of the survey process arises from the interactions between survey tasks. The survey artefact can be simplified as artefacts produced or consumed by each survey task. For instance a reporting task may have documentations as its outputs and a sampling task may require having its operational specification in a survey data diagram.

A survey task diagram does not have detailed specifications of its own. However it is a very important diagram to glue together various aspects of statistical surveys expressed by other three types of SDL diagrams: survey, survey data and survey technique. Survey tasks constitute survey contexts (survey diagrams) and survey artefacts used by survey tasks map to survey datasets and survey techniques modelled by survey data diagrams and survey technique diagrams respectively. Therefore it provides an integration point for visually disparate types of SDL diagrams. The importance of the survey task diagram in integrating SDL diagrams is demonstrated in the next chapter.

## Developing Survey Task Diagrams: Victimisation Survey

In the example of the survey data diagram for the victimisation survey, we presented how the stratified sampling and statistical metadata are modelled using SDL notations. In this section, we examine how discrete survey activities such as sampling design and statistical testing are put into context. Figure 3.13 shows the 'Data Collection' context and by definition the context is supported by one or more survey tasks. Thus building a task model around the survey data diagram not only presents it in relationship with related activities but also explicitly visualises the link between the survey context and the task model. This concept is best illustrated step by step as follows:

**Step 1** The survey data diagram is put into context by adding a survey artefact icon, which represents the survey data diagram, to the task model shown below.



**Step 2** The task model is then mapped to 'Data Collection' context of the survey diagram. This completes the thread of inter-diagram relationship starting from the survey diagram to the survey data diagram.

(b)

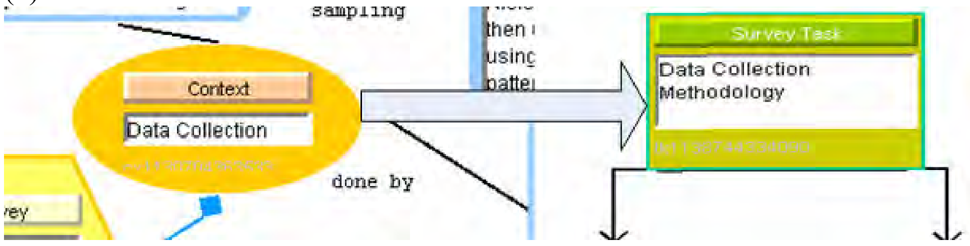


**Figure 3. 13 (a) role of survey artefact (b) survey context to task mapping**

### 3.5.5 Survey Process Diagrams

We have presented four types of SDL diagrams and the focus of this section a survey process diagram can be distinguished from them for capturing the dynamic nature of the survey process. The perspective of a survey process diagram centres on how survey tasks participate in the survey process. Survey task diagrams look at each task as an independent unit but in survey process diagrams all survey tasks exist in relationships as they play their roles throughout the lifecycle of the survey process. A survey process diagram can be composed in two layers. The first layer shows how survey tasks are associated with various stages and explicit process transitions when required. The second layer of a survey diagram brings out inner stage activities between participating tasks. The first layer consists of only two visual icons and three types of connectors. The second layer has three types of visual icons and two types of connectors. Therefore the survey process can be divided into two conceptual levels as follows:

1. Process flow from one stage to the next and stage participants.
2. Inner stage relationship between the participants.

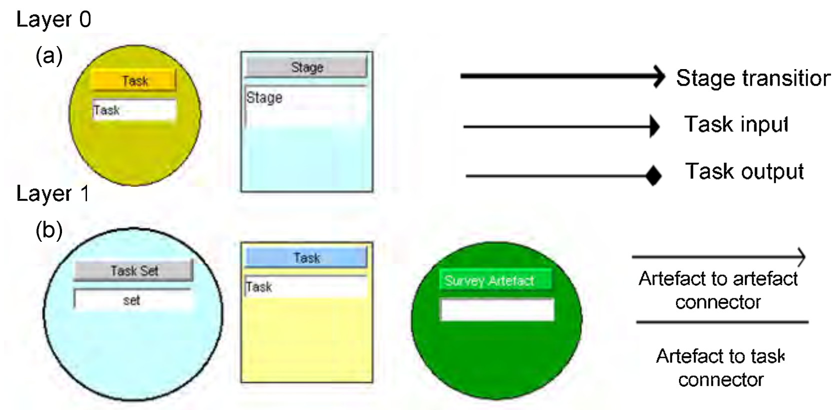


**Figure 3. 14 Diagrammatic notations of survey process diagram**

In Figure 3.14, we have shown the two layers of a survey process diagram which models data collection and subsequent data analysis stages.

**Developing Survey Process Diagrams: Victimisation Survey**

Survey process diagrams can be conceptually simplified as being a platform for visualising how survey tasks cooperate and collectively organise themselves into process stages.
The simple survey process diagram for our example is depicted diagrammatically as shown in Figure 3.15. This survey process diagram describes two stages involved in the survey process: sampling design (stage 0) and data analysis (stage 1). It must be noted that the descriptions for the stages are much simplified and generalised. Our example involves three tasks:
1. Population study: The population profile is studied to formulate data collection strategies which apply to the sampling design decisions.
2. Sampling design: Sampling method is chosen to meet the required precision of the survey and a trade-off between cost and precision.
3. Data analysis: To explore the association between the level of education and the crime victimisation.

The first two tasks are related to each other in the same task model and the two stages are sequentially linked.

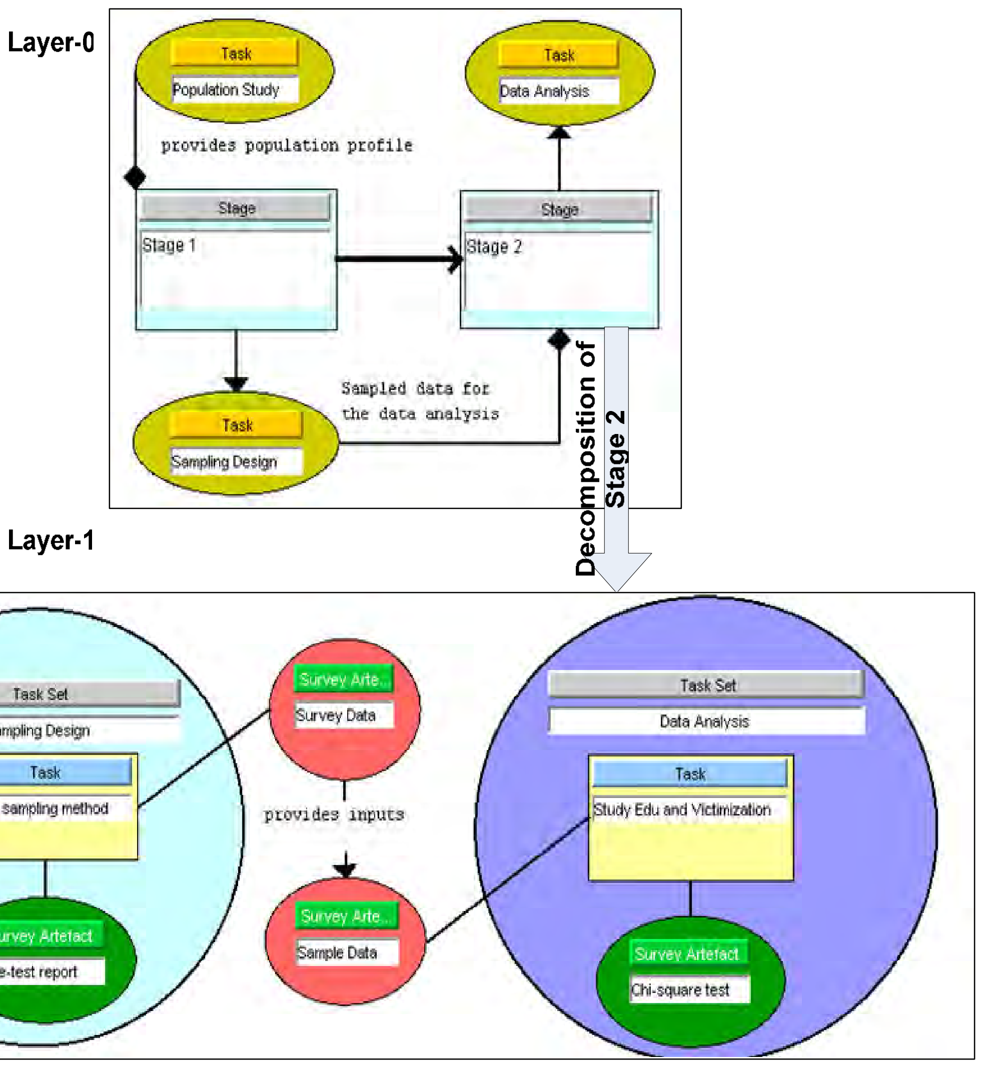


**Figure 3. 15 Survey process diagram for the victimisation survey**

**Layer – 0**
The stage 1 has two tasks associated (population study and sampling design) with it and so does the stage 2 (sampling design and data analysis). The sequential flow between the two stages is expressed by the directed arrow.
A high level description for the layer-0 diagram can be the stage one is built on the interactions between the two tasks and the transition to the stage two is made when the two tasks complete their goals such as providing a population profile report and sample dataset.

**Layer-1**
The layer-1 diagram is the decomposition of the stage 2 therefore we see two tasks and their survey artefacts. Tasks may form relationships through '*consumer and provider relationship*' between two or more survey artefacts. For example in Figure 3.15, the sampling method task has two survey artefacts associated with it and one of the artefacts (Survey Data), which is created from the selected sampling method,

becomes the input for the data analysis task looking at the relationship between education and victimisation data using the chi-square test. When a survey entity is shared between two or more tasks, it is drawn outside the circle and coloured in crimson red.

The survey process diagram accentuates the importance of breaking the survey process in terms of well-defined task models as they are the basic vocabulary of the survey process model. This point is further investigated in the section on the inter-diagram relationship of Chapter 4.

## 3.6 Summary

This chapter presented a suite of visual languages for statistical surveys (SDL). We first identified our primary design influences behind the SDL from four areas: multi-view modelling approach of UML, understanding of diagram as cognitive tool (Blackwell 1998), needs for multiple modelling spaces (Unhelkar 2005) and advantages of homomorphic representation scheme (Barwise and Etchemendy 1995).

Working examples for SDL diagrams were provided to understand them in the context of the survey process. We also briefly discussed the issues of inter-diagram relationships that will be elaborated in the next chapter.

A summary of syntactic rules of SDL diagrams can be found in Appendix A.

# Chapter 4 SDL Visual Model and Metamodel

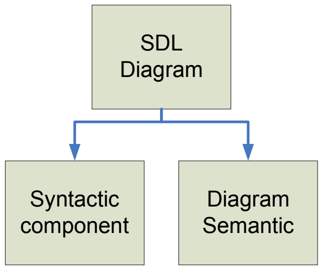
## 4.1 Introduction

In this chapter, we discuss the key concepts that evolve around the diagrammatic visual model, which is visualised by the use of graphical notations and implemented within the Pounamu environment, and the strategy in communicating a heterogeneous visual model to both software tools and humans in an integrated manner. The exhaustive formal treatment of the visual model and the metamodel will be avoided but we will present more practical topics that clearly impacted core design issues in building user supporting tools around SDL. The topics will be presented in an envisaged usage pattern of SDL diagrams and tools. The pattern starts with individual diagrams then moves on to the integration of multiple diagrams.

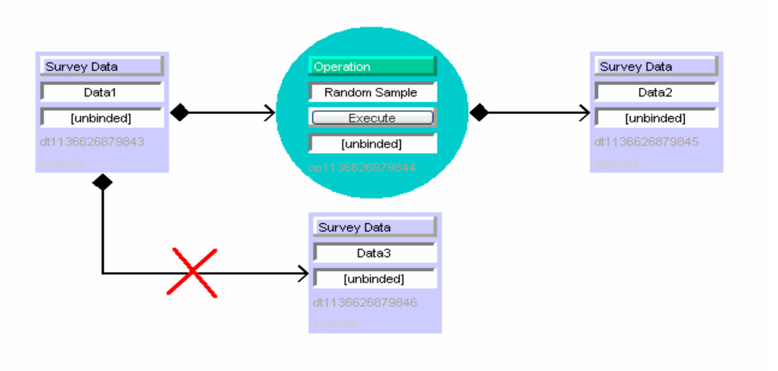
## 4.2 Visual Diagram

SDL has a suite of visual diagrams to express statistical surveys. An SDL diagram is an interface which users interact to construct models for statistical surveys. Visual elements of the diagrams are transcribed into a visual model following syntactic rules for a diagram. Therefore an SDL diagram has two definable components for involving user interactions, the syntactic component of the diagram which dictates how graphical elements should be put together as in Figure 4.1 (b) and the diagram semantic which transfer graphical information to precise meaning in a target domain as in Figure 4.1 (c).

(a)



(b)



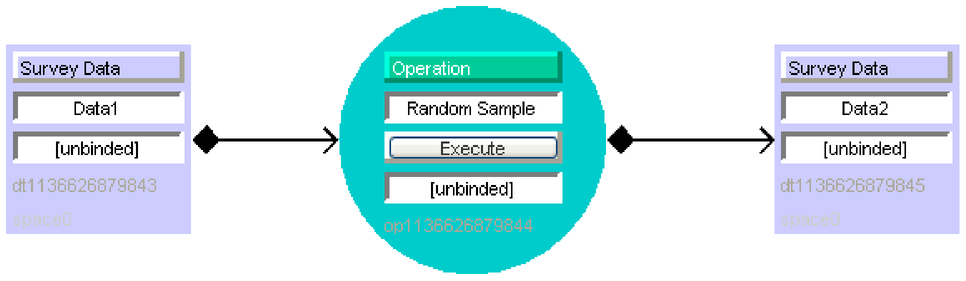


**Figure 4. 1**
**(a) Two components of An SDL diagram**
**(b) Example of illegal syntax for the diagram: Dataset must be an input to the operation only.**
**(c) Example diagram showing the semantic component depicted by the visual components. Random (operation) sample operation with one input (Data1) and one output (Data2)**

When an SDL diagram is used as a tool, it brings in two more dimensions into play in its user interactions. Imagine a user who is modelling the survey process using SDL-based diagrammatic techniques. The task entails such steps as

1. Target domain (statistical survey) to visual representation
2. Diagram refinement (requires a layered approach)
3. Diagram utilisation (requires diagram integration)

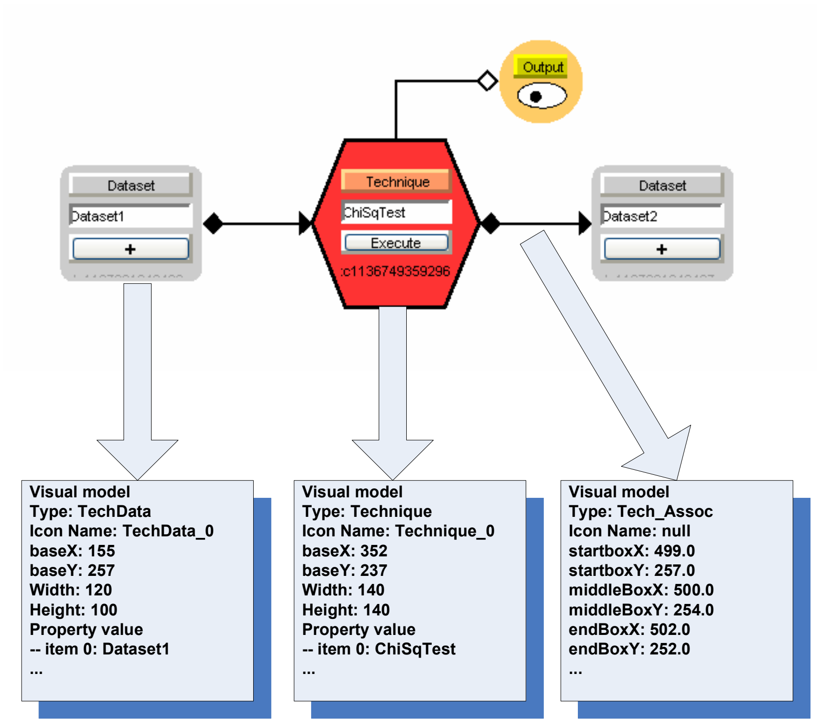
The first step builds a visual model using the above mentioned components of SDL diagrams. The first step does not lead to an executable SDL diagram but rather a purely visual representation. To use SDL diagrams as a tool, relevant attributes of graphical entities; the details behind the metadata must be added thus acquiring enough low-level information to yield a thorough executable model. The fine-grained details of SDL diagrams are captured using a layered approach.

The layered approach (Melnik and Decker 2000) inspired the diagram refinement process. The main purpose of the layered approach is to maintain a high-level of abstraction (visual scan friendly). While doing so it facilitates back-end integration into heterogeneous external services such as a computation engine and inter-diagram integration. We now examine each layer from the user's viewpoint.

### 4.2.1 First layer: Visual model - What you see on the surface

The first and outermost layer consists of SDL graphical notations. This layer acts both as a UI and a communication medium for the survey process built according to syntactic and semantic rules. The first layer information contains all the information needed to reconstitute visual representations as shown in Figure 4.2 so the first layer conserves such properties as shape coordinates, shape type, connector type and shape related attributes.

When SDL diagrams are used as a tool the first layer may reflect changes in lower layers through appropriate visual changes such as change in layer thickness. Thus it offers a two way representation of the survey process at the high level and the reflections of the changes at lower layers.



Visual model
Type: TechData
Icon Name: TechData_0
baseX: 155
baseY: 257
Width: 120
Height: 100
Property value
-- item 0: Dataset1
...

Visual model
Type: Technique
Icon Name: Technique_0
baseX: 352
baseY: 237
Width: 140
Height: 140
Property value
-- item 0: ChiSqTest
...

Visual model
Type: Tech_Assoc
Icon Name: null
startboxX: 499.0
startboxY: 257.0
middleBoxX: 500.0
middleBoxY: 254.0
endBoxX: 502.0
endBoxY: 252.0
...

**Figure 4. 2 First Layer – Visual model (Partial)**

**Second layer: Metamodel**

An SDL metamodel is the result of a process of abstraction, classification, and generalisation (*ACG*) on the visual model so it carries fundamentally the same definition as in Model-Driven Architecture (Mellor et al 2004 Ch2). While SDL visual models are concerned with visually oriented information, SDL metamodels are designed to communicate the visual model in terms of operationally important elements and structures (e.g. data flows). The metamodel can be defined as tool's view on its underlying metadata (Tannenbaum 2001). Metamodels can be created from visual models according to pre-built schemas. Metamodels are the products of the model reification process, and the process deals not just with physical files that persists visual models but also real-time user interactions that may not be persisted. Binding of graphical elements (e.g. hexagon shape representing a statistical technique), which are one only abstract entities, to real-world resources (e.g. Web service, R computation procedure, etc) is one such case. Example of the process of abstraction, classification, and generalisation. Figure 4.4 show the structural overview of the metamodel layer. Each visual icon at the visual layer can have the structural model. In this sense the metamodel layer is the collection of metamodel constructs that are instances of the structural model. Visual icons at the visual layer are mapped into metamodel entities which are derived from metamodel entity types. Each diagram is also derived from one of five diagram types and has collection of metamodel entities. A metamodel entity may have one or more relationships and attributes. In Figure 4.3 the metamodel entity '*ChiSqTest*' has a relationship which represents the dataflow from Dataset1 and to Dataset2. The relationship is derived from the Dataflow Metamodel relationship type.

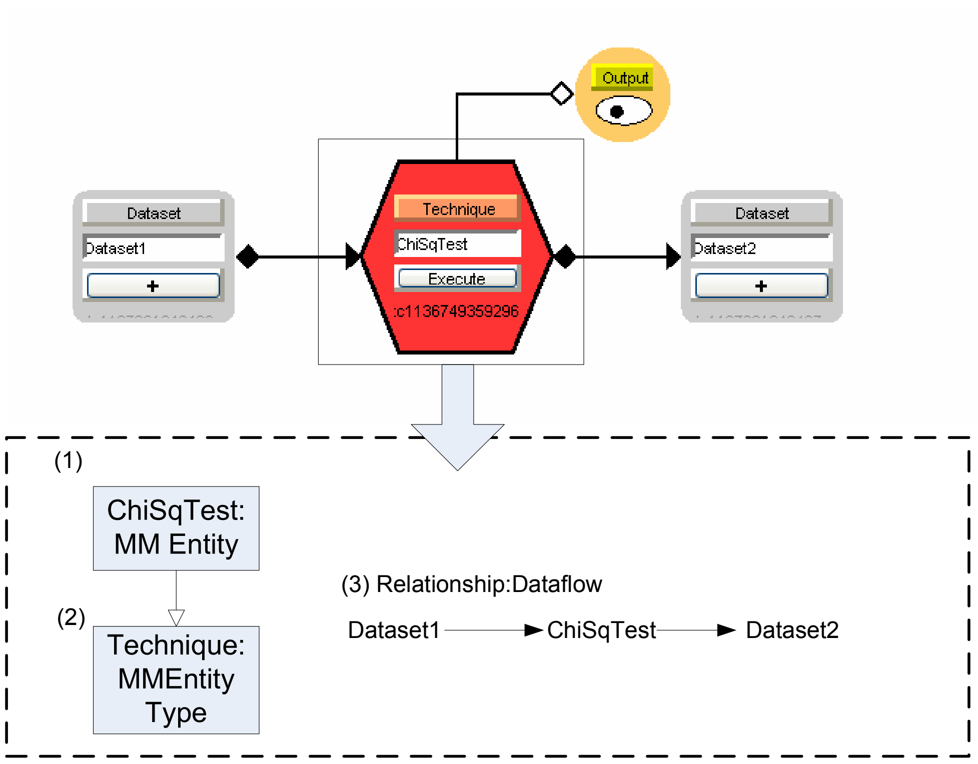


**Figure 4. 3 (1) abstraction of the technique icon (2) classification of the metamodel entity (3) metamodel relationship from the visual diagram**

The metamodel layer can be built by processing each non-connector type icon and the outcomes of the process are persisted via XML files that are bound to the icons. The evaluation of the top diagram in Figure 4.3 includes the following:

Abstraction of the visual icon into a metamodel entity.
Classification of the metamodel entity into a specific type: Technique
Generalisation of the structural components into a dataflow.

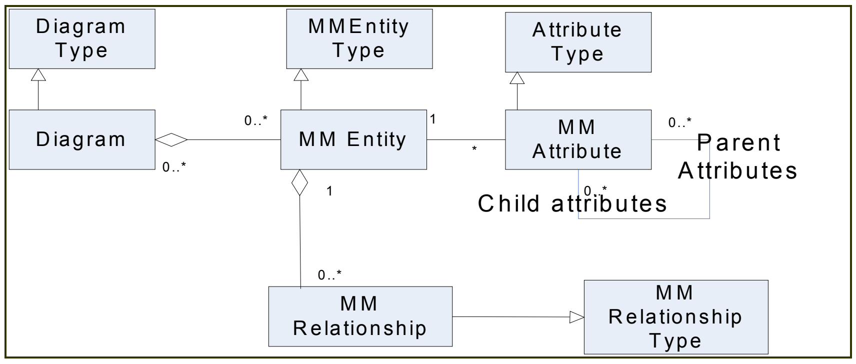


**Figure 4. 4 Structural overview of the metamodel layer**

## Third layer: Semantic Layer

Semantics concerns the study of meaning. The study and application of semantics is a far-reaching philosophical problem with innately enormous influences on other branches of sciences. Thus cautious warnings precede this section. Most of discussions and examples presented in this section should be explored within the domain of the statistical survey process only in the limited sense as we lay down no formal framework to build our case for general purposes and our proof of concept tool does not fully utilise the SDL semantic layer but relies on ad-hoc strategies to mine semantics out of the metamodel layer in many instances.

Our concept of semantics comes from realist semantics (Gärdenfors 2000 Ch 5). A semantics is defined as a mapping from the linguistic expression onto a world (Gärdenfors 2000 Ch 5) which can be universal or situation dependent. Visual notations at both individual and collective levels are assumed to be mapped onto a real world in the domain of a statistical survey.

The necessity of the semantic layer comes with SDL tool support. SDL diagrams as communication mediums from the perspective of human users requires no explicit semantic support as the domain specificity and highly specialised nature of SDL lack the semantic diversity of more general purpose environment such as the Web. Our predicament is that SDL diagrammatic notations anticipated the development of supporting tools and model-based approach in generating services. Our current proof-of-concept tools do not fully utilise the semantic layer but rely on static rule based reasoning to infer model semantics from the metamodel layer. However the semantic layer will be the primary tool in extending the language base of SDL thus it has been included as a part of our main body of discussion.

The semantic web (Davies and Fensel and Harnelen 2003, Melnik and Decker 2000 ) illustrates how layered technologies can work together to provide machine-aware semantics for disparate web resources. The W3C's semantic web's scope, foundational methodologies and technologies place exorbitant development overheads thus the SDL semantic layer is not the duplicated version of the semantic web but a limited conceptual subset. We leave the opportunity to incorporate the full semantic web into SDL as a future work item.

The SDL semantic layer is primarily organised by *topic maps* (Pepper 2002). Topic maps offer heterogeneous information repositories (Biezunski 1999) to tie the underlying semantic of the metamodel to the real world statistical survey topics. As we have presented more in-depth discussion of topic maps in chapter 2, our discussion here will focus on the functional goal and the practical usage of topic maps in the semantic layer.

### Creating Topic Maps for SDL

The SDL semantic layer consists of two special categories. The primary constructs in SDL diagrams such as dataset entities are organised into topics and they are explicitly related to the overall conceptual structure of SDL by means of association and instance membership. The ontology layer consists of taxonomies on statistical techniques, metamodel structures and relational templates to bind the metamodel to the semantic layer.

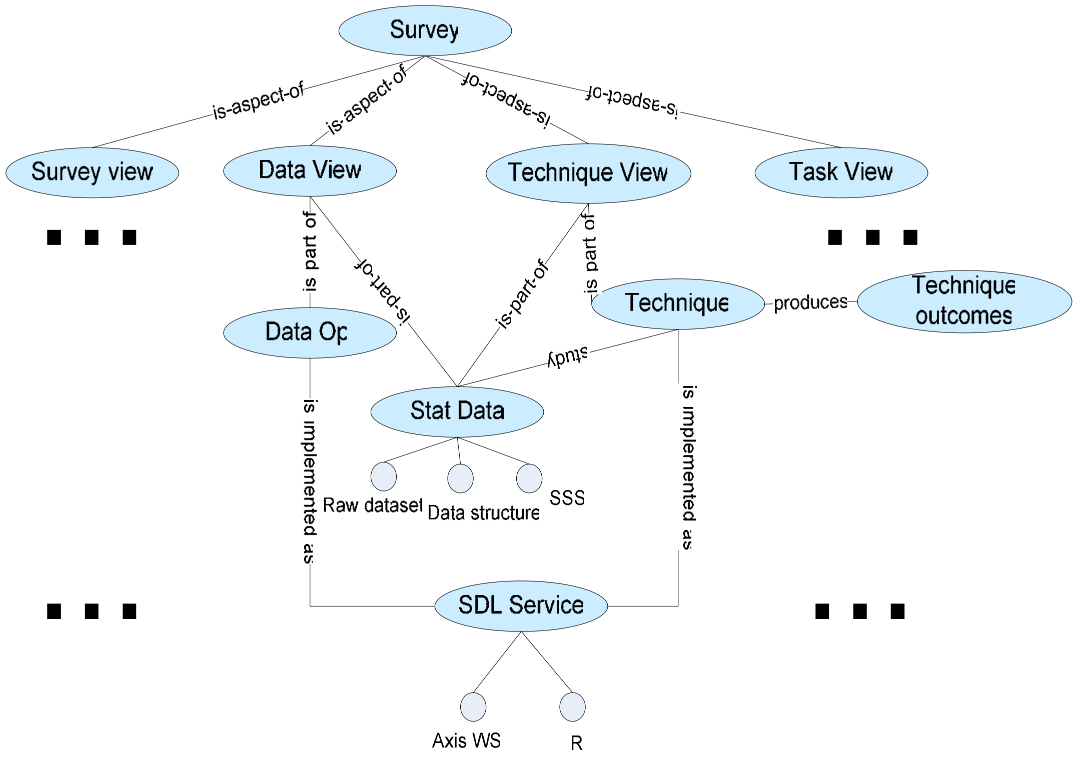


**Figure 4. 5 SDL in topic map representation (partial)**

The original intent of Topic maps was to merge multiple literature lists, glossaries, thesauri and tables of contents (Park and Hunting 2002 Ch2). And it is not surprising that they have been found to be useful in organising complex knowledge bases in significant real-world projects (Passin 2004 p87). If all important aspects and diagrammatic notations of SDL are considered to be "topics", SDL diagrams are "occurrences" and all inter-diagram relationships are "associations" then we can visualise topic maps as providing a unique platform to express the semantic layer. Also it is interesting to note that just as we can transverse from the visual layer to the semantic layer, reversing the process by having the visualisation of the topic map as a starting point can be a step forward to making SDL an extremely extensible visual language. When topics such as statistical data, which has physical resources that are referencing the topics, should have an occurrence the binding process should be supported by SDL tool support. The association of the topics to resources is one of key aspects of our research on tool support and chapter 7 deals with tool support issues related to make the association possible.

The semantic mappings that link metamodels and inference schemas are applied to the metamodel to draw out what the metamodel signifies in a target domain. We envisage the practical applications of model semantics can be mostly delegated out to external third party tools. An SDL visual model is in actuality is a Pounamu model file may have extra dimensions that would be cumbersome to accommodate for other software tools. By contrast SDL metamodels do not have visual elements that are designed to be rendered visually.  Hence SDL metamodels are much more accessible to a wide range of tools across heterogeneous platforms. We draw model semantics out of metamodels which have no visualisation overheads therefore making much room for the semantics inference to be performed and utilized by external tools.

**Functional Aspect**

The SDL metamodel layer provides the views for tools and the SDL semantic layer provides a way of representing the abstraction of the SDL-based survey modelling. Functionally the metamodel layer prepares SDL diagrams for tool support and the semantic layer provides a means to make sense of the information represented by metamodel entities and relationships. The basic premises of the layered approach are materialised in three stages of mapping. The first two stages are mentioned in the section on the metamodel layer and illustrated in Figure 4.3. The outcomes of the two stages still lack the whole spectrum of information to relate visual artefacts to user's domain specific knowledge.

The utilisation of the topic map brings the mapping template to representing and correlating the visual constructs, which are captured in the metamodel layer, to aggregated knowledge in the form of a topic map. The following example which extends Figure 4.6 shows the functional implication of the mapping. The semantic model of SDL binds the metamodel components that have been structurally processed from the visual layer into the various topics or associations of the semantic network.

The technique entity (ChiSqTest) at the metamodel layer maps into the technique node which has the following associations:

- Technique studies statistical data.
- Technique is a part of the technique view node which is one of many aspects of the survey.
- Technique can a member of a dataflow and is implemented as An SDL service. Physical occurrences of the SDL service can be R or Axis hosted services.

Likewise dataflow relationships of the technique entity are semantically linked to the dataflow node of the topic map.

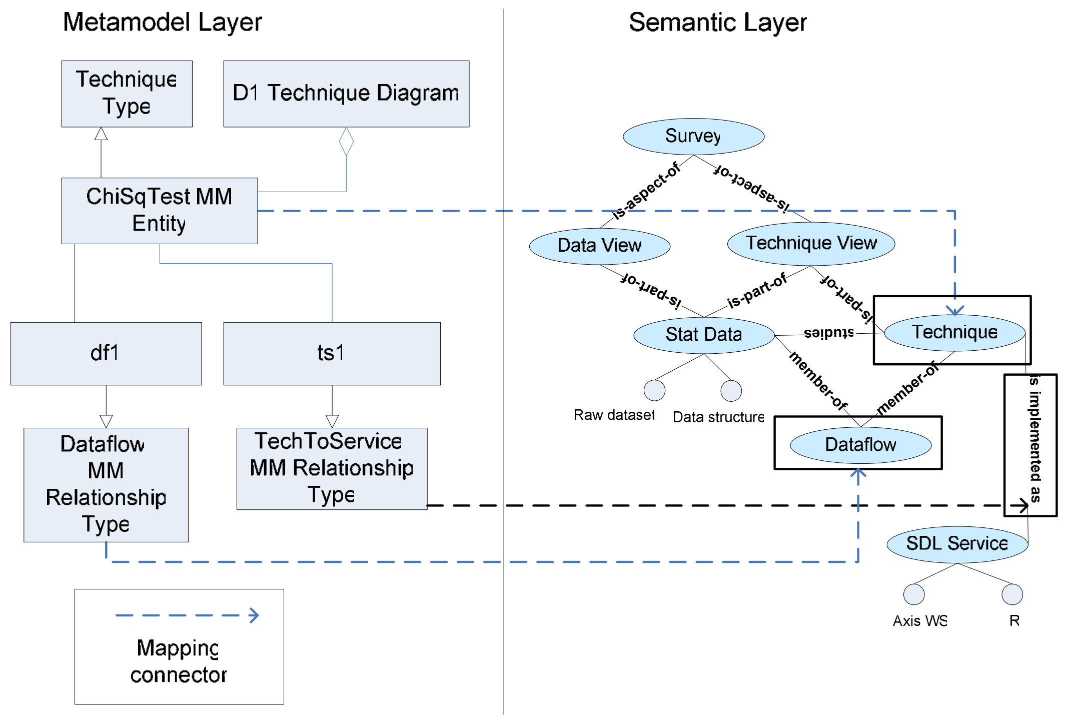


**Figure 4. 6 Metamodel to the semantic layer mappings**

The semantic layer creates a structural ontology for SDL. Hence the structural ontology also provides a template for which mapping operations to bind external resources (occurrences) with SDL metamodel entities. One such mapping operation is to turn static visual icons into dynamic ones, that is to give the icons the behavioural and functional nature of a widget, and which can then serve as a dynamic interface to control external resources. The icon-to-widget mappings for SDL tool support arise out of the needs to support occurrences associated with a topic node at the tool level. Thus the semantic layer also provides a new perspective in looking at visual tool support from the modelled ontology

## *4.3 Survey entities – What binds diagrams together for heterogeneous reasoning.*

Individual visual models serve the task of communicating a specific aspect of the survey diagram. This specialised nature of each diagram means that when we want to design and reason about the entire survey process, we need to consult all five types of diagrams together. This example of heterogeneous reasoning downplays the intrinsic complexities involved in the reasoning process due to the power of human cognition in analysing vastly complex visual information to form a unified perspective. However for this scheme to be supported by software tools we need to address the major reason hampering cross-model communication – discontinuity of information across different model (Medvidovic et al 2001). We introduced the concept of survey entity to largely negate the need for more technically and conceptually sophisticated approaches such as ‘model connectors’ (Medvidovic et al 2001). This does not mean that our approach can be mirrored in a straightforward manner in situations where more thorough model integration strategies are called for. Though the multifaceted modelling approach of SDL resembles many visual modelling languages, inter-diagrammatic relationships between SDL diagrams can be vastly different from others. To clarify this point, let us discuss the inter-diagram relationships between SDL diagrams before we delve into a discussion of survey entities.

## Inter-diagram relationships

Without a precise semantics a standard modelling notation can devolve to a Tower of Babel. In practice UML diagrams are often casually created with imprecise semantics in terms of key basic concepts (France 1999). And the imprecise semantics can introduce inconsistencies that make difficult to integrate a set of independent views in a complete fashion. SDL diagrams adhere to the strict relational rules between diagrams to prevent the imprecise inference ambiguity of diagrams without extensive refinements of diagrams. The enforcement of the strict inter-diagram relationships is implemented by setting out explicit rules for valid inter-diagram relationships, expressed in terms of directions and cardinalities of relationships. Inter-diagram relationships bind the multiple SDL diagrams together – diagram to diagram. The cardinality of a relationship is the number of SDL diagrams that can be associated with each type of SDL diagram and the relationship direction defines the flow of communication from one diagram to another. That is if two diagrams have to consult each other equally than the relationship between them is two-way. However when one diagram uses another diagram more as an external resource (Task diagram and survey data diagram), the direction of the relationship is one way.
The cardinality rules governing the inter-diagram relationships are as follows:

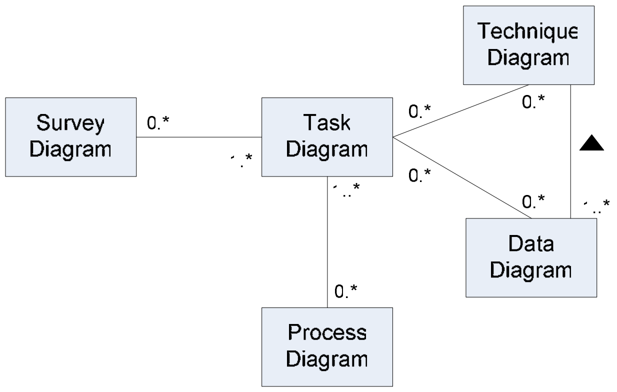


**Figure 4. 7 Cardinality rules and inter-diagram relationships.**

| *Diagram* | *Relationships* | *Detail* |
|---|---|---|
| Survey Diagram | Task Diagram | A survey context consists of survey tasks |
| Task Diagram | Process Diagram<br>Technique Diagram<br>Data Diagram | A survey task may utilise technique and data diagrams.<br>Fundamental participants of the survey process are survey tasks. |
| Technique Diagram | Task Diagram | Survey task's functional goals can be modelled in technique diagrams. |
| Data Diagram | Task Diagram | Survey task's functional goals can be modelled in data diagrams. |
| Process Diagram | Task Diagram | Tasks exist in relationship in the survey process. |

**Table 4. 1 Inter-diagram relationships**

The approach improves the efficiency and simplicity by offering highly deterministic integration of all diagrams. However the gain in efficiency comes with a loss in the flexibility in a wider context. However this should not be counted as a major shortcoming as SDL is designed to be domain specific and the proposed rules just reflect the agreed diagram evaluation order within the context of the survey process. The evaluation order represents the operational semantic in comprehending multiple diagrams in a common target domain. We may start with a survey diagram then the contexts of the survey diagrams are broken down into individual survey tasks. The individual tasks may employ statistical datasets and techniques and the tasks are the actors within the survey orchestration. In UML projects, there can be many types of UML diagrams for a given modelling space therefore it makes the construction of a deterministic evaluation order to be impractical for general purposes.

## Survey entities

Our approach in integrating all metamodels behind five types of diagrams use two main constructs: inter-diagram relationship and survey entities. Inter-diagram relationships lay down a broad inter-diagram network and survey entities belong to the network of the diagrams.

A survey entity can be defined as follows:

- A survey entity can be a form of a survey task, dataset, technique or any non-connector type graphical entities in the visual environment which comes into existence in one or more aspects of statistical surveys.
- A survey entity has its own unique identity though they may appear non-distinguishable visually. E.g. two dataset icons look the same but they are mapped to two unique survey entities.
- A survey entity belongs to at least one visual model.
- Inter-diagram relationships positively imply the existence of at least one survey entity which is shared by more than one diagram. In other words, a shared survey entity completes an inter-diagram relationship.

Model integration is made possible through the use of overlapping survey entities amongst heterogeneous visual models. Therefore the overlapping survey entities act as integration points to merge related views for a target domain (c.f. A set of UML diagrams that describe a problem are semantically tied to a target domain but they may/may not have explicit unions between them).

Users or tools can transverse the network of related diagrams by using overlapping survey entities as entry/exit points. The utilisation of survey entities in the integration process can also be visualised by Venn diagrams as shown in Figure 4.8.

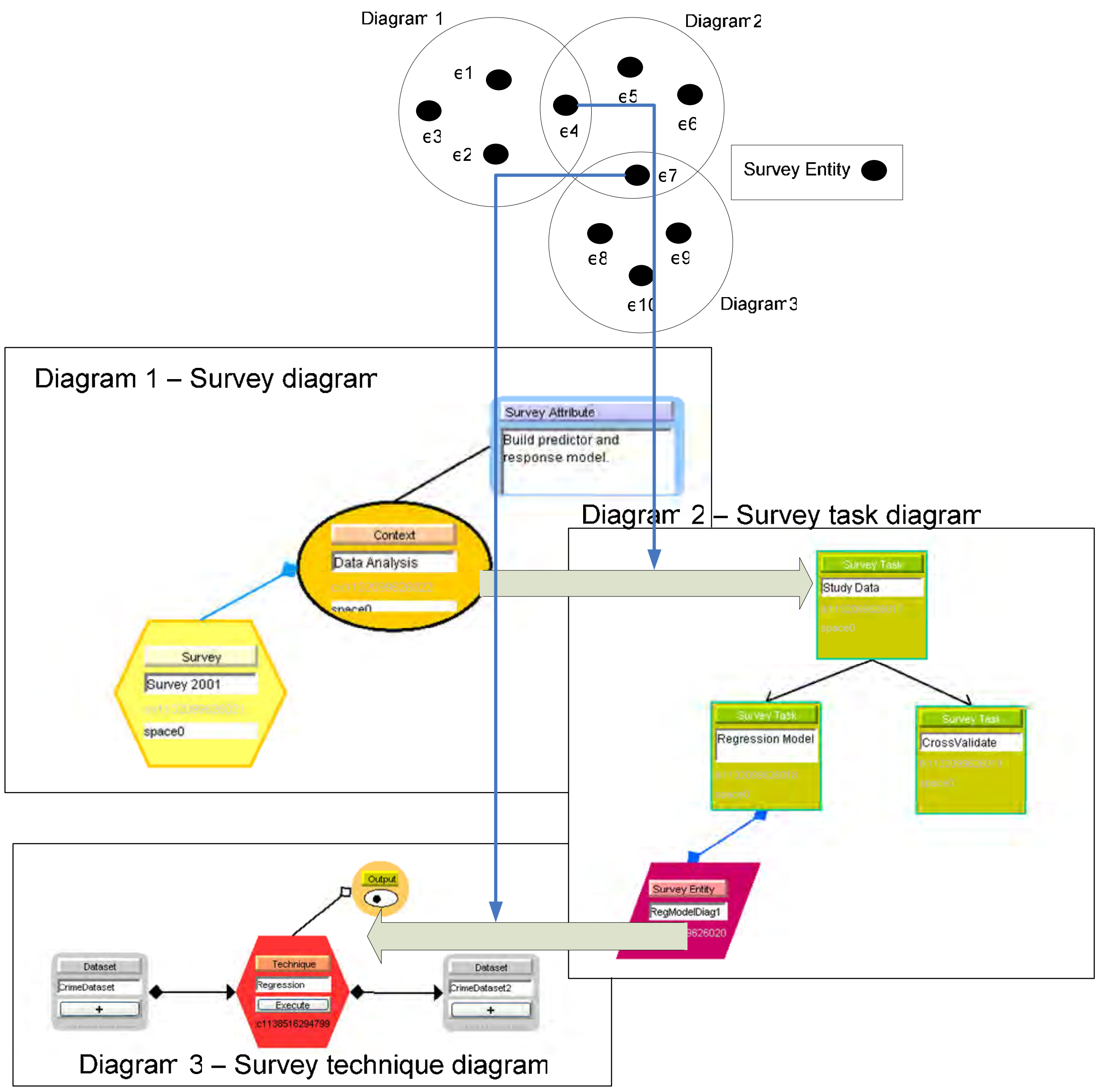


**Figure 4. 8 Survey entities and inter-diagram relationships**

Each circle represents a visual model. We have 10 survey entities in total and see that the two survey entities, which reside within the Venn diagram's overlapped regions form, two inter-diagram relationships. The inter-diagram relationships can be mapped to the rectangle section of the semantic layer as follows:

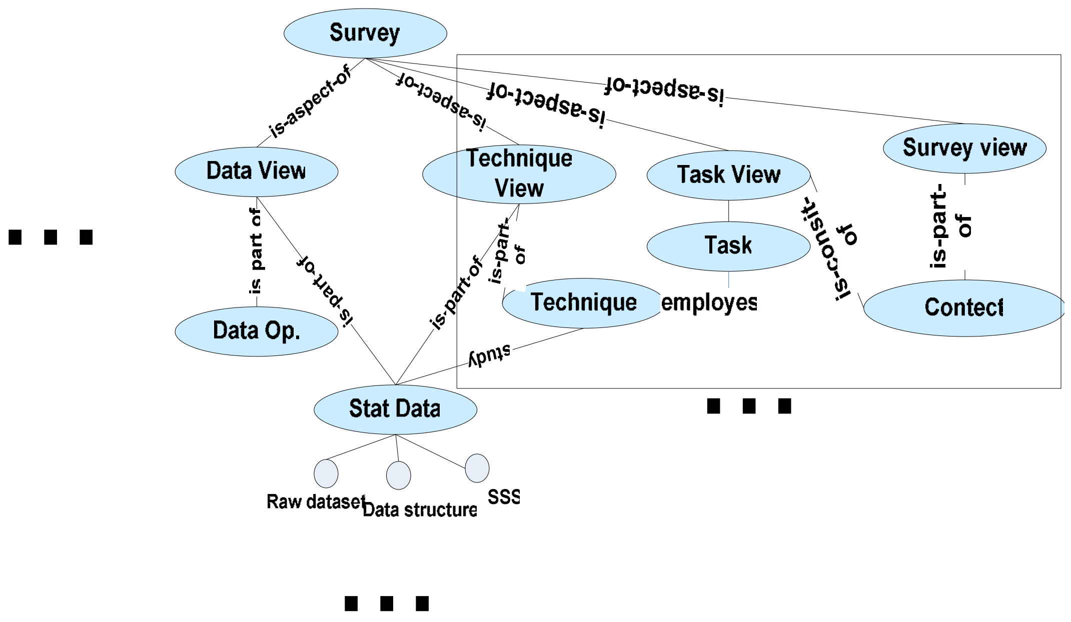


**Figure 4. 9 Mapping of the inter-diagram relationship (Figure 4.8) to the semantic layer (rectangular region)**

## *4.4 From Visual Description to Concrete Program*

This section is intimately related to the next two chapters on the development of our proof-of-concept tool for SDL diagrammatic notations. In a nutshell, our approach in building the supporting environment for SDL from an end-user's perspective can be described as turning the static diagrammatic representations into executable and interactive tool interfaces. For example, Petri Nets have been used to build various discrete distributed models for various problems that range from a workflow management to data analysis. There are instances where a live simulation of a Petri Net can be of great of value with respect to both end-users and developers especially when Petri Net models can be closely mapped to real world implementations. SDL diagrams are not only able to create conceptual models of statistical surveys, it is also possible that the modelled aspects can be given the addition of detail to make them a basis for executable visual tools. In this section, we discuss how the activity of specification can be done at the metamodel layer to accomplish the transition. Although we deal specifically with statistical surveys, the ideas can be applicable in general, to the design of any visual language. Our discussion here is largely dealing with conceptual issues. Chapter 6 and 7 offers more information on the actual implementations of executable SDL diagrams.

A more useful executable model involves the integration of visual diagrams into back-end software systems and resources that may or may not be represented symbolically by visual icons. The integration/binding process requires precise specifications so that abstract visual entities have cohesive integration points with concrete programs behind the scene. The integration/binding can be generalised as the process which creates metamodels that are bound to visual entities to bring precision. An SDL diagram is broken into a group of well defined metamodels and this requires interaction with user to specify necessary mappings as illustrated in the next section. These operations are involved in the transition to the executable program domain and are illustrated at a more detailed level from the tool's view in Chapter 6.

**Processing of Visual Diagrams**

This operation can be viewed as the process described as the ACG (abstraction, classification and generalisation). Syntactically valid visual models are starting points for their corresponding metamodels. Depending on the structural significance of visual entities, their metamodel elements at the metamodel layer are given both relational (e.g. dataflow) and entity-specific atomic attributes (e.g. unique element ID).

**Adding Details**

Necessary details must be added to the metamodels to make diagrams executable. The addition of details is made by users and the additional information aims to explicitly link the metamodel entities of the diagram with the real world implementation. The process can done be in stages using user-driven mapping mechanisms and can be regarded as the refinement of the abstract model.

**Example**

We can illustrate the two operations by using an informal diagram model.
A visual icon in a survey data diagram is required to represent some statistical data. The visual icon needs to be mapped to the statistical data and its relationship to the sample population and questionnaire metadata in Triple-S.

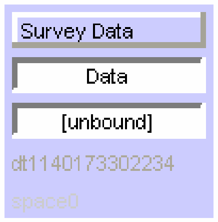
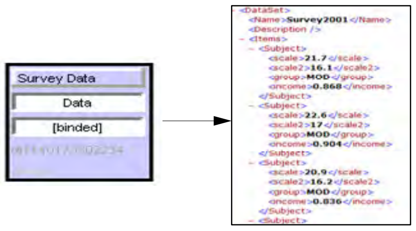


**Figure 4. 10 Unbound data icon**

Firstly, we identify the visual icon's type which is a data type. Then a metamodel for the visual icon is created with initial attributes that are driven from its visual attributes such as name (Data) and unique ID (dt1140173302234). This initial metamodel is the primitive specification from which the specification for the executable model is built. As further processing of the survey data diagram reveals no other relationships, the metamodel building ceases until more detail is added via other user interactions.

Even though we do not deal with the explicit tool support for the diagrammatic notation at this stage, the user inputs for the extended metamodel detail are assumed to be obtained via the mapping interfaces of our prototype tool:

1. Link to a statistical dataset
   A binary relationship is constructed to point the metamodel element, which corresponds to the visual icon at the visual layer, to a specific physical statistical dataset.



```
- <DataSet>
    <Name>Survey2001</Name>
    <Description />
  - <Items>
    - <Subject>
        <scale>21.7</scale>
        <scale2>16.1</scale2>
        <group>MOD</group>
        <income>0.868</income>
      </Subject>
    - <Subject>
        <scale>22.6</scale>
        <scale2>17</scale2>
        <group>MOD</group>
        <income>0.904</income>
      </Subject>
    - <Subject>
        <scale>20.9</scale>
        <scale2>16.2</scale2>
        <group>MOD</group>
        <income>0.836</income>
      </Subject>
    - <Subject>
```

**Figure 4. 11 Binded data icons and the dataset that has been mapped at the visual layer**

2. Link to the sample to population relationship
   The metadata element now exists in relationship with associated data variables and the metamodel element which is the sample space where the samples are taken.

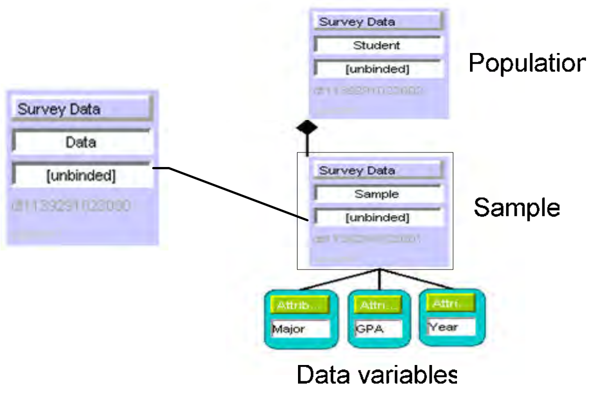


**Figure 4. 12 Data icon is mapped to the sample to population relationship**

With the newly gained additional information at the metamodel layer, it is possible to describe the visual icon in the domain of concrete specification. The abstract visual representation of the metamodel elements which has only limited uses as the type item is now defined by relationships to the statistical resource and other metamodel elements give a meaning to the specification. Starting with the simple model, the refinement of the metamodel element yields more complete model descriptions that can also be utilised by tools. The following structural model show how the additional information changed the initial model.

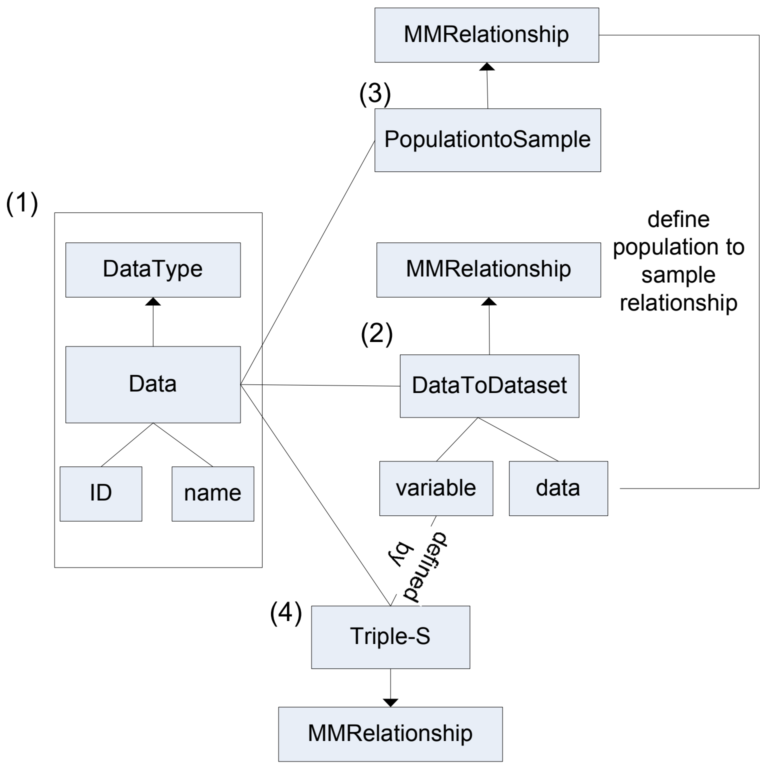


**Figure 4. 13 (1) Data icon's metamodel entity at the metamodel layer (2)Data-to-Dataset relationship linked to the metamodel enitity (3) population-to-sample relationship linked to the metamodel entity (4) Metamodel relationship to Triple-S**

## *4.5 Putting everything together – Model usage and Integration*

We now discuss survey designer's perspective on the visual model, the metamodel, and the model integration for the survey process in the context of real-life operations to clarify the discussed concepts transparently in action.

So far the description of the visual model and the metamodel has been topical. Let us investigate how the topical features are put together to form an integrated model step by step. It should be noted that some part of the discussion will be expanded further at chapter 7 when the nature of the discussion is better be presented along with the proof-of-concept tool.

**Example 1: Constructing an inter-diagram relationship**

Consider the example diagrams shown here:

1. Survey diagram with one survey context data collection.
2. Survey task diagram which represent a data collection process pattern.

Two diagrams are associated as the survey context utilises the data collection process which is visualised by the survey task diagram. Therefore the following inter-diagram relationship exists between the two diagrams as structurally shown in Figure 4.14(c).

Type: Survey to task
Direction: One-way

The relationship can be generalised to be expressed as follows:

The data collection task entity does not exist in the survey diagram at the visual model level. However further exploration at the metamodel reveals the presence of the overlapping survey entity which derived out of the data collection task thus forming the network between two diagrams at the metamodel level.

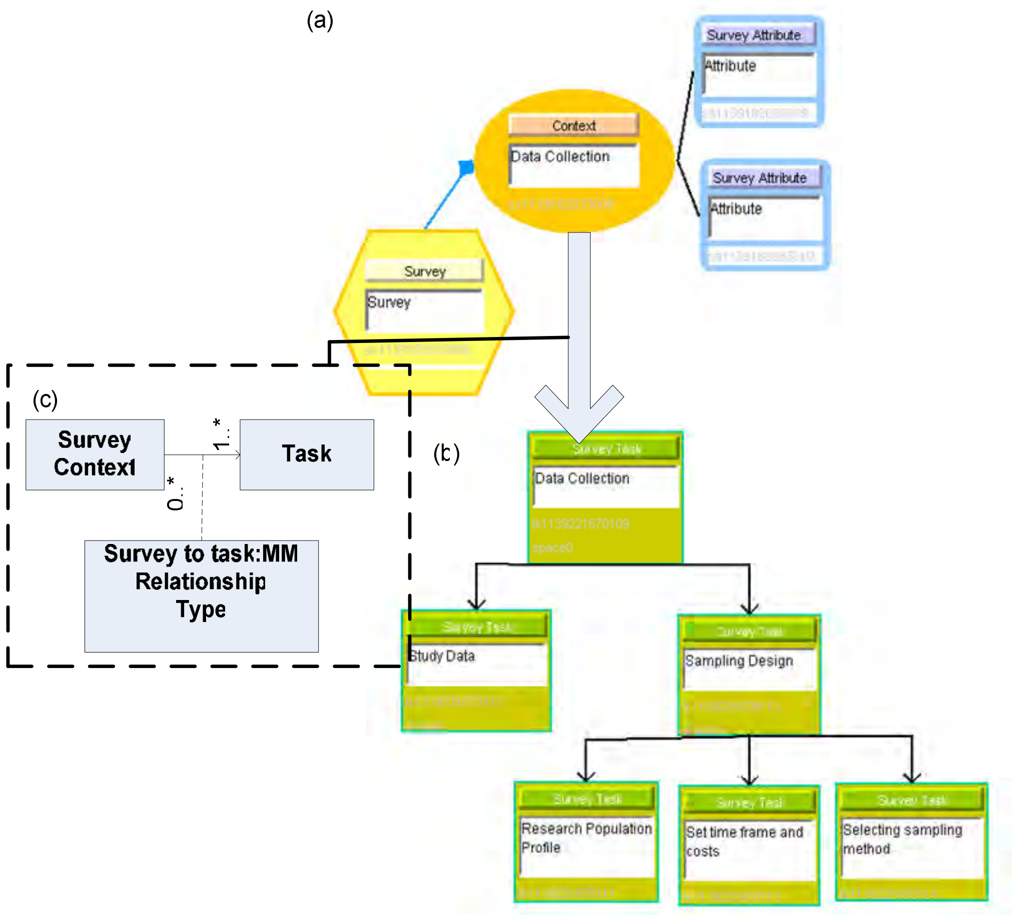


**Figure 4. 14 Interdiagram relationship between survey diagram (a) and survey task (b) diagram (c) structural representation of the relationship between the two diagram.**

## Example 2: Diagram Refinement and Utilisation

In this example, we present how an SDL diagram is refined to make a transition from a purely visual state to executable mode. Let us imagine a situation where we need to implement a two-stage cluster survey sampling operation.

Stage 1: Visual layer
The first stage in performing this task is mapping requirements from the problem domain into the target domain of the survey data diagram. The diagram in Figure 4.15 shows a survey data diagram which models a two stage cluster survey sampling process.

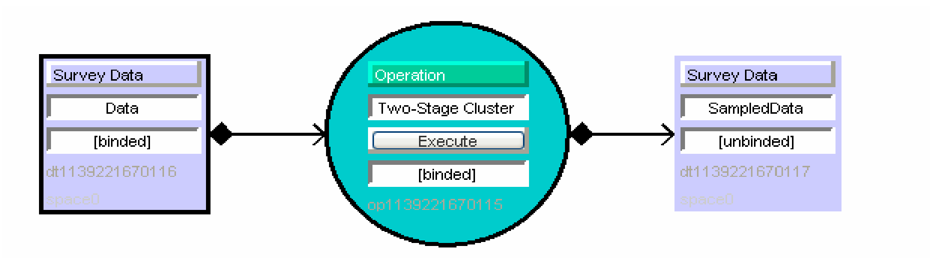
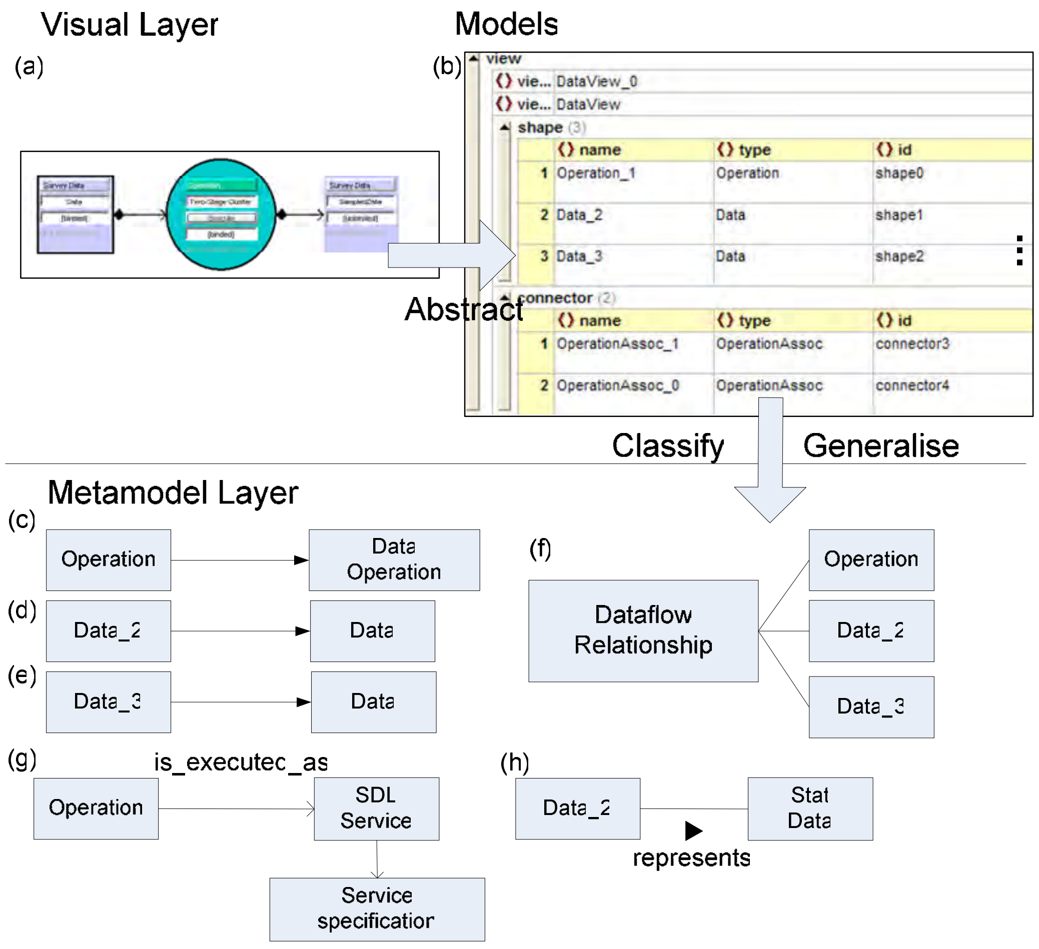


**Figure 4. 15 Survey data diagram which models a two stage cluster survey sampling process**

Stage 2: Metamodel generation and execution
The complete authoring of the visual layer may initiate the ACG (abstraction, classification and generalisation) process to build the metamodel which takes inputs from the visual layer and user generated events. At this stage two important things happen to turn the visual representation into a tool:

(a) Required external resources/services are mapped to the diagram.
(b) Event communication channels between the visual layer and the metamodel are established. When requests, which are in the form of user generated events, are made to the mapped services/resources the requests are routed to a service broker. Then the underlying metamodel is processed by the service broker to check out an appropriate proxy service. The outcomes of the interaction as outlined in Stage 2(b) may be propagated back to the visual layer and this two-way communication visualises (e.g. change in colours and layout at the visual model) the status of the operation as a standard software tool would.



view
- view DataView_0
- view DataView

shape (3)

| | name | type | id |
|---|---|---|---|
| 1 | Operation_1 | Operation | shape0 |
| 2 | Data_2 | Data | shape1 |
| 3 | Data_3 | Data | shape2 |

connector (2)

| | name | type | id |
|---|---|---|---|
| 1 | OperationAssoc_1 | OperationAssoc | connector3 |
| 2 | OperationAssoc_0 | OperationAssoc | connector4 |

**Figure 4. 16 (a) survey data diagram at the visual layer (b) metamodel file generated from the survey data diagram (c) – (h) metamodel entities and relationships that are derived from the metamodel using the ACG process**

## *4.6 Summary*

In this chapter, we put forward fundamental theoretical bases to clarify why SDL is dived into three layers and the implications of the approach in tool support. We have discussed the differences between the three layers and their nature and how the visual layer can be processed to build the metamodel layer. The concept of a survey entity was introduced to provide a simple platform for building and maintaining inter-diagram relationships.

As discussed in this section, SDL diagrams are much more than what users see on the surface. In the succeeding three chapters, we will observe how the layered approach is incorporated in building our prototype tool.

# Chapter 5 SDL Software Tool and Background

## 5.1 Introduction

The preceding chapters represented our research efforts to realise the visual modelling of statistical surveys in the context of diagrammatic notations. In this chapter, we will study software tool support for SDL by which survey users and designers utilise diagrammatic notations and blueprint the survey support infrastructure. We will familiarise ourselves with the assumptions and requirements for the development of supporting tools and investigate how tool support interact with the language design.

## 5.2 General Discussion on Tool Support

**Motivation**
Devising tool support for SDL is the topic the second phase of our research. Our motivation behind SDL tool support is two-fold. First, it makes sense to support SDL with software tools in the survey process. Let us expound on this rather broad statement.

One immediately apparent reason for tool support is to facilitate the acceptance of a new visual langue language in practice. The concept of incorporating a visual language into the survey process is still a largely untested novel approach in dealing with the myriads of operational difficulties, as discussed in Chapter 2, in statistical surveys. Proof-of-concept tools can give users real opportunities to create SDL-based methodologies and experiment them in the context of the survey process thus providing valuable feedbacks to us on the design decisions behind SDL at this critical stage of our research.

Secondly, in contrast to many visual languages, but in line with some notable visual programming languages such as LabView, SDL's diagrammatic notations were designed anticipating tool support. It may sound like an unjustified assumption in that we shaped SDL language features around the outwardly incongruent domain. The answer lies in the real world practice of the survey process and the suggestion of the most likely scenario for SDL. The usage of a visual language in the survey process is currently non-existent apart from the occasional use of organisational aids such as flowcharts, but there is significant tool support in many aspects of statistical computing and survey design since the early days of computing. Our consultation with a practicing statistician revealed that SDL without supporting software tools would not be viewed as a tangible asset in supporting the survey process due to the current state of statistical survey management in general. The necessity of tool support means that some design features of SDL should be seen in the light of the functional aspects of proof-of-concept tools.

Thus the second phase of this research is not a clear departure from the previous phase but has many shared components that influenced the preceding phase of our research. The design features of SDL that are affected by the presupposition will be presented in the main body of this chapter.

## Approaches and Inspirations

The lack of a visual approach in statistical surveys means we lack a rich pool of comparable solutions in the directly related areas of statistical surveys. A number of significant ideas from diverse sources have impacted this research and provided valuable insights into realising tool support for our visual language. Out of the insights we formulate our approaches to incorporate the positive aspects of such supporting tools into our problem domain.

It should be stressed that our notion of tool support must be clarified beforehand. Visual diagrams themselves in the forms of geometric structures without any software support can be described as tools (Blackwell 1998). Their cognitive values as tools precede the implementation of supporting software tools. Tool support for SDL entails any software artefact that is imposed upon the geometric structures that consist of SDL diagrammatic notations to achieve tasks in a target domain.

The issue of tool support for visual languages deserves a separate research consideration so we will limit our research scope in analysing existing approaches in tool support for UML. UML's modelling scope closely parallels that of SDL in that both aim to capture multiple aspects of a problem domain to build an integrated unified view. Also UML uniquely offers many solution candidates on heterogeneous platforms to offer us meaningful comparisons. Therefore their approaches to support the common language base illustrate core concerns of tool designers and different perspective on achieving quality software tool support.

Our choice for the candidate tools consisted of a diverse set of UML tools such as ArgoUML, Rational Rose, etc. It is easy to recognise many reasons why design disparity may arise amongst UML tools. It may result from using different development philosophies to turn perceived tool requirements to final implementations, from using different design concerns, or from having different history in their developments. Since all UML tools are part of more macro trends in software development, it is not surprising that they all share a large common functionality base.
The common functionality base of the UML tools are categorised as follows:

- Diagram authoring support
- Model Driven Architecture support
- Code generation
- Software reuse

| Category | Description |
|---|---|
| Diagram authoring support | Provide diagram editing features to create a new diagram and modify an existing diagram. |
| MDA | Support user generated models with related standards such as the Meta-Object Facility (MOF), the XML Metadata interchange (XMI), and the Common Warehouse Metamodel (CWM). |
| Code Generation | Provide a model based template or intelligent labelling using UML stereotypes to turn diagrammatic representation into working code. |
| Software Reuse | Support features to check out existing models and adapt them according to user requirements. Exploratory functionalities to look at reusable models (e.g. visualisation/documentation). Management of models using software engineering methodologies such as refactoring. |

**Table 5. 1 Common functionality base of the UML tools.**

To generalise from the categorised functionality list, it appears that a supporting tool for SDL should include graphical editing features to create and manipulate diagrammatic notations even at its minimally functional state.

The remaining three categories of the functionality base have a more common theme behind them. The common theme can be best described as a model based approach in building systems. The development of the approach presents an evolutionary path in the past decade of turning the amount of information presented in often ambiguous graphical diagrams into code level software artefacts, thus turning a purely visual diagram into an executable model. OMG's four-layer metamodel architecture is testament to the dominant macro trend in software engineering. Since the minimal functionality for SDL tool support can easily be met by the use of metatools such as Pounamu, we will put forward a brief introduction to the design aspects of SDL that fall into the three categories in conjunction of their relationship to the four-layer metamodeling architecture in the next section.

Regarding the methodology for measuring or estimating the usability of tools, the cognitive dimensions framework can be utilised throughout both the development stage of tools and the user testing stage. The cognitive framework provides a novel way of describing the usability of various aspects associated with software artefacts (Graphical notation, UI, etc). Our previous experience with the cognitive dimensions framework (Green 1996) driven analysis showed that in many instances the cognitive dimensions not only introduces the underlying concepts of "attention economics" (Blackwell and Green 1999) for the evaluation but extremely valuable usability gains can be obtained by retrofitting tool designs to realise optimal cognitive dimension trade-offs at early stages of the development. Therefore whenever essential tool design elements, which require the high level of user attention, such cognitive a dimension driven design decision is utilised.

**Model-based Approach and Tool**

All disciplines of natural sciences rely heavily on models to approximate and study complex phenomena (DHI) and a model is now a cornerstone of exploratory building blocks in many diverse fields outside natural sciences. The definition of the model can be imprecise but there seems to be collective consensus about what the model is and what the model should do. The explicative nature of models, which means they explains a phenomena in terms of more simpler, more basic processes (Howison 2005) without inherent complexity of the real instance of the phenomena, makes a model-based approach to be a significant force in separating the final software implementation  from its platform independent solution form.

As presented in chapter 4, a SDL diagrammatic representation has a three -layer architecture and as the four-layer architecture for UML is at the core of the model based approach in dealing with complex issues such as:

- Raising the level of abstraction
- Reuse
- Software asset management

SDL supporting tool should utilise the three layer architecture to
- alleviate operational difficulties associated with a platform specific code-driven activities and
- provide means to achieve the minimal loss of semantics between separate processes within the survey process.
- create a platform-free executable model

## 5.3 Language Design and Tool Support

In the previous discussion, we mentioned that the development of SDL diagrams anticipated software tool support and presented the contextual arguments surrounding the assumption. In our original research work (Kim, Hosking, and Grundy 2005) on an earlier version of SDL, it was noted that some language features of SDL can not easily be realised in the "pen and paper" mode of authoring diagrams and the absence of such features seemed to negatively affect the usability of the notations. Notable areas of interest were in:

- Implementation of a multiple-layered diagram
- Elision support for a graphical icons

Therefore SDL supporting tools are expected to aid users to make such features transparently accessible.

Our approach to the language which acknowledges the development of tools to handle sensitive cognitive dimensions trade-offs may be perceived as typical of the IT process is in that inherent shortcomings are always ready to be patched by another tool without addressing fundamental design issues. The counter arguments put forward to justify our position centre around following practical understanding of the SDL diagrammatic system:

- Diagrammatic systems are not inherently created with a fixed set of properties and concepts but must adjust legacy features or introduce new features in the light of new environments.
- Cognitive dimensions not only apply to notations themselves, but have been shown to be eclectically applicable across a wide range of programming related tasks and artefacts (Blackwell and Green 1999). Hence including tools related aspects of diagrammatic notation can be a more holistic approach to remedy the possible trade-offs in cognitive dimensions.
- We must decide which is the relevant usage model for SDL diagrammatic notions in practice. The most realistically envisaged model is not the "pen and paper" authoring system.

## *5.4 Requirements of Tool Support for SDL*

Looking back at the last few chapters, we see that there are many ways that software tools can help SDL to model the survey process in such areas as:

1. Diagram authoring support
2. Model management
3. Creating an executable diagram
4. Survey metadata repository

The requirements for our proof-of-concept tool to support SDL are:

1. It allows us the visual composition of SDL diagrams.
The visual composition support, which allows GUI based dynamic editing of visual diagrams and notations, is what is minimally required of our prototype tool. This is also the area that benefits most from the native functionalities of the 0Pounamu metatool. Pounamu takes model descriptions of the prototype tool and builds a visual diagram authoring tool and rich extensible APIs out-of-the-box. Hence Pounamu provides tremendous leverage in concentrating development efforts on the rest of the tool requirements.

2. It allows us to navigate back and forth between multiple diagram layers.
All five SDL diagrams have multi-layered visual components. A graphical icon may be decomposed into several sub layers. For example the survey dataset icon is mapped to the three aspects of statistical data: dataset, statistical metadata and the structural relationship. Our tool should not only support the design of the top layer but also enable editing of sub layers side-by-side.

3. Information elision support.
Even though each diagram addresses a specific aspect of statistical surveys, the complexity of the survey process may overload the visual information which is needed to be conveyed in a visual diagram. For instance whenever underlying

metamodels are updated to reflect user specifications, if all the information is visualized in a single layer then the visibility of the whole diagram can be deteriorated. Our proof-of-concept tool should implement design features to avoid those pitfalls that may arise from the lack of elision support.

4. It supports metadata usage in statistical computing.
As presented in Chapter 2, statistical data differ with respect to a number of aspects from traditional data (Grossmann 2003). And in many instances, the usefulness of statistical data is critically linked to the quality of metadata. The design of the survey data diagram is directly influenced by the metadata issue. The proof-of-concept tool should facilitate the utilisation of the various facets of statistical metadata so that the physical dataset description and data semantic can be made accessible seamlessly.

5. It supports metamodel generation.
Chapter 5 discussed how each diagram type consists of three layers. Amongst the three layers, the metamodel layer plays the most vital role from the tool point of view in preparing groundwork for specification generation. This is one of essential components in turning purely visual representations into interactive tools. Considering our target users, our proof-of-concept tool should provide necessary insulation from the detailed model generation as the tool user's foremost concern should be composing good visual diagrams.

5. Interactive evaluation of composed diagrams.
Survey data and technique diagrams mirror computation intensive survey activities such as sampling and statistical model verification. The proof-of-concept tool should provide dynamic design-time evaluation of composed data operations or techniques represented by SDL diagram. This will expand the user's experience of SDL diagrammatic notations in dual modes (communication medium and tool) and result in quick turnaround time from design to implementation.

7. Interoperable web service creation from diagrammatic specifications.
This is a valuable feature to be part of our overall solution. As the creation of interoperable and publicly accessible services, which are generated from the diagrammatical specifications, offers openness in client interaction choices.

8. Dynamic document generation.
The availability of metamodels provide opportunities for generating documents from metamodels. Their broad span allows many parts of record keeping activities to be done without explicit user interactions such as tracking statistical data production.

9. Third-party friendly tool outputs.
Tool outputs should be open for utilisation by third party tools.

## *Summary*

In this chapter we have presented a general discussion and exposed the requirements for the second phase of our research which focuses on the development of the proof-of-concept tool. We put forward our approaches and draw inspiration from the UML tool community. We identified key functional areas that must be supported by the proof-of-concept tool: diagram authoring support, model management, diagram execution, and metadata management. The tool requirements reflect the functional requirements.

In the next chapter, we will present the evolutionary path in creating our proof of concept tools.

# Chapter 6 SDL Tool Overview

## *6.1 Introduction*

In this chapter we present the evolutionary path creating our proof of concept tools. The starting point of the path is the suite of diagrammatic languages as presented in chapter 3 and 4. It has to be acknowledged first that our proof of concept tools do not exist themselves as part of a standalone software framework but are built on Pounamu metatool framework. Knowledge of Pounamu and metatool concepts should deepen the understanding of the proof of concept tools. However Pounamu's fluidity, logical componentisation and a low level of design interference for non-visual tasks should not make its presence a complicating factor within the whole architectural model.

The nature of prototype tool development with limited time and resources means that our research direction has focused on presenting broad high-level pointers and suggestions for future developments in the area of statistical survey process and domain specific visual language support tool. Therefore a number of shortcomings in software efficiency and insufficient fine-tuning of software components are expected to surface under the evaluation of underlying code and architectural descriptions.

In communicating our tool design and implementation, we have taken the following strategies to avoid unnecessary exhaustive design and implementation details and accentuate core research outcomes:

- Limited and reduced coverage on topical design problems such as transformation scheme for statistical datasets.
- Non-exhaustive architectural description with emphasis on how inner architectural components interact and correspond to SDL language features.
- User interaction models based on a small set of pre-defined usage patterns. Thus even when a prototype tool is not fully functional across the whole range of tool requirements, essential survey process activities can be modelled and executed using the tool.

Our discussion starts with a high-level architectural overview of the SDL supporting framework and is followed by discussion of how a set of problems is mapped to a tool-supported solution space. Then we cover tool features which cover the multiple aspects of the survey process then revisit them to explore how the features are realised by tool implementations. The anatomy of the software architecture sums up separately treated topics and some of the envisaged user interactions with the prototype tools will be presented at the end.

## 6.2 Architectural Overview

The primary goal of tool support is to facilitate the construction of the layered SDL model faithfully in every aspect for a given situation. Therefore the central point in our architectural model is the conceptual diagram layers.

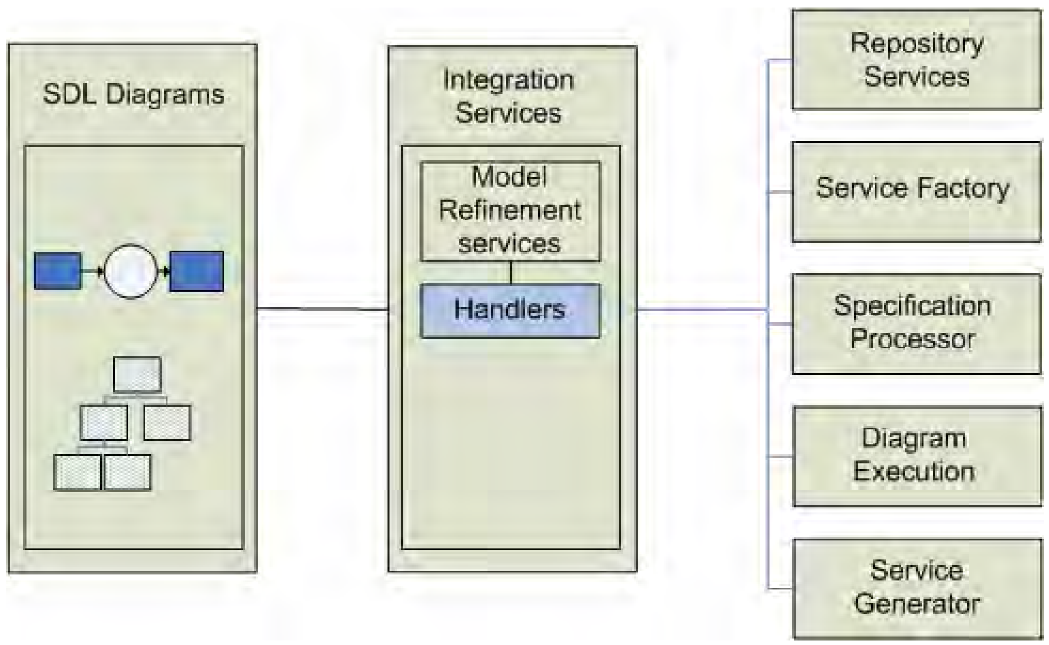


**Figure 6. 1 SDL tool support architecture**

Figure 6.1 shows key high-level components in the SDL tool support architecture and Table 6.1 summarises the role of each component.

| ***Component*** | ***Description*** |
|---|---|
| Handlers | The first component to handle both native events and user generated events of the diagram editor and relay them to appropriate components when it is required. |
| Model Refinement Services | Consist of server services to reflect changes in the visual layer by creating/updating the metamodel layer. |
| Repository Service | Registrar of SDL service (data operations and techniques) metadata. |
| Service Factory | Provides a proxy object to enable communication with a remote service. |
| Specification Processor | Translate the underlying diagrammatic model into execution-ready specifications. |
| Diagram Execution | Executes provided specifications. |
| Service Generator | Generates diagrammatic specifications into inter-operable services (e.g. Axis hosted web services) |

**Table 6. 1 Functional role of tool support components**

The key components collaborate to achieve the primary goal of SDL tool support. All SDL diagrams reside within Pounamu metatool environment therefore the message communication model for the SDL tool support is tied up with how Pounamu handles events. Understanding of how events are handled and propagated to collaborating components is essential in extending Pounamu's native support for diagram authoring.

**Processing events – Handlers, message propagation and model processing**

From the standpoint of the SDL diagram user, all first tier event handling is delegated to Pounamu handlers that wait for pre-specified types of events.
When users interact with SDL diagrams, a series of handlers may be invoked. The selectiveness of invocation is determined by types of events which the handlers subscribe to.

There are two different ways the events are raised:

1. Events are invoked when there are visual changes in a diagram. e.g. changing property values, deleting graphical icon and inserting a new shape.
2. User defined events are triggered when user-defined menu items are selected just like menu toolbar interactions in a typical GUI application.

In either case, each handler has an opportunity to do whatever it wants to do.
Let us investigate a simple representative scenario where users compose a diagram to express specifications for a sampling operation. We need to bind the input dataset to one of the real dataset files which are available at our dataset repository. Binding is one of core concepts to be grasped by users and is covered in detail in the latter section which is specifically devoted to resource and service binding. Prompting the menu on the icon result in the following event propagation as shown in Figure 6.2.

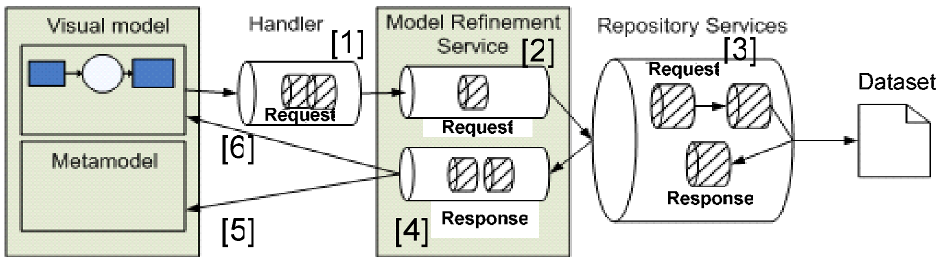


**Figure 6. 2 Event propagation model of the SDL prototype tool**

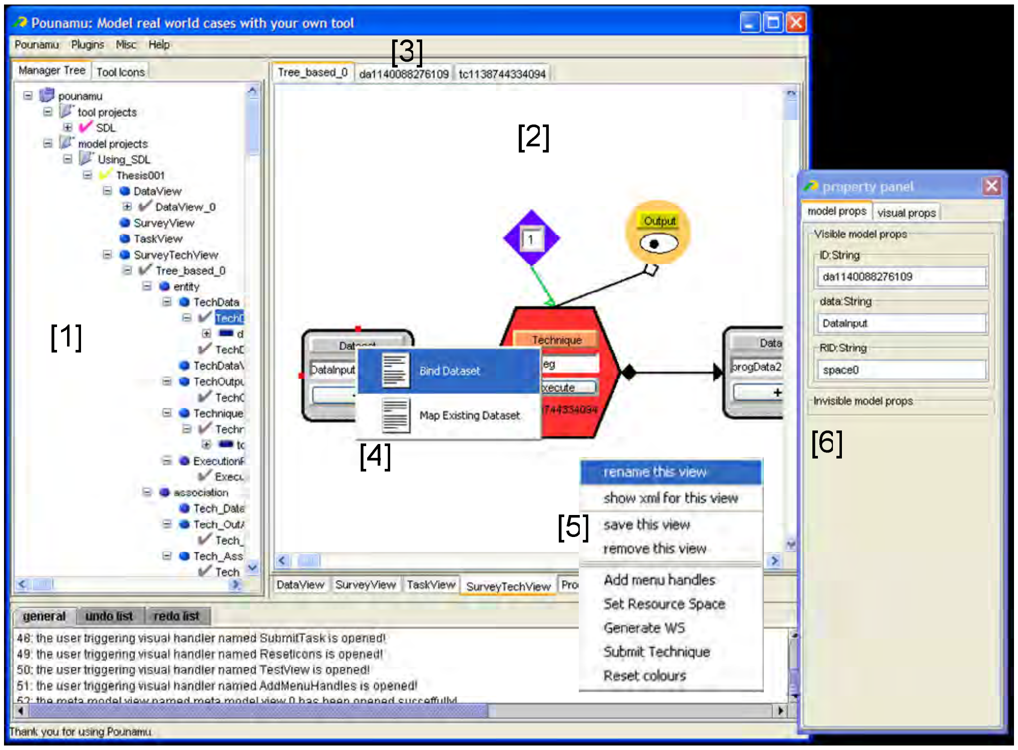
An event arrives at a designated handler [1]. The event signifies the user's decision to bind one of available datasets to a graphical entity. The handler interprets the event, generating an appropriate UI for user interactions and passes captured messages to the model refinement service [2] which updates/creates both visual and metamodel layers of the diagram accordingly.

The model refinement service first looks up the dataset by given resource ID [3]. Dataset repository service is an indexing service which acts as a registry for a dataset resource maps. If the requested dataset will be linked to the metamodel element behind the visual icon [4]. This will result in updating the metamodel [5] and the visual model for the visual icon [6] (visual layout of the icon will change to reflect the status of the binding operation). An interesting result of this approach is that assuming the same dataset can be bound to multiple visual icons we can trace all statistical operations that utilise the dataset and their outcomes by scanning and applying transformation schemas upon the metamodels behind the SDL diagrams.

## 6.3 Tool in Action

In this section, we present the prototype tool in action. Screenshots of the tool are taken and categorised by their functional purposes. We only offer an informal walkthrough of what users may encounter while using our prototype tool. Chapter 8 offers a systematic view on the contextual backgrounds, design choices and implementation issues regarding our tool.

***Diagram Authoring Environment***



ID:String da1140088276109
data:String DataInput
RID:String space0

48: the user triggering visual handler named SubmitTask is opened!
49: the user triggering visual handler named ResetIcons is opened!
50: the user triggering visual handler named TestView is opened!
51: the user triggering visual handler named AddMenuHandles is opened!
52: the meta model view named meta model view 0 has been opened successfully!

**Figure 6. 3 Screenshot of the prototype tool**

Figure 6.3 show the proof-of-concept tool being used to compose SDL diagrams. Beside usual features such as GUI based graphic icon placement and editing [1,6], contextual menus [4,5] are available to perform various tasks such as model generation, resource binding and inter-diagram mapping. Each diagram occupies one tabbed view panel [2] and associated sub layers of a graphical icon are revealed in the tabs of the tool window [3]. Appropriate user actions such as clicking on a contextual menu item causes to the tab view which represents a sub layer of the currently focused graphical icon as shown in Figure 6.4.

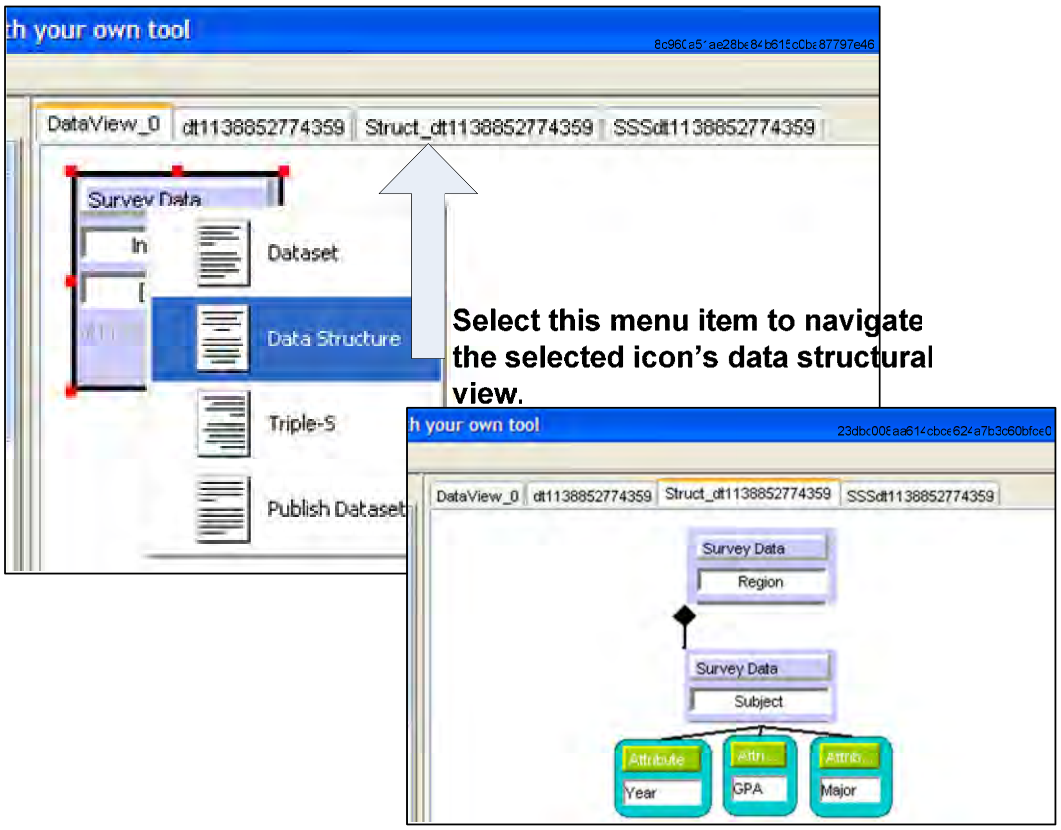


**Figure 6. 4 Multi-layer support in action, showing the population-sample relationship of the selected data icon**

The prototype tool supports all five types of SDL diagrams: survey, data, technique, task and process as shown in Figure 6.5. A typical diagram authoring session is show in Figure 6.6. To demonstrate the tool in action, in the following example we will go through necessary steps in creating a SDL diagram as shown in Figure 6.6. The example will only take us to the stage where every visual entity is a purely visual representation. To create a simple survey technique diagram, which describes a data exploration using boxplot, we place necessary visual icons on the tool window [1] and edit their attributes using the flowing property panel. Then the relationships between icons are expressed by connecting them with appropriate connectors [2]. Now we have an input dataflow from the data icon 'Eco_Crime' [a] to the technique icon 'Boxplot' [b] and technique outcomes are channelled to the output port [c] as shown in Figure 6.5 (c). Visually the diagram is complete however to utilise the diagram to run the boxplot technique, we need to map the visual icons to external resources such as data files and computation services as shown in Figure 6.7.

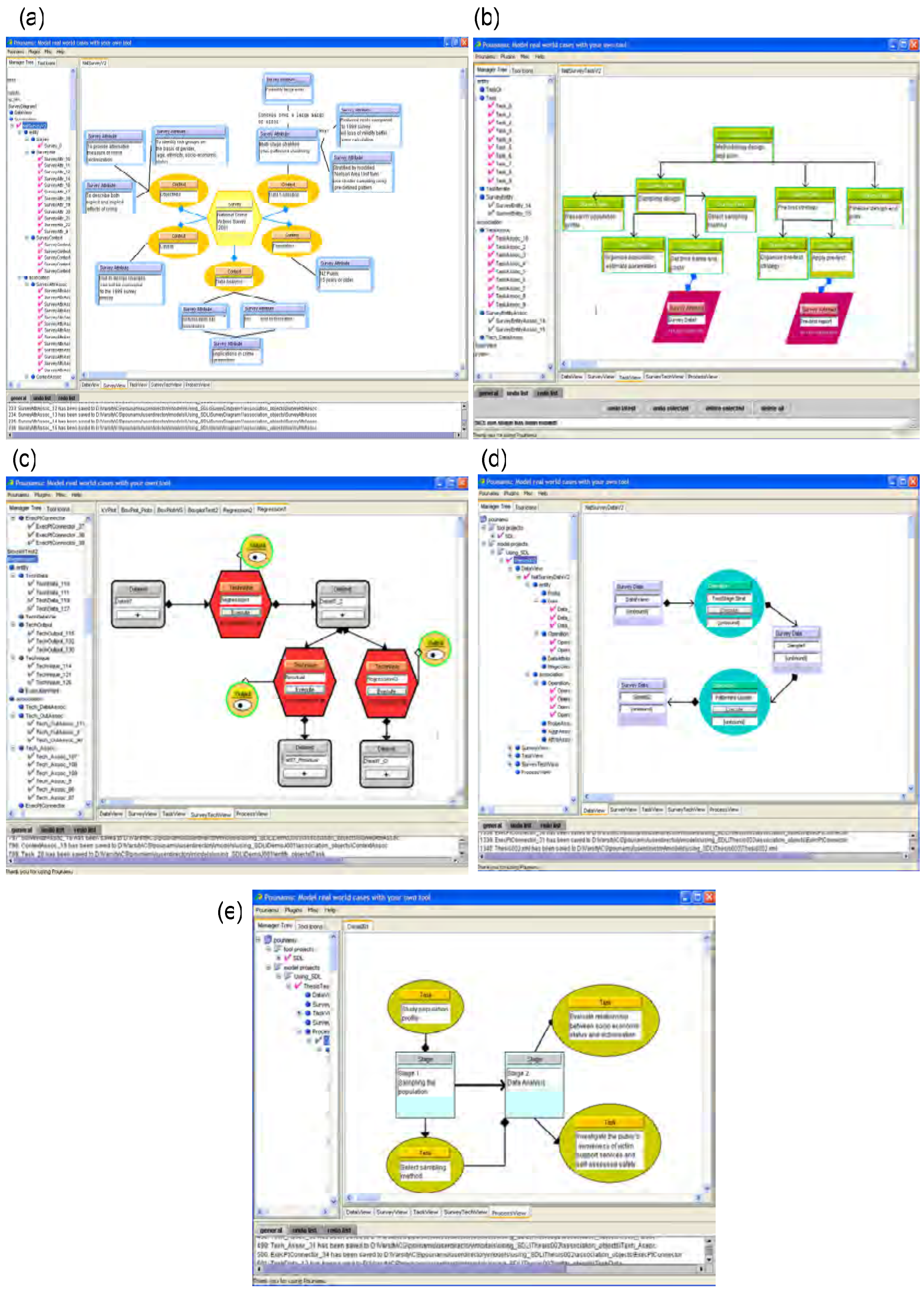
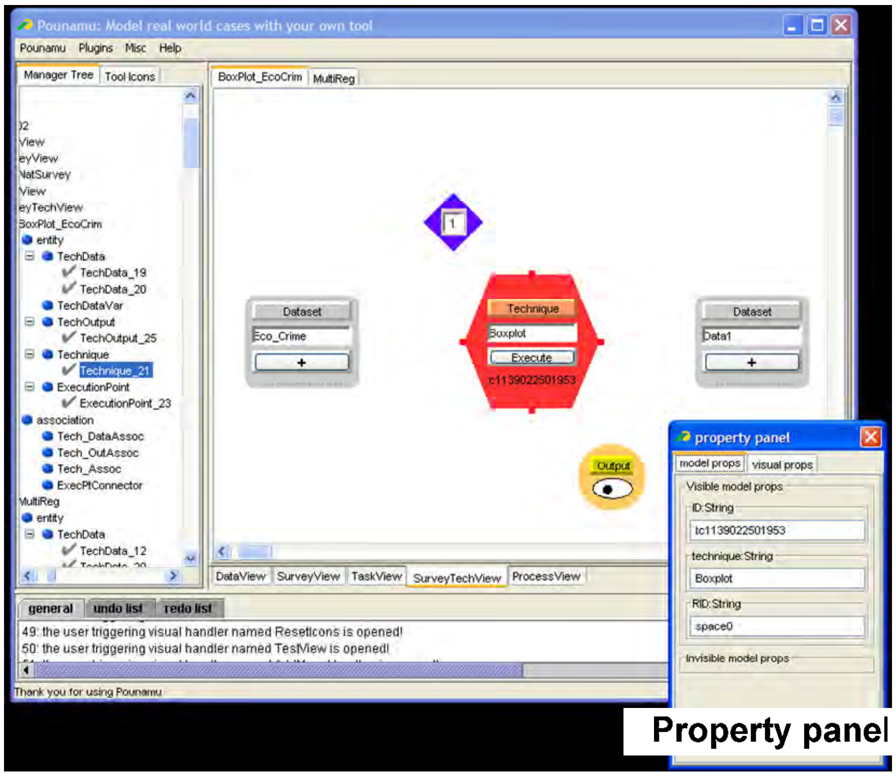


**Figure 6. 5 Five types of survey diagrams composed with the prototype tool (a) survey diagram (b) survey task diagram (c) survey technique diagram (d) survey data diagram (e) survey process diagram**

[1]

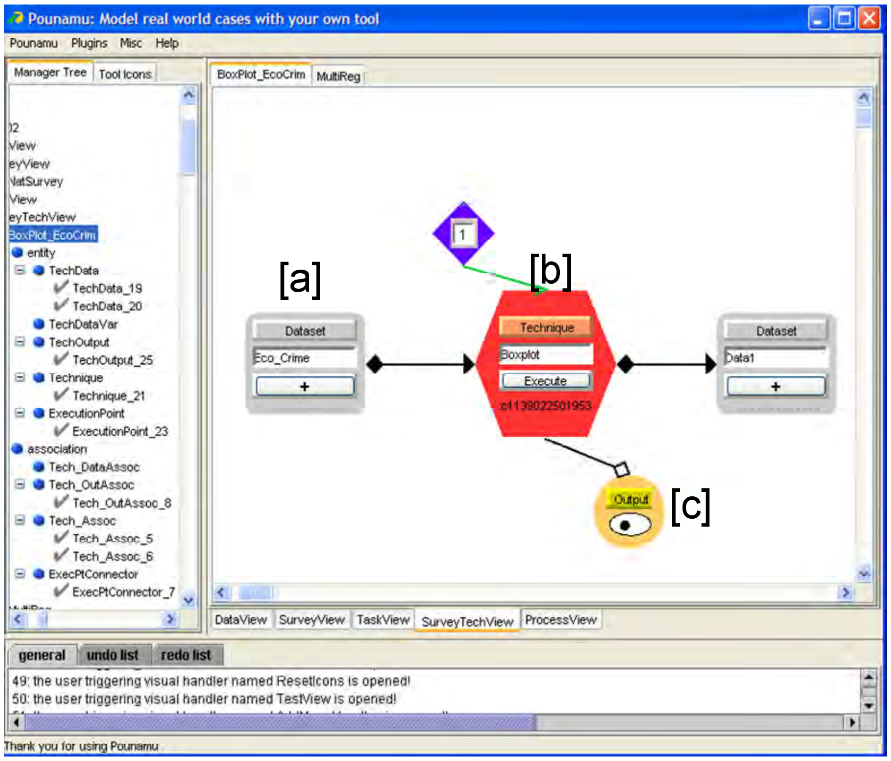


ID:String
tc1139022501953
technique:String
Boxplot
RID:String
space0

49: the user triggering visual handler named ResetIcons is opened!
50: the user triggering visual handler named TestView is opened!

Thank you for using Pounamu

[2]

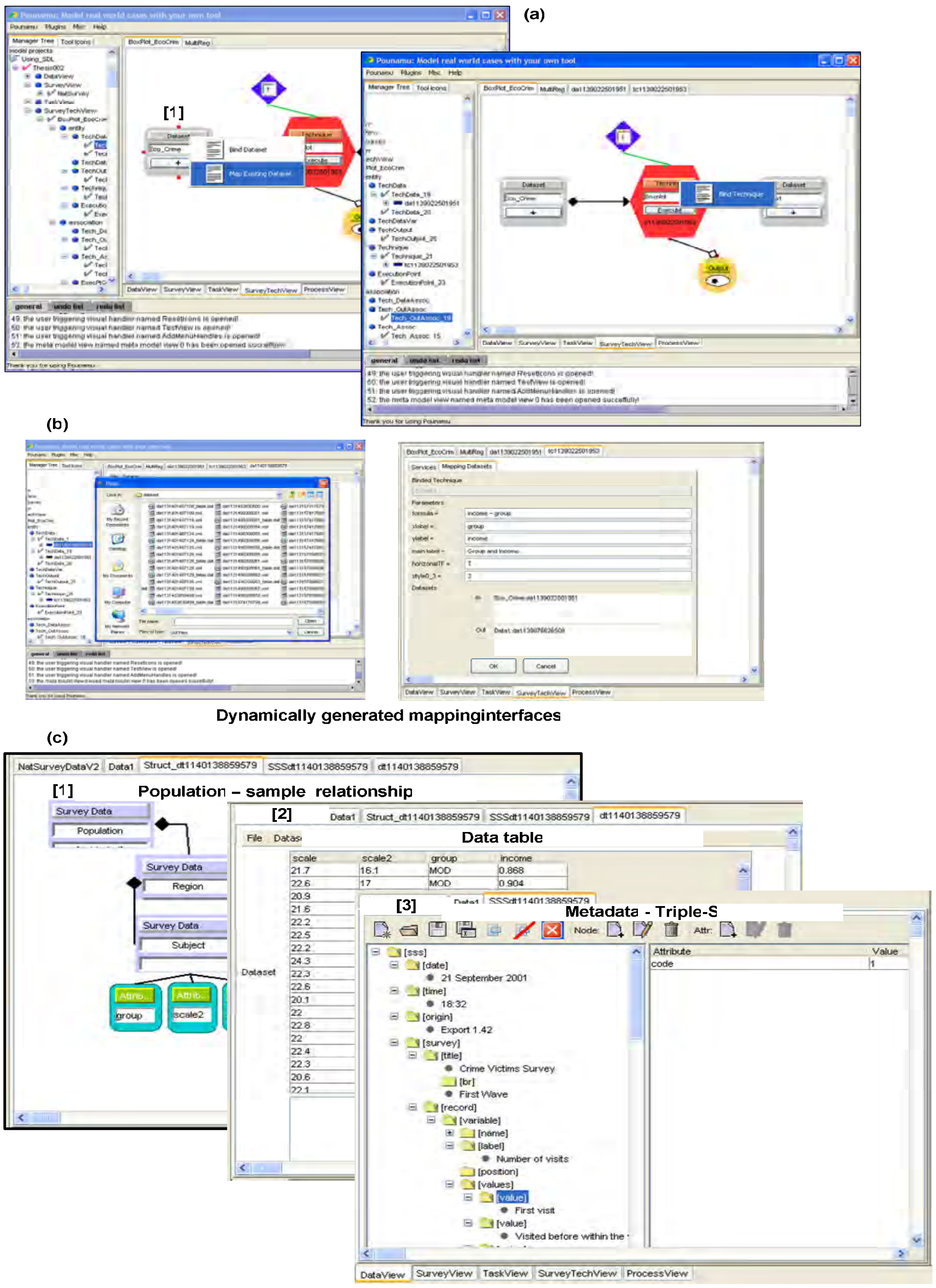


49: the user triggering visual handler named ResetIcons is opened!
50: the user triggering visual handler named TestView is opened!

Thank you for using Pounamu

**Figure 6. 6 A survey technique diagram for a boxplot technique**

*Resource Binding*



49: the user triggering visual handler named ResetIcons is opened!
50: the user triggering visual handler named TestView is opened!
51: the user triggering visual handler named AddMenuHandles is opened!
52: the meta model view named meta model view 0 has been opened succesfully!

Binded Technique

Parameters

formula = income ~ group
xlabel = group
ylabel = income
main label = Group and Income
horizontalTF = T
style0_3 = 2

Datasets
In: Eco_Crime:dat1139022501951
Out: Data1: dat1139078838508

| scale | scale2 | group | income |
|---|---|---|---|
| 21.7 | 16.1 | MOD | 0.868 |
| 22.6 | 17 | MOD | 0.904 |
| 20.9 | | | |
| 21.6 | | | |
| 22.2 | | | |
| 22.5 | | | |
| 22.2 | | | |
| 24.3 | | | |
| 22.3 | | | |
| 22.6 | | | |
| 20.1 | | | |
| 22 | | | |
| 22.8 | | | |
| 22 | | | |
| 22.4 | | | |
| 22.3 | | | |
| 20.6 | | | |
| 22.1 | | | |

```
[sss]
  [date]
    21 September 2001
  [time]
    18:32
  [origin]
    Export 1.42
  [survey]
    [title]
      Crime Victims Survey
      [br]
      First Wave
    [record]
      [variable]
        [name]
        [label]
          Number of visits
        [position]
        [values]
          [value]
            First visit
          [value]
            Visited before within the
```

| Attribute | Value |
|---|---|
| code | 1 |

Figure 6. 7 Binding visual icons to external resources (a) – (c)

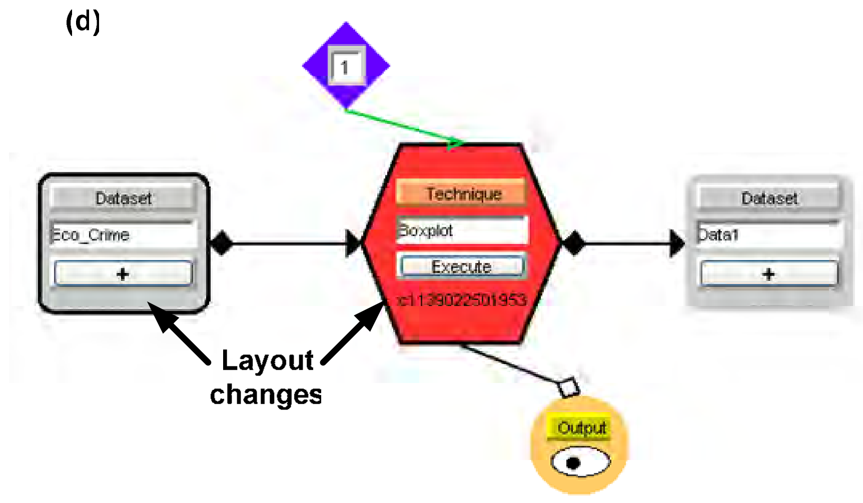


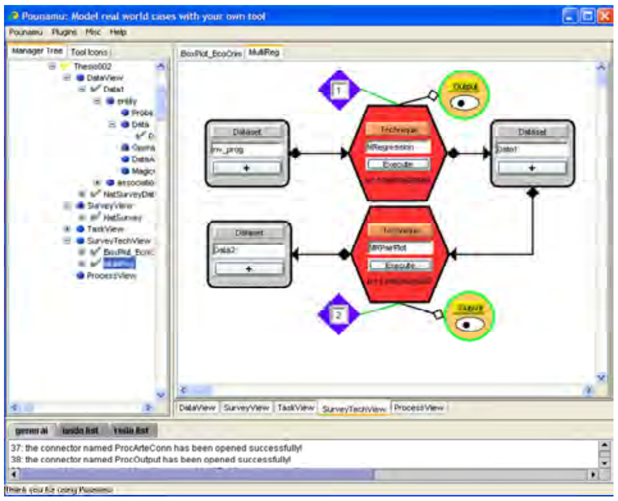
**Figure 6.7 (d) A survey technique diagram which is ready to be executed**

Binding operations which map graphical icons to external resources will be the most frequently used functionality of the tool once all diagrammatic components are laid out. The user may initiate a mapping process by invoking a menu associated with a selected graphical icon as in Figure 6.7 (a). Then a selecting appropriate mapping option will generate mapping forms for user to complete as shown in Figure 6.7 (b). Many fields of the mappings forms are filled automatically as the model refinement services can infer such attributes as data flows and available resources for the graphical icon type. Once the mapping forms are completed then a newly mapped aspect of the icon will appear in a separate view panel tab and the user is able to edit and create the new view visually (Figure 6.7 (c) [1,2,3]). When the visual icons are successfully mapped their outer layout line is changed as shown in Figure 6.7 (d).

### *Diagram Execution*

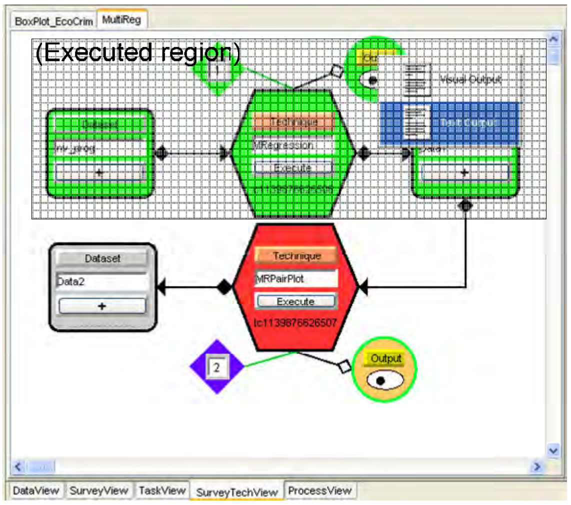
When a survey data or technique diagram is mapped to all required statistical data and techniques. We may begin to use the diagram as a tool to execute it in real-time and explore its textural and visual outcomes. Figure 6.8 shows a typical sequence of diagram execution. Figure 6.8(a) shows a complete survey technique diagram with all required graphical icons are mapped to appropriate resources as their dark-lined layout lines suggest.

Each technique icon has a button interface which can generate an event to execute the metamodel-based specification. The green coloured icons (hatched region) as shown in Figure 6.8(b) indicate that the execution was complete and successful. The bottom half of the diagram with the original colours (unhatched region) reflect the current status of the execution. The user may select the output port of the executed technique and probe into available outcomes as shown in Figure 6.8 (c) and (e).

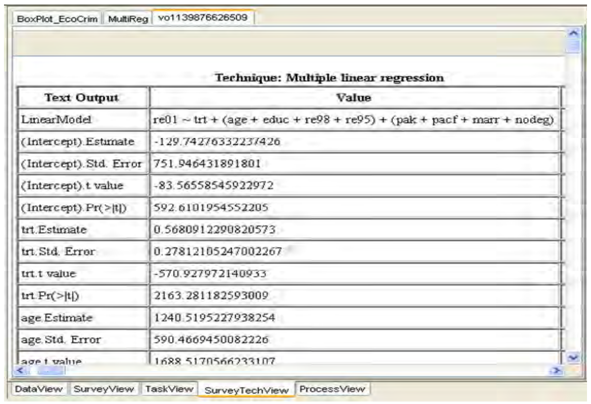


37: the connector named ProcArteConn has been opened successfully!
38: the connector named ProcOutput has been opened successfully!

(a)



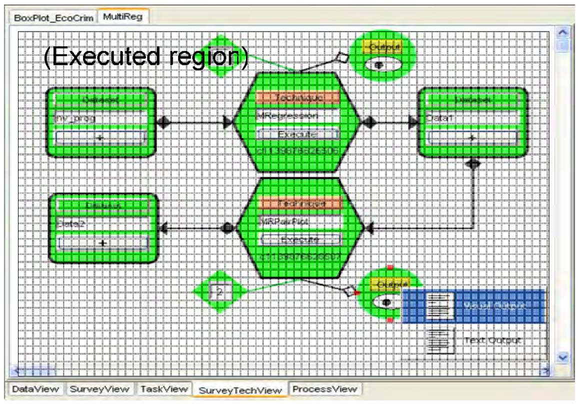
(b)



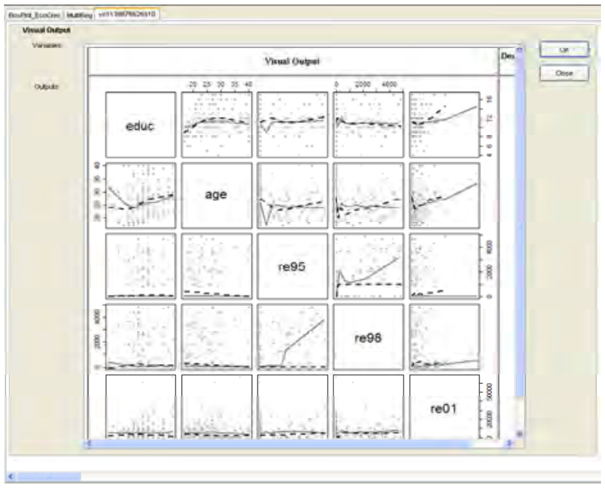
Technique: Multiple linear regression

| Text Output | Value |
|---|---|
| LinearModel | re01 ~ trt + (age + educ + re98 + re95) + (pak + pacf + marr + nodeg) |
| (Intercept).Estimate | -129.74276332237426 |
| (Intercept).Std. Error | 751.946431891801 |
| (Intercept).t value | -83.56558545922972 |
| (Intercept).Pr(>\|t\|) | 592.6101954552205 |
| trt.Estimate | 0.5680912290820573 |
| trt.Std. Error | 0.27812105247002267 |
| trt.t value | -570.927972140933 |
| trt.Pr(>\|t\|) | 2163.281182593009 |
| age.Estimate | 1240.5195227938254 |
| age.Std. Error | 590.4669450082226 |
| age.t value | 1688.5170566233107 |

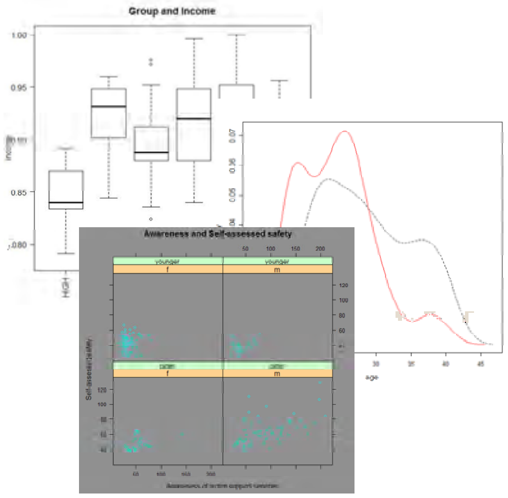
(c)



(d)



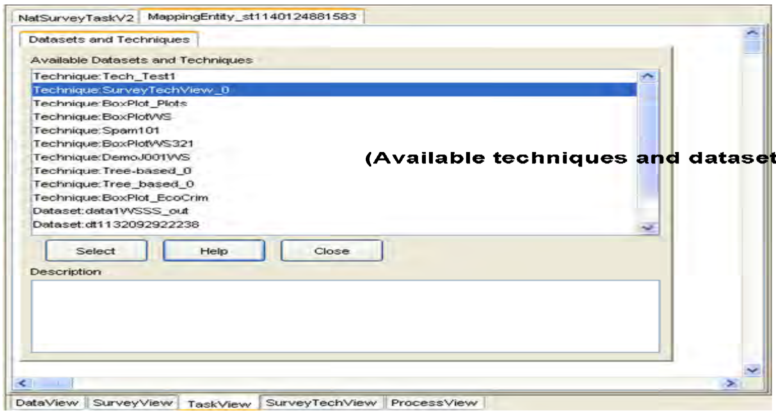
(e)



(f)

**Figure 6. 8 Survey technique diagram (b) Partially-executed diagram (c) Textural outcomes of the technique (d) Fully-executed diagram (e) Visualisation of the technique outputs (f) some of other visualisation techniques available**

(a)

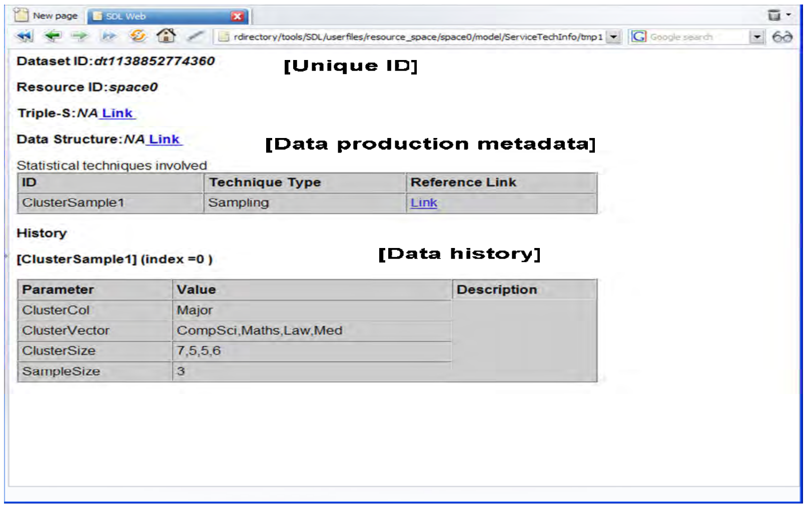


Available Datasets and Techniques

Technique:Tech_Test1
Technique:SurveyTechView_0
Technique:BoxPlot_Plots
Technique:BoxPlotWS
Technique:Spam101
Technique:BoxPlotWS321
Technique:DemoJ001WS
Technique:Tree-based_0
Technique:Tree_based_0
Technique:BoxPlot_EcoCrim
Dataset:data1WSSS_out
Dataset:dt1132092922238

Description

(b) Generated stat data documentation


New page
SDL Web
rdirectory/tools/SDL/userfiles/resource_space/space0/model/ServiceTechInfo/tmp1
Google search
[Unique ID]
[Data production metadata]
[Data history]

**Dataset ID:***dt1138852774360*

**Resource ID:***space0*

**Triple-S:***NA* Link

**Data Structure:***NA* Link

Statistical techniques involved

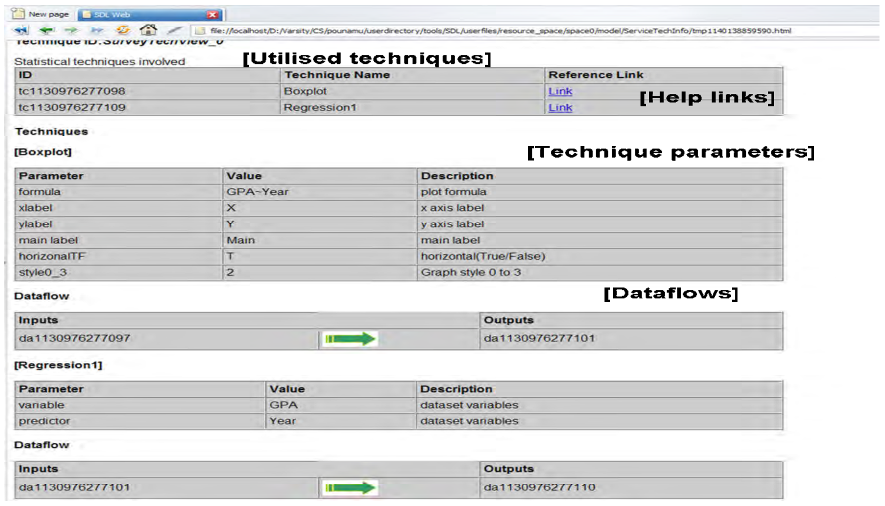
| ID | Technique Type | Reference Link |
|---|---|---|
| ClusterSample1 | Sampling | Link |

**History**

**[ClusterSample1] (index =0 )**

| Parameter | Value | Description |
|---|---|---|
| ClusterCol | Major | |
| ClusterVector | CompSci,Maths,Law,Med | |
| ClusterSize | 7,5,5,6 | |
| SampleSize | 3 | |

(c) Generated technique documentation


New page
SDL Web
file://localhost/D:/Varsity/CS/pounamu/userdirectory/tools/SDL/userfiles/resource_space/space0/model/ServiceTechInfo/tmp1140138859590.html
[Utilised techniques]
[Help links]
[Technique parameters]
[Dataflows]

Technique ID:SurveyTechView_0

Statistical techniques involved

| ID | Technique Name | Reference Link |
|---|---|---|
| tc1130976277098 | Boxplot | Link |
| tc1130976277109 | Regression1 | Link |

**Techniques**

**[Boxplot]**

| Parameter | Value | Description |
|---|---|---|
| formula | GPA~Year | plot formula |
| xlabel | X | x axis label |
| ylabel | Y | y axis label |
| main label | Main | main label |
| horizonalTF | T | horizontal(True/False) |
| style0_3 | 2 | Graph style 0 to 3 |

**Dataflow**

| Inputs | | Outputs |
|---|---|---|
| da1130976277097 | ➡ | da1130976277101 |

**[Regression1]**

| Parameter | Value | Description |
|---|---|---|
| variable | GPA | dataset variables |
| predictor | Year | dataset variables |

**Dataflow**

| Inputs | | Outputs |
|---|---|---|
| da1130976277101 | ➡ | da1130976277110 |

**Figure 6. 9 (a) mapping UI showing available techniques and data (b) Tool generated documentation on cluster sampled statistical data (c) Tool generated documentation for a statistical technique.**

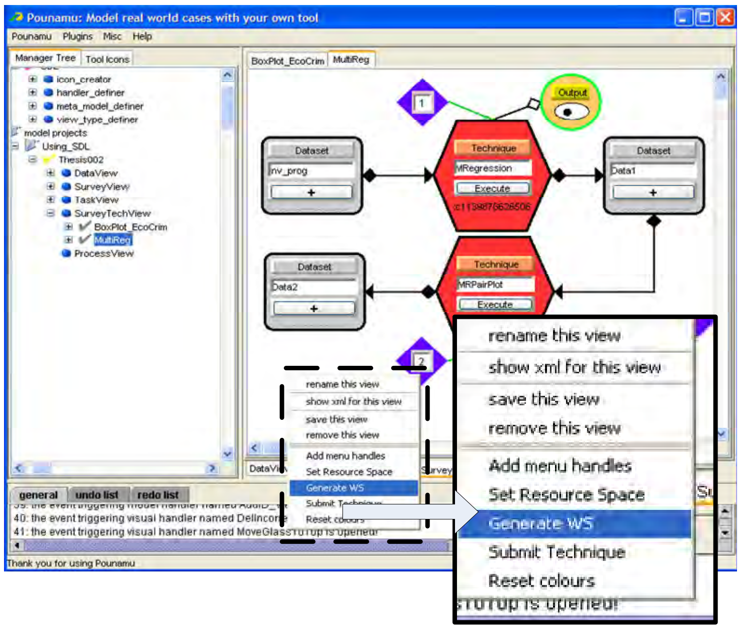
The visual layer information is processed and also refined at the metamodel layer. Considering the importance of documentations in managing survey process, the ability of our tool to generate documents opens doors for third-party tools or external clients to look at the semantics of visual diagrams without the need for the prototype tool. Suppose that the diagram in Figure 6.8 has been exported to the model repository service, the document generation can be demonstrated from within our own tool environment as follows:

- From one of mapping interfaces, we wish to select the technique which is represented by the exported diagram.
- To view the description of the diagram, we select the help button on the mapping interface as shown in Figure 6.9 (a).
- The underlying metamodels for the diagram is processed by document generating schemas then the resulting HTML document is shown in the browser as in Figure 6.9 (b) and (c).

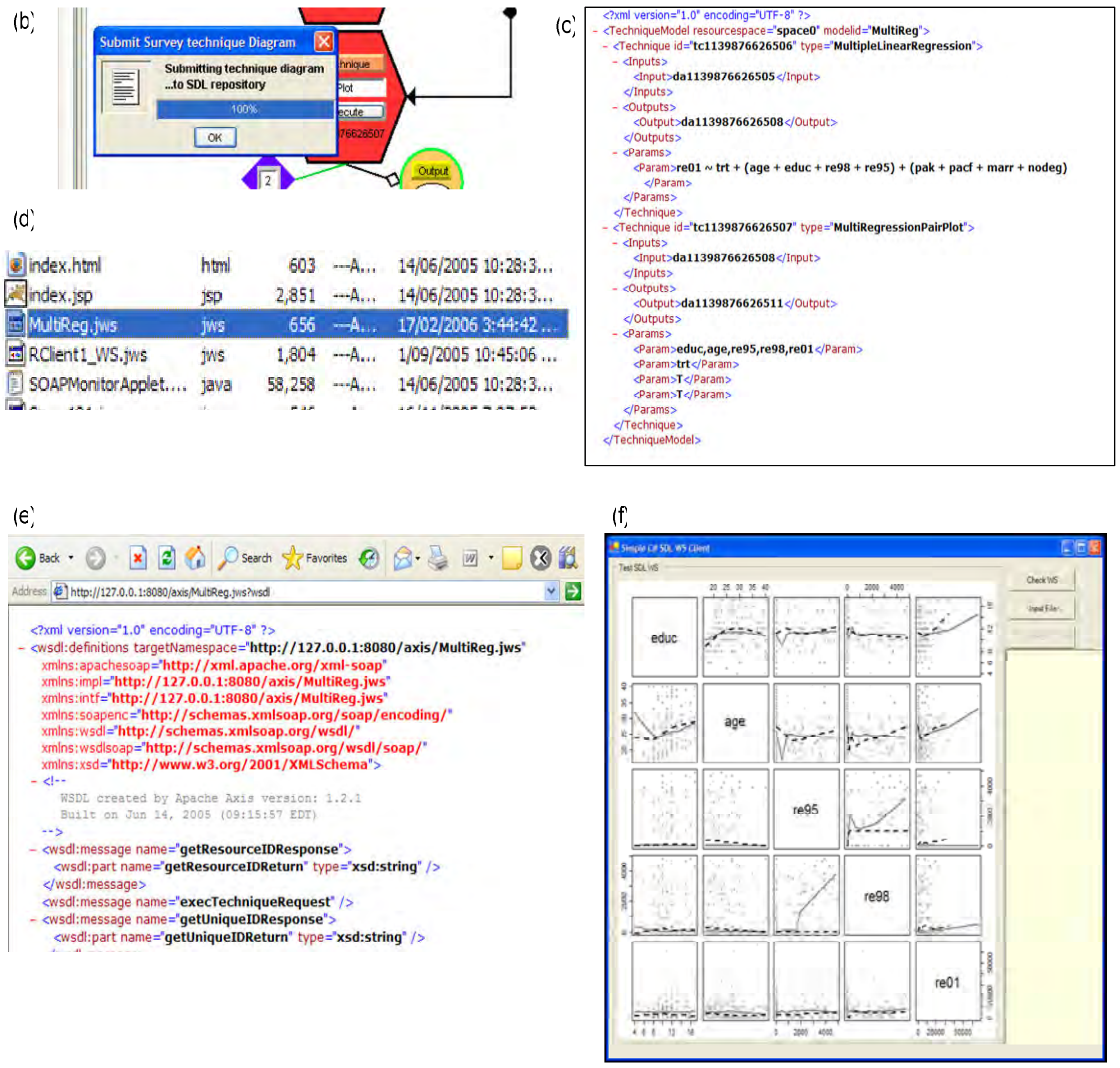
***Service Generation***

When the user is satisfied with the correctness of a functional technique diagram. The diagram can be turned into Java code and exposed in the form of a web service. Figure 6.10 (a,b) shows the user invoking the contextual menu to generate a service based on the diagrammatic specification of the current diagram. The diagram is submitted to the model repository in the form shown in Figure 6.10 (c). The submitted model is processed according to a web service generation template and the generated code is hosted by Axis (Figure 6.10(d)) and clients may access the service specification via WSDL interface as shown in Figure 6.10 (e). Figure 6.10(f) show a demo program which is built on the Microsoft .Net framework utilising the statistical technique developed with our prototype tool.  Now two very different development environments and approaches can cooperate together in building a survey supporting infrastructure.

(a)



40: the event triggering visual handler named Delincor
41: the event triggering visual handler named MoveGlassToTop is opened!



Submit Survey technique Diagram

Submitting technique diagram ...to SDL repository

100%

OK

```
<?xml version="1.0" encoding="UTF-8" ?>
- <TechniqueModel resourcespace="space0" modelid="MultiReg">
  - <Technique id="tc1139876626506" type="MultipleLinearRegression">
    - <Inputs>
        <Input>da1139876626505</Input>
      </Inputs>
    - <Outputs>
        <Output>da1139876626508</Output>
      </Outputs>
    - <Params>
        <Param>re01 ~ trt + (age + educ + re98 + re95) + (pak + pacf + marr + nodeg)
          </Param>
      </Params>
    </Technique>
  - <Technique id="tc1139876626507" type="MultiRegressionPairPlot">
    - <Inputs>
        <Input>da1139876626508</Input>
      </Inputs>
    - <Outputs>
        <Output>da1139876626511</Output>
      </Outputs>
    - <Params>
        <Param>educ,age,re95,re98,re01</Param>
        <Param>trt</Param>
        <Param>T</Param>
        <Param>T</Param>
      </Params>
    </Technique>
</TechniqueModel>
```

```
index.html          html     603  ---A...  14/06/2005 10:28:3...
index.jsp           jsp    2,851  ---A...  14/06/2005 10:28:3...
MultiReg.jws        jws      656  ---A...  17/02/2006 3:44:42 ...
RClient1_WS.jws     jws    1,804  ---A...  1/09/2005 10:45:06 ...
SOAPMonitorApplet.... java 58,258 ---A...  14/06/2005 10:28:3...
```

http://127.0.0.1:8080/axis/MultiReg.jws?wsdl

```
<?xml version="1.0" encoding="UTF-8" ?>
- <wsdl:definitions targetNamespace="http://127.0.0.1:8080/axis/MultiReg.jws"
    xmlns:apachesoap="http://xml.apache.org/xml-soap"
    xmlns:impl="http://127.0.0.1:8080/axis/MultiReg.jws"
    xmlns:intf="http://127.0.0.1:8080/axis/MultiReg.jws"
    xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  - <!--
      WSDL created by Apache Axis version: 1.2.1
      Built on Jun 14, 2005 (09:15:57 EDT)
    -->
  - <wsdl:message name="getResourceIDResponse">
      <wsdl:part name="getResourceIDReturn" type="xsd:string" />
    </wsdl:message>
    <wsdl:message name="execTechniqueRequest" />
  - <wsdl:message name="getUniqueIDResponse">
      <wsdl:part name="getUniqueIDReturn" type="xsd:string" />
```

**Figure 6. 10 Web service generation and utilisation**

## 6.4 Summary

In this chapter we have presented an architectural overview of our prototype tool that support SDL diagrams and showed the tool to exemplify some of the tool features in action. In Chapter 4 we put forward the theoretical bases to turn purely visual diagrams into executable diagrams and we have achieved the goal of making SDL diagrams more than just a conceptual aid. In the context of the survey process, this is a significant progress as survey designers now have an option to map conceptual modelling artefacts into real-world survey resources seamlessly.

Detailed implementation choices behind the SDL tool support is discussed in the next chapter.

# Chapter 7 Discussion on Tool Implementation

## *7.1 Introduction*

This chapter focuses on the functional decompositions of the proof-of-concept SDL tool and the design choices made during tool implementation. The SDL tool harnesses a vast array of functionalities. The functional components of our tool are closely connected to the layered strategies for the diagrammatic representations as presented in chapter 5. The goal of tool support is to facilitate the construction of the layered SDL model faithfully in every aspect for a given situation. Therefore all those functional decompositions can be better understood in the context of the layered model.

A high level view on the key requirements for the SDL tool in terms of the core functional areas is as shown here:

- Diagram authoring support
- Specification design and refinement ( e.g. statistical data operations and inter-diagram relationship)
- Diagram execution (includes run-time back-end integration to computations engines or services)
- Model-based service generation

When comprehending tool implementation, an important aspect is that each functional area corresponds to a solution for a particular facet of the survey process in practice. The core functional areas form the basis of much of the survey process. Thus not to lose the contextual perspective on the implemented functionalities, they will be presented with problem definitions and the contextual information for proposed design solutions whenever necessary.

## *7.2 Visual model support - Diagram authoring support*

**Problem**
We need to draw a visual diagram to express a particular aspect of statistical survey.

**Context**
Diagram authoring is directly related to the visual model layer of SDL diagrams since the visual model layer is an instance of what users see and manipulate on the tool window view. Visually SDL diagrams can be generalised into a diagram consists of shapes/graphical icons and connectors (e.g. directional arrow representing a dataflow). The functional components for diagram authoring support act within the following structural context, as shown in Figure 7.1

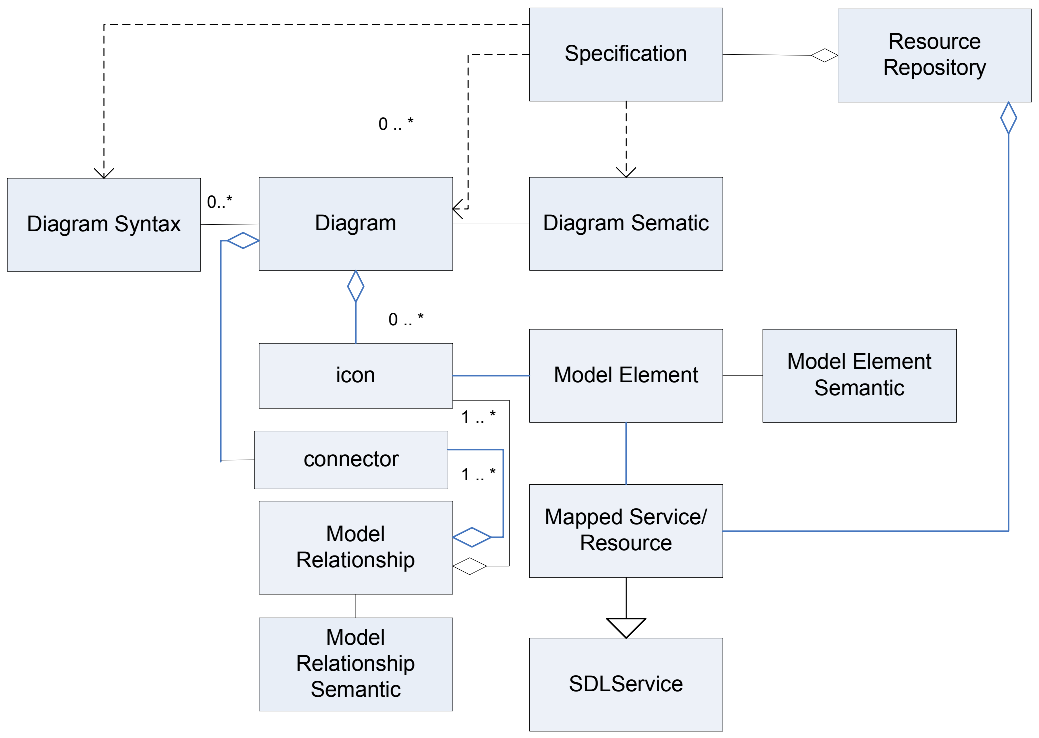


**Figure 7. 1 Structural contexts for diagram authoring support**

**Implementation Solution**

The implementation of the prototype tool relies on Pounamu's out of the box ability to provide the majority of functionalities expected from a diagram editor and the utilisation of Pounamu's event-based communication model which allows the insertion of custom built components to intercept and process broadcasted events as in Figure 7.2 of Chapter 6. The visual syntactic element of SDL diagrams can be imposed by intercepting user-triggered events or visual events that signal changes in diagram's syntactic structure. Then the intercepted event messages are handled by Pounamu's event handlers that can compare the intended user actions against the syntactic rules. Implementation details beyond the visual level support are presented in the next section.

## 7.3 Specification Design and Refinement (Refinement Process)

**Problem**

SDL diagrams have been composed but we need to refine the diagrams to create more detailed specifications that enable the execution of the diagrams.

**Context**

SDL diagrams at the visual level like many other visual diagrams may lack concrete specifications (Novak 1992). UML's approach to the problem is the use of OCL to put constraint based clarifications. The presence of ambiguity can be tolerated when communicating the survey process between human participants however SDL diagrams may serve many other roles then just the human-to-human communication medium. This is a significant challenge for software tools that demand more precise means of communication.

## Implementation Solution

SDL's domain specificity and narrower modelling scope mean that we can reduce a diagram's ambiguity more readily to produce executable specifications by using the following techniques:

1. The visual model is only a basic layer in which everything is represented by graphical icons and connectors. The visual model has dimensions of shapes, names, coordinates, etc. For each graphical icon, a corresponding metamodel entity at the metamodel layer can provide desired concrete descriptions of the graphical entity which resides at the visual layer. Let us demonstrate this concept of using the two layers to produce executable specifications in Figure 7.2, which shows a simple visual representation of an arithmetic operation. We see a typical dataflow metaphor driven representation at the visual layer and can easily recognise the purpose and outcomes of the diagram. But for the visual model to be executable, required mappings between the graphical entities and real-world entities (real data inputs and working version of the arithmetic operation with matching descriptions) should be made. The binding process results in the creation of the metamodel layer. Thus the abstract representation of the addition at the visual layer is implemented by the mapped addition operation (ADD) shown in the metamodel layer.

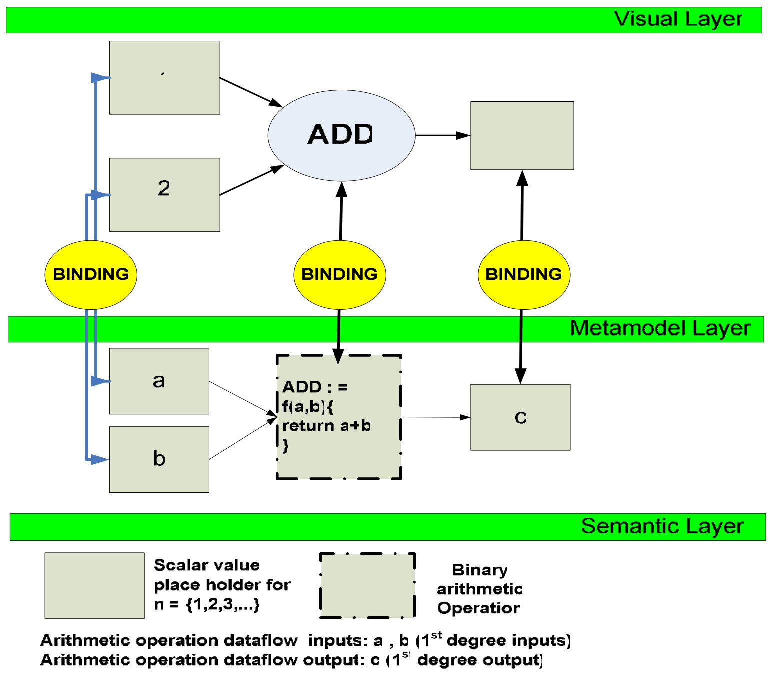

Visual Layer
ADD
2
BINDING
BINDING
BINDING
Metamodel Layer
a
b
ADD : = f(a,b){ return a+b }
c
Semantic Layer
Scalar value place holder for n = {1,2,3,...}
Binary arithmetic Operator
Arithmetic operation dataflow inputs: a , b (1st degree inputs)
Arithmetic operation dataflow output: c (1st degree output)

**Figure 7. 2 Three layers involved in a refinement Process**

2. Once the two layers are established then we can utilise event handlers that manage communication channels between the two layers. When an event that

calls for the execution of the diagram is intercepted the handler can pass the metamodel as a specification to be run. The interpretation of the specification by third party computation tools may require semantic support. For example, automatic processing of the metamodel specifications entails the inference process to comprehend the semantic of dataflow metaphor from given specifications.

$$(\text{Input}\,(a, b) \longrightarrow \text{Output}(c)\,)$$

Due to the scope of this research, the problem of semantic interoperability that is the difficulty for third party clients in interpreting the metamodel layer itself (Heflin & Hendler 2000) remains to be resolved in the future development of SDL.

The refinement activity is a top-down development from an abstract specification that visualises the functions that are required of the diagram at the top level. Reification proceeds from the abstract specifications by mapping abstract visual entities to more concrete resources and by composing functional specifications at the visual layer which will control the mapped resources. The mapped visual entities are not only symbolic representation of concrete resources but also inherit the behavioural aspect of the mapped resources. The multi-layered specification design and reification approach is frequently used in the following two areas:

- Resource binding (e.g. Statistical dataset, dataset metadata, graph object , etc)
- Statistical technique/data operation binding
- Inter-diagram relationship

We will present three working examples that are representative of the types of operations in practice when using SDL diagrams as tools.

**Resource Binding**
Resource binding process creates a metamodel layer that supports the link between a graphical entity and an existing statistical resource such as a dataset table. Statistical resources can be made available via shared file systems or repository services. Suppose that there is a statistical dataset that contains a list of undergraduate students. To model a simple random sampling on the dataset, we can first place a graphical icon which represents an input sampling frame list on a tool window.

The metamodel layer needs to be changed in order to relate the graphical icon to a real dataset. When a graphical icon is placed on the tool window it is attached with handlers that monitor user-generated events such as right-button click event. The event handling can be generalised by the generic interaction model depicted in the sequence diagram in Figure 7.3 shows typical sequences of interactions, that is, the sequence corresponding to all the types of resource bindings ranging from dataset to statistical metadata.

Figure 7.4 shows the steps taken to make the graphical icon to represent an instance of a statistical dataset. The SDL tool support allows the exposition of the mapped dataset by offering addition of two types of dimensionality to the dataset: Statistical metadata (Triple-S) and data structure (express the relationship between the population and the sampled data). The following model timeline in Table 7.1 demonstrates how the visual model and the metamodel are changed due to the types of interactions shown in Figure 7.4 and outlines our implementation strategies in refining executable models.

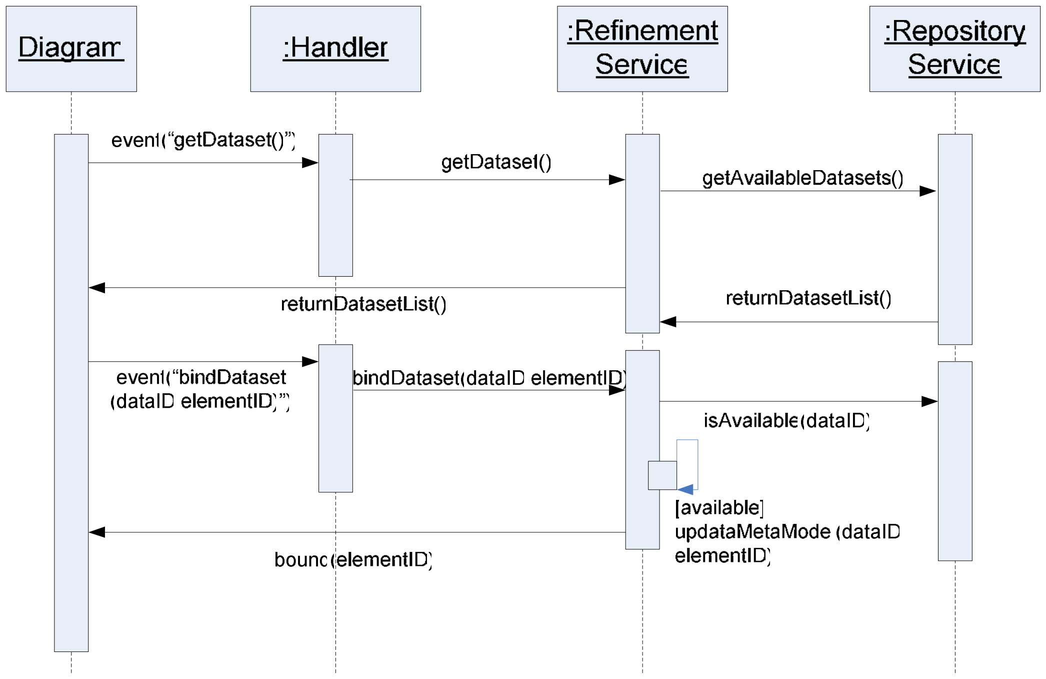
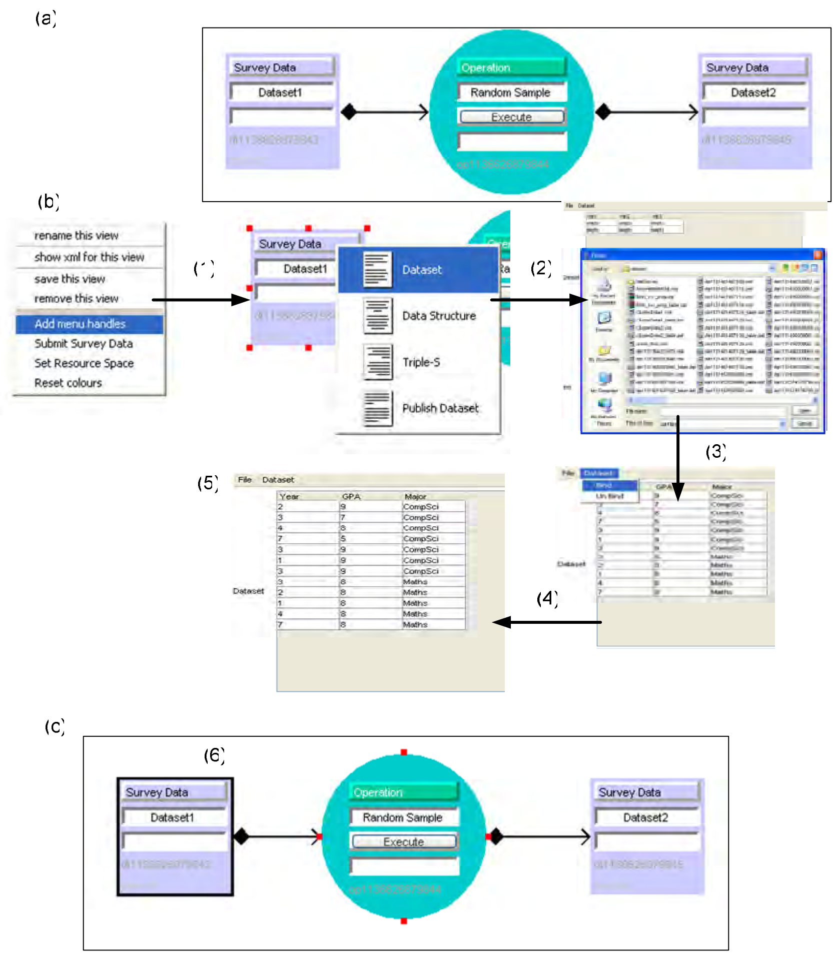


**Figure 7. 3 Typical sequences of interactions in a binding process**

| ***Sequence*** | ***Purpose*** | ***Model*** |
|---|---|---|
| **1** Fig 7.4 (a) | Create a diagram to represent a random sampling process. | Only the visual model layer exists. |
| **2** Fig 7.4 (b,1) | Add handlers to all visual icons. | No Change |
| **3** Fig 7.4 (b,2) | Mapping UIs are generated. | No Change |
| **4** Fig 7.4 (b,3) | Prompt a user to find by viewing the content of a dataset file and bind an appropriate dataset to the visual icon. | Metamodel element to dataset relationship is created. |
| **5** Fig 7.4 (b,4) | Dataset binding is complete. | Metamodel element's data binding attribute is set. |
| **6** Fig 7.4 (c) | The current status of the visual icon is indicated by the change in its blackened layout. | |

**Table 7. 1 Description for each sequence in Figure 7.4**



| Year | GPA | Major |
|---|---|---|
| 2 | 9 | CompSci |
| 3 | 7 | CompSci |
| 4 | 8 | CompSci |
| 7 | 5 | CompSci |
| 3 | 9 | CompSci |
| 1 | 9 | CompSci |
| 3 | 9 | CompSci |
| 3 | 8 | Maths |
| 2 | 8 | Maths |
| 1 | 8 | Maths |
| 4 | 8 | Maths |
| 7 | 8 | Maths |

**Figure 7. 4 Snapshot of the steps taken to make the graphical icon to represent an instance of a statistical dataset.**

## Technique Binding

The mechanism behind the technique binding process resembles the resource binding process in that the metamodel layer is refined iteratively to encode the explicit relationship between a graphical entity and a statistical resource however there are also notable differences between two processes. Statistical resources are by nature static but statistical techniques such as regression test and stratified sampling involve dynamic mathematical activities that require explicit parameters and input and output specifications. Thus at the metamodel level, simple binary relationships that exist in the resource binding processes must be extended to ensure the correctness of the mappings that specify data inputs, outputs and technique parameters of a mapped statistical technique which is to be executed according to the metamodel-driven specification. As we saw in Chapter 4, the mapping between the graphical icon and the technique finds its relational reference in the 'TechToService' relationship type in Chapter 4 Figure 4.5.

In the semantic layer, further information on the topic node is supported by the input/output specification node and the SDL service type. There are three fundamental operations that the SDL tool implements to accomplish the technique binding process.

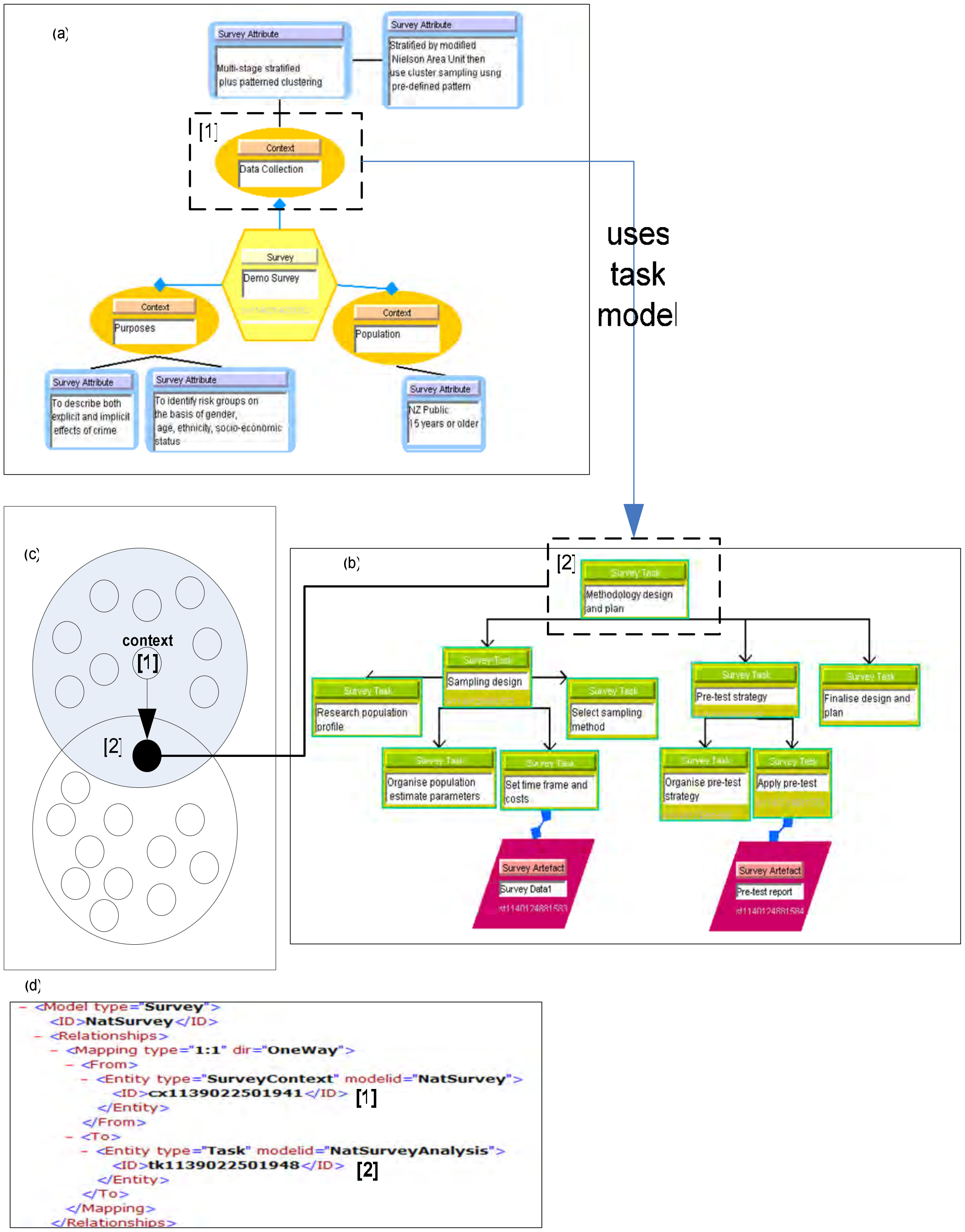
The technique binding operation inquires the technique repository, which is just an XML file with technique metadata entries for prototyping purpose, for a technique which matches the user's requirement. Upon a successful searching, the technique binding operation generates mapping UIs based on the input and output parameter specifications so that the TechToService relationship can be established in a concrete form. The mapped technique provides the body of the actual implementation and any visual icon using the technique must provide an interface that allows necessary mapping operations. The last operation persists the relationship in a metamodel file for the mapped visual icon and may change the visual appearance of the mapped icon accordingly.

#### *Topic Map Occurrences*

Whenever there exist the needs for linking a topic map node to its occurrences, our approach has been to turn a purely visual icon into a working widget and provide the widget with functionalities that can make appropriate the mappings and make use of supporting services such as technique/service repositories.

### Inter-diagram Relationship

Inter-diagram relationships are the glue that holds together all SDL diagrams. As defined in Chapter 4, an inter-diagram relationship is formed when a survey entity is shared by two or more diagrams. Therefore an inter-diagram relationship exists if there is a survey entity that is of interest to more than one diagram. From the perspective of tool users, the shared survey entity is a visual icon in the support tool environment which users wish to model using more than one diagram. Figure 7.5 shows one such example.



```
- <Model type="Survey">
    <ID>NatSurvey</ID>
  - <Relationships>
    - <Mapping type="1:1" dir="OneWay">
      - <From>
        - <Entity type="SurveyContext" modelid="NatSurvey">
            <ID>cx1139022501941</ID>  [1]
          </Entity>
        </From>
      - <To>
        - <Entity type="Task" modelid="NatSurveyAnalysis">
            <ID>tk1139022501948</ID>  [2]
          </Entity>
        </To>
      </Mapping>
    </Relationships>
```

**Figure 7. 5 Inter-diagram relationship between the survey diagram (a) and the task diagram (b) (c) shared survey entity between the diagrams shown in the Venn diagram (black oval shape) (d) a snippet of metamodel file showing the mapping between [1] and [2]**

In Figure 7.5, the survey diagram's context 'Data Collection' uses the task model shown in Figure 7.5 (b). Thus there is an inter-diagram relationship between those two diagrams. The Venn diagram in Figure 7.5 (c) shows that the task model should exist in two conceptual spaces.

At the metamodel layer, the relationships are encoded as mappings between two survey entities identified by unique identification numbers of participating survey entities. The XML file in Figure 7.5 (d) shows a snippet of such mappings at the metamodel layer.

Integrity rules and diagram cardinalities are managed by specification refinement service's utility objects which are called by handlers via the event processing model shown in Figure 6.2 of Chapter 6. The problems related to the preservation of inter-diagram integrity pose complex implementation challenges and deserve a specialised research especially in the distributed authoring environment (Grundy, Hosking, and Mugridge 1998). The prototype tool only implements a simplistic management scheme that relies on the diagram cardinality model to prevent the potential sources of inconsistencies.

## *7.4 Diagram Execution and Back-end Integration*

**Problem**
We need to execute diagrams (statistical data and technique) and use SDL diagrams as tools.

**Context**
Assuming that all required diagrams have their graphical entities mapped to appropriate resources and techniques, we need to investigate ways to execute the diagrams. One of the core design principles for SDL tool support is constructing a software architecture that avoids the tight coupling between computation package specific functionalities and the visual environment for survey users, this permits an extensible and scalable framework built on the expressiveness of SDL to integrate the survey process activities while providing both logical and physical separation from underlying tool/platform specific implementations. From the standpoint of systems architecture the design approach is in the form of a *Service-Oriented Architecture* (SOA). Due to the scope and limited time the following implementation options are not exhaustive and but they reflect a wide spectrum of solutions: ranging from legacy computing practice in the survey process to an SOA inspired approach:

1. Generated specifications are software and platform specific. The specifications are sent out to computation packages or batch processing files are created form the specifications when results are received and they are reflected at the visual layer. One obvious advantage is the simplicity of this solution is that the computing environment is largely monolithic thus the platform lock-on is not a major concern. One interesting implication of this solution is that the model-based service generation from SDL diagrams is designed for a ubiquitous web service hence an artificial split within the nature of the tool exits: Back-end lock-on and universal service generation.

2. Generated specifications are universal and cross-platform. Transformation schemas are applied to the specifications before they are run on back-end computing packages. The integration layer consists of proxy objects that establish communication channels between the SDL tool environment and computing packages such as R and S-Plus. The proxy object can be viewed as a service interface which has its implementation within the chosen computing package. This approach introduces no tool/platform specific design elements and produces a loosely coupled solution platform.

3. The last option is an evolution of the second option. The second approach yields a loosely coupled middle layer but the solution does not consider interactions between service providers and consumers within the context of the survey process. The encapsulation afforded by the middle layer is a large advantage, however as the number of services offered by providers grow (this is expected as SDL tool support enables a model-driven web service generation.) we encounter the problem of finding the right service unless the offered services exist in relationships. Survey techniques can be classified by purpose (most frequent), underlying theory, etc. Provided services need to be accompanied by self-describing descriptions so that consumers know how and when to use them. Therefore the inclusion of registry services to the previous architecture will be a natural progression toward making the back-end service discovery coherent to consumers.

**Implementation Solution**
Our implementation for the diagram execution and back-end integration entails the following process:

1. Specification processing. The core specification defines all techniques and data flows.
2. Generate proxy objects for techniques and a sequential workflow chain is established. Either full or partial execution of the chain is possible.
3. The proxy object maintains a communication channel over TCP/IP to computing packages (in our case R) and actual code is generated at run-time or uses an existing service obtained via the service registry and delivered to the computing packages.

Figure 7.6 shows a structural view of how diagram execution and back-end integration are implemented.

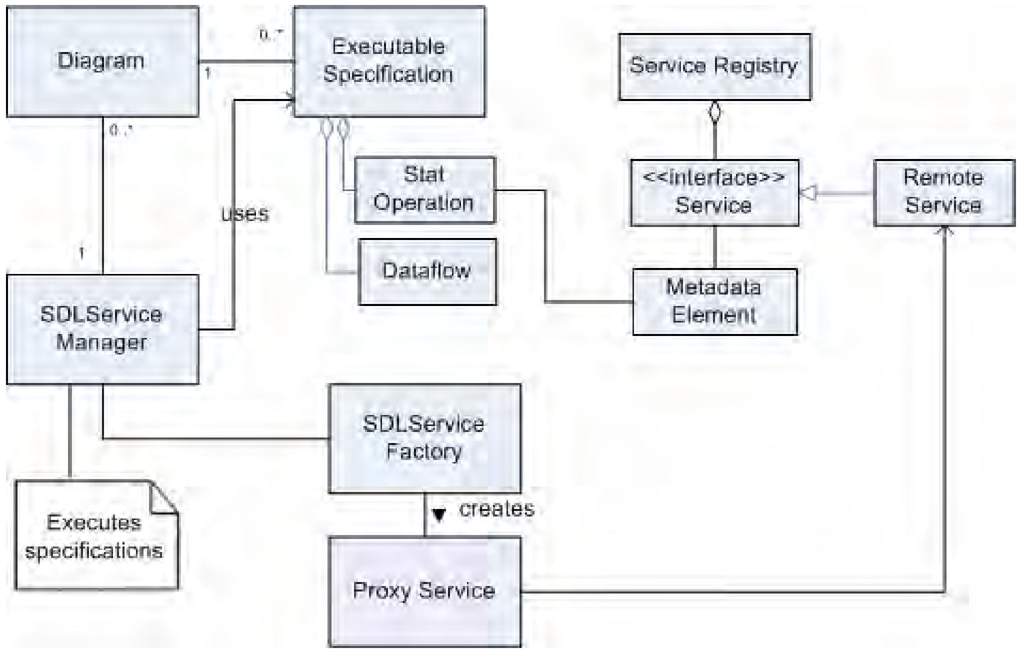


**Figure 7. 6 Structural view of how diagram execution and back-end integration**

## *7.5 Model-based service generation*

**Problem**
External services can bind to the visual entities of SDL diagrams and users interact with the services using the bound entity's user interaction mechanism. When the binding process is completed, the transition from being a service consumer to being a service provider can be made.

**Context**
A truly portable and cross-platform development environment should offer a multiplicity in how users interact with the back-end system. This means that users can utilise techniques based on diagrammatic specifications outside of the proof-of-concept tool environment.

The external interaction should be subject to the same operational behaviours as are expressed by SDL diagrams but allow client heterogeneity (e.g. from Pounamu to web-based application) on which the service will be invoked and consumed. Survey researcher might want to distribute a newly developed statistical technique to the peers or provide a newly developed technique as a general service which can be utilised by myriads of clients. In practice the distribution of statistical techniques is usually by document or in the form of raw code. There are several reasons why this distribution scheme has detrimental effects on management and operational overheads. First, users may not be able to set up a required software package if the code is application-specific. Secondly, users may not always have full knowledge of associated metadata that should be considered to make meaningful inferences.

Thirdly, the provided source and user's platform may be heterogeneous hence users may not be able to load the code and execute locally.

**Implementation solution**
From a service consumer's point of view, the service generated from a SDL model should be independent of the particular type of platform and undistinguishable from other generic software services. Hence a web service is the only commonly available solution to fulfil this requirement. A web service based solution should support two additional operations: a discovery operation that allows a service to be explored by users and a service execution operation that runs a requested service on the server side.

The full implementation of the discovery operation is out of scope for our research but it is supported by allowing users to access SDL metamodels and Pounamu model files. For a completely visual approach users may opt to render Pounamu files client-side then access metamodels which are based on the rendered visual diagram (REF). As with the specification generation, through processing SDL metamodels in conjunction with information from the semantic layer users may obtain as much as insight into available services.

**From a service provider's view**
The current proof-of-concept tool only turns a survey technique diagram into a web service. Survey technique diagrams model statistical computations and operation flows that are very susceptible to heterogeneity of the code that is needed to execute required operations. The susceptibility can be seen in practice as the computational tasks are defined in software package specific code. Another reason for only producing technique diagram-based web services is that statistical techniques fit the aspect of the survey process which is the most convenient to a service oriented architectural concept within limited time and resource constraints. The convenience comes since each technique diagram represents an independent, stateless and reusable set of atomic statistical techniques within the context of a single activity without orchestration overheads that are inherent in behavioural diagram types such as UML activity diagrams and SDL process diagrams. Regarding the orchestration of produced services, there is plenty of research on visual composition and orchestration of services (Liu and Hosking and Grundy 2005, Pautasso and Alonso 2003) hence we focused on the more practical problem of allowing survey researchers to publish their techniques to a wider audience across heterogeneous platforms and representing the model of orchestration not implementation of service orchestration.

**Service generation mechanism**
The first step in implementing the service generation is to create a diagram processing sequence. The implementation of the process looks into the underlying diagram structure for executions points according to the diagram's syntactic rules.

Once the sequence is complete, metamodel files bound to visual icons are accessed by the service specification generator to create web service specifications. The specifications file encapsulates the following:

- Dataflows
- Techniques (identified by unique identifiers and types)
- Parameters assigned to techniques

Once the specification file is ready then the code to execute the techniques according to the defined sequence in the specification is generated. The code generation process can be simplified as shown in Figure 7.7

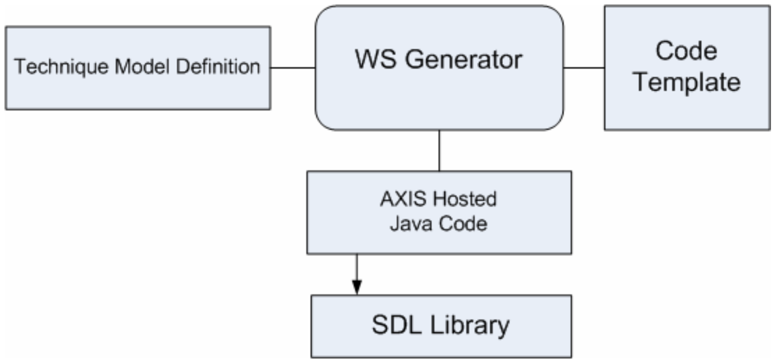


**Figure 7. 7 Web service code generation from diagrammatic specifications**

The generation process starts with a technique model definition which describes the data flow from one technique to another, types of techniques used, expected outputs, etc. Technique entities play a central role in code generation. Each technique is evaluated in the order of its assigned execution index which starts from 1 and then the code template is used to build up code in Java.

The code generation process entails no complex operations due to the autonomous nature of all techniques present in the survey technique diagram. Thus a simple template driven sequence can be iterated to from the first technique until the last one.

The web service code which is a front for requested back-end operations is hosted by Apache Axis. Due to the simple nature of the code generation pattern, mapping the specifications to alternative hosts should incur no additional technical difficulties.

Note that the model based service generation that we have discussed in this section does not achieve complete survey process orchestration for users but remove barriers to publish survey techniques which are utilised during the survey process. Along with the tool support for dataset distribution and sharing, the service generation constitutes the core of the survey resource management architecture that will be presented in the next section.

## 7.6 Survey resources and information management

In Chapter 2's section on statistical data and metadata for surveys, the importance of sharing and disseminating resources created during the course of the survey process was discussed. In this section we focus our discussion on statistical datasets and SDL tool support. We will begin with a brief introduction of how meta-models and datasets are made available to users.

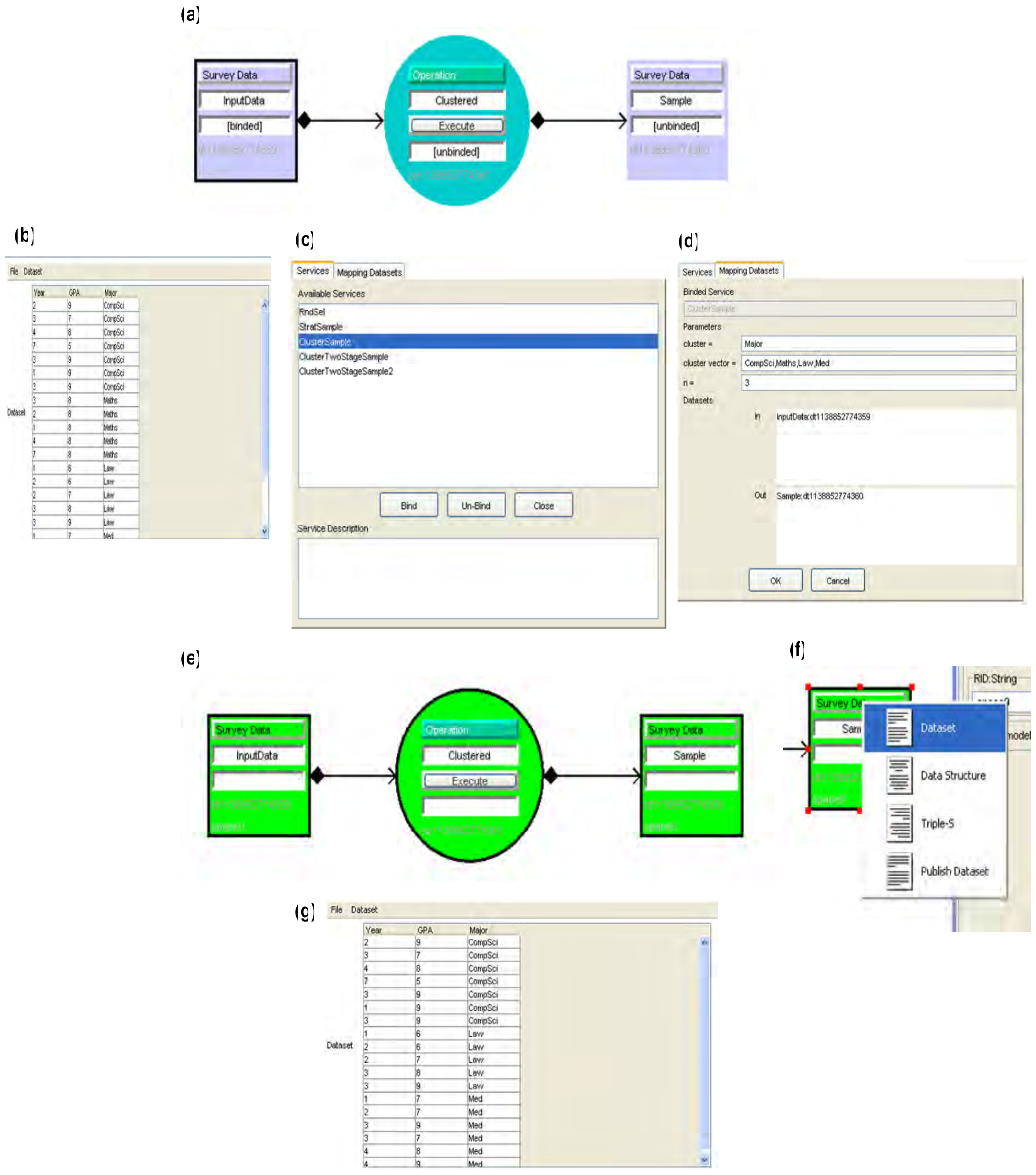
**Publishing Datasets, dataset metadata and Meta-Models (survey resources)**

The publishing of datasets, dataset metadata and meta-models permits survey users to examine datasets and meta-models and available survey resources. Any type of resource publishing should ideally consider a restrictive scheme to keep the resources where they should be used and share the resource with the right group of users. Our prototype tool support does not however cover the area of security; implementation of the security features is left for future development of the tool.

**Statistical Metadata Tool Support**

**Problem**
The correct analytical use of statistical data may require access to statistical metadata such as the history of how the data was obtained.

**Context**
Working with both survey data and technique diagrams produces statistical data. For instance the survey data diagram in Figure 7.8 (a) produces the dataset "Sample" as the result of a clustered sampling. Let us assume that the dataset is published so that it is made available to users. To utilise the statistical data in any meaningful way, we need not just the raw dataset in tabular form. To begin analysis, we need the population of the primary sampling unit (psu) and the size of psu. Sampled dataset without such metadata can be virtually useless for any serious study. Sampled data should be treated very differently from the concept of data in business computing where observational datasets do not need to be accompanied by such metadata as statistical databases violate the closed complete world assumption (Sato 1991). Similar metadata requirements can be extended to datasets which are produced as outcomes of statistical techniques. Statistical data specific features, which introduce significant differences at a number of aspects (Grossmann 2003), as can be in Figure 7.8. The execution of the diagram in Figure 7.8 (a) produces the dataset as shown in Figure 7.8 (g). For the dataset to be useful for analytical techniques at later stages, some of the information visualised in Figure 7.8 (c) and (d) should be kept with the dataset. The dataset should have the contextual metadata for sampling type, sampling parameters, input dataset size, chosen cluster, sampling size and cluster vector.



File Dataset

Dataset

| Year | GPA | Major |
|---|---|---|
| 2 | 9 | CompSci |
| 3 | 7 | CompSci |
| 4 | 8 | CompSci |
| 7 | 5 | CompSci |
| 3 | 9 | CompSci |
| 1 | 9 | CompSci |
| 3 | 9 | CompSci |
| 3 | 8 | Maths |
| 2 | 8 | Maths |
| 1 | 8 | Maths |
| 4 | 8 | Maths |
| 7 | 8 | Maths |
| 1 | 6 | Law |
| 2 | 6 | Law |
| 2 | 7 | Law |
| 3 | 8 | Law |
| 3 | 9 | Law |
| 1 | 7 | Med |

Services | Mapping Datasets

Binded Service: ClusterSample

Parameters

cluster = Major

cluster vector = CompSci,Maths,Law,Med

n = 3

Datasets

In InputData:dt1138852774359

Out Sample:dt1138852774360

OK Cancel

File Dataset

Dataset

| Year | GPA | Major |
|---|---|---|
| 2 | 9 | CompSci |
| 3 | 7 | CompSci |
| 4 | 8 | CompSci |
| 7 | 5 | CompSci |
| 3 | 9 | CompSci |
| 1 | 9 | CompSci |
| 3 | 9 | CompSci |
| 1 | 6 | Law |
| 2 | 6 | Law |
| 2 | 7 | Law |
| 3 | 8 | Law |
| 3 | 9 | Law |
| 1 | 7 | Med |
| 2 | 7 | Med |
| 3 | 9 | Med |
| 3 | 7 | Med |
| 4 | 8 | Med |
| 4 | 9 | Med |

**Figure 7. 8**
**(a) Survey data diagram to model a cluster sampling method.**
**(b) Showing the mapped dataset for the input data icon.**
**(c) Sampling method mapping**
**(d) Mapping interface to enter required parameters.**
**(e) Diagram execution**
**(f) Opening the newly mapped data icon.**

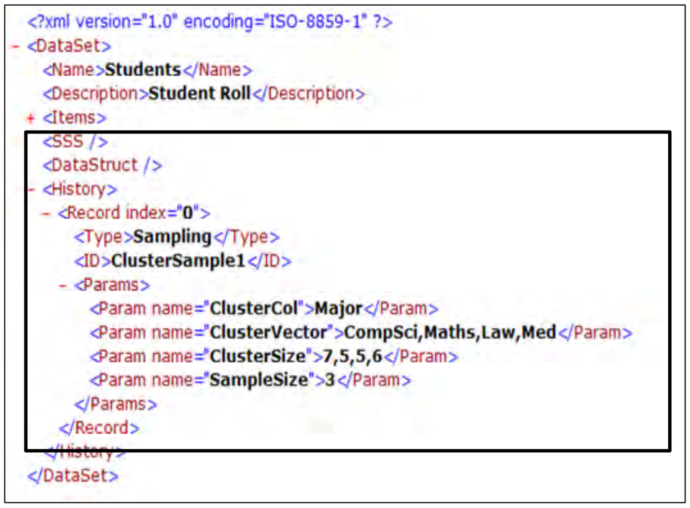
**Implementation solution**

The solution to the statistical metadata problem is addressed in two areas:

- Metamodel structure – preserving associated metadata together with data
- Additional user interface to explore the data metamodel of data easily.

Based on the layered approach of modelling the survey process, dataset shapes (visual icons) have physically separated metamodels and as described in the previous section. As shown in Figure 7.9, the corresponding metamodel for the dataset icon may have three types of relationships: Triple-S (SSS) metadata (History), data structure (DataStruct) and dataset (Items).

When the data creation process a statistical dataset should be captured as shown in Figure 7.9, metamodel attributes can wrap a dataset's metamodel entity with references to the creation process. The following metamodel file demonstrates how the metamodel layer implements the proposed solution using metamodel attributes which are shown in the rectangle area of Figure 7.9.

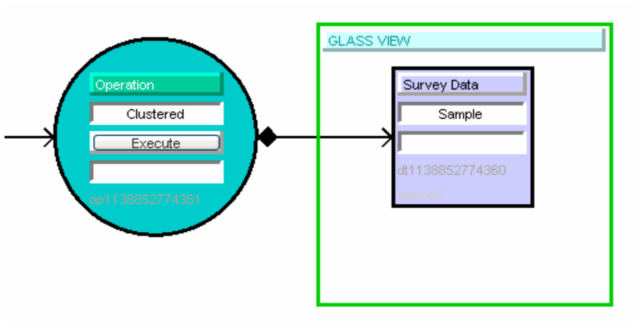


```
  <?xml version="1.0" encoding="ISO-8859-1" ?>
- <DataSet>
    <Name>Students</Name>
    <Description>Student Roll</Description>
  + <Items>
    <SSS />
    <DataStruct />
  - <History>
    - <Record index="0">
        <Type>Sampling</Type>
        <ID>ClusterSample1</ID>
      - <Params>
          <Param name="ClusterCol">Major</Param>
          <Param name="ClusterVector">CompSci,Maths,Law,Med</Param>
          <Param name="ClusterSize">7,5,5,6</Param>
          <Param name="SampleSize">3</Param>
        </Params>
      </Record>
    </History>
  </DataSet>
```

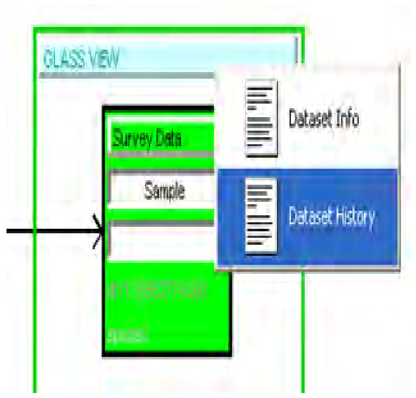
**Figure 7. 9 Added metadata attributes to preserve the history of data**

Having described the metamodel side of the solution we demonstrate how the tool support for the metamodel layer can be implemented. Once the metamodel is in place, it becomes relatively simple to add a new user interface and to quickly explore the dataset metadata. Amongst many user interface refinements that have been tested, the implementation of a transparent overlapping interface layer, which sits on the top of a visual icon had favourable feedback in presenting underlying metadata seamlessly. The interface in action is shown in Figure 7.10.

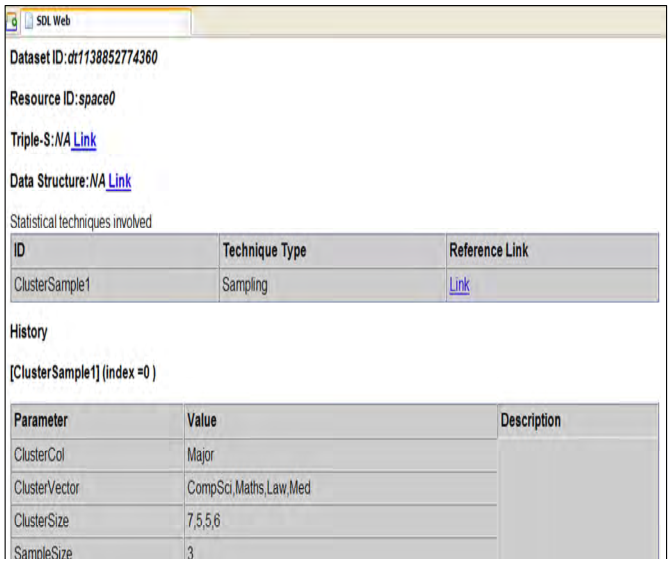
(a)



(b)



(c)


SDL Web

**Dataset ID:***dt1138852774360*

**Resource ID:***space0*

**Triple-S:***NA* Link

**Data Structure:***NA* Link

Statistical techniques involved

| ID | Technique Type | Reference Link |
|---|---|---|
| ClusterSample1 | Sampling | Link |

**History**

**[ClusterSample1] (index =0 )**

| Parameter | Value | Description |
|---|---|---|
| ClusterCol | Major | |
| ClusterVector | CompSci,Maths,Law,Med | |
| ClusterSize | 7,5,5,6 | |
| SampleSize | 3 | |

**Figure 7. 10**
**A transparent panel is place over a data icon.**
**Contextual menu of the transparent interface is invoked to obtain the dataset's history.**
**Document is generated to show how the dataset has been created. Details of sampling method and parameters are also displayed.**

As a simple UI component, the transparent panel represents a portal to the underlying metamodel information of the visual icon which resides within panel's area. The transparent panel obtains a reference to the visual icon's metamodel and may use its attributes for displaying dataset history records and generating metamodel-driven documents.

## 7.7 Summary

We took throughout this chapter a consistent discussion format that discoursed problems, contextual information and implementations to support our design decisions for our prototype tool coherently. The various design elements of the proof-of-concept tool were organised by five core functional areas: diagram authoring support, specification refinement process, diagram execution and model-based service generation. Research outcomes of this chapter cover diverse aspects of the survey process from visual modelling to information dissemination that are often neglected by the narrow scope of existing statistical packages.

Practical implications of the outcomes bring new perspectives into how statistical surveys are planned and managed and we will demonstrate the prototype tool in a realistic case study in the next chapter.

# Chapter 8 Case Study: Survey of Crime Victims

## *8.1 Introduction*

In this chapter, we present a case study where the survey process and its analytical components are simplified to a great extend, but the context of the case study raises various important aspects of statistical surveys which are frequently encountered. The case study follows the broad theme of the New Zealand National Survey of Crime Victims in 2001 (NZCS 2003). It should be noted that statistical data and methodologies presented in this case study are not the faithful duplicates of the national survey's.

## *8.2 Case Study Overview*

The main purpose of this case study is to show users how the method and approach presented in the previous chapters are used to model statistical surveys. The use of the existing survey as a basic template gives the case study a meaningful context. As we are bounded by both time and scope, the content of the case study will be largely task oriented that is the survey process is presented in terms of general tasks such as data analysis and survey management. The tasks are listed below by section.

8.3 Survey overview
8.4 Survey tasks
8.5 Statistical data, sampling and analytical techniques
8.6 Survey process
8.7 Survey information dissemination

## *8.3 Survey Overview*

The primary source of victimisation data is obtained from law enforcement agencies. The reality of victimisation may not be sufficiently captured from the one source therefore a statistical survey is planned to investigate the state of victimisation in the following three areas:

- Identify at risk groups.
- Provide an alternative measure of crime victimisation
- Describe both explicit and implicit effects of crime.

The survey is a door-to-door survey targeted at New Zealanders who are aged 15 or more. Each region is assigned to a unique area code, and a stratification of the area codes will be done. Random samples of strata will compiled, in the selected stratum a pre-defined visit pattern (e.g. every 10th street and every 10th house on the selected streets) will be applied in collection data. Figure 8.1 shows how the survey information above and more is depicted using a survey diagram. Notice how initial survey attributes which are clustered around the survey contexts spawn associated attributes that visualise consensus making processes in formulating the high-level nature of the survey.

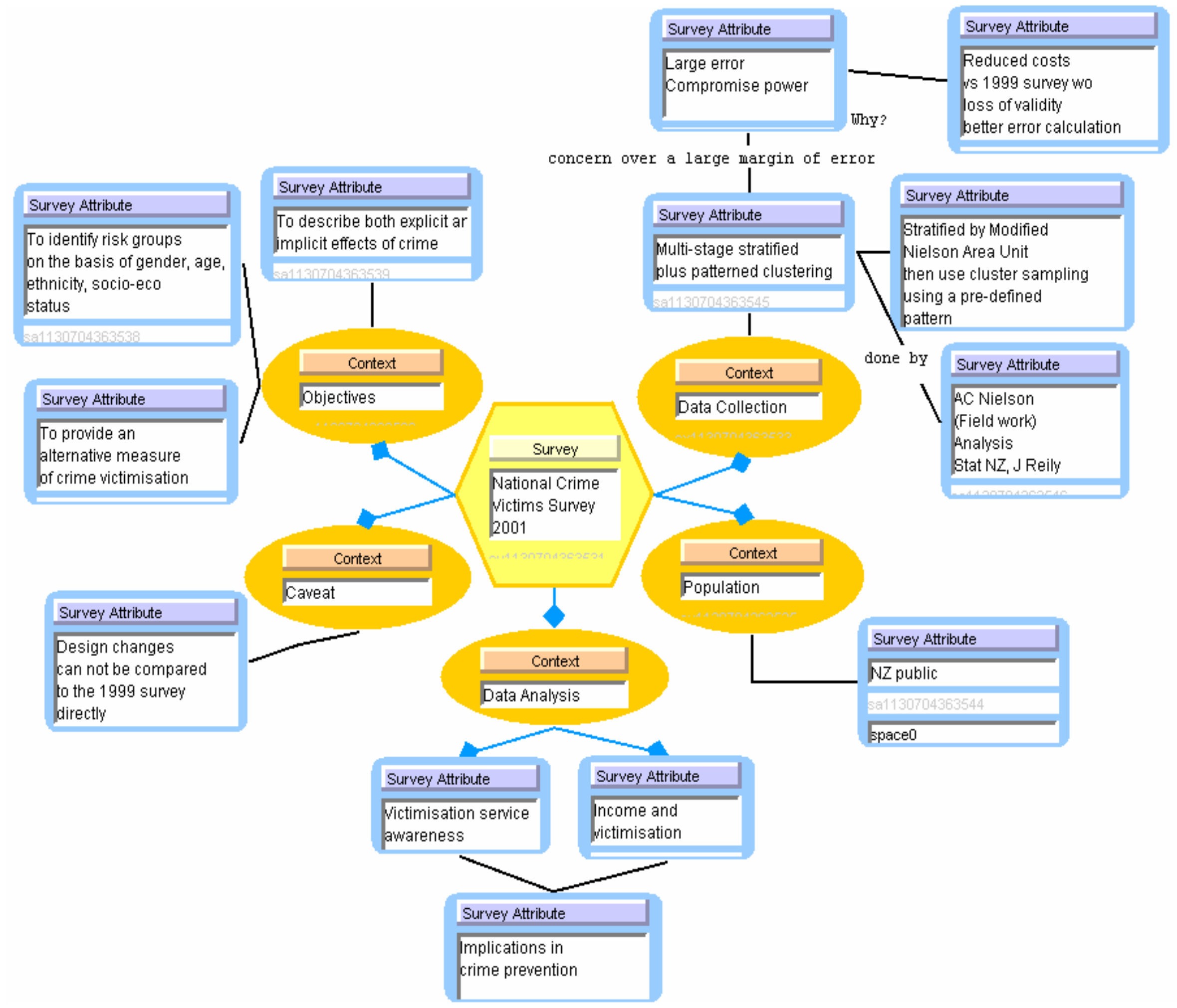
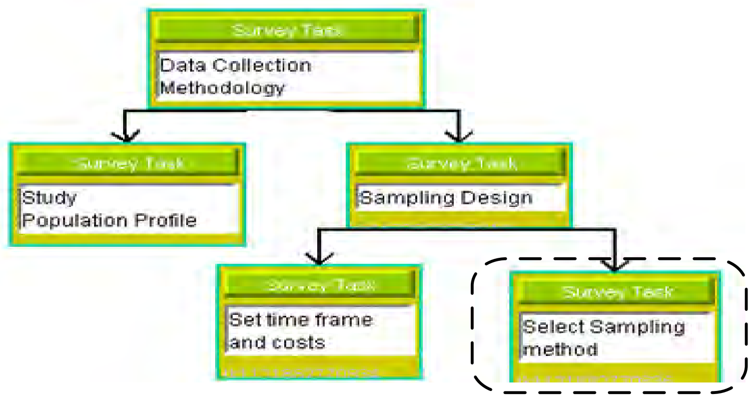


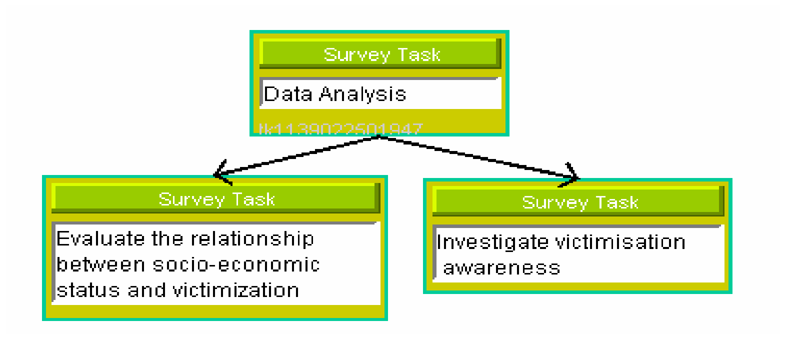
**Figure 8. 1 Survey diagram for the victimisation survey**

## 8.4 Survey Tasks

Each context of the survey is supported by survey tasks by definition. Two tasks will be presented in this section. The first task defined to model the sampling design which is a part of the data collection context in Figure 8.1. The sampling design has the closely relevant methodology design and plan survey pattern. Our approach here will be to reuse the pattern and adapt it for our requirements. Figure 8.2(a) shows the simplified version of the pattern. The dotted survey task is directly related to our current task and we will attach the survey diagram, which models the design process, to the survey task in the next section.



(a) Data collection


Survey Task
Data Analysis
Survey Task
Evaluate the relationship between socio-economic status and victimization
Survey Task
Investigate victimisation awareness

(b) Data Analysis

**Figure 8. 2 Two task diagrams for the case study**

The second task is to design and apply statistical techniques to investigate some of the risk factors and how the existing victim support systems are utilised by people. Figure 8.2 (b) shows the task model for the second task. The design of the task model will be refined further in the next section as we develop a sampling method and statistical techniques to support the tasks.

## 8.5 Sampling and Analytical Techniques

It is not feasible in most circumstances to implement all aspects of the data collection stage in SDL models. Survey activities such as a door-to-door visit is still a very implicitly done task which is hard to model in practice. So when we deal with the data collection stage, our main concern is to express the specifications of sampling techniques used in the survey process and types of statistical metadata related to collected statistical data. Referring to Figure 8.1, the chosen sampling strategy consists of two stages: The sampling fame is stratified in two stages by the modified area unit (Figure 8.3 (1)) then household visits are planned according to the patterned clustering (Figure 8.3 (2)). The specification of the sampling method is captured as shown in Figure 8.3. A mock statistical dataset can be bound to the 'Data Frame' icon for pre-testing purposes. In the post data collection stage, collected statistical data can be binded to 'Sample' icon in Figure 8.3. Associated statistical metadata such as a Triple-S file for the dataset and the structural information on the dataset can also be bound to the 'Sample' icon thus giving users a unified description of the collected data.

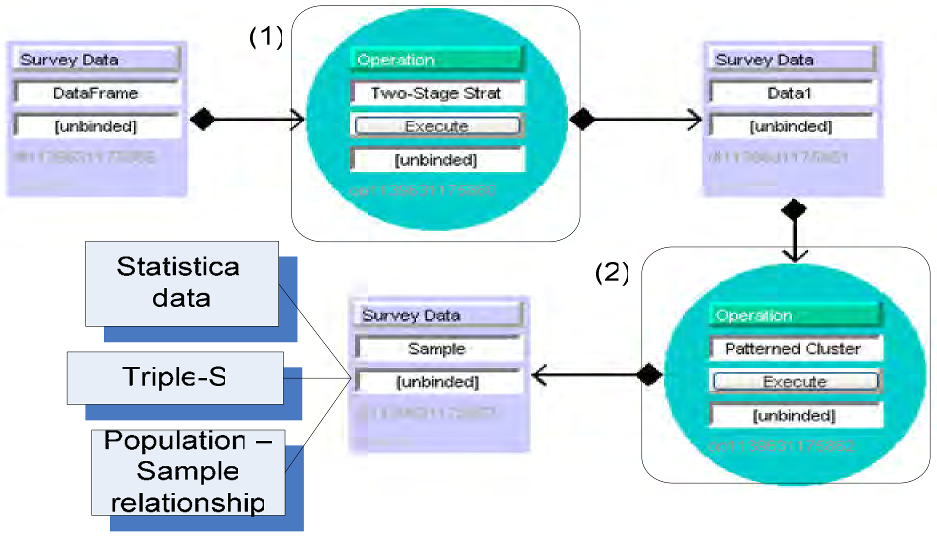


**Figure 8. 3 Sampling method used for the case study**

The analytical component of the survey process can be more intimate than other aspects of the survey process for tool users, since they would not only compose the specifications for statistical techniques but also execute and export them as they explore the collected data. In our case, two of the goals in the data analysis stage are:

1. Investigate the relationship between socio economic status and victimisation.
2. Investigate the public's awareness of victim support services and self-assessed safety.

The collected data is published into the shared repository as shown in Chapter 7 section 7.6. The statistical techniques then map the data into their data icons to construct data flows into statistical techniques. In this case study, we utilise visualisation methods to assess whether there is evidence of a statistical association between data variables. Statistical techniques used for the both problems are shown in Figure 8.4. The two techniques are composed in the same diagram as they feed on the same statistical data. Then the diagram is separately executed to produce the boxplot for the socio economic status and victimisation relationship (Figure 8.5(2)) and the multivariate plot to visualise the public awareness (Figure 8.5(3)). When satisfactory results have been obtained the diagram is put into the model repository and can be exported in the form of a web service. Prior to the next section of the survey process modelling, the newly composed survey data  and technique diagrams are put into the context of the task models by creating survey artefacts for the task models and mappings them onto the diagrams as shown in Figure 8.4.

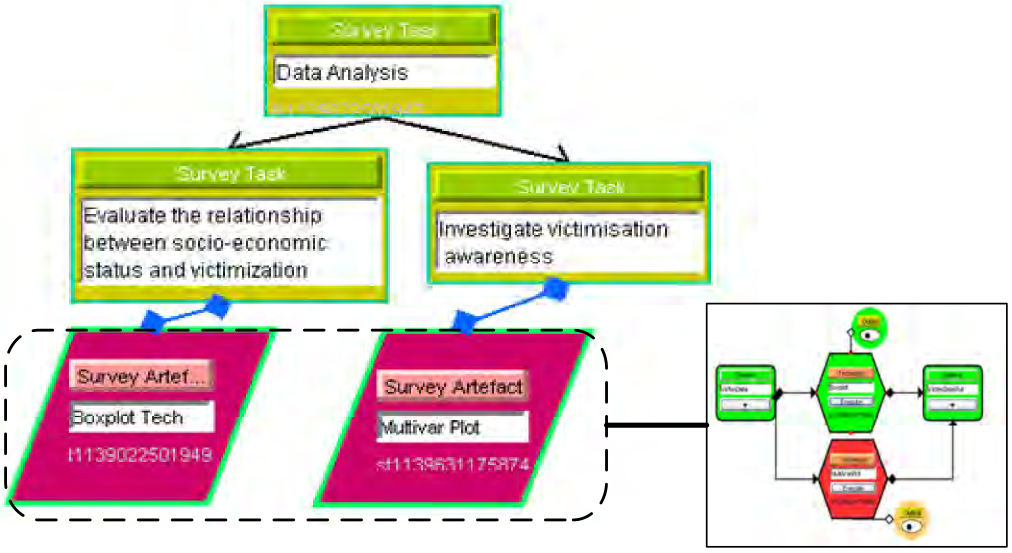


Figure 8. 4 Survey artefacts to technique mappings

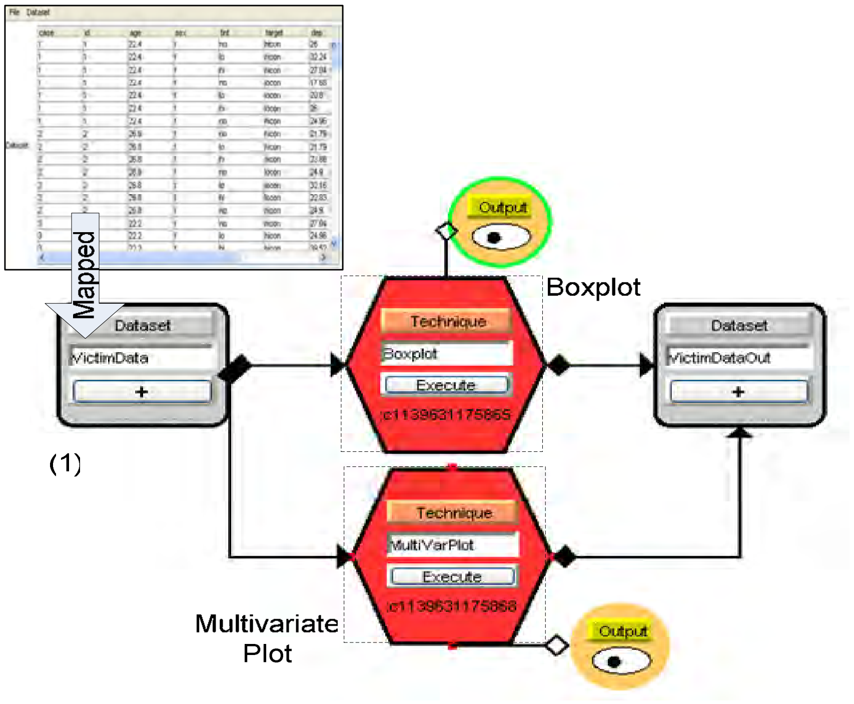
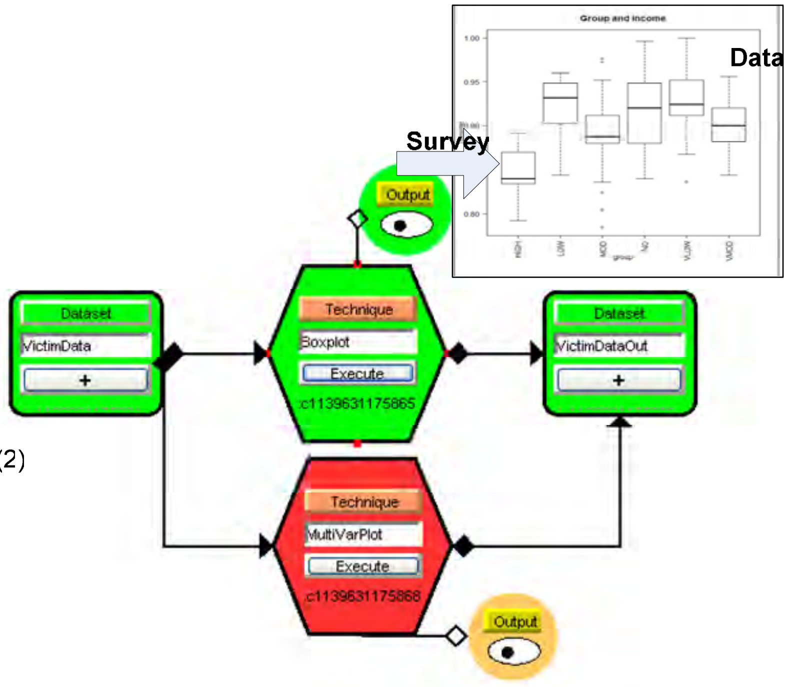


Figure 8. 5 Survey technique diagram for the two statistical techniques

(2)



(3)

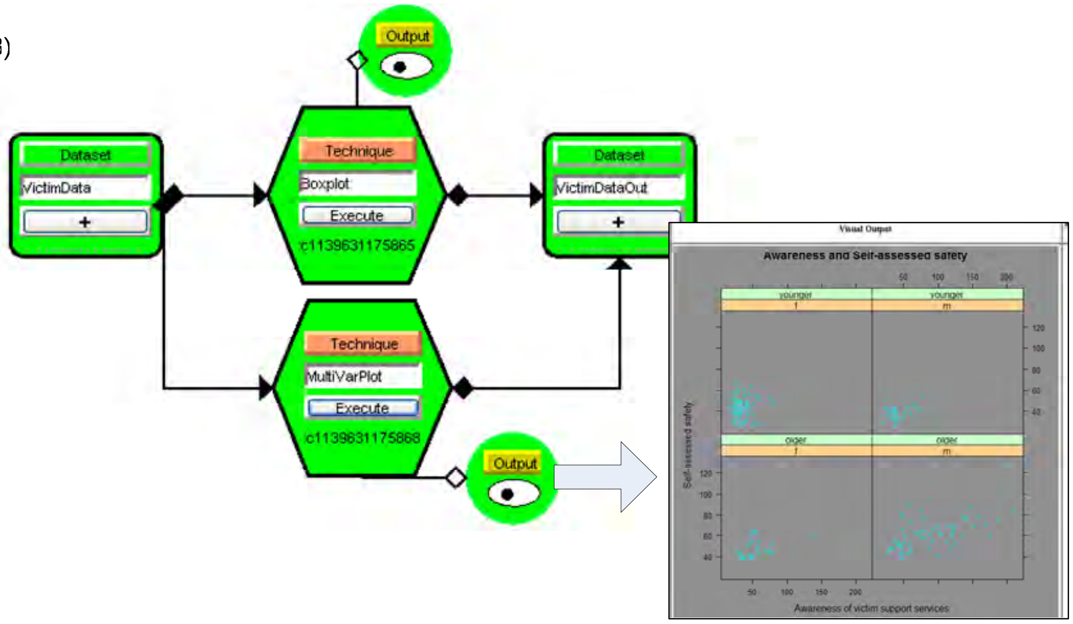


**Figure 8. 6 Survey technique diagram for the two statistical techniques**

## *8.6 Survey Process*

The completion of the task models means that we are able to access the survey tasks which are mandatory for modelling the survey process. Figure 8.2 shows such tasks and Figure 8.4 shows how the models are refined to glue survey and technique diagrams into the context. As described in Chapter 3 and 4, a survey task is a building block for the survey process. Each survey process stage is in fact an interaction between survey tasks. Let us examine the first two stages of our survey based on the two task models presented in the previous sections.

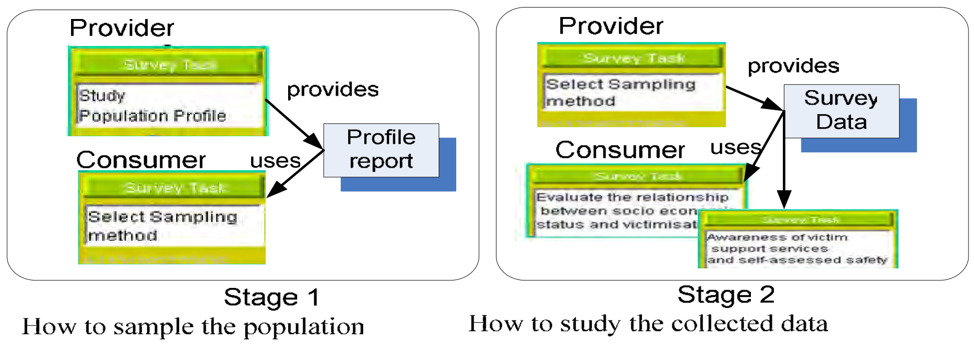


**Figure 8. 7 Task interactions in the first two stages of the case survey**

The top layer process diagram in Figure 8.8(a) combines the provider and consumer relationships and the sequential process flow between the two stages are in Figure 8.8(b).

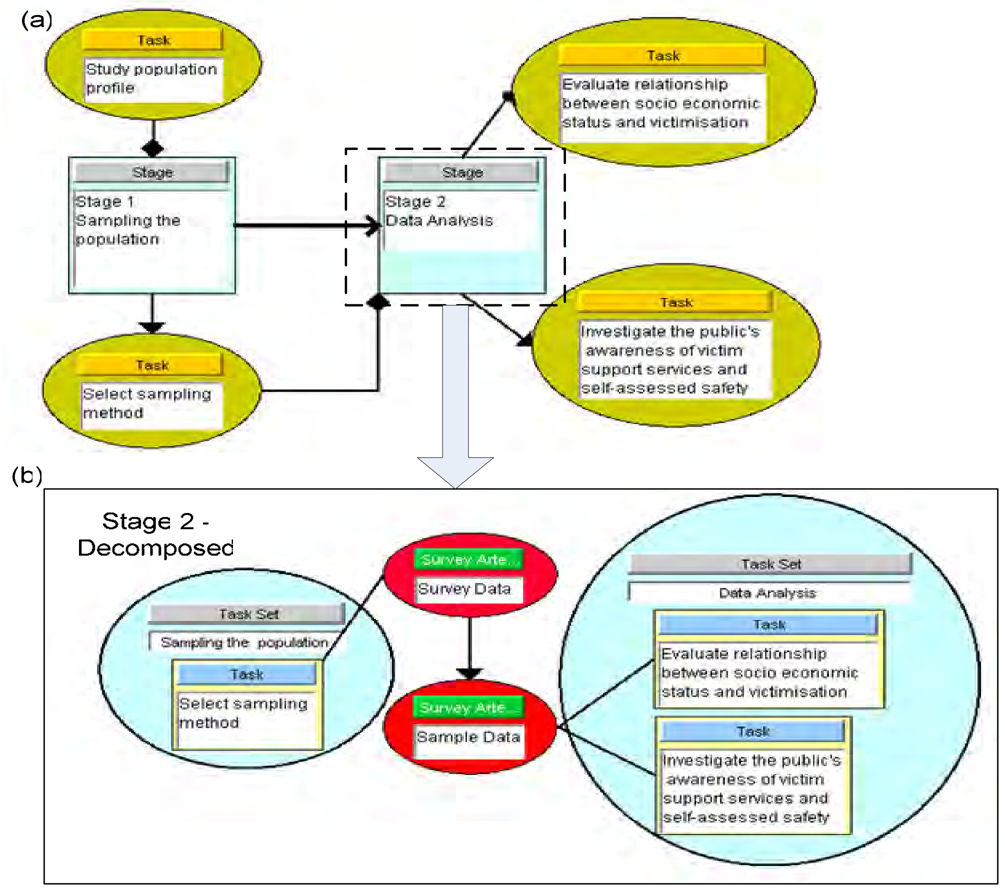

(a)
Task
Study population profile
Stage
Stage 1
Sampling the population
Task
Select sampling method
Stage
Stage 2
Data Analysis
Task
Evaluate relationship between socio economic status and victimisation
Task
Investigate the public's awareness of victim support services and self-assessed safety
(b)
Stage 2 -
Decomposed
Task Set
Sampling the population
Task
Select sampling method
Survey Arte...
Survey Data
Survey Arte...
Sample Data
Task Set
Data Analysis
Task
Evaluate relationship between socio economic status and victimisation
Task
Investigate the public's awareness of victim support services and self-assessed safety

**Figure 8. 8 Survey process diagram for the case survey**

The second layer illustrates that how the outcome of the stage 1 (Survey data) is mapped to the sample data which are utilised by the two statistical techniques. From the stage 2's view, upon completion of the stage 1 the sampled survey data must be made available to the subsequent analytical techniques. Thus the artefact-to-artefact mapping represented by the oval shapes visualise the contractual condition.

## *8.7 Survey Information Dissemination*

In Chapter 4 and 6 and the last six sections of this chapter we discussed the acquisition of survey information based on the metamodels. In this section, we focus on the practical aspects of the metamodel based survey information from end user's view. The starting point of our discussion is set to the survey technique diagram shown in Figure 8.5 to limit our scope. Imaging a user who just accessed the metamodel behind the survey technique diagram. At the metamodel layer, the diagram tells us the following:

- Input data
- Techniques used (Boxplot and multivariate plot)
- Results
- Operation flows

Backtracking from the input data, users can gather where the data is from, how the data is obtained and associated metadata for the dataset from the survey data diagram's (Figure 8.3) metamodels. Recursive tracking back to the sampling methods also reveal the details about the methods. For instance, the victim data used in the technique diagram in 8.5 is traced back to the Sample dataset in Figure 8.3 and the type of the sampling method 'Two-Stage Stratified' provides link to supporting information for the sampling method. From inter-diagram relationships of the metamodels, the trace can be extended readily back to the task diagram in Figure 8.2 then to the survey diagram in Figure 8.1. Therefore the survey technique diagram can be put into the context of the survey process by tracing back to mapped survey entities that also reside in other diagrams at the visual layer.

The reverse of the above example is also equally true that once all functional requirements of the survey are modelled then we can start form the survey diagram and find out all mapped tasks. A mapped task's survey artefacts point to all relevant statistical data and techniques. Therefore provided that there are no inconsistencies in inter-diagram mappings, even the small part of the metamodel such as the metamodel element, which corresponds to the input data in Figure 8.5, can lead users to the high-level contextual information. In reverse the high-level metamodels derived from the survey diagram as shown in 8.1 can also unveil low-level details that are used to execute diagrammatic specifications. The disseminated survey information is not just in the forms of generated documents or metamodel-based presentations but users may participate in the actual execution of the techniques from their own development environment to test and verify them. They also have an option to use the available techniques in their own applications without investing in computing facilities.
No longer is survey information is hidden in static reporting focused documentations or statistical package specific code but shared in an open space.

## 8.8 Summary

In this chapter, we illustrated the use of the proof-of-concept tool by following a walkthrough designed around a real-life survey (NZCS 2003). The core functional areas of the tool were discussed in the context of the real-life survey.

In the final section, particular focus was on how users can benefit from statistical surveys designed and managed by our tool after the survey process is completed.

A similar survey scenario will be explored in the next chapter but in the environment of formal user testing.

# Chapter 9 Evaluation

## *9.1 Introduction*

This chapter presents an evaluation of our research outcomes both SDL the language and the proof-of-concept tool. The evaluation focuses on the effectiveness, efficiency, and satisfaction with which users can undertake survey tasks when using SDL and the tool. This evaluation study has two main parts:

Part1: User testing (Human participants)
Part2: Cognitive dimensions evaluation

The usual note of caution must be stated. The nature of the tool and the development stage of the language must be taken into account in assessing its usability. All our research outcomes are prototypical so the focus of the evaluation study is not to be exhaustive fault finding but to provide coherent viewpoints for appraising the language features using cognitively-driven processes: cognitive dimensions and walkthroughs (Green 1996, Green and Petre 1996, Ko et al. 2002, Dix et al. 2003).

## *9.2 User Testing (Human Participants)*

This section presents a brief summary of the user testing, which took place between 31 Oct 2005 and 17 Nov 2005 for SDL and SDL software tools.
The usability testing was subject to approval by the *Human Participants Ethics Committee* of the University of Auckland's approval and all usability testing sessions adhered to the recommended guidelines of the committee.

### 9.2.1 Executive Summary

User testing sessions were conducted to explore end-user perceptions of SDL notations and supporting software tools. Our primary focus was to study how well SDL and tools assist users throughout the survey process. Observational data, participant responses and questionnaires were used to infer key aspects of diagrammatic and tool usability. Testing outcomes obtained in this study helped us make decisions about how user requirements for the survey process should be actualized in the context of visual language and software development. We found that participants expressed generally positive views of presented research outcomes during the study and the majority of the participants responded favourably to the efficacy of visually oriented survey software tools throughout the life cycle of the survey process. Although this brief study must not be taken out of context in terms of its size and duration, participant responses have reinforced our research direction and approach.

### 9.2.2 Background

*Research questions:*
Visual notations with appropriate software tool support can be used to facilitate statistical survey design and implementation. This research hypothesis was tested using interactive test scenarios which were based on an actual survey. Prototype software tools model future software solutions that would utilise SDL diagrammatic notations and the underlying model architecture.

*Testing settings:*
Test sessions took place at the graduate workspace for CS graduate students at the University of Auckland. At each session a test subject was provided with a PC with pre-installed Pounamu SDL tools.

*Test subjects:*
Eight test subjects participated in the user testing. These test subjects were chosen out of the potential candidate pool according to recommendations from a tutor at the Department of Statistics. All test subjects were invited to attend the introductory meeting which was designed to convey some of core concepts behind SDL All of the test subjects were new to the concept of a visual environment for statistical surveys. However they all had working knowledge of statistical packages, survey theory and survey design in either academic or commercial settings.

### 9.2.3 Methods

One aim of the user testing was to evaluate the usability of our diagrammatic notation designs and SDL based software tools in order to improve the SDL visual language design, to fine-tune software tool solutions, and to validate that research outcomes closely map to the user requirements. Another important aim of the user testing was to study how our approach to the survey process support would be perceived by users at high level and their comparative views of existing software tools and practices for statistical surveys.

There was a wide range of user activities that were conducted throughout a testing session included:

- Pre and post demonstration interviews
- Diagram comprehensions
- Survey technique implementations using provided software tools
- Comparative evaluations

Two types of user testing outcomes were compiled:

- User-completed questionnaires which included the feedback collected from users regarding their perceptions, opinions and satisfaction regarding the SDL, tools and their own performance.
- Performance evaluations included task correctness, mistakes, and time to complete given tasks recorded by the investigator during testing sessions.

Many aspects of the qualitative and quantitative measures were viewed in the light of their relationships to the cognitive dimensions framework.

### 9.2.4 Description

Each testing session ran for between two and half and three hours. With activities summarise in Table 9.1, one research investigator was responsible for the management of an entire session. The investigator briefed test subjects as outlined in the information sheet (see Appendix C) and explained goals and objectives of the user testing.

Test subjects were introduced to SDL diagrammatic notations and software tools with the aid of simple working examples. During the testing session users were given ample opportunity to query the investigator on issues related to visual notations and tools. Once test subjects expressed no difficulty in understanding and using SDL diagrams and tools, the investigator guided test subject to the next phase which was to test user's diagram comprehension ability.

Test diagrams were based on the 2001 NZ crime victims survey to simulate real-life survey communication problems. Test subjects were asked to explain the semantics of presented diagrams and to give feedback on their effectiveness, expressiveness, usefulness and usability.

User activities utilizing the software tools were to implement survey techniques to produce solutions for given scenarios. The scenarios were again based on the theme of the 2001 NZ crime victims survey. The scenarios were designed to permit user initiated actions and task execution to build the ideal cognitive walkthrough environment.

Lastly, the test subjects finalized their written questionnaires and participated in the discussion to explore additional issues such as comparative advantages and disadvantages against existing tools and methodologies and a personal evaluation of the testing session.

| *Activity* | *Time (approx min)* | *Data collection methods* |
|---|---|---|
| Introduction | 10 | |
| User profile | 10 | Questionnaire, observation sheet |
| SDL overview | 50 | Observation sheet |
| Diagram comprehensions | 40 | Questionnaire, observation sheet |
| Technique implementations | 50 | Questionnaire, observation sheet |
| Concluding Discussion | 20 | observation sheet |

**Table 9. 1 User activities table**

### 9.2.5 Evaluation Methodology

The theoretical testing basis for the evaluation is the cognitive dimensions framework (Green 1996) and the testing session incorporated the cognitive walkthrough (Ko et al 2002, Dix et al 2003). Unfortunately, there are few published research papers for testing visual languages. Beside the lack of research, the maturity of SDL complicates

an evaluation process, as it is often hard for users to pinpoint whether factors that degrade the usability of the language are largely stemmed from the language itself or the user interface users deal with in a testing session. Thus all negative feedbacks are examined thoroughly to differentiate the actual source of the problems.
The motivation for our choice of the cognitive walkthrough methodology is that with a limited number of research publications in the area of visual language testing, the cognitive walkthrough offers robust empirical study outlines that support our evaluation process. The cognitive walkthrough simulates a user-centred testing environment that could detect the potential difficulties of novice users in applying SDL in practice without a lengthy testing phase. The approach was helpful in designing the actual testing, as it specifically meets the profile of a test subject or a potential user and apparent time constraints due to the time span of this project.

### 9.2.6 Major Findings and Discussions

Even though the test subjects were new to the concept of using visual language for statistical surveys, the subjects were able to understand and use diagrammatic notations to express various aspects of the survey process and compose statistical techniques to solve test scenarios. The learning curves of the subjects varied but nonetheless all of them were able to comprehend testing diagrams to the level required for their tasks.

The following results are general quantitative indications on the usability of SDL and SDL tools (Appendix C contains more information on raw quantitative data):

| *Category* | *Results* |
|---|---|
| General tool usability | Positive 7/8 |
| | Negative 1/8 |
| Overall notation usability | Positive 6/8 |
| | Negative 2/8 (1 partial negative) |
| Diagram comprehension | Excellent 5/8 |
| (user performance) | Good 2/8 |
| | Average 1/8 |
| | Incomplete 0/8 |
| Task completion | Excellent 3/8 |
| (user performance) | Good 3/8 |
| | Average 1/8 |
| | Incomplete 1/8 |

**Table 9. 2 User performance**

The preceding table only shows a partial picture of end users' perceptions so it should be interpreted with the following additional information to draw out useful insights into the design issues and the usability of SDL and SDL tools.

***Major findings***

1. SDL accentuates and integrates the multiple aspects of a survey that are often not addressed by existing practice. This makes SDL based solutions to be more user oriented. Users interacted with visual notations not just to formulate numerical computations but also to convey actual operational semantics behind those activities. The test subjects responded that this approach would scale well to communicate large scale survey projects.

2. The visual modelling approach and inter-diagram mappings helped users to think of a survey project as set of tasks within the context of the whole survey process and individual survey constructs to be conceptualized as reusable components. Visual notations favourably supported the test subjects in the shift as visual notations and inter-diagram mappings explicitly illustrated a survey project as the sum of integrated aspects and potential compartmentation of some survey activities.

3. Communication efficiency
The effect of the visual approach on comprehension performance of both high and low level details of the survey process varied. Each diagram drew different responses with overwhelmingly positive overall feedback. Survey diagrams were received well by the test subjects for being easy-to-use, expressive and a time-effective alternative to conventional documentation.

Survey task diagrams were viewed as too radical a departure from existing practices by two test subjects while the majority of the test subjects commented that the diagrams visualized a valuable aspect of the survey process.

It was interesting to note that the three of the test subjects who had previous experience in highly specialized statistical research roles were most enthusiastic about the concept of task models. Their professional or academic backgrounds were ranged from financial modelling to accounting. The tacit existence of prevalent patterns in those specialized fields seemed to be largely responsible for the interesting reactions.

Both survey technique and data diagrams were received favourably. They shared their strengths and weaknesses as both types of diagrams look alike and convey information at similar levels. Their most notable strength was in capturing the integrated view of sampling, statistical metadata and techniques using an intuitive dataflow metaphor. Negative feedbacks on the diagrams were mostly originated from the lack of secondary notations and this aspect of SDL design is explored further in a later section.

4. For visual language novices, it was not a trivial task for them to visualize inter-diagram relationships and to harmonize disparate diagrams into a single unified model. Even though test subjects did very well in comprehending and utilizing individual diagrams they expressed a slight difficulty in mapping all the diagrams together in designing the survey process. This problem could be analogous to a novice UML user's difficulty in merging all UML diagrams mentally together to form a unified view. One of the manifestations of user errors, which can be traced back to user's incorrect usage of inter-diagram mapping, was the potential introduction of inconsistencies. One promising solution suggested by the test subjects to remedy this

design issue was the creation of a visual layer to dynamically illustrate how the underlying model is formed by contributing diagrams and relationships amongst them. Future development of SDL tools should explore the feasibility of this approach.

5. Lack of secondary notation
Some of the test subjects shared the suggestion that centred on additional visual or textual aids for SDL diagrams to make the central theme of an SDL diagram more accessible in one brief visual scan especially for survey data and technique diagrams. This problem can be multi-faceted as follows:

> (1) The negative responses were first evoked while the test subjects were interacting with survey data and technique diagrams. Survey and task diagrams were designed to be part of an organization's consensus forming processes hence they have less fine-grained uses. The generic natures of those two latter diagrams were perceived as informal and required less attention from users to acquire high-level details. Upon further discussions with the test subjects, it was revealed that one shared design element of the two diagrams greatly helped the test subjects in comprehension tests. The two diagrams mandate one graphical icon which represents the core contextual theme of a diagram and users are able to scan a whole diagram by traversing away from the icon in contrast to three other diagrams that had no such graphical entity to tie all graphical entities centrally. Hence it could be inferred that introducing a graphical entity which imparts the central theme of a diagram at the beginning of data flow may be a promising solution to the design issue.
>
> (2) As noted by some of key researchers (Green and Blackwell 1998) in visual language research, some notable visual languages such as the LabView and Prograph lack secondary notation support. SDL diagrammatic notations will be reviewed to supplement notational and tool supports for secondary notation.

6. Busy diagram – Visibility
All five diagrams allowed all graphical entities to be viewed and manipulated in a single view pane at the individual level. This may have addressed a significant portion of visibility criteria but without automatic layout management to beautify graphical constructs, visual distractions accumulated as the size of a diagram grew. Redesign efforts should include a diagram layout management component (e.g. automated tree layout management) to alleviate the visual distraction level of large scale diagrams as well as more research into visual elements that enhance the visibility of SDL diagrams.

7. Juxtaposability
One important aspect of SDL is its ability to have a survey represented in multiple diagrams and a SDL diagram's graphical entity may have several sub-level layers that edify lower level information associated with the graphical entity. This design feature of SDL could contribute to some of potential problems in juxtaposability of SDL. For example a survey data diagram's data entity may have up to 3 sub layers associated with the entity. The presence of the associations is visually noted by change in the entity's boundary line shape and colour. An ad-hoc remedy such as putting lower-level information on one top-level layer may address the juxtaposability issue but the

remedy hinges on a heavy trade off in a diagram's level of visual clutter. Assuming SDL's primary authoring environment will be software tools, the level of juxtaposability can be enhanced by offering tool level enhancements while minimising the visibility issues. For instance more fluid navigation between diagram layers allows user's working memory to be minimally disrupted by changing scenes. Another tool-level approach to remedy the trade-off can be the interactive 3D visualisation of SDL diagram layers. A 3D environment may provide better a visual perspective for users that side-by-side presentation in 2D with the same screen dimensions cannot offer.

8. Flexibility in changing designs
The test subjects responded favourably to the degree of freedom that SDL tools offer in designing the survey process. There are no specialized survey tools which correspond to the functionalities of survey and task diagrams hence survey technique and data diagrams were obvious candidates for the test subjects' comparative reviews. Caution must be taken as the following comparative reviews should be taken in the context of the test subjects' exposure to statistical software tools.
Test subjects noticed that, unlike existing tools they were not restricted to a sequential batch mode or an interactive mode which tends to require high attention investment (Blackwell and Green 1999) to articulate a technique which should be frequently modified. During manipulation of implemented statistical techniques, the existing tools put emphasis on result oriented step-by-step batch operations or UI based pre-packaged procedures. SDL tools' target emphasis is not exclusively result oriented. SDL diagrammatic notations aim to capture the whole semantics embodied in a statistical technique thus allowing a type of expressiveness that is intuitive to the user. The data flow metaphor utilized in survey data and technique diagrams provided to the users design-time freedom in changing input and output data flows and the dynamic mapping of a graphical entity to a physical dataset or a statistical technique meant that data flows could be routed to multiple techniques in a variety of ways to easily form many variations of the initial design model. All these positive aspects made SDL tools successful in giving visual cues for the whole process while providing direct manipulation interfaces for a wide range of operations.

9. No explicit needs to know programming languages
Developing and implementing statistical techniques in many popular statistical computing packages entails the translation of symbolic mathematical statements into tool specific languages. When operational requirements are embedded into the survey process, users need to switch back and forth between two vastly different modes of mental operation. If implementations in both mental modes are not consistent or demand hard mental operations, the whole survey process can be influenced by many harmful side effects such as a high abstraction barrier, and expensive modification activities. In the visual environment which is supported by the prototype tool, the test subjects were expected to know correct usage of statistical techniques but they were separated from low-level representations of the techniques that only exist in the forms of service components. Technique constructions are metaphoric abstractions that reside in a single domain. Thus a technique composition in actuality is a model building exercise that is entirely independent from underlying platforms.
Although these attributes of the prototype tool brought a new level of usability to the test subjects, some instances where the model-level coupling would work against performance efficiency were also found. For example current SDL tools do not allow users to do code level modifications; an expert who is comfortable with raw code

modifications in R's interactive mode would outperform the SDL-based approach when the statistical techniques, which are provided as a service to SDL tools, themselves are subject to frequent changes. Therefore possible tradeoffs in implementing a dual-mode access to mapped services should be considered in the future development of SDL tools. The example illustrates how established practices may unexpectedly deteriorate the quality of the solution which is founded on theoretically presumed user patterns.

10. Extra control interfaces
One of the design principles behind SDL was to minimise set of graphical entities in a diagram. This design decision was principally made to lower the accessibility of the barrier to accommodate a diverse pool of users. The approach was justified by positive user responses regarding the usability of diagrammatic notations and SDL tools. However the presumption that less equals better usability was not the case across all user experiences. As the test subjects became more immersed in the visual environment, their demand for more direct visual control of data flow grew in scope and functionality.

Our prototype tool allows users to navigate to a sub layer to modify parameters of a mapped service via a dynamically generated UI but once they leave the layer they are no longer able to edit the mapped service either by indirect or direct manipulation and the values of the parameters are hidden behind a graphical icon. Direct control of the mapped service is delegated to the sub layer primarily to reduce users' memory load and to provide an exact contextual perspective for each visual layer as discussed in one of the design principles of SDL. One of the interesting observations during the testing sessions was that more competent users requested the ability to manipulate mapped services within the top level diagram without navigating to appropriate sub level diagrams. Further post session discussions revealed that the user request is reminiscent of *shortcuts.* Shortcuts provide a secondary access to application functionalities in a typical GUI design for users who wish to bypass GUI based interactions to save time and effort. Likewise, visual shortcuts should improve user performance by offering time-saving alternatives. The design and implementation of shortcuts in the visual language framework will be briefly discussed in the following sub chapter (glass view). The visual shortcut approach will be investigated in conjunction with better elision support in our future work.

## 9.3 Cognitive Dimensions Evaluation

This section is a follow-up of the user testing and parallels our previous research efforts in understanding diagrammatic notation usability in terms of cognitive dimensions (Green 1996, Green and Blackwell 1998). Our discussion begins with how the language features of SDL are relevant to core cognitive dimensions. Then the discussion moves on to the issues regarding SDL's cognitive relevance, trade-offs and design manoeuvres. The latter part of this section will be re-examined in the context of software tool support. The cognitive framework provides basic toolsets to discuss the multifaceted usability issues of information artefacts.

Our approach is to break applicable cognitive artefacts down into the individual cognitive components. Since SDL consists of a set of diagrams, this complicates our analysis as thorough dimensional taxonomies would be excessively vast for the scope of this chapter. Therefore we do not present all cognitive dimensions categorically but steer the direction of the analysis along the most significant cognitive dimensions that surfaced during user testing sessions and a speculative study of the diagrammatic notations and tool support.

**Closeness of mapping: Real World View to Model View**
The level of closeness of mapping between a user's model of a statistical survey and SDL can be evaluated on two fronts: the language design principles and user testing session feedbacks.
In the area of the language design principles the following observations can be made:

- The cognitive tool models behind SDL diagrams add conceptual clarity in navigating the problem domain and its corresponding visual entities in SDL.
- The decision to develop multiple representation models to capture the complex and multifaceted nature of the survey process progressed into the design of highly "homomorphic" models (Barwise and Etchemendy 1995) and their closeness to the problem domain are expected to be better than alternatives using one universal diagrammatic representation to model statistical surveys.

The following observations can be made from the user testing session feedbacks:

- Overwhelmingly positive first-time user feedback in using SDL diagrams seem to justify the design decisions made to ensure a high level of closeness of mapping. Dataflow oriented aspects of survey data and technique diagrams were well received by users as the mode of abstraction fitted well with user's conceptual model of sampling and data analysis. Task diagrams did not rated as well as other types of SDL diagrams as the novel nature of the task model contributed to the higher than average entry barrier.

**Viscosity: Resistance to Local Change**
Multiple diagram layers and multiple diagram mappings add abstractions to enhance the viscosity of SDL. However the following trade-offs should be considered in counterbalancing the merits of the design decisions:

- Users are required to comprehend the underlying model which integrates all of the different diagram types. Users must be knowledgeable about the survey process to conceptualize the integration to take full advantage of SDL and SDL tools (heightened entry barrier).
- Hidden dependencies may introduce inconsistencies. (e.g. a graphical icon is mapped to a non-existent dataset, logical corruption in inter-diagram links) Tool-level supports can detect and mitigate user activities that may cause such inconsistencies (hidden dependencies).
- The data flow metaphor and the dynamic mapping to external resources allow more design-time freedom for users in changing input and output data flows and employed techniques.

Since there are no comparable visual languages in the domain of statistical surveys, it is difficult to place comparative judgment on SDL's diagrammatic notations and tool support. Our generalisation based on the user feedback and speculative probe into the language and the tool features indicates a comparatively much reduced viscosity against existing alternatives such as batch mode processing, code based interactive models supported by R and procedure-based statistical packages. User feedback indicated that some of the notable features of the proof-of-concept tool in reducing viscosity were:

- Compartmentalisation of sampling operations and techniques into abstract visual entities.
- Easiness in changing the chain of inputs and outputs that are tied to a statistical technique.

**Progressive evaluation: Evaluate As You Go**
The tool support for SDL enables dynamic resource bindings, partial/full execution of data operations and techniques. Also immediate feedback can be obtained via output ports. The data flow metaphor based visual presentations offer rich flexibility in the definition, modification and execution of survey techniques and data operations. Multiple/parallel technique authoring is supported to reflect the iterative and incremental nature of developing data exploration techniques.

Progressive evaluation to test partial/complete systems and models is one area where our approach demonstrated significant advantages over existing software packages which the test users had been exposed to previously.

**Visibility**
Each diagram type has an exact contextual domain so given a particular problem domain it is simple to search and obtain details for various aspect of the survey process. Hidden dependencies such as bound external resources (e.g. datasets and techniques) can be revealed by navigating to an appropriate sub layer. Unstructured large-scale visual constructs may exacerbate the visibility of diagrams. Automatic layout management should be investigated to alleviate the visual distraction.

**Juxtaposability: Viewing diagrams side by side**
The added abstraction (diagram layers and mappings) may require frequent navigation among diagram layers and disrupt user's working memory.
Tool supports for more fluid navigation and possible 3D visualization that takes up less spatial dimensions can be one of solutions to improve the Juxtaposability dimension of the SDL tool. Some of the Juxtaposability issues can be traced to our tool's framework - Pounamu. From developer's point of view, Pounamu offers virtually all the extensionalities of the underlying Java framework. However implementing the elaborate navigation or visualisation schemes, which demand substantially different behaviours and interaction models from usual GUI editing tool, may require developers to bypass Pounamu's native support and negate the model-driven light-weight approach of Pounamu. However this is as a momentary issue considering Pounamu's pre-release status.

**Secondary Notation**
The survey data and technique diagram lack visual or textural cues to express the central theme of a diagram. The two diagrams are less self documenting for the novice. User feedback indicates that the introduction of a graphical entity which

imparts a central theme of a diagram may be required. One post-user evaluation effort to bring an increased level of secondary notation support was the implementation of a transparent interface (glass view) to aid dynamic exploration of extra information contained by a visual entity for a given layer. The approach can be analogous to the situation where we need to enhance user comprehension on the top of what already exists. E.g. the use of different fonts to accentuate special dates (Calendar).

The tool level approach utilises the adjunct interface to add extra dimensions to what can be expressed by a diagram. For instance, users may place the transparent interface over a diagram and it may reveal extra information such as a central subject and user interaction history behind a visual icon without resorting to the inter-layer navigation or accessing metamodel files. They also have an option to cover the whole diagram and view metamodel-level information unobtrusively.

**Abstractions**
SDL diagrams have varying level of abstraction for each diagram type.
Considering the novelty of visual languages in survey design and implementation, SDL may exhibit high entry barriers. Multiple SDL diagram types with specialized contextual goals provide a development environment for users to utilize SDL diagrammatic notations and tools in part without mandatory requirements for defining all the aspects of the survey process. This approach means SDL diagrams are abstract-tolerant as a whole.

## *9.4 Summary*

In this chapter, we discussed the evaluation of the proof-of-concept tool and diagrammatic notions of SDL. We centred our discussion on the user testing following the cognitive walkthrough methodology in addition to some notable user feedback. The major findings of the user testing were presented with their relevance to the cognitive framework. Appendix C provides more detailed information on the user testing.

# Chapter 10 Conclusion

## 10.1 Introduction

This thesis covered diverse research areas to create a suite of visual languages and the proof-of-concept tool for statistical surveys. In this chapter we take a last look at what we have accomplished and show plans and directions for future work.

## 10.2 Contributions of this Thesis

Our initial attempt to investigate statistical survey support from visual approach (Kim, Hosking, and Grundy 2005) suggested providing back end integration with other statistical survey tools so that our SDL environment can be used to not only design a conceptual model, but also to make it executable. Future work envisaged in the initial research was a starting blueprint of our work. However our research diversified in many areas and we discussed the following main contribution of this thesis.

*Survey Design Language (SDL)*

SDL is the major piece of our work and is a novel approach to describe statistical surveys in a unified manner. It provides a conceptual framework for survey designers to express survey specification and supports model-based development of the survey process.

*Layered Diagram Model*

SDL diagrams exist in three layers: visual, metamodel and semantic. The main benefit of the layered approach is to maintain a high-level of abstraction for easy visual comprehension. The metamodel layer provides concrete specifications that are not available the visual layer. While doing so it facilitates back-end integration into heterogeneous external services such as a computation engine and inter-diagram integration. The metamodel layer provides a foundation for executable models.
The semantic layer provides a means to make sense of the information represented by metamodel entities and relationships.

We feel that the rich semantic support makes SDL very extensible. The semantic layer reflects the understanding of diagrams from the human perspective. When there is need for an update of SDL, we can start from the semantic layer and then build themetamodel representation of required changes and create visual constructs that represent the metamodel representation. This language extension mechanism provides a novel option for those who wish to design a highly extensible visual language with an ontological base.

*Survey Process Support*

The survey process has multi-faceted sources of complexity and our multiple representation scheme with its unified model base provides the first conceptual modelling method to model the survey process. The favourable user evaluation feedback indicates that further development in this direction would be interesting.

*Mitigation of the Barrier between Conceptual and Physical Implementation*

We have identified during interviews with our statistical consultant that due to the extremely low adoption of any form of visual communication, it would be very unlikely that SDL would be useful in a practical sense if it came with an explicit barrier between two types of activities (modelling and tool utilisation).

With SDL and the visual tool environment, users can turn a purely visual representation into a working software tool or an executable web service. This is a significant step in reducing the barrier that exists between modelling and implementation activities. From the practicing survey designer' view, this is a radical departure from the common tool interaction models offered by existing statistical software packages.

*Openness and Survey Information Dissemination Infrastructure*

The proof-of-concept tool promotes collaboration with third party users by prompting users to publish statistical data, metadata, metamodel, and technique.
At least in the limited sense, users are able to access not only static resources such as a data file but to also utilise multiple views of statistical data and actively make use of statistical techniques in their own environment. This supports web service consumption.
Our proof-of-concept tool illustrates that sophisticated statistical computing capabilities are now easily accessible without the use of proprietary computing packages. Our tool's ability to integrate itself with R's computing engine via light-weight TCP/IP connections provide a template for developers who want to introduce sophisticated statistical techniques into their software tools.

## 10.3 Future Work

As we mentioned in the beginning of this thesis, the scope of this thesis is very eclectic and we feel that the depth of our research has been severely limited by imposed time limits. This also means that virtually all areas of our work can be considered for future work. We summarise some of the most significant areas of extension as follows:

- The refinement of the layered diagram structure. So that we have a solid ontological foundation for diagram for language extension and possible tool support for customising SDL. Formalisation of diagrams, inter-diagram and inter-layer relationships should also be considered.

- Implementation of the survey process automation by supporting executable survey process diagrams.

- Support for more sophisticated service and resource repositories.

- Resource management mechanism for published data, metadata and techniques.

- Better visualisation and management of inter-diagram relationships and inconsistencies.

- Support reusable survey process models that are expressed in SDL.

- Investigate the automation of statistical code script (e.g. R code script) conversion into readily usable SDL services. This is an important step in increasing supported statistical functionalities and offer interesting options for practicing survey researchers to reuse their current stock of statistical techniques implemented in statistical languages.

- Support for better user experiences (tool issues):
    - More visualisation options
    - Fluid inter-view navigation
    - Elimination of repetitive manual operations in diagram authoring
    - Intelligent elision support to improve diagram usability

From a practical point of view, the further development of the existing prototype tool should be the first concern. The full development of the prototype tool will be a considerable software project and the following areas should be prioritised in achieving a production quality tool:

- Better visualisation of the binding process.

- Better visualisation of data and support of sophisticated data editing functionalities.

- Better mapping form generation mechanism (in both aesthetic quality and functionality) in the binding process.

- Enhance the back-end computation engine interface efficiency.

- Intelligent visual component layout

- Support visual skins for diagram icons that can give visual cues on the status of underlying models.

## Bibliography

*Ahmed 2002*
Kal Ahmed, Topic Maps - A Practical Introduction with Case Studies XML Europe 2002, Barcelona, Spain.
http://www.idealliance.org/papers/xmle02/dx_xmle02/papers/03-05-01/03-05-01.html

*ASC 2005*
the Association for Survey Computing (ASC) Software Register,
http://www.asc.org.uk/Register/

*Barwise and Etchemendy 1991*
Jon Barwiseand John Etchemendy, Visualization in teaching and learning mathematics table of contents, Pages: 9 - 24, 1991

*Barwise and Etchemendy 1995*
Jon Barwiseand John Etchemendy, Logical Reasoning with Diagrams Oxford University Press, USA (June 13, 1996) Part C

*Biemer and Lyberg 2003*
Paul P. Biemer and Lars E. Lyberg, Introduction to survey quality Wiley Inter-Science 2003, Ch 2

*Biezunski 1999*
Michel Biezunski Topic Maps at a glance Proceedings of XML EUROPE '99. 26-30 April 1999 Granada, Spain [OASIS; W3C World Wide Web Consortium] 1999"

*Biezunski 2003*
Michel Biezunski The IRS Tax Map XML Europe 2003 Conference, May 5-8, 2003, London, UK.

*Blackwell 1998*
Alan Frank Blackwell , PhD Thesis, Darwin College Cambridge

*Blackwell and Green 1999*
Alan F. Blackwell and T.R.G. Green,Investment of Attention as an Analytic Approach to Cognitive Dimensions Collected Papers of the 11th Annual Workshop of the Psychology of rogramming Interest Group (PPIG-11), pp. 24-35

Chambers and Ryan 1990
John M. Chambers and Barbara F. Ryan
"The American Statistician, May 1990 (Vol 4, No. 2 pp87-89)"

*Dalenius 1995*
Dalenius, T., Elements of Survey Sampling, Swedish Agency for Research Cooperation with Developing Countries, Sweden

*Davies and Fensel and Harnelen 2003*
John Davies Dieter Fensel Frank van Harmelen,TOWARDS THE SEMANTIC WEB Ch2 2003

*DDI*
The Data Documentation Initiative
http://www.icpsr.umich.edu/DDI/

*Degler and Battle 2003*
Duane Degler and Lisa Battle "Extreme Markup Languages 2003 Montréal, Québec August 4-8, 2003"

*DHI*
The Dictionary of the History of Ideas
http://etext.lib.virginia.edu/DicHist/dict.html

*Dix et al 2003*
Alan Dix, Janet Finlay, Gregory Abowd and Russell Beale, Human-Computer Interaction, Prentice Hall, International, 2003 3rd Ed Ch12

*Dome*
Honeywell, 2000
http://www.src.honeywell.com/dome

*France 1999*
Robert France "Proceedings of the 14th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications"

*Froeschl, Grossmann, and, Vecchio 2003*
Karl A. Froeschl, Wilfried Grossmann, Vincenzo Del Vecchio, MetaNet project - the concept of statistical metadata 2003, Part of the Information Society Technology strand of the European Union Fifth framework Research and Development program (IST-1999-29093),
http://www.epros.ed.ac.uk/metanet/deliverables/D5/IST-1999-29093_D5.doc

*Gärdenfors 2000*
Peter Gärdenfors, Conceptual Spaces: The Geometry of Thought The MIT Press (March 20, 2000)

*Garshol 2003*
Lars Marius Garshol, Living with topic maps and RDF the proceedings of XML Europe 2003, 5-8 May 2003

*Goldin, Keil, and Wegner 2001*
Dina Goldin, David Keil, Peter Wegner
Unified Modeling Language: Systems Analysis, Design and Development Issues, Idea Publishing Group, p249—263

*Green and Petre 1996*
T. R. G. Green, M. Petre, Usability Analysis of Visual Programming Environments: a cognitive dimensions framework, Journal of Visual Languages and Computing, 1996

*Green 1996*
T.R.G. Green, An Introduction to the Cognitive Dimensions Framework Extended abstract of invited talk at MIRA workshop, Monselice, Italy Nov 1996

*Green and Blackwell 1998*
Thomas Green and Alan Blackwell   Cognitive Dimensions of Information Artefacts: a tutorial 1998 Part 2

*Grossmann 2003*
Grossmann, Wilfried  Computing Science and Statistics, 35, I2003Proceedings

*Grundy, Hosking, and Mugridge 1998*
John Grundy,John Hosking,Warwick B. Mugridge
Inconsistency Management for Multiple-View Software Development Environments
IEEE Transactions on Software Engineering archive
Volume 24 ,  Issue 11  (November 1998) table of contents
pp 960 - 981   1998

*Howison 2005*
Sam Howison OCIAM Mathematical Institute Oxford University, Practical Applied Mathematics Modelling, Analysis, Approximation Ch2  Cambridge University Press (March 24, 2005)

*ISO/IEC 13250*
International Standard ISO/IEC 13250
www.y12.doe.gov/sgml/sc34/document/0129.pdf

*Jenkins 1996*
Stephen G Jenkins, The Triple-S survey interchange standard
http://www.triple-s.org/sssasc96.htm

*Keller 2004*
David Keller, UML Hierarchy diagram Unified Modeling Language: Superstructure version 2.0 formal/05-07-04

*Kelly and Rossi 1996*
Kelly, S., Lyytinen, K., and Rossi, M., Meta Edit+: A Fully configurable Multi-User and Multi-Tool CASE

*Kim, Hosking, and Grundy 2005*
Kim, C., Hosking, J., and Grundy, J., A Suite of Visual Languages for Statistical Survey Specification. IEEE VL/HCC 2005: 19-26

*Klein and Schürr 1997*
P. Klein, A. Schürr, Constructing SDEs with the IPSEN Meta Environment  in Proc. 8th Conf. on Software Engineering Environments SEE'97, Los Alamitos: IEEE Computer Society Press (1997), 2-10

*Ko et al. 2002*
A. J. Ko, Margaret M. Burnett, Thomas R. G. Green, Karen J. Rothermel, Curtis R. Cook: Improving the Design of Visual Programming Language Experiments Using Cognitive Walkthroughs. J. Visual. Lang. Comput. 13(5): 517-544 (2002)

*Lakoff 1993*
Lakoff, G., The contemporary theory of metaphor. In A. Ortony (Ed.), Metaphor and Thought (2nd ed.). Cambridge: Cambridge University Press. pp. 202-251.

*Lamb 2001*
Lamb, J.
Sharing best methods and know-how for improving generation and use of metadata. European Commission Joint Research Centre ETK &NTTS conference 2001
http://webfarm.jrc.cec.eu.int/etk-ntts/Papers/final_papers/19.pdf

*Ledeczi et al. 2001*
Akos Ledeczi, Miklos Maroti, Arpad Bakay, Gabor Karsai,
The Generic Modeling Environment, Proceedings of WISP 2001, May, 2001

*Liu and Hosking and Grundy 2005*
Na Liu, John Hosking, John Grundy
A Visual Language and Environment for Specifying Design Tool Event Handling
2005 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC'05) pp. 278-280

*Madansky 1996*
Madansky A., Journal of Official Statistics, Vol.2, No.4, 1986. pp. 561-569

*Mahabal et al. 2002*
Ashish Mahabal (Caltech), George Djorgovski (Caltech),Robert Brunner (Caltech), Roy Williams (Caltech), Topic Maps for the Virtual Observatory SPIE 2002 (Hawaii)

*Medvidovic et al 2001*
Nenad Medvidovic, Paul Grünbacher, Alexander Egyed, Barry W. Boehm
Bridging models across the software lifecycle International Conference on Software Engineering and Knowledge Engineering (SEKE 2001)

*Meller et al.2004*
Stephen J. Mellor, Kendall Scott, Axel Uhl, Dirk Weise, MDA Distilled: Principles of Model-Driven Architecture Addison Wesley 2004 Ch4

*Melnik and Decker 2000*
Melnik, Serge and Decker, Stefan, the Proc. of the ECDL'00 Workshop on the Semantic Web 2000

*MetaNet 2003*
EC MetaNet project final conference report 2003

*Moore and McCabe 1993*
D Moore and G McCabe, The practice of statistics 2ed, Freeman, 1993, pages 220 – 223

*Narasimhan 2005*
Balasubramanian Narasimhan, Standford University
Journal of Statistical Software February 2005, Volume 13, Issue 4.

*Novak 1992*
Gordon S. Novak Jr.
Diagrammatic Reasoning: Cognitive and Computational Perspectives, Janice Glasgow, N. Hari Narayanan, and B. Chandrasekaran, eds., AAAI Press / MIT Press, 1995, pp. 753-774.

*NZCS 2003*
New Zealand National Survey of Crime Victims in 2001    First published in May 2003 by the Ministry of Justice

*Olenski 2003*
Józef Olenski, Global Standard for Harmonization of Social Statistics,
Expert Group Meeting on Setting the Scope of Social Statistics United Nations Statistics Division in collaboration with the Siena Group on Social Statistics New York, 6-9 May 2003

*OMG 1999*
OMG   OMG UML Specification Version 1.3

*OMG 2004*
OMG   Unified Modeling Language: Superstructure

*OMG 2005*
OMG   Introduction to OMG's  UML

*Park and Hunting 2002*
Jack Park, Sam Hunting,  XML Topic Maps: Creating and Using Topic Maps for the Web Addison Wesley 2002

*Passin 2004*
Thomas B. Passin, Explorer's Guide to the Semantic Web, Manning Publications Co. 2004

*Pautasso and Alonso 2003*
Cesare Pautasso, Gustavo Alonso,  the Proceedings of the 2003 IEEE Symposia on Human Centric Computing Languages and Environments (HCC 2003), Auckland, New Zealand, October 2003.

*Pepper 2002*
Steve Pepper,  The TAO of Topic Maps, Ontopia AS 2002
http://www.ontopia.net/topicmaps/materials/tao.html

*Sato 1991*
Sato, H.
Statistical data models: From a statistical table to a conceptual approach. In Michalewicz, Z. (Ed.), Statistical and Scientific Databases. Horwood, 167-199, 1991

*Sutcliffe 2002*
Alistair Sutcliffe, Domain Theory: Patterns for Knowledge and Software Reuse, Lawrence Erlbaum Associates, Inc. Mahwah, NJ, USA, 2002
*Tannenbaum 2001*
Adrienne Tannenbaum, Metadata Solutions: Using Metamodels, Repositories, XML, and Enterprise Portals to Generate Informatio on Demand. Addison-Wesley, 2001

*Tienhaara*
Tienhaara, N.,
DASHati Software Ltd. http://www.triple-s.org/ssstienhaara.htm

*Unhelkar 2005*
Bhuvan Unhelkar, Verification and validation for quality of UML 2.0 models, Wiley-Interscience, p28

*Unhelkar and Henderson-Sellers 2004*
Unhelkar, B., and Henderson-Sellers, B., Modelling Spaces and the UML, the Proceedings of the IRMA conference, New Orleans, 2004

*Vatant 2004*
Bernard Vatant, Ontology-driven topic maps
www.idealliance.org/europe/04/ call/xmlpapers/03-03-03.91/.03-03-03.html

*XRef*
DocBook: The Definitive Guide Version 1.0.3. OASIS http://www.oasis-open.org/docbook/documentation/reference/html/xref.html DocBook: The Definitive Guide Version 1.0.3.

*XTM 2001*
Members of the TopicMaps.Org Authoring Group XML Topic Maps (XTM) 1.0
http://www.topicmaps.org/xtm/1.0/

*Young and Bann 1997*
Forrest W. Young & Carla M. Bann "ViSta: A Visual Statistics System Statistical Computing Environments for Social Research., Sage Publications, Inc. (1997), pp. 207-235"

*Zhu, Grundy, and Hosking 2004*
Nianping Zhu, John Grundy, and John Hosking, Pounamu: A Meta-tool for Multi-View Visual Language Environment Construction 2004 IEEE Symposium on Visual Languages - Human Centric Computing (VLHCC'04) pp. 254-256

# Appendix A – SDL in Details

This appendix contains a succinct primer to present SDL in details in the areas of:

- Diagram usage at the visual layer
- Metamodel description
- Metamodel to semantic layer relationships

The scope of this appendix is limited to visual descriptions of the language. The prototype SDL tool usage is the topic of Appendix C. However the content of this appendix will clarify the mapping processes between the three layers and recommended directions for future extensions.

## *A.1 Diagram usage*

This section presents diagram usage that dictates how visual symbols are assembled together to express various semantics. SDL has five types of diagrams and each diagram has unique syntactic rules that should be applied to instances of the diagram. We summarises the syntactic rules of SDL in the following tables for convenient look-up of the rules. Since diagram syntactic rules are frequently referred to novices to compose a diagram, our presentation style is direct and task-oriented. For more information on the underlying semantics of SDL diagrams refer Chapter 3 and the case study in Chapter 8.

### A 1.1 Survey Diagram

**Table A.1 Syntactic rules for survey diagrams**

| *Task* | *Rules* | *Description* |
|---|---|---|
| Create a central survey topic | Put a survey symbol on a diagram. | A survey symbol has no dependencies. But there has to be just one survey topic per diagram. |
| Add survey contexts | Attach survey contexts to a survey symbol using survey context connectors. | Survey contexts must be supported by a survey topic. |
| Add survey attributes | Attach survey attributes to a survey context using survey attribute connectors. Survey | A survey attribute represents a significant attribute that is tied to a |

| | | |
|---|---|---|
| | attributes can not be part of more than one survey context. | particular context. |
| Annotate consensus forming process | Connect related attributes that are involved in a consensus forming process using survey attribute connectors and add descriptions of the connectors to express the process flow. | |

## A 1.2 Survey Task Diagram

**Table A.2 Syntactic rules for survey task diagrams**

| *Task* | *Rules* | *Description* |
|---|---|---|
| Build a task hierarchy | Put a top level task first then build according to their execution order from left to right. Tasks at the left side precede those at the right side sequentially and tasks are executed bottom-up. | There can be only one top task. |
| Change the default order (Left to Right) of task execution | Add task operators. | Refer to Section 5.4 of Chapter 3 for more details of their usage |
| Express task outcomes/inputs | Attach survey artefacts to a task. A survey artefact can be connected to multiple tasks and vice versa. | |

## *A 1.3 Survey Data Diagram*

**Table A.3 Syntactic rules for survey data diagrams**

| *Task* | *Rules* | *Description* |
|---|---|---|
| Specify input data to a data operation | Build a data flow into a data operation by connecting a data icon to a data operation symbol. (arrow in) | There can be as many or as few inputs into a data operation but must abide by the specification of the data operation. |
| Express a sample to population relationship | Set up a data entity tree and express 'part-of' relationship | |

|  |  |  |
|---|---|---|
|  | using aggregation connectors. |  |
| Specify output data of a data operation | Build a data flow out of a data operation by connecting a data icon to a data operation symbol. (arrow out) | There can be as many or as few outputs out of a data operation but must abide by the specification of the data operation. |

### *A 1.4 Survey Technique Diagram*

**Table A.4 Syntactic rules for survey technique diagrams**

| ***Task*** | ***Rules*** | ***Description*** |
|---|---|---|
| Specify input data to a technique | Build a data flow into a technique by connecting a data icon to a technique symbol. (arrow in) | There can be as many or as few inputs into a technique but must abide by the specification of the data operation. |
| Capture visual or textual outcomes of a technique | Connect a technique output port to a technique icon. |  |
| Specify output data of a technique | Build a data flow out of a technique by connecting a data icon to a technique symbol. (arrow out) | There can be as many or as few outputs out of a technique but must abide by the specification of the data operation. |

### *A 1.5 Survey Process Diagram*

**Table A.5 Syntactic rules for survey process diagrams**

| ***Task*** | ***Rules*** | ***Description*** |
|---|---|---|
| Set up process stages | Each stage is represented by a stage symbol. If two or more stages form (a) sequential process flow(s) then connect them with stage transition connectors. | When there are no inter-stage relationships. This may imply a parallel process. |
| Model participating tasks in a stage (Layer-0) | There can be one or more tasks involved in a stage. Connect them with either output or input connectors according to the nature of their goal. | All tasks either contribute or consume. This is a mandatory condition imposed on tasks. |

| | | |
|---|---|---|
| Express task interactions within a stage (Layer-1) | Each stage has it own interaction layer. Model tasks along with their survey artefacts. Interactions between tasks are expressed in relationships between survey artefacts. The relationships are explicitly visualised using artefact-to-artefact connectors. | |

## A.2 Metamodel Description

Visual icons at the visual layer are abstracted, generalised and classified to produce metamodel representations of them. The structural description of the metamodel is shown in Figure A.1.

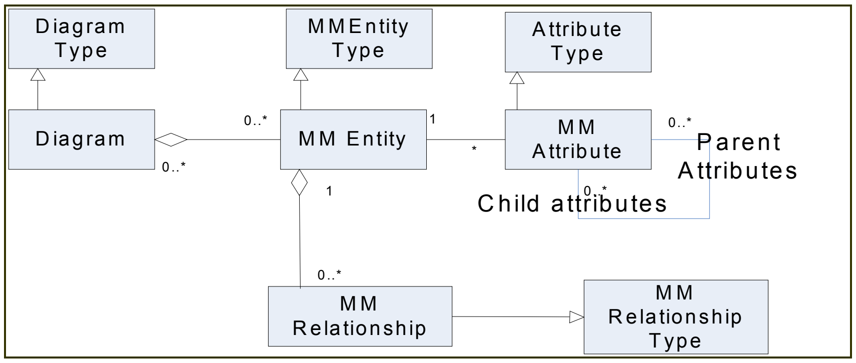


**Figure A.1 Structural overview of the metamodel layer**

The metamodel layer can be built by processing each non-connector type icon. The outcomes of the process are persisted via XML files that are bound to the icons. The process includes the following tasks:

- Abstraction of the visual icon into a metamodel entity.
- Classification of the metamodel entity into a specific type.
- Generalisation of the structural components.

The first part of this section contains diagrams of several of the metamodel entity types of the SDL. The entity types are explored further in the second part of this section, which lists the nature of each type.

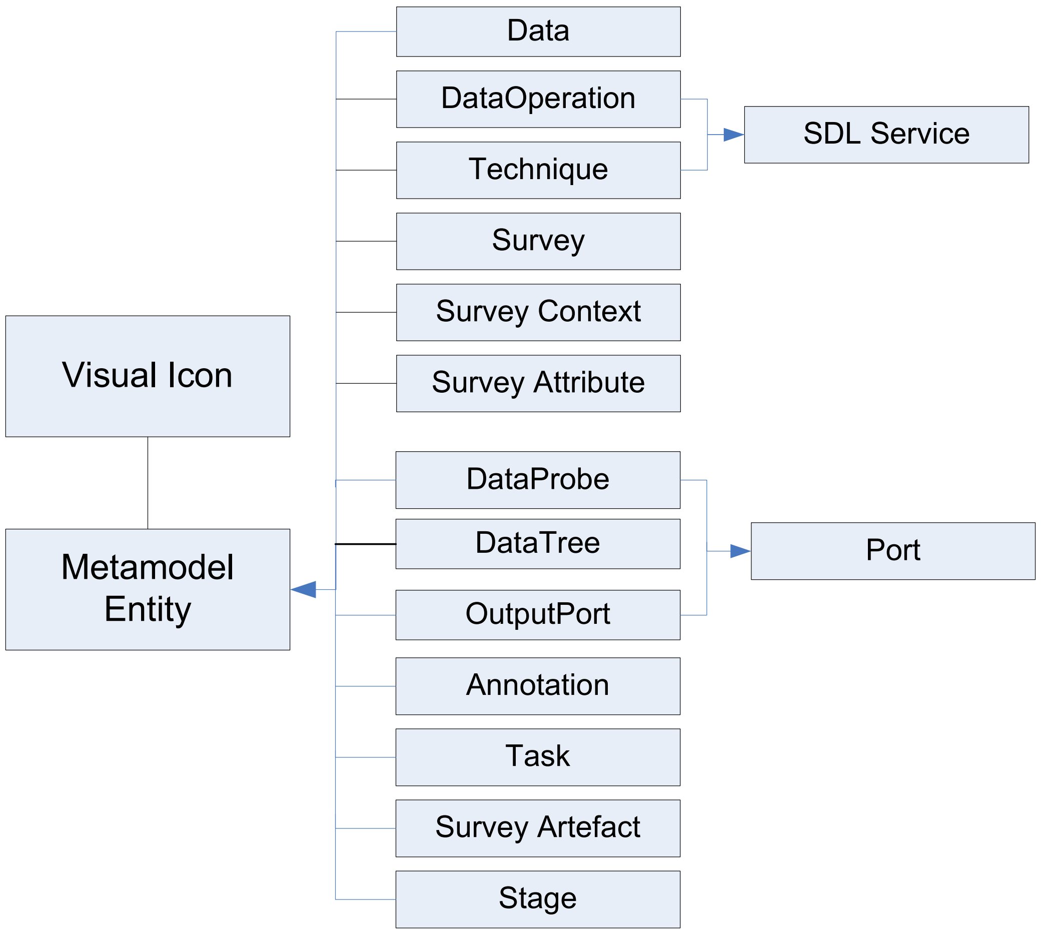


**Figure A.2 Metamodel entity types of SDL**

### A.2.1 Description of metamodel entities

| ***Type*** | ***Description*** | ***Attributes*** |
|---|---|---|
| Data | Represent statistical data and associated metadata | • Unique ID<br>• Base name<br>• Dataset reference<br>• Triple-S reference<br>• DataTree reference<br>• Data History |
| DataOperation | Represent data related non-analytical operations | • Unique ID<br>• Base name<br>• Input data list<br>• Output data list<br>• SDLService reference<br>• Operation definition |

| | | |
|---|---|---|
| Technique | Analytical technique | • Unique ID<br>• Base name<br>• Input data list<br>• Output data list<br>• SDLService reference<br>• Technique definition<br>• Execution order annotation |
| Survey | Represent a statistical survey | • Unique ID<br>• Base name<br>• Context list |
| Survey Context | Contextual space of a survey | • Unique ID<br>• Base name<br>• Survey attribute list<br>• Task reference list |
| Survey Attribute | Attributes associated with a contextual space | • Unique ID<br>• Base name<br>• Attribute annotation |
| DataProbe | Visualisation of statistical data | • Unique ID<br>• Base name<br>• Visualisation reference |
| DataTree | Structural representation of population to sample relationship | • Unique ID<br>• Base name<br>• Data node list |
| OutputPort | Outgoing interface for the outcomes of a statistical technique | • Unique ID<br>• Base name<br>• Presentation reference<br>• Visualisation reference |
| Task | Survey task | • Unique ID<br>• Base name<br>• Execution operator<br>• Artefact list<br>• Task list |
| Survey Artefact | Output/input need by a survey task | • Unique ID<br>• Base name<br>• Task list<br>• Input/output reference list<br>• Data reference<br>• Technique reference |

| | | |
|---|---|---|
| Stage | Survey process stage | • Unique ID<br>• Base name<br>• Task list<br>• Stage reference list |

**Table A.6** ***Description of metamodel entities***

## A.2.2 Metamodel entity attributes

*Unique ID*
This is a unique integer value assigned to every metamodel entity. This is a primary attribute used to provide a reference point to other entities that form relationships together.

*Base name*
A name given to a metamodel entity. The name does not have to be unique.

*Dataset reference*
A data file pointer. It can be in the forms of file system paths or data repository IDs.

*Triple-S reference*
A pointer to a Triple-S metadata file.

*DataTree reference*
A reference to a data tree that depicts a population-to-sample relationship.

*Data History*
Indexed list of data operations and techniques that have been used in the production of data. This enables users to track down production of statistical data.

*Input data list*
List of references to input data

*Output data list*
List of references to output data

*SDLService reference*
A pointer to an SDL service that actually implements a statistical technique/data operation.

*Operation definition*
Specification for a data operation. It consists of input/output data lists, operation parameters and an SDL service type.

*Technique definition*
Specification for a statistical technique. It consists of input/output data lists, technique parameters and an SDL service type.

*Execution order annotation*
1-based index indicates sequential order of execution.

*Context list*
List the unique IDs of contexts

*Survey attribute list*
List the unique IDs of contexts

*Task reference list*
List the unique IDs of tasks

*Attribute annotation*
Textual notations to indicate preceding conditions or information about a survey attribute.

*Visualisation reference*
Reference to a visualisation service (e.g. reference to a predefined visualisation service implemented as an SDL service) that will be used to present data.

*Presentation reference*
Reference to a non-visual presentation service (e.g. XSLT templates) that will be used to present data.

*Data node list*
List of references to first-degree siblings in a data entity tree which depicts a sample-to-population relationship.

*Execution operator*
Show the nature of entry conditions for a task. For example a task should not begin unless all preceding tasks are completed or a task should not wait for other related tasks to finish their goals but carry out its activities in parallel.

*Artefact list*
List the unique IDs of artefacts

*Task list*
List the unique IDs of tasks

*Input/output reference list*
List of survey artefacts that need outgoing/incoming inputs.

*Stage reference list*
List of references to stages that are linked by process flows.

## A.3 Semantic Layer – Topic Map Representation of SDL

In this section, we will present the semantic layer in a series of Topic Maps.
The SDL semantic layer is primarily organised by *topic maps*. Topic maps tie the underlying semantic of the metamodel to the real world statistical survey topics.

The SDL metamodel layer provides the views for tools and the SDL semantic layer provides a way of representing the abstraction of the SDL-based survey modelling. Functionally the metamodel layer prepares SDL diagrams for tool support and the semantic layer provides a means to make sense of the information represented by metamodel entities and relationships.

Due to the verbosity and huge amount of space required to present the semantic layer in XML Topic Maps 1.0 (XTM 2001). Our approach here is present them in an alternative visual format as shown in topic map papers (Pepper 2002, Mahabal et al. 2002)

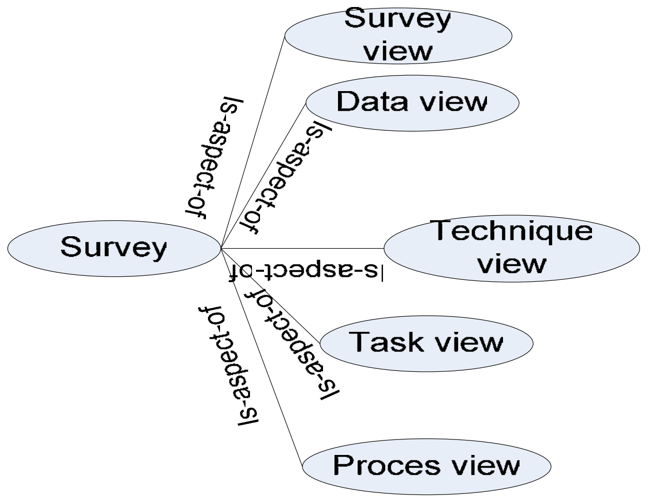


**Figure A.2 Survey aspects**

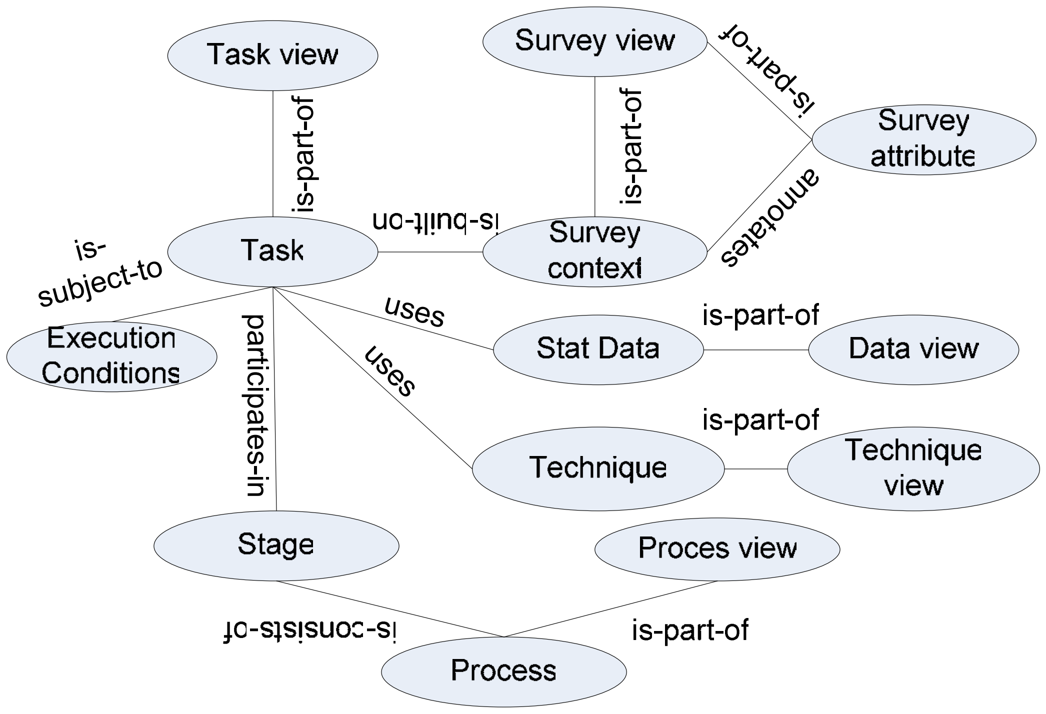


**Figure A.3 Inter-diagram relationships**

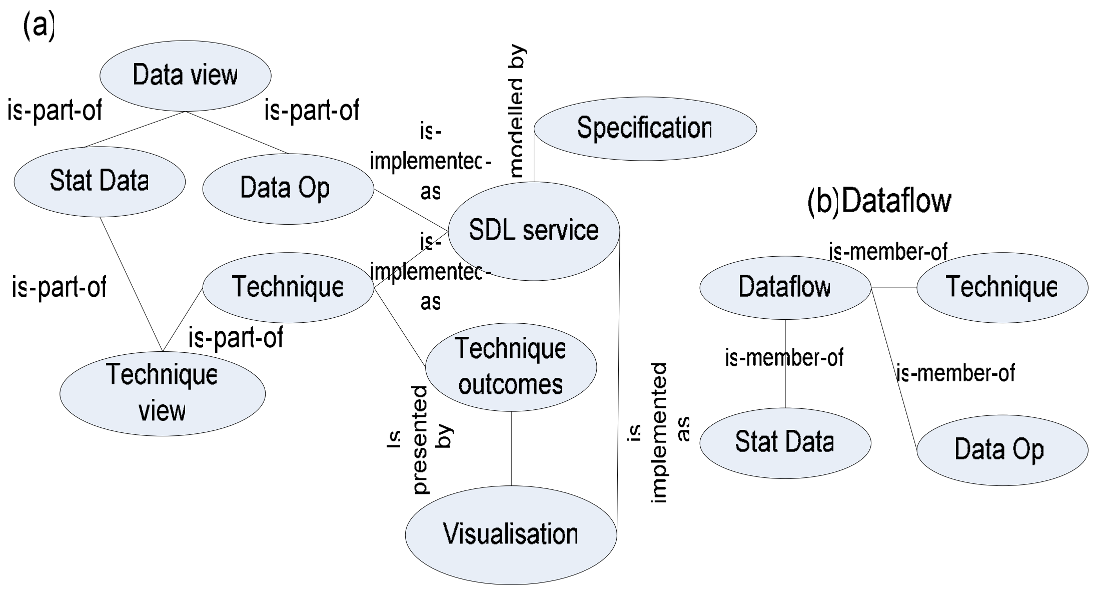


**Figure A.4 (a) Data-technique view (b) Dataflow constructs**

## Appendix B – Usability Testing

In this section we provide supporting documents and more detailed results for the usability testing that was discussed in Chapter 9. The usability testing was subject to approval (REF# 2005/339) by the *Human Participants Ethics Committee* of the University of Auckland's approval and all usability testing sessions adhered to the recommended guidelines of the committee.

A test for systematically examining SDL using a walkthrough approach consists of the following elements:

- Overview of SDL

The test subject is briefly introduced to SDL and some working examples are explained to give a chance to see SDL in action.

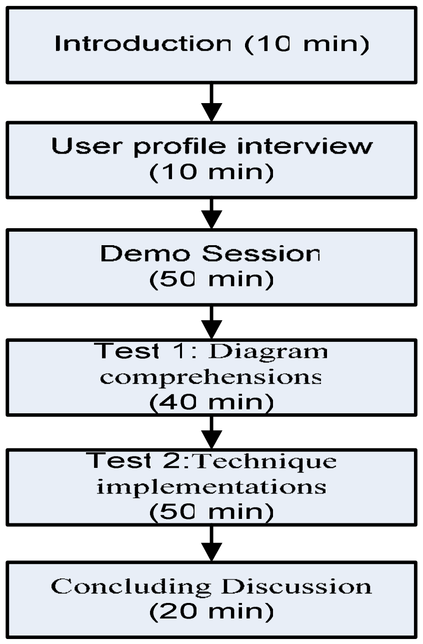
- Users Tasks

A list of tasks to be performed by the test subject will be given. As the walkthrough approach focuses on user–oriented solution finding, the tasks are intended to give the test subject opportunities to set a self- initiated exploratory path to complete the given tasks. Thus the tasks attempt to simulate the cognitive context of a survey researcher in practice rather than imposing fine-grained questions.

- User Awareness

A well-designed visual language should give users the sense of self-awareness. In other words, users should be able to tell whether they on the right track in terms of meeting final goals during the course of the using SDL. Insufficient user awareness can especially impact the usability of the diagrams, which are affected greatly by local changes, as late changes could imply a significant overhaul. Therefore the evaluation of SDL will examine user awareness throughout the testing session.

Figure B.1 shows a workflow for the usability testing that was followed in each session. For more detailed look at the content of the usability testing refer to the actual user testing script in section B 2.2.


Introduction (10 min)
User profile interview (10 min)
Demo Session (50 min)
Test 1: Diagram comprehensions (40 min)
Test 2:Technique implementations (50 min)
Concluding Discussion (20 min)
**Figure B.1 Usability test workflow**

## *B.1 Usability Test Results*

We had eight participants in total. They were all selected with help from the Department of Statistics, the University of Auckland. All participants had enrolled and completed at least one advanced 300-level paper on statistical surveys. Test results are from two types of sources.  The test participants were monitored throughout the testing sessions for their performance in two areas: SDL diagrams comprehension and statistical technique implementation using the prototype tool. The test participants also filled out questionnaire forms which asked for their opinions on various topics as shown in section B.2.2. This section mainly present tabulated raw data for the general discussion of qualitative feedback from the participants refer to section 9.2.6 Chapter 9.

## B 1.1 User Profile

*Education and Experience*

The user profile by experience and education are summarised in table B.1.

| *Statistics undergraduate* | *Statistics post-graduate* | *Professional statistics user* |
|---|---|---|
| 5 | 1 | 2 |

**Table B.1 User profile table**

We had one professional programmer/analysts who had extensive experience in statistical computing. One of our test participants was very familiar with the concept of using statistical surveys in assisting business decisions in professional settings. We also had one masters student in Statistics who was competent in both practical and theoretical aspects of statistical survey design. Remaining students were all undergraduate students completed at least one advanced course (Stat 340) in statistical surveys.

They were all had a good working knowledge of to popular statistical packages such as R and SAS and three of the participants were competent programmers of general purpose high-level languages such as Java and C.

## B 1.2 Testing Scores

***Test 1*** Diagram Comprehension (Task 2)

(0 Failed, 1 Average, 2 Good, 3 Excellent)

| **Diagram comprehension #** | **Survey Diagram** | **Task** | **Data** | **Technique** | **mappings** | **Average** | **Rounded** |
|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 2 | 2 | 2 | 2.2 | 2 |
| 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| 3 | 3 | 2 | 3 | 3 | 2 | 2.6 | 3 |
| 4 | 1 | 1 | 2 | 3 | 1 | 1.6 | 2 |
| 5 | 3 | 3 | 3 | 3 | 2 | 2.8 | 3 |
| 6 | 1 | 1 | 2 | 3 | 1 | 1.6 | 1 |
| 7 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| 8 | 3 | 2 | 3 | 3 | 3 | 2.8 | 3 |
| **Average by diagram** | 2.375 | 2.25 | 2.625 | 2.875 | 2.125 | | |

*Process diagram was not part of the user testing

| | |
|---|---|
| **Survey Diagram** | **Survey Diagram** |
| **Task** | **Survey Task Diagram** |
| **Data** | **Survey Data Diagram** |
| **Technique** | **Survey Technique Diagram** |
| **mappings** | **Diagram mappings and SDL tools** |

**Table B.2 User profile table**

Performance checkpoints
A test subject: (0(Failed) -3(Excellent))

- is able to understand the survey diagram and identity the major survey semantics.
- can modify the survey diagram to express his/her design concerns.
- can explain all aspects of graphical icons used in the survey.
- can implement a correct solution.

***Test 2*** Express and implement fundamental survey techniques (Task 3)

(0 Failed, 1 Average,
2 Good, 3 Excellent)
**Test 2**

| **#** | | **Task1** | **Task 2** | **Task 3** | **Task 4** | **Average** | **Rounded** |
|---|---|---|---|---|---|---|---|
| | 1 | 1 | 1 | 1 | 0 | 0.75 | 1 |
| | 2 | 3 | 3 | 3 | 3 | 3 | 3 |
| | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| | 4 | 2 | 2 | 2 | 1 | 1.75 | 2 |
| | 5 | 2 | 2 | 2 | 1 | 1.75 | 2 |
| | 6 | 0 | 1 | 0 | 0 | 0.25 | 0 |
| | 7 | 3 | 3 | 3 | 2 | 2.75 | 3 |
| | 8 | 2 | 2 | 2 | 1 | 1.75 | 2 |
| **Task Average** | | 2 | 2.125 | 2 | 1.375 | 1.875 | 2 |

**Table B.3 User profile table**

## B 1.3 User Observation

Ideally we would like all user feedbacks to be analyzed to the level of certainty as observable quantitative data however so much part of this user testing is based on contextual inquiry which is open to subjective interpretations. Unlike the investigator monitored task-based ratings in the previous section, qualitative feedbacks from the users as shown in section B.2 are more subjective and context dependent.
In this section we do not present inferences we draw from user feedbacks as they were discussed in Chapter 9 but show types of feedback we obtained from users in various categories.

During the entire process, we employed an observation guide for the inquiry so that some of the qualitative feedbacks can be grouped into pre-set patterns rather than giving the user explicit choices first. This approach was taken as we wanted capture as much information (more than yes and no) as possible during given times with limited resources.

For instance the second question of the task 1, we do not ask our users few multiple options to choose from but ask "What does the tool do well in your opinion? ", to learn more about their observations.

Throughout the task-based walkthrough we deliberately checked how users react to proposed problems that were open-ended in nature. Question 4 of the task 3 is a good example of an open-ended task that requires users to operate in the mental state of a statistician in the survey process.

## B 1.4 Summary of Observation Guide

Users are given a survey scenario which is derived from the New Zealand National Survey of Crime Victims in 2001 (NZCS 2003). The detail of the scenario is found in Chapter 8 and they were shown the same SDL diagrams used in the chapter.

***Task 2: Diagram Comprehension***

Q1. Survey Diagram
Users must be able to identify all survey contexts and their relevance to the survey. Elaborate on a survey context using associated survey attributes. Identity and explain consensus forming processes found in the diagram.

Q2. Survey Task Diagram
Given two task models, users must able to present how each task model is structure and executed. Understanding of task operators are also tested.

Q3. Survey Data Diagram
Users must be able to answer the nature of sampling methods employed in the diagram. They should be able to explain how statistical metadata is related to statistical data produced by the sampling methods. Clarity of sample-to-population relationship is tested.

Q4. Survey Technique Diagram
They should be able to elaborate on a data flow metaphor used in the diagram, the chain of techniques, and technique outputs.

Q5. Inter-diagram Relationships
They should understand how separate diagrams can exist in relationship. They should be able to start from the survey diagram and find its relevance in the task diagrams vice versa.

***Task 3: Express and Implement Fundamental Survey Techniques***

Q1. The first phase of this task is to lay down appropriate visual icons and connect them using correct connectors. Then the visual icons should be mapped appropriate resources with correct specification for the visualisation technique. They should be able to use a visual layer which is accessible from a technique output port.

Q2. & Q3. Users must be able to formulate appropriate a diagram structure to express the analytic task then make necessary bindings to obtain results from an executable diagram.

Q4. This task involves an analytical component which is more open-ended. So that users should find and utilise an appropriate technique from a service registry. At the end of exercise they may generate a web service based on their solution.

## *B.2 Supporting Documents*

### B 2.1 USABILITY TESTING INSTRUCTIONS AND QUESTIONNAIRE

We would appreciate it if you would take the time to answer the following questions. The questions are really just a guide to the area of the tool we would like to obtain feedback on but please feel free to write down any comments you have on your impressions of the visual language and the tool implementation.

This end-user evaluation test contains examples of the type of essential statistical survey techniques and communication aids which may be encountered in survey design and management.

This test session is design to work in conjunction with an instructor led tutorial on SDL and SDL tools. The survey process varies for most survey requirements and the number of published statistical surveys is enormous. The techniques included in this test are based on our experience from consulting with a statistician and existing surveys. This is only to give you a basic flavour of what you should expect in designing and studying a statistical survey.

While software tools and visual notations used in this test are useful to express and implement various aspects of a survey, they are only in early pre-beta stage. So your feedbacks on such areas:

- Relevancy to a given task
- Fluidity in implementing requirements
- Fundamental design issues
- Economic aspect of tools and visual notations
- General ease of use

will be most valuable.

Take time to read through instructions given in this questionnaire and feel free to ask any questions and suggest your own ideas.

## 1. Survey literacy

Skillset Profile

Do you understand the concept of a statistical survey design?

Do you understand the concept of a statistical survey process?

## B 2.2 User Testing Guide

### Task 1: Survey Design Language (SDL) Preview (Instructor led demonstration)

### Task 2: Diagram Comprehension

**Survey Diagram**
Survey diagrams provide an overview of a survey in terms of contexts and attributes of those contexts. The main purpose of this diagram is to support survey requirement engineering and consensus forming. This diagram can be used casually both by experts and non-experts at early stages of the survey process to facilitate interactive brainstorming.

When viewing a survey diagram, it is recommended to identify survey contexts first then traverse their attributes. Note that attributes connectors may have important descriptive comments attached to them to guide the thought flow within a particular context.

Consider the survey diagram loaded in Pounamu and follow the instructions given as one way to understand the diagram.
(>> Survey diagram for 2001 National Crime Survey)

**Survey Task Diagram**
Survey task diagram describes abstractions of survey activity using a hierarchical task model. This abstraction process involves organizing survey activities in a coherent representation and then inferring hierarchical properties of the modelled activities.

A task model contains one top level goal and may contain sub tasks that support the top level goal. Generic survey activities can be shown using task models. During survey design, users could also refine existing task models to turn them into explicit patterns for a particular survey. For example, a single task can be broken down into sub tasks and task related outcomes and statistical operations can be shown by attaching survey entities.

Consider the task diagram loaded in Pounamu and answer given questions.

Please identify all relevant information from the diagram, such as a main goal of the task model, sub tasks, and explain all their attributes and relationships.

**Survey Data Diagram**
Survey data diagrams capture the main concepts and techniques involved in the sampling and data collection process. These include raw datasets, questionnaires, sample-to-population relationships sampling operations and etc.

In the survey process, we model the real-world attributes of the population from sampled data. Therefore for any valid inferences based on the sample should consider how the sample is collected and define technical guidelines that make sense in the data analysis stage.

As shown in the example survey data diagram, a dataset of the survey encompasses its raw data table, questions asked to obtain the raw data (Triple-S), the values of those questions have (Triple-S) and its relationship to the population.

A survey data operation may have one or more dataset inputs and one or more dataset outputs. The survey data operation is simply an action such as random sampling that is performed during the data collection stage.

Consider the survey data diagram loaded in Pounamu and answer given questions.
(>> Sampling diagram for the 2001 survey)

Please identify all relevant information from the diagram, such as survey metadata, sample-to-population relationship, and explain all their attributes and relationships.

**Survey Technique Diagram**
Survey diagram depict the data analysis stage which typically involves statistical inferences on datasets and representing data to study underlying relationships between various variables. The various activities performed during the stage are abstracted as a technique. A common form of simple technique can be represented by a graphical icon with one/more input and output datasets forming data flows and outcomes of the technique such as graphical plots or numerical values. A survey technique diagram can represent the aggregation of statistical techniques as part of one technique.

Consider the survey technique diagram loaded in Pounamu and answer given questions.
(>> Techniques used in the 2001 survey)

Please identify all relevant information from the diagram, such as the chain of techniques, techniques outputs, data flows and all their attributes and relationships.

**Diagram mappings and SDL tools**
The individual diagrams can useful for graphically depicting their target domain. However to present a complex survey effectively, it is necessary to bring multiple aspects of a survey to form a unified model. SDL software tools allow the integration process by giving users ability to specify inter-diagram relationships as shown in the demonstration.

Survey diagram to survey task diagram relationships integrate high-level operational specifications of a survey in a complete fashion. While survey technique/data diagram to survey task diagram relationships show how survey tasks are implemented in practice.

Those relationships make it possible to index surveys, automate document generation, and create software services.

A SDL diagram can play dual role of purely graphical diagram and functional software tool. As each diagram represents a particular perspective of a survey, it can also have a potential to be a functional software tool user interface for diagram's target perspective.

SDL diagrams can be mapped to external physical entities or software services and act as an abstraction layer to control them.

**Task 3: Express and implement fundamental survey techniques**
In this test a number of fundamental techniques and characteristics common to many statistical surveys will be introduced. We have used NZ National Crime Survey 2001 as a reference survey and distilled many aspects of the survey in the following exercises.

Give examples of a survey technique diagram. Your examples should involve the following four techniques:

i) Graphical plots to explore the relationship between socio-economic status and victimization.

ii) A chi-square test for association to study if non-high school graduates are strongly represented in the victimization data.

iii) A single predictor linear regression analysis to implement a method for inference for two exploratory variables:
- 1998 victimization data
- 2001 victimization data

iv) The national crime prevention program has been operating to provide practical training for potential victims. The given dataset contains individuals for a control group and a treatment group. Use multiple linear regression methods to estimate of the program on victimization data. If differences in the two distributions are not significant, you may filter out outliners. Support your model with available techniques.

(You may generate a web service once a complete diagram has been created.)

*(General questions – answer sheet)*

**Task 2**
Is it easy to model the survey with the tool?

What does the tool do well in your opinion?

Are there any notations that should be made clearer for the user? How? (e.g. more user interaction behaviours, colour codes)

Is there anything the tool does not let you do that you would like to?

What other information of views of information should the tool display?

Are there any improvements you would like to see in the tool?

**Task 3**

In this test a number of fundamental techniques and characteristics common to many statistical surveys will be introduced. We have used NZ National Crime Survey 2001 as a reference survey and distilled many aspects of the survey in the following exercises.

Would you like to use the notations if you had a large survey project to do? Why/Why not?

Are there any concepts/ideas in the SDL diagrams that are difficult to comprehend?

Is there anything the diagram does not let you understand that you would like to?

What other information or views of information should the diagram convey?

Are there any improvements you would like to see in the diagram?

## General Comments

Any general comments you have on the concepts of the tool and notations.

**Thank you.**

## *PARTICIPANT INFORMATION SHEET – USABILITY TESTING*

Project title: **Visual Language and environment for statistical survey design language**

Researcher name: Chul Hwee Kim

To: Students

My name is Chul Hwee Kim I am an MSc student at The University of Auckland conducting research into visual methods to support statistical surveys. As a participant of this testing session your feedback will be recorded in response to a number of questions. The questionnaire you are asked to complete will help us gauge the efficiency and effectiveness our research.

While I would appreciate any assistance you can offer me, your participation is voluntary and will have no effect on your course grade or course participation in any way.

The questionnaire you are asked to complete is anonymous and none of the information on it will identify you personally. Once completed your questionnaire information cannot be withdrawn. The individual questionnaire responses will summarised and analysed and this summary information may be used both to improve our research outcomes that we are developing and report on the findings of the study. The questionnaire data will be held in secure storage for six years and then destroyed. A summary of the results of the testing and any resulting publications will be made available to you on request

| **Researcher name and contacts** | **Supervisor name and contacts** | **HOD name and contacts** |
|---|---|---|
| Chul Hwee Kim | John Hosking<br>john@cs.auckland.ac.nz | John Hosking<br>john@cs.auckland.ac.nz |

For ethical concerns contact: The Chair, The University of Auckland Human Participants Ethics Committee, Office of the Vice Chancellor, Research Office, Level 2, 76 Symonds Street, Auckland. Tel: 373-7599 extn. 87830.

APPROVED BY THE UNIVERSITY OF AUCKLAND HUMAN PARTICIPANTS ETHICS COMMITTEE ON …(date)... TO …(date)…FOR ……(3) YEARS REFERENCE NUMBER 200../…

5f5ba4080e8fdb677aec16839a3e3452

# Appendix C - Tool Tutorial

This is a brief tutorial on how to get started with the SDL prototype tool. Our tool has a range of diverse features and this tutorial only covers the elementary working of the tool. The following tasks are presented in this tutorial:

1. Authoring a survey technique diagram and perform a diagram execution.
2. Authoring a task diagram.
3. Make a task-to-technique inter-diagram mapping.

## *C. 2 Prerequisites*

To do statistical computations, we need an R computation server (Rserve) with TCP/IP interface running at the remote/local server. Detailed tutorials on setting up the R server are found at http://stats.math.uni-augsburg.de/rserve/

Apache Axis must be set up and running on your machine or you must have an access to Axis for web service generation and hosting. Our tool runs inside of Pounamu metatool environment. Pounamu usage is not presented here for more information on Pounamu contact

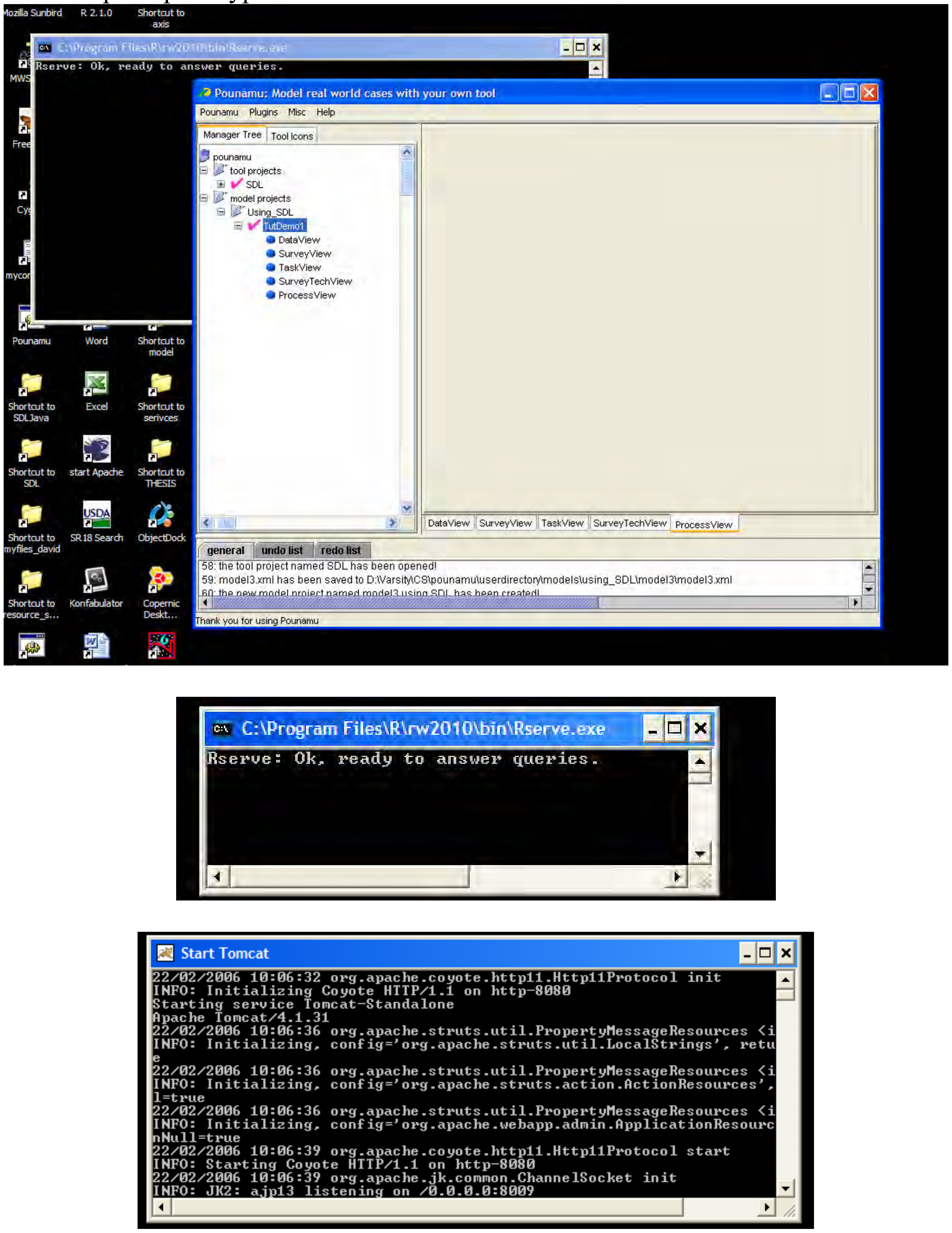
Prof. John Hosking
Department of Computer Science
The University of Auckland
Private Bag 92019
Auckland
New Zealand
Phone: +64-9-3737599 ext 88297
FAX: +64-9-3737453
Email: john@cs.auckland.ac.nz

## C.3 Working with a Survey Technique Diagram

### *Task*

We want to compose a survey technique diagram which depicts a visualisation technique to investigate the distribution of undergraduate GPA against year of study.

1. Start up the prototype tool.



```
58: the tool project named SDL has been opened!
59: model3.xml has been saved to D:\Varsity\CS\pounamu\userdirectory\models\using_SDL\model3\model3.xml
60: the new model project named model3 using SDL has been created!
```

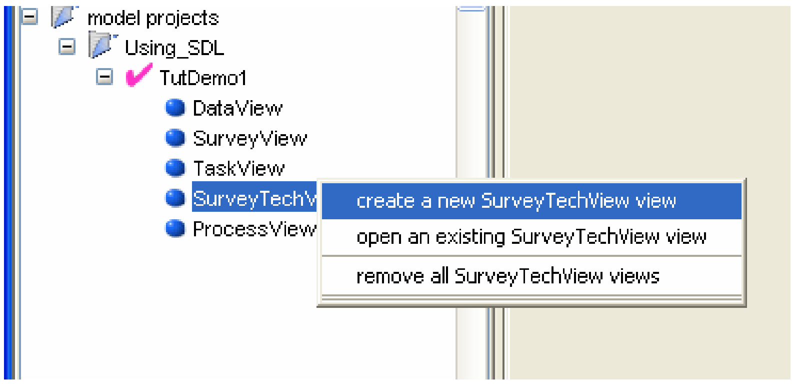
Thank you for using Pounamu

```
Rserve: Ok, ready to answer queries.
```

```
22/02/2006 10:06:32 org.apache.coyote.http11.Http11Protocol init
INFO: Initializing Coyote HTTP/1.1 on http-8080
Starting service Tomcat-Standalone
Apache Tomcat/4.1.31
22/02/2006 10:06:36 org.apache.struts.util.PropertyMessageResources <i
INFO: Initializing, config='org.apache.struts.util.LocalStrings', retu
e
22/02/2006 10:06:36 org.apache.struts.util.PropertyMessageResources <i
INFO: Initializing, config='org.apache.struts.action.ActionResources',
l=true
22/02/2006 10:06:36 org.apache.struts.util.PropertyMessageResources <i
INFO: Initializing, config='org.apache.webapp.admin.ApplicationResourc
nNull=true
22/02/2006 10:06:39 org.apache.coyote.http11.Http11Protocol start
INFO: Starting Coyote HTTP/1.1 on http-8080
22/02/2006 10:06:39 org.apache.jk.common.ChannelSocket init
INFO: JK2: ajp13 listening on /0.0.0.0:8009
```

**Figure C. 1 Main application screenshot with Rserve and Tomcat running at the back**

2. Go to the tool management tree panel and create a survey technique view.



model projects
Using_SDL
TutDemo1
DataView
SurveyView
TaskView
SurveyTechV
ProcessView
create a new SurveyTechView view
open an existing SurveyTechView view
remove all SurveyTechView views

**Figure C. 2 Creating a new survey view**

3. You can choose graphical icons or connectors from the tree as shown in Figure C.3

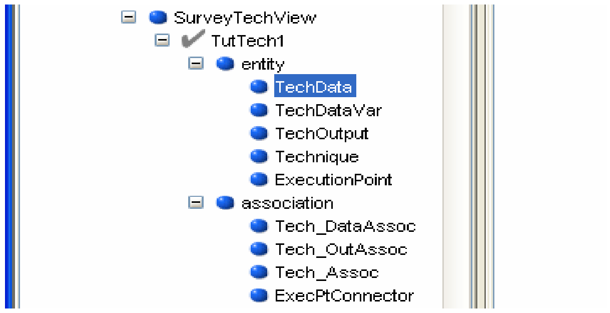


**Figure C. 3 Selecting a visual icon**

4. We start with the 'TechData1' icon.

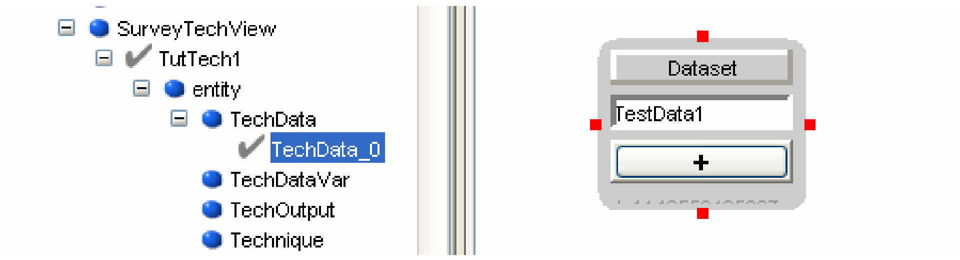


**Figure C. 4 Authoring a survey view diagram**

5. Layout all the visual icons and connect them with appropriate connectors and annotate them with descriptions.

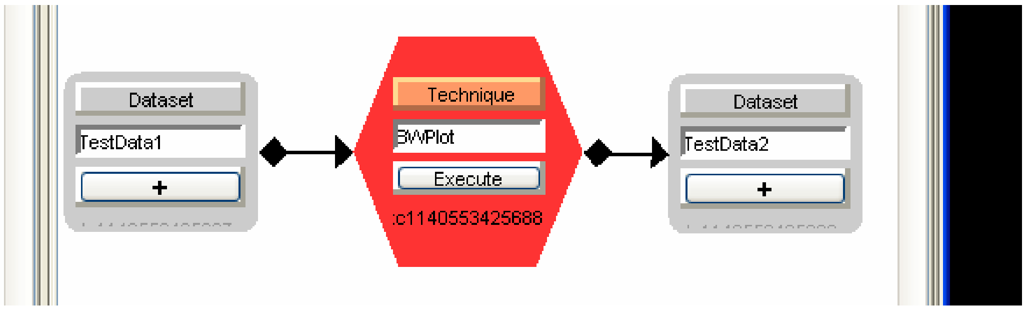


**Figure C. 5 Connecting the visual icons to depict the visualisation technique**

6. Add a technique output port for the 'BWPlot' technique icon.

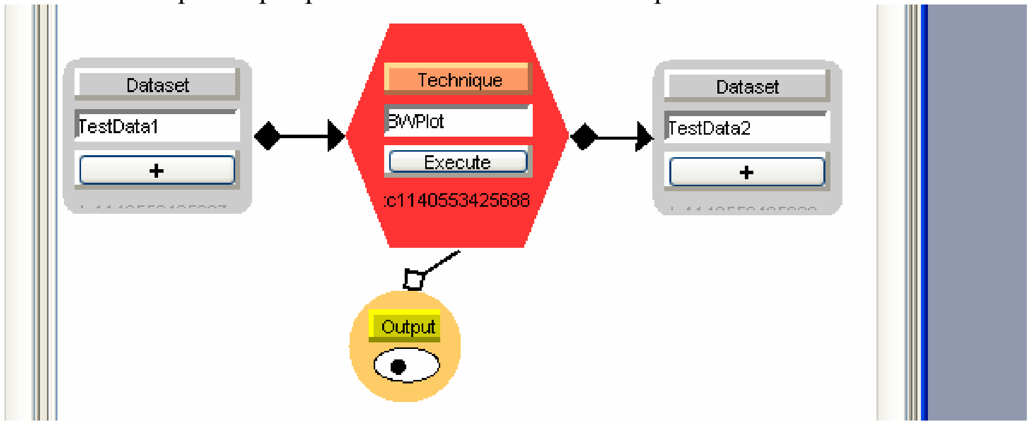


**Figure C. 6 Connecting a technique output port to the 'BWPlot'**

6. From the tool window's context menu, select the items 'add menu handles'. To turn on the diagram's menu handlers.

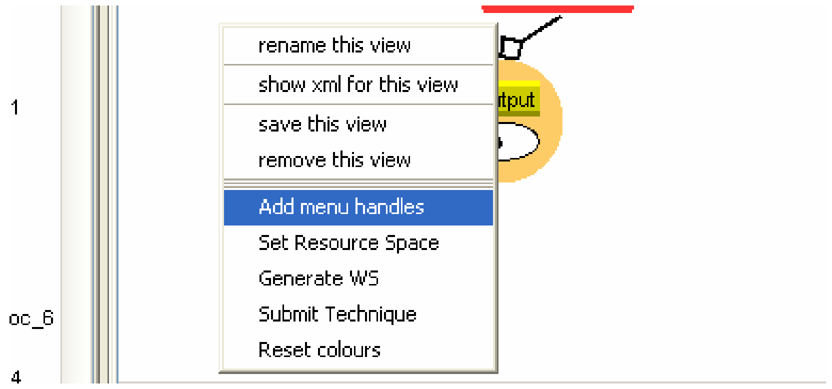


**Figure C. 7 Activate menu handlers**

7. Bind a data file to the 'TestData1' icon. Find a data file and view it using the built-in data table of the mapping interface as shown in Figure C.9. You can check out binded data variables by clicking on the '+' button. This will generate a collapsible data variable tree as shown in Figure C.10.

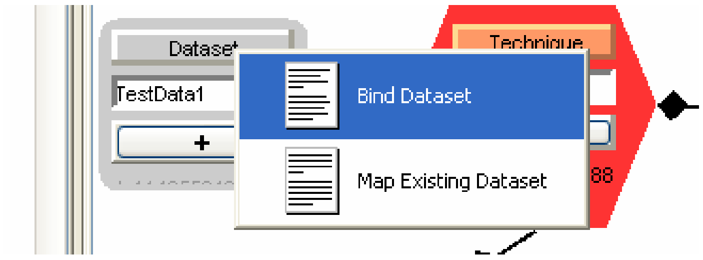
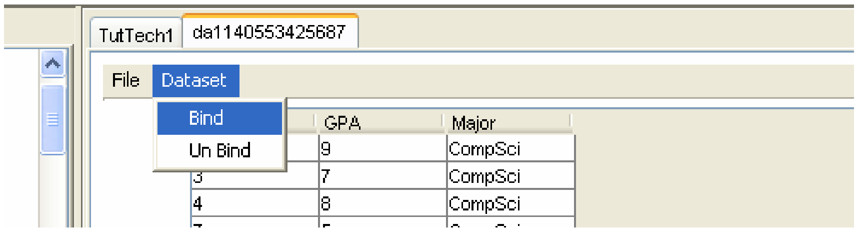


**Figure C. 8 Binding the visual icon to a data file**



| | GPA | Major |
|---|---|---|
| | 9 | CompSci |
| 3 | 7 | CompSci |
| 4 | 8 | CompSci |

**Figure C. 9 Binding the data file to the visual icon**

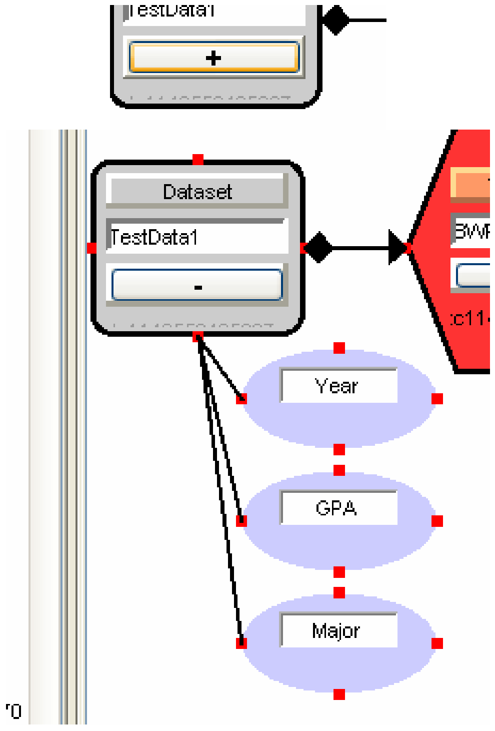


**Figure C.10 Collapsible data variable tree**

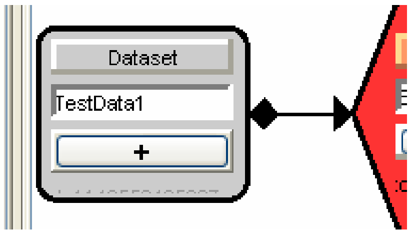


Figure C. 10 Layout change to reflect the status of the visual icon

8. Bind a technique to the 'BWPlot' icon. The mapping interfaces as shown in Figure C.12 and C.13 are generated. The first tab of the interface prompts the user to select a statistical technique form the pool of available SDL services. Then the user is asked to enter the specification for the selected service (some of them such as dataflow specification are auto-filled).

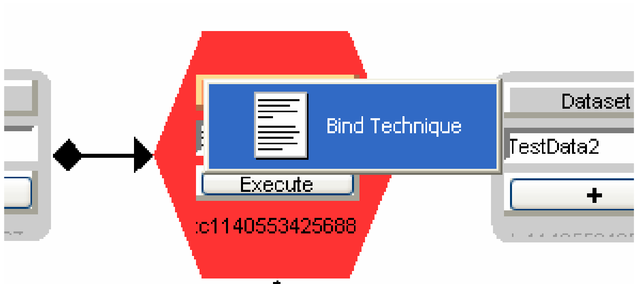


Figure C. 11 Technique binding

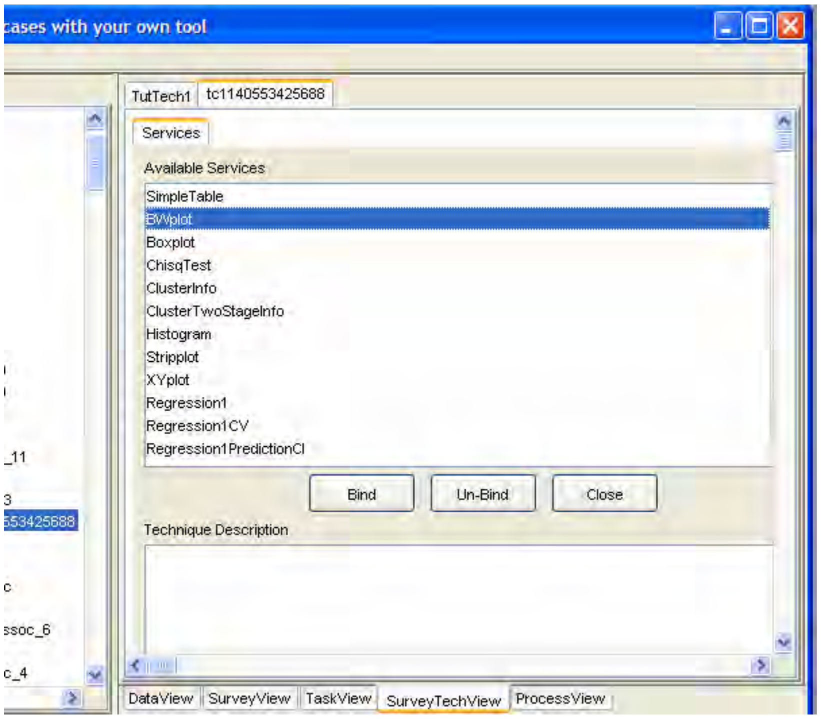
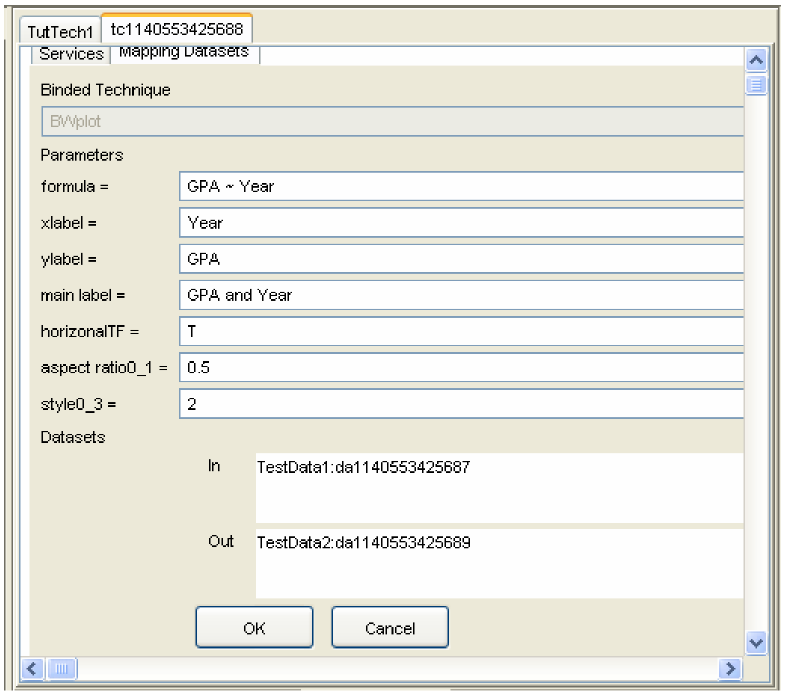


Figure C. 12 Dynamically generated mapping interfaces



TutTech1 | tc1140553425688

Services | Mapping Datasets

Binded Technique

BWplot

Parameters

| | |
|---|---|
| formula = | GPA ~ Year |
| xlabel = | Year |
| ylabel = | GPA |
| main label = | GPA and Year |
| horizonalTF = | T |
| aspect ratio0_1 = | 0.5 |
| style0_3 = | 2 |

Datasets

| | |
|---|---|
| In | TestData1:da1140553425687 |
| Out | TestData2:da1140553425689 |

OK | Cancel

Figure C. 13 Mapping interface for specifying technique specification

9. Screenshot of the survey technique diagram with binded data and technique icons. (Note the layout changes).

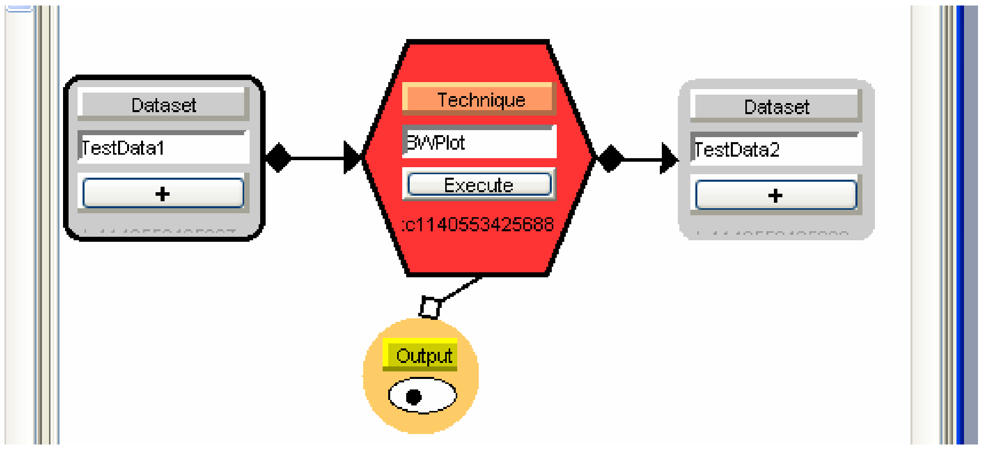


Figure C. 14 Survey diagram with binded data and technique icons

10. Execute the diagram by pressing ‘Execute’ button of the ‘BWPlot’ icon. Upon successful execution, all icons turn to green. Not you can access the visualisation of the input data by opening up the output ports’ visualisation view as shown in Figure C.15. Figure C.16 show the visualisation view.

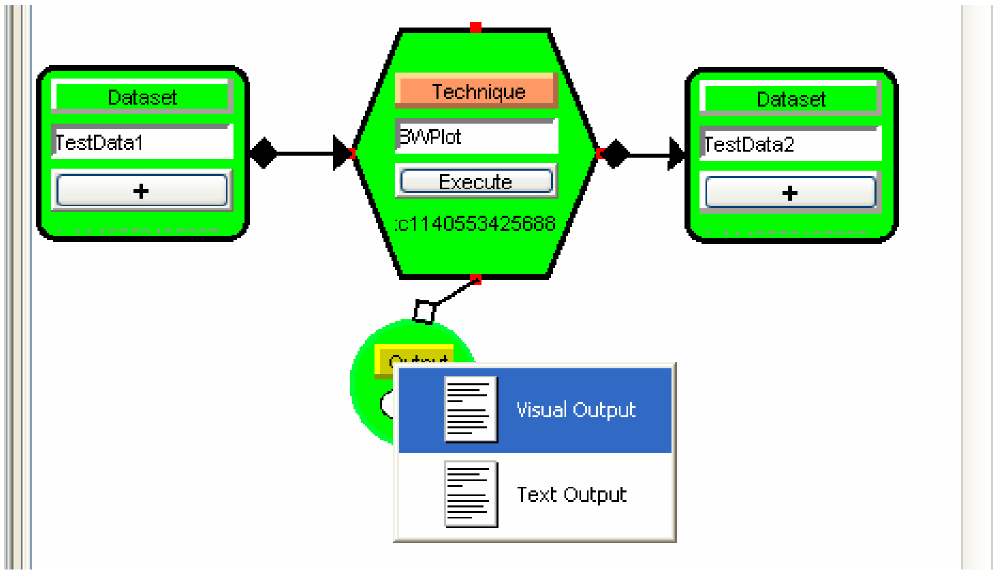


**Figure C. 15 Diagram execution and visualisation output**

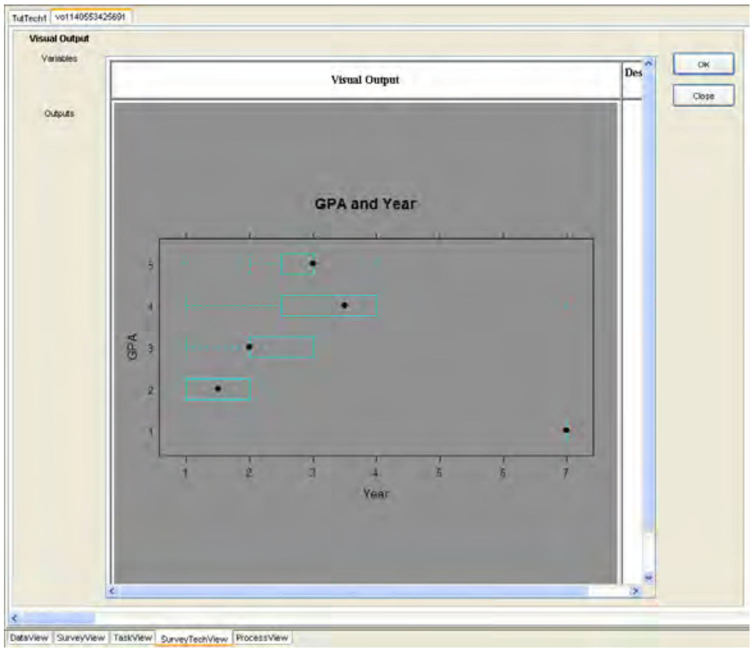
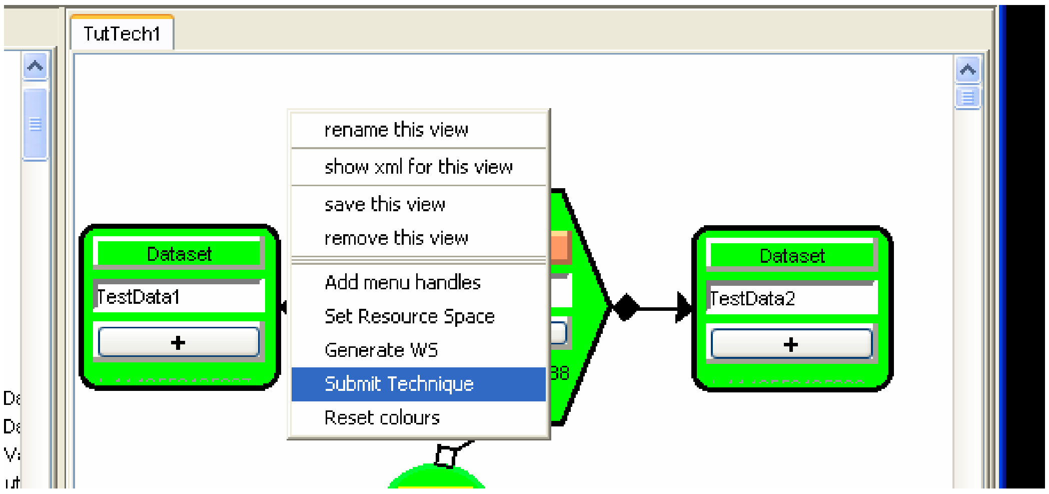


**Figure C. 16 Visualisation output of the diagram (BWPlot)**

11. When the diagram meets all the initial requirements then the diagram can be submitted to the model repository as shown in Figure C.17. Then we can generate a web service based on the submitted model as shown in Figure C.18 – C.22.



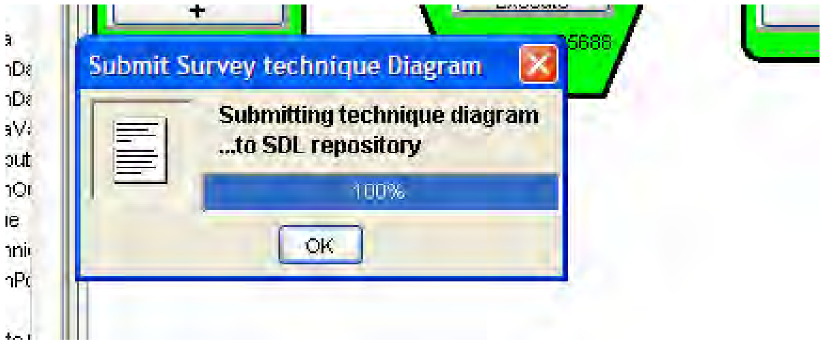


Figure C. 17 Model submission

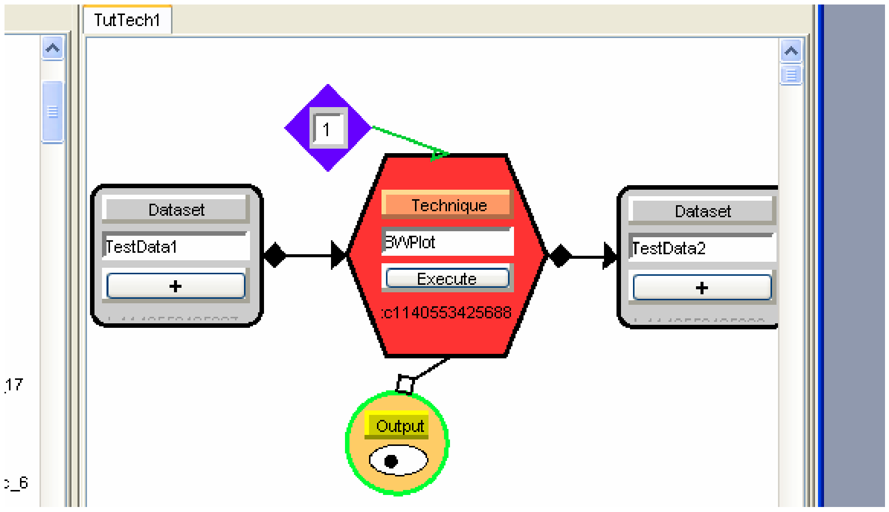


Figure C. 18 Assign an execution order

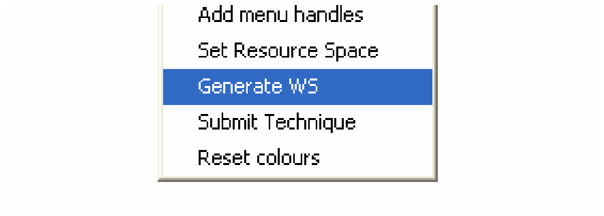
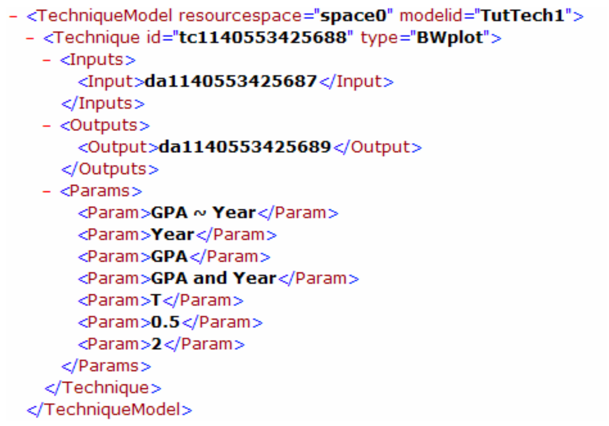


**Figure C. 19 Generate a web service**



```
- <TechniqueModel resourcespace="space0" modelid="TutTech1">
  - <Technique id="tc1140553425688" type="BWplot">
    - <Inputs>
        <Input>da1140553425687</Input>
      </Inputs>
    - <Outputs>
        <Output>da1140553425689</Output>
      </Outputs>
    - <Params>
        <Param>GPA ~ Year</Param>
        <Param>Year</Param>
        <Param>GPA</Param>
        <Param>GPA and Year</Param>
        <Param>T</Param>
        <Param>0.5</Param>
        <Param>2</Param>
      </Params>
    </Technique>
  </TechniqueModel>
```

**Figure C. 20 Model specification**

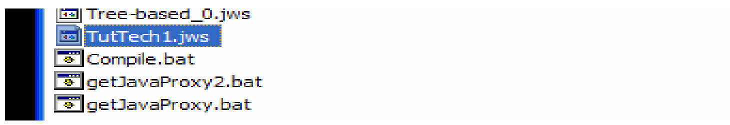
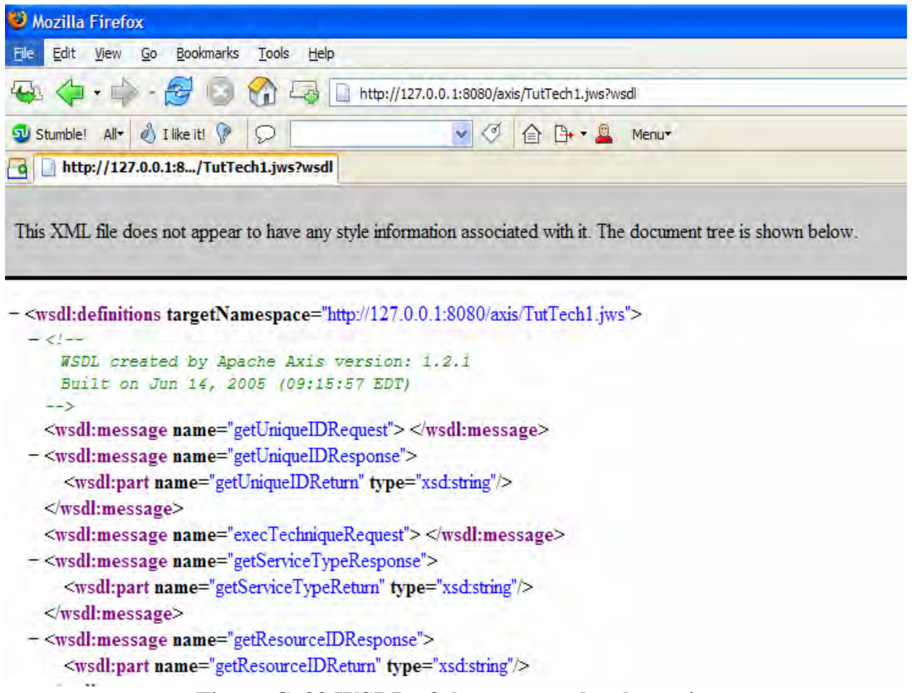


**Figure C. 21 Generated Java code (Axis hosted)**


Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://127.0.0.1:8080/axis/TutTech1.jws?wsdl
Stumble! All I like it! Menu
http://127.0.0.1:8.../TutTech1.jws?wsdl

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
- <wsdl:definitions targetNamespace="http://127.0.0.1:8080/axis/TutTech1.jws">
  - <!--
      WSDL created by Apache Axis version: 1.2.1
      Built on Jun 14, 2005 (09:15:57 EDT)
    -->
    <wsdl:message name="getUniqueIDRequest"> </wsdl:message>
  - <wsdl:message name="getUniqueIDResponse">
      <wsdl:part name="getUniqueIDReturn" type="xsd:string"/>
    </wsdl:message>
    <wsdl:message name="execTechniqueRequest"> </wsdl:message>
  - <wsdl:message name="getServiceTypeResponse">
      <wsdl:part name="getServiceTypeReturn" type="xsd:string"/>
    </wsdl:message>
  - <wsdl:message name="getResourceIDResponse">
      <wsdl:part name="getResourceIDReturn" type="xsd:string"/>
```

**Figure C. 22 WSDL of the generated web service**

## C.4 Working with a Task Diagram

1. Create a task view using the management tree.

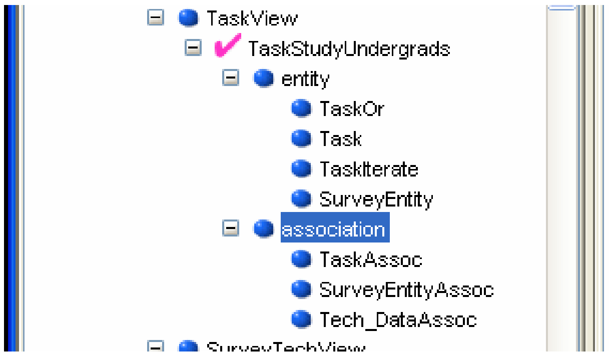


Figure C. 23 Task view creation

2. Lay out tasks required.

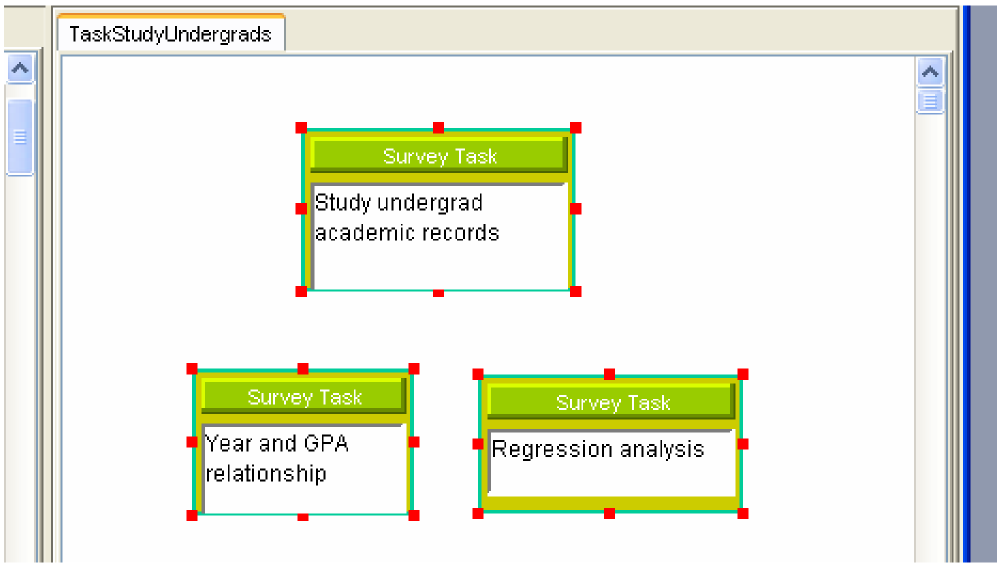


Figure C. 24 Survey tasks

3. Express hierarchical orders.

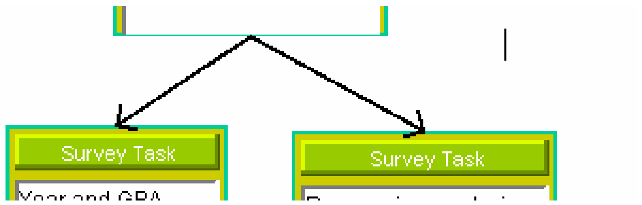


Figure C. 25 Tasks in hierarchical relationships

4. Completed task model with the survey artefact which will be bound to the technique created in the previous task.

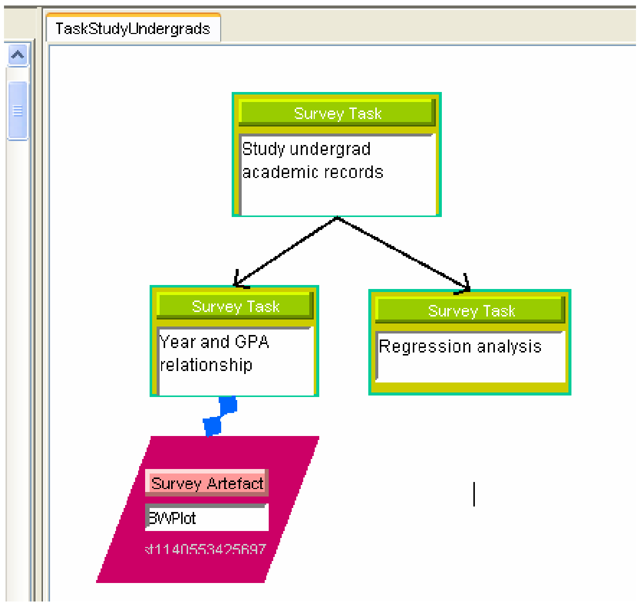


**Figure C. 26 Task model with a survey artefact**

## C.5 Task-to-Technique Inter-Diagram Mapping

1. Activate context menus for the visual icons on the diagram.

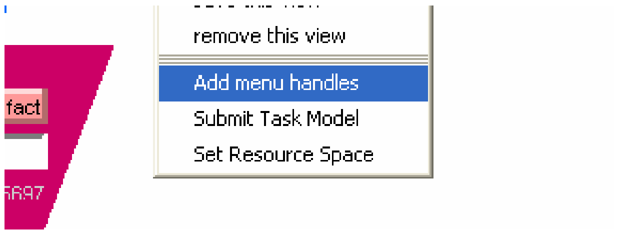


**Figure C. 27 Activating menu handlers**

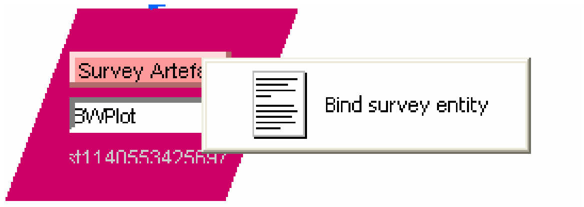
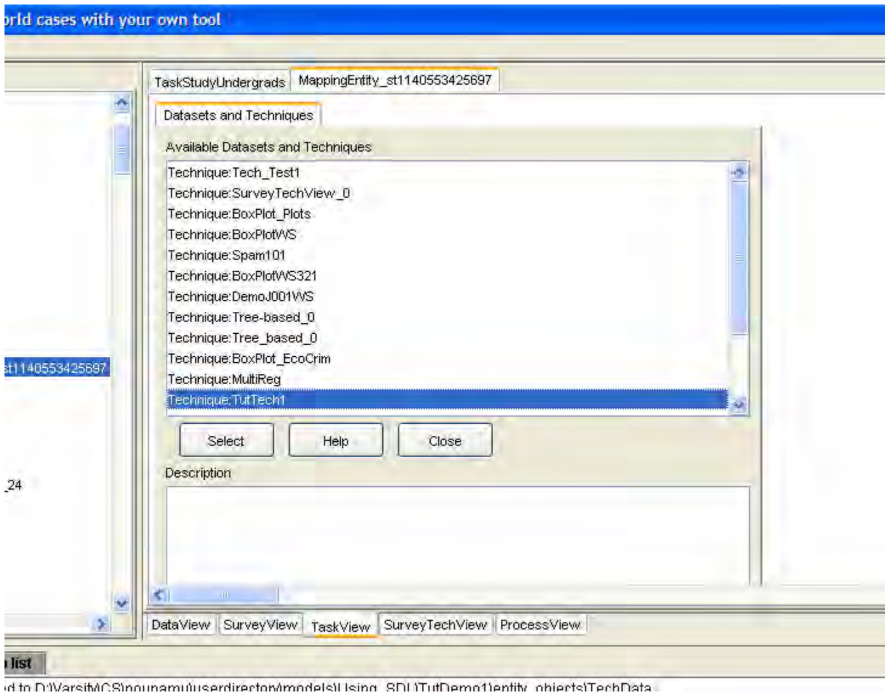


**Figure C. 28 Binding the survey artefact to the submitted technique model**

2. Mapping interface shows available survey artefacts, we can see that the technique that we just created (highlighted list item).



Technique:Tech_Test1
Technique:SurveyTechView_0
Technique:BoxPlot_Plots
Technique:BoxPlotWS
Technique:Spam101
Technique:BoxPlotWS321
Technique:DemoJ001WS
Technique:Tree-based_0
Technique:Tree_based_0
Technique:BoxPlot_EcoCrim
Technique:MultiReg
Technique:TutTech1

Figure C. 29 Mapping interfaces for the survey artefact

2. To find out what the technique is about. We can click on the 'Help' button, this will generate help documents for the selected technique dynamically and start up a web browser to display them as shown in Figure C.30.

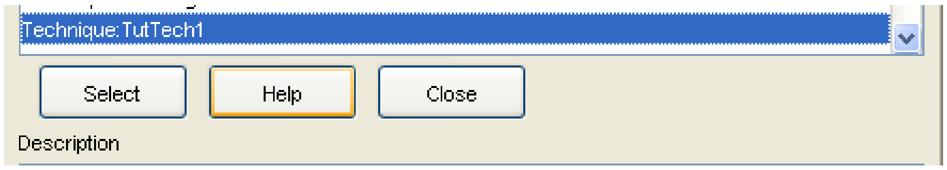
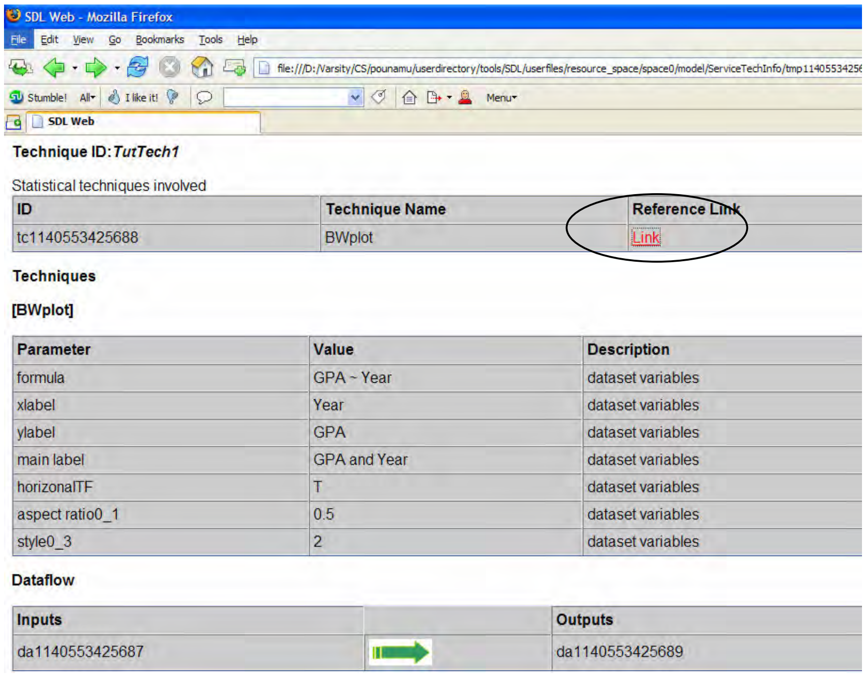


**Figure C. 30 Looking up information on the technique 'TutTech1' which we created in the last task.**

3. The binded survey artefact now has a new layout. An explicit inter-diagram relationship between the technique diagram and the task diagram has been made at both visual and metamodel layers.



Technique ID: *TutTech1*

Statistical techniques involved

| ID | Technique Name | Reference Link |
|---|---|---|
| tc1140553425688 | BWplot | Link |

**Techniques**

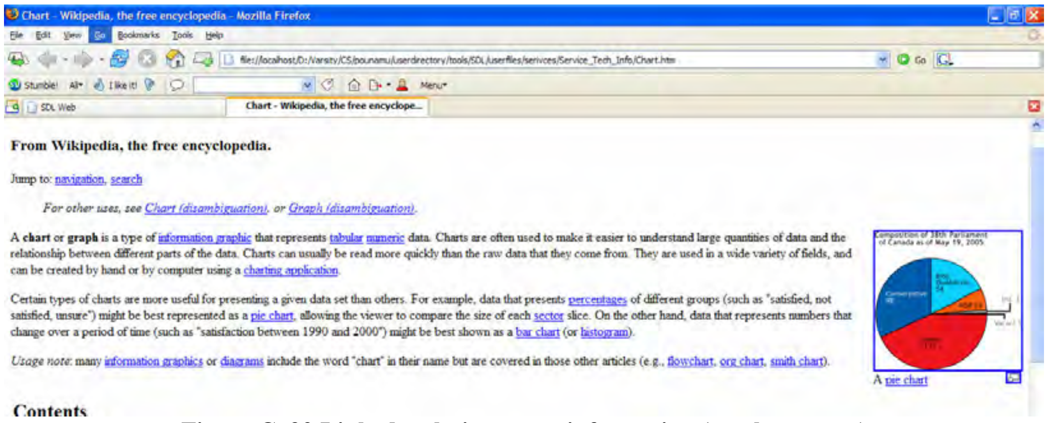
**[BWplot]**

| Parameter | Value | Description |
|---|---|---|
| formula | GPA ~ Year | dataset variables |
| xlabel | Year | dataset variables |
| ylabel | GPA | dataset variables |
| main label | GPA and Year | dataset variables |
| horizonalTF | T | dataset variables |
| aspect ratio0_1 | 0.5 | dataset variables |
| style0_3 | 2 | dataset variables |

**Dataflow**

| Inputs | | Outputs |
|---|---|---|
| da1140553425687 | | da1140553425689 |

Figure C. 31 Information of the selected technique with the link to the technique's type



**From Wikipedia, the free encyclopedia.**

Jump to: navigation, search

*For other uses, see Chart (disambiguation), or Graph (disambiguation).*

A **chart** or **graph** is a type of information graphic that represents tabular numeric data. Charts are often used to make it easier to understand large quantities of data and the relationship between different parts of the data. Charts can usually be read more quickly than the raw data that they come from. They are used in a wide variety of fields, and can be created by hand or by computer using a charting application.

Certain types of charts are more useful for presenting a given data set than others. For example, data that presents percentages of different groups (such as "satisfied, not satisfied, unsure") might be best represented as a pie chart, allowing the viewer to compare the size of each sector slice. On the other hand, data that represents numbers that change over a period of time (such as "satisfaction between 1990 and 2000") might be best shown as a bar chart (or histogram).

*Usage note*: many information graphics or diagrams include the word "chart" in their name but are covered in those other articles (e.g., flowchart, org chart, smith chart).

A pie chart

**Contents**

Figure C. 32 Linked technique type information (mock-up page)

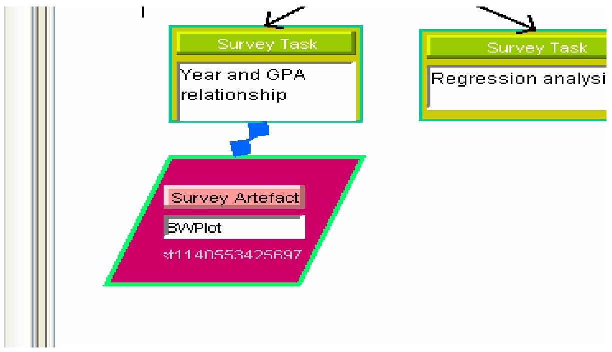


Figure C. 33 Binded survey artefact

# Table of Contents

# *List of Figures and Tables*