Model Driven Design and Implementation of Statistical Surveys

Chul Hwee Kim[1], John Hosking[1] and John Grundy[1, 2]
Department of Computer Science[1] and Department of Electrical and Computer Engineering[2],
University of Auckland, Private Bag 92019, Auckland, New Zealand
{ckim001@ec | john-g@cs | john@cs}.auckland.ac.nz

Abstract

We describe the evolution of a statistical survey design visual language from a standalone design-time modelling language into an environment supporting design, coordination, execution and publication of complex statistical surveys. This involved, firstly, elaboration of the notation to support additional requirements, notably in the area of task modelling. Secondly, tool support has been extended to allow association of model components with survey artefacts, such as data sets, metadata, and statistical package analysis procedures, with the ability to then execute elements of the survey design model to implement the survey analysis. This permits rapid exploration of statistical questions, together with the ability to publish both analysis results and the techniques and processes used, the latter in the form of executable web services and generated documentation. Thirdly, we have undertaken a usability evaluation with a target end user sample that demonstrates strong satisfaction with the tool.

Keywords: statistical surveys, visual language, visual environment

1. Introduction

Statistical surveys have extensive roots running back to ancient times [10]. The development of probability theory and mathematical statistics provided a scientific foundation for statistical surveys [1] and they have become a valuable and ubiquitous tool for obtaining trustworthy information about a target population.

Our research concerns practical issues in supporting survey processes. From a survey practitioner's viewpoint, statistical computing is well supported by high quality software packages like R [6], so low-level statistical technique implementation is not a major operational concern. However many other aspects such as statistical metadata [12] [8] heterogeneity in statistical data semantics [13] and other non-mathematical activities are less well addressed.

In previous work, we attempted to address this through development of the Survey Design Language (SDL) [9], an integrated set of notations aimed at supporting survey design in the same way UML [14] supports software development. In this paper, we describe the evolution of SDL from a set of design notations to a fully integrated environment that supports design, implementation and publication of the statistical survey process. Our hypothesis is that when the semantics of survey designs are visually specified and modelled in a visual language the following benefits are obtained:

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

We begin with a brief background before describing the current state of the SDL notations. We then discuss the association of resources with SDL elements, the execution of fully defined statistical techniques and operations, visualisation of results, and publication of survey documentation and code. This is followed by results of a user evaluation of SDL, and a summary.

2. Background

The inception of the large-scale surveys such as the US census brought enormous managerial complexity prompting development of statistical computing [3]. Many software packages have been developed to support aspects of the survey process including SurveyCraft [18] and Blaise [19] for questionnaire design and response collection, SAS [16] for complex data analysis and SPSS [18] for statistical process design. However, such tools are typically narrow in their focus and the lack of integration and cross tool support, hindered by the absence of mature and widely accepted inter-operable standards, are common sources of difficulty in the survey process.

A small number of tools aim to support the survey process more generally. ViSta [20] provides a visually guided and structured environment for data analysis with multiple visual models to suit various end user groups. GuideMaps help users carry out data analyses even when they lack detailed technical knowledge. WorkMaps visualise a data analysis capturing valuable contextual information such as analytical steps taken. Task based organisation is well expressed but there is little support for merging tasks to support survey contexts, such as objectives, and data collection and analysis or for non-analytic tasks e.g. questionnaire design and metadata support is lacking. ViSta's Lisp-based extensions are a barrier to use given the dominance of S-based languages. ViSta also has poor back end integration support.

CSPro [4] assists users from the data entry stage to produce error and inconsistency free data. It supports data entry, batch edit, and tabulation applications with powerful logic programming support. It only covers some stages of the survey process, with no support for sampling design and data analysis. Thus it is not a solution platform for users to integrate multi-faceted aspects of statistical surveys but it is a very capable tool in the areas it covers.

Statistical data and metadata standard initiatives include Triple-S [8], DDI [5] and MetaNet [12]. These efforts tend to be localised or discipline-specific, but understanding them has given us useful insight into incorporation of statistical data and metadata into SDL.

Our work has been strongly influenced by UML [14], particularly its use of multiple notations, supporting multiple modelling spaces [22]. However, the UML is too software domain-specific for direct use in survey design [12], but the historical development of UML provides valuable insight into visual languages that deal with a complex multidimensional problems. It is no coincidence that the high level aims of SDL closely parallel that of UML, viz:

Visualisation: SDL notations and diagrams provide visual representations of a statistical survey, spanning the entire survey process, but with each diagram visualising a specific aspect of the survey.

Specification: the SDL diagrams as a whole assemble specifications of survey artefacts, their attributes and relationships and resource descriptions mapping artefacts to physical resources and services.

Implementation and execution: similar to UML's MDA approach, SDL diagrams can generate and orchestrate software solutions for a statistical survey.

Documentation: SDL can largely automate the documentation of the survey process.

3. An overview of SDL

Figure 1 shows the relationships between the five main SDL notations, while Figure 2 shows their use to model a national crime survey. The development of SDL from its original form presented in [9] has been strongly influenced by a Cognitive Dimensions [6] evaluation of the original tool which assisted us in defining usability enhancements and notational changes to mitigate issues raised in the evaluation. Survey diagrams, Figure 2 (a) support collaborative brain-storming of the overall survey design. The central element represents the survey, with contexts (eg survey objectives) and their attributes (eg individual objectives) linked from it. Survey data diagrams (b) model datasets, metadata and data operations (e.g. sampling). The metaphor is dataflow, with data sets manipulated by sampling operations to generate sampled data sets. In this example *two stage stratification* is applied to an initial population followed by *patterned clustering*, to obtain a final dataset. Data diagrams may be "drilled into" for more detailed information about their data structures (c). Survey technique diagrams (d) also use a dataflow metaphor, however, whereas survey data diagrams perform operations, such as sampling, which, in a statistician's eye, modify the data, technique diagrams describe non-modifying data analysis probes such as regression analysis or multivariate graphing. They have undergone significant notational change from [9], in particular supporting flow between techniques. In (d), various regression analyses are performed on a dataset. Output icons represent visual outputs, e.g. graphs, resulting from a technique. Survey task diagrams (e), an addition to SDL since [9], hierarchically represent tasks undertaken in the survey. They may be instantiated from a template representing a reusable survey design and can be viewed as a specialisation of Sutcliffe's task model [21]. Tasks may be associated with (bound to) survey artefacts such as data sets and reports (see bottom of the diagram).

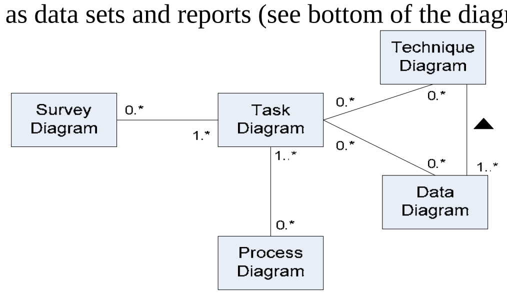


Diagram	***Relationships***	***Detail***
Survey Diagram	Task Diagram	A survey context consists of survey tasks
Task Diagram	Process Diagram Technique Diagram Data Diagram	A survey task may use technique & data diagrams. Survey tasks are survey process participants.
Technique Diagram	Task Diagram	Survey 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 to the survey process.

Figure 1: Relationships between SDL diagrams

Survey process diagrams (f) model the dynamic relationship between tasks, replacing the survey analysis diagrams of [9]. These aggregate tasks into stages (thin arrows), and represents both flow between stages (thick arrows) and use of task outputs as inputs to subsequent stages (diamond end connectors). Here, the initial task involves a study of the population profile as a precursor to

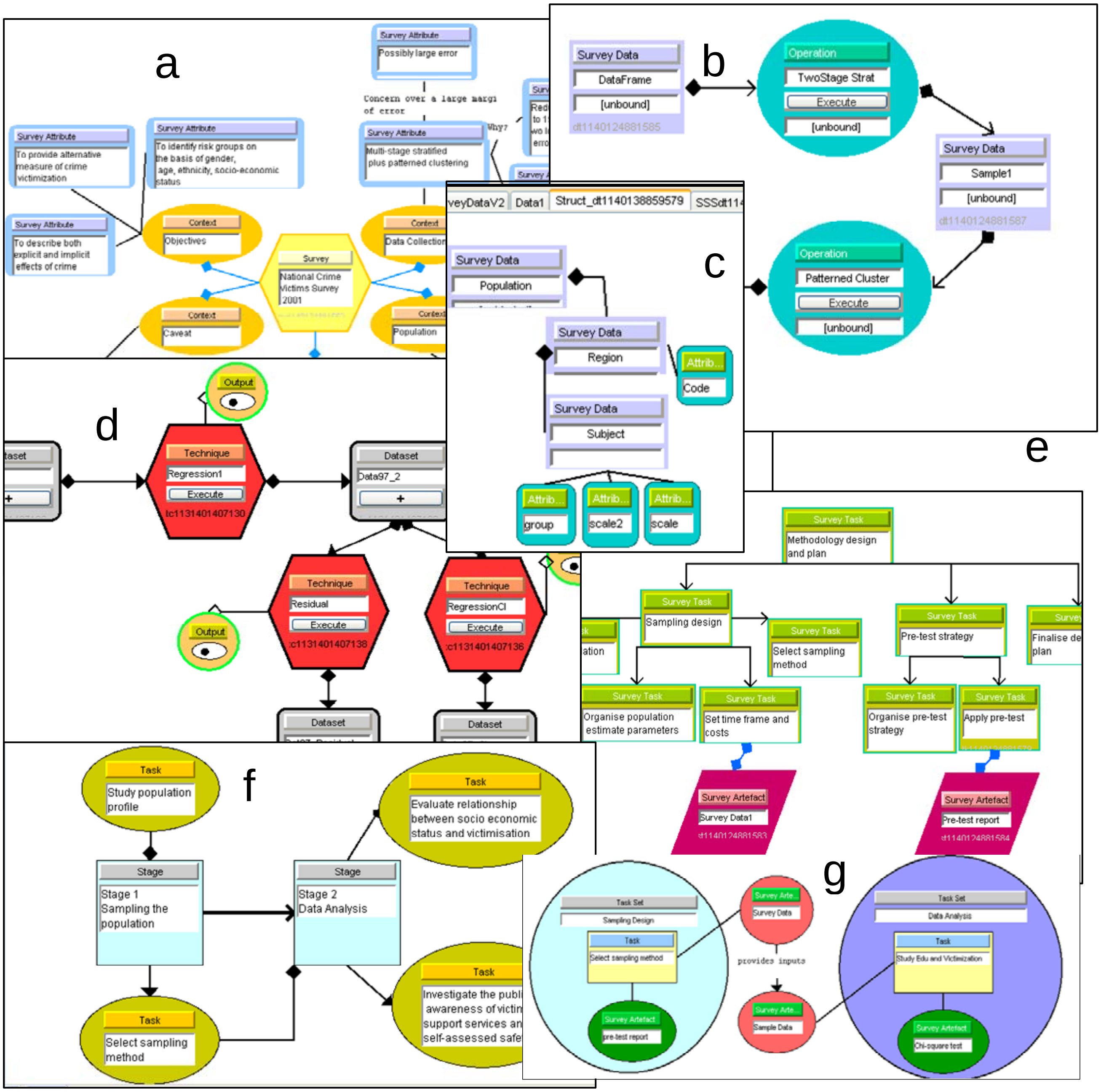


Figure 2: Example SDL diagrams

stage 1, selection of a population sampling method. The result of the selection task is an input to stage 2, where 2 types of data analysis aim to understand both socio-economic effects on victimisation and public awareness. Process diagrams can be "drilled into" to understand detailed relationships between a stage's tasks (g).

4. From modelling to execution

We have made major tool enhancements to support implementation and coordination of surveys designed using SDL. This is via mechanisms to: associate statistical artefacts with SDL model elements; execute bound statistical techniques and operations; visualise results; generate survey documentation; and export standalone executables as web services. This means that out SDL Tool provides comprehensive, integrated support for all phases of the survey process.

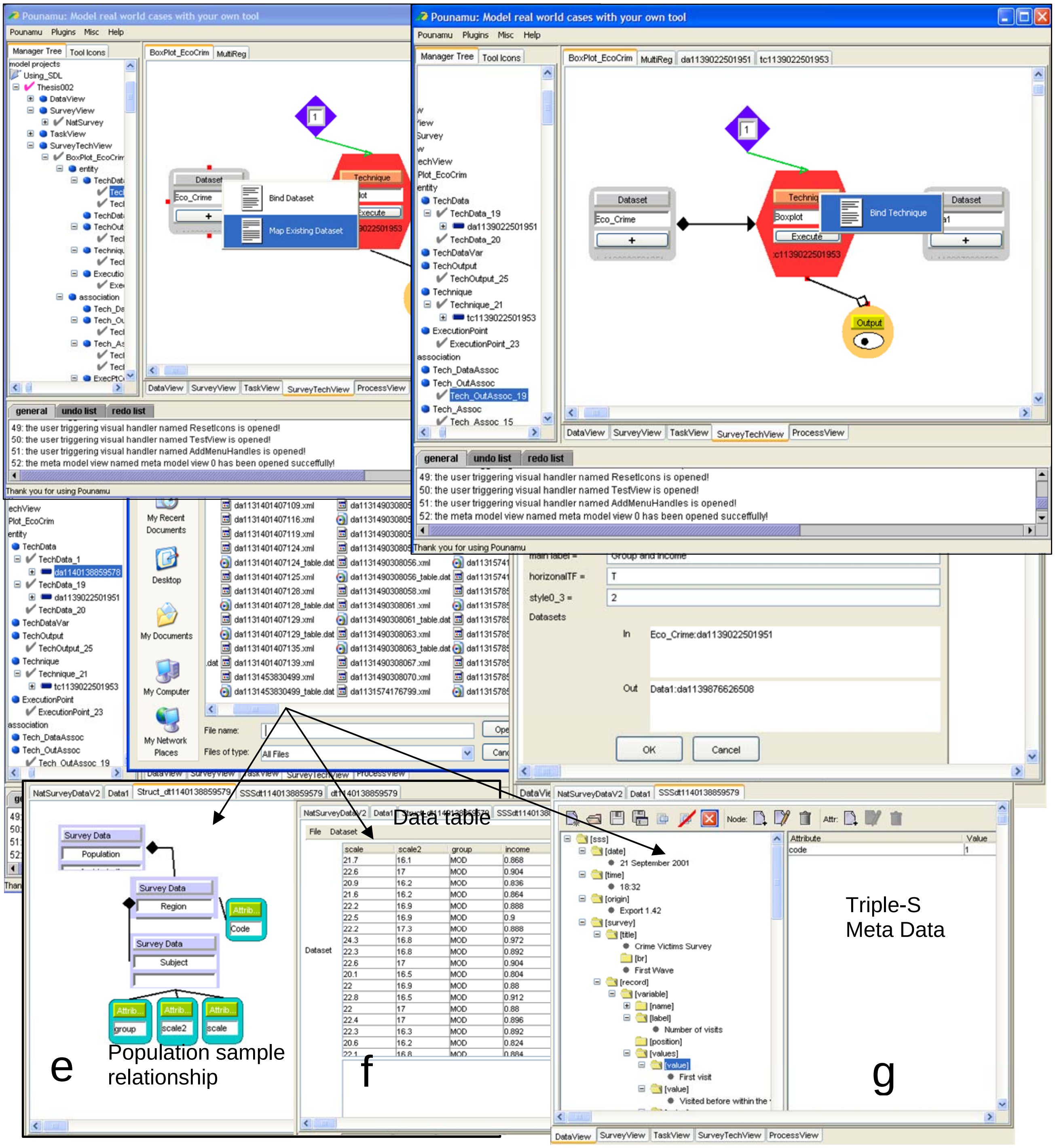


Figure 3: Binding statistical resources to SDL elements

4.1 Binding artefacts

Binding operations map SDL icons to external resources (Figure 3). Menus associated with graphical icons (a,b) initiate resource mappings (dataset and technique resp). Mapping forms are generated for the user to complete (c,d), with many fields automatically filled, inferred from dataflow and other bindings (maintaining inter diagram consistency). Aspects of the icon related to the mapping appear in separate view tabs where the user can view or edit the resource, eg a data structure (e), data set (f) or metadata description (g). A fully mapped SDL icon has a thicker border indicating readiness to execute.

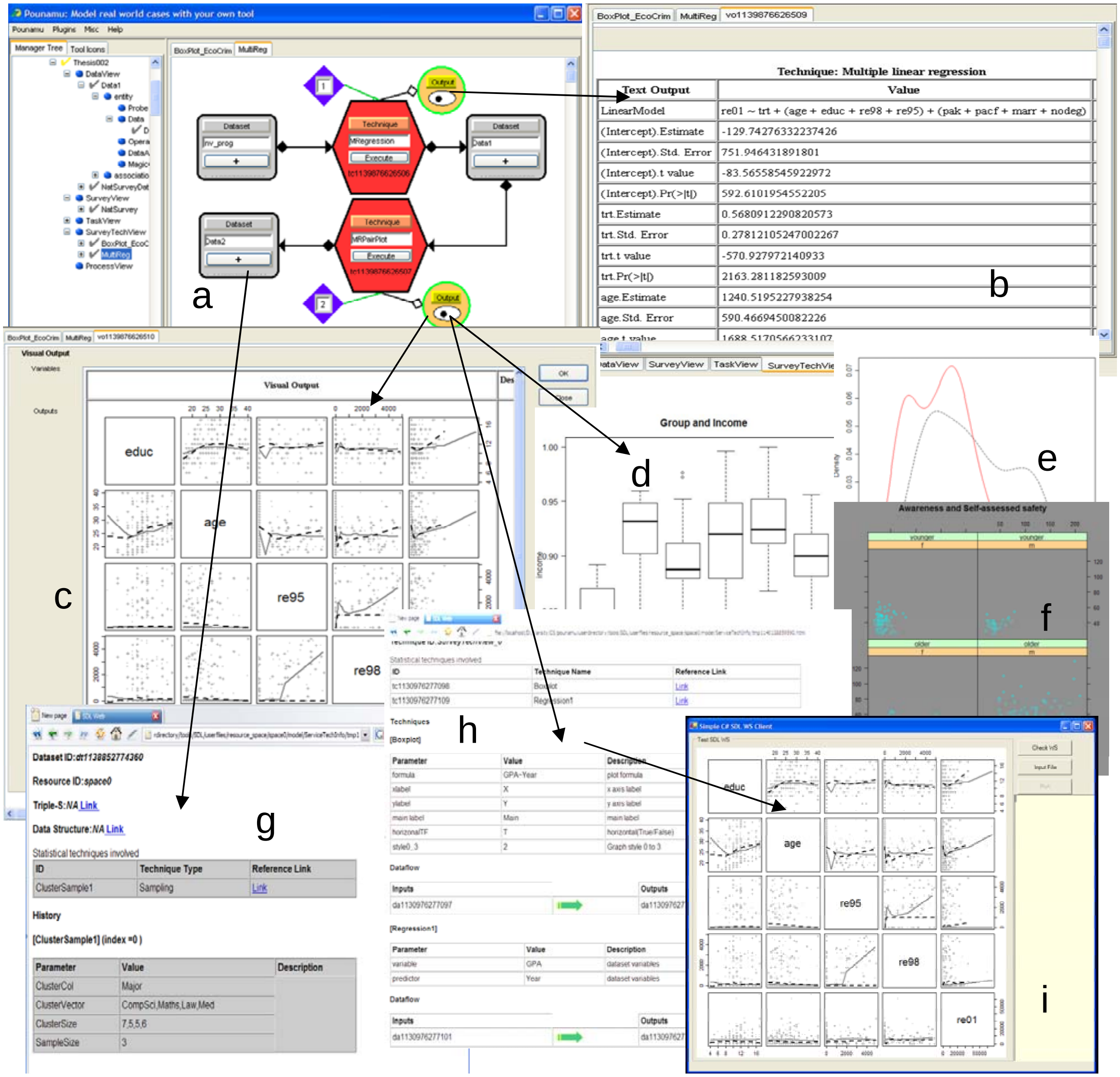


Figure 4: Execution of statistical technique diagram

4.2 Execution

When a survey data or technique diagram is mapped to all required statistical resources the user can execute it in real-time and explore its textual and visual outcomes. Figure 4 shows a typical diagram execution sequence. A completed survey technique diagram has all required graphical icons mapped to appropriate resources (a). Each technique or data operation icon has a button interface used to execute the specification (with icon colour change indicating successful execution). Following execution, the user may select the output port of the executed technique or operation and probe into available outcomes as shown in (b) through (f), all from within the SDL Tool environment, providing an integrated design, coordination, and implementation environment for the statistical survey. The environment is thus live, providing good progressive evaluation support, with the ability to try out techniques and operations as they are defined and bound to appropriate resources.

4.3 Documentation and Publication

Documentation is important in managing the survey process. Our tool can generate documentation in a form accessible to third-party tools or clients allowing them to understand the semantics of our visual diagrams without the need for the prototype tool. Figure 4 (g,h) show generated HTML documents describing a dataset and statistical technique respectively, as specified in (a).

When the user is satisfied with the correctness of a diagram, it can be "published" by turning it into Java code and exposing it in the form of a web service, further eliminating the need to have the tool environment executing in order to understand the survey and promulgate its results. To do this, a web service generation template is used to process the diagram with generated code stored in a web services repository hosted by an Axis server. Clients can access the service specification via a WSDL interface. Exposure as a web service provides platform independent access to survey results. For example, Figure 5 (i) shows a .NET program using the java-implemented statistical technique of Figure 5 (a) via its web service interface.

5. Metamodel and Semantic Layer

An SDL metamodel is the result of a process of abstraction, classification, and generalisation on the collection of SDL visual models in a similar fashion to Model-Driven Architecture [11]. Metamodels are the products of the model reification process, and this process deals not just with physical files that persists visual models but also real-time user interactions that may not be persisted. Binding of a graphical element (e.g. hexagon shape representing a statistical technique), which is only an abstract entity, to a real-world resource (e.g. Web service, R computation procedure, etc) is one such case. Figure 5 show a structural overview of the metamodel layer synthesised from the various SDL diagrams. Visual icons at the visual layer are mapped into instances of metamodel entities derived from these entity types.

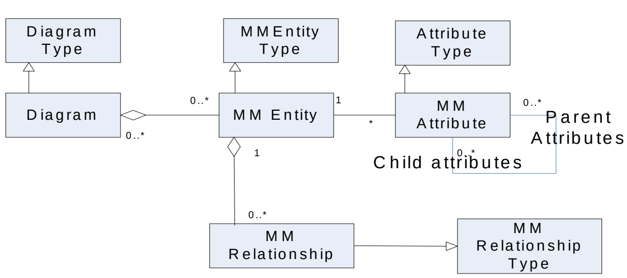


Figure 5: Structural SDL meta model

In order to integrate the SDL model as expressed by multiple SDL diagram types we need to understand the relationship between elements in each diagram. Figure 1 showed the high level relationship between diagrams. At a finer grained level, our approach in integrating the five diagram metamodels uses 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 survey task, dataset, technique or non-connector graphical entity in the visual environment.
- 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.

Survey entities thus provide the basis for both model integration (in the meta modelling process) and inter diagram consistency (at an operational level). The overlapping survey entities act as integration points to merge related views for a target domain, as visualised by Venn diagrams in Figure 6.. Users or tools can transverse the network of related diagrams by using overlapping survey entities as entry/exit points.

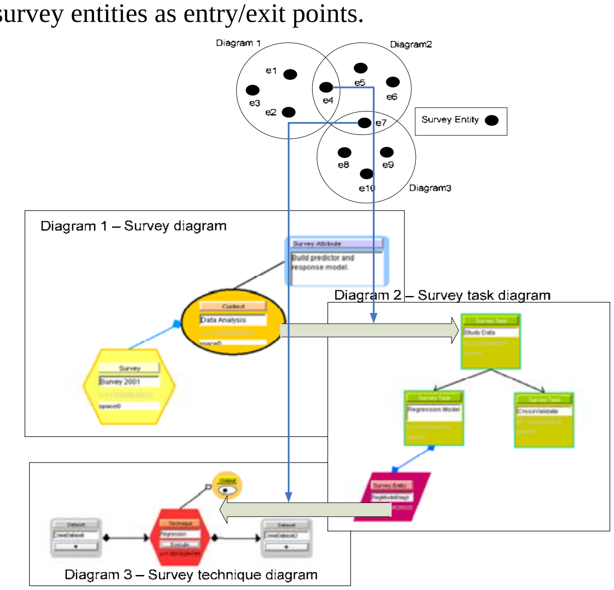


Figure 6: Inter diagram mappings

The necessity for an additional semantic layer comes from SDL tool support. SDL diagrams as communication media from the perspective of human users require no explicit semantic support. However SDL diagrammatic notations anticipated the development of supporting tools and a model-based approach for generating services. Our current proof-of-concept tools do not fully utilise the semantic layer, relying instead on static rule based reasoning to infer model semantics from the metamodel layer. However the semantic layer will be the primary tool for extending the language base of SDL thus we include a brief description here for completeness.

The SDL semantic layer is primarily organised by *topic maps* [15]. Topic maps offer heterogeneous information repositories [2] to tie the underlying semantic of the metamodel to real world statistical survey topics. 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 (see Figure 7). The ontology layer consists of taxonomies of statistical techniques, metamodel structures and relational templates to bind the metamodel to the semantic layer.

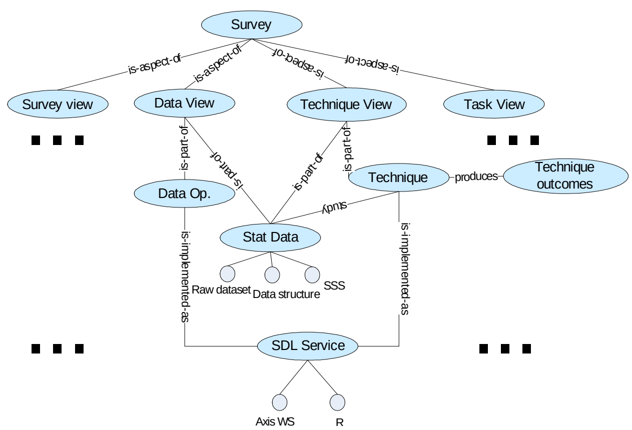


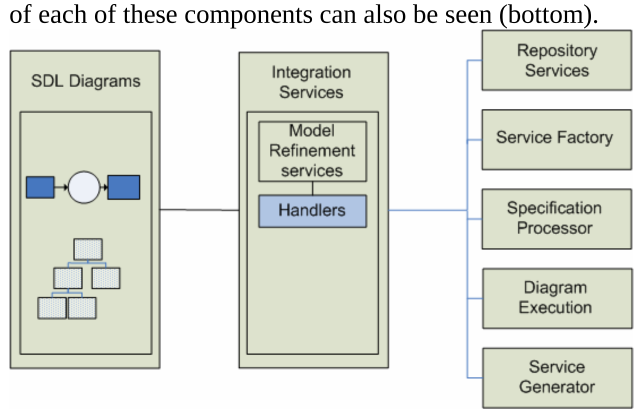
Figure 7: Partial topic map representation of SDL

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. 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 could potentially be used as the basis for making SDL an extremely extensible visual language.

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.

6. Architecture and Implementation

Our SDL tools use an event-driven, loosely-coupled architecture as outlined in Figure 8. SDL diagrams are represented as diagram data (or "views", top left) and a shared repository (or "model", top middle) following the Model-View-Controller paradigm. A set of extensible components (top right) are used to provide repository support, external tool integration, model compilation, execution engine, and a web service generator to make SDL designs accessible to other users. Brief descriptions of each of these components can also be seen (bottom).



Component	***Description***
Handlers	The first component to handle diagram editor events 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)

Figure 8: SDL tool architecture

Figure 9 shows how events are used to couple the components in our architecture. Changes to diagrams are sent as event notifications to an "event handler" for the diagram (1). This passes the event onto a model refinement service (2) which propagates the event to components subscribing to the diagram change type e.g. the Repository Service. The Repository Service translates SDL data into the dataset file format and updates this (3). A response event is generated by the service to indicate success or failure. This event is processed by the model refinement service (4) to determine if any updates to the shared SDL model are necessary. If so, these are applied to the model (5). Such changes may mean other SDL diagrams sharing the changed model data need updating (6). This event-based notification mechanism provides incremental multi-view consistency, persistent repository support, compilation of SDL models, an execution engine for diagrams, and an external tool integration platform.

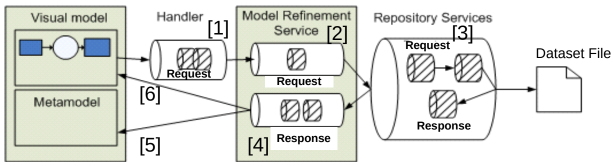


Figure 9: SDL toolset analysis pipe-line

We used the Pounamu meta-tool [23] to implement our SDL multiple-view design tools. Pounamu provides a set of meta-tools for visually specifying diagram and model components, view types and event handlers. SDL was specified as a canonical Pounamu meta-model and a set of view types, one for each different SDL visual notation. Each view type (diagramming specification) has its own set of shapes, connectors and editing constraints. The SDL diagram editors are realised by Pounamu interpreting the SDL tool specifications to provide multi-view and multi-user diagram editors with a shared model. The Pounamu model produced by the SDL diagramming tools is used to provide a data-oriented integration platform to external statistical analysis tools. The Pounamu SDL shared model is "walked" by Pounamu "event handlers" developed specifically for each external tool, transforming it into a format understood by that tool and the tool is invoked via service factory components. Handlers also provide presentation and control integration for external tools allowing results to be displayed in the SDL tool.

7. Evaluation

In addition to guiding the development of our SDL tools through ongoing Cognitive Dimensions evaluations, we have carried out a user survey of the tool involving 3rd Year Statistics students. The key aims for our user testing of SDL were: to evaluate the usability of our diagrammatic notation designs and our software tools in order to improve the SDL visual language design; to fine-tune our SDL software tool solutions; and to validate that research outcomes closely map to our target user requirements. A further important aim was to study how our approach to the survey process support would be perceived by users at a high level and their comparative views of existing software tools and practices for statistical surveys.

Eight test subjects participated in the user testing, chosen out of a potential candidate pool according to recommendations from a tutor in our Department of Statistics. All test subjects were invited to attend an introductory meeting which was designed to convey some of SDL's core concepts. All were new to the concept of a visual environment for statistical surveys but all had good working knowledge of statistical packages, survey theory and design in academic or commercial settings.

A wide range of user activities were conducted in each testing session: pre- and post-demonstration interviews; diagram comprehension exercises; survey technique implementations using provided software tools; and comparative evaluations of SDL and existing statistical survey support tools. Two types of testing outcome were compiled: user-completed questionnaires including user perceptions, opinions and satisfaction regarding the SDL tools and their own performance; and investigator recorded performance evaluations, including task correctness, mistakes, and time to complete given tasks.

Introductory SDL diagrams were based on the 2001 New Zealand 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. Activities utilizing the software tools were to implement survey techniques to produce solutions for given Crime Victims Survey scenarios. The scenarios were designed to permit user initiated actions and task execution.

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 all of them were able to comprehend testing diagrams to the level required for their tasks. Figure 10 summarises the results of our questionnaire-based survey and performance analysis providing general quantitative indications on the usability of SDL and SDL tools.

Category	*Results*
General tool usability	Positive 87.5% Negative 12.5%
Overall notation usability	Positive 75% Negative 25% (half these were only partial negative)
Diagram comprehension (user performance)	Excellent 62.5% Good 25% Average 12.5% Incomplete 0%
Task completion (user performance)	Excellent 37.5% Good 37.5% Average 12.5% Incomplete 12.5%

Figure 10: Results of user survey

Key feedback from users was that SDL accentuates and integrates the multiple aspects of a survey that are often not addressed by existing practice and tools making SDL-based solutions more user-centric. Users used the visual notations not only to formulate numerical computations but also to convey actual operational semantics behind those activities. 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.

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 as being an easy-to-use, expressive and time-effective alternative to conventional documentation. 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. Survey technique and data diagrams were both received very favourably.

It was not a trivial task for our users 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. This problem is analogous to a novice UML user's difficulty in merging 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 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 the contributing diagrams. The test subjects responded favourably to the degree of freedom that SDL tools offer in designing the survey process.

There are no existing tools which have survey and task diagrams hence the SDL survey technique and data diagrams were obvious candidates for the test subjects' comparative reviews. Test subjects noticed that, in contrast to existing tools, they were not restricted to a sequential batch mode or an interactive mode which tends to require high attention investment to articulate a technique which requires frequently modification. During manipulation of implemented statistical techniques, existing tools put emphasis on result oriented step-by-step batch operations. The data flow metaphor utilized in SDL survey data and technique diagrams provided users with design-time freedom to change input and output data flows and the dynamic mapping of a graphical entity to a physical dataset or a statistical technique. This meant that data flows could be routed to multiple techniques in a variety of ways to easily form variations of initial designs.

Many aspects of the above qualitative and quantitative measures were further viewed in the light of their relationships to the Cognitive Dimensions framework. Key findings from this comparison and analysis were: SDL provides good Closeness of Mapping to the statistical survey design process; multiple SDL diagrams increase Viscosity and Hidden Dependencies but this is mitigated by high-level visual entities enabling rapid change; SDL's Progressive Evaluation is a significant advantage over existing tools; SDL would benefit from support for secondary notation on diagrams; current support for diagram juxtaposability and consistency is too limited; and SDL diagrams' multiple levels of Abstraction assist the statistical survey design process.

8. Summary

We have developed SDL, a set of visual languages for complex statistical survey design, together with a proof-of-concept implementation of SDL, integrated with several off-the-shelf, commonly used statistical analysis packages. Despite its proof-of-concept nature, we have successfully carried out a user study with the toolset with final year Statistics undergraduates on a real-world set of statistical survey problems. This experiment demonstrated that SDL's visual languages and integrative tools significantly enhance the design and execution of such surveys. Our current work includes improving layered diagram structure and the introduction of a diagram to explicitly relate other SDL diagrams.

We plan to add executable statistical process diagrams, user scripting via the R language for more powerful analysis capabilities for expert users, and repositories for both statistical data and reusable process models. Further enhancements include secondary notation support for annotative diagrams, visual presentation of underlying statistical survey process enactment and data analysis state, improved visual presentation of mapping forms, and automatic layout of some diagrams.

Acknowledgements

The authors gratefully acknowledge the assistance of James Reilly as statistical consultant, our panel of user evaluators, and Nianping Zhu for implementing the Pounamu meta tool.

References

[1] Biemer, P.P. and Lyberg, L.E. Introduction to survey quality Wiley Inter-Science 2003, Ch 2

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

[3] Chambers, J.M. and Ryan, B.F., The ASA Statistical Computing Section: A History, The American Statistician, May 1990, 4(2) pp87-89

[4] CSPro, http://www.census.gov/ipc/www/cspro/

[5] DDI, The Data Documentation Initiative, http://www.icpsr.umich.edu/DDI/
[6] Green, T.R.G and Petre, M, Usability analysis of visual programming environments: a 'cognitive dimensions' framework, JVLC 1996 (7), pp.131-174.
[7] Ihaka, R. and Gentleman, R. R: A Language for Data Analysis and Graphics, J Computational and Graphical Stats, Vol. 5, No. 3 1996, pp. 299-314
[8] Jenkins, S. G. The Triple-S survey interchange standard http://www.triple-s.org/sssasc96.htm
[9] Kim, C., Hosking, J., and Grundy, J., A Suite of Visual Languages for Statistical Survey Specification. IEEE VL/HCC 2005, 19-26
[10] Madansky A., Journal of Official Statistics, Vol.2, No.4, 1986. pp. 561-569
[11] Mellor et al 2004 Ch2
[12] MetaNet 2003, Metanet Conference INEAG, Samos, May 2003
[13] Olenski, J. 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
[14] OMG, OMG Unified Modelling Language http://www.uml.org/
[15] Pepper 2002
[16] SAS Institute Inc. http://www.sas.com
[17] SPSS, SPSS Inc. SPSS statistical software http://www.spss.com
[18] SPSS Inc., SurveyCraft, http://www.spss.com/surveycraft/
[19] Statistics Netherlands, *Blaise*, http://www.cbs.nl
[20] Young, F.W. & Bann, C.M. ViSta: A Visual Statistics System Statistical Computing Environments for Social Research., Sage Publications, Inc., 1997, 207-235
[21] Sutcliffe, A. Domain Theory: Patterns for Knowledge and Software Reuse, Lawrence Erlbaum Associates, Inc. Mahwah, NJ, USA, 2002
[22] Unhelkar, B., and Henderson-Sellers, B., Modelling Spaces and the UML, Proc IRMA, New Orleans, 2004
[23] Zhu, N., Grundy, J.C. and Hosking, J.G., Pounamu: a meta-tool for multi-view visual language environment construction, Proc *VL/HCC'04* Rome, Italy, 25-29 Sept 2004, pp. 254-256.