

PROVENANCE OF VISUAL INTERPRETATIONS IN THE EXPLORATION OF DATA

A THESIS SUBMITTED TO THE UNIVERSITY OF MANCHESTER
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY
IN THE FACULTY OF ENGINEERING AND PHYSICAL SCIENCES

2015

By
Aqeel A. Al-Naser
School of Computer Science

Contents

Abstract	11
Declaration	12
Copyright	13
Dedication	14
Acknowledgements	15
1 Introduction	16
1.1 Interpretation Through Visualization	16
1.1.1 What is Visualization?	16
1.1.2 Visual Interpretation and Features Extraction	17
1.2 Provenance of Visual Interpretations	18
1.3 Research Problem	19
1.3.1 The Oil and Gas Exploration Workflow	20
1.4 Aims and Objectives	22
1.5 Contributions	23
1.5.1 Publications	24
1.5.2 Other Contributions	25
1.6 Thesis Approach	26
2 Background and Related Work	28
2.1 Chapter Introduction	28
2.2 The Visualization Pipeline	29
2.2.1 Use of Visualization for Interpretation	31
2.2.2 Pipeline Evolution	31
2.2.2.1 Metadata	32

2.2.2.2	Parallel Pipeline	32
2.2.2.3	Locality	34
2.3	Features in Visualization	34
2.3.1	Features Detection Techniques	36
2.4	Provenance	36
2.4.1	What is Provenance?	37
2.4.2	Characterizations of Types of Provenance	38
2.4.2.1	Usage of Provenance	38
2.4.2.2	Subjects of Provenance	39
2.4.3	W3C's PROV—A Provenance Model	39
2.4.4	Provenance in Visualization	40
2.5	Chapter Summary and Conclusion	42
3	A Provenance-Enabled Interpretation Pipeline	44
3.1	Chapter Introduction	44
3.2	Interpretation Pipeline	45
3.3	A Provenance Model for Interpretations	47
3.3.1	Definitions	47
3.3.1.1	Classes	48
3.3.1.2	Relationships	49
3.3.1.3	Namespaces of Core and Extended Structures	50
3.3.2	Illustration of The Model by Examples	51
3.3.2.1	Example: Basic Case	52
3.3.2.2	Example: Multiple Interpretations	52
3.3.2.3	Example: Assessment of Interpretation	53
3.4	Implementation Consideration	54
3.4.1	Fine-Grained Data Granularity	54
3.4.2	Open-Standard, Single-View and Multi-User Access	57
3.5	Chapter Summary	58
4	Use of Visual Interpretation in Geoscience	59
4.1	Chapter Introduction	59
4.2	Definition and Acquisition of Seismic Data	60
4.2.1	SEG-Y Format	62
4.3	Geological Features and Seismic Interpretations	64
4.3.1	Geological Features in Seismic Data	64

4.3.2	Seismic Interpretation	66
4.3.3	Uncertainty in Seismic Interpretation	68
4.4	Volume Visualization for Seismic Data	69
4.4.1	Seismic Imaging Applications	70
4.4.1.1	Petrel	70
4.5	Data Management for Seismic Interpretations	71
4.6	Demanding Oil and Gas Workflow—The Present and Future	72
4.7	Motivation Survey	73
4.7.1	Survey Purpose and Population Frame	73
4.7.2	Data Collection Method	74
4.7.3	Survey Sample	74
4.7.4	Response from Geoscientists	75
4.7.5	Response from Information Technology Personnel	75
4.7.6	Survey Analysis	76
4.8	Chapter Summary	78
5	Implementation for Geoscience	80
5.1	Chapter Introduction	80
5.2	Prototype Architecture Overview	81
5.3	Provenance-Enabled Data Structure for Seismic Interpretation	85
5.3.1	Implementation Hardware and Technology	85
5.3.2	Data Model	87
5.3.2.1	The <i>geo</i> Namespace	87
5.3.2.2	Database Relational Model—An Overview	87
5.3.2.3	Raw Seismic Imaging Data	90
5.3.2.4	Geological Features	94
5.3.2.5	Data Index	97
5.3.3	Multi-Interpretations and History Tracking—Model Mapping Example	98
5.3.4	Data Feed	101
5.4	Feature-Embedded Spatial Volume (FESVo)	104
5.4.1	Data Loading	105
5.4.1.1	Gridding, Coordinate Mapping and Level-of-Detail	107
5.4.1.2	Data Lookup	110
5.4.1.3	Data Filtering	111
5.4.2	Data Caching	112

5.4.3	Capturing Interpretations	112
5.4.4	Optimization	113
5.5	User Interface and Rendering Engine	115
5.5.1	Rendering	115
5.5.2	Tooltip—Provenance Information Per Point	116
5.5.3	Filtering and Highlighting	117
5.6	Chapter Summary	118
6	A Walk-through of the Implemented Application	120
6.1	Basic Functionalities	120
6.2	History Tracking	124
6.3	Tooltip	125
6.4	Highlighting and Filtering	126
6.5	Amendment	129
6.6	Assessment	131
7	Evaluations and Results	134
7.1	Chapter Introduction	134
7.2	Functionality Test	135
7.2.1	Task 1: Multi-User Horizon Time Shifting	137
7.2.2	Task 2: Deletion of an Interpreted Object	138
7.3	Interviewing Domain Experts	139
7.3.1	Data Collection Method	139
7.3.2	Data Outcome—Using Coding	141
7.3.3	Data Analysis	144
7.3.3.1	Case Examples	144
7.3.3.2	Concerns	145
7.3.3.3	Suggested Enhancements	146
7.3.3.4	Future Applications	147
7.3.3.5	Challenges	148
7.4	Performance Measures	148
7.4.1	Using Commodity Hardware	148
7.4.1.1	Effect of Threads	149
7.4.1.2	Effect of Multi-Statement Length	150
7.4.2	Scalability Tests	150
7.5	Chapter Summary	153

8	Conclusions and Future Work	155
8.1	Summary and Conclusion	155
8.2	Limitations and Future Work	158
8.2.1	Future Research	159
8.2.1.1	Features Detection	159
8.2.1.2	Scalability	159
8.2.1.3	Social Aspects	160
8.2.2	Further Development	161
8.2.2.1	Provenance Language	161
8.2.2.2	Advanced Functionalities Interface	161
8.2.2.3	Enhanced Versioning	162
8.2.2.4	Data Handling	162
8.2.2.5	Enhanced Rendering	163
	Bibliography	164
A	Usability Evaluation Questionnaire	178
A.1	Sample A	178
A.2	Sample B	181
A.3	Sample C	184
B	Survey Summary	187
C	Interviews Transcript	193
C.1	Interview 1	193
C.2	Interview 2	202
C.3	Interview 3	217
D	Background on Rendering Techniques	220
D.1	Surface Rendering vs. Direct Volume Rendering	220
D.2	Texture Mapping	221
D.3	Programming Libraries	222
D.3.1	OpenGL	222
D.3.2	Shading Languages	223

List of Tables

3.1	Classes of The Model	49
3.2	Relationships of The Model	50
3.3	Defined Types	51
5.1	Defined Types of the <i>geo</i> Namespace	89
5.2	Fields of Table TRACE	92
5.3	Fields of Table SURVEY	93
5.4	Fields of Table FEATURE_POINT	96
5.5	Fields of Table INTERPRETATION	97
5.6	Table INTERPRETATION – Example Data	99
7.1	Case Study Dataset Size	136
7.2	Fetching Time Comparison	149
7.3	Queries Bandwidth	150

List of Figures

1.1	An example of 2D seismic interpretation	17
1.2	An Abstract Visualization Pipeline	17
1.3	Two Different Interpretations	21
1.4	Examples of Seismic Interpretations and Petrophysical Modeling . . .	22
1.5	Research Problem	23
2.1	Background Disciplines	29
2.2	AVS Network	30
2.3	PROV Data Model	40
3.1	Proposed Interpretation Pipeline	46
3.2	PROV Graph Layout Conventions	48
3.3	Provenance of Interpretation – Basic Case	52
3.4	Provenance of Interpretation – Multiple Interpretations Case	54
3.5	Provenance of Interpretation – Assessment Case	55
3.6	Coarse-grained vs. Fine-grained Provenance	56
3.7	Conventional Architecture	57
4.1	Upstream Workflow	60
4.2	Illustration of Seismic Acquisition	61
4.3	Seismic Trace and Section	62
4.4	Seismic Cube Slicing	62
4.5	SEG-Y File Structure	64
4.6	Structural Traps	65
4.7	Subsurface Structure in Time vs. Depth Domains	66
4.8	Seismic Sparse Extrema Representation	67
4.9	Contribution Towards Subsurface Understanding	68
4.10	Upstream Workflow with EOR	73
4.11	Survey Response from Geoscientists	76

4.12	Survey Response from IT Personnel	77
5.1	The Architecture—Implemented for Geoscience	82
5.2	Granularity in Different Fields	83
5.3	Provenance of Seismic Interpretation – Basic Case	88
5.4	Database Schema for Geoscience	90
5.5	Exported Horizon File	94
5.6	Provenance of Seismic Interpretation – Multi-Interpretation Case . . .	100
5.7	Data Feed	102
5.8	The SEG-Y Loader	103
5.9	FESVo Internal Architecture	105
5.10	Data Mapping	109
5.11	Seismic Survey Dimension	109
5.12	Coordinate Mapping for Interpretation	114
5.13	Seismic Slice	116
5.14	Horizons Rendering – Comparison to Petrel	117
5.15	Tooltip	117
7.1	Screenshots of Task 1	138
7.2	Screenshots of Task 2	139
7.3	Effect of Threads	151
7.4	Effect of Multi-Statement Length	152
7.5	Effect of Threads Using 4-Node Database	153
8.1	Overlapping	162
D.1	Polygonal Mesh Representation	221
D.2	3D Texture Volume Rendering Technique	222
D.3	2D and 3D Texture Slicing	223
E.1	Voreen’s SEG-Y Reader Outputs	226

Abstract

The thesis addresses the problem of capturing and tracking multi-user interpretations of 3D spatial datasets. These interpretations are completed after the end of the visualization pipeline to identify and extract features of interest, and are subjective to human intuition and knowledge. Users may also assess regions of these interpretations.

Consequently, the thesis proposes a provenance-enabled interpretation pipeline. It adopts and extends the W3C PROV data model, producing a provenance model for visual interpretations. This was implemented for seismic imaging interpretation in a proof-of-concept prototype architecture and application. Accumulation of users' interpretations and annotations are captured by the provenance model in a fine-grained form. The captured provenance information can be utilised to filter data.

The work of this thesis was evaluated in three parts. First, a usability evaluation by geoscientists was conducted by postgraduate students in the field of geoscience to illustrate the system's ability in allowing users to amend others' interpretations and trace the history of amendments. Second, a conceptual evaluation of this research was approached by interviewing domain experts. The importance of this research to the industry was assured. Interviewees perceived and shared potential implementations of this work in the workflow of seismic interpretation. Limitations and concerns of the work were highlighted. Third, a performance evaluation was conducted to illustrate the behaviour of the architecture on commodity machines as well as on a multi-node parallel database, such that a new functionality in fine-grained provenance can be implemented simply but with an acceptable performance in realistic visualization tasks. The measures suggested that the current implementation achieved an acceptable performance in comparison to conventional methods.

The proposed provenance model in an interpretation pipeline is believed to be a promising shift in methods of data management and storage which can record and preserve interpretations by users as a result of visualization. The approach and software development in this thesis represented a step in this direction.

Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright

- i. The author of this thesis (including any appendices and/or schedules to this thesis) owns certain copyright or related rights in it (the “Copyright”) and s/he has given The University of Manchester certain rights to use such Copyright, including for administrative purposes.
- ii. Copies of this thesis, either in full or in extracts and whether in hard or electronic copy, may be made **only** in accordance with the Copyright, Designs and Patents Act 1988 (as amended) and regulations issued under it or, where appropriate, in accordance with licensing agreements which the University has from time to time. This page must form part of any such copies made.
- iii. The ownership of certain Copyright, patents, designs, trade marks and other intellectual property (the “Intellectual Property”) and any reproductions of copyright works in the thesis, for example graphs and tables (“Reproductions”), which may be described in this thesis, may not be owned by the author and may be owned by third parties. Such Intellectual Property and Reproductions cannot and must not be made available for use without the prior written permission of the owner(s) of the relevant Intellectual Property and/or Reproductions.
- iv. Further information on the conditions under which disclosure, publication and commercialisation of this thesis, the Copyright and any Intellectual Property and/or Reproductions described in it may take place is available in the University IP Policy (see <http://documents.manchester.ac.uk/DocuInfo.aspx?DocID=487>), in any relevant Thesis restriction declarations deposited in the University Library, The University Library’s regulations (see <http://www.manchester.ac.uk/library/aboutus/regulations>) and in The University’s policy on presentation of Theses

Dedication

This thesis is dedicated to the memory of my grandfather, Abdulkareem Al-Naser (1926–2013), and to my beloved son, Aziz.

Acknowledgements

I would like to thank my supervisor, Dr. John Brooke, and my co-supervisor, Dr. Duncan Irving, for their tremendous help and support through out this PhD research. Special thanks go to Masroor Rasheed for his great help, particularly for his support in the data structure. Also, I would like to thank Teradata for allowing me to run performance tests of my application on one of their parallel clusters. In addition, I would like to thank Saudi Aramco, the national oil company of Saudi Arabia, for sponsoring this PhD project. Last but not least, many thanks go to my wife, Ayat, and my parents for their moral support.

Chapter 1

Introduction

1.1 Interpretation Through Visualization

Visualization allows users to visually explore datasets and thus build an understanding of the underlying features, which are structures within the visualized dataset. In the context of this thesis, the process of understanding the dataset and extracting features through visualization is referred to as *interpretation*. Figure 1.1 shows an example of a simple interpretation of a 2D seismic data from the oil and gas industry, where some geological features were manually marked. Seismic data is obtained to represent the structure of subsurface layers of the earth’s interior. This is the domain in which this thesis is evaluated.

1.1.1 What is Visualization?

Visualization is the transformation of raw data into graphical primitives. Using computer graphics, these primitives can then be rendered to produce a set of images that represents the original raw data. These steps form the basic functionalities of what is known as the *visualization pipeline*; Figure 1.2 illustrates an abstract pipeline as described by Haber and McNabb [1]. Throughout this pipeline, raw data undergoes filtering and transformation functions to be visually rendered [2]. Development in three-dimensional (3D) visualization has opened great opportunities for improvements in many industries such as medicine, entertainment and the oil and gas industry. Human beings are naturally capable of analysing visual rather than numerical representations. Therefore, visualization for data analysis is becoming more and more important as sizes of datasets are dramatically increasing. This increase is due to the increase

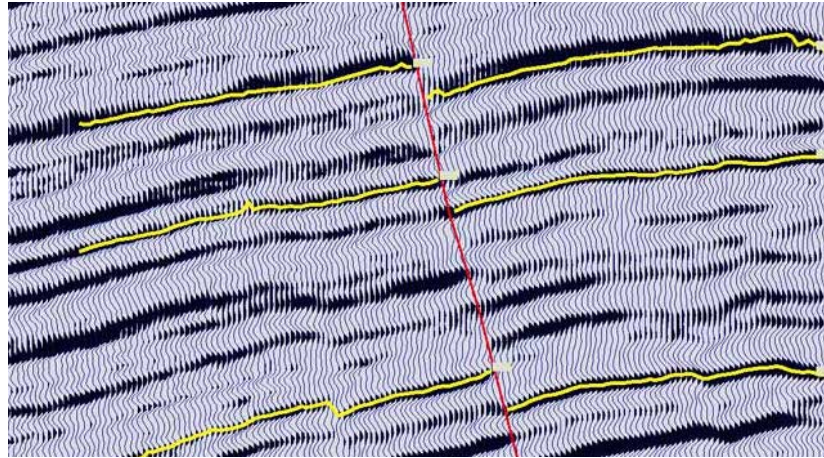


Figure 1.1: The figure shows a 2D slice of seismic data with simple manual interpretation represented by the yellow and red marks. The yellow and red marking represents geological horizon and faults, respectively. (image source: Schlumberger <http://www.slb.com/>)

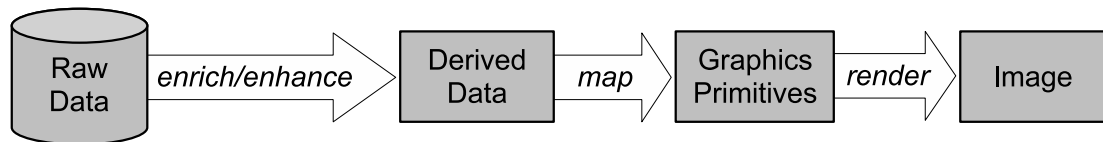


Figure 1.2: An abstract visualization pipeline as described by Haber and McNabb (1990)

in computing power and advancement of data acquisition instruments, such as medical scanners and seismic data receivers, that allow scientists to obtain more complex and accurate data than in the past. This had also led to the necessity of multi-user, and perhaps multi-team, collaborative environments in order to better understand the visualized data and thus to take well established decisions.

1.1.2 Visual Interpretation and Features Extraction

One of the most powerful benefits that visualization brings to data analysis is the ability to harness the intuition of the user in the process of understanding the data. Human visual abilities are particularly tuned to respond to features embedded in such a space. However, human intuition has to be supported by algorithms that help to identify and highlight features of the data, such as deformable models techniques for medical image analysis [3]. Referring back to the visualization pipeline, these algorithms are applied within the pipeline's functions. According to Roberts [4], users are encouraged to

manipulate the pipeline's functions to generate multiple views of the desired dataset as a single representation may lead the user to misinterpret the data.

However, it can often be the case that algorithms cannot completely identify the features of interest. Human intuition must complete the process, and given the nature of intuition this can be a source of differing interpretations depending on the human expertise. This may occur in data that is noisy or visually complex, such as seismic imaging data. The task of interpretation and features extraction that this thesis refers to takes place at the end of the visualization pipeline, independent of any applied algorithm on the pipeline. Thus, we do not have a single feature but multiple interpretations of a feature.

1.2 Provenance of Visual Interpretations

Considering the case where users must manually complete the identification of features of interest according to their intuition, at some stage, collaborative visualization may be required for experts to discuss, reconcile and perhaps evaluate different interpretations. This thesis considers asynchronous collaboration in which the collaborators can examine each other's interpretations but this does not have to be done synchronously and the collaborators can work independently of each other. It may occur over an extended period, sometimes lasting several years, where each time the visualization is performed it may only be a single user who is viewing. Users (scientists) also need to track the origin, or source, of such interpretations; sometimes earlier interpretations may need to be revisited. In this thesis, the notion of *provenance* is adopted to describe the origin and development of these interpretations.

The literature shows successful attempts in integrating provenance into the visualization pipeline through the introduction of VisTrails [5, 6, 7]. These provenance-tracking systems adopt what can be classified as a *process-oriented* provenance, which is concerned with the process of how visualization was achieved. They capture changes on the pipeline modules and parameters; part of Figure 1.5 shows this kind of provenance. However, it does not capture the results of interpretations that occur as the user is viewing the rendered output of the visualization and acting with pointing actions to trace and manipulate the features of interest.

1.3 Research Problem

This research aims to resolve the issue of capturing interpretations which can only be performed after the visualization pipeline has executed and are derived from user actions recorded in pointing devices. This requires that these actions should be propagated back to the original source, in effect reversing the direction of the visualization pipeline. The data of interest to this research, which to be interpreted, has the following characteristics; further explanation is presented afterward.

- Noisy – data that is gathered in the field may have noise because of limitations of the apparatus that collects the data.
- Large – usually residing in massive data repositories.
- Co-located with other data of different types.
- Evolving – data can change over time.
- Requires multiple specialties and disciplines to be interpreted and processed within a workflow.

The noisy nature of the data means that level of uncertainty in interpretation increases and thus data is prone to multiple interpretations. This leads to the development of different opinions, about the data. In addition, the data itself (i.e. the raw data which is to be interpreted) can evolve over time, due to natural factors for example. Thus, opinions may change as data evolves. Data interpretation requires the expertise of experts with specialisms in different sub-disciplines as part of a workflow. Therefore, initial judgments about the data often change when multiple users from different background collaborate.

The above considerations lead to a challenge in managing the resulting collections of interpretations of features. Identifying who produced or contributed to producing these interpretations or who evaluated them is difficult. Capturing those various interpretations and linking them to their corresponding datasets, which they describe, is the challenge that this thesis addresses.

The manner in which provenance is captured and recorded depends on the nature of the data which is being analysed. In this thesis the emphasis is on data that represents the structures contained within a volume that is described by the three dimensions of physical space. Within this volume there are believed to exist features that can be envisaged as geometrical structure, which can be irregular to some extent. These features

are considered to have physical significance; they are more than rendered artefacts. As previously described in this chapter, it is adequate in some cases to capture the algorithms that extracted the feature within the visualized dataset. However, in other fields, such as the one this thesis is applied to, the features of the visualized dataset heavily depend on a user's view. In addition, those features that are generated by the various opinions are required as an input into subsequent stages such as modeling and simulation. Therefore, interpretations that represents a user's understanding and opinion must be explicitly captured and saved, independent of any algorithm.

Conventionally, these interpretations are captured within a software application. However, most, if not all, of the metadata associated to these interpretations are lost when exported. Generally, this creates a challenge in tracing the origin of the feature and any attached metadata. Other associated challenges include: integration of analysis, scalability, consistency and longevity. In this research, the aim is to formally capture the evolving opinions, which are represented as interpretations, along with annotation of interpretations of multiple users.

1.3.1 The Oil and Gas Exploration Workflow

To test the idea in a representative application, this thesis uses the oil and gas exploration workflow. This is a field that has received a great deal of attention in terms of research [8, 9, 10, 11, 12, 13, 14, 15] as well as software development such as Avizo Earth¹, GeoProbe² and Petrel³. In this thesis, this field is also referred to as *geoscience*, a more general term. The oil and gas industry is of great importance due to its dominant position in the supply of the world's energy demands. In 2011, the consumption of hydrocarbon fuels (oil and natural gas) accounted for 57% of the world's primary energy as calculated by BP [16]. By 2030, the demand for fossil fuels (oil, natural gas and coal) will continue to grow with a noticeable increase in gas's market share [17]. Thus, this field requires to manage its massive datasets more efficiently.

This field illustrates the problem of interpretation described above. The aim of seismic interpretation [18] (also referred to as *subsurface interpretation* in this thesis) is to extract geological features. Expert interpretation in this field is very important and very central to the definition of the features. The data is noisy and the features to be extracted have a very complex spatial structure owing to processes of buckling,

¹<http://www.vsg3d.com/avizo/earth>

²<https://www.landmarksoftware.com/Pages/GeoProbe.aspx>

³<http://www.software.slb.com/products/platform/Pages/petrel.aspx>

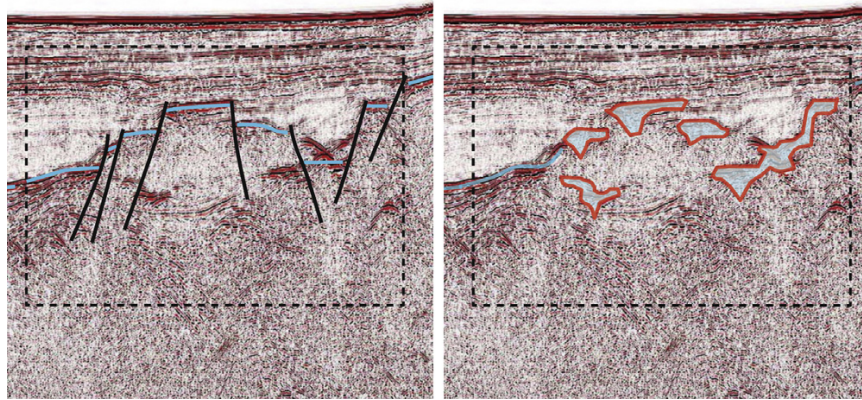


Figure 1.3: Two different interpretations on a same visualization view of a seismic section. The sketches represent interpreted geological features. Drawings from Lidal et al. [20].

folding and fracturing [19]. This makes a purely automated approach to feature extraction very difficult to achieve. It must always be completed by a human. Thus, it is subjective in the sense that two interpretations may result from a single visualization view. This is illustrated in Figure 1.3 where two distinct interpretations of the same visualization view were generated [20]. In addition, interpreted surfaces, or feature, are required to be explicitly captured as they form an input to the next stage of the oil and gas exploration, which is modeling. Seismic data can evolve over time. An example of this was shown by Demyttenaere et al. [21] where three different seismic surveys were acquired every 10 years on a single field. Because of technology advancement as well as hydrocarbon production from the field, the resulting seismic reflection was different between the second and third stages. According to the domain expert collaborators consulted in this thesis, one of the industry's challenges is managing subsurface interpretations, linking them to their corresponding raw datasets and perhaps walking through how an interpretation was achieved by multiple users.

Currently, geoscientists still rely on file naming conventions to record metadata for searching and identifying datasets [22], which include interpretations of users that were exported from software applications. As highlighted by the domain expert collaborators consulted in this thesis, the filename is the only form of metadata in this system and it only contains the name of the last person to make the interpretation. The possibility to view the whole provenance of the interpretation by all collaborators is lost.

As illustrated in Figure 1.5, this thesis is addressing the lack of provenance information of seismic interpretation. Looking at the data pathway, a provenance on raw

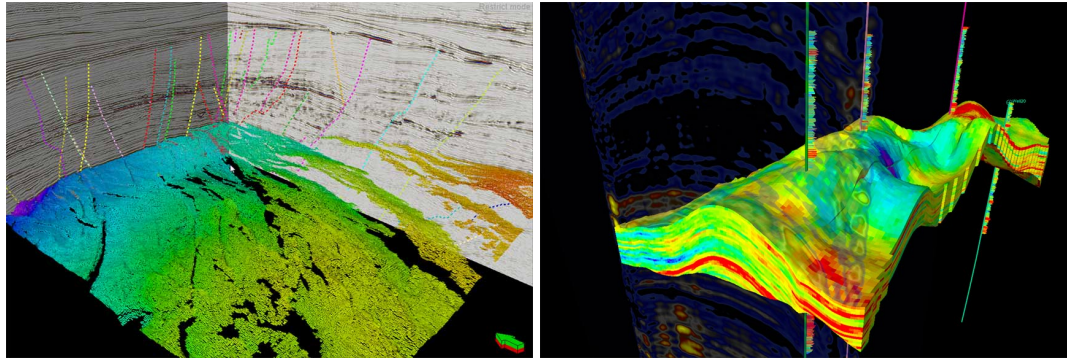


Figure 1.4: The screenshot on the left hand side shows horizons and faults (two geological features) interpreted and extracted from a seismic cube. The screenshot on the right hand side shows a petrophysical model which is created from the result of seismic interpretations and rock properties. Screenshots were taken from Petrel software (<http://www.software.slb.com/products/platform/Pages/petrel.aspx>).

data describing its origin is available to some extent, a method of collecting provenance on the visualization pipeline was well illustrated by VisTrails [5], but the literature lacks research on provenance for interpretations that occur as a result of users' identifications of areas of interest on the final products of the visualization pipeline. This thesis addresses this problem by proposing a provenance model for multi-user interpretations to be captured and recorded back to the data repository.

1.4 Aims and Objectives

The aim of this thesis is:

to formally capture the evolving visual interpretations of 3D spatial data where they occur as a result of users' identifications of areas, representing feature of interest, on the final products of the visualization pipeline. These interpretations may represent both visual interpretations and annotations of interpretations made by multiple users.

Interpretations of this thesis's interest relies on human intuition in a sense that different users (scientists) may reach different interpretations of the same visualization.

Building on this aim, the objectives of this thesis are as follows:

- **Obj. 1:** to investigate the requirements in the workflow of visual interpretation for managing the resulting interpretations by multiple users

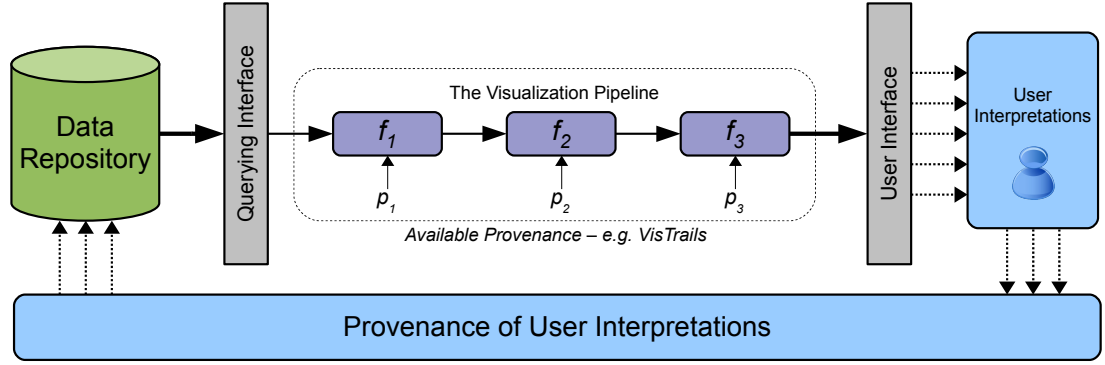


Figure 1.5: The figure illustrates the area which this thesis addresses, filled in blue colour. That is capturing a user's interpretation and its provenance back to the data repository, in the reverse direction to the visualization pipeline. On the other hand, provenance of the visualization pipeline (e.g. VisTrails) captures functions (f) and parameters (p) of its modules.

- **Obj. 2:** to build a model for provenance of users' interpretations of data
- **Obj. 3:** to evaluating the model in the domain of seismic imaging data for the oil and gas exploration workflow

1.5 Contributions

Based on this set of objectives, the contributions of this thesis are as follows:

1. A concept of provenance to track and link multi-user interpretations was developed. This is independent of visualization functions (applied algorithms) and independent of their visual interpretation software. The thesis adopted and extended the W3C PROV data model to build a provenance model for user interpretations.
2. A prototype provenance-enabled architecture was implemented for seismic imaging interpretation. It captures a user's changes of an interpretation along with its provenance to be loaded back into the repository of the raw data. Case studies and interview, with domain experts, were conducted to test the prototype application and conceptually evaluate the contribution, respectively.

1.5.1 Publications

In addition to the above main contributions, the following publications were presented during the period of this PhD. The author of this thesis is the lead author in these publications. The contributions of co-authors M. Rasheed and D. Irving were limited to the utilisation of their previous work, *a massively-parallelised spatially-registered data structure* (SRDS) [23, 24], as a foundation for the data structure of the implementation presented in this thesis. This is further clarified in Section 5.3.1.

1. **A. Al-Naser, M. Rasheed, J. Brooke and D. Irving, “Enabling Visualization of Massive Datasets Through MPP Database Architecture,” in *Theory and Practice of Computer Graphics* (H. Carr and I. Grimstead, eds.), pp. 109-112, Eurographics Association, 2011.**

In this publication [25], the authors introduced the concept of feature-aware parallel queries to a database in order to create a volume in real time ready for direct volume rendering. In this approach, features—which are classically represented by meshes—are stored as points tagged into the database. The implementation of the work of this thesis is based on this concept.

2. **A. Al-Naser, J. Brooke, M. Rasheed and D. Irving, “An Architecture for Shared Multi-User Client Rendering of Massive Geodatasets,” in *The American Geophysical Union (AGU) Fall Meeting*, (San Fransisco), poster, Dec. 2012.**

This is a poster [26] which was presented in a geoscience conference. Its main purpose was to share the architecture with the users community.

3. **A. Al-Naser, M. Rasheed, D. Irving and J. Brooke “A DataCentric Approach to Data Provenance in Seismic Imaging Data,” in *the 75th EAGE Conference & Exhibition*, EAGE Publications bv, Extended Abstract, June 2013**

This is an accepted extended abstract [27] and was orally presented at the 75th EAGE Conference & Exhibition, which is “the largest and most comprehensive geoscience event in the world,” as claimed by the conference organization. This publication “received among the highest ratings from its reviewers” as stated by Gerhard Diephuis, Editor in Chief of First Break which is an EAGE journal. The author, therefore, was invited to submit a journal paper to that journal.

4. **A. Al-Naser, M. Rasheed, D. Irving and J. Brooke “A Visualization Architecture for Collaborative Analytical and Data Provenance Activities,” in *the 17th International Conference on Information Visualisation*, pp. 253-262, 2013**

In this publication [28], which was presented to the visualization community, the authors introduced a reverse direction to the visualization pipeline to store users’ understanding of seismic imaging datasets. The results of this work were presented as case studies illustrating the capabilities of the introduced architecture through the implemented system. The case studies showed the ability to capture users’ amendments of earlier interpretations by others while maintaining data provenance and, thus, allowing history tracking. This article was nominated by the conference committee to be extended into a chapter for a volume titled *Visual Analytics & Business Intelligence*.

5. **A. Al-Naser, M. Rasheed, D. Irving and J. Brooke “Fine-Grained Provenance of Users Interpretations in a Collaborative Visualization Architecture,” in *the 5th International Conference on Information Visualization Theory and Applications*, pp. 305-317, 2014**

This publication [29] extends on the publication above [28]. It includes a survey conducted with geologists and information technology (IT) personnel in both academia and the oil and gas industry. This survey is discussed in Section 4.7. The publication received a best paper award.

6. **A. Al-Naser, M. Rasheed, D. Irving and J. Brooke “User’s Interpretations of Features in Visualization,” in *Computer Vision, Imaging and Computer Graphics: Theory and Applications*, ch. –, pp. –, 2015 (submitted and in review)**

This is submitted as a book chapter, which is an extension to the above publication.

In addition, a solo-author paper is being written to be submitted to First Break, an EAGE journal.

1.5.2 Other Contributions

In addition to the previously mentioned publications, the following were shared with the public domain during the period of this PhD.

1. The code of the prototype implementation was shared with the research community [30].
2. A screen recording video [31] was published on the Web by the author of this thesis demonstrating a walkthrough on some of the main functionalities of the prototype application. This illustrated the capability of the prototype architecture which implemented the proposed provenance model for user interpretations.

1.6 Thesis Approach

The research problem in this thesis is approach as the following structure.

- **Chapter 2** presents background research and related work in visualization and provenance. It first presents general background about visualization. Then it discusses the meaning of a feature in relation to this thesis. Finally, it discusses related work on provenance, focusing on provenance of visualization.
- **Chapter 3** proposes a provenance-enabled interpretation pipeline. It presents the adoption and extension of the W3C PROV data model and illustrates its use for user interpretations via examples. It then discusses implementation considerations.
- **Chapter 4** defines seismic data and explains its acquisition and standard data format. Then it discusses the meaning of geological features, how they are interpreted and extracted and their uncertainty. After that, the chapter gives a background about the visualization of seismic data and discusses an example of industrial application. Then, it discusses the current status of managing seismic interpretation. Finally, it presents and discusses a survey conducted with staff from the oil and gas industry as well as geoscientists from academia as a motivation to this research.
- **Chapter 5** presents a prototype implementation of the proposed interpretation pipeline and provenance model for subsurface interpretation.
- **Chapter 6** presents a walk-through of the implemented applications aided by screenshots. This demonstrates selected scenarios in seismic interpretation.

- **Chapter 7** firstly presents a usability evaluation (functional tests) through some case studies to illustrate the functionality of the prototype architecture. It secondly presents a conceptual evaluation of this research approached by interviewing domain experts. Finally, it presents performance measures (non-functional tests) on the implemented system using commodity machines and then a parallel processing database cluster.
- **Chapter 8** concludes and presents plans for future work.
- **Appendix A** presents responses from some participants from academia to the case study presented in Chapter 7.
- **Appendix B** presents a summary of the survey conducted on staff from the oil and gas industry, which is presented in Chapter 4.
- **Appendix C** presents transcripts of the conducted interviews for the purpose of evaluation.
- **Appendix D** presents a general background information on rendering techniques, some of which were used in the prototype implementation.
- **Appendix E** presents an extension to Voreen for reading seismic SEG-Y format files. This was developed by the author of this thesis during the timeframe of this PhD. However, it is not considered as a contribution to this thesis.

Chapter 2

Background and Related Work

2.1 Chapter Introduction

As discussed in Section 1.3, this thesis is addressing the issue of capturing interpretations which can only be performed after the visualization pipeline has executed. These interpretations are produced by human interaction with a visualization based on the user's knowledge, experience and intuition. This chapter starts with background information about the visualization pipeline and its evolution (Section 2.2), since interpretations are results of visualization. The section also focuses on the use of visualization in interpretation and decision making. Having visualized a desired dataset, this thesis is concerned with *interpretation of features*.

The chapter, therefore, gives brief background about what is meant by a *feature* in Section 2.3. The section also discusses some of the feature extraction techniques. Comprehensive background information about features extraction is not provided in this section since this thesis is not concerned with the algorithms and human mental processes involved in identifying features, but rather with the management of these features within a collaborative decision making process based on visualization.

To evaluate the methods of the thesis, features that have been previously identified by geoscientists have been used as the basis for the provenance management. Therefore, the rest of the chapter presents background information and related work about provenance in the field of computer science (Section 2.4). Background information on seismic imaging data, the field to which the work of this thesis is applied, is presented later in Chapter 4. It includes a closer look into the meaning of a feature in seismic data and how it is interpreted. Figure 2.1 illustrates a conceptual diagram of the disciplines related to this thesis.

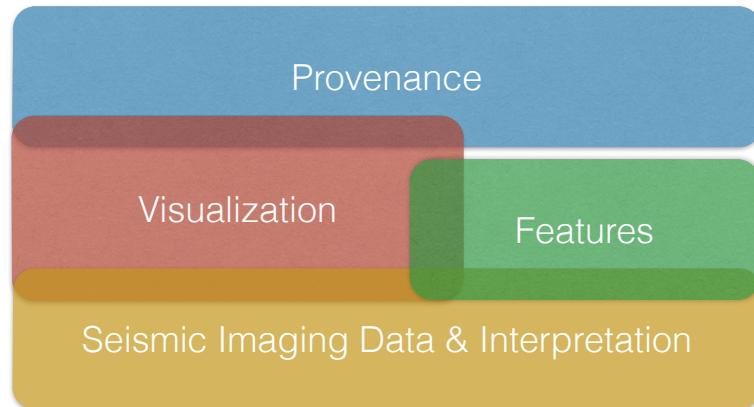


Figure 2.1: A conceptual diagram of the disciplines related to this thesis.

2.2 The Visualization Pipeline

Most visualization implementations follow the conventional visualization pipeline described by Haber and McNabb in 1990 [1]. Referring to Figure 1.2, raw data is firstly modified to be ready for visualization; noise reduction and derived data calculations are examples of such modifications. Derived data is then mapped into graphics primitives. These are geometric representations such as points, lines and triangles along with attributes such as colours, texture and transparency. Finally these primitives are rendered (drawn) by specialist graphics processing hardware or software to produce images, nowadays usually implemented on *graphics processing units* (GPUs). Thus, the visualization pipeline builds the visual objects presented to the user in the form of a data processing workflow that starts from the original data continuing right up to the rendering on the display device. A basic visualization pipeline features the following modules in the order of execution: reader \rightarrow geometry generator \rightarrow renderer, as highlighted by Moreland [2].

The visualization pipeline is of interest to the work of this thesis because one of its main purposes is that it allows users to use their visual abilities and intuition in understanding numerical data. In the case of seismic imaging, as will be explained in Section 4.2, data in its original form is mainly a collection of *amplitude* values. The understanding of seismic data is known to geoscientists as *interpretation*, where geological features of interest are extracted. These features, in fact, exist in the data but need to be identified for their importance in the potential discovery of a hydrocarbon resource. Seismic interpretation is a subjective process, at least in part. This means

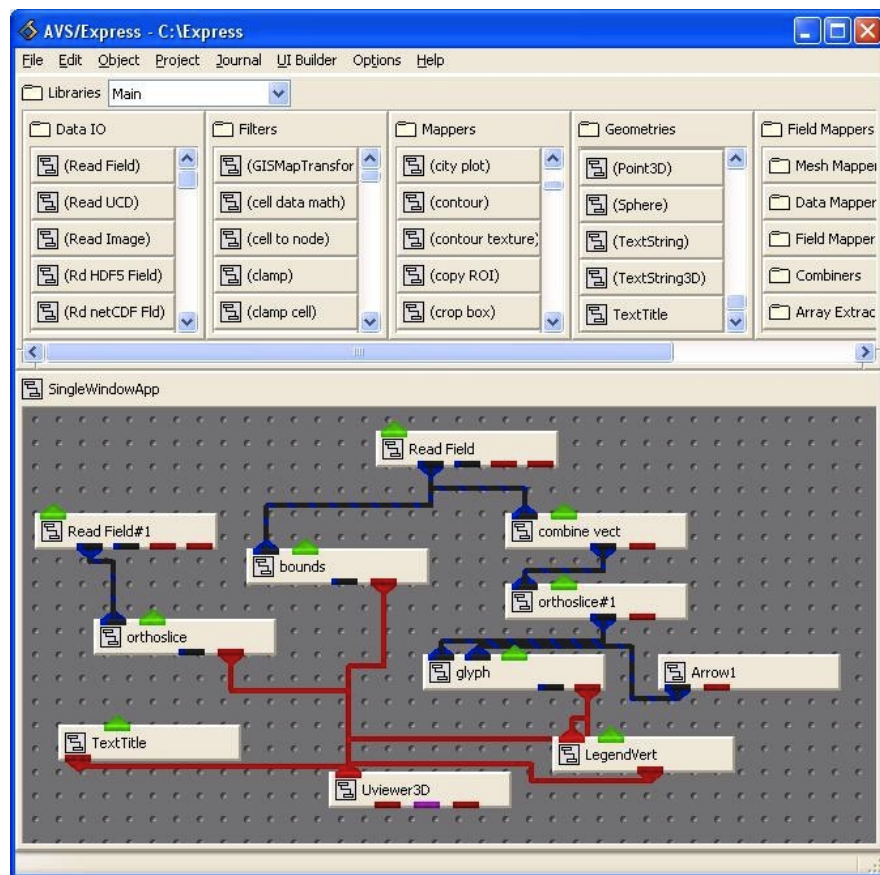


Figure 2.2: An “AVS network” created by selecting and connecting modules.

that, in addition to the use of extraction algorithms, the process involves manual interpretations based on human intuition. Thus interpreted features are stored explicitly to be identified independently of any extraction method. Conventional visualization pipelines are uni-directional. This forces seismic visualization systems to store the extracted features separately from their original datasets and thus might lead to inconsistency. Therefore, this thesis investigates a returning pathway for interpretations to tag the original datasets in a consistent model.

2.2.1 Use of Visualization for Interpretation

Visualization allows us to view a graphical representation of numerical data. As previously mentioned, one of the main advantages of visualization is to understand the data and thus interpret it. The term *interpretation* in association with *visualization* might slightly differ among its applications. The following presents some examples of applying interpretations on different scientific fields.

In the field of medicine, 3D visualization is used on computed tomography (CT) scans. Mang et al. [32], for example, compared two visualization methods for interpreting images of colonic wall for detecting colorectal polyps. One method is using a standard (non-panoramic) view and the other is based on a panoramic view; the latter provided a faster interpretation. In this example, the interpreters are looking for colorectal polyps, which are the desired feature in this case. These polyps are marked digitally by the users to record them.

In the field of software engineering, Moraga et al. [33], for example, used a visualization method to easily interpret the quality of component-based software systems. Their method was an alternative to the conventional quality models that produce numerical measures, which might be challenging and time consuming for users to understand and interpret. Gras [34] discussed the importance of distinguishing between interpretation skills of individuals and team collaboration in the field of oil and gas exploration. Exploration heavily relies on individuals' knowledge, which cannot be generalised, but also requires a collaboration at some stages.

2.2.2 Pipeline Evolution

The visualization pipeline has undergone extensive elaboration since its formulation more than twenty years ago [2]. Improvements and elaborations have been proposed to address a variety of issues such as visualizing multiple datasets, visualizing large

datasets and enhancing performance and efficiency. The following presents some areas where the classical visualization pipeline has evolved highlighting their association to the research problem of this thesis.

2.2.2.1 Metadata

A fuller description of data by metadata enhanced the power of visualization by allowing a more full-featured view of the data taking into account its special properties and allowing users flexibility in creating visual objects. Using metadata, users can select a region or multiple regions to process; for example this allowed Ahrens et. al. to visualize large-scale datasets using parallel data streaming [35]. In addition to regional information, a time dimension can be added to metadata, adding time control to the visualization pipeline [36].

The usefulness of metadata was further developed with the introduction of *query-driven visualization* [37, 38, 39]. It enables users to specify a data of interest matching certain criteria to analyse. Query-driven pipelines require the following technologies: file indexing, a query language and a metadata processing mechanism to pass queries. For fast retrieval, indexing technologies are used, such as FastBit [40, 41] (based on compressed bitmap indexing) and hashing functions. For example, the work of Stockinger et al. [37] allows to visualize data based on a compound Boolean queries against properties like pressure and temperature; i.e. “ $(temperature > 1,000) AND (70 < pressure < 90)$ ”. In the field of subsurface data, properties of interest include porosity and permeability [42]. The query should be passed from the renderer end to the data source through the visualization pipeline.

Recalling the research problem of this thesis, multi-user visual interpretations need to be captured and tagged on the original (visualized) dataset. This would involve capturing multivariate metadata which need to be queried, that is in addition to the metadata of the original data (e.g. spatial reference), for users to visualize data of their interest. Therefore, in this thesis a form of query-driven visualization is implemented as will be explained in Chapter 5.

2.2.2.2 Parallel Pipeline

To handle massive datasets efficiently, the visualization pipeline can be executed in parallel over multiple nodes. This was well illustrated with MapReduce [43]. Database management systems (DBMS) are also capable of parallel execution of analysis but

were not widely used for the purpose of visualization. A comparison between MapReduce and DBMS was presented by Pavlo et. al. [44], where the authors suggest that DBMS has a performance advantage over MapReduce while the latter is easier to set up and to use.

Two popular solutions allowing consumer machines to visualize massive datasets, from the parallelism offered by a cluster, are remote geometry delivering—also called remote visualization—and remote rendering (as described in the thesis of Ge [45]). In fact, remote visualization was the only method to visualize large datasets in the 1990s, when commodity machines lacked parallelism.

In remote geometry delivering mode, the geometry stage of the visualization pipeline is performed on a remote visualization server producing rendering commands sent over the network to the client for local rendering. This mode gains the advantage of servers' power in data retrieval and geometry calculations but still relies on the client end machines to perform rendering. This means that a complex volume would still require a high-end machine for rendering.

In the second solution, remote rendering, both geometry generation and rendering are performed on remote visualization servers producing a stream of images as pixels sent over the network to the client. The rendering, and thus image production, is based on the client's viewing requests. In this mode, low level client machines can visualize massive models as the complexity of the geometry and rendering calculations is actually independent, being handled by powerful servers. VirtualGL¹ provides such a solution.

Remote rendering may provide a faster and more integrated experience. However, in general it has some disadvantages including the following: (1) it requires a permanent connection with the server and thus offline rendering is not possible, (2) it may not provide a high quality rendering and (3) it is more expensive to establish a remote rendering infrastructure than a client-only rendering solution. Therefore, Jomier et al. [46] suggest that combining remote rendering and client rendering solutions into a hybrid system will allow a greater experience in terms of flexibility.

Some seismic visualization applications offer an option of a remote rendering environment. However, geoscientists often prefer to work on local subdatasets (locally rendered) for the above mentioned reasons. Concerning to the work of this thesis, remote rendering may offer the possibility of maintaining the results of multi-user interpretations physically next to their original datasets. However, this maintenance is

¹<http://www.virtualgl.org/>

application dependent.

2.2.2.3 Locality

Brooke et al. [47] discussed the importance of locality in visualizing large datasets. They highlighted issues with remote visualization for the purpose of real-time interactive analysis, including limitation of bandwidth and latency of wide area network (WAN). The authors addressed the problem of visualizing large oceanographic data bringing real-time visual analysis to users' desktops. This was achieved by performing visualization locally on a personal desktop (or laptop) utilising spatial locality and through accessing data remotely via web services. A local rendering solution was proposed for analysing large datasets when selecting a specific time slice, as this type of analysis is required to be performed interactively in real-time. Clients were also offered to utilise visualization servers to render time-variant datasets in fine detail as this is usually not required interactively or in real-time. A direct volume rendering technique was performed on the client side. Voxel data were represented as 3D texture that is supported by programmable graphics units. View-aligned slices of 2D textures are rendered, creating a 3D non-photorealistic illustration of the data. A disadvantage of this approach was that voxel data with floating point data type cannot be directly mapped to a colour with a value ranged from 0 to 255. This was in fact a limitation of the graphics hardware at that time. Therefore, a pre-processing stage was performed to compress voxel data into a colour value. Also, level of detail (LOD) was not addressed in order to minimize data over the network.

The work of Brooke et al. [47] has inspired the work of this thesis through its method of rendering interesting surfaces. As is the case with seismic data, surfaces of interesting features can be fractured and noisy; they are not regular shapes.

2.3 Features in Visualization

Feature-based visualization (i.e. visualizing a dataset that contains features) aims to extract structures within the visualized dataset, and possibly displaying only those, desired, structures to the user [48]. A feature, as described in the thesis of Reinders [49], is “any object, pattern or structure in the (visualized) data that is of interest and that is a subject of investigation”. The goal here is to extract those features along with quantitative attributes that describe their characteristics.

The term feature extraction may describe the process of extraction, attribute calculation and visualization of features; these are described as follows. The feature extraction process aims to algorithmically and explicitly extract features, rather than relying on the visual perception of the user to observe the features within a dataset. The user can then use his or her perception and judgment to either confirm or reject an interpretation of the dataset. Attribute calculation provides a quantitative measure of the characteristics of a feature. Thus, it further describes the feature, compared to just a pure extraction. Quantitative measures also allow selecting a subset of interest of the extracted feature; thus data reduction.

With the availability of many feature extraction methods, choosing a suitable one is not straightforward. Existing features might be missed from being extracted. In other cases, features that were extracted may not be features as defined by the observer (user). Thus extracting all features remains uncertain; it is an exploration process. Some extracted features require visual symbols to be represented, as they do not have concrete shapes. Methods differ according to the required application, such as medical (anatomical features) and flow visualizations.

As presented above, feature extraction generally relies on algorithms to define the desired feature based on some user inputs. However, in fields like seismic imaging interpretation, users must manually complete or modify the result of the automatic extraction by the algorithm to identify a feature based on their understanding and intuition of the visualized data.

Application Example—Medical Visualization

In medical visualization, data is measured by devices such as CT and MR scanners. This produces 3D images of the human body. Human organs and tissues can be extracted from these 3D images using automated segmentation methods with input from users (specialists). Features in this case, which users can visualize, include human brain, kidney, blood vessels, heart, liver and tumours. Medical features extraction techniques are mainly based on segmentation or surface reconstruction; these techniques are widely applied for features of other fields as well. In medical visualization, feature extraction techniques are challenging because of the various shapes of anatomical structures and lack of distinct boundaries between them.

2.3.1 Features Detection Techniques

Extensive research has been done on features detection and extraction² techniques (e.g. [50, 51, 52]). Feature detection techniques can be associated with the following four concepts [52]: image processing, topological analysis, physical characteristics and partition-based approaches. This section presents general background about feature detection techniques. Examples of extraction methods for seismic imaging data are discussed in Section 4.3.2.

Many of the extraction techniques are based on *segmentation*, which is the process of partitioning an image into multiple regions [53]. Segmentation relies on the result of image processing to partition [54]. Pixels of a region have similar image characteristics or computed properties; thus objects in an image can be identified. Image segmentation is commonly initiated with *edge detection* techniques [53]. An edge in an image is detected through a significant change in image intensity, which is a sign of discontinuity. Another segmentation-based technique is *region growing*, which is common in the medical field (a survey on interactive image segmentation techniques in the medical field is provided by Zhao and Xianghua [55]). The process usually starts with user selection of starting (or seeding) points. Kadlec et al. [56] adopt segmenting using level sets with domain knowledge for detecting geological features.

To construct a surface from the result of segmentation, a surface reconstruction technique is needed. This can be achieved by creating a triangular mesh over the segmented area. Surface reconstruction techniques include marching cube. The resulting number of triangles can be large and therefore it would be required to reduce it using surface simplification techniques. Another way of reconstructing surfaces of anatomical structures is by fitting the image volume into a parametric model. The model can then be mapped into a geometric object for display.

2.4 Provenance

In this section, the term *provenance* is first defined as an English word then in the context of computer and information science; the latter is of a concern to this thesis. Then, it discusses types of provenance. The W3C PROV data model is then discussed.

²In the literature, the term *detection* and *extraction* may overlap. Feature detection may refer to the process of finding a feature of interest while feature extraction may refer to the process of representing the found feature. In this thesis, the overall meaning of detection and extraction are referred to by either terminologies.

Last, the section discusses background information about provenance in visualization.

2.4.1 What is Provenance?

In relation to this thesis, two definitions of the term *provenance* by the Oxford English Dictionary³ are:

- (i) *“The fact of coming from some particular source or quarter; origin, derivation.”*
- (ii) *“The history of the ownership of a work of art or an antique, used as a guide to authenticity or quality; a documented record of this.”*

In the field of computer science, provenance is metadata that describes the history of a dataset and how it was derived. It has gained importance in various research areas in computer science, such as curated databases⁴, e-science, Semantic Web⁵ and recently visualization. Provenance is an “intrinsic property of data” since it gives a value to the data [58]. In databases, provenance resolves the question of who created a record and when and where it was created. In the Web, provenance helps to identify the source of a content, and thus determining trust. In e-Science, Simmhan et al. [59] describe provenance as:

“information that helps determine the derivation history of a data product, starting from its original sources.”

Such definition is, to a certain extent, applicable to other sub-fields of computer science. In workflow applications, “Provenance is the result of a query over a set of assertions about execution” [58]. Information about executed processes are accumulated to be, then, retrieved (as provenance) via query interfaces. This provenance approach is widely adopted by workflow systems including Taverna⁶ and Kepler⁷.

A variety of standard querying technologies are used to query provenance, such as SQL and SPARQL. Their disadvantage to provenance-enabled systems is the difficulty of changing the schema in future. Alternatively, provenance-oriented query languages

³<http://www.oed.com/>

⁴ A *curated database* is “any kind of structured repository such as a traditional database, an ontology or an XML file, that is created and updated with a great deal of human effort” [57].

⁵ According to the W3C, “The **Semantic Web** provides a common framework that allows **data** to be shared and reused across application, enterprise and community boundaries”. It aims to make the Web a “web of data” that can be analysed by computers.

⁶<http://www.taverna.org.uk>

⁷<https://kepler-project.org>

were proposed, which are generally a translation to standard query language; examples of these include the query language for provenance (QLP) [60] and the VisTrails provenance query language (vtPQL) [61].

There is a wide range of provenance literature across subfields of computer science. For example, Simmhan et al. [59], Cheney et al. [62], Freire et al. [63] and Di et al. [64] surveyed provenance work in e-science, databases, workflow systems and geoscience respectively. This chapter does not go in detail about work on provenance across those subfields but instead focuses on work related to the visualization pipeline.

It should be highlighted that although the application of this thesis is on geoscience related data, the term provenance of its work does not refer to the context of geology but to the field of computer and information science. In geology, provenance is commonly related to sedimentary rocks; it refers to the origin of the sediments. The study of sedimentary provenance investigates the sediments source area, how they traveled over time from their source area to where they are today (basin of deposition) and what influenced their composition [65].

2.4.2 Characterizations of Types of Provenance

2.4.2.1 Usage of Provenance

The usage of provenance is considered when its metadata is captured. Goble [66] identifies applications (or uses) of provenance, which include:

- **Reliability and Quality** – determining the quality of data based on its sources and the processes that lead to the production of data.
- **Justification and Audit** – using provenance metadata as a historical record of the sources and methods that were used to produce data.
- **Reusability and Reproducibility** – repeating and validating an experiment.
- **Ownership and Credibility** – determining who should be credited for a dataset.

In reference to the above list and in relation to this thesis, provenance of users' interpretations need to be captured mainly for reliability, reusability, ownership and credibility. Provenance information should allow users to determine the reliability of an interpretation. Also, having the ability to access and track multiple opinions (interpretations) should allow users to reuse some previous attempts. In addition, provenance information should determine ownership of an interpretation for credibility, and

can also determine its reliability.

2.4.2.2 Subjects of Provenance

Simmhan et al. [59] and Ikeda et al. [67], in their surveys, categorize provenance techniques with different characteristics, two of which are (1) subject, or type, of provenance and (2) granularity. Data provenance can be of two types:

1. *data-oriented* (or *where-lineage*)—which is a provenance applied to the data explicitly answering the question of which dataset(s) contributing to produce the output.
2. *process-oriented* (or *how-lineage*)—which is a provenance applied to the process answering the question of how the resulting output was produced from the input data.

Both types can be applied on one of two granularity levels: (1) coarse-grained (schema level) and (2) fine-grained (instance level). The latter deals with individual data items separately.

2.4.3 W3C’s PROV—A Provenance Model

PROV [68] is a family of documents with specifications to model, describe and publish provenance information. It was recently formalised by the W3C provenance working group for interchanging provenance information in heterogeneous environments such as, and not exclusive to, the Web. The PROV specification overview [68] defines provenance as:

“information about entities, activities, and people involved in producing a piece of data or thing, which can be used to form assessments about its quality, reliability or trustworthiness.”

PROV data model (PROV-DM)⁸ [70] is domain independent. Native (field specific) provenance can be transferred to PROV-DM. Thus, heterogeneous systems can interchange their native provenance data through this (PROV) model. Provenance data is important to (1) trust data, (2) integrate data with others and (3) give credit when re-using data. In the Web, information can be contradictory and thus provenance can help

⁸PROV-DM, informally, succeeded the Open Provenance Model (OPM) [69]; both were co-founded by Luc Moreau.

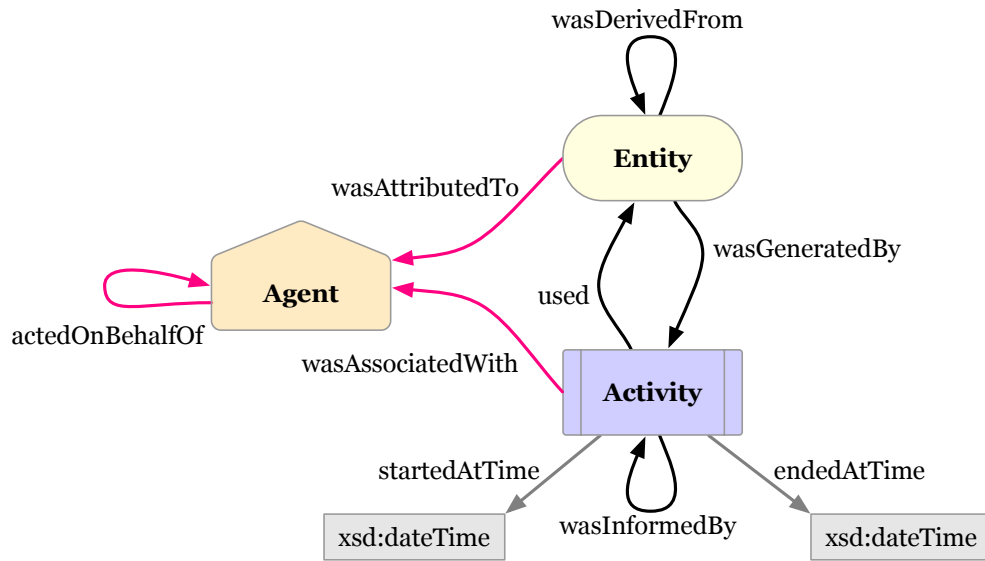


Figure 2.3: The figure illustrates core elements of PROV data model. (diagram source: <http://www.w3.org/TR/2013/REC-prov-o-20130430/>)

in making a “trust judgment”. PROV-DM is a graph-based provenance description. Its core elements are illustrated in Figure 2.3. PROV-DM has three core types: *entities*, *agents* and *activities*. An entity is what we intend to describe its provenance; it may be a piece of data or information. An agent is a user, an organization or a software that causes an activity to occur. An activity is the usage, generation or modification to an entity by an agent. These core concepts are joined through well-defined relationships which include usage, derivation, attribution and generation. For example, in a *usage* relationship an activity utilises an entity while in a *generation* relationship an activity produces an entity. PROV-DM allows subtyping. For example, a *person* and a *software agent* are two subtypes of the *agent* core type. A full description of the PROV-DM is provided by the W3C [70]. The primary implementation of the PROV model is the PROV Ontology (PROV-O) [71], which is implemented using Semantic Web standards. Different implementations can utilise the PROV model as a foundation for their provenance description. This thesis adopts and extend this model for the purpose of provenance of users’ visual interpretations; this is presented in Chapter 3.

2.4.4 Provenance in Visualization

Moreland’s recent survey on the visualization pipeline [2] defines *provenance* as the captured changes to the pipeline during an exploratory process. This was demonstrated by the model put forward by Jankun-Kelly et al. [72] and the introduction of *VisTrails*

[5, 6, 7]; the latter dominated the work surveyed by Moreland in regards to integration of provenance into the visualization pipeline. An earlier attempt at integrating provenance into a visualization environment was by Woodruff et al. [73], which is based on storing an inverse function to allow scientists to refer to earlier generated pixel images. Also, the work of Groth and Streefkerk [74] captures user interactions and annotations to the visualization. The following text further discusses VisTrail and the work of Groth and Streefkerk, which are of interest to this thesis.

VisTrails is a visualization application that captures the changes to the pipeline modules and parameter values performed by the user. It uses a *changed-based* provenance mechanism for workflows [61, 75], where provenance data are organized in layers. It allows scientists to revisit a workflow and its parameters settings and thus reproduce desired results. Users' actions, such as adding and deleting modules and setting and modifying values of parameters, are captured by the *VisTrails History Manager* and saved in the *VisTrails Repository*. Users can also explore different parameter settings. *VisTrails Log* records which modules were executed plus other run-time information such as when, where and by whom. In VisTrails, changes to a workflow is stored as a version tree. Changed-based mechanism allows to save only the changes to workflows, to eliminate redundancy and save in required space. Provenance information in VisTrails is organised into three layers:

1. *workflow evolution layer* — (also called version tree or *vistrail* level) captures information about the relationship among the different workflows created by the user.
2. *workflow layer* — contains information about specifications of each workflow, including information about modules and its parameter values.
3. *execution layer* — (logs) captures run-time information such as which modules were executed, when, where and by whom. Such information is captured by the *Workflow Execution Engine* and stored in *VisTrails Log*.

Provenance information is automatically captured. In addition, users can manually add annotations at any layer. VisTrails implementation stores version trees in an XML file and it stores logs (execution information) onto a relational database (MySQL). VisTrails architecture also includes a query server for users to query provenance information. VisTrails query language (vtPQL) takes advantage of the different layers; it is an SQL-like language. It breaks a query statement into pieces for each to be assigned to its referenced layer. Query results are visually displayed.

As opposed to VisTrails, Groth and Streefkerk [74] presented a model to capture steps (state of visualization) that were taken by a user to achieve a visual representation, such as *zooming* and *rotating*. Thus, it is a provenance of a visualization's viewing manipulations. User annotations on each state of the visualization are also captured in this model. The interactions and annotations are recognized as first-class objects.

The above work on provenance of visualization is process-oriented. In VisTrails and the work by Groth and Streefkerk [74], it is assumed that the knowledge discovery process is associated with the steps taken by the user to discover the knowledge, from which the provenance information is derived. Looking at the visualization pipeline, the user interactions for exploration take place prior to rendering. Those interactions are the steps that were taken by a user to create a visualization pipeline or achieve a visual representation. This thesis recognises the importance of capturing this knowledge discovery process. However, in fields like seismic imaging data, the exploration process, represented by interpretation of features, is completed after the visualization pipeline has executed and requires user identifications of structures within the visualized data. In the latter case, a new provenance form is needed to capture the evolving interpretations of users as a result of visualization.

2.5 Chapter Summary and Conclusion

The chapter presented background information and related work in three main areas: visualization, features in visualization and provenance.

The visualization pipeline is a processing workflow that builds and renders a visual object from a dataset. Visualization allows users to visually understand, and thus interpret, data in many fields such as medicine and subsurface imaging. The visualization pipeline has undergone impressive developments since its formulation more than two decades ago. The chapter presented examples of such developments that are of interest to this thesis. Querying metadata for visualization is of importance since it is aimed, in this thesis, to capture users' interpretations that include multivariate metadata, which need to be later queried. Also, the thesis adopts a parallelism at the data level, by utilising a parallel database as will be presented in Chapter 5. These notions are adopted to introduce a reverse direction to the visualization pipeline to load back multi-user interpretations to the original datasets in a consistent model (i.e. provenance model).

The chapter then presented brief background about features in visualization. It

covered some examples of feature detection techniques; those dedicated to the interpretation of seismic imaging data will be discussed in Section 4.3.2. It should be noted that this thesis is not concerned with the algorithms involved in identifying features, but rather with the management of these features. Therefore, features that have been previously identified by users will be used to evaluate the method of this thesis.

Finally, the chapter presented background information about provenance, discussing the W3C PROV data model and state of provenance in visualization. Provenance is metadata that describes the history of a dataset, and thus this thesis adopts its notion to resolve the issue of tracking users' interpretations. Previous work has introduced provenance into the visualization pipeline, such as VisTrails which captures metadata of the pipeline's modules. It can be noted that VisTrails treats workflow modules as first-class objects in its provenance solution. However, research currently reported in the literature lacks a method of recording provenance of users' interpretations, which encodes the intuition of users through actions performed on the rendered visualization. Therefore, this thesis adopts and extends the W3C PROV data model for this purpose, where users' interpretations are captured and treated as first-class objects. This is presented in the next chapter.

Chapter 3

A Provenance-Enabled Interpretation Pipeline

3.1 Chapter Introduction

The previous chapter discussed background information in three main areas: visualization in general, features in visualization and provenance in visualization. Feature extraction methods help in identifying features of interest in the visualized dataset. However, in domains where data is of a great uncertainty due to noise and other factors, the identification of features can only be completed manually by human, after the help from feature extraction methods. In such domains, two users visualizing the same view of a dataset may interpret a feature differently based on their background knowledge and intuition. Thus an interpretation represents a user's opinion that cannot be generated via an algorithm. Therefore, such features are explicitly stored to be able to revisit, compare, and reuse them in subsequent processes within an exploratory workflow.

Tracking the origin of the different interpretations and annotations applied to them is challenging. The literature, as discussed in the previous chapter, introduced the notion of provenance to the visualization pipeline to capture metadata about the origin of a pipeline's modules. However, research currently reported in the literature lacks a provenance for users' interpretations that only occur after the visualization pipeline. These interpretations, which are derived from actions performed with tools that enable the user to point at and to draw on the rendered visualization, form a basis for encoding the intuitive interpretation of the feature.

Through this chapter, this thesis proposes an *interpretation pipeline* which runs in

the reverse direction to the visualization pipeline to encode and store interpretations and their metadata next to the raw data; this is presented in Section 3.2. This interpretation pipeline is provenance enabled as it incorporates a provenance model to record the origin of users' interpretations. The presented provenance model is based on the W3C PROV data model, which was discussed earlier in Section 2.4.3; the proposed model of this thesis is presented in Section 3.3. Some implementation considerations are then discussed in Section 3.4. The next chapter (Chapter 4) discusses the application field of this thesis, which is visualization and interpretation of seismic imaging data. This information presented in this chapter and the next chapter is used for a prototype implementation for interpretation of seismic data which is presented in the subsequent chapter, Chapter 5.

3.2 Interpretation Pipeline

In this section, the thesis proposes a provenance-enabled interpretation pipeline; see Figure 3.1. The primary purpose of the interpretation pipeline is to bring multi-user visual interpretations into a consistent provenance model and to tag them onto the raw data being interpreted. The next section (Section 3.3) is dedicated to discussing and presenting this provenance model. This is in contrast to the conventional method where extracted features are stored as independent objects. Often, they are stored as ASCII files, each containing x,y,z information of the points that identify the feature in a 3D coordinate. Otherwise, any kind of a relation to the corresponding raw data and annotations are usually maintained under the visualization and interpretation application; therefore metadata is lost when interpretation (features) are exported. In such applications, like Petrel [76] for seismic imaging interpretation, interpreted features are often stored as binary representations (application-owned format) within their corresponding projects' file system.

An interpretation, in the context of this thesis, results in defining and extracting a feature from a visual scene. In addition to features, users should also be allowed to annotate any part of a previous interpretation to, for example, evaluate or assess its quality or accuracy. Thus, a feature or annotation requires the definition of an explicit object (e.g. a region of a surface). Whether a user intends to interpret or annotate, the input to the interpretation pipeline is a selection on the screen; that is in the case of a manual interpretation which is the focus here.

The interpretation pipeline runs in the reverse direction to the visualization pipeline.

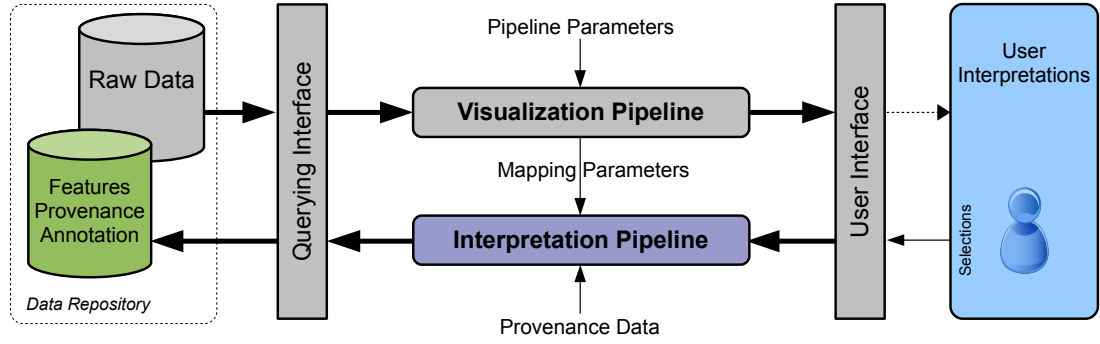


Figure 3.1: The figure is a high level conceptual illustration of the proposed provenance-enabled interpretation pipeline which captures multi-user interpretations. The pipeline incorporates a provenance model to record metadata about the origin of the interpretations. Coloured components, excluding gray-coloured, indicates areas of contributions by this thesis.

Referring to the abstract visualization pipeline described by Haber and McNabb (Figure 1.2), the interpretation pipeline proposed here is, conceptually, in a reverse direction that enables the interpretations by users to be recorded as metadata alongside the original data. Bearing in mind that interpretations are first-class objects in this pipeline, they also have metadata. In abstract, the interpretation pipeline executes as follows. It takes an input from a user’s selection on a rendered volume. Then it reverse maps the selected area, utilising mapping parameters from the visualization pipeline, to define its absolute location against the raw data. Finally, it uploads the interpretation along with its metadata to the provenance model, which tags the corresponding raw data with this interpretation and its provenance data. It is the role of the provenance model to link between the following: raw data, an interpretation, a modification to an interpretation and an annotation to an interpretation.

The visualization pipeline can then take full advantage of the provenance data to query interpretations related to the visualized raw data as well as against user-defined filtering criteria. Using the provenance data, users should be able to, for example, track historical changes to an interpretation and to visualize an interpretation made by a selected (queried) user or assessed according to certain criteria.

3.3 A Provenance Model for Interpretations

This section presents a model that captures multiple versions of interpretations and annotations on existing interpretations by multiple users. An annotation on interpretations includes for example a user assessing the confidence of their own or others' interpretations. This model serves the purpose of (1) asynchronous collaboration where one user can take over an interpretation task from another, (2) generating multiple opinions (interpretations) on a single area to be later discussed among colleagues and experts or (3) annotating an exiting interpretations (e.g. assessing a region of one or multiple interpretations). This model is the basis for the implementation presented in Chapter 5.

The description of the model is based on the notions of *provenance*, adopting the PROV data model (PROV-DM) [70]. PROV-DM, which was first presented in Section 2.4.3, is adopted to describe the origin of data or annotation, to answer the question of who created such data or assessment and when they were created. In this section, a definition of the proposed model is first presented, where the adoption and extension to the PROV-DM is discussed. Then, the proposed model is illustrated through examples.

3.3.1 Definitions

Consequently, “the model” is referred to the provenance model for interpretations presented by this thesis. It is represented as a directed acyclic graph (DAG), adopting the PROV-DM which was discussed in Section 2.4.3. The provenance diagrams presented in this chapter follow, to a high extent, the PROV graph layout conventions¹. Types (*entities*, *activities* and *agents* to be discussed next) are represented as nodes with shapes as illustrated in Figure 3.2. Relationships, such as *usage* and *generation*, are represented as directed edges. Entities are laid out in the order of their generation. Thus, a graph shows time progression from left to right, and edges for relationships typically point leftwards. This is because the notion of time ordering is crucial to the recording of provenance. The time order in which actions are performed and interpretations are created is important. Graphs are expressed using the PROV notation (PROV-N) [77].

¹<http://www.w3.org/2011/prov/wiki/Diagrams>

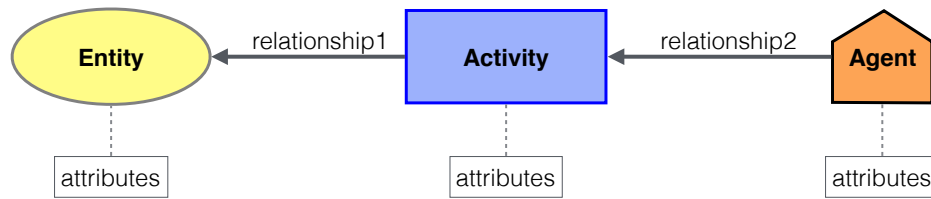


Figure 3.2: The figure illustrates the colouring and shapes of PROV core elements as part of its graph layout conventions. Types (*entities*, *activities* and *agents*) are represented as nodes. Relationships, such as *usage* and *generation*, are represented as directed edges.

3.3.1.1 Classes

The model adopts the three classes, or core types, of the PROV-DM according to the following description. Subtypes that this thesis presents for the model are referred to within the following list and later in this section when discussing the *namespaces*. An instance of a class has a mandatory *identifier* and optional *attributes*. Table 3.1 summarises the classes presented below with their corresponding PROV-N syntax style.

- *Entity* – an entity refers to a raw data being visualized, an interpretation or an annotation; the latter two are data being generated as a result of an interpretation activity. The type of entity is defined through the value of its *type* attribute.
- *Agent* – an agent is either a software agent, which performs a feature auto-extraction as a form of interpretation, or a person, who performs a manual interpretation. A person type agent can also take the role of setting up parameters of the extraction method by the software. The type of agent is defined through the value of its *type* attribute.
- *Activity* – an activity refers to an interpretation. An interpretation activity can be of the following subtypes: (1) addition, which contributes toward an interpretation of a feature; (2) deletion, which deletes from a previous interpretation; or (3) annotation, which annotates (e.g. assess) a previous interpretation. The entity generated by a deletion or annotation type activities are related to either a whole or part of another entity. The type of activity is defined through the value of its *type* attribute.

Type	PROV-N Functional-Style Syntax
Entity	<code>entity(id)</code>
Agent	<code>agent(id)</code>
Activity	<code>activity(id)</code>

Table 3.1: A summary of the model’s classes.

3.3.1.2 Relationships

The model adopts the following relationships, or properties, which join the above core types. Sub-relations that this thesis presents for the model are referred to within the following list and later in this section when discussing the *namespace*. A relationship may consist of mandatory and optional *arguments*, such as an identifier, time and attributes. The following list mentions arguments that are mandatory by the model; such arguments may be optional by the original PROV-DM. Table 3.2 summarises the relationships presented below with their corresponding PROV-N syntax style.

- *Generation* – is the production of an entity by an activity. An interpreted feature (entity) is generated by an interpretation (activity). The model requires a generation relationship to define the following arguments: (1) an identifier for a created entity, (2) identifier for the activity that creates the entity and (3) the generation time.
- *Usage* – is the utilisation of an entity by an activity. An interpretation process (activity) uses raw data (entity) to produce interpreted features. The model requires a usage relationship to define the following arguments: (1) an identifier for the activity that used an entity and (2) an identifier for the entity being used.
- *Derivation* – is a relationship between two entities which can take several forms. To the interest of the model, an interpreted feature (first entity) is derived from the visualization and interpretation of a raw data (second entity). The model introduces two main sub-relationships of derivation: addition and deletion. A new interpretation (first entity) can be added to an existing interpretation (second entity) of the same feature; this describes a continuation of contribution to the interpretation of a single feature. A user can also delete from an existing interpretation; this can be represented as an interpretation (first entity) which is deleted from a previously existed interpretation (second entity). The model also allow sub-relationships of derivation that fall under *annotations*; one example is

Relationship	Related Types	PROV-N Functional-Style Syntax
Generation	$\text{entity}(e1) \rightarrow \text{activity}(a1)$	<code>wasGeneratedBy(e1, a1, time)</code>
Usage	$\text{activity}(a1) \rightarrow \text{entity}(e1)$	<code>used(a1, e1)</code>
Derivation	$\text{entity}(e1) \rightarrow \text{entity}(e2)$	<code>wasDerivedFrom(e1, e2, type)</code>
Attribution	$\text{entity}(e1) \rightarrow \text{agent}(a1)$	<code>wasAttributedTo(e1, a1)</code>
Association	$\text{activity}(a1) \rightarrow \text{agent}(ag1)$	<code>wasAssociatedWith(a1, ag1)</code>
Membership	$\text{collection}(c1) \rightarrow \text{entity}(e1)$	<code>hadMember(c1, e1)</code>

Table 3.2: A summary of the model’s relationships.

assessment on a previously existed interpretation. The model requires a derivation relationship to define the identifiers of the two related entities. The type of derivation is defined through the value of its *type* attribute

- *Attribution* – relates an entity to an agent. An interpreted feature (entity) is attributed to a user or a software (agent) for its generation. The model requires an attribution relationship to define the identifiers of the related entity and agent.
- *Association* – defines the assignment of responsibility of an agent for an activity. An interpretation task (activity) is associated with a user or software (agent). The model requires an association relationship to define the identifiers of the related agent and activity.
- *Membership* – defines the belonging of an entity to another entity of type *collection*. A *collection* is a type of entity acting as a structure to some other entities. If a feature was interpreted by two users or interpreted in two phases, the latest view, or version, of the interpretation (third entity) is composed of the initial interpretation (first entity) plus the additional interpretation (second entity). Thus, the first and second entities are members of the third one. The model requires a membership relationship to define the identifiers of the related entity and collection.

3.3.1.3 Namespaces of Core and Extended Structures

The concepts mentioned above are expressed, as will be shown in the examples, using qualified names that belongs to namespaces. Core concepts adopted from the PROV-DM are expressed using the PROV namespace. In addition, the thesis introduces a new namespace, named *interpret*, to accommodate new qualified names for the presented by the model.

Type	Core concept
prov:Person	Agent
prov:SoftwareAgent	Agent
prov:Collection	Entity
interpret:add	Activity, Derivation
interpret:delete	Activity, Derivation
interpret:assess	Activity, Derivation

Table 3.3: Types adopted by the model for different constructs. The model adopts types from the PROV namespace and introduces a new namespace, *interpret*, with new types.

The attribute `prov:type` provides an information about the type, or sub-type, of any construct. The model adopts some of the PROV-DM predefined types:

- `prov:Collection` to describe an entity that has members; and
- `prov:Person` and `prov:SoftwareAgent` to describe type of agents.

In addition, the model introduces the following to describe type of activities as well as derivation:

- `interpret:add`,
- `interpret:delete` and
- `interpret:assess`.

The first two are essentials, while the latter is one example of annotation type activity (further types of annotation can be introduced). Table 3.3 summarises the previously mentioned types. For entities, types are domain-specific. The thesis introduces such types when discussing the implementation on subsurface data. The attribute `prov:value` provides a value for an entity. Such attribute can be utilised by entities of type `interpret:assessment` for their assessment value.

3.3.2 Illustration of The Model by Examples

In this section, the concepts of the proposed model presented above are put into examples to illustrate its (the model's) application. According to PROV common practice, the illustrations are not intended to include all the details of the model but to show some provenance descriptions.

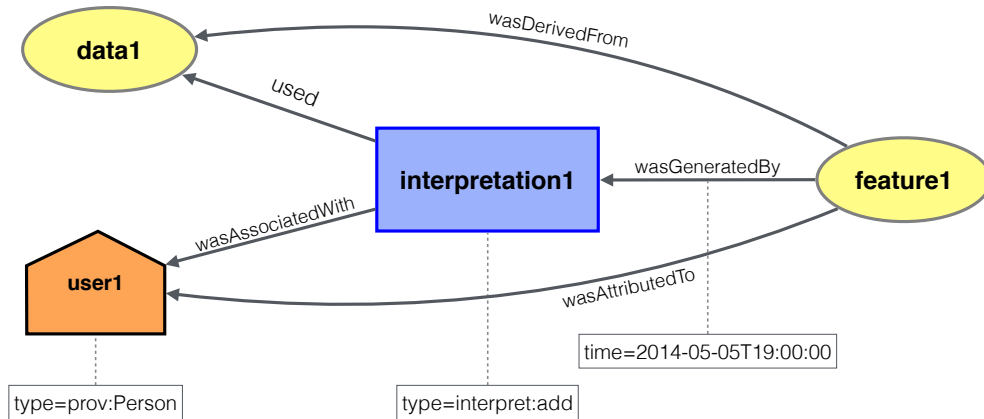


Figure 3.3: A graphical illustration of a basic interpretation case.

3.3.2.1 Example: Basic Case

A simple case is when a software or a user makes some interpretations on raw data to extract a feature. In Figure 3.3, a user (agent) made an interpretation to a data set which generated a feature at a specific time. The type of data and interpreted feature is domain specific; thus they were omitted from the illustration. The model's instances of this case can be expressed as follows, using PROV-N.

```
agent (user1)
entity (data1)
activity (interpretation1, interpret:add)
wasAssociatedWith (interpretation1, user1)
used (interpretation1, data1)

entity (feature1)
wasGeneratedBy (feature1, interpretation1, 2014-05-05T19:00:00)
wasDerivedFrom (feature1, data1)
wasAttributedTo (feature1, user1)
```

3.3.2.2 Example: Multiple Interpretations

Another, more realistic, case is when multiple interpreters contribute to the interpretations of a feature. In this example, illustrated in Figure 3.4, a software (agent) made an interpretation to a dataset which generated a feature (entity) at a specific time; the role of a user who sets parameters for the software are omitted for simplicity. At a later time, made an additional interpretation to be added the original interpretation by the software. The user's interpretation plus the software's interpretation form a view

of next version of the feature. Derivation of type `interpret:add` are expressed on the graph as *wasAddedTo*. The model's instances of this case can be expressed as follows, using PROV-N.

```
agent (software1)
entity (data1)
activity (interpretation1, interpret:add)
wasAssociatedWith (interpretation1, software1)
used (interpretation1, data1)

entity (feature1.1)
wasGeneratedBy (feature1.1, interpretation1, 2014-01-01T10:00:00)
wasDerivedFrom (feature1.1, data1)
wasAttributedTo (feature1.1, software1)

agent (user1)
activity (interpretation2, interpret:add)
wasAssociatedWith (interpretation2, user1)
used (interpretation2, data1)

entity (addition1)
wasGeneratedBy (addition1, interpretation2, 2014-01-02T14:00:00)
wasDerivedFrom (addition1, feature1.1, type=interpret:add)
wasAttributedTo (addition1, user1)

entity (feature1.2, type=prov:Collection)
wasDerivedFrom (feature1.2, data1)
hadMember (feature1.2, feature1.1)
hadMember (feature1.2, addition1)
```

3.3.2.3 Example: Assessment of Interpretation

A third case is when a second user assess the interpretation resulted from, for example, the previous case. An assessment is one type of annotation that the model can capture. The case is illustrated in Figure 3.5; details from the previous case were omitted for simplicity of illustration. In this case, a second user (agent) made an assessment of a previous interpretation, giving an assessment value of 70. The interpretation of the assessment value is defined according to the implementation. Derivation of type `interpret:assess` are expressed on the graph as *wasAssessmentTo*. The assessment part of the model of this case can be expressed as follows, using PROV-N.

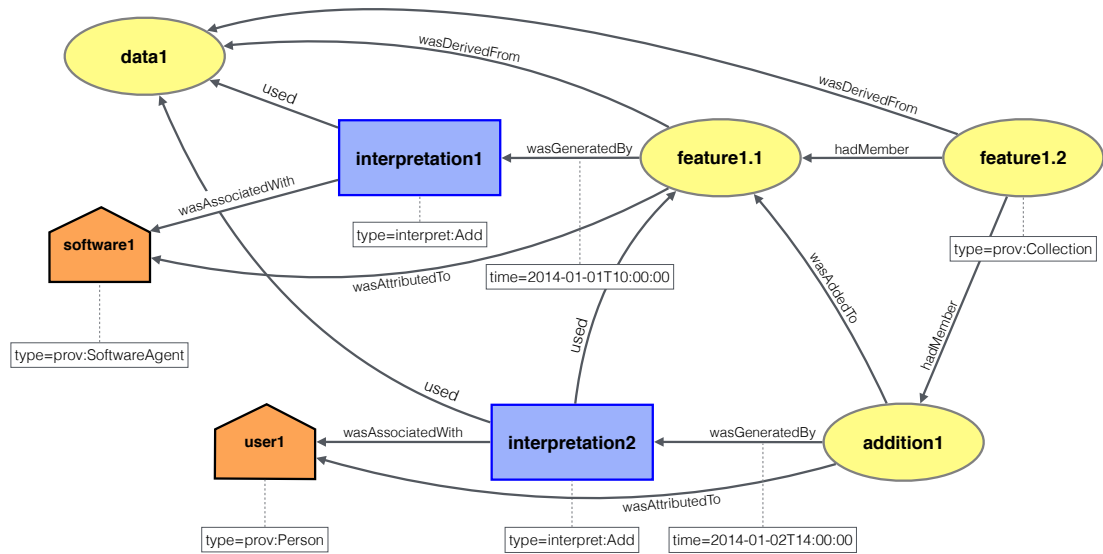


Figure 3.4: A graphical illustration of a multi-interpretation case.

```

agent (user2)
entity (data1)
entity (feature1.2)
activity (assess1, interpret:assess)
wasAssociatedWith (assess1, user2)
used (assess1, data1)
used (assess1, feature1.2)

entity (assessment1, value=70)
wasGeneratedBy (assessment1, assess1, 2014-01-10T14:00:00)
wasDerivedFrom (assessment1, data1)
wasDerivedFrom (assessment1, feature1.2, interpret:assess)
wasAttributedTo (assessment1, user2)

```

3.4 Implementation Consideration

This section provides high level considerations for the implementation of the proposed model. It addresses the granularity and access of data.

3.4.1 Fine-Grained Data Granularity

Data granularity defines the size of the smallest field which can be individually referenced and thus accessed. Finer granularity provides greater flexibility in accessing

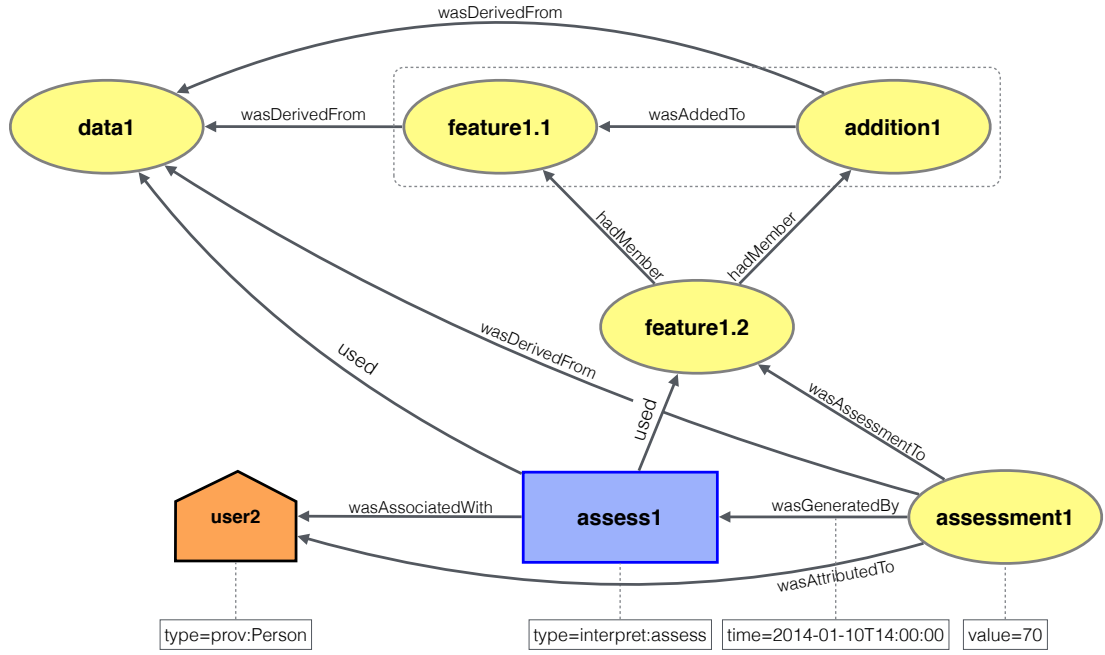


Figure 3.5: A graphical illustration of an assessment of an interpretation case. Details related to agents and activities of the features are omitted for simplicity of illustration, as the focus of this case is on assessment.

only the desired fields. However, this comes at the cost of increasing data size.

For the purpose of data visualization only, accessing a desired dataset as a single object (or file) is feasible. However, it becomes challenging when incorporating interpretation into visualization. An interpretation might be created by multiple users. Some users might slightly or greatly modify previous interpretations. Treating each interpretation version as an independent object creates duplication due to the similarities of these versions. Combining, comparing or reusing these multi-user versions becomes challenging in a coarse-grained model. For example, this would hide the type and areas of amendments performed on a previously interpreted object. Figure 3.6 shows a conceptual illustration of the difference between discrete versions (coarse-grained) of interpretations and relational fine-grained interpretations. In the seismic imaging domain, interpretation applications like Petrel [76] and DecisionSpace Geophysics (DSG) [78], treat and store interpretations as coarse-grained objects. This means that changes made onto an interpretation cannot be identified. The PROV model, as well as the model proposed in this thesis, allows the application to address the granularity level.

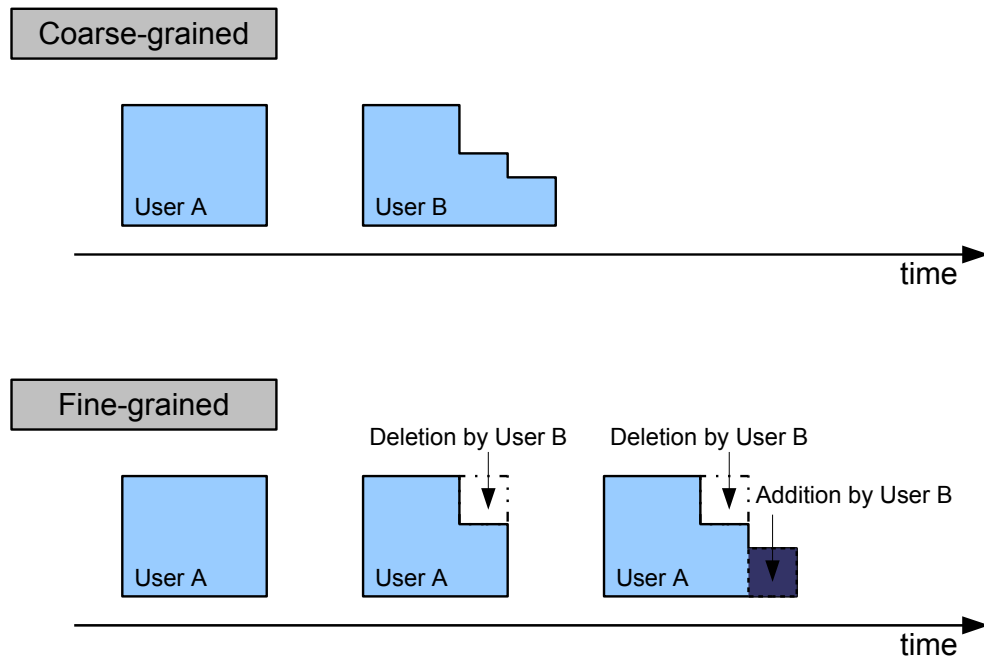


Figure 3.6: These are conceptual diagrams showing a 2D object created by User A and modified by User B. The top diagram is a coarse-grained mode where the modification by User B produces a new object. The bottom diagram is a fine-grained mode where the modifications by User B (deletion and addition) are being recorded. Unlike the coarse-grained mode where the final version is an independent object, the final version of the fine-grained mode is a result of the original object (by User A) and the series of modifications (by User B).

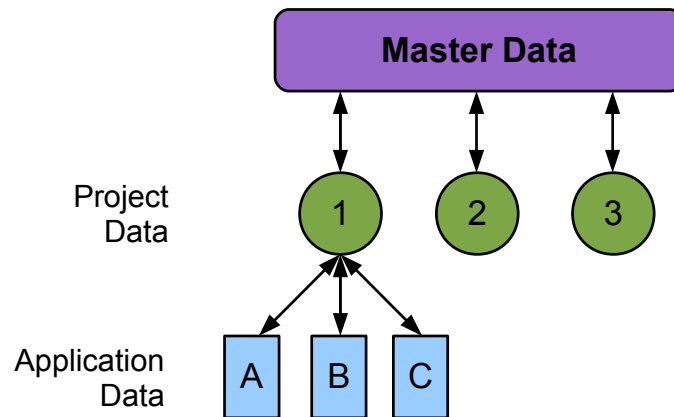


Figure 3.7: This is a conceptual diagram showing a conventional data architecture in the oil and gas industry.

3.4.2 Open-Standard, Single-View and Multi-User Access

In industries such as the oil and gas, 3D visualization, modeling and interpretation applications are often designed to work on a proprietary data format and preferably located locally (on the same machine as the application). This creates a challenge and data redundancy when moving data around applications, sharing data among users and reconciling the different results. Figure 3.7 illustrates a realistic example from the oil and gas industry, as perceived by working closely with the industry and advised by the domain expert collaborators consulted in this thesis. For each project, prerequisite data is initially prepared and copied from a master (approved) repository. It is often the case that multiple applications are used for a single project with each application dedicated to perform one or more tasks of the workflow. For example, one corporation may prefer to use Application A for data interpretation, and Application B for modeling and Application C for simulation. Using built-in or stand-alone tools, datasets are transferred from the *project data* repository into an *application data* repository or between applications.

The author of this thesis perceives that the reason for maintaining the current architecture (Figure 3.7) over time, in the presence of the challenges mentioned, is the intensive focus by the industry over the past two decades on developing advanced applications (addressing challenges in e.g. rendering, large data handling and feature extraction) while partially neglecting data management. The implementation of the model should, therefore, offer an open-standard data access to allow multiple interactive interfaces to be integrated. Also, multiple users should be able to access a single view of the data for immediate sharing and to eliminate duplication of datasets.

3.5 Chapter Summary

The chapter proposed a provenance-enabled interpretation pipeline, addressing the second objective of this thesis as presented in Section 1.4. The interpretation pipeline runs in the reverse direction to the visualization pipeline. It brings multi-user visual interpretations into a consistent provenance model and tags them onto the raw data being interpreted. These interpretations are user-defined regions on the visual scene. They are reverse mapped and load to the proposed provenance model, which links them to their related data. This model adopts and extends the W3C PROV data model, producing a provenance model for visual interpretations. Adopted classes (*entity*, *agent* and *activity*) and relationships (e.g. *generation*, *usage* and *derivation*) for the purpose of recording user interpretations were explained. Also, the *interpret* namespace was introduced to accommodate interpretation related activities within the model: *addition*, *deletion* and *assessment*. The use of this model was illustrated by examples of a single-user interpretation, a multi-interpreter contribution and an assessment of interpretation cases. The provenance model of each example case was expressed using the PROV notation (PROV-N) and graphically illustrated using the PROV graph layout convention. The chapter ended with some considerations for the implementation of the proposed pipeline and model. It discussed the granularity of the model, leaning towards fine-grained granularity for user interpretation to record area of changes by multi-users. It then discussed the direction towards a central open-standard implementation of the data model, to be multi-user accessible. This is to address the challenge of accessing and sharing data among users in industries where siloed applications are dominant.

Chapter 4

Use of Visual Interpretation in Geoscience

4.1 Chapter Introduction

Worldwide demand for oil and gas as an energy source is continuously increasing and therefore there is a continuous demand for advanced technologies in this field. Today's technology is not capable of both discovering¹ all available natural resources and recovering² those discovered resources [79]. The oil and gas industry, with the aid of technology, is continuously challenged to discover more resources with improved recovery. According to Saggaf [79], 3D seismic imaging has been, and will continue to be, one of the most influential technologies in the discovery of oil and gas. Challenges faced in regards to the visualization of seismic data include its noisy nature and the continual increase in the size of datasets [9].

In abstract, a window of the oil and gas upstream workflow is as follows (some steps were omitted for simplicity); see Figure 4.1. It starts with (1) *seismic data acquisition and processing* to obtain a subsurface image. Then (2) *seismic interpretation* takes place, where scientists visualize the subsurface images and attempt to understand it using their knowledge and expertise. In this step, scientists extract geological structures (features) such as horizons and faults. After that, (3) *3D geological modeling* is performed. A geological model is a representation of subsurface properties of physical quantities. The interpreted geological structures, from the previous stage, along with

¹*Discovery* is the process of exploring a new field of oil or gas.

²*Recovery* is how much can be extracted from the discovered field. Recovered resources are known as “reserves”.



Figure 4.1: The figure illustrates major steps within the oil and gas upstream workflow. This thesis is focused on the results of seismic interpretation.

ground-truth data from wells are used to generate a geological model. Finally (for the purpose of this context), (4) *simulation* is performed, where drilling scenarios are simulated to find out the most productive one. This thesis is concerned with the results of the second stage of the previous explanation, which is seismic interpretation. The other stages are out of the scope of this thesis.

This chapter generally provides a background information about the field at which this thesis is applied. First, it starts with a definition of seismic data explaining how it is acquired (Section 4.2), which reflects the first stage from the workflow previously explained. The section also includes information about the industrial format of seismic imaging data, known as SEG-Y. Having acquired and processed seismic imaging data, Section 4.3 corresponds to the second stage of the workflow. It starts by explaining what geological features are and their importance to the oil and gas discovery. Then, it discusses how these features are interpreted by geoscientists and the uncertainty in such process. Section 4.4 gives a background about the visualization of seismic data. It also discusses an example of industrial application. Section 4.5 discusses the current status of seismic interpretation management. The chapter then presents two sections as a motivation for this research. Section 4.6 briefly discusses time-lapse (4D) seismic data which is becoming more desired by the industry to enhance oil recovery; this demands more seismic data. Section 4.7 presents and analyses a survey for a motivation purpose conducted by the author of this thesis with staff from the oil and gas industry as well as geoscientists from academia.

4.2 Definition and Acquisition of Seismic Data

The word seismic in English is an adjective meaning “pertaining to, of the nature of, or caused by an earthquake or vibration of the earth, whether due to natural or artificial causes.”³ In the oil and gas industry, seismic data is obtained to represent the structure of subsurface layers of the earth’s interior. Exploration for oil or gas starts

³<http://www.dictionary.com>

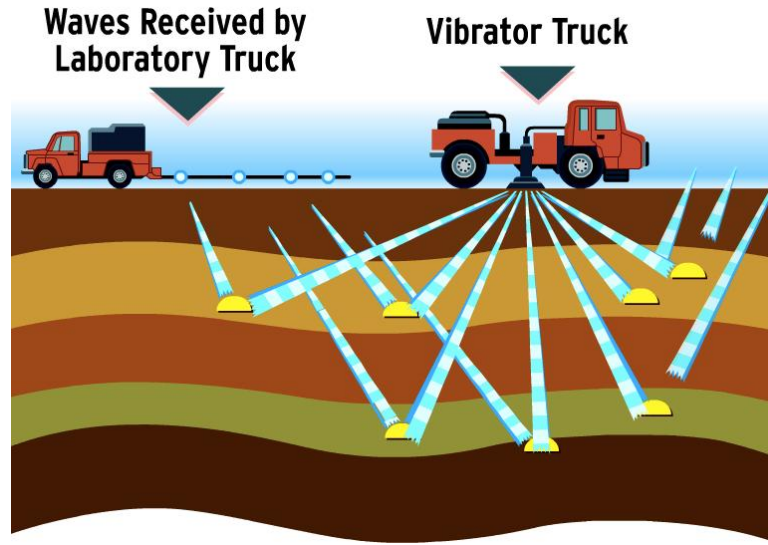


Figure 4.2: Seismic acquisition—waves are generated penetrating subsurface geological layers; they then reflect back and their amplitudes and received time are recorded. Illustration adapted from: <http://www.sjgs.com/exploration.html>—accessed 1 June 2010.

with analysing such data.

To acquire seismic data, acoustic waves are artificially generated on the earth's surface penetrating subsurface geological layers. Due to the variation of each layer's materials, these waves are reflected back to the surface and their amplitudes and received time are recorded via receivers [9, 80], as illustrated in Figure 4.2. This data is then processed to generate a 2D or 3D image illustrating the subsurface layers; see Figure 4.3 as an example of a 2D seismic image.

A 2D or 3D seismic dataset is composed of vertical *traces*, each consisting of the sampled amplitudes; refer to left hand side of Figure 4.3. In a 3D seismic volume (also called a *seismic cube*), traces are regularly distributed over a 2D area. Seismic traces are located on points $\{\mathbf{p}^\alpha : \alpha = 1, 2, 3, \dots\}$ [81], which can be viewed as the nodes of a 2D rectilinear⁴ grid. At point $\{\mathbf{p}^\alpha\}$, a trace of regularly sampled amplitudes can be expressed as $F^\alpha(T)$ where T is a time value along the vertical time axis. The set of traces on one line that is parallel to the direction of data acquisition is called *in-line*, while the set of traces in a line that is perpendicular to the direction of data acquisition is called *cross-line*. Figure 4.4 illustrates the different ways to slice a seismic cube; these slices can then be directly rendered using the texture mapping technique explained in Section D.2. The data is graphically colour-coded in respect to their seismic amplitude

⁴The spacings in a *rectilinear* grid along the axes are arbitrary [82].

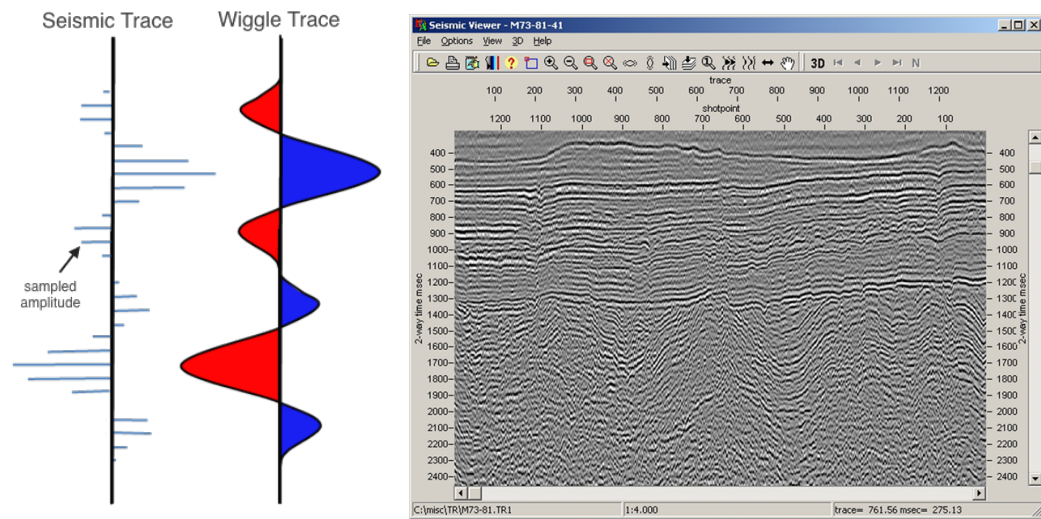


Figure 4.3: Left hand side figure illustrates a single (1D) seismic trace and its oscillating-line representation, wiggle trace; illustration partially adapted from E. Bianco (2011). Right hand side figure shows a seismic viewer displaying a 2D seismic image; image adapted from: <http://sight.cjb.net>—accessed 1 June 2010.

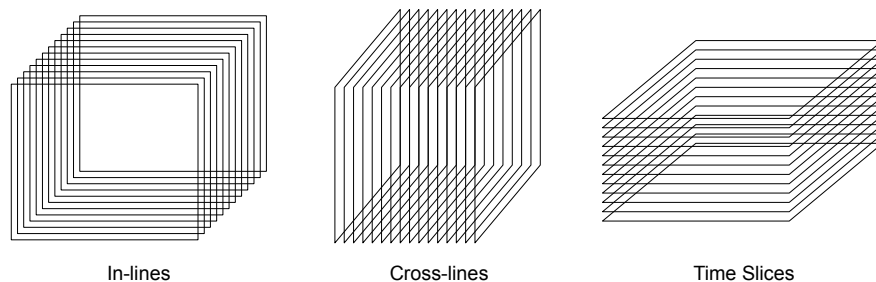


Figure 4.4: This figure illustrates three different ways of slicing a seismic cube [18].

values.

A seismic image is then interpreted by geoscientists to extract geological features. Two important features are *horizons* and *faults*. The next section presents the meaning of geological features in seismic data and how they are interpreted. The interpretation of seismic data could lead to a potential hydrocarbon trap: a source of oil or gas.

4.2.1 SEG-Y Format

SEG-Y is a standard file format developed and maintained by the technical standards committee of the Society of Exploration Geophysicists (SEG)⁵ for storing geophysical data, commonly seismic data. Its specification was first published in 1975 as *revision 0*

⁵<http://www.seg.org/>

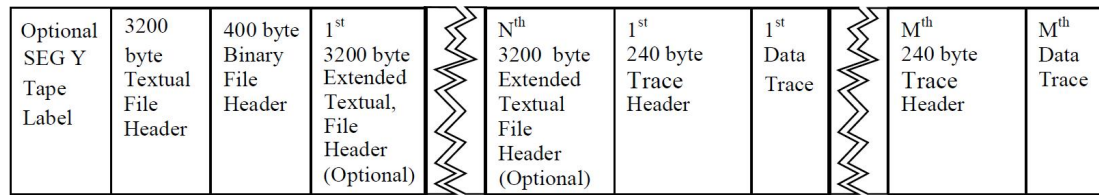
(or *rev 0*). Since then, there have been many advancements in the oil and gas industry including the introduction of 3D seismic acquisition and the introduction of high speed and capacity storage; thus, SEG-Y has been revised. Its latest update was published in May 2002 as *rev 1* [83].

SEG-Y was originally designed to be recorded on 9-track tapes by IBM. The new revision, however, allows recording of SEG-Y on any medium that supports multiple records. Disks, such as CD-ROM and hard disks, have been commonly used to record SEG-Y files. Recording and reading SEG-Y data must follow the SEG-Y file structure. As illustrated in Figure 4.5, SEG-Y file structure consists of the follow components [83, 84].

1. A 3200-byte *Textual File Header* consisting of 40 lines of human readable textual information such as the company name, area, crew number and date. In the previous revision of SEG-Y, the textual file headers could only be written in EBCDIC⁶ format. As of the recent revision, ASCII may be used, in addition to EBCDIC, to write the textual file headers of SEG-Y.
2. A 400-byte *Binary File Header* containing important details about the whole SEG-Y file that determine the reading process of the trace data. These particularly include the sampling interval, trace length and format code. The format code determines the format type of the recorded samples. Supported format types include 4-byte, 2-byte and 1-byte two's complement integers as well as a 4-byte IEEE floating point.
3. Zero or more 3200-byte *Extended Textual File Headers*, similar to the Textual File Header, to be used for extra information. The Binary File Header determines the number of Extended Textual File Headers in the SEG-Y file.
4. Multiple trace records, each consists of the following:
 - (a) A 240-byte binary *Trace Header* containing attributes for a single trace record. In particular, it includes the trace sequence number within its line.
 - (b) *Trace Data* containing multiple sample values of the format recorded in the Binary File Header.

The values in the Binary File Header and Trace Header are either two-byte or four-byte two's complement integers. Also, all binary values in SEG-Y—these include

⁶Extended Binary Coded Decimal Interchange Code (EBCDIC) [85]

Figure 4.5: An illustration of the **SEG-Y** file structure [83].

values in Binary File Header, Trace Header and trace data—are defined as *big-endian* byte ordering. This means that the most significant byte is written first, i.e. closer to the beginning of the file. However, integers and floating-point types defined in current programming languages (e.g. C, C++ and Java) are defined as *little-endian* byte ordering—least significant byte is written first. For this reason, a conversion from big-endian to little-endian is required for each sample value when reading seismic data using modern machines.

The trace data of SEG-Y files are recorded in the order of in-lines. Which means that traces of line 1 come first, then traces of line 2 come next, and so on till the last line. For this reason, extracting a cross-line section is time consuming as it requires sequential access to navigate through every line to extract a single trace from each. This is one of the reasons why applications like *Petrel* offer a conversion from and to an internal format applying techniques for fast access.

4.3 Geological Features and Seismic Interpretations

4.3.1 Geological Features in Seismic Data

The interpretation of geological features is based on the interpretation of changes in sampled amplitudes, as a change in amplitude reflects a change in geology [56, 86]. Geological features can be categorized into *structural* and *stratigraphic*⁷. A feature in the geology could be a reservoir trap which is capable of retaining hydrocarbons. Thus, reservoir traps follow the same category of geological features, i.e. *structural* and *stratigraphic* [87].

Structural features include *domes*, *anticlines* and most commonly *faults* as illustrated in Figure 4.6. Faults represent fractures, and thus discontinuities, in geological

⁷**Stratigraphic:** an adjective meaning pertaining to the arrangement of *strata*. *Strata* is the plural form of a *stratum* which, in geology, means “a single bed of sedimentary rock, generally consisting of one kind of matter representing continuous deposition.” (<http://www.dictionary.com>)

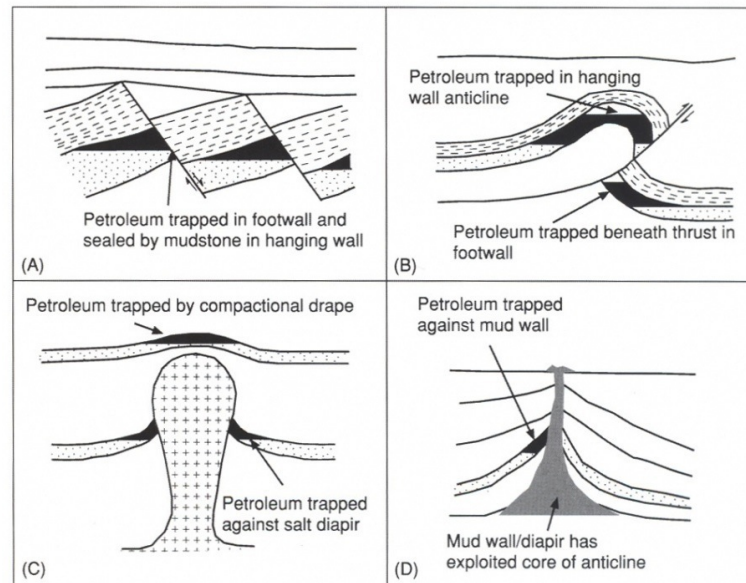


Figure 4.6: Diagrams of some structural traps: (a) tilted fault, (b) rollover anticline with fault seal, (c) salt dome (diapir) and (d) mud wall. Diagram adapted from Ikelle and Amundsen [89].

layers as a result of their movement over millions of years. Faults are important in seismic interpretations as they can either be sealing and thus act as hydrocarbon traps or provide a leakage path for hydrocarbons to flow along. Interpreting faults is a time consuming and a challenging task for geoscientists. Stratigraphic features, on the other hand, are formed due to the deposition of rock layers (strata) over each other [56]. For example, *channels* are stratigraphic features formed by flowing rivers. *Horizons* may also refer to stratigraphic units. They represent horizontal surfaces at the interface between two geological layers; refer to Figure 1.1 which shows horizons in a 2D seismic section marked in yellow. Structural traps, in comparison to stratigraphic traps, are easy to detect. Most discovered reservoirs are of this type. In contrast, stratigraphic traps are involved in most undiscovered reservoirs [88, 87].

To extract a horizon for example, algorithms are available to perform what is called *auto-picking* [18]. This is a process to extract a set of points located on a geological feature that is passing a seed point which was manually selected by a user. To help users to verify the result of the auto-picking process, or their manual interpretation, parameters known as *seismic attributes* are calculated which give a different visual result aiming to highlight features. Randen et al. [87] presents a list of 3D seismic attributes.

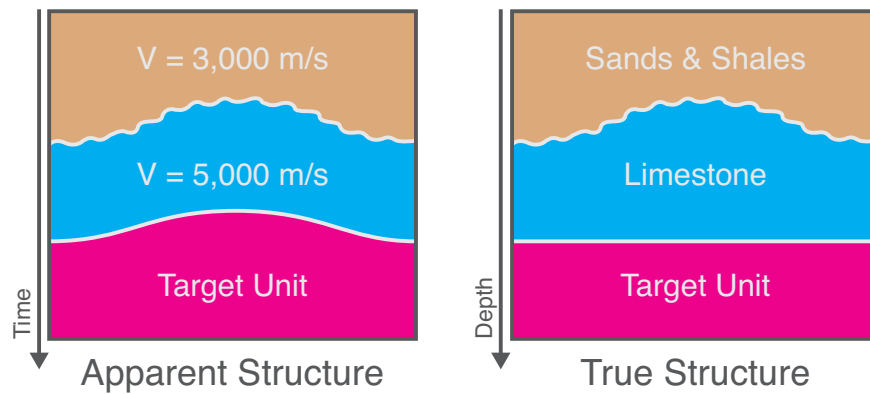


Figure 4.7: This figure conceptually illustrates the different representations of subsurface structure in time domain (left) and depth domain (right), after conversion. Shale and limestone are subtypes of sedimentary rocks. Illustration adapted from Höllt [90] and Etris [91].

4.3.2 Seismic Interpretation

As described by the thesis of Höllt [90], seismic interpretation is about extracting geological features such as horizons and faults. The extracted features are used to build an earth model (a model of the subsurface structure). Along with seismic data, geoscientists use additional information from physical drilling (well data) to assure their interpretations. Well data is in depth domain while seismic data is in time domain. Therefore, geoscientists can only correlate their seismic interpretation with well data after depth conversion, which computes the spatial depth for seismic structure. Figure 4.7 illustrates a subsurface structure in both domains, time and depth.

Seismic interpretation is neither a straightforward nor a simple process. This is due to reasons including its ambiguity, complexity, density and noise. Seismic data may therefore be misinterpreted, or different views of interpretations may result by different interpreters (geoscientists). The current practice of seismic interpretation requires manual inspection by geoscientists on in-line and cross-line slices prior to running an extraction method, and manual corrections on the same bases afterward. As illustrated in Figure 1.3 and presented in Section 1.3.1, it is very likely that different interpretations results from one visualization view of a seismic section. This uncertainty in interpretation is one of the major issues in subsurface exploration [92].

Horizons are often placed at the minimum or maximum seismic values (extrema events). When interpreting a geological horizon manually or automatically, similar extrema events are picked to form a surface within the 3D cube. Many methods for

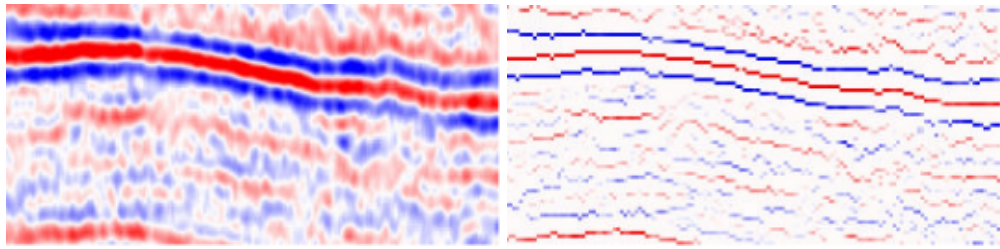


Figure 4.8: This figure shows a 2D vertical seismic section (left) and its corresponding sparse extrema representation (right), which highlights the minima and maxima seismic events. Illustration adapted from Borgos et al. [95].

automated horizon extraction from 3D seismic data are based on this notion; the following discusses some of these methods. In the work of Tu et al. [93], 2D events are first extracted as curves. These are then grouped resulting in extracted 3D surfaces. Using seismic attributes, seismic wave around an extrema event can be described to group horizon segments with similar local wave form [94]; thus classifying seismic reflectors. This is based on the assumption that seismic waves follow a similar characteristic laterally within a specific horizon. This addresses the challenge of tracking a horizon in cases like discontinuity of seismic signal. The result of the classification may then form a sparse cube⁸ containing only information about the minimum and maximum events [95, 96]; see Figure 4.8 for illustration. These are called *horizon patches* which act as primitives for geometry extraction of horizons. Faraklioti and Petrou [97], and later Blinov and Petrou [98], proposed an algorithm to connect related 3D horizons patches according to the relative position of the horizon. This is based on the assumption that connected patches do not cross. However, it is computationally expensive to run such algorithm on a complete seismic volume [90].

The above methods were examples that support the common bottom-up interpretation approach, which starts with detailed interpretations of the data. Alternatively, Patel et al. [99] proposed a top-down interpretation approach via providing 2D illustration of interpreted seismic data for easy communication among team members. This supports coarse interpretations to reach an preliminary understanding about the data prior to detailed interpretations, which is traditionally performed via sketching tools. Subsequently, Patel et al. [12] supported 3D illustration of seismic data.

⁸A sparse volume, or cube, is a volume which most of the data elements are zero

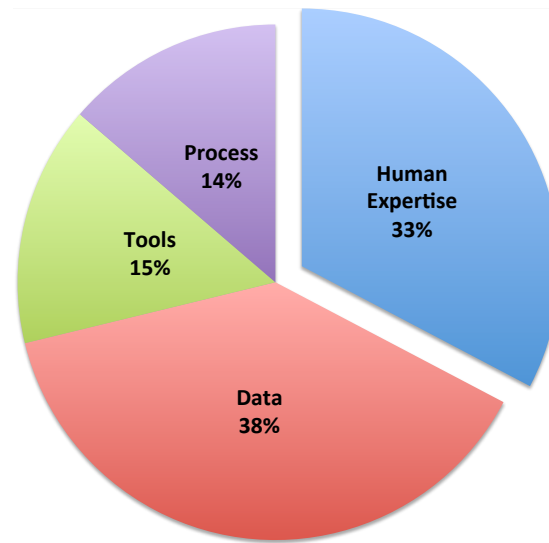


Figure 4.9: The chart illustrates the elements that contribute towards understanding the subsurface [100]. The concern of this thesis is the value of human expertise, which contributes by around 33%.

4.3.3 Uncertainty in Seismic Interpretation

A recent study [100] showed that *human expertise* contributed by 33% towards the subsurface understanding (i.e. interpretation); the other elements were *data availability and quality* which contributed by 38%, *tools* which contributed by 15% and *process* which contributed by 14% (see Figure 4.9). This major role of *human expertise* in the interpretation process, along with the bias nature of human opinions, raises uncertainty in seismic interpretation.

Bond et al. [101] attempted to quantify the conceptual uncertainty in seismic interpretation, which measured the range of concepts that different interpreters apply to a single dataset. A synthetic seismic section from a 2-D geological model was given to 412 interpreters; thus the “correct” interpretation was known to the experiment designers. Factors such as background knowledge and length of experience of the interpreters were the cause of diversity on the interpretations they produced. Part of the study results showed that only 23% of the participants correctly interpreted the major faults in the given 2D seismic section.

Lidal et al. [20] addressed the issue of different interpretations (Figure 1.3) of a single dataset by proposing a sketching-based storytelling graphics system. Geoscientists can sketch their reasoning behind the different interpretations in a storytelling form for the purpose of discussion and illustration among peer-experts and decision

makers.

4.4 Volume Visualization for Seismic Data

Seismic imaging data is often rendered using texture mapping [80] (see Section D.2 for general background), which is a direct volume rendering technique. *Texture-mapped slices* are implemented by selecting a plane from the volume and then a colour (texture image) is assigned for each vertex according to the intersected data value; this value could be a seismic amplitude or any seismic attribute value. Thus in a 3D seismic volume, each voxel corresponds to a sample value. On rendering, values are scaled to be displayed as a histogram which represents the distribution of these values, as described by Kidd [102]. Such algorithm is adopted in the prototype application presented by this thesis.

A more advanced rendering technique for seismic visualization was used by Castanie et al. [9, 103]. They were first to use pre-integrated volume rendering for seismic visualization. The pre-integration is performed in the transfer function (i.e. colour mapping) as a pre-processing step and fetched at rendering time. This technique renders the volume as slabs that were pre-computed, instead of slices as in the conventional textured-slicing technique. Since slabs are precomputed, high quality rendering is achieved with less computational resources. In comparison, classical textured-slicing volume rendering requires high sampling rate for high quality results, which is computationally expensive and thus decreases performance.

Others have addressed the issue of gradients in classical volume rendering for seismic visualization. Strong gradients usually corresponds to boundaries. In the case of seismic imaging, boundaries are located at extrema events which usually have small gradients and are of variant values along a horizon due to lateral geological variations. Silva et al. [104] proposed a gradient of instantaneous phase attribute which computes a local surface normal for each seismic sample. Patel et al. [13] proposed gradient-free shading for seismic volume rendering and implicit surface representation for horizon surface rendering. This integrates horizon rendering with volume rendering.

Three-dimensional volumes are stored in one dimension and therefore not accessed in the order they are stored. This creates a bottleneck, and thus decreases performance, when accessing large seismic datasets that do not fit inside a machine's memory, which is an *out-of-core visualization* issue. *Octreemizer* by Plate et al. [8] addressed this issue by restructuring the 3D seismic data into multi-resolution bricks.

4.4.1 Seismic Imaging Applications

A variety of seismic applications are available in the market for the oil and gas industry. Some of them are specialised particularly in seismic interpretation and visualization such as *GeoProbe* [105] which is owned and maintained by Halliburton⁹. Other available applications offer a complete workflow, part of which is seismic interpretation and visualization, such as *Petrel* which is owned and maintained by Schlumberger¹⁰. These applications offer a large number of features. The following text covers *Petrel* as an example of a widely used application in the seismic interpretation and visualization domain. This illustrates the most current market-available solution in this domain, focusing on how it utilises visualization, rendering and data management techniques. However, as is the case with most vendor applications, the underlying mechanisms of such techniques remain classified in most cases.

4.4.1.1 Petrel

Petrel is a “seismic-to-simulation” Windows PC software owned by Schlumberger and widely used by the oil and gas industry [76]. It is designed to be a single application supporting a complete workflow which starts from seismic interpretation and includes tasks such as well correlation, reservoir modelling, submitting jobs for simulation and visualizing simulation results.

Some basic features about *Petrel* [106, 107, 108] are as follows. Both 2D and 3D displays are offered for objects. It handles different types of data, including the following.

- *original seismic data*— read from a SEG-Y file or *Petrel*’s internal format, ZGY file.
- *generated attributes*
- *interpreted data*—points picked manually or automatically selected to represent geological features such as horizons and faults.

For seismic interpretation, users first manually *pick* some points, usually on a 2D view, to select geological features. To complete the interpretation, *Petrel*, as well as many other similar solutions, offers *auto-picking* to automatically pick the rest of the points that form the desired feature based on those initially picked manually. The

⁹Halliburton is an oilfield services corporation, <http://www.halliburton.com/>.

¹⁰Schlumberger is an oilfield services corporation, <http://www.slb.com/>.

more points users pick manually, the more accurate is the result they obtain from the auto-picking.

Based on the object to be rendered in Petrel, there are different renderers as follows.

- *Seismic volume renderer*—allows an interactive visualization of multiple seismic volumes in 3D with a transparency option to identify areas of interest. A volume is treated as a bulk of data.
- *Seismic slice renderer*—renders a slice of seismic data, i.e. a seismic line, in a 2D or 3D window display using the textured-slice rendering technique.
- *Grid renderer*—renders interpreted picked points as a polygon.

4.5 Data Management for Seismic Interpretations

According to the domain expert collaborators consulted in this thesis, managing seismic interpretation is an industrial challenge. Hawtin and Lecore [100] published a study highlighting the value of data management in the oil and gas industry. The study showed that executives in this industry often exclude their perception of *data management* to raw and unprocessed data. However, “best data managers” believe in the importance of tracking and managing subsurface interpretations, the study added.

Interpretations of users’ current activities are often maintained by their interpreting applications. They are stored as binary files, of a proprietary format, under their *project data*, as is the case in *Petrel* [76] and *DecisionSpace Geophysics (DSG)* [78], two commonly used applications by the oil and gas industry. The application *project* bounds between a seismic survey and its interpretations. It also maintains metadata about the interpretation, such as who created it and when it was created. In the case of *Petrel*, which is a single-user application, project data are stored in a file system. To address data management and sharing at the application level, Schlumberger introduced *Studio E&P* [109] as an extension to *Petrel* allowing a multi-user collaboration via a central data repository. Users import a dataset to visualize and interpret from *Studio E&P* to their local *Petrel*, then commit their changes back to the central *Studio E&P*. A master project is also used to share referenced data among users. In the case of *DecisionSpace Geophysics (DSG)*, which is a multi-user application, it relies on its own OpenWorks¹¹ project data management system to allow instantaneous sharing of

¹¹<https://www.landmarksoftware.com/Pages/OpenWorksDatabaseApplication.aspx>

data. Despite its multi-user setup, it records the last modifier user name in the case of modifying an interpretation without identifying where it was precisely modified, treating interpretations as coarse-grained objects as discussed in Section 3.4.1. In either case, they (*Petrel* and *DSG*) remain application-closed solutions. Metadata of interpretations, including the linkage to the corresponding raw dataset (seismic survey), is lost if they are exported to be loaded into a different application for example. Also, they do not allow assessment by users on regions of previous interpretations. Thus it becomes challenging to trace a previously interpreted object.

The current practice in the industry relies on file name conventions to record metadata for searching and identifying interpreted objects that were exported from the software application. File names of interpretations usually include the last interpreter's initials. Information about the other interpreters who participated, uncertainty and any assessment or quality check performed during the interpretation process is lost. Chelmiss et al. [22] proposed a semantic web based approach to annotate datasets with missing metadata from filenames.

4.6 Demanding Oil and Gas Workflow—The Present and Future

A decade ago, the common hydrocarbon exploration workflow (Figure 4.1) obtains seismic data once for a region to be studied. Re-shooting seismic was only performed in certain cases such as receiving a bad quality data in the first place. Nowadays, 4D time laps of seismic data has received a greater interest. It is a series of snapshot of changes to the subsurface taken over short periods of days to few years [110]. In 4D seismic data, geoscientists attempts to determine fluid movement in the reservoir due to production. This data helps in lowering uncertainty of engineering simulations to economically optimize oil recovery; see Figure 4.10. In other words, 4D seismic data supports *enhanced oil recovery* (EOR) methods which aim to increase the amount of fluid that can be extracted from a reservoir.

Interpretation of 4D seismic data is different from the conventional 3D seismic data interpretation; the latter is concerned on extracting geological features as discussed earlier in this thesis. Interpretation of time-lapse seismic data produces an understanding of subsurface fluid migration and changes on reservoir attributes such as saturation and pore pressure; Stammeijer and Hatchell [111] presented standards in 4D seismic interpretation. Despite such difference, the increasing demand in 4D seismic data is of a

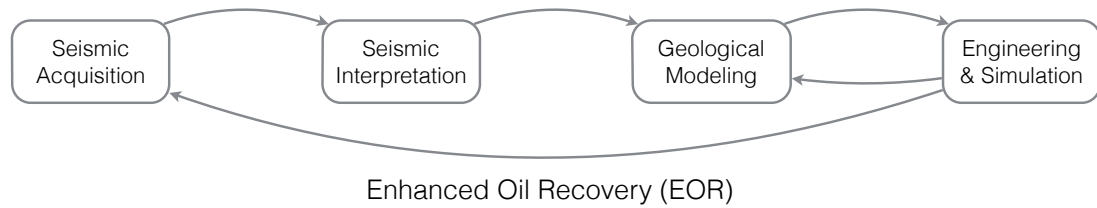


Figure 4.10: This figure is similar to Figure 4.1 with enhanced oil recovery (EOR) consideration. Snapshot of seismic data (4D time laps) after producing feeds the simulation to economically optimize oil recovery.

motivation to the research of this thesis. More interpretations are created when using time-lapse images which create a need of proper tracking and linking.

4.7 Motivation Survey

4.7.1 Survey Purpose and Population Frame

A survey was conducted to support the motivation behind this research. The purpose of the survey was to understand the following:

1. the importance of a collaborative environment and data provenance for seismic interpretation,
2. the differences between a seismic interpretation environment in universities and in the industry,
3. the challenges IT staff encountered in satisfying the need of seismic visualization and interpretation in an efficient data management manner.

To address the above purpose, the targeted survey population (users) was limited to the following criteria.

1. Users should have worked on or supported a seismic interpretation and visualization application
2. The sample should cover geoscientists and IT staff who meet the above condition
3. The sample of geoscientists should cover two subsets: (1) staff in academia or postgraduates; and (2) staff in the oil and gas industry.

4.7.2 Data Collection Method

Having defined the survey population frame, users (participants) were approached through direct connections. Thus the survey was based on a non-strictly random sampling, also known as a *non-probabilistic* [112]; this sampling method is more acceptable to the computer science community than the social science and statistics communities. The oil and gas sector is highly sensitive and operated by busy personnel who often are reluctant in releasing information. Therefore, it was challenging to secure participants from this sector for the above purpose, and thus a non-probabilistic sampling was adopted. This also explains the relatively small number of participants as explained next.

Referring to the survey population, a stratification method was applied such that the survey covered two user groups based on their discipline: geoscience and IT backgrounds. Each group were given a separate questionnaire. Survey structured included both close-ended questions and open-ended questions; the latter were added to allow users to further elaborate on corresponding close-ended questions. The survey for participants from academia was paper based as they were reachable by distance. On the other hand, it was an invitation-only online based survey for respondents from the industry due to their global location.

Around half of the participants from the industry came from a single oil and gas producing company. In attempt to avoid potential biases, individuals from different companies operating in different countries were approached to participate in the survey. All participant from the industry had more than 4 years of experience. Also, geoscientists from the industry covered a cross section of different sub-background: geology, geophysics and petroleum engineering. In addition, it was attempted to avoid the use of negative words in the questions which may lead to biased responses.

4.7.3 Survey Sample

A total of 34 individuals participated in this survey, who reasonably represent cross-section of seismic applications users. The participants are classified as follows:

- 28 senior staff from oil and gas companies:
 - 8 geoscientists
 - 20 information technology (IT) staff who explicitly support and maintain subsurface data, hardware and software infrastructures

- 6 geoscientists from a university:
 - 5 postgraduate students
 - 1 senior staff

A summary of the responses from the company staff who participated is included in Appendix B.

4.7.4 Response from Geoscientists

Responses of participants of geoscience background from academia and the industry are illustrated in Figure 4.11. *Collaboration* in this context refers to the ability of users to share the results of their interpretations and work together to produce such results. Thirteen out of fourteen (93%) of the participating geoscientists perceived that a collaborative visualization and interpretation environment is “important”. Most participants from the industry added that collaboration “raises productivity”. It was observed that the need for a collaboration was more obvious to geoscientists in the industry than those in universities where the work is almost always performed individually. The same ratio (93%) also perceived that, overall, it is challenging to collaborate on seismic interpretation using existing software. The geoscientists from the industry highlighted this challenge on sharing their interpretations with other teams; for example between the interpreters and engineers. In such an industry, each team often uses different software, and each software has its own proprietary internal format. All participating geoscientists perceived that data provenance, including history tracking, is important in seismic interpretation; most of the participants from the industry added that history tracking “raises efficiency”. Looking into the challenge of history tracking, university geoscientists did not find this challenging while the industry saw that it is “manageable for recent interpretations but a bit challenging for very old datasets.”

4.7.5 Response from Information Technology Personnel

The response from the IT staff (see Figure 4.12) highlighted the technical challenges and thus the area for improvement, which geoscientists might not perceive in the same way, since it falls within the role of IT staff. Sixteen out of twenty IT staff (80%) perceived that users cannot immediately access all subsurface data for visualization or interpretation in a highly available fashion; but data access needs some initial preparation. They share with the geoscientists the perception of challenging collaboration;

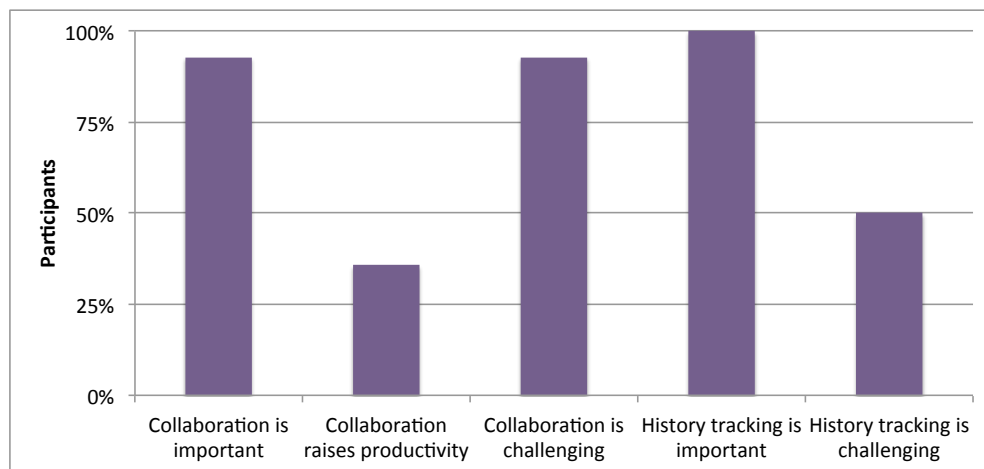


Figure 4.11: The chart illustrates what the 14 participants of geoscience background, from both academia and the industry, perceived on the importance and challenge of collaboration and history tracking of data. Interestingly, the importance of collaboration in raising productivity and the challenge of performing history tracking using existing tools were only perceived by geoscientists from the industry and not academia.

55% found that sharing users' results with other staff members is a time consuming task as it requires the export and import of the data to be shared. The participants suggested that to enhance the infrastructure, for a collaboration that maximizes productivity, the industry needs to introduce a "more integrated data environment", "use cloud-based technology", "standardize the workflow between all users" and "have shared databases among all disciplines (teams)." In addition, they evaluated the movement of data between applications as it negatively affects productivity; 75% of them emphasised this as a "high" effect which "needs a great attention". Seventeen out of twenty (85%) believed that centralizing subsurface datasets would improve data management, but this "has been challenging up to now." Some of their suggested reasons for not centralizing datasets were the proprietary internal format of multiple applications and the lack of a well integrated environment in the industry. The need for a "standard data repository" was highlighted.

4.7.6 Survey Analysis

The survey responses showed that the importance of collaboration in the interpretation of subsurface data and its challenge were perceived as highly important by geoscientists. It was noticed that individuals who did not perceived such importance were petroleum engineers. Their task within the oil and gas exploration workflow comes

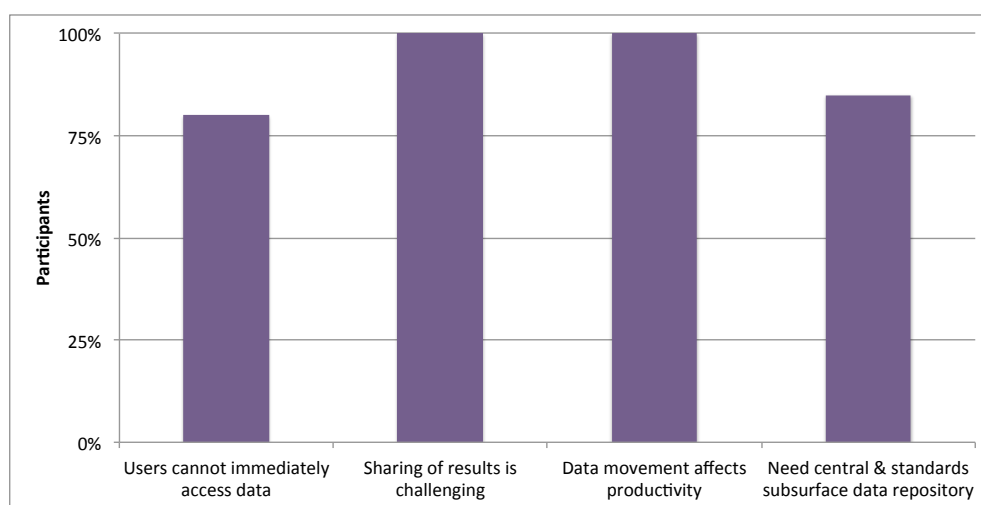


Figure 4.12: The chart illustrates what the 20 participants of information technology background, from the industry, perceived on (1) the ability of users accessing subsurface data, (2) challenge of sharing data among users, (3) impact of data movement among different application repositories and (4) the need of a central and standard subsurface data repository.

at a late stage when interpretations were performed and geological models were created. They expect the interpretations and models to be finalised and approved, paying less concern on how geologists came into agreement. Also, the workflow may restrict petroleum engineers from reconciling different views of interpretations. Therefore, the importance and challenge of collaboration were not perceived by them (petroleum engineers). For geologists and geophysicists, the challenge in collaboration is greater when it comes to sharing data between different applications as each has a proprietary internal format. In the oil and gas industry, it is very common for a single firm to use software from different vendors. Transferring data between applications is one of the main challenges in this industry. Metadata is usually lost when exporting data from one application into another.

The responses showed that all geoscientists agreed on the importance of tracking historical interpretations. However, the reason why this was not challenging to users from academia is perhaps due to the small dataset they have and small environment they operate in. In the industry, the data is very large and often goes back to many years. That is why it is more challenging to track historical interpretations in the industry.

IT staff in the oil and gas industry are responsible to support and maintain the IT infrastructure including the applications being used by the geoscientists. Part of

their role is to ensure proper transfer of data among the different applications. That is why all perceived that it is challenging to share the resulting data among the different teams. Their responsibilities also allowed them to perceive the effect of data movement on productivity due to data loss and need of man hours. This also creates the challenge of tracking historical interpretations. The response to the open-ended questions given to the participated IT staff all suggest the need for an integrated and shared data repository among the different functionalities within the workflow to maximize users' productivity. This is in contrast to the current application-centric fashion where data is maintained under the application repository. This explains the general agreement on the need of centralizing subsurface datasets to improve data management, which is, however, of a great challenge to the industry. However, to the interest of this thesis, tracking of visual interpretations is proposed to be implemented as a data-centric fashion as desired by the industry and as discussed in the coming sections.

4.8 Chapter Summary

Through this chapter, the thesis focused on the domain of its application, seismic imaging interpretation for the oil and gas industry, by presenting background information, related work and a motivation survey.

Seismic data represents the structure of subsurface layers of the earth's interior. It is conventionally stored in a standard file format, named SEG-Y. Seismic interpretation extracts geological features such as horizons and faults. The current practice of seismic interpretation requires manual inspection on the data, then running an extraction method and later manual corrections. Most of the horizon extraction methods are based on picking similar extrema events. Seismic data is of a high uncertainty. It is common that different interpretations are achieved by different interpreters examining the same region.

The data is often rendered using a texture mapping method. The chapter presented examples of related work to seismic visualization. Some work has looked at the issue of gradients in classical volume rendering for seismic visualization. Others have looked at the issue of accessing large seismic datasets, e.g. *Octreemizer*. It then briefly described a well-known commercial application for seismic imaging visualization and interpretation, *Petrel*.

Seismic interpretation management is of a great challenge to the industry. Current practice relies on the application level to maintain seismic interpretations. When they

are exported, the industry heavily relies on file naming conventions to tag interpretations with metadata (e.g. creator's initial, field name); other metadata is lost.

The chapter briefly discussed 4D time laps of seismic data, which has recently received a greater interest by the industry. This will produce more seismic data which need to be interpreted and thus managed and traced. This motivates the need of the research of this thesis. Finally, the chapter presented a survey for a motivation purpose. The participants indicated the importance of history tracking in seismic interpretation activities and its challenge with existing solutions.

Chapter 5

Implementation for Geoscience

5.1 Chapter Introduction

Chapter 3 proposed an interpretation pipeline and a provenance model for interpretation. The interpretation pipeline captures users' interpretations and their metadata, running in the reverse direction to the visualization pipeline, to tag the raw data through the proposed provenance model. The proposed provenance model for interpretation adopts and extends the W3C PROV model. The presented concepts were domain-independent, addressing the need of tracking multi-user interpretations. Those interpretations explicitly identify visual features according to users' knowledge and intuition.

Chapter 4 gave background information about the domain of interest to this thesis, which is seismic imaging data. Interpretation of seismic data is about extracting geological features, such as horizons. Seismic data is noisy and of a high uncertainty. Different interpreters may produce different interpretations, despite visualizing the same view. Managing and tracking these interpretations is a challenge to the industry.

This chapter combines the knowledge presented in the two previously mentioned chapters to present an implementation of a provenance model and pipeline for seismic interpretation. An overview of a proof-of-concept prototype architecture is presented in Section 5.2. The subsequent sections discuss implementation details of the architecture's components. The source code of the prototype implementation is published online [30].

5.2 Prototype Architecture Overview

The proposed provenance-enabled interpretation pipeline (Figure 3.1) is translated into a proof-of-concept prototype architecture (Figure 5.1), which consists of three loosely coupled components as follows.

1. The data repository, which include the provenance model, is implemented as a fine-grained data structure. This is built on a parallel relational database management system (RDBMS), discussed in Section 5.3.
2. The visualization and interpretation pipelines, are combined and partially implemented as an intermediate user view, named FESVo as will be further discussed in Section 5.4.
3. The rendering and user interface are implemented in a third component, further discussed in Section 5.5.

This section provides a high level overview on the prototype architecture. The implementation details of the components are discussed in the next sections. This is a client-server architecture. It adopts the notion of *data-centric*. The term in this context means that the focus is more on the data (as described by Breitbart et al. [113]) in contrast to an *application-centric* approach. A *data-centric* approach allows the data to be managed by its data model, where an *application-centric* approach depends on the application to manage user data as is the case with *Petrel* and *DSG* (refer to Section 4.5). Also, this falls in line with the survey outcome presented in Section 4.7, which raised the need of an integrated and shared data repository among the different functionalities within the workflow.

It can be argued whether the data model should be added to the architecture. Although the term *architecture* has no unique definition in the field of computer science [114], an architecture must always consider what makes a system achieves its requirements. In a system, several types of architecture can interact; this might include a *software architecture* and a *data architecture* [115]. The latter considers the design of the data model. For the proposed architecture of this thesis, the data model is considered to be within the architecture due to its strong influence on achieving the objective of the whole architecture. The data structure (or data model) can be considered as a data architecture inside the complete (larger) architecture.

The data structure has the following essential features, which will be defined below:

1. fine-grained granularity

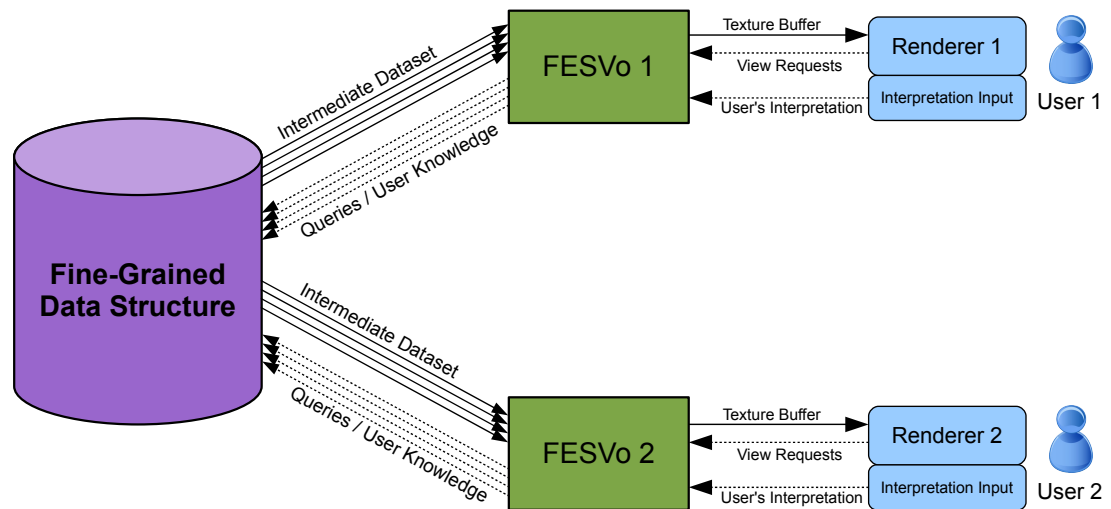


Figure 5.1: This is an implementation of the proposed abstract architecture as a data-centric visual analysis architecture which consists of three loosely coupled components. The fine-grained data structure in a centrally located database (1) is linked to multiple on-the-fly created feature-embedded spatial volumes (FESVo) (2) through parallel connections. The FESVo is linked to a renderer engine (3). In this diagram, two users are connected simultaneously. Dashed arrows indicate SQL queries from the FESVo to the database (0.2–0.7KB each), user inputs or requests for a texture buffer from a renderer. Full arrows indicate dataset transfer from the database to the FESVo (around 4KB each tuple consisting of a 1D seismic trace) and a texture buffer from the FESVo to a renderer (multiple megabytes).

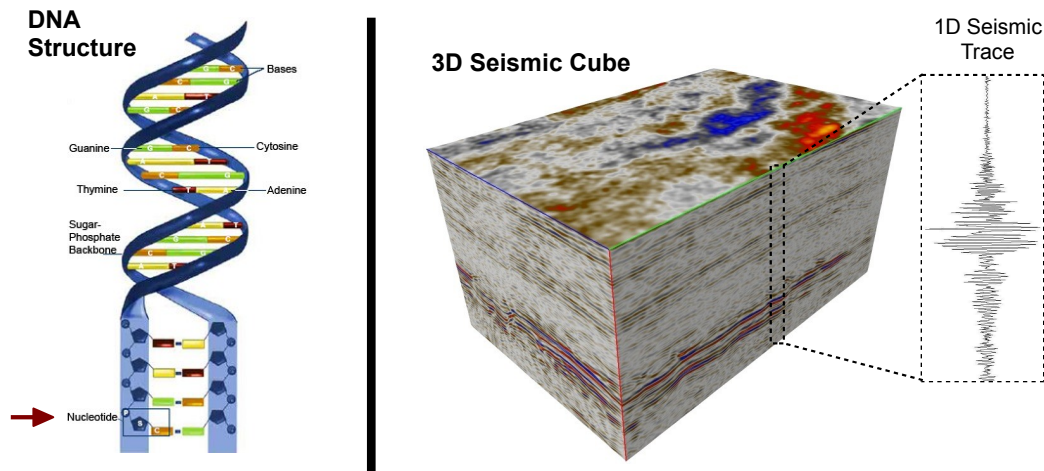


Figure 5.2: Data granularity is chosen based on the applied field. Working on DNA datasets, a nucleotide can be chosen as a granularity level as shown on the left hand side. Working on seismic imaging datasets, a valid granularity level is a seismic trace as shown on the right hand side. [Images sources: <http://katienelson4b.edublogs.org/> (left); <http://www.rwe.com/> (right).] (Note: The trace zoomed in on the figure is not actually extracted from the cube; the figure is for illustration purposes only.)

2. provenance-enabled host of raw data and users' inputs¹
3. multi-user single view

The first essential feature of the data structure is that it is fine-grained. Rather than dealing with coarse-grained objects, such as files, it is proposed to restructure the desired datasets into finer granularity for greater flexibility. The granularity of the data structure depends on the implementation and applied field. For example, the finest granularity could be a *nucleotide* in a DNA structure (biomedical science field), a *1D trace* in a 3D seismic dataset or a point with an x,y,z location in a 3D spatial dataset; see Figure 5.2. Section 5.3 discusses the application of this structure to 3D seismic imaging data and thus will determine a granularity level.

The second essential feature of the data structure is that it combines raw data and users' interpretations while preserving data provenance. The latter should include information about when a dataset was created, by whom it was created and how it was created. This means that a user visualizes data from, and contributes knowledge to, the same data repository. This would link raw data and users' knowledge into one

¹The terms *users' inputs*, *users' knowledge* and *users' interpretations* are used interchangeably in the context of this thesis. Although they literally do not mean the same, users' inputs and knowledge conceptually refer to their interpretations in this thesis.

data model for greater tightness. Also, the provenance should allow users to track the history of their input. The linkage between datasets depends on the application and implementation. Section 5.3 discusses how raw data and multi-users' inputs are linked together in the application of 3D seismic imaging data.

The third essential feature is that this data structure acts as a single-view for simultaneous multi-user access. This means that all users should be accessing the most updated information at any given time and can directly contribute for immediate access by other users.

Since it is proposed to deal with fine-grained data in this architecture, an intermediate stage is required to build, in real-time, a volume in a format which can be directly rendered. This stage must also capture users' interpretation (i.e. knowledge). In this implementation, the intermediate stage is named a *feature-embedded spatial volume* (FESVo) since it holds both raw data and users' interpreted features; this is discussed in Section 5.4.

The FESVo provides a renderer (discussed in Section 5.5) with a texture buffer ready for direct volume rendering. The primary roles of this intermediate stage are as follows: (1) load data, (2) cache data and (3) capture users' interpretations. The data loader in this component is customised based on the data type and granularity in the data structure. Thus, it is required to map between the coordinate of the data structure and a local volume coordinate which can be directly rendered. The data loader loads a desired dataset to be visualized considering the following: (1) the requested level of detail (resolution), (2) network bandwidth and (3) the graphics capability of the renderer (client's machine). The caching role of this intermediate stage is a basic optimization in order to eliminate re-visiting the data structure for loading previously fetched datasets. Other optimization techniques are discussed in the coming sections. The intermediate stage also captures users' interpretations. Thus, it is customised to take inputs from the user's interface. The component translates these inputs into a form which can be loaded back into the data structure.

The data structure explained above can be accessed by multiple pipelines, one for each connected user. As illustrated in Figure 5.1, the architecture links the fine-grained data structure on a database to one or more FESVo, created on the fly through parallel feature-aware and global spatially-referenced queries, which results in a parallel streaming of data units. The FESVo, on the other side, is linked to a rendering engine. Users' interpretations are stored back to the database.

5.3 Provenance-Enabled Data Structure for Seismic Interpretation

As discussed in Chapters 1 and 4, subsurface imaging datasets and geoscientists' interpretations are often stored in conventional files. In the proposed architecture, these datasets are restructured into a fine-grained form. The level of granularity depends on the type of the dataset. The choice of the granularity level is essential in this structure since datasets are referenced by, and accessed via, individual data fields; these fields are a result of dividing the datasets by the selected granularity level.

The following three types from the geoscience field are considered in this thesis and thus in the presented structure.

1. Seismic survey, which composed of traces—occasionally referred to as “raw data” in this thesis
2. Geological features as a result of users' interpretations
3. Assessment of users on interpretations

This section first discusses in detail the hardware and technology used for the prototype application. Then, an implementation (database schema) of the proposed provenance model is presented; it covers the three data types mentioned above as well as related metadata. The section ends with a discussion on data input into this structure; this includes data preparation.

5.3.1 Implementation Hardware and Technology

The following data model was implemented on a Teradata² relational database management system (RDBMS). Teradata, as explained by Bellatreche et al. [116], is a massively parallel processing database system, which is a shared-nothing database architecture³. The basic unit of parallelism in a Teradata system is called *access module processor* (AMP), which is a virtual processor [118]. A node runs multiple linked

² Teradata (<http://www.teradata.com/>) is a database software company. It develops and sells a relational database management system (RDBMS) under the same name, *Teradata*.

³ According to the Encyclopedia of Database Systems [117], a shared-nothing architecture is a distributed computing system in which each node is independent such that it has its own (non-shared) processor, main memory, disk and copy of the operating system. A shared-nothing architecture is often called massively parallel processor (MPP).

AMPs. Each AMP is assigned to a data portion. A hashing algorithm is used by Teradata for distributing and indexing entered data across the system's AMPs. The value of primary indexed columns is passed into a hash function to determine the AMP that owns the disk in which the row will be located as well as to determine the logical location of the row within the disk. The retrieval of a row follows the same procedure if primary index values were provided in the query [117]. This is known, in some literature, as a hash-partitioning physical layout [119]. The advantage and usage of this hash-partitioning is discussed later in Section 5.3.2.5.

Throughout the development of the architecture, a virtual version of Teradata running on a single-node 64-bit Windows Server 2003 was used. A 4-node parallel processing Teradata was then used to run a performance test; this was setup at Teradata labs and is discussed in Section 7.4. The former single-node environment was originally setup for the *massively-parallelised spatially-registered data structure* (SRDS) project by Irving et al. [23, 24] at the School of Earth, Atmospheric and Environmental Sciences (SEAES) of The University of Manchester. SRDS addresses the issue of storing terabyte-scale subsurface datasets. It aims to integrate subsurface data of different types into one data store.

The production schema of the SRDS data model and implementation details (apart from the previously mentioned publications by Irving et al.) were never released to the public, including the author of this thesis, as the work (of SRDS) was under commercialisation. However, the author was given limited access to an early prototype version of SRDS. In particular, this included a database table to store seismic surveys in a fine-grained form such that each trace occupies a row. Also, it included a converter and loader from conventional seismic imaging files (SEG-Y format) to that table on the database. This tool, which is further described in Section 5.3.4, was used by this PhD project to load the SRDS tables with seismic data to be then transferred to the database tables of this thesis.

This PhD used this Teradata environment to build its provenance model, as described below in Section 5.3.2. In addition to this, a collaboration with SEAES of The University of Manchester took place in the form of supplying datasets (i.e. seismic surveys and interpreted horizons) and participating in usability tests for the work of this PhD (presented in Section 7.2).

Despite the choice of Teradata RDBMS in this PhD, it should be possible to implement the presented data model using alternative RDBMSs such as Oracle⁴ and

⁴<https://www.oracle.com/database/index.html>

MySQL⁵. However, this has not been tested as part of this thesis. Due to the various indexing and partitioning techniques that RDBMSs offer, the scalability of each implementation (i.e. choice of an alternative RDBMS) should also be tested.

5.3.2 Data Model

5.3.2.1 The *geo* Namespace

Prior to translating the provenance model presented in Section 3.3 into a database relational model, types of entities need to be defined for the application domain. In Section 3.3, the thesis presented the *interpret* namespace to define types of activities and relationship for the purpose of general interpretation. As stated in that section, the model delegates the definition of entity types to the application as they are domain-specific. Thus, the thesis introduces a namespace for geological data, named *geo*. For the current implementation, it includes the following data types: *seismic survey*, *trace*, *horizon*, *fault* and *point*. Other types, which serve the workflow of seismic interpretation and geological modeling, can fit into the *geo* namespace, such as *well log*, *reservoir model* and *pick*; but they are out of the scope of this thesis. Another type of interest introduced in this thesis is *assessment*. An assessment is an example of an annotation on interpretation, which the provenance model allows. It represents a user's evaluation of their own or others' previous interpretations.

Table 5.1 presents the defined types of entities under the *geo* namespace and to the interest of this thesis. Figure 5.3 shows a provenance model illustration of a basic seismic interpretation case. It is similar to Figure 3.3 but entities from the *geo* namespace were used instead. Also, the figure illustrates that *seismic survey* and *horizon* entities are of type collection, composing of traces and points, respectively. Each of these types are further discussed in the coming subsections.

5.3.2.2 Database Relational Model—An Overview

To address the implementation concerns discussed in Section 3.4, the data structure of the prototype architecture of this thesis is implemented as a relational database. Other implementations, such as a NoSQL database, may address the concerns differently; but none were examined in this PhD. The following explains how an implementation of a relational database would address those concerns:

⁵<http://www.mysql.com/>

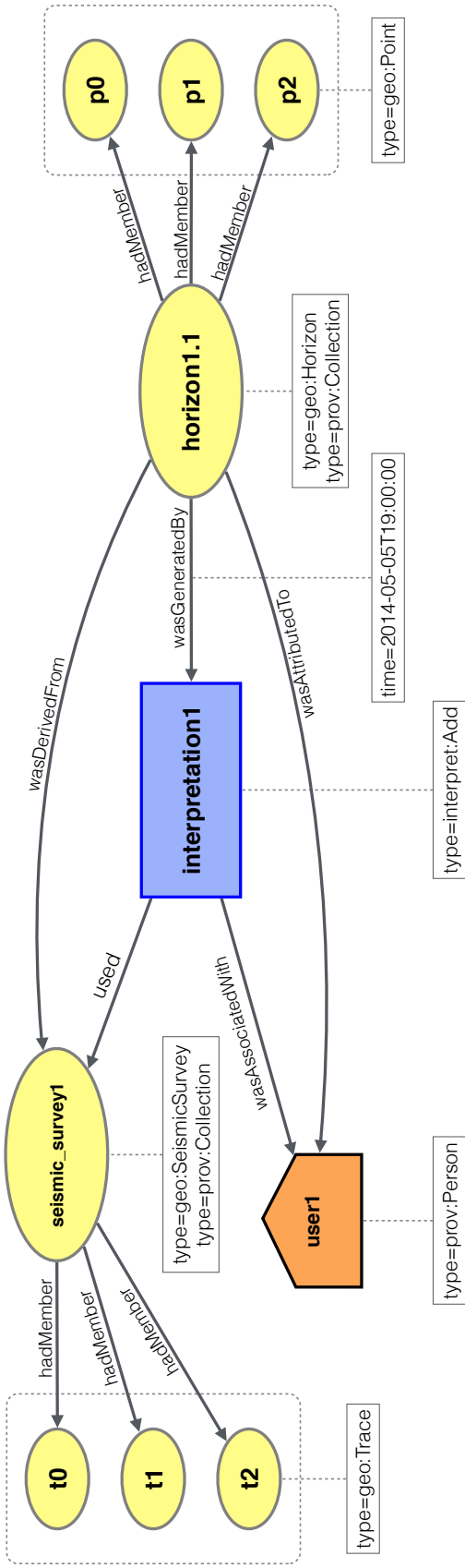


Figure 5.3: A graphical illustration of the provenance model of a basic seismic interpretation case. Properties of the illustrated entities, such as geographical location, were omitted for illustration simplicity.

Type	Core concept
geo:SeismicSurvey	Entity
geo:Trace	Entity
geo:Horizon	Entity
geo:Fault	Entity
geo:Point	Entity
geo:Assessment	Entity

Table 5.1: The table shows defined entity types under the *geo* namespace which is introduced in this chapter to accommodate geological data.

1. The concern of a “fine-grained data granularity” is achieved by representing fine-grained data units as tuples, which are implemented as rows in a database. Each considered data types (raw data and users’ interpretations) has a data granularity level. The following text will explain this in detail.
2. The concern of an “open-standard access” is achieved by the ability to access the data via a standard structured query language (SQL).
3. The concern of a “single-view multi-user access” is achieved by implementing the data structure in a central database where multiple users can access and update a single-view of the data concurrently.

Referring to Figure 5.3, the provenance model presented in Section 3.3 is translated into a database relational model for seismic interpretation as follows. First, the following assumptions are adopted for the purpose of the prototype implementation; these can also be viewed as implementation restrictions which are neither adopted by the original PROV model nor the provenance of interpretation model presented in this thesis.

1. An activity of interpretation must only generate one entity, such as a horizon.
2. The agent associated with the activity that generated the feature (entity) is the same agent attributed to the generated entity.
3. The generation time is adopted, while start and end times of the activity are neglected.

Figure 5.3 plus the above discussion can translate the provenance model for interpretation into three fundamental entities in the context of an entity-relationship model: *survey*, *interpretation* and *agent*. A seismic survey is an entity composed of multiple

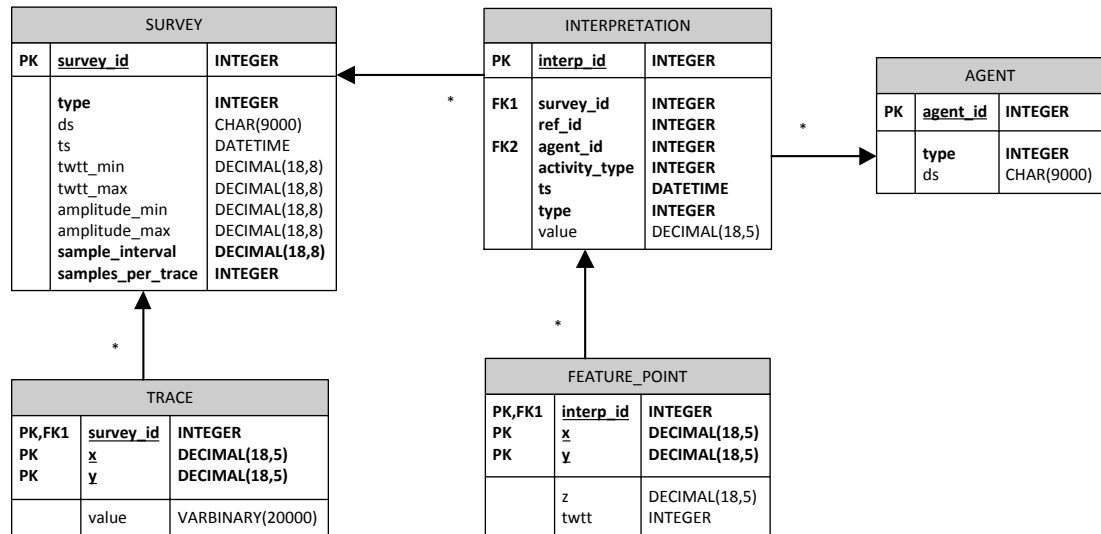


Figure 5.4: This diagram illustrates a database schema as an implementation of the proposed provenance model for seismic interpretation. Abbreviations used here are as follows; ds : source description; ts : timestamp; twtt : two-way-travel time. Supporting tables within the schema were omitted from this figure for simplicity of illustration.

geospatially located seismic traces. An interpretation can represent an entity of either a geological feature (e.g. horizon) or an assessment; the latter falls under the general meaning of interpretation and thus it is included. An interpretation is composed of geospatially located points. The interpretation activity and its generation relationship, as shown in the provenance model graphical illustrations, are both included in the interpretation class of this implementation. An agent is responsible of performing an interpretation. For implementation, an agent is translated from the notion of a provenance agent into an entity in the context of an entity-relationship model. Figure 5.4 illustrates this entity-relationship model which represents the implemented database schema; each entity is represented by a table. The following subsections discuss the data and corresponding tables of this database model in detail as well as its data indexing.

5.3.2.3 Raw Seismic Imaging Data

Raw seismic data can be divided into the following different granularity levels:

1. 3D volumes—each represents a seismic survey and is conventionally stored in large SEG-Y (the standard format) or an application’s proprietary format files
2. 2D slices—in-lines or cross-lines of a seismic survey (described in Section 4.2)

3. traces—each corresponds to a 1D dataset that consists of an array of amplitudes
4. amplitude values—each is a 4-byte floating point real number

From the list above, a 3D volume is the conventional method of storing seismic data and is the coarsest granularity level. On the other hand, amplitude values is the finest granularity level. However, the proposed data structure sets the granularity at the seismic trace level for the following reasons. The trace length of raw seismic data is fixed across one dataset (i.e. seismic survey). Also, it is relatively small compared to the survey size that it belongs to; it often consists of around 200 to 2000 samples per trace. On the other hand, setting the granularity at the amplitude level would result in an overhead of data storage and retrieval, while most retrieval processes in practice fetch a complete trace⁶. Therefore, this thesis proposes a granularity at the trace level for seismic data. This granularity, which is at a level of 1D dataset, may not suit other data types from other fields such as medicine or oceanography. Each field has to be studied carefully to select a suitable granularity level for its data to be implemented in a fine-grained data structure that fits the abstract proposed architecture of this thesis.

Now that granularity level is set at the trace level, each trace in this data structure is placed in a tuple. This is implemented as a row in table **TRACE** as illustrated in Figure 5.4. A trace is described by the following fields:

1. **A survey identifier (ID) (`survey_id`):** this identifies the seismic survey to which this trace belongs.
2. **Geographical location (`x`, `y`):** this is the geographical location of the trace as recorded in the trace header of the source SEG-Y file from which the trace was imported.
3. **Property value (`value`):** this is a customised binary-based type field that holds the trace samples. The samples can be of different data types but it is commonly an *IEEE floating-point* format. These samples are in time scale. Geophysically, the time in this scale is known as *two-way travel time* (TWTT)⁷. The time of the first and last samples of all traces of one seismic survey are fixed and thus

⁶One case where only one sample value per trace is required to be fetched, and not the complete trace, is when visualizing a *time slice*; refer to Figure 4.4.

⁷In geophysics, a *two-way travel time* (TWTT) is “the elapsed time for a seismic wave to travel from its source to a given reflector and return to a receiver at the Earth’s surface,” as defined by Schlumberger’s Oilfield Glossary (<http://www.glossary.oilfield.slb.com/>).

Filed Name	Type	Description
survey_id	integer	The survey identifier to which the trace belongs
x	decimal	x value of the trace's geographical location
y	decimal	y value of the trace's geographical location
value	binary	trace samples

Table 5.2: A summary of table **TRACE** fields.

recorded in the survey table, as described next. The sample interval (time between samples) is also fixed among traces from one survey and is thus recorded in the survey table too.

Traces are indexed on the combination of (x, y) coordinate and survey_id. Table 5.2 summarizes the fields of table **TRACE**.

As illustrated in Figure 5.3 and explained above, a seismic survey is composed of traces. Table **TRACE** is a child of table **SURVEY**, related through a survey ID. Table **SURVEY** holds the common attributes of the related traces. A survey is described by the following fields:

1. **A survey identifier (ID) (survey_id):** this identifies the seismic survey.
2. **Property type (type):** this field describes the type of the property value of the related traces. For example, a property ID of 1 may describe a raw seismic trace while an ID of 2 may describe a derived trace of one seismic attribute. The current implementation of this thesis only deals with raw seismic traces.
3. **A description (ds):** this describes the source such as the name of the dataset.
4. **Timestamp (ts):** this is the time at which this survey was created. This field was not utilised in the current work of this PhD. However, the field was created for future work such as creating a derived version of the survey in which a timestamp would be useful⁸.
5. **Two-way travel time (TWTT) range (twtt_min, twtt_max):** this is the range of the time scale of this survey's traces. This range is constant for all

⁸Derived data can be calculated from raw seismic data. This is known to geoscientists as a *seismic attribute*; a list of 3D seismic attributes is presented in "Atlas of 3D Seismic Attributes" by Randen, T. et al. [87].

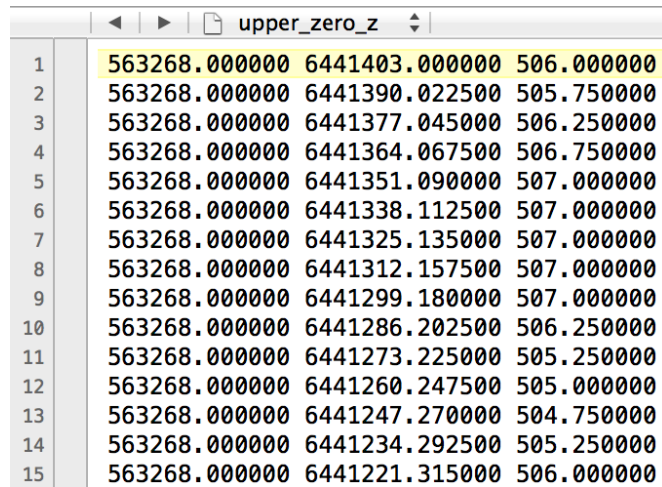
Filed Name	Type	Description
survey_id	integer	an identifier (ID)
type	integer	type of related traces' property values
ds	varchar	a description
ts	timestamp	the time at which this survey was created
twtt_min	decimal	minimum TWTT value of this survey's traces
twtt_max	decimal	maximum TWTT value of this survey's traces
amplitude_min	decimal	minimum amplitude value of all traces' samples of this survey
amplitude_max	decimal	maximum amplitude value of all traces' samples of this survey
sample_interval	decimal	sample interval of this survey's traces in milliseconds
samples_per_trace	integer	number of samples in each trace of this survey

Table 5.3: A summary of table **SURVEY** fields.

traces of one survey. The range is used to match the location of a feature point against a single trace sample during the preparation of rendering; this mapping process is further discussed in Section 5.4.1.1.

6. **Amplitude range (`amplitude_min`, `amplitude_max`):** this is the range of the amplitude values for all traces of this survey. This range is required to create a proper grey-scale colour for the raw data to be rendered; the rendering aspect is further discussed in Section 5.5.
7. **Sample interval (`sample_interval`):** this is the sample interval for the traces of this survey, in milliseconds (since samples are on a time scale). This value is constant for all traces of one survey. It is used, in addition to the TWTT range, to match the location of a feature point against a single trace sample during the preparation of rendering.
8. **Samples per trace (`sample_per_trace`):** this is the number of samples per trace. This value is constant for all traces of one survey.

Survey records are indexed by the *survey ID* (`survey_id`). Table 5.3 summarizes the fields of table **SURVEY**.



	x value	y value	twtt
1	563268.000000	6441403.000000	506.000000
2	563268.000000	6441390.022500	505.750000
3	563268.000000	6441377.045000	506.250000
4	563268.000000	6441364.067500	506.750000
5	563268.000000	6441351.090000	507.000000
6	563268.000000	6441338.112500	507.000000
7	563268.000000	6441325.135000	507.000000
8	563268.000000	6441312.157500	507.000000
9	563268.000000	6441299.180000	507.000000
10	563268.000000	6441286.202500	506.250000
11	563268.000000	6441273.225000	505.250000
12	563268.000000	6441260.247500	505.000000
13	563268.000000	6441247.270000	504.750000
14	563268.000000	6441234.292500	505.250000
15	563268.000000	6441221.315000	506.000000

Figure 5.5: This is a screenshot of part of an exported geological horizon from the Petrel application. Each line represents a point; thus a cloud of points. The columns from left to right represent the following: the x value of the point’s geographical location, the y value of the point’s geographical location and the point’s two-way travel time (twtt).

5.3.2.4 Geological Features

A geological feature object, a result of a user’s (or users’) interpretation, is conventionally stored as a polygon mesh. During an early interpretation phase, these features can also be stored and represented as a cloud of points⁹. These objects can be exported from a conventional seismic interpretation application in different formats, one of which is a cloud of points. Commonly, such a format would include a list of x , y and *two-way travel time (TWTT)* values of the points that lie on the surface of the exported feature object. Figure 5.5 shows a screenshot of part of an exported horizon from the Petrel application.

In terms of granularity, a single feature object is conventionally represented as a whole (one mesh); thus it is a coarse-grained granularity. In the proposed data structure, features objects are represented as a cloud of points, a fine-grained granularity at the point level. Each point is placed in a tuple. This is implemented as a row in table **FEATURE_POINT** as illustrated in Figure 5.4. The following text refers to such a point, which lies on the surface of a geological feature, as a *feature point*. A feature point is described by the following fields:

⁹A cloud of points is a set of data points in a coordinate system; e.g. in a 3D space each point is defined by an (x,y,z) coordinate.

1. **An interpretation identifier (ID) (`interp_id`):** this identifies the interpretation object (and activity) to which this point belongs.
2. **Geographical location (`x`, `y`):** this is the geographical location of the feature point. This should match the location of the trace which this feature point marks.
3. **Depth (`z`) and two-way travel time (`twtt`):** these two represent the third dimension of the location of this feature point as follows.
 - In the seismic interpretation phase, a feature point should lie on a seismic trace sample to which it marks. Thus, the feature point should match the sample's location with a time scale in the third dimension (`x`, `y`, `TWTT`). In this case, the `twtt` field is used to compute the location of the feature point, in addition to `x` and `y`. The TWTT of a trace sample, and thus of the feature point that marks it, is calculated as follows: $T_p = FirstSampleTWTT + SampleIndex \times SampleInterval$.
 - After completing the seismic interpretation phase, geoscientists convert their time scaled seismic and interpreted objects into a depth model through a process called *time-depth conversion*. This process requires a knowledge about the velocity at which a wave travels through the different materials of the subsurface layers. This process is beyond the scope of this PhD. Also, the `z` field was not utilised as no depth scaled data was handled in this PhD. However, the `z` field in this table is available for future work, i.e. if time were to be converted to depth or if a depth scaled data were imported into the proposed data structure.

Feature points are indexed by the combination of (`x`, `y`) coordinate and `interp_id`. Table 5.4 summarizes the fields of table **FEATURE_POINT**. This representation of a cloud of points is needed to record provenance of users' evolving opinions that might represent partial modifications to interpretations. It should be noted that this is the primary representation of geological features in the data model of this thesis. However, features can be of the same or different representation after being loaded to the FESVo (5.4.1) for rendering.

As illustrated in Figure 5.3 and explained earlier, an interpretation activity generates an interpreted feature which is composed of feature points. Table **FEATURE_POINT** is a child of table **INTERPRETATION**, related through an interpretation ID. Table

Filed Name	Type	Description
interp_id	integer	The interpretation identifier to which this point belongs
x	decimal	x value of the feature point's geographical location
y	decimal	y value of the feature point's geographical location
z	decimal	the third dimension in depth (z value)
twtt	integer	the third dimension in two-way travel time

Table 5.4: A summary of table **FEATURE_POINT** fields.

INTERPRETATION is a translation of both an interpretation activity and its generated interpretation entity. An interpretation is described by the following fields:

1. **An interpretation identifier (interp_id):** this identifies the interpretation object (and activity).
2. **A survey identifier (survey_id):** this identifies the survey to which this interpretation was derived from.
3. **A referenced interpretation identifier (ref_id):** this identifies a previous interpretation that this interpretation modifies. The value of the referenced identifier can be equal to this interpretation identifier if it is the first interpretation of an object.
4. **An agent identifier (agent_id):** this identifies the agent that was associated with this interpretation.
5. **Activity type (activity_type):** this indicates the type of activity of this interpretation. Three relations are considered in the prototype implementation:
 - (a) **Insertion:** the feature points belonging to this interpretation of this type represent a continuation (addition) to the referenced interpretation (ref_id).
 - (b) **Deletion:** the feature points belonging to this interpretation discard points of the same locations from the referenced interpretation.
 - (c) **Assessment:** the feature points belonging to this interpretation adds a confidence level, set by this interpretation's value, to the referenced interpretation.

Filed Name	Type	Description
interp_id	integer	an identifier for this interpretation
survey_id	integer	the survey identifier
ref_id	integer	the identifier of a previous (referenced) interpretation
agent_id	integer	the identifier of the responsible agent
activity_type	integer	type of activity
ts	timestamp	interpretation generation timestamp
type	integer	type of resulting interpretation
value	decimal	(optional) property value

Table 5.5: A summary of table **INTERPRETATION** fields.

6. **Timestamp (ts):** this is the time at which an interpretation was generated. It should be the same time at which related points were inserted into their table. This field is essential for history tracking of interpretations.
7. **Interpretation Type (type):** this field describes the type of interpretation. For example, a type of value 1 may describe a geological horizon, value 2 may describe a geological fault and value 3 may describe an assessment of interpretation.
8. **Property value (value):** depending on the type of property, this field may either represent a single measured value (e.g. porosity, permeability) or an assessment value (e.g. confidence level); only the latter is demonstrated in this thesis.

Interpretations are indexed by their identifiers (`interp_id`). Table 5.5 summarizes the fields of table **INTERPRETATION**.

5.3.2.5 Data Index

Indices in databases are used to speed up the retrieval process of data from tables. In an environment like seismic imaging data, where data size is hundreds of gigabytes to some terabytes stored in a fine-grained granularity, it is important to consider the type of the index to use and the column(s) to index to address the cost of the index structure created by the RDBMS. Depending on its type, a database index often increases in size as table size increases. This may then slow the data write back into tables.

As previously mentioned in this chapter, Teradata, the hosting RDBMS of the presented data model, adopts a hashing algorithm for partitioning and indexing. This strategy determines the row location on-the-fly without construction or storage complexity. Thus it should overcome the above mentioned issues of increasing the indexing structure size and slow write back; this is only the case with primary indices as a subtable is required for a secondary index. However, this indexing strategy requires a careful selection of the primary index to provide an even distribution of the rows among Teradata AMPs [120]; each AMP is assigned to a data portion as explained in Section 5.3.1.

In this prototype implementation of the data model the indexed columns for traces and feature points are the *geographical location* (x and y) and *ID*, as previously presented in this chapter. Using explicit values of these columns in a query, data retrieval can be performed at a complexity that is proportional only to the working dataset (the size of the dataset being retrieved) and not to the total size of the tables. A similar performance is also achieved during a write back to the database if a secondary index is not used. A failure to provide an indexed column value in a query may result in a full table scan for data retrieval. The hashing algorithm of Teradata was used in the implementation of the presented data model as provided by Teradata without any modification or access to its internal hash functions.

5.3.3 Multi-Interpretations and History Tracking—Model Mapping Example

Referring to the research problem of this thesis, multi-user understandings (interpretations) need to be captured. As presented so far in this chapter, interpretation activities are recorded in the **INTERPRETATION** table. It links multiple interpretations of a single feature through its `ref_id` field and to their corresponding survey through `survey_id`. In this subsection, an example of multiple interpretations is first illustrated in a provenance model form then mapped to the presented relational database model.

Figure 5.6 illustrates a multi-user seismic interpretation case. Initially, a software generated an interpretation (i.e. extraction) of a geological horizon (`horizon1.1`). Later, `user1` added `addition1` towards the initial interpretation, generating `horizon1.2`. Later, `user2` deleted a portion, which is defined in `deletion1`, from `horizon1.2`; this generated `horizon1.3`. Later, `user3` assessed part of `horizon1.3`, giving a confidence level of 70% as defined in `assessment1`. This model can also be expressed

interp_id	survey_id	ref_id	agent_id	activity_type	ts	type	value
1	1	1	1	1	T1	101	-
2	1	1	2	1	T2	101	-
3	1	2	3	2	T3	101	-
4	1	3	4	3	T4	201	'70'

Table 5.6: Populated table **INTERPRETATION** with data reflecting the example case illustrated in Figure 5.6. Note: timestamps were written as T1, T2, etc, for a symbolic purpose only.

using PROV-N as described in Section 3.3.2.

The above example is translated into the relational data model as follows. The prototype implementation does not physically record the derived `horizon1.2` and `horizon1.3` since they are views (references) that contain no feature points. They are generated on a query-demand basis. Users' interpretations are accumulated and are never (as per the prototype implementation) deleted physically, but can be tagged as deleted so users can roll back on their interpretations chronologically. Note that users' interpretations form a fraction of the size of raw data (less than 10%). Table 5.6 shows the **INTERPRETATION** table populated with information of this example in assumption of the following:

1. Interpretation IDs of `horizon1.1`, `addition1`, `deletion1` and `assessment1` are coded as 1, 2, 3 and 4, respectively.
2. Survey ID of `seismic_survey1` is coded as 1.
3. Agent IDs of `software1`, `user1`, `user2` and `user3` are coded as 1, 2, 3 and 4, respectively.
4. Activity type IDs of `interpret:add`, `interpret:delete` and `interpret:assess` are coded as 1, 2 and 3 respectively.
5. Types IDs of `geo:Horizon` and `geo:Assessment` are coded as 101 and 201, respectively.

The following is a simple method to retrieve a geological feature that was contributed by multiple users; this method has some limitations which are discussed in Section 8.2.2.3. The database is queried such that the feature points of an interpretation of `interpret:add` activity type are added and feature points of `interpret:delete` activity type are subtracted. This is illustrated in the following pseudo query, which

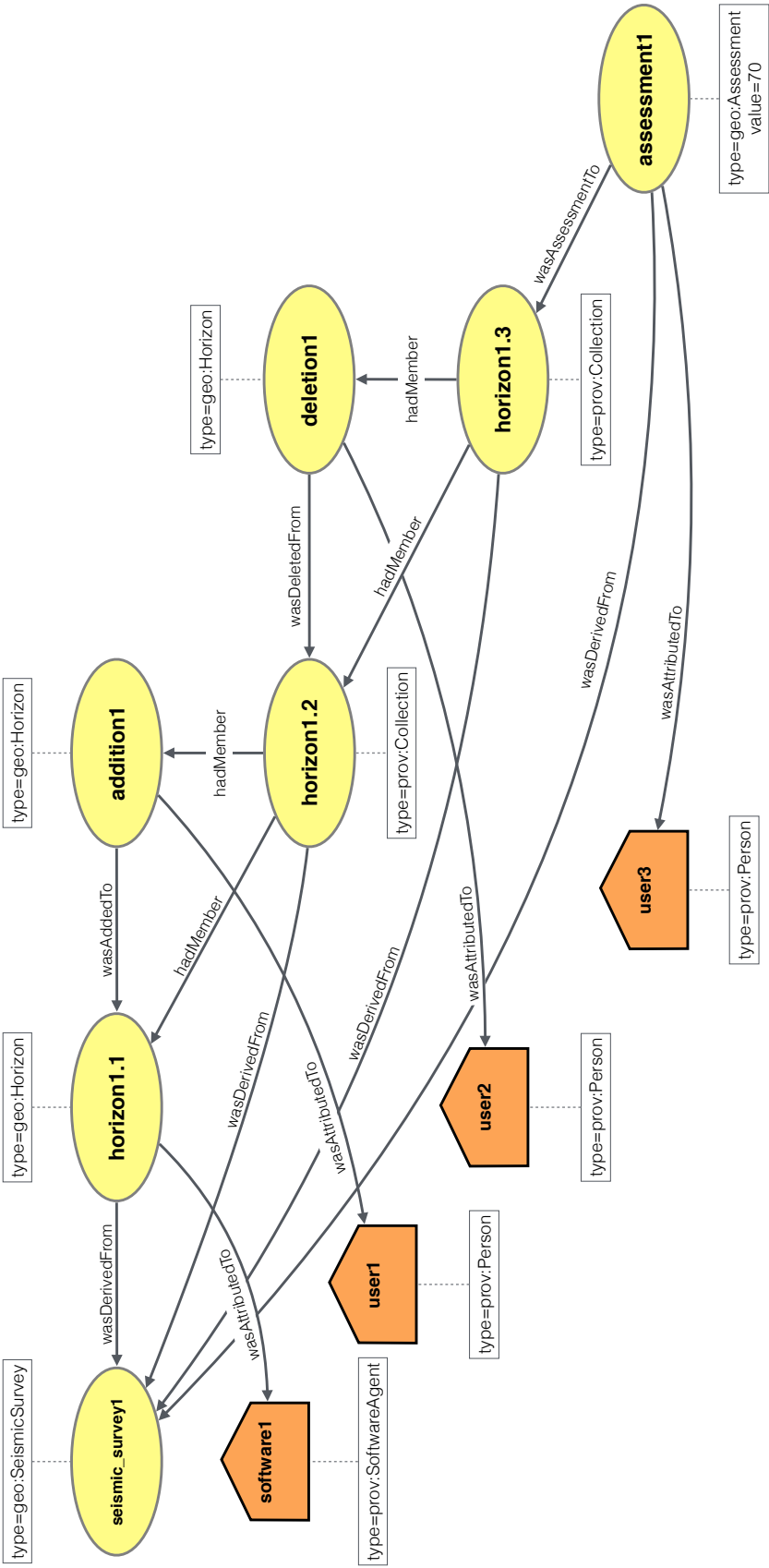


Figure 5.6: A graphical illustration of the provenance model of a multi-interpretation seismic interpretation case. Activities and some properties of entities and relationships were omitted for simplicity of illustration.

retrieve points of all interpretation of survey *s* before time *t*. History tracking is controlled by manipulating the `timestamp` value. Also, tree branches can be tracked via the `ref_id` field of the **INTERPRETATION** table; this is not illustrated in the pseudo query.

```

SELECT points from FEATURE_POINT
JOIN INTERPRETATION
ON interp_id
WHERE survey_id = s
      AND timestamp < t
      AND activity_type = interpret:add

EXCEPT
SELECT points from FEATURE_POINT
JOIN INTERPRETATION
ON interp_id
WHERE survey_id = s
      AND timestamp < t
      AND activity_type = interpret:delete

```

Assessment data is retrieved separately to graphically annotate matching feature points. Also, it is utilised in filtering the retrieval of feature points matching a certain assessment criteria. This is performed via complex queries, as implemented in the prototype application. Graphical results of assessment related data are shown in the next chapter, which is dedicated for a walkthrough and screenshots.

5.3.4 Data Feed

The implemented data repository is initially loaded with data from SEG-Y files and geometry as illustrated in Figure 5.7; the latter is loaded to have some initial interpretation of geological features. Data needs to be prepared and converted from its conventional format to the form of the presented data model.

The *SEG-Y loader* of SRDS [23] was utilised to load traces of a SEG-Y file into the **SURVEY** and **TRACE** tables on the database. As illustrated in Figure 5.8, *SEG-Y loader* reads a SEG-Y file extracting its traces' locations and samples. Common survey information and a generated survey ID are loaded into the **SURVEY** table. Extracted information of each trace (location and samples) of this survey is loaded into the **TRACE** table along with its parent survey ID.

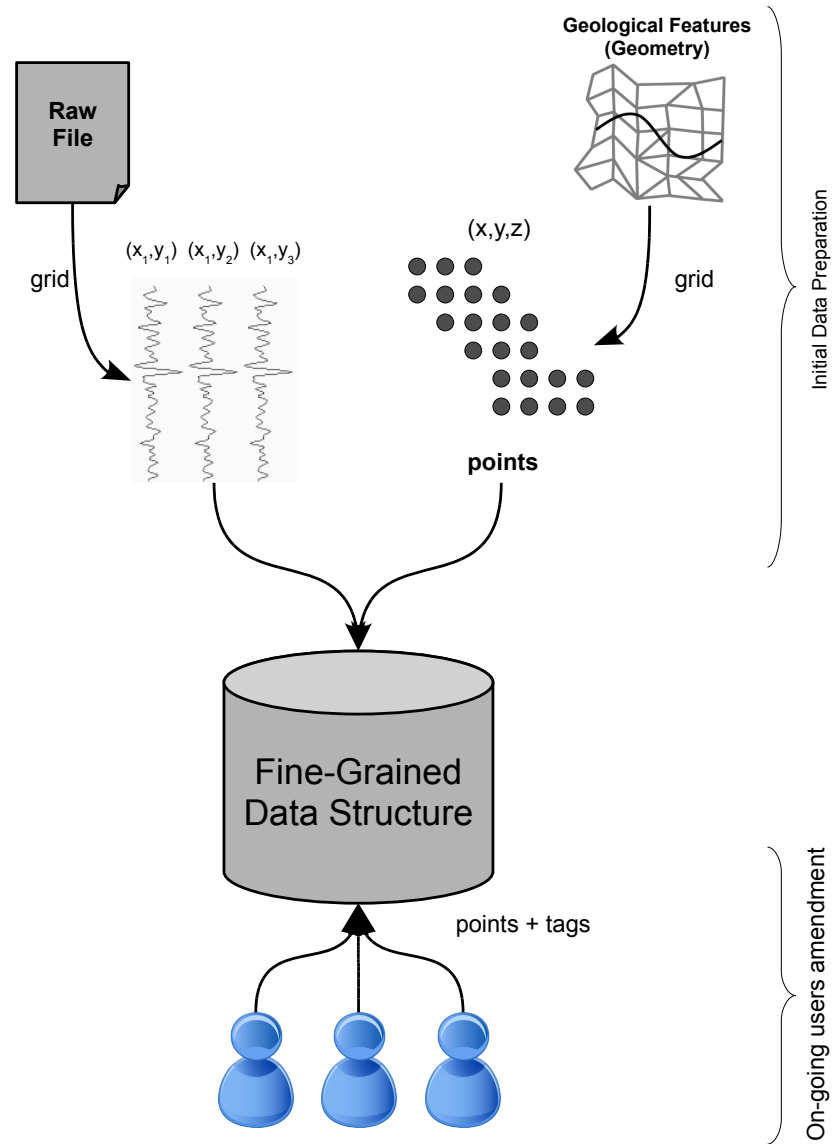


Figure 5.7: This diagram illustrates an overview of the data feed into the database. Seismic datasets on the database are initially prepared from SEG-Y files for raw data and geometry files for geological features. Amendments to feature objects (addition/deletion) are later updated directly from users to the database.

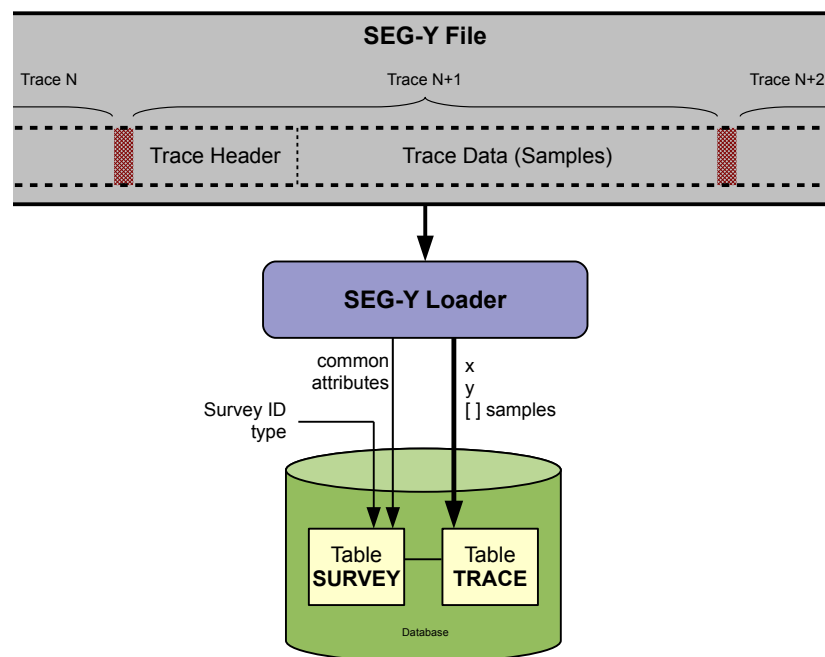


Figure 5.8: The diagram illustrates how the SEG-Y loader operates. It reads a SEG-Y file to extract individual traces. For each trace, its geographical location is extracted from the trace header and its samples are extracted from the trace data. These values are combined with a generated survey ID to be loaded into table **TRACE** on the database. Common information for the whole survey is loaded to table **SURVEY**.

To start with some initial data, geological features are exported from Petrel, a well-known conventional seismic interpretation application, as a cloud of points; these exported features have previously been interpreted by users. Figure 5.5 shows a screenshot of part of an exported horizon as a list of x , y and *two-way travel time (TWTT)* values of the points that lie on the surface of the exported feature object. A record in table **INTERPRETATION** is first created for each feature object, generating an interpretation ID and populated with common information such as related survey ID and feature type. Then, information of each feature point (x , y , $twtt$) combined with the generated interpretation ID are loaded into the **FEATURE** table.

As illustrated in Figure 5.7, on-going users' amendments to the features are directly stored in the same format, as a cloud of points with the proper metadata. Further details about capturing users' interpretations are discussed in Section 5.4.3.

5.4 Feature-Embedded Spatial Volume (FESVo)

Following the prototype architecture, which was presented in Section 5.2, the visualization and interpretation pipelines, are combined and partially implemented as an intermediate user view. In general, this component prepares a dataset from an unconventional data structure¹⁰ in a form which can be rendered by a graphics card. Also, most importantly to this thesis, it captures users' input (interpretations and annotations) from the user interface and transfer it back to the central data repository following the provenance model presented in Chapter 3. In this prototype implementation for seismic imaging data, this component is named FESVo which was historically short for *feature-embedded spatial volume* as illustrated in Figure 5.1.

The name FESVo was inspired by the original purpose of the component during the development of the prototype application, which was to provide an intermediate view. This intermediate view includes both raw data and features interpreted by users. Hence, the name *feature-embedded spatial volume* was coined as part of this PhD. However, the role of the FESVo is more than to provide an intermediate view or volume. The roles of the FESVo are: (1) data loading, (2) data caching and (3) capturing users' interpretations. The internal architecture of the FESVo is illustrated in Figure 5.9, which was implemented on Java standard edition (SE) platform¹¹. As Figures 5.1

¹⁰ This thesis refers to the data structure presented in Section 5.3 as *unconventional* since it has not been a standard to store volumetric data for visualization and interpretation purposes (e.g. seismic imaging) in database rows at a fine-grained granularity.

¹¹<http://www.oracle.com/technetwork/java/index.html>

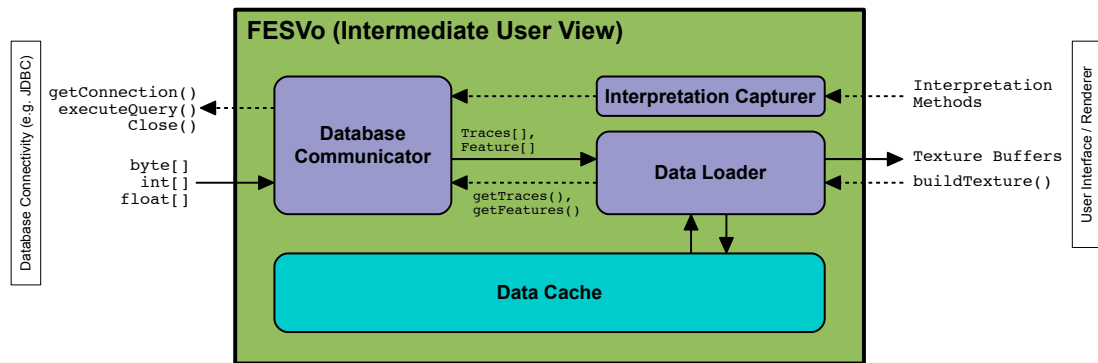


Figure 5.9: This is a conceptual digram of the internal architecture of the FESVo. Dashed arrows indicate requests. Full arrows indicate data movement. Text around arrows are only examples of requests and data types.

and 5.9 suggest, the implementation of the architecture includes three memory stages as follows.

1. main data repository—located in the database (server side) and considered as permanent
2. FESVo cache—located in the client’s main memory
3. 3D texture¹² image—located in the client’s GPU texture memory; this is the actual rendered 3D image

Therefore, the architecture can be viewed as adopting an out-of-core visualization pipeline since the dataset (located in the database) is larger than a computer’s internal memory (FESVo cache); that is in addition to being a client-server architecture. The following subsections discuss the three roles of the FESVo in detail as well as some optimization techniques.

5.4.1 Data Loading

The role of *data loading* is performed by the **Data Loader** object of the FESVo architecture as illustrated in Figure 5.9. The FESVo uses the indexing capability of the data structure to run queries that explicitly include values of indexed columns. This thesis refers to such queries as *point-to-point*, since by providing a point of x , y and ID (i.e. *survey_id* or *interp_id*) values the location of the associated row can be directly

¹² In the context of this thesis, the term *texture* always refers to the 3D image loaded into the texture memory of the client’s graphical processing unit (GPU).

found without performing a full-table scan, as discussed in Section 5.3.2 regarding data indexing. The result of queries is an intermediate volume which can be directly rendered on the GPU. As previously discussed in Section 4.2, seismic traces in a conventional SEG-Y file are stored in a sequence according to their in-line and cross-line numbers, which are local references (and thus local locations) of traces within the file. A distinguishing feature of this method is that the traces, as well as feature points, in the data structure in this thesis are referenced by, and thus retrieved by querying, their global (geographical) locations.

The following explains the initialization and loading process steps in a conceptual manner; more details about the actual implementation can be obtained by accessing the code of the FESVo [30]. Some of the following steps are further explained under separate headings. The prototype software implementation of an FESVo allows a user session¹³ to load a single seismic survey dataset (raw data) and multiple related features selected by the user at the launch of the session. However, to provide the possibility of using multiple dates in the future, the data structure and the FESVo are designed to allow this. A user session starts with the selection of a survey dataset and some related features from the data repository. Upon this selection, the session performs the following processes in sequence.

1. Metadata of the selected data (raw and features) are cached in the FESVo.
2. A mapping between the coordinate systems of the three data stages (database, FESVo cache and texture object) is defined. For this, the selected raw dataset is gridded. As a result, the dimension of highest resolution level and the rendered texture (considering the allowed GPU texture memory size) are both computed. These are further explained below in Section 5.4.1.1.
3. If not already cached in a local file, the selected datasets (raw and features) are scanned and their data coordinates are loaded into FESVo and cached in a local file for future use. This is further discussed below in Section 5.4.1.2.
4. A rendering engine (as part of the user interface) requests a buffer¹⁴ object of a volume, at a desired level of detail, and a viewing reference point to be loaded into a texture object for direct volume rendering.

¹³ A *user session* or an *application session* in this context is defined to be a period of time during which a user interacts with the application. In particular, the session starts when a user selects a dataset to work with and it is ended by closing the application or at least closing the working dataset.

¹⁴ A *buffer*, in general, is a container of data—as defined by many application programming interfaces (APIs).

5. Each point on the x-y plane of the requested texture buffer is to be loaded with a 1D trace or an array of feature points on a 1D line; each is named a *data unit* in this context. For each data unit, the loader first maps its location into the FESVo coordinate system and then checks the FESVo cache. If found, the data unit is placed in the texture buffer at the computed (mapped) position. If a data unit is, otherwise, not cached, it is added to a queue. Data units to be loaded from the database are balanced among multiple queues, each is associated with a loading thread. This is to optimize the loading process, further discussed below in the section on optimization (5.4.4).
6. After completing the search in the internal cache of the FESVo, the threads, each with its queue of data units (locations), start to be concurrently fetched from the database. For each data unit in the queue, the location is mapped into the coordinate system of the real data (i.e. geographical locations). Each fetched data unit is loaded to the FESVo and placed in the texture buffer at the computed (mapped) position.

The use of a hashing indexing algorithm by the RDBMS, combined with queries against indexed column values, allows data to be retrieved at a complexity of $O(k)$, where k is the number of data units being fetched from the database; this is independent of the total size of the table (dataset). Thus, downsampling (by decimation) is performed on the fly when a lower resolution is requested.

5.4.1.1 Gridding, Coordinate Mapping and Level-of-Detail

As mentioned earlier in this section, the architecture is a client-server that can be viewed as an out-of-core pipeline. To stream data into this pipeline, a *mappable* algorithm must be applied such that it knows what pieces of data to fetch to create the required output, as highlighted by Moreland [2]. The author (Moreland) adds that execution management, and thus data streaming, is more effective with regional metadata. The following explains the use of geographical locations (i.e. regional metadata) of data units in setting up a straightforward mapping¹⁵.

For the purpose of data loading and filtering, the architecture deals with the following three coordinate systems. Each coordinate system corresponds to a memory stage

¹⁵ The work of this thesis does not implement a sophisticated out-of-core algorithm or external memory technique but a straight forward mapping applied to the data structure; examples of surveys in the literature that discuss such techniques include [121, 122] but none is discussed in this thesis.

as previously discussed in this section. Other coordinate systems are dealt with but for rendering purpose (e.g. screen coordinate).

1. texture image coordinate (tX , tY , tZ)
2. local volume coordinate ($localX$, $localY$, $localZ$) per level-of-detail
3. global geographical coordinate (x , y , z)

The mapping between coordinates takes place on the X-Y plane. No mapping is performed on the z -axis; a seismic trace is fully loaded to a 1D texture image location (tX , tY). This is because the trace length of raw seismic data is, usually, fixed across one dataset and relatively small, around 200 to 2000 samples per trace; each sample is a 4-byte floating point.

As illustrated in Figure 5.10, seismic traces of a survey are not perfectly gridded but regularly distributed. In other words, distances between traces are very close to being equal but not exactly equal. Thus, they can be gridded by the following straightforward method. The survey is divided into *GIS cells* of equal areas, each containing one trace. Each *GIS cell* corresponds to a 2D location in the local volume. The slight alteration as a result of the gridding (from a non-perfect grid in the real-world to a perfect grid in the local volume) is accepted by the industry; the average deviation on the datasets that the author had access to was less than 4% of the separation of the grid points. The dimension of the *GIS cells* (or grid size) is calculated based on the raw data (traces) distribution; data referring to the features is neglected for this calculation. Using metadata, the dimension of the selected seismic survey is indicated as follows: *number of in-lines* \times *number of cross-lines* \times *samples per trace* (trace length); see Figure 5.11. Considering the geographical width and depth (area) of the selected survey, the *GIS cell* dimension (grid size) is computed as follows, using the ceiling function to round up:

$$GIS\ Cell\ Width = \left\lceil \frac{Region\ Width}{Cross\ Lines} \right\rceil \quad (5.1)$$

$$GIS\ Cell\ Depth = \left\lceil \frac{Region\ Depth}{In\ Lines} \right\rceil \quad (5.2)$$

The local volume now becomes the highest resolution level (*LOD0*). To generate levels of detail (*LODs*), a decimation-based technique by downsampling is implemented. Looking at a mesh-based rendering, *decimation* (or *mesh simplification*) is a process that reduces the number of polygons while preserving the object appearance

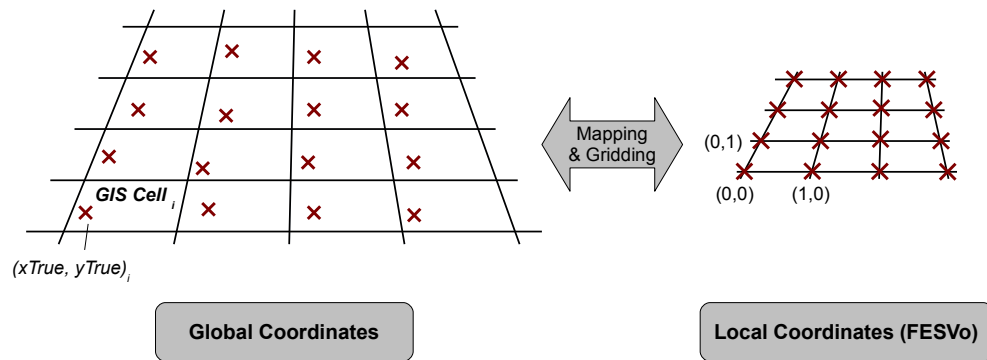


Figure 5.10: Seismic traces (illustrated as \mathbf{x}) are not perfectly gridded, but the region can be divided into “GIS cells” where each cell contains a trace. Each GIS cell can then be mapped to a local point in the FESVo.

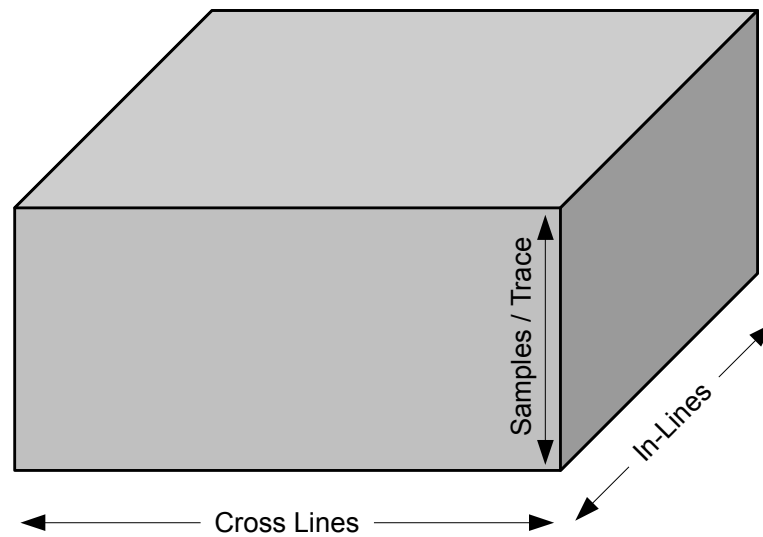


Figure 5.11: The dimension of a seismic survey is *number of in-lines* \times *number of cross-lines* \times *samples per trace* (trace length).

[123]. Decimation is one method for generating LODs [124]. The concept can be applied to seismic traces but in a slightly different way. As implemented in seismic applications like Petrel [108], decimation by downsampling is applied to reduce the number of traces in the rendered cube in order to produce a lower resolution version¹⁶; the implementation in this thesis adopts this technique. Traces are reduced on both axes (x and y) by a factor of 2^n where n is the LOD number. For example, at LOD 3, a trace every 8th in-line and cross-line is fetched. Thus, a low resolution image can be generated through direct mapping to cells at *LOD0*. Only one resolution version (the highest) of the dataset exists and real-time mapping is performed for lower resolution levels.

The mapping of (x, y) feature points is performed similarly to the case of raw seismic data, as explained previously. However, unlike a trace which is fully loaded to the texture buffer, a feature point's time (TWTT), as discussed in Section 5.3.2.4, is mapped to the z -axis as follows. First, the time range of this dataset, which is recorded in the metadata (the **SURVEY** table) is recalled. Using this range, a feature point's time is normalized into an integer index on a 1D array of length equal to the trace length of the associated raw seismic data. This is illustrated in the equation below, which uses a floor function $+0.5$ to perform a half round up to the nearest integer.

$$index = \left\lfloor \frac{twtt_max - twtt}{twtt_max - twtt_min} \times NumberOfSamples + 0.5 \right\rfloor \quad (5.3)$$

The feature texture buffer stores the feature ID (discussed in Section 5.3.2.4) at the calculated coordinate $(tX, tY, index)$. The above normalization method assumes that maximum time is mapped to index zero and minimum time is, thus, mapped to the index at the end of the array. This is because geoscientists conventionally measure time of returned waves that have traveled downwards as a negative value.

5.4.1.2 Data Lookup

A GIS cell holds a subsurface dataset under a rectangular area (e.g. 12×12 metres squared) of the real world. In a 3D texture image world, this is mapped to a single 1D dataset. To search inside a GIS cell in the database, two modes are implemented: (1) general discovery mode and (2) specific cached mode; the latter is a default choice.

¹⁶ Seismic applications, like Petrel, usually adopt more than one LOD generation technique, one of which is decimation (simplification). Petrel also adopts the *Octreemizer* technique by Plate et. al. [8] which stores seismic data in multi-resolution bricks inside its internal format for fast data access.

In the first mode, there is no knowledge in advance about the exact coordinates of the data units; thus it is a *discovery* mode. To query the database for a dataset which lies on a GIS cell, the data loader explicitly queries every possible location (x,y) with a minimum step (e.g. 1 meter). The reason why explicit values of x and y are provided in the query is to perform point-to-point queries in order to avoid a full table scan by the database. Due to the massive size of seismic datasets, it is always desirable to avoid a full table scan which leads to a decrease in performance.

In the specific mode, as presented in the third step in Section 5.4.1, the data loader pre-scans the tables for the required region and dataset source (i.e. seismic survey and interpretations) and then caches all the (x,y) coordinates, using a sorted table. During the loading process, the 3D texture image coordinate which needs to be loaded is mapped to a geographical coordinate: $(x_{Initial}, y_{Initial})$. As all valid data coordinates are cached on the client, the loader can efficiently look for a point (x_{True}, y_{True}) which lies on the location of the current GIS cell. Having a valid and explicit coordinate, a point-to-point query is executed per required GIS cell. Overall, this mode performs at a complexity of $O(k)$, where k is the number of data units returned; this is because point-to-point queries with a hash-based indexing allow the database to compute row locations on the fly as explained in Sections 5.3.1 and 5.3.2.5. The specific mode, however, requires a cache size that fits all coordinates of required sources during an application session.

5.4.1.3 Data Filtering

The data loader can accept a filtering criteria from the user interface. It is implemented as an object which its parameters represent the filtering criteria set (optionally) by the user and is passed to the **Data Loader** of the FESVo. The prototype implementation allows the following parameters to be optionally set in the filtering object: *feature ID*, *created by user*, *assessed by user* and *assessed confidence level range*. The loader utilises the provenance information to address this filtering criteria. It prepares and then returns a filtered dataset to the user interface to be rendered by the rendering engine. This filtering object was designed with intent to be extended in future to export and import user defined criteria using a provenance language represented in XML format; this is further discussed in Section 8.2.2.1.

5.4.2 Data Caching

The use of the **Data Cache** object of the FESVo (illustrated in Figure 5.9) was implicitly discussed above. To recap, the cache in the FESVo is used to store the following.

1. Metadata of selected data sources
2. Survey traces (raw data)
3. Feature points (interpretations)
4. Data coordinates in the specific mode

The FESVo cache minimizes access to the database, since a client-server architecture is implemented. Also, it allows users to fully interact with data in the case of a connection loss.

5.4.3 Capturing Interpretations

The FESVo captures users' visual interpretations along with provenance metadata via the **Interpretation Capturer** object as illustrated in Figure 5.9. The interpretation capturer offers predefined *interpretation methods* to users to add, amend or assess an interpretation.

Using the graphical user interface of the prototype application, users can amend or assess a previously interpreted horizon by drawing a polygon on its surface to select a region of interest; screenshots are illustrated in the next chapter which is dedicated for a walkthrough on the prototype application. Selecting one of the predefined interpretation methods, users can apply their modifications or assessment to the selected region. The following explains the steps taken by the prototype application in capturing users' interpretations to the data structure. Note that the first two steps are under the role of the user interface and rendering engine, but for the purpose of a comprehensive explanation they are mentioned in this section of the FESVo.

1. User creates a polygon on a horizon surrounding an area of interest to be modified or assessed.
2. The rendering engine reverse maps the polygon vertices to the 3D volume coordinate, and passes them to the interpretation capturer of the FESVo.
3. The interpretation capturer of the FESVo reverse maps the above vertices to global coordinates.

4. The interpretation capturer of the FESVo computes global points that are inside the polygon and lies on the selected horizon. These are passed to the database communicator.
5. The database communicator generates SQL queries based on the selected method (e.g. assessment). These include the above computed points with parameters (e.g. *seismic survey ID*, *user ID*, and *timestamp*), and are sent to the database.

This reverse coordinate mapping is achieved using OpenGL's `gluUnProject()`¹⁷ function to determine a world coordinate, then using the FESVo parameters to determine the corresponding local volume coordinate and global geographical coordinate. As illustrated in Figure 5.12, a user's 2D mouse coordinate, i.e. of a polygon point on a horizon, is first mapped to a 3D window, or cursor, coordinate by (1) correcting (reversing) the *y* value in respect to the window size and (2) computing the *z* value from the frame buffer at the corresponding pixel considering its depth. For this to function correctly, the shader must ensure a correct fragment depth is set such that unseen objects do not block seen objects. This window coordinate is then fed into `gluUnProject()` function to map it to an OpenGL world coordinate. Using seismic cube's position and dimension, a world coordinate is mapped to the 3D image coordinate which represents the rendered seismic cube. Using FESVo parameters, a mapping to a coordinate of the cached local volume and then to the global coordinate of the original data are performed; these coordinates were described in Section 5.4.1.1.

The generated queries depend on the selected method. In specific cases, such as a user discarding another user's interpretation, a single query can delegate this action to the database. However, in general cases, explicit feature points are required to be embedded in the generated query to perform the required action. In either case, the steps in recording interpretations are the same, as previously described in 5.3.3. Users' contributions are recorded in the **INTERPRETATION** table. Every set of interpretation, including assessment, is linked to its referenced interpretation as well as to the survey (raw data). Thus, provenance of interpretations are preserved.

5.4.4 Optimization

The prototype implementation of this thesis adopts standard optimization techniques; the following text highlights three of these. Major optimizations are left for future work.

¹⁷<https://www.opengl.org/sdk/docs/man2/xhtml/gluUnProject.xml>

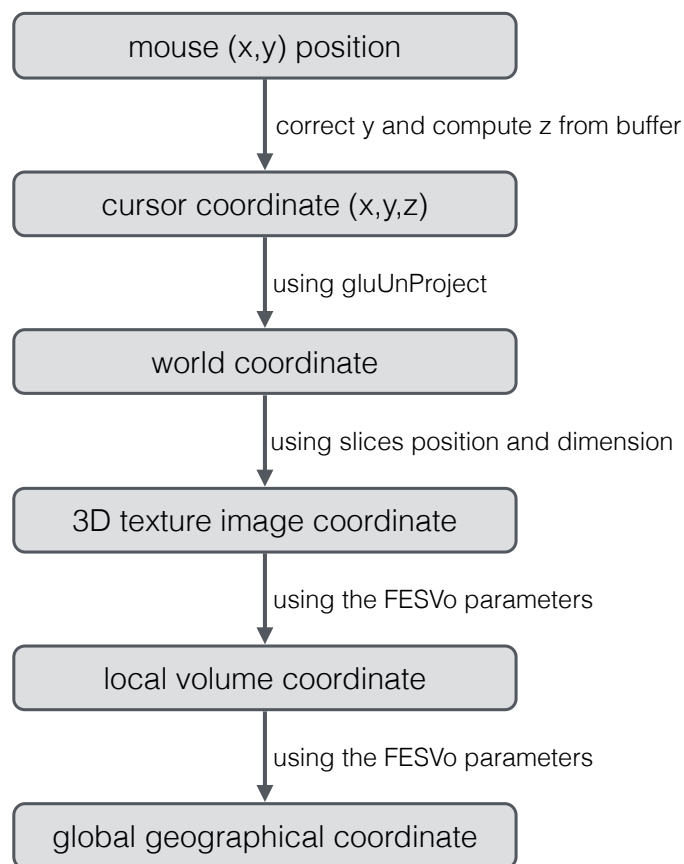


Figure 5.12: The graph shows the mapping sequence of coordinate systems to bring users' interpretations from a user's screen to the fine-grained data repository.

1. **Concurrent Loading:** The data loader utilises the parallelism capability of the database by running multiple threads to concurrently fetch and update data. The optimum number of threads to run depends on the capability of the client's machine and on the database host. This is discussed in Section 7.4, along with results on the gain from running concurrent loading threads in terms of performance. Figure 5.1 shows four parallel connections (threads) between the database and the FESVo since it is a default option in the prototype application.
2. **Initial Area Scanning:** This technique was discussed in Section 5.4.1.2, which called it a *specific mode* data lookup. In this technique, desired datasets are pre-scanned to cache their coordinates in order to run point-to-point queries; refer back to that section for more details.
3. **Background Fetching:** This is a basic out-of-core (or external memory) technique which prefetches more data units to enhance the resolution of an initial lower resolution version, usually used in a low-speed connection (e.g. broadband). The coordinate mapping and LOD described in Section 5.4.1.1 allows the implementation of this technique.

5.5 User Interface and Rendering Engine

The user interface was implemented on the Java standard edition (SE) platform, as is the FESVo. For graphics rendering, Java OpenGL (JOGL¹⁸) library [125] was used, which is a Java binding to OpenGL¹⁹ [126, 127]. Appendix D presents background information on basic rendering techniques adopted by the prototype application of this thesis.

5.5.1 Rendering

The current implementation adopts a back-to-front *textured slice mapping* rendering technique [128] along with a shader program, using OpenGL Shader Language (GLSL) [129]. Texture mapping is a standard technique for rendering seismic imaging data [80]. In this work, the technique is also used to render geological features due to its

¹⁸<http://jogamp.org/jogl/>

¹⁹<http://www.opengl.org/>

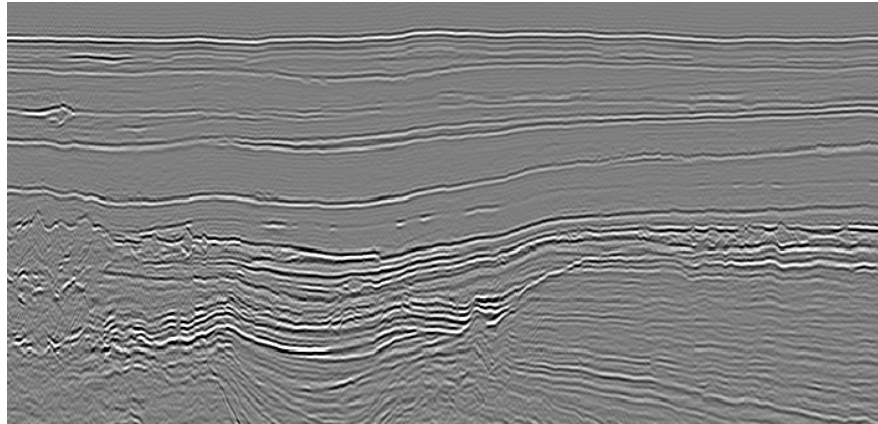


Figure 5.13: A slice-based visualization of a 6.3GB seismic dataset. In this example, each slice is of size 4.35MB, collected through 1448 point-to-point queries over concurrent threads.

simplicity; seismic data for interpretation does not have to be rendered photorealistically, as would be provided by ray tracing for example. Thus, two texture objects exist at any time: one for seismic raw data (traces) and the other for all geological features.

To render seismic raw data, the shader program is provided with the texture data along with the minimum and maximum values of the seismic amplitudes; these are read from the **SURVEY** table. A lookup table [130] is implemented in the shader to render a proper gray scale seismic image via normalizing the amplitude value on the texture data at each voxel; see Figure 5.13 for an example.

For geological features, a feature ID, minimum height, and maximum height are passed to the shader program. Horizons are coloured, based on the local height of each, using a lookup table with a palette function. The prototype application attempted to follow a standard look similar to commercial seismic imaging applications (e.g. Petrel). This to familiarise the targeted users when testing the prototype application. Figure 5.14 shows the similarity with Petrel by rendering the same horizons.

5.5.2 Tooltip—Provenance Information Per Point

The user interface of the prototype application of this thesis provides users with a tooltip feature that shows metadata about the point being hovered over. Information such as its local and global geographical locations appear to the users. In addition, if the point belong to a geological horizon, for example, users get information about who created the interpretation that this point belongs to and if any assessment exists. The retrieval of such information follows the coordinate mapping sequence illustrated

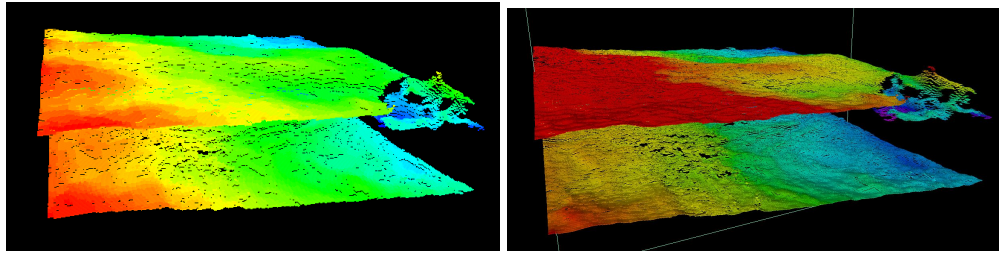


Figure 5.14: Left hand figure shows two horizons fetched from the database and rendered using direct volume rendering; this is using the prototype application of this thesis. Right hand figure shows the same horizons rendered on Petrel seismic application. The rendering of this thesis application used a similar, but not exactly the same, colour scale to Petrel's.

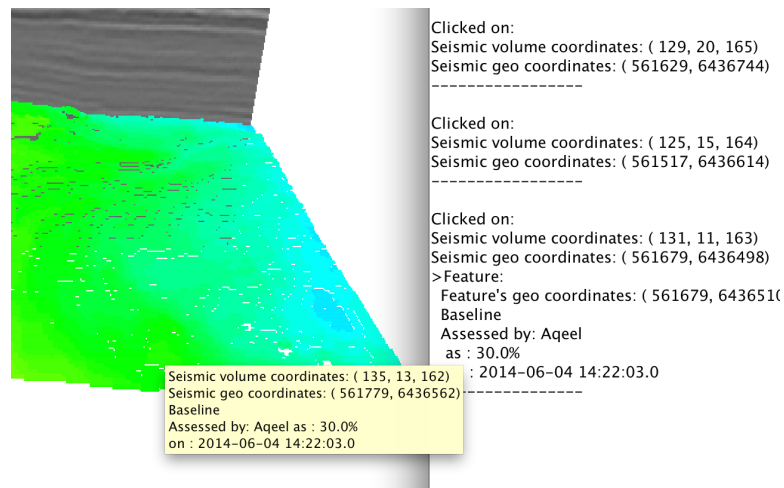


Figure 5.15: The tooltip feature of the prototype application provides users with information on a point level, such as location, creator and assessment.

in Figure 5.12. If required provenance information was loaded into the FESVo, the mapping stops at the local volume coordinate. Figure 5.15 shows an example of the information provided by the tooltip feature. The capturing of interpretations and meta-data through the provenance model allows the application to trace the origin of a point and its related information, and thus can display these to users.

5.5.3 Filtering and Highlighting

In addition to the provided tooltip, users can filter or highlight the visualized data according to specified criteria. Users may, for example, wish to filter out regions of horizons that were assessed with a low confidence level or only load regions that were interpreted or assessed by a certain user. The user interface creates a filtering object

with parameters representing user defined criteria and pass it to the data loader along with the data loading request, as described in Section 5.4.1.3. Users may also highlight those criteria on the visual scene. As with the tooltip feature, the captured and linked provenance information allowed the implementation of these features. Screenshots are provided in the next chapter as part of the walkthrough.

5.6 Chapter Summary

The chapter presented an implementation of the proposed provenance-enabled interpretation pipeline in Chapter 3 for the domain of seismic imaging interpretation. It partially addressed the third objective of this thesis; Chapter 7 completes the addressing of this objective. The implementation resulted in a proof-of-concept prototype architecture. The architecture is server-client, consisting of three-loosely coupled components: (1) fine-grained data structure implemented in a database, (2) FESVo (an intermediate stage) and (3) rendering and user interface.

First, the provenance model was further extended to accommodate geological data by introducing the *geo* namespace, to include entities like *geo:trace* and *geo:horizon*. Then, this provenance model was translated into a relational database model, implemented on a central Teradata database (server side). Assumptions were put on the behaviour of the model to achieve its implementation as database tables. The implemented tables and underlying data were explained in the chapter. Information of a seismic surveys (raw data) is recorded in the **SURVEY** table. Traces of a survey are restructured into tuples (rows), one for each seismic trace, and recorded in the **TRACE** table. Interpretation activities are recorded in the **INTERPRETATION** table, each of its record is linked to a set of feature points recorded in the **FEATURE POINT** table representing the interpreted features. The data in this model were referenced, and thus indexed, on their global geographical coordinates. Thus, the model accumulated users' interpretations and assessments of features as fine-grained data linked together as well as with the raw data. Using a multi-interpretation case example, a mapping between the provenance model and the implemented relational database model was presented, along with a pseudo query on how to track history of interpretation. In practice and for a proof-of-concept purpose, the data structure was initially loaded with seismic surveys using the *SEG-Y loader* of the SRDS project of Irving et al. [23, 24]. Also, the structure was initially loaded with some user interpreted horizons, which were exported from *Petrel* as a cloud of (x,y,z) points.

The FESVo, historically short for *feature-embedded spatial volume*, combined and implemented part of the visualization pipeline (excluding rendering) and part of the proposed interpretation pipeline; the user interface and rendering engine completed the implementation of the two pipelines. In the prototype, the FESVo resided on the client side; but it can also be implemented on the server side (possible future work). It had the role of loading data, caching data and capturing users' interpretations, between the central database and user interface. It mapped between different coordinate systems, from a global geographical coordinate (as recorded in the data model) to a local volume coordinate (recorded in the FESVo) and then all the way to a user screen's coordinate for loading and rendering a dataset, and vice versa when capturing a user interpretation. On loading, data is gridded from a global (geographical) coordinate to a local volume coordinate by dividing the survey area into global cells of equal areas. This is maintained in the FESVo and utilised for the reverse mapping to later capture users' interpretations. A filtering mechanism was built in the data loader, taking advantage of the provenance information to select a dataset of interest based on a user-defined criteria.

When users attempt to amend or assess a horizon for example, the prototype implementation allows users to draw a polygon on an area of interest. Vertices are then mapped to the local volume coordinate in the FESVo which then computes points inside the polygon that are included in the selected horizon. The FESVo then generates a query to update these points based on the selected interpretation method.

The rendering of data was based on a texture slice mapping technique. Utilising the reverse mapping explained above, users are provided with a tooltip that provides provenance information of the points being hovered over. A walkthrough of selected scenarios using the prototype application is demonstrated in the next chapter.

Chapter 6

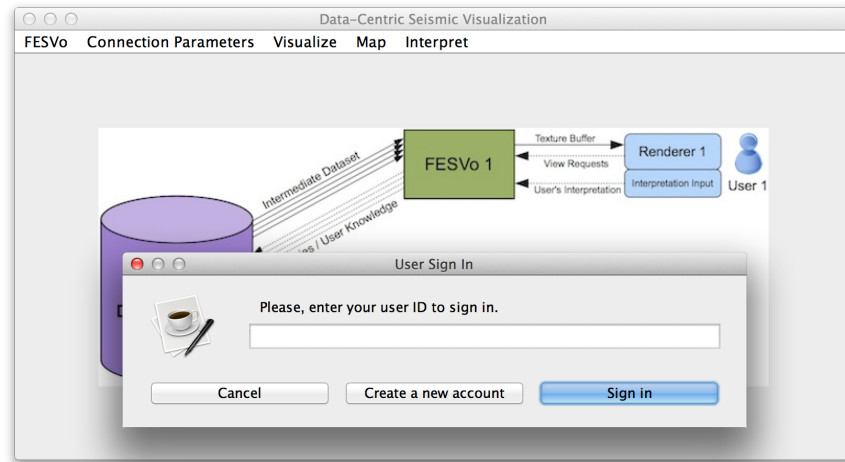
A Walk-through of the Implemented Application

This chapter only presents a walk-through of the implemented application aided by screenshots. The chapter is divided into sections, each demonstrates a scenario. Some of what is demonstrated in this chapter is included in a screen recording video [31] which was published on the Web by the author of this thesis; this can be accessed via the following link: <http://youtu.be/fyBQYEEoHiQ>. The seismic data used in this walk-through has an amplitude range from -58 to $+77$ rendered on a greyscale, varying from black at the lowest value to white at the highest value. Its z-domain is in time scale ranging from $2ms$ to $-1002ms$.

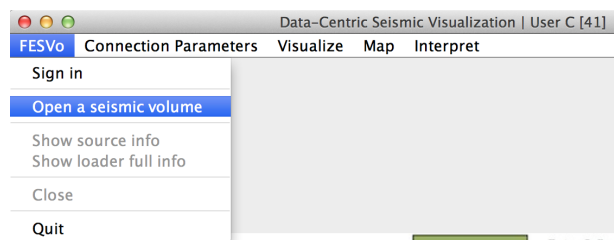
6.1 Basic Functionalities

This section demonstrates basic functionalities of the prototype application, which include loading and visualizing a seismic survey dataset and an interpreted horizon.

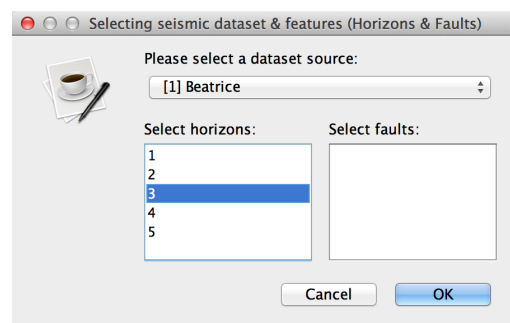
1. Launching the application and signing in.



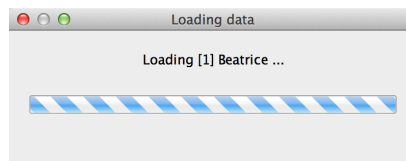
2. Opening a seismic volume, by clicking on **FESVo** → **Open a seismic volume** from the menu.



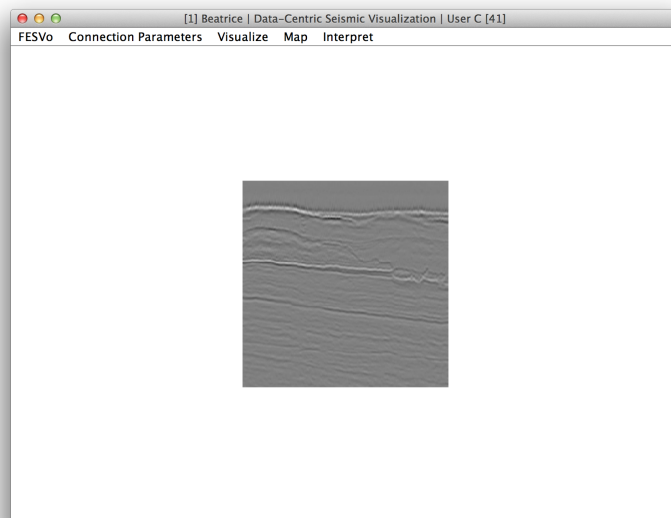
3. Selecting a seismic survey (volume) and available feature(s). In this scenario, the volume *Beatrice* and horizon number 3 are selected.



4. Loading data; the application loads data from the database.

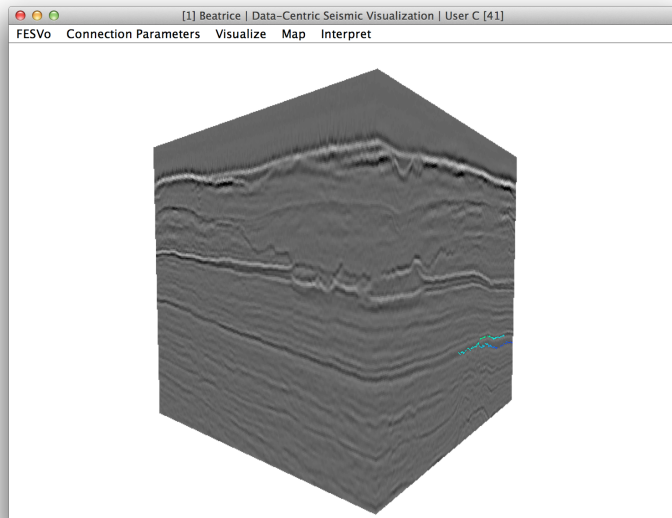


5. Users initially view the data as a volumetric cube.

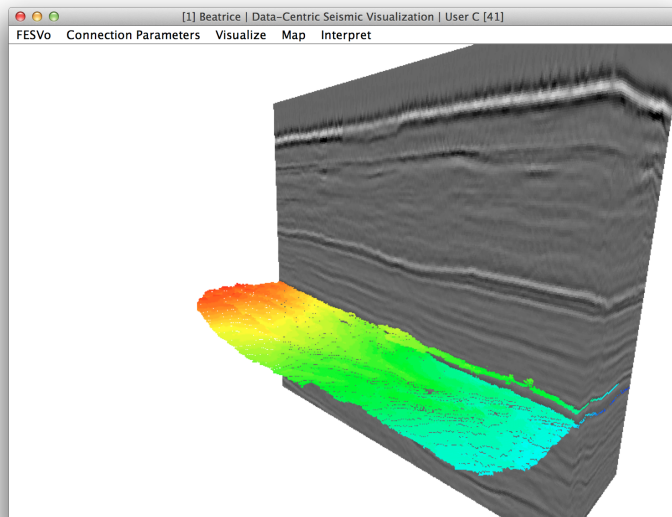


6. Users can:

- zoom in and out by pressing on + and - on the keyboard, respectively.
- rotate the volume using the keyboard's arrows.
- adjust the contrast by adjusting the minimum and maximum amplitudes, of seismic samples, to be calculated by the shader program, by clicking on **Visualize** → **Show amplitude control** from the menu.



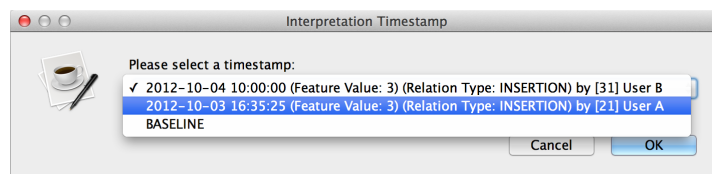
7. Users can create a cross section through the seismic volume to show the internal interpreted horizon; pressing < and > on the keyboard opens and closes the intersection, respectively.



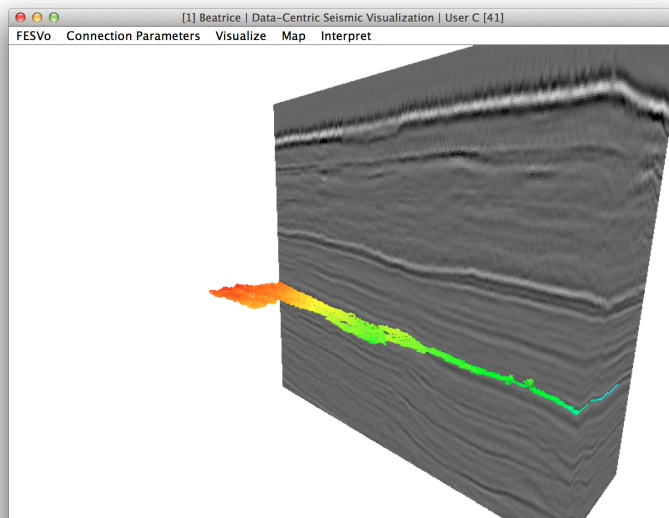
6.2 History Tracking

This section shows how a user can track the history of interpretation of a geological horizon.

1. Users can navigate through the history of interpretations, by clicking on **Interpret** → **History** from the menu then selecting a timestamp. As shown in the screenshot below, User B added some interpretations to the work of User A and to a previously available interpretation (named BASELINE in this scenario).



2. By selecting the timestamp of User A, the user can visualize how the horizon was interpreted before the work of User B was added.

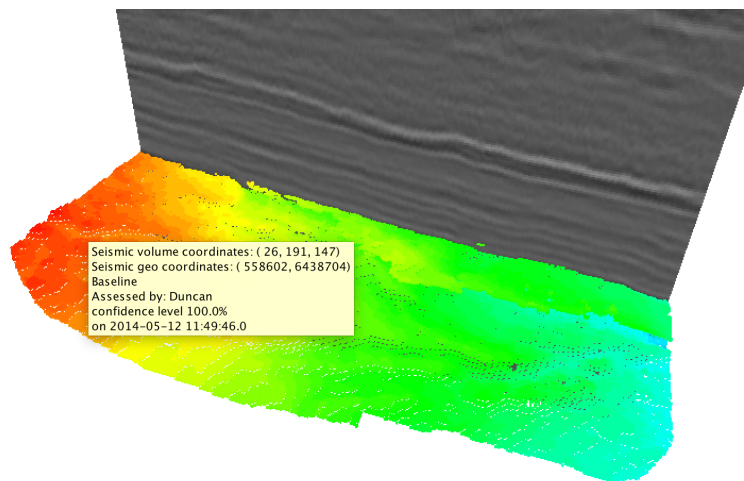


It is worth mentioning that a network graph demonstrating the historical timestamps would provide an easier representation of the provenance to the user. However, for a proof-of-concept purpose, a pull-down menu was implemented instead. A network graph representation is considered as a future work.

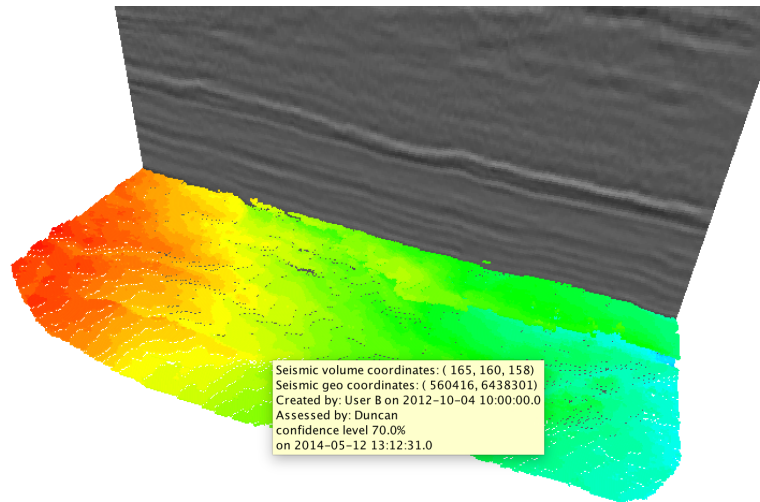
6.3 Tooltip

This section presents a scenario where a user attempts to get an overview about the loaded geological horizon in the previous sections. Using the tooltip feature, metadata about the point being hovered over is provided as shown below.

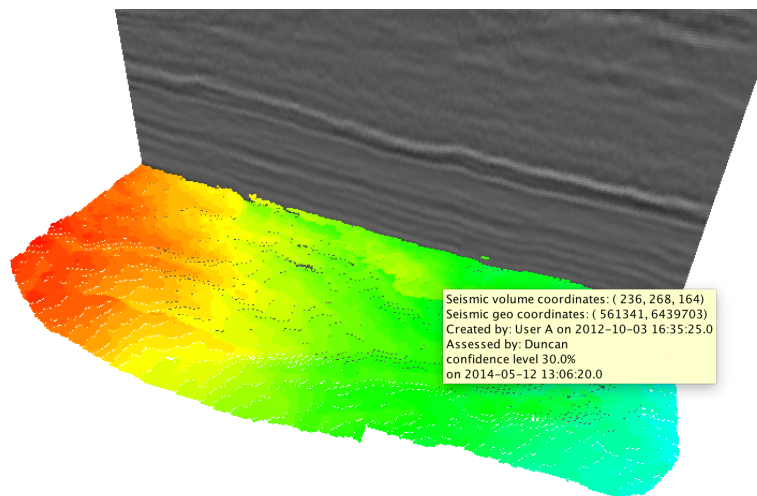
1. Below screenshot shows the following information about the point being hovered over. First, the user gets its local (FESVo) and global geographical location. In addition, the information shows that the point belongs to a *baseline* (initial interpretation). Also, it belongs to a region that was assessed by a user named *Duncan* with a confidence level of 100% on a certain date and time. By clicking on this point, this information is printed on an information panel, which is not shown below.



2. By hovering around, the user gets different information per point. Below screenshot shows that the point being hovered over belongs to a region that was interpreted by *User B* in 2012. It also belongs to a region that was assessed by *Duncan* with a confidence level of 70% at a different time in 2014.



- Below screenshot shows that the point being hovered over belongs to a region that was interpreted by *User A* in 2012. It also belongs to a region that was assessed by *Duncan* with a confidence level of 30% at a different time in 2014.

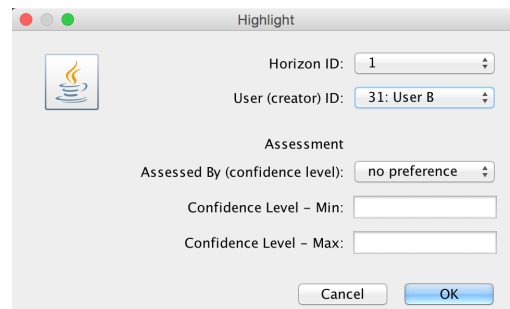


6.4 Highlighting and Filtering

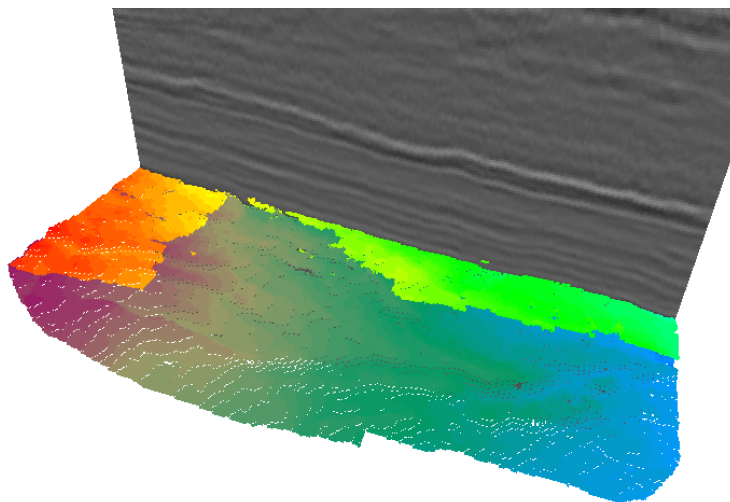
The tooltip presented in the previous scenario provided information on a point level. In this section, the user attempts to highlight and filter the data based on certain criteria.

- To highlight a region, the user clicks on **Visualize** → **Highlight** from the menu.

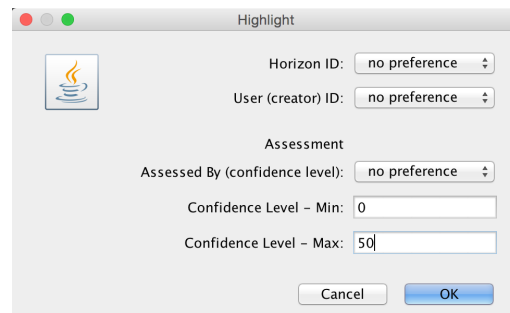
2. The user then enters the highlighting criteria. In this scenario, the user attempts to highlight regions that were interpreted by *User B*, with no preferences to assessment criteria.



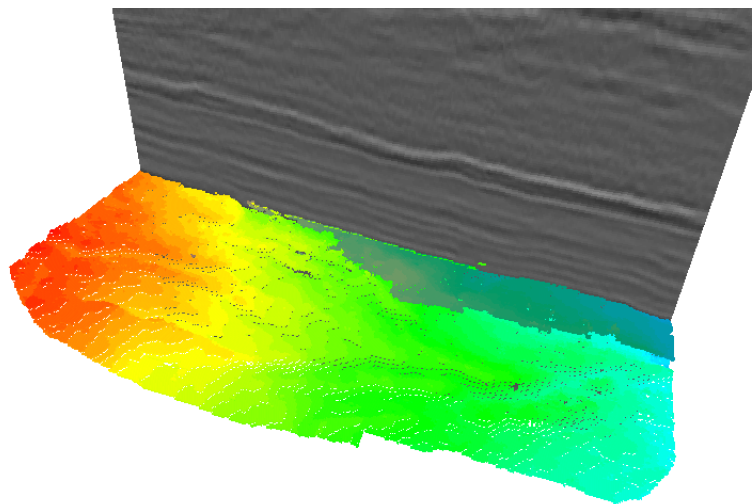
3. Regions of the loaded horizon that were interpreted by **User B** gets highlighted (i.e. shaded) as shown below.



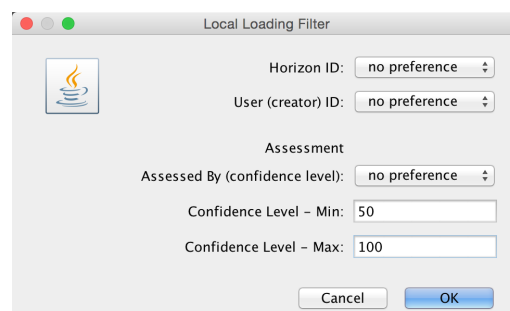
4. After clearing the above highlighting, the user now attempts to highlight regions that were assessed by any user with a low confidence level (0 to 50%), with no preferences to the interpreted user.



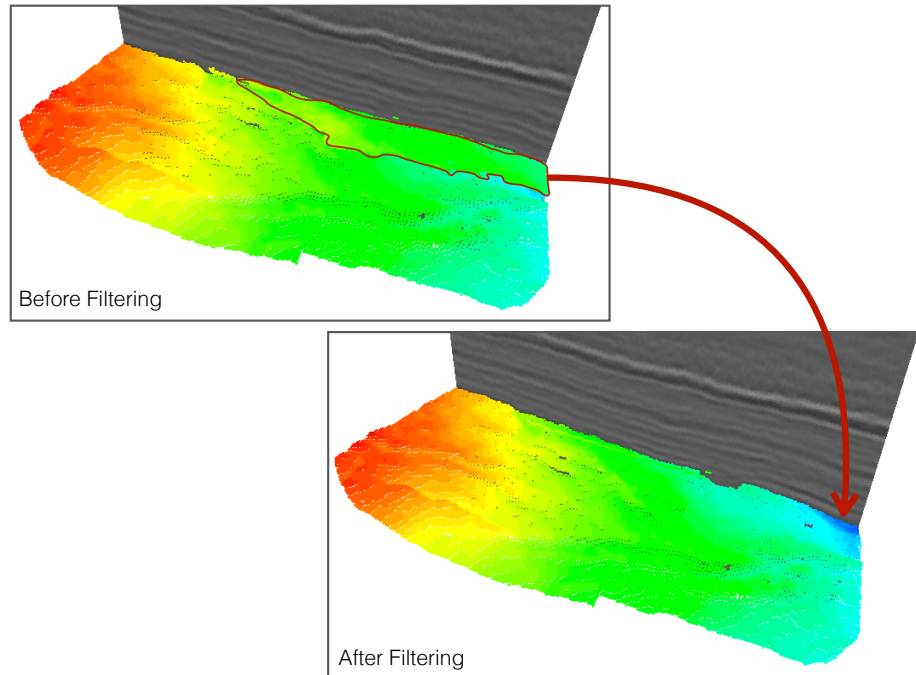
5. Regions of the loaded horizon that were assessed with a confidence level of less than 50% gets highlighted (i.e. shaded) as shown below.



6. Now, the user attempts to filter out this region. This can be done by the filtering feature. To filter, the user clicks on **Visualize** → **Filter** from the menu.
7. To filter out regions with a confidence level of less than 50%, the user needs to set the filtering criteria to load the complemented region, i.e. assessed with a confidence level of more than 50%.



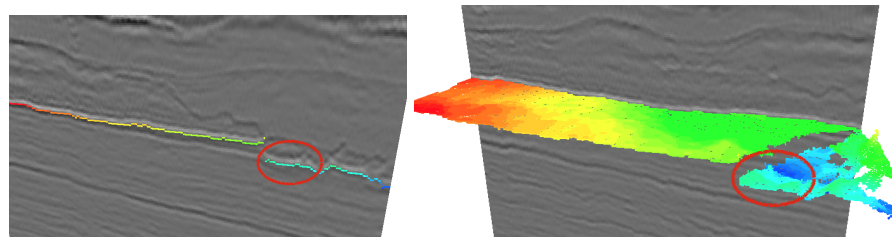
8. Regions of the loaded horizon that were assessed with a confidence level of more than 50% are loaded to the visual scene as illustrated below. To clarify the illustration, the region with low assessment value was located on top of another region; the latter appeared after filtering.



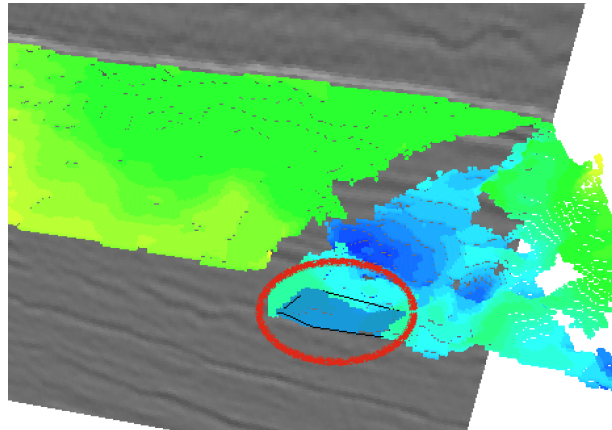
6.5 Amendment

This section presents a scenario where a user (*User X*) is attempting to slightly amend an interpretation by statically correcting its time value. In this scenario, the same seismic volume is loaded but with a different horizon than the previous sections.

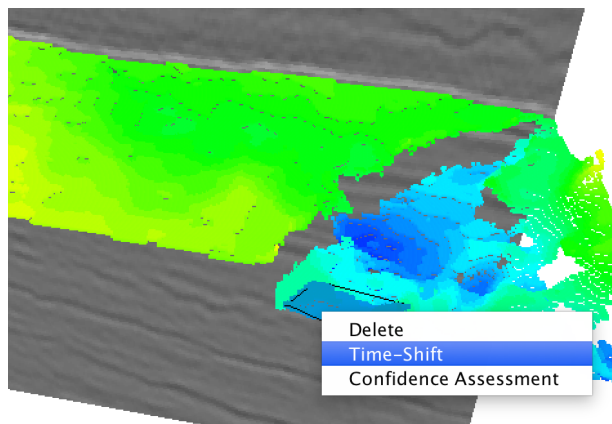
1. The user decides to amend the circled region in below screenshots; the circle is for illustration purpose. The right hand side screenshot is just an open cross section of the same dataset.



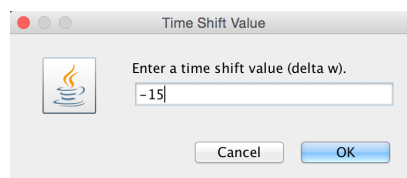
2. The user needs to draw a polygon on the region of interest by first clicking on **Interpret** → **Activate Feature Area Selection** from the menu. The polygon gets shaded automatically as illustrated below.



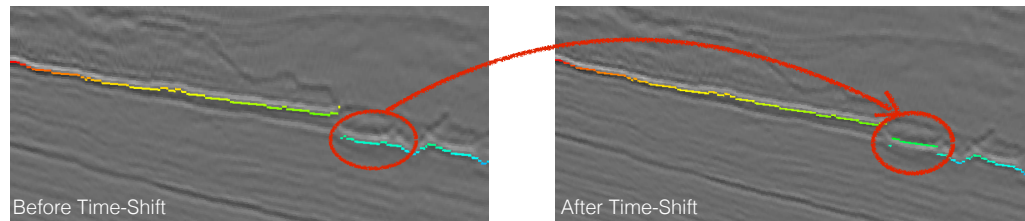
3. By clicking on the polygon using the right mouse button, the user selects **Time-Shift** from the popup menu.



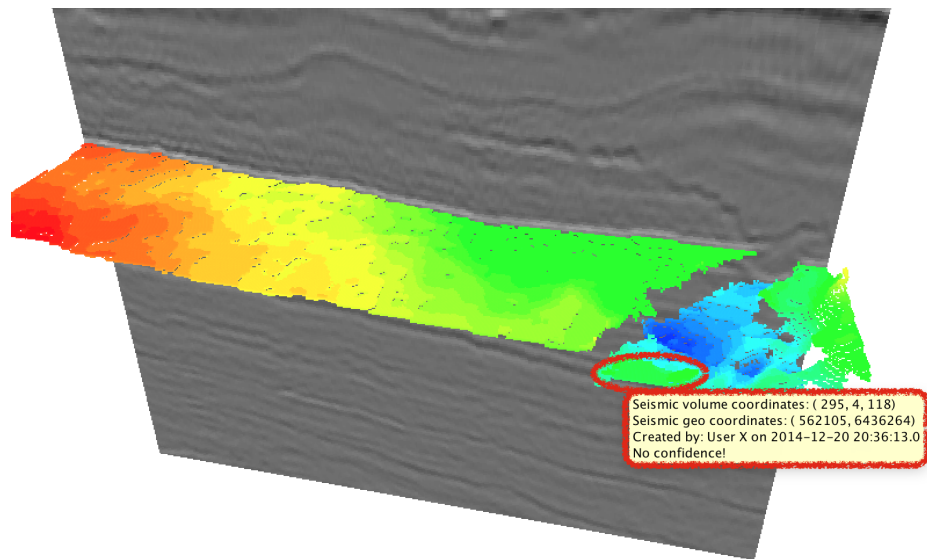
4. The user then enters a value in milliseconds (ms) to shift the user created polygon.



5. The user created polygon is then shift as illustrated below.



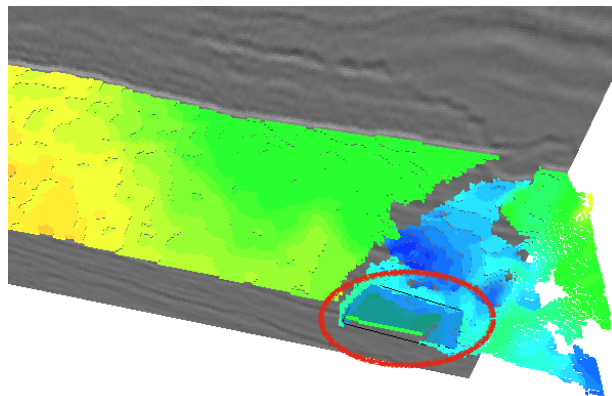
6. By hovering over the shifted area, the information provided shows that it was created by *User X*, who made the time-shift of this scenario.



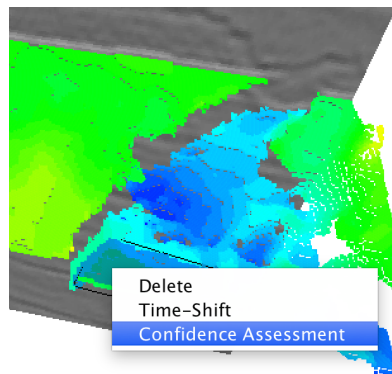
6.6 Assessment

This section presents a scenario where *User X* from the previous section is attempting to assess his/her interpretation.

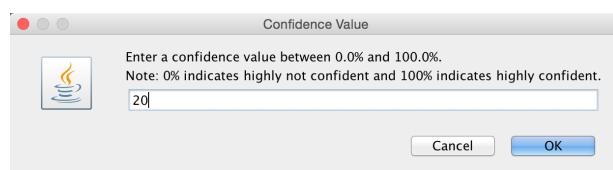
1. To assess a region, the user draw a polygon over a region of interest in the same way as presented in the previous section. In this scenario, the user draw a polygon over the region that was time-shifted as illustrated below.



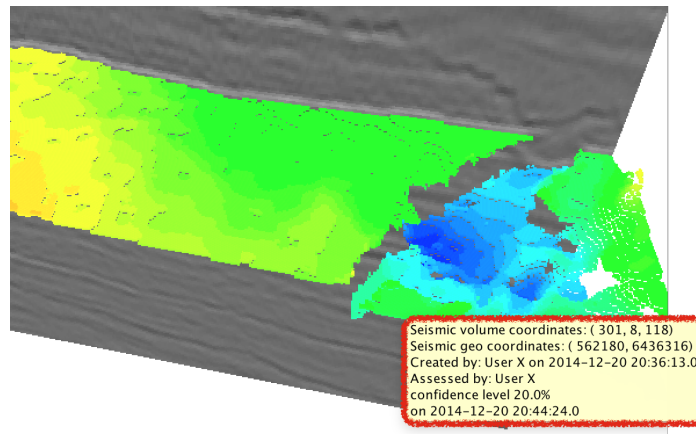
2. By clicking on the polygon using the right mouse button, the user selects **Confidence Assessment** from the popup menu.



3. The user then enters a confidence value between 0% and 100%, 20% in this case.



4. Now, by hovering over the region, the information provided shows that it was created by *User X* and assessed by *User X* with a confidence level of 20%.



Chapter 7

Evaluations and Results

7.1 Chapter Introduction

The implementation discussed in Chapter 5 resulted in a prototype application of a proposed architecture as a proof of concept. Its main purpose was to evaluate and test the proposed interpretation pipeline and provenance model for seismic imaging interpretation. The architecture allows multi-user interpretations and annotations to be captured and accumulated into a fine-grained provenance model integrated with the raw data (seismic surveys). This was demonstrated through a walk-through of the implemented application in Chapter 6. As the application’s purpose in this PhD work is a proof of concept, only a limited range of the potential functionality of the seismic application was implemented due to time limitations. Section 8.2 discusses desirable extensions of the functionality for future work.

It is challenging to evaluate the ultimate aim of this research in real-case scenarios over many years. As described by Groth and Streefkerk [74], the value of history tracking is “accretive in nature”, meaning that the system needs to be evaluated by users for a long period to fully derive its benefits. This is challenging due to the time constraints of this research study (PhD). Another challenge rises from the sensitivity and confidentiality of data in the oil and gas industry. For these reasons the evaluation was based on the methods established in the thesis of Lidal [131], which introduced the concept of storytelling into the geological interpretation workflow. The system was first evaluated by geologists working in a collaborative environment, then by conducting meetings with domain experts in the oil and gas industry. This thesis adds a third evaluation, which is performance related.

First, the application’s functionalities were tested by geoscientists from the School

of Earth, Atmospheric and Environmental Sciences (SEAES) of The University of Manchester. This is to ensure that the prototype application performed correctly. This is presented in Section 7.2. Second, a conceptual evaluation was conducted through interviews with domain experts. The concept of the research was presented and the prototype application was demonstrated to them. In return, feedback was collected through semi-structured interviews. This is presented in Section 7.3. These two parts are similar to the theme of Lidal's [131] evaluation, as mentioned above. The third part of the evaluation of this thesis is a test of the aspects of the software that are indirectly related to its functional design, e.g. performance metrics, interactivity, scalability. Performance measures were conducted firstly using commodity desktop and laptop computers and then using a dedicated parallel processing database cluster. This is presented in Section 7.4.

7.2 Functionality Test

Six geoscientists from the School of Earth, Atmospheric and Environmental Sciences (SEAES) of The University of Manchester participated in a case study to test the main functionalities of the implemented application. The participants were divided into three groups, i.e. two in each group. Each group participated in one evaluation session. An *evaluation session* in this context is defined to be the period of time during which a number of users were concurrently participating in an evaluation task using different machines, not to be confused with the terms *user session* or its equivalent *application session* which were mentioned in Section 5.4.1. Thus, generally, an *evaluation session* could involve multiple *user sessions*. At each evaluation session, the participants were given an explanation of the system and the purpose of the case study. Then each of the two participants in an evaluation session was given access to the system, and they were guided to load the same dataset at the same time from the centrally located database.

First, all participating geoscientists confirmed that the dataset was rendered correctly; their commercial software, which was *Petrel*, was considered as a guideline. They were asked to confirm this since the data has been completely restructured compared with the original raw seismic dataset. Thus, this confirmation verifies the loading method described in Section 5.4.1.

The participants were then asked to perform the following selected tasks using the

Format	Size (MB)
Trace Samples	161
SEG-Y	200
Database Tuples	351

Table 7.1: The table shows the size of the dataset used in the case study in different formats. The dimension of the dataset was 401 (in-lines) \times 401 (cross-lines) \times 251 samples per trace. The data in the database was not compressed.

system. These cases are simple core tasks that may be a part of their usual interpretation workflow. These tasks were selected primarily for the purpose of demonstrating the functionality of the application, such that users' interpretations are captured through the proposed interpretation pipeline and their provenance is maintained by the proposed provenance model. Responses from the participants were obtained using a questionnaire (a sample is shown in Appendix A) as well as using the *think aloud* methodology. The latter is a software usability evaluation technique which allows the capture of the thoughts and feelings of the participants while testing the application [132]. Despite its strength in usability testing, two weaknesses of the *think aloud* methodology are that capturing users' thoughts might be misinterpreted and it lacks the accuracy of measuring qualitative aspects. For these reasons, the questionnaire was used as a supporting method.

The dataset used in this case study was very small (around 200MB in a SEG-Y format) compared to those used by geologists in actual academic or industrial projects (usually tens of gigabytes). The intention of this evaluation was to test the usability and functionality of the implemented system and thus the small dataset served this purpose and allowed the evaluation sessions to be performed within a reasonable time; scalability tests which handled gigabytes of data were later performed as will be discussed in Section 7.4.2. Table 7.1 shows the size of this dataset in different formats: (1) a samples-only raw file, (2) a SEG-Y file and (3) a fine-grained data structure in database tuples. The first format only contains trace samples without any headers, which is the imaging data being visualized. The dataset increases in size (around 75% compared to a SEG-Y file) when restructured according to the proposed data model and put into the database. This is an expected price of a fine-grained granularity. However, the size in the database could be reduced by applying a compression but might impact on the searching time; the thesis has not investigated this.

Each of the following subsections presents and discusses a task carried out by the participants. The discussion includes a definition of the task and some technical details.

The reader may refer to Chapter 6 for step-by-step example scenarios that cover to some extent the following tasks.

7.2.1 Task 1: Multi-User Horizon Time Shifting

In this task, one user was asked to adjust a horizon by shifting its two-way-travel time (TWTT). Graphically, this *time* is the *z* axis of the early-stage seismic imaging data; it is later converted into real depth. The process of time shifting a horizon requires the user to define a region of interest and a shift value (+/-) in milliseconds.

After setting these parameters, the application performs the time-shifting as follows (following the implementation presented in Chapter 5). A record of interpretation is created (in the interpretation table) with activity type `interpret:delete`. This record is linked to the original interpretation, the interpreted seismic survey and this user's identity. Then, all feature points lying within the user defined region are inserted into the feature point table with the identity of the newly created interpretation record. Next, another record of interpretation is created but with activity type `interpret:add`. This record is linked to the previously created interpretation record, the interpreted seismic survey and this user's identity. Then, the same feature points are inserted into the feature point table but with a shifted time (TWTT) value; they are tagged with the identity of the newly created interpretation record. Feature points are inserted in parallel threads. User amendments are saved while the original interpretation is also maintained centrally with the original dataset.

After the time-shift task was completed (taking around 2-4 seconds in terms of communications between the application and the database), the second user in this evaluation session (using a different user session from the first user) refreshed the application's view and was able to immediately visualize the changes on the horizon made by the first user. Figure 7.1 illustrates a similar view of this result.

In Figure 7.1, the original horizon, as shown in the left-hand screenshot, consisted of 160,300 points (each is stored as a row in the database). The process of time-shift, as shown in the right-hand screenshot, resulted in 1,560 points tagged as deleted (from the original interpretation) and the same number of points (1,560) tagged as inserted (forming the shifted area). Thus, a total number of 163,420 points represent this horizon and its provenance of changes in the database; this is a result of a fine-grained granularity model. For rendering, however, only 160,300 points are being retrieved and rendered after the time-shift. By contrast, with a coarse-grained (file-based) granularity, any small change to a feature object requires the regeneration of

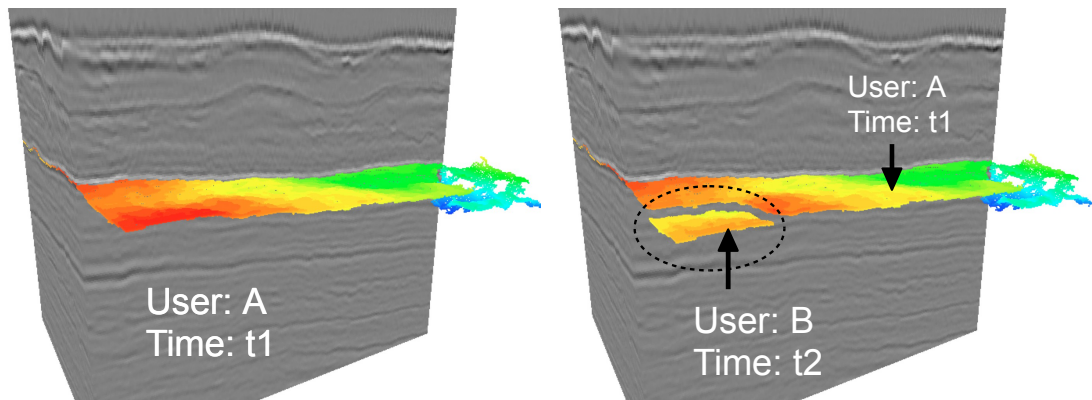


Figure 7.1: The screenshot on the left shows an interpreted horizon by user A at time t_1 . The screenshot on the right shows a partially shifted horizon by user B at time t_2 . The change (shift) takes place only for the affected points since the data model adopts a fine-grained provenance. The displayed seismic data has an amplitude range from -58 to $+77$ rendered on a greyscale, varying from black at the lowest value to white at the highest value. Its z-domain is in time scale ranging from $2ms$ to $-1002ms$.

the whole object in order to preserve the provenance

7.2.2 Task 2: Deletion of an Interpreted Object

This task assumes that two users have previously added to an existing interpretation of a horizon from a particular source. A senior (more expert) user has later visualized both interpretations and decided that one is more accurate than the other and therefore has wanted to delete the less accurate interpretation.

One user of an evaluation session (acting as an expert) was asked to select a session with a data *insertion* tag to delete. Technically, this is achieved as follows. As in Task 1, an `interpret:delete` type interpretation record is created with the expert user's identity and current timestamp, and linked to the seismic survey. Then, a single update query results in re-inserting the points of this session but tagged with the newly created interpretation record. Then, the FESVo is reloaded with the latest version of interpretations which includes the original (previously existing) version and the additional interpretation by the more accurate user.

As a timestamp is recorded when starting an interpretation record, users can go back in history to visualize earlier versions. In this task, the second user of this session was able to track the history of this horizon; this is illustrated in Figure 7.2.

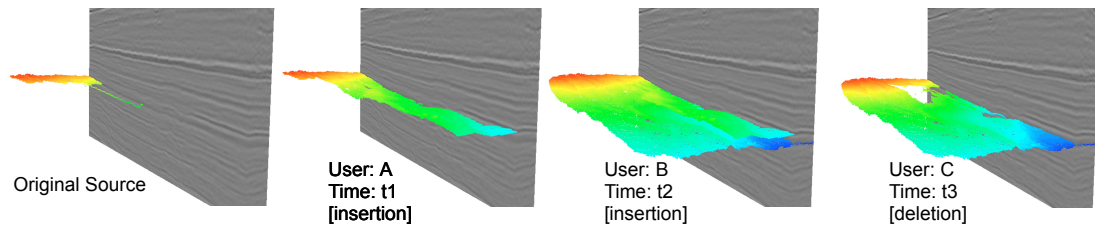


Figure 7.2: From left, the second screenshot shows some interpretation added to the original one (extreme left) by User A at time t_1 . The third screenshot shows a further contribution by User B at time t_2 . In the fourth screenshot, an expert user (User C) decided to delete the interpretation by User A due to, for example, lack of accuracy. The interpretation of User A is in fact not deleted but tagged as deleted. Users, therefore, can go back in history and visualize previous versions of interpretations. The displayed seismic data has an amplitude range from -58 to $+77$ rendered on a greyscale, varying from black at the lowest value to white at the highest value. Its z-domain is in time scale ranging from $2ms$ to $-1002ms$.

7.3 Interviewing Domain Experts

To conceptually evaluate the contribution of this research, interviews with three domain experts from the oil and gas industry were conducted. Interviewing domain experts for evaluation is a qualitative research method.

This thesis introduces the concept of provenance for visual interpretations, demonstrated via a prototype application for seismic imaging data. The previous section presented a functionality test of this prototype. This section presents a conceptual evaluation of this work by the industry of the applied domain. First, the data (i.e. evaluation feedback) collection method is described in Section 7.3.1. Then, the outcome of the interviews are presented in Section 7.3.2. Finally, Section 7.3.3 presents an analysis of the outcome.

7.3.1 Data Collection Method

The *data* here refers to data gathered in the evaluation and to any additional feedback obtained, not to the actual seismic data itself. For domain experts to evaluate the work of this thesis, a qualitative research approach was followed using *semi-structured interviews* to collect their feedback.

A semi-structured interview is guided with some questions, or a framework, but allows interviewees to naturally divert to bring up new ideas. Open-ended questions

were used to stimulate the conversation and allows in depth feedback from interviewees [112]; however, this comes at the price of difficult analysis. Interviewees were first given a quick introduction about the research project. Then, a demonstration of the prototype application was given; the same demonstration was given to all interviewees. It consisted of two parts. First, a seismic survey and a geological horizon, which was previously interpreted and assessed by multiple users, were loaded and visualized. The author (interviewer) demonstrated the tooltip feature, which retrieves provenance information. Also, the data filtering feature was demonstrated, using criteria given by the interviewee to filter the horizon. Then, the history tracking feature was demonstrated retrieving how the horizon was previously interpreted by different users. In the second part, the previous user session was closed to start a new one. A seismic survey and a single-interpreter horizon which was not assessed previously were loaded and visualized. With aid from the author, the interviewees performed a time-shift modification to a region they selected by drawing a polygon over the horizon. Then, they assessed another region of their choice, using the same polygon drawing method. They were able to retrieve the recorded provenance of their actions (time-shift modification and assessment) through the tooltip feature by hovering around the areas of their contributions. Readers can refer to Chapter 6 for screenshots of similar scenarios. After that, the interviewees were asked three main open-ended questions about the following:

1. conceptual evaluation of this research outcome to the oil and gas industry,
2. cases where the outcome of this research can benefit the industry,
3. challenges in implementing the outcome into industrial practice.

Following the guidelines of Lazar et al. [112], the questions were attempted to be simple and unbiased. Following the semi-structured approach, two-way discussions took place prior to and during the time of the questions.

The interviews were audio recorded, after obtaining the consents of the participants. Later, they were transcribed; a full transcription of the three interviews is available in Appendix C. Transcription text is then read carefully to be coded. *Coding* is a qualitative data analysis method which assigns a code to passages of a similar theme or concept [133]. Coding for the conducted interviews was performed manually by carefully reading and analysing the transcripts. There exist computer software tools that facilitate the coding process, but none were used for the purpose of this thesis due to the intent of the author to fully read and manually analyse the transcripts. The

next subsection presents the outcome of the interviews using the coding method. The subsequent subsection presents further analysis to the data outcome for the purpose of evaluating the work of this thesis.

7.3.2 Data Outcome—Using Coding

To describe the outcome of the interviews, transcripts were coded starting with *priori* codes. These are identified from the questions of the interviews. In addition, new codes emerged from the data since the interviews were semi-structured with open-ended questions. The following describes the outcome of the interviews grouped into codes; some items listed below are out of this thesis scope but mentioned since they were brought up by the interviewees. An analysis is presented in the next subsection.

Appreciation — The interviewees showed a high interest in the subject of this thesis. “It is definitely an interesting subject and quite rightly it does not have a simple answer”, one said expressing its challenge as well. “It fills an important gap in the workflow,” he added. Also, he described the outcome as “very useful” and “has an intrinsic value”. The concept of this research was described as “very important for the oil and gas industry”, another one expressed. They appreciated that the implemented provenance model records multi-user interactions and links between different data types (i.e. seismic survey, horizon and assessment as currently implemented in the prototype). Having the ability of the provenance model to link interpretations with related data and metadata would “save the modellers time”, an interviewee said; it is “one level of importance” in this research, he added. Interviewees admired the ability of precisely knowing the contribution per interpreter for a single geological horizon. Also, one interviewee pointed that “flagging and pinpointing uncertainty in a more quantitative way rather than word of mouth” is a useful outcome; it is “good to track”, he added. In addition, the system’s ability of providing access to all different versions of interpretation of a single horizon was described by one interviewee as it will pay interpreters “more benefits” and give them “a better judgment call”. It should be noted that negative comments are mentioned in below text as part of interviewees’ concerns, suggested enhancements and challenges.

Case Examples — The following were cases brought up by the interviewees which can be addressed by the outcome of this research. These cases are listed here and then described and analysed in more detail in Section 7.3.3.1 below.

- Enables geoscientists to work safely on data with high uncertainty by using a proper flagging, since the provenance model allows to flag (tag) data (i.e. seismic survey) and thus this becomes effective on related data products (i.e. seismic interpretations).
- Provides interpretations to other teams (e.g. modellers) with richer metadata (i.e. provenance information) instead of relying on limited information provided by file names.
- Helps in evaluating data for new project by knowing more about the data (i.e. provenance information).
- Helps in decision making by quantitatively determining confidence level (or uncertainty).
- Helps other teams (drilling) to know the data (seismic survey) that geological horizons or faults were interpreted on.

Concerns — The interviewees raised the following concerns. These are listed here and then described and analysed in more detail in Section 7.3.3.2 below.

- Increase of storage due to the fine-grained granularity of the metadata.
- Frequency of recording a time-stamp, or history point, as a result of user interactions.
- Passing data (e.g. interpretations) to another company (e.g. a joint venture or a service company)—what to pass and how to integrate returned results.
- Dealing with data of different types that have different alignment grids.
- Misusing interpreter information for accusing, in case of a failure in outcome, or blind reliability, because of the reputation or standing of someone in the company.
- Confusion when multiple users simultaneously interpret over the same area.

Enhancements — The following were raised during the discussion of the interviews; they were coded as *enhancements* of the current features. These are listed here and then described and analysed in more detail in Section 7.3.3.3 below.

- Using *low*, *medium* and *high* confidence levels instead of a scale of 100.

- Enhancing the visual highlight, or shading; it did not properly stand out as described by one interviewee.
- User friendly interface.
- Speed—updating the database with users’ interpretations should be faster or performed as a background process.

Future Applications — The following were perceived by the interviewees as potential future application, or work, as permitted by the current work of this thesis. These are listed here and then described and analysed in more detail in Section 7.3.3.4 below.

- Adapting more data and metadata types (e.g. 4D seismic data, well logs data, grid spacing and rotational angle)
- Visualizing cross-referenced multiple data types (e.g. well logs with seismic data)
- Using plug-ins with existing applications (e.g. Petrel)
- Implementing a provenance interface (e.g. XML or PROV-XML)
- Automatically detecting high uncertainty area, in addition to the implemented manual user assessment.
- Implementing a social network on the proposed architecture to allow better collaboration among users such as being notified of changes to data of a user’s interest.
- Allowing multiple commercial applications to update the single data repository.

Challenges — The following were raised by the interviewees as challenges in implementing the outcome of this research into industrial practice. These are listed here and then described and analysed in more detail in Section 7.3.3.5 below.

- Building an interface with existing commercial applications.
- Migrating pre-existing data into the provenance model of this thesis.
- Building a single central database to be accessed by all commercial applications.

7.3.3 Data Analysis

During the interviews, many questions were raised by the interviewees to get in depth understanding of the work. Although the intent was not to touch on technical details, the interviewees, as a result of their many years of experience, questioned some technical details, mainly about the data model and underlying data.

7.3.3.1 Case Examples

A common case in the industry was discussed, which is duplicating interpretations to make changes, even if small. A concern was raised if previous metadata stays with duplication. As described by one interviewee, interpretations are imported back into the interpretation application to make some changes, then they are exported to others. The new interpretations will be tagged, by proper file names or other means, with the last user's name or initial. Metadata about where each participated on this interpretation is lost. This results in incredibility, as described by the interviewees. Therefore, it was appreciated when they experienced the ability to track fine-grained changes on a horizon interpretation.

Another case where the work of this thesis can address is tracking interpretations of surveys with certain flags (tags), as the interviewees mentioned. For example, scientists may work on a seismic survey with a very high uncertainty for a preliminary study. Currently, they use the survey's file name to indicate its status. Thus, tracking resulting interpretations is challenging. The provenance model of this thesis can address this systematically by linking and keeping track of the resulting interpretations. Also, the fine-grained granularity of the provenance model implementation allows flagging uncertainty on certain areas within an interpretation. As described by one interviewee, this allows users to pay more focus on areas of high uncertainty (low confidence) rather repeating a full study. Current practice may rely on word of mouth to communicate with team members areas of high uncertainty to be further studied or reinterpreted.

Use of file names as a method of flagging certain conditions or recording the interpreter name was mentioned a number of times as a common practice in the industry, as also mentioned in Section 4.5. Several issues could rise from such method as described by the interviewees. If a user makes a small change to an interpretation, a new file is created with a new name that includes his or her name or initial. Thus, scientists lose track of previous interpreters as well as what each did. Also, file names cannot accommodate all required flags. The case of passing interpretations from interpreters to

modellers was raised. Files of interpretations would only include name of the horizon, last interpreter, year of creation or modification, domain of picking (time or depth) and optionally the seismic survey it was interpreted from. In some companies, a file name generator is provided to users to keep a convention within the organisation. This was clearly a limitation in the industry which can be addressed by the provenance model of this thesis.

Interviewees liked the ability of knowing the contribution per interpreter for a single geological feature. It gives a credibility to each contributor, as well as the ability to approach him or her for further discussion or any other purpose. This is a result of the fine-grained granularity of the provenance model implementation. It can be perceived that providing provenance information to subsurface data allows scientists to easily and better evaluate them as input to new projects.

In the oil and gas industry, it is common to return to a field which was appraised years earlier, for economical or political reasons, while still in the exploration phase. When bringing back the previous data, scientists have to decide whether they will use the old interpretation or reinterpret the field. In most cases, users tend to repeat interpretation, as mentioned by the interviewees, due to the lack of reliability and judgmental information. Thus, it can be perceived that certain provenance information can allow users to efficiently utilise their time. Regardless of how old an interpretation is, if provenance information exists, a user can feel closer to the thoughts of the original interpreter, as described by one interviewee.

7.3.3.2 Concerns

A concern was raised about the frequency of capturing history points, that is how often an interaction is recorded on as a time stamp. Currently, the prototype application defines a history point as any process of change on the data (e.g. amendment to an interpretation). In practice, hundreds of iterations, of auto-tracking and manual alteration, may occur at a short period of time. Some of these may only change a very small region as part of a larger change. It might be impractical to record each process of change as a separate history point, as mentioned by an interviewee. This can be customised so that changes are first applied in the FESVo (local) then commit changes every 5 minutes, for example, or at a user request to the central database.

Another concern was raised about sharing data (interpretations) with bodies outside the company, such as a joint venture, partner or service company. The owner of the data usually attempts to share a single approved version of the interpretation. The

concern here is whether this data will include all of its original metadata. If so, this raises another question on how to pass the selection of the approved version, since it is a collection of multi-user interpretations and assessment in the presented model of this thesis.

One interviewee raised a concern about the ability of tracking the exact contribution (i.e. fine-grained) of each interpreter as it may result in an “automatic blame gaming”. He also raised a concern of misusing this information blindly for the purpose of reliability. That is treating regions interpreted by certain people as highly accurate due to seniority for example. It was perceived from the interviews, as well as working closely with people from this domain, that there is a social aspect in how they would use such information for credibility and reliability. This can be studied as a future work.

Another concern was raised about allowing multiple users to interpret over the same area. In practice, if multiple users may interpret a single survey by dividing it geographically. Another case would be to work on an area asynchronously, meaning that one user will work after another completes their part; they may later discuss any disagreement. The prototype application would allow users to record all interpretations as well assessing disagreement through the confidence assessment feature.

7.3.3.3 Suggested Enhancements

A number of enhancements were suggested. One suggested to have the confidence factor as low, medium, and high, instead of a scale of 100. He preferred not to use an integer value from the user. “Giving 3 options is much easier than 100 options”, he said. This a human and organisation preference which can be customised. Also, it was suggested to enhance the highlighting (shading) feature as it “did not make the area stand out very well.” A better method should be investigated and implemented to resolve this as a future work. In addition, one interviewee critiqued the slowness of interpretation updates into the database. This was partially because of the use of a local database at the time of the interview due to the need of travel to the location of the interviewee, who was geographically far from the primary setup in which the default database was running on a high spec machine. He suggested to perform the update behind the scene; this was already considered as a future work. Having seen the capability of the model linking interpretations to related data (e.g. survey) and metadata, it was suggested to adapt more metadata for modellers. These include rotational angle and grid spacing. This would save modellers time in correctly importing interpretation

into a geological model, i.e. if they immediately knew related information like rotational angle and grid spacing. The interviewee expressed their awareness that some of their suggested enhancements are not within the main scope of this thesis.

7.3.3.4 Future Applications

The interviewees perceived the potential for future work. This is an indication of the limitation of current solutions provided to the industry. Most importantly, it is an indication that the work of this thesis demonstrates a groundbreaking foundation that can address some of the industry's current challenges.

One interviewee wished to access and visualize cross-referenced data from different types, possibly from a petroleum system model. These may include well data and 4D seismic data. The implementation of this is possible due to the spatially referenced provenance model of this research. The proposed provenance model can be extended to link for example between an interpretation and a reservoir (geological) model. The latter is updated when wells are drilled, so a geological model can be also linked to geological picks which are interpreted from well logs. Commercially, a plug-in to Petrel, named *Blueback Geodata Investigator*¹ allows data selection and import from spatially cross-referenced data of different types and sources into Petrel. In other words, it performs data mining. However, it lacks the context that the model of this thesis brings through provenance. Therefore, it was suggested to try such solutions to link between the integrated data model of this thesis with a commercial application like Petrel for visualization and interpretation.

The concern mentioned earlier about passing a selection criteria of the approved, or an interesting, interpretation can be resolved as follows. The implementation of the data model is a result of a conceptual translation of the proposed provenance model, which is based on the PROV-DM. Having this base of the PROV-DM, the PROV-XML [134] can be utilised and extended to provide a method of storing and sharing provenance information outside the implemented application. This solution was shared with the interviewed domain experts.

In addition to the manual user assessment capability, it was suggested by one interviewee to add a mechanism to detect areas of high uncertainty. The provenance model and its implementation allows this to be accommodate as a type of annotation, similar to user assessment.

¹ <http://www.blueback-reservoir.com/software/software-products/geodata-investigator>

7.3.3.5 Challenges

To implement the outcome of this thesis into industrial practice, two interviewees raised the difficulties in connecting the proposed provenance model with existing commercial applications. This is not just about preparing a dataset to be visualized in Petrel, for example. It is more about allowing users to utilize the provenance information in selecting data of a certain criteria using a commercial application and more importantly capturing and translating users' interactions in such application into the provenance format. This should be a development in future work.

Another challenge that was raised by an interviewee is having the implemented data repository, which represents the provenance data model, as a single standard database which connects to each commercial applications used by the hosting organization.

7.4 Performance Measures

The primary aim of this thesis is to achieve data provenance with an acceptable performance; this means testing the non-functional aspects of the application. In this section, performance is measured firstly using commodity hardware, which was used throughout the development of the system, and secondly using a massively parallel processing (MPP) database. The latter is illustrated to examine the scalability of the architecture in fetching datasets from a multi-node database. These measurements are presented here to give an indication of how the implemented system performs with its current limitations.

7.4.1 Using Commodity Hardware

These tests, and also the above case study, were performed on laptop and desktop machines equipped with graphics cards with 256MB to 1GB of memory. The database was running virtually on a 64-bit Windows Server 2003. The total size of the tables, which included seismic imaging datasets, users' interpretations and metadata, was around 35GB; this is similar to a typical dataset in actual practice.

Over a local area network (LAN), the system was able to initially load seismic traces of one seismic volume at a level of detail with a lower resolution in around 7 to 8 seconds; this level had a size of 38MB which occupied the GPU texture memory. The feature object (horizon) was loaded in around 1 to 2 second(s); both traces and the feature object form the output as illustrated in Figures 7.1 or 7.2. The loading

Data Source	Fetching Time
Local Drive	2.6s
NAS 1	38.4s
NAS 2	76.7s
Database	29.0s

Table 7.2: The table shows the fetching time for a 155MB-dataset from four different sources: using a conventional SEG-Y file on a local drive, network attached storages on two different locations (NAS1 and NAS2) and using the architecture in this thesis which had its database physically located near NAS2.

process used 4 threads concurrently. Each SQL command is a multi-statement request consisting of a maximum of 16 point-to-point queries. Each query is a fetch request of a macro previously set up to query a single trace or a set of feature points against a unique geographical location (x,y) . The tests showed that the fetching time from fine-grained structure on the database was less than fetching a SEG-Y file located remotely in a network storage; see Table 7.2 for a comparison. Referring to this table, note that this is not the transfer time of the dataset from the different sources to the local machine but it is the time to fetch the trace samples of the dataset which requires reading each trace's header. The speed gained from the architecture can be related to the use of parallel threads which are not used when fetching from SEG-Y files in the network attached storages (NASs).

A number of parameters had an influence on the overall performance: (1) number of threads running concurrently; (2) the length of a multi-statement request, i.e. the number of queries executed in a single statement; and (3) the type of query. For the query type, a query to fetch a dataset contributing to building the data buffer must always be against primary indices, thus avoiding a full table scan which is expensive. This guarantees the highest throughput independently of the dataset size, a powerful advantage when dealing with massive datasets. The following discusses the first two parameters mentioned.

7.4.1.1 Effect of Threads

By experiment, it has been found preferable for the number of threads running concurrently to match the database's parallelism units, which are known by Teradata as *access module processors* (AMP) as mentioned in Section 5.3.1. This would only be the case when eliminating bottlenecks such as connection speed and capability of the client machine to run a number of parallel threads. The database used in this test had four

Multi-Statement Length	Data In (KB)	Data Out (KB)
1	1.4752	0.7151
2	1.3119	0.4082
3	1.2584	0.3294
4	1.2303	0.2759
8	1.1824	0.1999
12	1.1655	0.1745
16	1.1627	0.1605

Table 7.3: Query bandwidth (data in and out) is reduced by executing larger multi-statement requests; i.e. running more queries in a single transaction. Figures in the table are computed per query. Measures were taken using the same total number of queries in each case.

AMPs. Thus, as shown in Figure 7.3, the throughput became stable when four threads or more were used. This test was performed over a LAN between the client machine and the database. When using a slower connection, more threads, up to a limit, would boost the overall throughput as the number of threads overcomes the slowness of the connection. This is due to the lag in the arrival of the queries at the database and thus it does not saturate as fast as when using a high speed connection.

7.4.1.2 Effect of Multi-Statement Length

On a local area network (LAN) with speed of 100Mbps, the dominant overall performance factor becomes the database performance. On a broadband Internet connection with a shared speed of around 2 to 4 Mbps, the connection speed becomes a dominant factor. Therefore, at a broadband connection, the bandwidth of input and output data should be reduced by stacking more queries in a single statement (see Table 7.3) to achieve a higher throughput. As illustrated in Figure 7.4, the speedup difference of larger multi-statements in a lower speed connection is much higher compared to a high speed connection.

7.4.2 Scalability Tests

These tests were intended to measure the capability of the implemented architecture to run multiple threads on a large cluster database, aiming to speedup the data fetching and data lookup process. The database was running a Teradata RDBMS on a four-node cluster, each had two eight-core 2.6GHz processors and 256GB of memory. The total number of AMPs was 36. The tests were performed on a set of dummy data equivalent

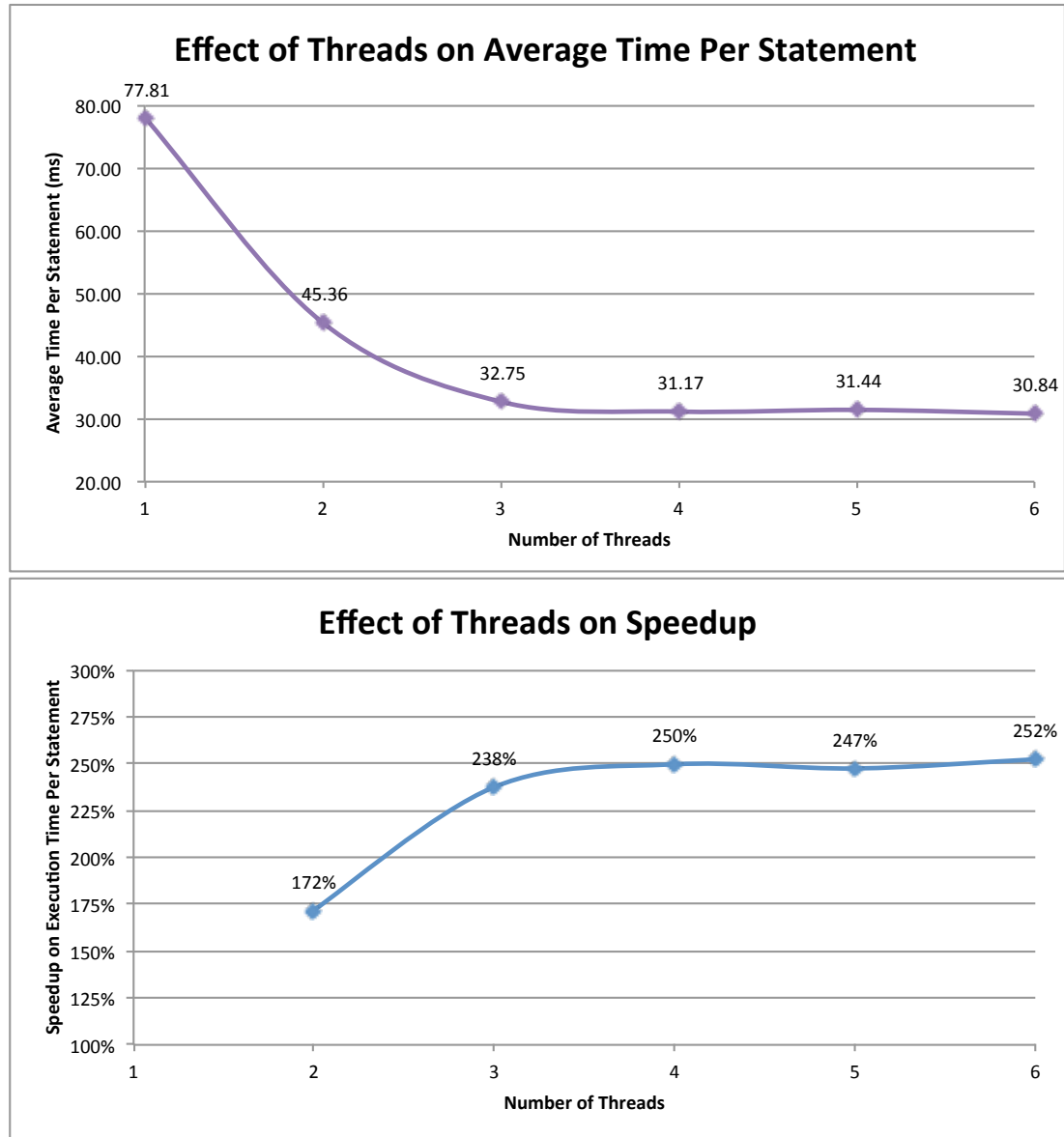


Figure 7.3: The top and bottom graphs show the average time per statement and speedup percentage, respectively, for different numbers of threads; the cases queried identical datasets over high speed connection (LAN). The graphs show that throughput became stable after four concurrent threads, which is the number of the database's parallelism units.

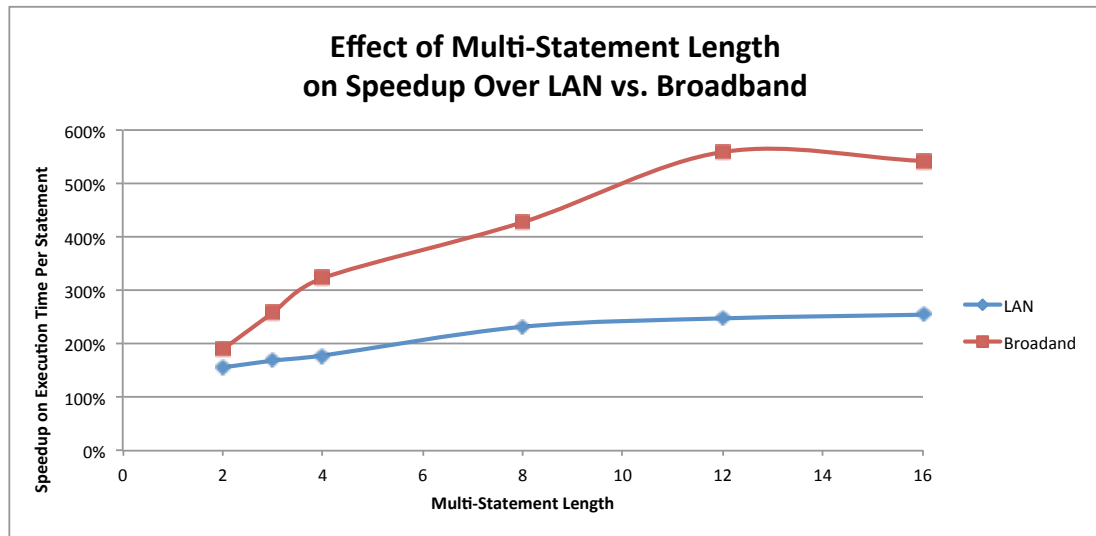


Figure 7.4: The graph shows the speedup percentage for different numbers of statements per transaction (multi-statement length) over a high speed connection (LAN) and over a broadband connection; the cases queried identical datasets and used a fixed number of concurrent threads. As the graph suggests, larger multi-statements have more influence on boosting the throughput over a slow connection since they reduce the query bandwidth.

to a 22GB seismic dataset, implemented in more than 25 million rows. The client machine which ran the testing code was a Linux server with Java virtual machine (JVM) 1.6 and 256GB memory. Due to limited access to the hardware, which was hosted in a Teradata owned laboratory, the tests did not cover the complete architecture but only measured query execution time; i.e. it did not measure performance at the rendering stage but only at the data loader.

One scalability test included running the same job size over cases of an increasing number of threads; i.e. as the number of threads increased fewer jobs were given to each thread. Each case queried a total of 4 million (x,y) points. Each SQL query was a 26×26 (676) multi-statement query. As illustrated in Figure 7.5, no speedup was gained beyond 16 threads. Ideally, we should expect a continuous speedup up to 36 threads (number of AMPs).

Other tests with different parameters and conditions were performed. All clearly showed some degree of scalability. In comparison to the commodity hardware its results of which were discussed earlier, the speedup was 240% to 294% only despite the power of the cluster. It was expected to reach a maximum speedup at 36 threads which is equivalent to the number of AMPs. However, the maximum speedup was

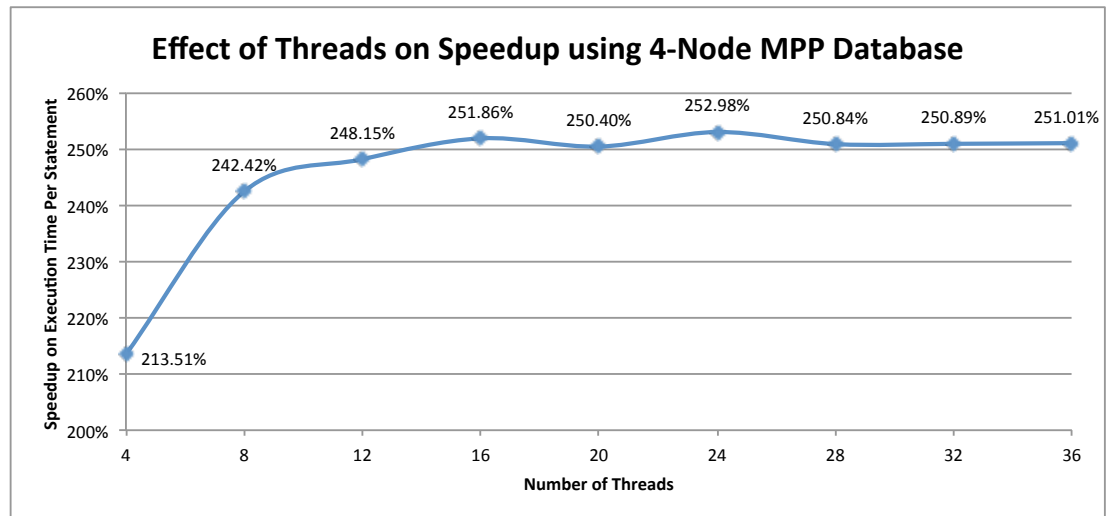


Figure 7.5: The graph shows the speedup percentage in different numbers of threads using a 4-node parallel database; the cases queried identical datasets. The graph shows that throughput became stable after 16 concurrent threads.

gained at 16 threads; performance remained steady with more than 16 threads.

One possible assumption for not gaining (1) a greater speedup in general and (2) a speedup beyond 16 threads is that this might be related to the Java Virtual Machine (JVM) and / or Java Database Connectivity (JDBC) as they may not have handled the increasing number of multi-threaded connections efficiently. This assumption needs investigation beyond the scope of this thesis; it is therefore considered as future work.

7.5 Chapter Summary

The chapter presented an evaluation of this research. With the presentation of Chapter 5, the third objective of this thesis was addressed. The evaluation was conducted in three parts: (1) a usability evaluation by geoscientists (functional test); (2) a conceptual evaluation by interviewing domain experts; and (3) performance measures (non-functional test).

Six postgraduate students in the field of geoscience participated in the usability evaluation. This illustrated the system's ability in allowing users to amend others' interpretations and trace the history of amendments. They were introduced to the prototype application and then asked to perform two tasks in a collaborative environment. Participants were able to see amendments by their partners working on a different machine and also trace back the history of changes.

The conceptual evaluation of this research was approached by interviewing three domain experts. They were first introduced to the research. Then they were given a demonstration of the prototype application. Also, they were allowed to interact and perform some scenarios, i.e. amending and assessing a previously interpreted horizon. The provenance of their contributions immediately appeared to them through the tooltip feature of the user interface. Then, the interviews followed a semi-structured form, guided by some open-ended questions. They were audio recorded, transcribed, coded and then analysed. The interviewees assured that the research is of a high importance to the industry. They shared current limitations in their workflow and potential future work which the contribution of this thesis can address. On the other hand, they shared concerns about the current prototype implementation and challenges in implementing it in industrial practice.

Performance measures were presented in this chapter to illustrate the behaviour of the architecture firstly using commodity machines then using a multi-node parallel database. Using a commodity machine, the measures suggested a boost in performance when using multiple threads and multi-statements queries. These figures depends on the connection speed with the central database and the specs of the machine running the database. The measures also suggested that the current implementation achieved an acceptable performance in comparison to conventional methods. Using a multi-node parallel database, an overall speedup of 240% to 294% was gained over the results of the commodity hardware setup. An investigation of possible bottlenecks causing this speedup limit is beyond the scope of this research.

Chapter 8

Conclusions and Future Work

This chapter firstly provides a summary and conclusion for the thesis. It then discusses future work, part of which addresses the limitations of the current implementation.

8.1 Summary and Conclusion

The thesis has achieved its aim stated in Section 1.4, which is addressing the problem of capturing and tracking multi-user interpretations of three-dimensional spatial datasets. In the context of this thesis, this interpretation refers to the process of understanding and extracting features. This process is completed after the end of the visualization pipeline and is subjective to human intuition and knowledge. An algorithm may extract a preliminary view of a feature, but its definition can only be completed by the user marking areas that contribute to the identification of the structure of the feature. Therefore, two interpreters may produce different interpretations of the same visualization. This thesis developed a provenance-based method to address the issue of tracking users' interpretations. Provenance information can answer the question of who produced or contributed in producing an interpretation and whether it was evaluated. The field to which this research was applied is seismic imaging which services the oil and gas industry. The thesis is fundamentally related to the management of visual interpretations of features and to the integration of a model of provenance into the process of visual exploration. It illustrated and evaluated these novel methods by applying them to the process of interpretation of features in seismic data.

In this thesis, a concept of provenance to track and link multi-user interpretations, which is independent of visualization functions, was developed. Using this concept, a prototype provenance-enabled architecture was implemented and evaluated for seismic

imaging interpretation. Also, a number of publications resulted from the work of this thesis. The following provides a summary of the chapters presented in this thesis with a reference to its objectives as stated in Section 1.4.

Chapter 2 presented a survey of the background literature and highlighted some related work in three main areas: visualization in general, features in visualization and provenance. It implicitly fulfilled **Obj. 1**, which was to investigate the visual interpretations workflow and managing the resulting interpretations. The literature has addressed integrating the notion of provenance into the visualization pipeline to capture metadata of its modules that creates the pipeline. However, research currently reported in the literature lacks a method of recording provenance for user interpretation, which encodes the intuition of users through actions performed on the rendered visualization. To fill this gap, the thesis proposed a provenance-enabled interpretation pipeline in Chapter 3, fulfilling **Obj. 2**. This is to bring users' interpretations and related metadata next to the repository of the raw data in a provenance model. The thesis adopted and extended the W3C PROV data model, producing a provenance model for visual interpretations. The *interpret* namespace was introduced to accommodate interpretation related activities within the model. The use of this model was illustrated by examples, such as a multi-interpreter contribution and an assessment of interpretation.

The thesis then focused on the domain of its application, seismic imaging interpretation for the oil and gas industry, achieving **Obj. 3**. This is a field that supplies most of the world's energy demand. Chapter 4 presented background information about this field. It defined seismic imaging data and how it is acquired and interpreted to extract geological features. Seismic data is noisy and of a high uncertainty. Current status of seismic interpretation management was discussed, which is of a great challenge to the industry. It heavily relies on file naming conventions to tag interpretations with metadata (e.g. creator's initial, field name). Other solutions relies on the application level to manage project data including users' interpretations, at the cost of losing metadata when an interpretation is exported. The need for such tracking of the provenance of interpretations was supported by a survey of practitioners within the oil and gas industry who are involved in the analysis and interpretation of seismic data. The participants confirmed the importance of history tracking in seismic interpretation activities and its absence from existing solutions.

The proposed interpretation pipeline and provenance model were implemented for seismic imaging interpretation in a proof-of-concept prototype architecture and application, as presented in Chapter 5. The architecture is a server-client and consists of

three-loosely coupled components: (1) fine-grained data structure implemented in a database, (2) FESVo (an intermediate stage implemented on the client side) and (3) rendering and user interface on the client side. The data structure is based on an implementation of the provenance model as a relational database model, in a central database.

The provenance model was further extended to accommodate geological data by introducing the *geo* namespace. The implemented tables and underlying data were explained in the chapter. Seismic surveys (raw data) were restructured into tuples (rows), one for each seismic trace. Geological features were represented as points, a row for each point. The data in this model were referenced on their global geographical coordinates. The model accumulated users' interpretations and assessments of features as fine-grained data linked together as well as with the raw data.

The FESVo was originally considered as a *feature-embedded spatial volume* but later developed as a more general method that partially combines and implements the visualization pipeline (excluding rendering) and the proposed interpretation pipeline. It has the role of loading data, caching data and capturing users' interpretations, between the central database and user interface. It maps between different coordinate systems, from a global geographical coordinate to a local volume coordinate and then all the way to a user screen's coordinate for loading and rendering a dataset, and vice versa when capturing a user interpretation. A filtering mechanism was built in loading process, taking advantage of the provenance information to select a dataset of interest based on a user-defined criteria. A walkthrough of selected scenarios using the prototype application was demonstrated in Chapter 6.

The evaluation of this research was conducted in three parts as presented in Chapter 7: (1) a usability evaluation by geoscientists (functional test); (2) a conceptual evaluation by interviewing domain experts; and (3) a performance measures (non-functional test). Six postgraduate students in the field of geoscience participated in the usability evaluation. This illustrated the system's ability to allow users to amend others' interpretations and trace the history of amendments. The conceptual evaluation of this research was approached by interviewing three domain experts. The interviews were audio recorded, transcribed, coded and then analysed.

The interviewees assured that the research is of a high importance to the industry. They shared current limitations in their workflow and potential future work which the contribution of this thesis can address. On the other hand, they shared concerns about the current prototype implementation and challenges in implementing it in industrial

practice.

For the performance evaluation, although the thesis did not claim a performance improvement over other work in seismic visualization, performance measures were presented to illustrate the behaviour of the architecture on commodity machines as well as on a multi-node parallel database. The measures suggested that the current implementation achieved an acceptable performance in comparison to conventional methods. It should be noted that conventional methods, as embodied in industry standard software such as Petrel, have benefited from major investment in optimization on different architectures. The purpose of this thesis was to demonstrate that a new functionality in fine-grained provenance can be implemented simply but with an acceptable performance in realistic visualization tasks.

The above evaluation supports the conclusion that the proposed model is a promising shift in methods of data management and storage which can record and preserve interpretations by users as a result of visualization. The author has presented evidence that interpretation of seismic data is a promising area for developing this approach. The approach and its prototype application in this thesis represent a step in this direction. It requires further research and developments to achieve the stage where it can be used in industrial practice.

The evidence presented in Chapter 7 on evaluation allows the author to be confident in the future of this direction in principle—this was evident from the reception of publications of this work and in comments received from the geoscience community as well as from direct communication with senior staff in the industry. The publication of this work in EAGE 2013 [27] “received among the highest ratings from its reviewers” as stated by Gerhard Diephuis, Editor in Chief of First Break. The author, therefore, was invited to submit a journal paper to First Break, an EAGE journal. Also, the paper published in the 5th International Conference on Information Visualization Theory and Applications (IVAPP 2014) [29] received a best paper award and was extended into a book chapter submitted to *Computer Vision, Imaging and Computer Graphics: Theory and Applications*.

8.2 Limitations and Future Work

The work presented in this thesis can be extended in two directions: future research and further development. The latter addresses the limitations of the implemented system

and further suggestions. With both directions, future work should include a long-term evaluation to sufficiently measure the benefit of the proposed architecture—since it is more challenging to manage multi-user interpretations in the long term, than in the short term, as suggested by the survey conducted in Section 4.7 and discussed in Chapter 7.

8.2.1 Future Research

The following discusses research areas that can be initiated from or linked to the work in this thesis.

8.2.1.1 Features Detection

The problem of geological features detection (extraction) was not considered in this work. It is a field of research to which many have contributed, including work such as Höllt et al. [14], the use of level sets by Kadlec et al. in computing fault surfaces [135] and later with assistance from users' knowledge [56], and the use of implicit surface representation of horizons by Patel et al. [13] for rapid seismic interpretations. However, to the knowledge of the author, there is still a gap between user expertise and automation of features detection.

A future avenue for investigation is the potential to combine the fine-grained provenance of interpreted features (presented in this thesis) with machine learning techniques. As was demonstrated in this thesis, interpretations can be revisited and modified by multiple users to eventually reach the most accurate result, which is often ratified by an expert user. Machine learning techniques can potentially benefit from the revisiting and modification process to understand how an algorithm can produce objects that do not accord with the intuition of trained experts in the field. It has to be noted that the final decision about whether to explore physically in a particular location is still a collaborative process involving human communication, reasoning and decision making. It can then be possible to assist less-expert users in the interpretation process.

8.2.1.2 Scalability

As demonstrated in Chapter 7, the implemented system underwent a scalability test. Some degree of speedup was achieved but it was not what would have been expected given the potential parallelism available to the database queries. Future research can

investigate the potential of scalability in both the loading and the database sites. The former should address the implementation of database connectivity (e.g. JDBC) while the latter should investigate the parallelism of the database. This future work should be conducted in collaboration with the leading research in the area of parallel databases.

Addressing the increasing size of the datasets, as shown in this work with restructuring into a fine-grained granularity, can also fall into the scalability investigation. Current seismic surveys use datasets of gigabytes and terabytes but this could increase as visualization methods improve to handle larger datasets. Also, improved surveying technologies could increase the size of the survey datasets. Adding the preservation of the provenance of multi-user interpretations to this can cause continuous growth in the size of the stored data, even though the size of an interpreted feature is a fraction of its corresponding raw data. For these reasons, a scalability related extension to this thesis should take into consideration the data size issue. As was mentioned in Section 7.2, the current implementation does not compress data on the database. A possible direction would be to use a standard compression technique offered by the RDBMS. However, a direction that may lead to a more efficient result would be to investigate an optimum compression technique for each data type.

8.2.1.3 Social Aspects

The fine-grained data provenance model records the contribution of a user on the interpretation of a feature. It allows to know the region each user interpreted. Also, it allows to assess, or in general annotate, a user-selected region within an interpreted feature. Thus, users can annotate, for example, area of high uncertainty to be further studied or discussed. This was appreciated by the domain expert interviewed in this research project. It would help geoscientists to better utilise their time by focusing on demanding areas. However, a concern was raised on the utilisation of such information. Users may blindly accept an interpretation that was interpreted or assessed by a user of high reputation. Or, such information may be used to accuse someone in the company if in case of a failure related to the user's interpretations or assessments. This is a valid concern that was raised by one of the interviewees. The social aspect on how users would utilise such provenance information should be studied as a future research work.

8.2.2 Further Development

As highlighted in Chapter 7, the prototype application served the purpose of evaluating the proposed architecture which implemented the provenance model for interpretation. This prototype, however, has limitations which prevent its use for production. Some of these were brought up by domain experts during their interviews for the conceptual evaluation of this thesis. In this section, a number of limitations are discussed and presented as future work.

8.2.2.1 Provenance Language

The implementation of the data model, presented in Chapter 5, is a result of a conceptual translation of the proposed provenance model, which is based on the W3C PROV-DM and was presented in Chapter 3. The PROV-DM is intended to interchange provenance information in heterogeneous environments using widely available formats such as XML. The PROV-XML [134] can be utilised for this purpose, which defines an XML schema for the provenance data model. Following this approach, an XML interface can be built for the implemented provenance model of this thesis, accommodating the introduced namespaces. This should allow different applications to communicate with the data model using a unified language. As a result, it should also allow users to share selected dataset represented by provenance information, independently of the application.

8.2.2.2 Advanced Functionalities Interface

An advanced interface should be offered to users to sufficiently test the architecture over a long period while running actual production work.

A potential and comprehensive solution to the problem of integrating the proposed architecture into an advanced visualization and interpretation interface would be to develop a plug-in that interoperates with a commercial application, such as Schlumberger's Ocean¹ which is a software development framework to write plug-ins for Petrel². Another suggestion would be to integrate the proposed architecture into a mature open-source seismic visualization and interpretation application such as OpendTect³.

¹<http://www.ocean.slb.com/Pages/default.aspx>

²Ocean plug-in was given in the above context as an example of a plug-in to a seismic visualization application. However, it was not tested to support the architecture of this thesis.

³<http://www.opendtect.org/>

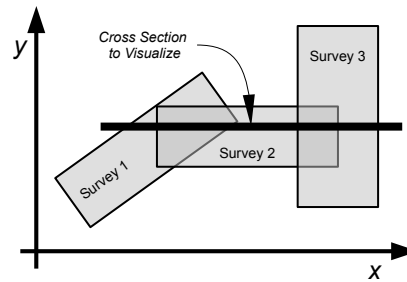


Figure 8.1: The figure illustrates a section crossing multiple seismic surveys.

8.2.2.3 Enhanced Versioning

The current prototype application does not address branching in versioning, or history tracking. In other words, if user B modified user A's interpretation, the current implementation does not allow user C to start a new branch by directly modifying user A's work, but can only modify the latest version. Thus, the current versioning should be enhanced. This enhancement is only at the application level as the provenance model supports branching in history tracking.

8.2.2.4 Data Handling

As mentioned in Section 5.4.1, the current implementation only handles one seismic survey. Although the data structure was designed to fetch multiple raw datasets, the current data loader was not extended for this purpose. It is suggested to address this as a future enhancement. It would allow the visualization of a section crossing two or more seismic surveys, as illustrated in Figure 8.1.

The current implementation is not good at handling seismic surveys that are at an angle to geographical latitudes; i.e. the survey's in-lines and cross-lines are not in line with geographical latitudes and longitudes. The gridding and mapping technique should be enhanced to deal with such a case.

Implementation of the methods proposed in the thesis has been limited to consideration of geological horizons. However, it can be assumed that other feature types should be handled in the same way by the system in terms of storage, loading and rendering. However, this should be tested.

The current implementation renders raw seismic data as either a cubic volume or a single slice parallel to the x-axis. As future work, users should be allowed to visualize slices in any direction including visualizing time slices as illustrated in Figure 4.4.

One issue with fetching a time slice from the implemented data model is the inability to access a single amplitude sample instead of a whole trace. A customised retrieval function for the database should be built to query a single sample from a trace given a time value.

The computation of seismic attributes was not addressed in this thesis. Seismic attributes are calculated quantities derived from the original seismic data; they result in a new *attribute volume*. They help geoscientists in detecting structural and stratigraphic features. Future work can investigate the utilisation of the parallel database, used in this work, to compute seismic attributes. This may potentially fall into research, and not just development.

8.2.2.5 Enhanced Rendering

The current implementation utilises a standard and basic rendering technique, which is texture mapping. Other rendering techniques can be investigated such as GPU-based ray casting [136] and splatting [137, 138]. The latter is a potential technique to render geological features; neither was investigated in this PhD.

Bibliography

- [1] R. Haber and D. McNabb, “Visualization Idioms: A Conceptual Model for Scientific Visualization Systems,” *Visualization in Scientific Computing*, pp. 74–93, 1990.
- [2] K. Moreland, “A Survey of Visualization Pipelines,” *Visualization and Computer Graphics, IEEE Transactions on*, vol. 19, pp. 367–378, Mar. 2013.
- [3] T. McInerney and D. Terzopoulos, “Deformable models in medical image analysis: a survey,” *Medical Image Analysis*, vol. 1, pp. 91–108, June 1996.
- [4] J. C. Roberts, “On Encouraging Multiple Views for Visualization,” in *Information Visualization, 1998. Proceedings. 1998 IEEE Conference on*, pp. 8–14, IEEE, July 1998.
- [5] L. Bavoil, S. Callahan, P. Crossno, J. Freire, C. Scheidegger, C. Silva, and H. Vo, “VisTrails: Enabling Interactive Multiple-View Visualizations,” in *Visualization, 2005. VIS 05. IEEE*, pp. 135–142, IEEE, 2005.
- [6] C. E. Scheidegger, H. Vo, D. Koop, J. Freire, and C. T. Silva, “Querying and creating visualizations by analogy,” *IEEE transactions on Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1560–1567, 2007.
- [7] C. Silva, J. Freire, and S. Callahan, “Provenance for Visualizations: Reproducibility and Beyond,” *Computing in Science Engineering*, vol. 9, no. 5, pp. 82–89, 2007.
- [8] J. Plate, M. Tirtasana, R. Carmona, and B. Fröhlich, “Octreemizer: a hierarchical approach for interactive roaming through very large volumes,” in *Data Visualisation*, pp. 53–60, Eurographics Association, 2002.

- [9] L. Castanie, B. Levy, and F. Bosquet, "VolumeExplorer: Roaming Large Volumes to Couple Visualization and Data Processing for Oil and Gas Exploration," in *IEEE Visualization*, vol. im, pp. 247–254, Ieee, 2005.
- [10] J. C.-R. Lin and C. Hall, "Multiple oil and gas volumetric data visualization with GPU programming," *Proceedings of SPIE*, vol. 6495, pp. 64950U–64950U–8, 2007.
- [11] J. Plate, T. Holtkaemper, and B. Froehlich, "A flexible multi-volume shader framework for arbitrarily intersecting multi-resolution datasets.," *IEEE transactions on visualization and computer graphics*, vol. 13, no. 6, pp. 1584–91, 2007.
- [12] D. Patel, O. y. Sture, H. Hauser, C. Giertsen, and M. E. Gröller, "Knowledge-assisted visualization of seismic data," *Computers & Graphics*, vol. 33, pp. 585–596, Oct. 2009.
- [13] D. Patel, S. Bruckner, I. Viola, and E. M. Gröller, "Seismic volume visualization for horizon extraction," in *Pacific Visualization Symposium (PacificVis), 2010 IEEE*, (Taipei, Taiwan), pp. 73–80, IEEE, 2010.
- [14] T. Höllt, J. Beyer, F. Gschwantner, P. Muigg, H. Doleisch, G. Heinemann, and M. Hadwiger, "Interactive seismic interpretation with piecewise global energy minimization," in *Pacific Visualization Symposium (PacificVis), 2011 IEEE*, (Hong Kong), pp. 59–66, Mar. 2011.
- [15] T. Holtt, W. Freiler, F. Gschwantner, H. Doleisch, G. Heinemann, and M. Hadwiger, "SeiVis : An Interactive Visual Subsurface Modeling Application," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 18, no. 12, pp. 2226–2235, 2012.
- [16] BP, "BP Statistical Review of World Energy June 2012," tech. rep., June 2012.
- [17] BP, "BP Energy Outlook 2030," tech. rep., London, Jan. 2012.
- [18] M. Bacon, R. Simm, and T. Redshaw, *3-D Seismic Interpretation*. Cambridge University Press, 2003.
- [19] E. Robein, *Seismic Imaging: A Review of the Techniques, their Principles, Merits and Limitations*. EAGE Publications bv, 2010.

- [20] E. M. Lidal, M. Natali, D. Patel, H. Hauser, and I. Viola, “Geological storytelling,” *Computers & Graphics*, vol. 37, pp. 445–459, Aug. 2013.
- [21] R. R. A. Demyttenaere, A. H. Sluijk, and M. R. Bentley, “A fundamental reappraisal of the structure of the Cormorant Field and its impact on field development strategy,” in *Petroleum Geology Conference*, vol. 4, (London), pp. 1151–1157, Geological Society of London, 1993.
- [22] C. Chelmis, J. Zhao, V. S. Sorathia, A. Suchindra, and V. K. Prasanna, “Toward an Automatic Metadata Management Framework for Smart Oil Fields,” *SPE Economics & Management*, vol. 5, pp. 33–43, Jan. 2013.
- [23] D. Irving, M. Rasheed, and N. O’Doherty, “A Massively-parallelised Spatially-registered Data Structure for Integrated Subsurface Data Management,” in *72nd EAGE Conference & Exhibition*, (Barcelona), EAGE Publications bv, June 2010.
- [24] D. Irving, N. O’Doherty, and M. Rasheed, “Multiresolution Spatiotemporal Data Structure for E&P Subsurface Data Storage and Analysis,” in *73rd EAGE Conference & Exhibition*, (Vienna), EAGE Publications bv, May 2011.
- [25] A. Al-Naser, M. Rasheed, J. Brooke, and D. Irving, “Enabling Visualization of Massive Datasets Through MPP Database Architecture,” in *Theory and Practice of Computer Graphics* (H. Carr and I. Grimstead, eds.), pp. 109–112, Eurographics Association, 2011.
- [26] A. Al-Naser, J. Brooke, M. Rasheed, and D. Irving, “An Architecture for Shared Multi-User Client Rendering of Massive Geodatasets,” in *The American Geophysical Union (AGU) Fall Meeting*, (San Fransisco), Dec. 2012.
- [27] A. Al-Naser, M. Rasheed, D. Irving, and J. Brooke, “A Data Centric Approach to Data Provenance in Seismic Imaging Data,” in *75th EAGE Conference & Exhibition incorporating SPE EUROPEC*, (London), EAGE Publications bv, June 2013.
- [28] A. Al-Naser, M. Rasheed, D. Irving, and J. Brooke, “A Visualization Architecture for Collaborative Analytical and Data Provenance Activities,” in *Information Visualisation (IV), 2013 17th International Conference on*, pp. 253–262, 2013.

- [29] A. Al-Naser, M. Rasheed, D. Irving, and J. Brooke, "Fine-Grained Provenance of Users Interpretations in a Collaborative Visualization Architecture," in *Proceedings of the 5th International Conference on Information Visualization Theory and Applications*, pp. 305–317, SCITEPRESS - Science and Technology Publications, 2014.
- [30] A. Al-Naser, "FESVo." [Online]. Available: <https://github.com/al-naser/FESVo> [Accessed 5 January 2015], 2013.
- [31] A. Al-Naser, "FESVo - Sep 2014." [Online]. Available: <http://youtu.be/fyBQYEEoHiQ> [Accessed 20 December 2014], Sept. 2014.
- [32] T. Mang, F. T. Kolligs, C. Schaefer, M. F. Reiser, and A. Graser, "Comparison of diagnostic accuracy and interpretation times for a standard and an advanced 3D visualisation technique in CT colonography.," *European Radiology*, vol. 21, pp. 653–662, Mar. 2011.
- [33] M. Moraga, C. Calero, and M. Bertoa, "Improving interpretation of component-based systems quality through visualisation techniques," *Software, IET*, vol. 4, pp. 79–90, Feb. 2010.
- [34] R. Gras, "The Human Factor in Interpretation and Visualisation," in *70th EAGE Conference & Exhibition*, (Rome, Italy), pp. 127–131, EAGE Publications bv, June 2008.
- [35] J. Ahrens, K. Brislawn, K. Martin, B. Geveci, C. Law, and M. Papka, "Large-scale data visualization using parallel data streaming," *IEEE Computer Graphics and Applications*, vol. 21, no. 4, pp. 34–41, 2001.
- [36] J. Biddiscombe, B. Geveci, K. Martin, K. Moreland, and D. Thompson, "Time dependent processing in a parallel pipeline architecture.," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1376–1383, 2007.
- [37] K. Stockinger, J. Shalf, K. Wu, and E. Bethel, "Query-Driven Visualization of Large Data Sets," in *Visualization, 2005. VIS 05. IEEE*, pp. 167–174, IEEE, 2005.
- [38] L. Gosink, J. Anderson, W. Bethel, and K. Joy, "Variable Interactions in Query-Driven Visualization," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 13, no. 6, pp. 1400–1407, 2007.

- [39] L. J. Gosink, J. C. Anderson, E. W. Bethel, and K. I. Joy, “Query-driven visualization of time-varying adaptive mesh refinement data,” *IEEE transactions on visualization and computer graphics*, vol. 14, no. 6, pp. 1715–1722, 2008.
- [40] K. Wu, “FastBit: an efficient indexing technology for accelerating data-intensive science,” *Journal of Physics: Conference Series*, vol. 16, pp. 556–560, Jan. 2005.
- [41] K. Wu, S. Ahern, E. W. Bethel, J. Chen, H. Childs, E. Cormier-Michel, C. Geddes, J. Gu, H. Hagen, B. Hamann, W. Koegler, J. Lauret, J. Meredith, P. Messmer, E. Otoo, V. Perevoztchikov, A. Poskanzer, Prabhat, O. Rübel, A. Shoshani, A. Sim, K. Stockinger, G. Weber, and W.-M. Zhang, “FastBit: interactively searching massive data,” *Journal of Physics: Conference Series*, vol. 180, p. 012053, July 2009.
- [42] F. J. Lucia, “Petrophysical Rock Properties,” in *Carbonate Reservoir Characterization, An Integrated Approach*, ch. 1, pp. 1–27, Springer Berlin Heidelberg, 2nd ed., 2007.
- [43] H. Vo, J. Bronson, B. Summa, J. Comba, J. Freire, B. Howe, V. Pascucci, and C. Silva, “Parallel Visualization on Large Clusters using Map Reduce,” in *Large Data Analysis and Visualization (LDAV), 2011 IEEE Symposium on*, pp. 81–88, IEEE, 2011.
- [44] A. Pavlo, E. Paulson, A. Rasin, D. J. Abadi, D. J. DeWitt, S. Madden, and M. Stonebraker, “A comparison of approaches to large-scale data analysis,” in *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*, pp. 165–178, ACM, 2009.
- [45] J. Ge, *A Point-Based Remote Visualization Pipeline for Large-Scale Virtual Reality*. PhD thesis, University of Illinois at Chicago, 2007.
- [46] J. Jomier, S. Jourdain, and C. Marion, “Remote Visualization of Large Datasets with MIDAS and ParaViewWeb,” in *Proceedings of the 16th International Conference on 3D Web Technology*, (Paris, France), pp. 147–150, ACM, 2011.
- [47] J. M. Brooke, J. Marsh, S. Pettifer, and L. S. Sastry, “The importance of locality in the visualization of large datasets,” *Concurrency and Computation: Practice and Experience*, vol. 19, pp. 195–205, Feb. 2007.

- [48] K. Gruchalla, M. Rast, E. Bradley, and P. Mininni, "Segmentation and Visualization of Multivariate Features Using Feature-Local Distributions," *Advances in Visual Computing*, vol. 6938, pp. 619–628, 2011.
- [49] K. F. J. Reinders, *Feature-Based Visualization of Time-Dependent Data*. PhD thesis, Delft University of Technology, 2001.
- [50] F. H. Post, B. Vrolijk, H. Hauser, R. S. Laramée, and H. Doleisch, "The State of the Art in Flow Visualisation: Feature Extraction and Tracking," *Computer Graphics Forum*, vol. 22, pp. 775–792, Dec. 2003.
- [51] T. Tuytelaars and K. Mikolajczyk, "Local Invariant Feature Detectors: A Survey," *Foundations and Trends in Computer Graphics and Vision*, vol. 3, no. 3, pp. 177–280, 2007.
- [52] H. Jänicke and G. Scheuermann, "Towards Automatic Feature-based Visualization," in *Scientific Visualization: Advanced Concepts* (H. Hagen, ed.), vol. 1 of *Dagstuhl Follow-Ups*, ch. 5, pp. 62–77, 2010.
- [53] N. Senthilkumaran and R. Rajesh, "Edge Detection Techniques for Image Segmentation A Survey of Soft Computing Approaches," *International Journal of Recent Trends in Engineering*, vol. 1, no. 2, 2009.
- [54] T. Salzbrunn, H. Jänicke, T. Wischgoll, and G. Scheuermann, "The State of the Art in Flow Visualization: Partition-Based Techniques," in *Simulation and Visualization 2008 (SimVis 2008)*, (Magdeburg), pp. 75–92, SCS Publishing House e.V., Feb. 2008.
- [55] F. Zhao and X. Xie, "An Overview of Interactive Medical Image Segmentation," *Annals of the BMVA*, vol. 2013, no. 7, pp. 1–22, 2013.
- [56] B. Kadlec, H. Tufo, and G. Dorn, "Knowledge-Assisted Visualization and Segmentation of Geologic Features," *IEEE Computer Graphics and Applications*, vol. 30, no. 1, pp. 30–39, 2010.
- [57] P. Buneman, "Curated Databases," in *Research and Advanced Technology for Digital Libraries* (M. Agosti, J. Borbinha, S. Kapidakis, C. Papatheodorou, and G. Tsakonas, eds.), vol. 5714 of *Lecture Notes in Computer Science*, pp. 2–2, Springer Berlin Heidelberg, 2009.

- [58] L. Moreau, “The Foundations for Provenance on the Web,” *Foundations and Trends in Web Science*, vol. 2, pp. 99–241, Feb. 2010.
- [59] Y. L. Simmhan, B. Plale, and D. Gannon, “A survey of data provenance in e-science,” *SIGMOD Rec.*, vol. 34, pp. 31–36, Sept. 2005.
- [60] M. Anand, S. Bowers, T. McPhillips, and B. Ludäscher, “Exploring Scientific Workflow Provenance Using Hybrid Queries over Nested Data and Lineage Graphs,” in *Scientific and Statistical Database Management* (M. Winslett, ed.), vol. 5566 of *Lecture Notes in Computer Science*, pp. 237–254, Springer Berlin Heidelberg, 2009.
- [61] C. Scheidegger, D. Koop, E. Santos, H. Vo, S. Callahan, J. Freire, and C. Silva, “Tackling the Provenance Challenge one layer at a time,” *Concurrency and Computation: Practice and Experience*, vol. 20, no. 5, pp. 473–483, 2008.
- [62] J. Cheney, L. Chiticariu, and W.-C. Tan, “Provenance in Databases: Why, How, and Where,” *Foundations and Trends in Databases*, vol. 1, no. 4, pp. 379–474, 2009.
- [63] J. Freire, D. Koop, E. Santos, and C. T. Silva, “Provenance for Computational Tasks: A Survey,” *Computing in Science and Engineering*, vol. 10, pp. 11–21, May 2008.
- [64] L. Di, P. Yue, H. Ramapriyan, and R. King, “Geoscience data provenance: An overview,” *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 51, pp. 5065–5072, Nov. 2013.
- [65] P. D. W. Haughton, S. P. Todd, and A. C. Morton, “Sedimentary provenance studies,” *Geological Society, London, Special Publications*, vol. 57, pp. 1–11, Jan. 1991.
- [66] C. Goble, “Position statement: Musings on provenance, workflow and (semantic web) annotations for bioinformatics,” in *Workshop on Data Derivation and Provenance, Chicago*, 2002.
- [67] R. Ikeda and J. Widom, “Data Lineage: A Survey,” technical report, Stanford University, 2009.

- [68] P. Groth and L. Moreau, "PROV-Overview." [Online]. Available: <http://www.w3.org/TR/prov-overview/> [Accessed 31 July 2014], Apr. 2013.
- [69] L. Moreau, B. Clifford, J. Freire, J. Futrelle, Y. Gil, P. Groth, N. Kwasnikowska, S. Miles, P. Missier, J. Myers, B. Plale, Y. Simmhan, E. Stephan, and J. V. den Bussche, "The Open Provenance Model core specification (v1.1)," *Future Generation Computer Systems*, vol. 27, pp. 743–756, June 2011.
- [70] K. Belhajjame, R. B'Far, J. Cheney, S. Coppens, S. Cresswell, Y. Gil, P. Groth, G. Klyne, T. Lebo, J. McCusker, S. Miles, J. Myers, S. Sahoo, and Curt Tilmes, "PROV-DM: The PROV Data Model," in *W3C Recommendation* (L. Moreau and P. Missier, eds.), World Wide Web Consortium (W3C), Apr. 2013. [Online–Accessed: 31 Jul. 2014].
- [71] K. Belhajjame, J. Cheney, D. Corsar, D. Garijo, S. Soiland-Reyes, S. Zednik, and J. Zhao, "PROV-O: The PROV Ontology," in *W3C Recommendation* (T. Lebo, S. Sahoo, and D. McGuinness, eds.), World Wide Web Consortium (W3C), Apr. 2013. [Online–Accessed: 31 Jul. 2014].
- [72] T. J. Jankun-Kelly, K.-L. Ma, and M. Gertz, "A Model for the Visualization Exploration Process," in *Visualization, 2002. VIS 2002. IEEE*, pp. 323–330, 2002.
- [73] A. Woodruff and M. Stonebraker, "Supporting fine-grained data lineage in a database visualization environment," in *Data Engineering, 1997. Proceedings. 13th International Conference on*, pp. 91–102, 1997.
- [74] D. P. Groth and K. Streefkerk, "Provenance and Annotation for Visual Exploration Systems," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 12, no. 6, pp. 1500–1510, 2006.
- [75] C. E. Scheidegger, *Provenance of Exploratory Tasks in Scientific Visualization : Management and Applications*. PhD thesis, The University of Utah, 2009.
- [76] Schlumberger, "Petrel E&P Software Platform." [Online]. Available: <http://www.software.slb.com/products/platform/Pages/petrel.aspx> [Accessed 29 September 2013].
- [77] J. Cheney and S. Soiland-Reyes, "PROV-N : The Provenance Notation," in *W3C Recommendation* (L. Moreau and P. Missier, eds.), no. April, World Wide Web Consortium (W3C), Apr. 2013.

- [78] Halliburton, “DecisionSpace Geophysics.” [Online]. Available: <https://www.landmarksoftware.com/Pages/DecisionSpaceGeophysics.aspx> [Accessed 4 December 2014], 2014.
- [79] M. Saggaf, “A Vision for Future Upstream Technologies,” *Journal of Petroleum Technology*, vol. 60, Mar. 2008.
- [80] C. Ma and J. Rokne, *3D Seismic Volume Visualization*, vol. VI, ch. 13, pp. 241–262. Norwell, MA, USA: Springer Netherlands, 2004.
- [81] J.-L. Mallet, “Seismic interpretation: an introduction,” in *Numerical Earth Models*, ch. 3, pp. 53–78, EAGE Publications bv, 2008.
- [82] A. Kaufman and K. Mueller, “Overview of Volume Rendering,” in *The Visualization Handbook* (C. D. Hansen and C. R. Johnson, eds.), ch. 7, pp. 127–174, Elsevier Butterworth-Heinemann, 2005.
- [83] Society of Exploration Geophysicists, “SEG Y rev 1 Data Exchange Format,” Tech. Rep. May, 2002.
- [84] K. Gaillot, “The SEG-Y Format for Geophysical Data.” [Online]. Available: <http://walter.kessinger.com/work/segy.html> [Accessed 1 June 2010], 1994.
- [85] IBM Corporation, “Extended Binary Coded Decimal Interchange Code (EBCDIC).” [Online]. Available: <http://publib.boulder.ibm.com/infocenter/zos/v1r9/topic/com.ibm.zos.r9.adms700/adms7a05158.htm> [Accessed: 1 June 2010], 2007.
- [86] A. Brown, *Interpretation of three-dimensional seismic data*. Soc of Exploration Geophysicists, 2004.
- [87] T. Randen and L. Sonneland, “Atlas of 3D Seismic Attributes,” in *Mathematical Methods and Modelling in Hydrocarbon Exploration and Production* (A. Iske and T. Randen, eds.), vol. 7, pp. 23–46, Springer Berlin Heidelberg, 2005.
- [88] R. E. Sheriff and L. P. Geldart, *Exploration Seismology*. Cambridge University Press Cambridge, 2nd ed., 1995.
- [89] L. T. Ikelle and L. Amundsen, “Petroleum Seismology.” [Online]. Available: <http://petroleumgeophysics.com> [Accessed 1 June 2010].

- [90] T. Höllt, *Visual Workflows for Oil and Gas Exploration*. PhD thesis, King Abdullah University of Science and Technology (KAUST), 2013.
- [91] E. L. Etris, N. J. Crabtree, and J. Dewar, “True Depth Conversion: More than a Pretty Picture,” *CSEG Recorder*, vol. 26, Nov. 2001.
- [92] D. Gao, “3D seismic volume visualization and interpretation: An integrated workflow with case studies,” *Geophysics*, vol. 74, pp. W1–W12, Jan. 2009.
- [93] P. Tu, A. Zisserman, I. Mason, and I. Cox, “Identification of Events from 3D Volumes of Seismic Data,” in *Image Processing, 1994. Proceedings. ICIP-94., IEEE International Conference*, vol. 3, pp. 309–313, IEEE Comput. Soc. Press, Nov. 1994.
- [94] N. Keskes, P. Zaccagnino, D. Rether, and P. Mermeij, “Automatic Extraction of 3D Seismic Horizons,” in *SEG Technical Program Expanded Abstracts*, pp. 557–559, Society of Exploration Geophysicists, 1983.
- [95] H. G. Borgos, T. Skov, T. Randen, and L. Sonneland, “Automated Geometry Extraction From 3D Seismic Data,” in *SEG Annual Meeting*, pp. 1541–1544, Society of Exploration Geophysicists, Oct. 2003.
- [96] H. G. Borgos, T. Skov, and L. Sønne land, “Automated Structural Interpretation Through Classification of Seismic Horizons,” in *Mathematical Methods and Modelling in Hydrocarbon Exploration and Production* (A. Iske and T. Randen, eds.), vol. 7 of *Mathematics in Industry*, pp. 89–106, Springer Berlin Heidelberg, 2005.
- [97] M. Faraklioti and M. Petrou, “Horizon Picking in 3D Seismic Images,” *Image Analysis*, vol. 2749, pp. 216–222, 2003.
- [98] A. Blinov and M. Petrou, “Reconstruction of 3-D Horizons from 3-D Seismic Datasets,” *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 43, pp. 1421–1431, June 2005.
- [99] D. Patel, C. Giertsen, J. Thurmond, J. Gjølberg, and E. Gröller, “The Seismic Analyzer: Interpreting and Illustrating 2D Seismic Data,” *Visualization and Computer Graphics, IEEE Transactions on*, vol. 14, pp. 1571–1578, Nov. 2008.

- [100] S. Hawtin and D. Lecore, "The business value case for data management - a study," tech. rep., CDA & Schlumberger, 2011.
- [101] C. Bond, A. Gibbs, Z. Shipton, and S. Jones, "What do you think this is? Conceptual uncertainty in geoscience interpretation," *GSA Today*, vol. 17, no. 11, pp. 4–10, 2007.
- [102] G. D. Kidd, "Fundamentals of 3-D seismic volume visualization," *The Leading Edge*, vol. 18, pp. 702–709, June 1999.
- [103] L. Castanie, F. Bosquet, and B. Levy, "Advances in seismic interpretation using new volume visualization techniques," *First Break*, vol. 23, no. 10, pp. 69–72, 2005.
- [104] P. M. Silva, M. Machado, and M. Gattass, "3D Seismic Volume Rendering," in *International Congress of The Brazilian Geophysical Society*, no. 1, (Rio de Janeiro, Brazil), pp. 14–18, 2003.
- [105] Halliburton, "GeoProbe Volume Visualization." [Online]. Available: <https://www.landmarksoftware.com/Pages/GeoProbe.aspx> [Accessed 15 September 2013].
- [106] Schlumberger Information Solutions, "Petrel 2007.1 Release Notes," 2007.
- [107] Schlumberger Information Solutions, "Petrel 2008.1 Release Notes," 2008.
- [108] Schlumberger Information Solutions, "Petrel 2009.1 Release Notes," 2009.
- [109] F. Alvarez, P. Dineen, and M. Nimbalkar, "The Studio Environment: Driving Productivity for the E&P Workforce," white paper, Schlumberger, Jan. 2013.
- [110] B. Goodway, "Introduction to this special section: 4D," *The Leading Edge*, vol. 33, pp. 130–133, Feb. 2014.
- [111] J. G. F. Stammeijer and P. J. Hatchell, "Standards in 4D feasibility and interpretation," *The Leading Edge*, vol. 33, pp. 134–140, Feb. 2014.
- [112] J. Lazar, J. H. Feng, and H. Hochheiser, *Research Methods for Human-Computer Interaction*. West Sussex, United Kingdom: John Wiley & Sons, 2 ed., Sept. 2011.

- [113] Y. Breitbart, A. Deacon, H.-J. Schek, A. Sheth, and G. Weikum, “Merging application-centric and data-centric approaches to support transaction-oriented multi-system workflows,” *SIGMOD Rec.*, vol. 22, no. 3, pp. 23–30, 1993.
- [114] O. Vogel, I. Arnold, A. Chughtai, and T. Kehrler, “Introduction,” in *Software Architecture*, ch. 1, pp. 1–21, Springer Berlin Heidelberg, 2011.
- [115] O. Vogel, I. Arnold, A. Chughtai, and T. Kehrler, “Architectures and Architecture Disciplines (WHAT),” in *Software Architecture*, ch. 3, pp. 39–64, Springer Berlin Heidelberg, 2011.
- [116] L. Bellatreche, S. Benkrid, A. Ghazal, A. Crolotte, and A. Cuzzocrea, “Verification of Partitioning and Allocation Techniques on Teradata DBMS,” in *Algorithms and Architectures for Parallel Processing* (Y. Xiang, A. Cuzzocrea, M. Hobbs, and W. Zhou, eds.), vol. 7016 of *Lecture Notes in Computer Science*, pp. 158–169, Springer Berlin Heidelberg, 2011.
- [117] P. Valduriez, “Shared-Nothing Architecture,” in *Encyclopedia of Database Systems* (L. Liu and M. T. Özsu, eds.), pp. 2638—2639, Springer, 2009.
- [118] C. Ballinger, “The Teradata Scalability Story,” tech. rep., Teradata Corporation, 2009.
- [119] M. Stonebraker, D. Abadi, D. J. Dewitt, S. Madden, E. Paulson, A. Pavlo, and A. Rasin, “MapReduce and parallel DBMSs: friends or foes?,” *Communications of the ACM*, vol. 53, pp. 64–71, Jan. 2010.
- [120] R. Wang, “Teradata Indexes.” The University of Arkansas, [Online]. Available: <http://teradata.uark.edu/research/wang/indexes.html> [Accessed 13 August 2013], 2000.
- [121] C. Silva, Y.-j. Chiang, W. Corrêa, J. El-sana, and P. Lindstrom, “Out-of-Core Algorithms for Scientific Visualization and Computer Graphics,” in *Course Notes, IEEE Visualization 2002*, no. October, 2002.
- [122] J. S. Vitter, “External memory algorithms and data structures: dealing with massive data,” *ACM Comput. Surv.*, vol. 33, pp. 209–271, June 2001.
- [123] T. Akenine-Möller, E. Haines, and N. Hoffman, “Polygonal Technique,” in *Real-Time Rendering 3rd Edition*, ch. 12, pp. 531–574, A. K. Peters, Ltd., 2008.

- [124] T. Akenine-Möller, E. Haines, and N. Hoffman, “Acceleration Algorithms,” in *Real-Time Rendering 3rd Edition*, ch. 14, pp. 645–695, A. K. Peters, Ltd., 2008.
- [125] G. Davis, *Learning Java Bindings for OpenGL (JOGL)*. AuthorHouse, 2004.
- [126] D. Shreiner, “Texture Mapping,” in *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Versions 3.0 and 3.1*, ch. 9, pp. 389–488, Addison Wesley, 7 ed., 2009.
- [127] R. S. Wright, N. Haemel, G. Sellers, and B. Lipchak, *OpenGL SuperBible: Comprehensive Tutorial and Reference (5th Edition)*. Addison-Wesley Professional, 5 ed., July 2010.
- [128] T. McReynolds and S. Hui, *Volume Visualization with Texture*, ch. 13, pp. 144–153. SIGGRAPH, 1997.
- [129] R. Rost, B. Licea-Kane, D. Ginsburg, J. Kessenich, B. Lichtenbelt, H. Malan, and M. Weiblen, *OpenGL Shading Language*. Pearson Education, 2009.
- [130] J. Selan, “Using Lookup Tables to Accelerate Color Transformations,” in *GPU Gems 2: Programming Techniques For High-Performance Graphics And General-Purpose Computation* (M. Pharr and R. Fernando, eds.), GPU Gems, ch. 24, Pearson Addison Wesley Prof, 2 ed., 2005.
- [131] E. M. Lidal, *Sketch-based Storytelling for Cognitive Problem Solving*. PhD thesis, University of Bergen, 2013.
- [132] J. Rubin, D. Chisnell, and J. Spool, “Skills for Test Moderators,” in *Handbook of Usability Testing: Howto Plan, Design, and Conduct Effective Tests*, Wiley, 2008.
- [133] G. R. Gibbs and C. Taylor, “How and what to code,” in *Online QDA Web Site*, 2010. [Online]. Available: http://onlineqda.hud.ac.uk/Intro_QDA/how_what_to_code.php [Accessed 1 December 2014].
- [134] L. Moreau, “PROV-XML: The PROV XML Schema,” in *W3C Working Group Note* (H. Hua, C. Tilmes, and S. Zednik, eds.), World Wide Web Consortium (W3C), Apr. 2013. [Online]. Available: <http://www.w3.org/TR/2013/NOTE-prov-xml-20130430/> [Accessed 26 December 2014].

- [135] B. J. Kadlec, G. a. Dorn, H. M. Tufo, and D. a. Yuen, “Interactive 3-D computation of fault surfaces using level sets,” *Visual Geosciences*, vol. 13, pp. 133–138, June 2008.
- [136] M. Hadwiger, P. Ljung, C. R. Salama, and T. Ropinski, “Advanced Illumination Techniques for GPU Volume Raycasting,” in *ACM SIGGRAPH Courses Program*, (New York, NY, USA), pp. 1–166, ACM, 2009.
- [137] M. Zwicker, H. Pfister, J. van Baar, and M. Gross, “EWA splatting,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 8, pp. 223–238, July 2002.
- [138] M. Botsch and L. Kobbelt, “GPU Splatting,” in *Point-Based Graphics* (M. Gross and H. Pfister, eds.), The Morgan Kaufmann Series in Computer Graphics, ch. 6.2, pp. 271–293, Elsevier Science, 2011.
- [139] W. Schroeder, K. Martin, and B. Lorensen, *The Visualization Toolkit: An Object-oriented Approach to 3-D Graphics*. Prentice Hall, 2 ed., 1997.
- [140] A. Dietrich, E. Gobbetti, and S.-e. Yoon, “Massive-Model Rendering Techniques: A Tutorial,” *IEEE Computer Graphics and Applications*, vol. 27, no. 6, pp. 20–34, 2007.
- [141] L. D. Floriani, E. Puppo, P. Cignoni, and R. Scopigno, “Level-of-Detail in Surface and Volume Modeling,” *EUROGRAPHICS*, 1999.
- [142] D. Weiskopf, *Visualization of 3D Scalar Fields*, vol. vi, ch. 2, pp. 11–79. Springer Berlin Heidelberg, 2007.
- [143] M. Oneppo, “High Level Shader Language (HLSL),” in *SIGGRAPH*, 2007.
- [144] University Of Münster, “Voreen - Volume Rendering Engine.” [Online]. Available: <http://www.voreen.org/> [Accessed 1 September 2013], 2013.

Appendix A

Usability Evaluation Questionnaire

The following pages include scanned copies of written responses from three anonymous participants from academia. These responses were used for the case study presented in Section 7.2. Also, they partially contributed to the survey presented in Section 4.7.

A.1 Sample A

Usability Evaluation Questionnaire

[Scenario 1] Single-user multi-interpretations

A. Using Petrel:

1. How important is to have a history-tracking feature?	<div>Not Important</div> <div>0 0 0 0 0 0 0 0 0 0</div> <div>1 2 3 4 5 6 7 8 9 10</div>
2. Were you able to perform history tracking in Petrel?	<div><input type="checkbox"/> Yes</div> <div><input checked="" type="checkbox"/> No</div>
3. If your answer to Q. 2 was "NO", how would you workaround it?	<div>save multiple time-labelled</div> <div>copies</div>
4. Where do you usually, in an institute, save a "master" or "approved" interpretation?	<div>in the Petrel project</div> <div>e.g. final folder</div>
5. How challenging is it to collect the "master" interpretation of your project using the existing software?	<div>Not Challenging</div> <div>0 0 0 0 0 0 0 0 0 0</div> <div>1 2 3 4 5 6 7 8 9 10</div>

B. Using the prototype application (FESVo):

1. Compared to the mature application (Petrel), how accurate did you see the visual result of the datasets?	<div>Not accurate</div> <div>0 0 0 0 0 0 0 0 0 0</div> <div>1 2 3 4 5 6 7 8 9 10</div>
2. Were you able to perform history tracking?	<div><input checked="" type="checkbox"/> Yes</div> <div><input type="checkbox"/> No</div>
3. How easy was it to retrieve earlier versions of the interpreted horizon using the prototype application?	<div>Very Difficult</div> <div>0 0 0 0 0 0 0 0 0 0</div> <div>1 2 3 4 5 6 7 8 9 10</div>

[Scenario 2] Multi-user multi-interpretations

A. Using Petrel:

1. How important is to collaborate on features interpretations?	Not Important <input checked="" type="checkbox"/> 1 0 2 0 3 0 4 0 5 0 6 0 7 0 8 0 9 <input checked="" type="checkbox"/> 10 Very Important
2. Were you able to pass your project data?	<input type="checkbox"/> No <input type="checkbox"/> Yes (features only) <input checked="" type="checkbox"/> Yes (features & survey)
3. What do you usually share in a collaborative scenario? How?	<i>Interpretations (horizons, faults) Project Reference t-models</i>
5. How challenging is to collaborate on seismic interpretation using the existing software?	Not Challenging 0 1 0 2 0 3 0 4 0 5 <input checked="" type="checkbox"/> 6 <input checked="" type="checkbox"/> 7 0 8 0 9 0 10 Very Challenging

depends on project complexity & duration

B. Using the prototype application (FESVo):

1. Were you able to immediately view the other user's interpretations?	<input checked="" type="checkbox"/> Yes <input type="checkbox"/> No
2. How easy was to change the other user's interpretations?	Very Difficult 0 1 0 2 0 3 0 4 0 5 0 6 0 7 0 8 0 9 <input checked="" type="checkbox"/> 10 Very Easy
3. Were you able to perform history tracking and view the very first (original) interpretation and then user's amendment?	<input checked="" type="checkbox"/> Yes <input type="checkbox"/> No

A.2 Sample B

Usability Evaluation Questionnaire

[Scenario 1] Single-user multi-interpretations

A. Using Petrel:

1. How important is to have a history-tracking feature?	Not Important <input type="radio"/> 1 <input type="radio"/> 2 <input type="radio"/> 3 <input type="radio"/> 4 <input type="radio"/> 5 <input type="radio"/> 6 <input type="radio"/> 7 <input checked="" type="radio"/> 8 <input type="radio"/> 9 <input type="radio"/> 10 Very Important
2. Were you able to perform history tracking in Petrel?	<input type="checkbox"/> Yes <input checked="" type="checkbox"/> No
3. If your answer to Q. 2 was "NO", how would you workaround it?	<i>save different projects</i>
4. Where do you usually, in an institute, save a "master" or "approved" interpretation?	<i>share network drive</i>
5. How challenging is it to collect the "master" interpretation of your project using the existing software?	Not Challenging <input type="radio"/> 1 <input type="radio"/> 2 <input checked="" type="radio"/> 3 <input type="radio"/> 4 <input type="radio"/> 5 <input type="radio"/> 6 <input type="radio"/> 7 <input type="radio"/> 8 <input type="radio"/> 9 <input type="radio"/> 10 Very Challenging

B. Using the prototype application (FESVo):

1. Compared to the mature application (Petrel), how accurate did you see the visual result of the datasets?	Not accurate <input type="radio"/> 1 <input type="radio"/> 2 <input type="radio"/> 3 <input type="radio"/> 4 <input type="radio"/> 5 <input type="radio"/> 6 <input type="radio"/> 7 <input checked="" type="radio"/> 8 <input type="radio"/> 9 <input type="radio"/> 10 Very accurate
2. Were you able to perform history tracking?	<input checked="" type="checkbox"/> Yes <input type="checkbox"/> No
3. How easy was it to retrieve earlier versions of the interpreted horizon using the prototype application?	Very Difficult <input type="radio"/> 1 <input type="radio"/> 2 <input type="radio"/> 3 <input type="radio"/> 4 <input type="radio"/> 5 <input type="radio"/> 6 <input type="radio"/> 7 <input type="radio"/> 8 <input type="radio"/> 9 <input checked="" type="radio"/> 10 Very Easy

[Scenario 2] Multi-user multi-interpretations**A. Using Petrel:**

1. How important is to collaborate on features interpretations?	Not Important <input type="radio"/> 1 <input type="radio"/> 2 <input type="radio"/> 3 <input type="radio"/> 4 <input type="radio"/> 5 <input type="radio"/> 6 <input checked="" type="radio"/> 7 <input type="radio"/> 8 <input type="radio"/> 9 <input type="radio"/> 10 Very Important
2. Were you able to pass your project data?	<input type="checkbox"/> No <input checked="" type="checkbox"/> Yes (features only) <input type="checkbox"/> Yes (features & survey)
3. What do you usually share in a collaborative scenario? How?	<i>shared network drive</i>
5. How challenging is to collaborate on seismic interpretation using the existing software?	Not Challenging <input type="radio"/> 1 <input type="radio"/> 2 <input type="radio"/> 3 <input type="radio"/> 4 <input type="radio"/> 5 <input type="radio"/> 6 <input type="radio"/> 7 <input type="radio"/> 8 <input type="radio"/> 9 <input checked="" type="radio"/> 10 Very Challenging

B. Using the prototype application (FESVo):

1. Were you able to immediately view the other user's interpretations?	<input checked="" type="checkbox"/> Yes <input type="checkbox"/> No
2. How easy was to change the other user's interpretations?	Very Difficult <input type="radio"/> 1 <input type="radio"/> 2 <input type="radio"/> 3 <input type="radio"/> 4 <input type="radio"/> 5 <input type="radio"/> 6 <input type="radio"/> 7 <input type="radio"/> 8 <input type="radio"/> 9 <input checked="" type="radio"/> 10 Very Easy
3. Were you able to perform history tracking and view the very first (original) interpretation and then user's amendment?	<input checked="" type="checkbox"/> Yes <input type="checkbox"/> No

A.3 Sample C

Usability Evaluation Questionnaire

[Scenario 1] Single-user multi-interpretations

A. Using Petrel:

1. How important is to have a history-tracking feature?	Not Important 0 0 0 0 0 0 0 0 0 0 1 2 3 4 5 6 7 8 9 10
2. Were you able to perform history tracking in Petrel?	<input type="checkbox"/> Yes <input checked="" type="checkbox"/> No
3. If your answer to Q. 2 was "NO", how would you workaround it?	copying a surface before making changes to it
4. Where do you usually, in an institute, save a "master" or "approved" interpretation?	within a folder in a project
5. How challenging is it to collect the "master" interpretation of your project using the existing software?	Not Challenging 0 0 0 0 0 0 0 0 0 0 1 2 3 4 5 6 7 8 9 10

can use project reference root

B. Using the prototype application (FESVo):

1. Compared to the mature application (Petrel), how accurate did you see the visual result of the datasets?	Not accurate 0 0 0 0 0 0 0 0 0 0 1 2 3 4 5 6 7 8 9 10
2. Were you able to perform history tracking?	<input checked="" type="checkbox"/> Yes <input type="checkbox"/> No
3. How easy was it to retrieve earlier versions of the interpreted horizon using the prototype application?	Very Difficult 0 0 0 0 0 0 0 0 0 0 1 2 3 4 5 6 7 8 9 10

[Scenario 2] Multi-user multi-interpretations

A. Using Petrel:

1. How important is to collaborate on features interpretations?	Not Important 0 0 0 0 0 0 0 0 0 0 1 2 3 4 5 6 7 8 9 10
2. Were you able to pass your project data?	<input type="checkbox"/> No <input type="checkbox"/> Yes (features only) <input checked="" type="checkbox"/> Yes (features & survey)
3. What do you usually share in a collaborative scenario? How?	<u>Projects... Reference book</u>
5. How challenging is to collaborate on seismic interpretation using the existing software?	Not Challenging 0 0 0 0 0 0 0 0 0 0 1 2 3 4 5 6 7 8 9 10

B. Using the prototype application (FESVo):

1. Were you able to immediately view the other user's interpretations?	<input checked="" type="checkbox"/> Yes <input type="checkbox"/> No
2. How easy was to change the other user's interpretations?	Very Difficult 0 0 0 0 0 0 0 0 0 0 1 2 3 4 5 6 7 8 9 10
3. Were you able to perform history tracking and view the very first (original) interpretation and then user's amendment?	<input checked="" type="checkbox"/> Yes <input type="checkbox"/> No

Appendix B

Survey Summary

The following pages include a summary of the responses from staff of oil and gas companies who were known to the author of this thesis. The survey was implemented using Google Forms, part of Google Docs¹. The summary report on the following pages was exported, unaltered, from Google Forms.

¹Google Docs is an online office suite offered by Google, <http://docs.google.com/>.

28 responses

[View all responses](#)

[Publish analytics](#)

Summary

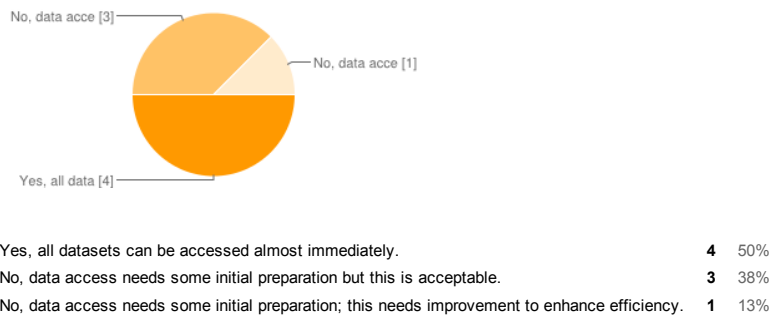
What is your role?



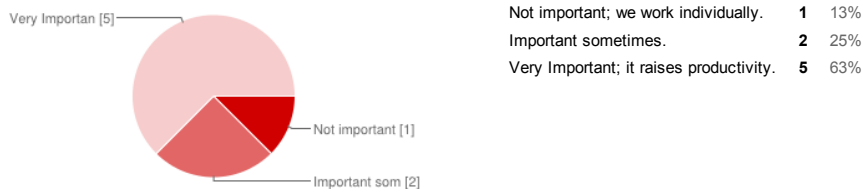
Questions for Users (Geoscientists & Engineers)

Data Interaction

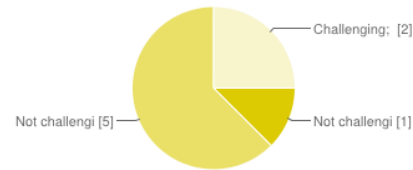
Do you perceive that access to subsurface data is highly available for visualization and/or interpretation?



How important is it to have a collaborative visualization and interpretation environment in E&P?



How challenging is to visually collaborate on subsurface data?



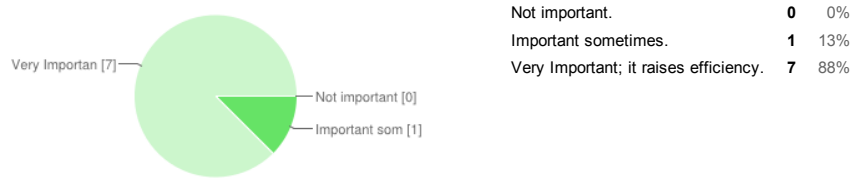
Not challenging; we can immediately share our results.	1	13%
Not challenging to share results with team members but challenging with other teams (e.g. due to their use of different software).	5	63%
Challenging; sharing our results with any other staff members requires some time consuming tasks (i.e. export/import).	2	25%

How do you perceive that interaction with data can be made in a way that maximizes your productivity?

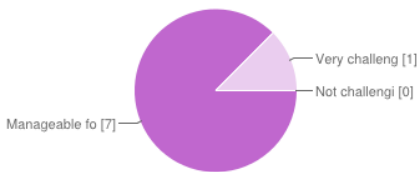
Better data management that control authority and ease accessibility As a data vendor, visual interaction with data is very important.
Early decisions can be made, i.e. go/no go before moving data to local in-depth interpretation environment.

Data Provenance

How important is data provenance (history and origin tracking) for the E&P industry?



How challenging is it to track data history?

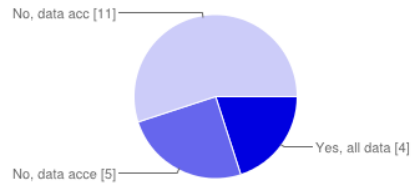


Not challenging; we can immediately track how a feature object was interpreted overtime and can easily access very old interpretations.	0	0%
Manageable for recent interpretations but a bit challenging for very old datasets.	7	88%
Very challenging; I perceive this area needs improvement.	1	13%

Questions for IT Staff

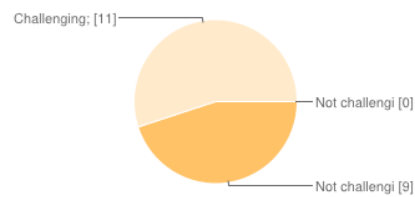
Data Interaction

Do you perceive that users in the E&P can currently access all subsurface data for visualization and/or interpretation in a high availability fashion?



Yes, all datasets can be accessed almost immediately.	4	20%
No, data access needs some initial preparation but this is acceptable.	5	25%
No, data access needs some initial preparation; this needs improvement to enhance efficiency.	11	55%

How challenging is for users to visually collaborate on subsurface data?



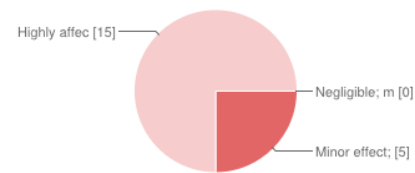
Not challenging; users can immediately share their results.	0	0%
Not challenging to share results with team members but challenging with other teams (e.g. due to their use of different software).	9	45%
Challenging; sharing users' results with any other staff members requires some time consuming tasks (i.e. export/import).	11	55%

How do you perceive that interaction with data can be made in a way that maximizes users' productivity?

I think cloud based collaboration is key. The cloud technology has proven to be efficient in very complicated industries like the creative industry in general. There is no reason that it can't work in oil/gas industry. . All data needs to be contained within one centralized database, preferably within a GIS environment. This will take much work, but is truly the only way to share data and information seamlessly. H Search for "Hadoop" or BigData Organize access by asset. By having shared databases among all disciplines Starting with a standered format and standered input and output between the interpretation apps This might be fart fetched but the time that our love for the company and what we do goes first that our egos that might be the time that will definitely really try to create solutions that interact with each other instead of patching solutions that need some sort of manual manipulation. The fact that solutions are created from several sources (form the business, all the IT outfits you can in the company, etc) some due to frustration that the right outfit does not come with the goods or because it ticks the box, will not help us out to reach the goal of Data Integration. Sometimes the simplest solution is the best solution. I continuously see over engineered answers to problems that at first seem to solve the problem, however end up being very expensive and nobody wants to use it. There is nothing wrong with the big picture, however there are short term and long term goals. If we create a solution, first with the VALUABLE insight of what the user actually needs, and then we divided it into phases to deploy, we might get somewhere in delivering a fit for purpose, integrated piece of work. A virtual master data store coupled with a web based interpretation package. By having more integrated data environment . Try as much as possible to standardize the workflow between all users.

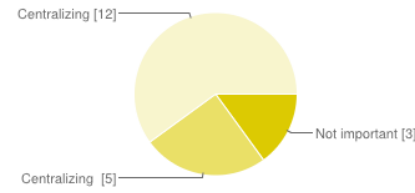
Data Architecture

How would you evaluate the negative effect on moving data between applications?



Negligible; moving data between applications does not effect productivity.	0	0%
Minor effect; it consumes a little time and manpower.	5	25%
Highly affects productivity; this issue needs a great attention.	15	75%

How useful would it be to centralize subsurface data?



Not important; there is no added value by centralizing subsurface datasets.	3	15%
Centralizing subsurface datasets might improve data management.	5	25%

Centralizing subsurface datasets would definitely improve data management, but has been challenging up to now. **12** 60%

What prevents E&P from centralizing datasets and thus avoiding movement of data?

There are many reasons that might make them slower to move toward that solution: 1. price of oil. Oil companies are making huge profits given the current oil prices, therefore there is no incentive for them to improve their internal workflows. 2. Fear of central point of failure: whenever there is a solution that kind of working, everybody feel afraid of experimenting with other solutions, there is a big fear of having a central point of failure with the centralizing datasets. I think could be the security of the information; different geophysical engineers like different programs and are often rewarded based on individual success and not a team effort, so often they will hoard or not want to share. No standard data format and no solution to do the data transfer. Cost and lack of geospatial technology awareness at the executive level. Know how Fear of change Data security Performance. Using multiple applications and databases. Y Data acquisition. - De-centralized E&P outfits. - Different regions and type of business in the regions - Different ways of working in different places in the world - budget and get the right Resource - Negative reaction to change. Our two main interpretation platforms require setting up proprietary project databases. They cannot directly work from a common central repository although their are tools to help move data between platforms. Some applications require their own data stores. Connectivity and network issues, security. We do have remote visualization solutions in our cooperation but there is a lot of challenges making users trusting a machines under their desks is a way more safe bet. tedious work, no special team or procedures to do so. Mainly due to security reason. different teams work on different projects. Lack of well integrated environment. Project databases. The lack of strict adherence to a data model by a company and software developers. If a data model were strictly used, connectors (add-ins) can be developed to transform data from a master data set to applications. Businesses have their own business rules that require customization of a standard data model. Different domains employ radically different data structures to match their own needs. Attempts to impose consistent definitions across the whole of E&P have always (and will always) fail Defining a centralised data set would be a retrograde step

General Questions

What are the main challenges in E&P data management?

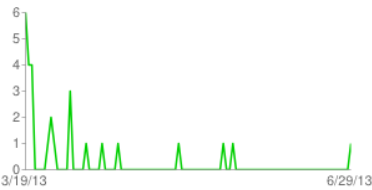
In seismic, maintaining the relationship of the interpretation to a particular version of processing can be difficult. The sheer volume of seismic forces companies to have "near" data sets that are easily accessible and distant sets that require time to place online. Integration and optimization of the infrastructure. Data integration. Cleaning up old unnecessary data, network bottlenecks. People behavior. Central Data governance. Integration and efficient collaboration. Managing multiple, proprietary data models. Managing and grading multiple interpretations, i.e. each user may have a different legitimate interpretation, there is no single 'right answer' so the system needs to manage multiple interpretations and allow users to set precedence for which interpretations are trusted more than others. 1. unifying terminology. 2. convincing multiple engineers from multiple to one universal workflow. 3. many oil companies work together. for example, one might handle exploration, another will do the analysis. bringing them all to a common ground can be a challenge. Scattered repositories Varying security constraints Users willingness Limited /very busy manpower efficient standardization. Taking data from various sources with unreliable completeness and accuracy and making this information useful to a broad audience within an organization. Different perspectives of the same data that requires data to be viewed from a different taxonomy. Size and amount of data being received. This is actually called the three V's in Big Data concepts (Volume, Velocity, Variety). Data delivers 25-33% of the value generated by oil companies but executives assign less than 1% of the budget to managing it. - Governance - Entitlement & obligations of the data (E&O) - Not enough responsibility taken from the business for point 1 & 2 - Lack of integration between of platforms globally. It should be the rule not the exception - Not enough recognition to the importance of E&P Data management as an asset - Not enough recognition to the Professionalism of Data Managers - Lack of established. Industry standards. There are some like SEGY, Las but still every software comes with their own solution interoperability of data and applications and few in this area that will help export/import data and with limited vendors there is less competition and can be expensive for companies smaller in size. Having a data governance and clear business process

What would you do to have an optimum data management in the E&P?

central data location where all data is Q.C. and a workflow on the project is implemented with commentary, etc so that all information around the decisions around the well were documented saved and later searchable. Start with mapping the business process and put a data governance. U Base changes in the budget on the value delivered. management support Unified operational definitions in a single glossary Proper awareness of data management benefits Partnership between users and IT support. Rather than sending huge files over the network and perhaps taking an hour to do so, we could utilize them wherever they are by executing a query that returns results in minutes if not less. - Get the support from top to bottom of the strategy and vision of Data Management for E&P - Have strong Resource base: Geologist, petroleum engineers, that want to do E&P Data management, so they can talk the same language as the business they are - Clear roles and responsibility in the organization in the Data management outfits in IT or E&P. Clear Focal points in each area. - After that then implement that a proper line of communication between those outfits in such a way that whatever solution comes out of them is an integrated one. - Deliver what the business needs within the Data Management strategy - Be more

visible for the business. Show where you are contributing to their business value. Even in small scale - Create a clear environment of understanding what are the roles and responsibilities of the E&P business in ownership of the data and roles and responsibilities of the Custodianship of the E&P data form Data Management. These is still grey....So Governance again - Be present at the table when E&P projects are being Framed or Developed Empower and accountability. Roles and responsibilities Having a standard data repository and make as a requirent for any new acquired application to be integratable with the repository. Unify workflows, elemenate bottle necks Providing single database for all applications and disciplines One cloud-based, high-end environment with friendly user interface that allows access to data from all the engineers/system analysit on E&P organization. Break down the silos that exist within organizations and build an enterprise GIS capable of handling the data load of today and tomorrow. Build this within an open-source environment with cloud based data storage. . A master data store for all data with add-ins to transform the data to a particular application. make efficient standards and strict procedures. Data governance at every level.

Number of daily responses



Appendix C

Interviews Transcript

This appendix includes a complete transcription of the interviews conducted to evaluate the outcome of this PhD; the interviews were discussed in Section 7.3. Sensitive information such as company names and names of individuals were removed from these transcripts. Interviews 1 and 2 were fully recorded such that their transcripts include the demonstration of the prototype application and discussions prior to the main questions. Interview 3 was only recorded during the time of the main questions.

C.1 Interview 1

Interviewee 1 has a background in geology and more than 9 years of experience in the oil and gas industry. Symbols in below dialog are: I1 – the interviewer (author of this thesis); R1 – the respondent of interview 1; and I2 – a third person (advisor to this research). Text of respondent is shaded.

- I1** [00:00] I think we earlier described the project, but I will quickly go over that.
- R1** My understanding of the project—the project is a database architecture linearly scalable for projects that will grow through time, not application specific or project specific, just generally being able to store a certain type of data—access it through—but it is outputted through normal industry standards—can interface with anything, those things can query in a two-way communication with that database.
- I1** The focus or the application of this research is seismic interpretations, as I mentioned previously, we are introducing a provenance of interpretations through visualizations. So allow multiple users, through multiple sessions, to contribute in the process of creating these interpretations, and also to enrich the metadata behind these interpretations.
- R1** Oh; this is something I did not appreciate at the beginning. So you will allow multi-users to work on the same database. Are they going to generate multiple new data types that they are all linked to see and update?
- I2** Yes, not necessarily asynchronously. You might be re-do the interpretations. There are different steps You have a lineage of all the steps to the person who have done that.
- R1** And will it keep track of all the different changes?
- I2** That is the idea.

- I1** So a single horizon object is actually a collection of all the contribution by different users.
- R1** And will that metadata be automatically generated?
- I1** Some of them yes. That is the addition and deletion process. But assessment—for example senior users looking at these interpretations—so that not automatic; that entered manually input by the user—so they assess other's interpretation when, for example, have conflicts on a certain area of a horizon, for example, then a senior staff can assess these interpretation and approve one of them.
- I2** Some of it will be, as Aqeel said, this is the competent person's opinion on what is right based on what has been presented to him or her, but also as part of the provenance there will be simple things like numerical uncertainty so you know that these source location were well constrained in this part of the survey and less constrained here. All that will go through the provenance as well.
- R1** And all that will stay with the horizon no matter what happened? Or stay with an interpretation—
- I2** There is a relationship back to it; so you can always pull it through when you need to see it.
- I1** Just to go back to provenance, when I speak with geologists I believe that is a terminology is geology called provenance. So just to be clear here, we are talking about the computer science provenance.
- R1** and the definition here is?
- I1** [5:00]the origin of the data.
- I2** the other name for it is data lineage.
- I1** It is also important to mention here that we are not dealing with file like the conventional applications. So a horizon, for example, is not a file; it is not an object. It is a collection of points.
- R1** I was going to ask this question. How is the metadata assigned—because you are saying that a horizon is a collection of points—I completely understand that—and then it is visualized as gridded horizon but is stored as a set of points. So does each of this point has all the metadata? Or does it have a unique ID that takes it to a metadata file?
- I1** [6:12] The reference here is the location and a horizon ID. So all metadata go along with this combination of location and ID.
- R1** Does not that—say that my horizon had ten thousands points then you duplicate your metadata ten thousands times.
- I1** Yes, this is the current implementation.
- R1** So you actually wasting space rather than having one item of metadata, like having a king metadata for that data.
- I2** Yes, so there are shortcuts
- R1** What I am saying is that each point has all the metadata duplicated on each point, or is a collection of point so they are referenced back to one metadata file?
- I1** The current implementation is that each metadata you add is a point. So if we have 5 metadata
- R1** So that will slow your system down when you scale massively, say to 10 million points, even if the metadata is tiny.
- I2** Yes, there is going to be an overhead—there is a cost. But the value then is that you persist—
- R1** No, I am not concerning myself with the cost. I am just more thinking about when you scale it up—the computational cost or even the storage cost; tiny little files starts to add up.
- I1** So the aim of this research is to provide users with the ability to enrich seismic data in a collaborative environment in order to trace back the origin, reliability, and credibility of interpretations. So that is the focus of the project.
- R1** So one other point—or one other question—if a data item is duplicated—it is quite commonly in our industry if somebody comes to look at something, he or she will make a copy of what is done and then they will modify that copy either—they will make a copy of the interpretation, deletes some parts of it, and updates other parts of it. Will the previous metadata stay with a duplication of that data? Because currently on our ... that we use now, that is not the case. So I can go and—I know this is not the focus of your work on the end application, but if I go to Petrel and I copy and paste a product and then I convert a horizon into points and back to an interpretation, it will look like that I created this interpretation. But actually what I want to see is where they come from. Because all what I will be given is this set of interpretation and when I am looking at them I can say instantly that these were generated from they get upset somebody copied their work ...

- I1** [11:17] So that is one case where the architecture can show who contributed what. Another case, let's say a reservoir engineer or a modeler wanted to check back the reliability of an interpretation, then they can immediately get all the information about who assessed the data and how confidence they were. So what I am going to demonstrate to you now is a prototype application. Do not expect any sophisticated rendering on the user interface like Petrel, for example. I will demonstrate a small seismic dataset with few preloaded horizons. You would be able to read all the metadata, make some slice changes like delete, make a time-shift alteration—this is an example to demonstrate how a simple change is sent back to the database—.
- I2** This is a static correction
- I1** then you can trace the history, filter, highlight, showing who did what.
- I1** [Demonstration Started] This is a seismic cube with a preloaded. They are stored in a fine-grained format in a database. There is a virtual Teradata database running here.
- R1** So this data is loaded from a SEG Y then—.
- I1** It is converted from a SEG Y to rows. Each trace is stored in a row, and each of these points is stored in a row. What we see here is—. This is a horizon. Three different persons contributed in this interpretation process. If we clicked on history, we can see that—. Baseline means that this is the very first interpretation. Then user A did some insertion, then user B came and did more insertion. If we go back—.
- R1** Insertion is just adding more point?
- I1** Yes. If we go back to this point, we can see up to what use A did. Now let's see the baseline.
- R1** [15:00] So these are history points you are calling them. At what frequency they are taken? As in time. If I am working on this, I will do some interpretations then when I am on auto-tracking then I will do some alterations. So there might be hundreds of iterations. So will I end up with hundreds of history points?
- I1** We did not define that. We could for example create a commit button.
- I2** Or call it a milestone button. Or snapshot every 5 minutes or at the end of the session. Anything can be enabled.
- I1** As you move with your mouse you can see the location of this point within this dataset and also the geocoordinates of that point. You can see who created this interpretation, and that it was assessed by Duncan with a confidence level of 70%. If we moved here then we can see that this point was created by user A and was assessed as 30%. If we go here then that is—.
- R1** The assessment of confidence level how was that put in there?
- I1** We will demonstrate that.
- I2** The idea of it was that firstly there will be a numerical confidence straight from data quality
- R1** We have this confidence value in the auto-tracker ...
- I2** You can then harvest that as well and add that into the index. But there can also be once you get into a review meeting and looking at it, you can say that we are not happy with this area because conceptually we do not understand the It is an arbitrary number you give to it. You either very happy about it, you give it 95%. It could be 1 to 10, or percentage of whatever.
- R1** If you had an auto-generated confidence factor, give it an integer number like that, but then a user confidence factor. Probably best to go low, medium and high.
- I1** That is the user (on the screen). As Duncan said, we can add the quality value from
- R1** I will probably not use an integer value from the user. Some will say 95%, others will say 70%. Is 70% high, or is that a medium. So a low, medium and high is probably better. Because that is what we use when making comments on segment maps, where you have low, medium and high of risks on different things. Giving 3 options is much easier than 100 options.
- I1** Here you can do some highlighting. You can say show me what user A did. Or I need anything above 50% of confidence value. Of course, the list of properties can grow. So we can have the quality from the survey shooting. That highlights the areas of with confidence value 50% or more.
- R1** So the area that is highlighted meets these criteria?
- I1** [20:40] Yes. Let's clear highlighting. You can say discard any area with less than 50%; so you just want to see 50% or more of confidence. Let us say you are a modeler and you just want to work on high confidence interpretations.
- R1** Just want to mention that the highlight was not clear. It did not make the area stands very well.

- I1** I will now close this session and open a different horizon to play with. This is a different horizon. You can by the way load multiple horizons. Let us make some assessment here. You can draw an area and say I am confident about this interpretation—, say 90%. Queries are now running to the database to give all of these points 90% value of confidence. If we clicked here (on the area that was assessed), then we can see that it was assessed by user X as 90%. If we clicked somewhere else, then we can see that there is no assessment. So shall we start with the questions now?
- R1** [25:40] OK.
- I1** How would you evaluate the usefulness of this research into the oil and gas exploration workflow.
- R1** Of course the examples you show is just one set of examples that could be implemented. You can obviously implemented any type of metadata. But it would be good if that can be taken from other data types. Say for example if you had a petroleum system model and then you were easily able to tag a horizon with what you just showed and then just when you hover over your mouse, you do not have to bring multiple ..., once that data cross referenced between the data types—.
- I1** Something like well data?
- R1** Well data will be good, but also cultural data, or as Duncan said earlier, the 4D seismic data if you could quickly see on your horizon. I want to see whether there is a 4D difference between these two volumes, populate this horizon and show me what areas are different. Do a query, and highlight those areas between the two cubes
- I2** Why they are different? Is it real or is it numerical artifact, or is there an uncertainty here so you cannot tell if there is a different or not.
- R1** Exactly. In that difference show me the confidence in that shot receiver spacing, roll out those areas so that you can quickly start to do multi-level queries on the data.
- I2** It is implicit in this approach, because we load data into a table in a relational database, it is not tight to a particular data type. So we can put in a petroleum system model, or we can put in a reservoir simulation model.
- R1** That is because it is all spatially referenced?
- I2** Yes. So you can link a horizon pick to a reservoir model
- R1** What happens if the grids between different data types are differently aligned? Petroleum systems model are coarser grid cell size than native seismic data. You do not have anything of a finer grid than your seismic data.
- I2** Apart from your borehole measurement.
- R1** Oh, yes.
- I1** This has not been addressed.
- R1** I think it (the PhD project) is defiantly very useful. I think the crock is the user interface into interrogating that database. You want to make it user friendly, but I know that is not purpose of this PhD. In order for end users to use it that are not numerically gifted, or people that like working on databases, they need to be able to interface with that data easily. The kind of things you showed me here are good. But you have to make custom queries, and each query will be unique in the way that is built up. So simple work for people where they can see—. Just click on the different at the things they Just drag them in I want to query shot receiver spacing, I want to query 4D effect, I want to query an interpreter—different types of data, then the ranges, show me that. Or, some of the staff that is coming out of the plug-ins providers for Petrel, ... for Blueback, they did a geo-data investigator which will automatically cross plot any properties against any properties. It will just make this screen of cross plots. You then visually see what has relationships. So that may be a good way of visualizing your datasets. It might be worth looking at—.
- I2** [explaining a drawing] So Aqeel's stuff watches all the stuff back and forth. So what you are looking at the moment is just the things that look around aside ... Talks to that bit, but the idea is to handle all the traffic.
- R1** [32:40] So it will be just observing ... and just making some kind of metadata database. But the metadata will be stored back in here. Then that will be streamed. What I have seen from Blueback, not specifically looking at confidence factor, this is more looking at ... versus fluid factors Looking at two properties against each other, and the relation ship between them. And that kind on visual look at the metadata would be very useful.

- I2** The idea of Blueback is that— because they see all of Petrel projects, they are trying to come up with some data mining ... but data mining only has limited value What you want is ... then look across all of this and understand who did what and when and where as well; so more context around it.
- R1** I think it (the PhD project) has an intrinsic value. Just need to be quick, or not noticeable.
- I2** Or work behind the scene.
- R1** Because as soon as it starts slowing down—. Obviously, the metadata would need to be put it at each of these levels when a data item is made. When will the metadata be put? Or will you have a software that is ... with ECLIPSE (reservoir simulation software), when a user is doing something, it puts metadata in. Because unless you know what is happening in ECLIPSE, it does not matter what is pushed back here. You can only capture what is outputted and pushed back to the database.
- I2** [35:22] ECLIPSE is possibly a bad example, as it is not as interactive as the other tools. So, you will always have a synchronization issue. The high degree of interactivity ..., the closer to the user the data has to be. So, you need to try to keep up with the user. You can think of a lot of use cases where that will be challenging, like when a user is on a plane working remotely and then have to synchronize. I think the scope of this at the moment is at the team level rather than at individual user. It will be for the asset team on a production field, rather than ... production style work.
- I1** We can, for example, cache all changes behind the scene then update the database. That will be unnoticeable to the user. They can continue working—.
- R1** You built here a tool to visualize and interrogate the data. What you are showing me with your tool is that you can visualize and interrogate and classify the data. But the end product visualization was not really the scope of you PhD. In order for this to be used, you would have to have a front end. Whether that is in an existing product as a plug-in or a stand-alone front end where this data is being attached to the point, the metadata is being attached, but the user in Petrel does not realize that this metadata is being attached, for example. Then somebody else goes in and interrogate the database through a specific application.
- I2** We did talk about XML schema for components and provenance.
- I1** This can be passed from one user to another. For example, there could be many versions of this horizon. Then you want to make a selection of the production, or the approved, version of this horizon, then you can save this an XML. This is the selection criteria.
- I2** This could then be maintained by We also talked about, and this is a future work, a python library that enables that type of connectivities between the back-end and the user interface.
- R1** So the XML file you are talking about would just be—. Say if I ... QC the horizon. That horizon won't necessarily have metadata attached to it. Or would still have metadata attached to it?
- I1** [40:02] It will still have metadata attached to it.
- R1** But it would have a separate XML file?
- I1** Yes, which contains the selection criteria.
- R1** So why this won't be aligned with the metadata, if the metadata was still attached to that?
- I1** Because the XML is a selection by the user for what data is approved. So it could be for example, the final approved version of a horizon is the work done by Aqeel plus the work done by Duncan but ... only what has been assessed by person X with a medium confidence and above. So it is a mix of criterias.
- R1** But that can still be within the metadata—.
- I2** But if it is in an XML, you can read it and application could read it in time. But if it was stored in the schema as metadata, you then have to maintain something that can read it—
- R1** While the data is in the database, it has all the metadata. When the data in Petrel, it still has all of that metadata. When I export it out of Petrel in an industry compatible format, that is just in simplest x, y, z and ... but everything else is lost. If that will be put back in here (into the application), it would look like it is a brand new item. But from what you said earlier, your interface can flag up all of those points that are duplicate of another unique entity in there that had all the metadata. So it could then refund all the metadata. But if I sent it to another company, it won't have the metadata, unless you have the ability to have x, y and z and the XML file that has all the metadata. Then load it to the database.

- I2** This is an industry, not just an oil, problem, in which case a data management framework around it. You data life cycle says, or your data governor says, data always have to flow that way through this type of architecture. This is where your data governance starts here. You do not take something out and put it back at that end. Because the data manager won't know about it.
- R1** But even if I take this out here, then send it to company X, then they put it back in there, it would still have no metadata; and they won't have my metadata. I think this is a crucial part. Because this is irrelevant, or independent, of any application. So if from the database— if your tool you can say: I want to send— I finished my work, I QC'd it, I approved it, send it to our partners. Your tool that is watching what is going back and forth could say: OK, we will flag that. It comes out simple format for the horizon that is readable any where. Plus you have some kind of metadata, XML that selects metadata that you wish to share with a partner—.
- I2** This is a common problem. It is just not making one up. It is unavoidable as well if you are working in a joint venture where you are passing data around. You (I1) should consider and write few lines on how you would address what is known as “orphaned data.” It is easy enough to spot because it was relocated. But as [name of R1] says, you have lost all lineage on how it gone there. So you need to raise it as an issue with this. Someone then has to perform some kind of intervention to understand how to tie that back in to the provenance architecture.
- R1** [45:06] So within your company if you have got your piece of interface, architecture in place, you should be able to reconnect all that XML or that metadata within your company. But if it goes to another company, they will never be able to do that because they won't have the database and they won't have the architecture in place to spot that these are duplicate points. Therefore, there is a very high likelihood that this product is actually this in the database. Human nature is that, when somebody asks me for something, the easiest thing for me to do, is just export it and send it to them. The more difficult thing for me to do, is request the data manager— or me move it to a result project in Petrel world, then me move it to OpenWorks they (data managers) move it back into our database. They then export it, and send it to company X. At the end of the day, the person at company X just gets x, y, z. So it does not matter whether I send it to them or the data manger send it to them. To them, it does not matter. But to a company wishing to keep control of the database, it does matter.
- I1** So what is the benefit of export it and put it into OpenWorks?
- R1** The benefit—.
- I2** Just for Aqeel to understand this, [name of R1]'s world is company X— which is not that dissimilar to company Y or any of the other larger operators—you have an enterprise upstream data store, which is a whole collection of different databases—seismic data, production data, wells data—all lives in there, not very well joined up, but that is their referenced database. Then you have data that is kept in corporate data stores, which would be at country of region level. Then there will be at the asset team level, for separate fields. Then there will be project databases. So what we are looking out here is this bit.

- R1** So in order for us not to lose any knowledge, data from here needs to be moved back to here [pointing to the corporate database on a diagram] once it is approved. And that is the reason that if I am saying that it is approved and I want to send it to my partner, it is quite commonly tight to a decision gate in the capital value process. So when you are making a decision—or add a decision gate—to go from one way or another, you say: OK, that is the piece of data that is relevant to that decision point, if I ever want to look back why the decision is made and it needs to be in relation to this interpretation at this time, not the interpretation I am currently using 5 years later because that is not what the decision was made on. That is why it is important to put it back in there and for it to be archived as person A's approved interpretations of ... from ... on ..., for example. ... I sent it to the partner and said this is that, but if it has not gone back here, somebody in X number of years won't have seen it or won't ever exist or won't ever find it. That is why it is important to push data back to the database. As far as I am concerned, I tell my data manager to push it back. Then it is up to them to push it back through here. It should then come back out of here, and that should be available again in my reference projects that I can see. This part I really do not want to get involved with. I only see it as being a stop “between a rock and a hard place”. Trying to interact with creative people who do not really want to follow—, they want everything to be loaded yesterday. They do not understand why things take time. For example, they say I have got these 20 surveys, can you load them all now? I would personally say I won't be looking at this until this date, just load it some time between now and then; most people would say I want them now. We had some data from yesterday, from ..., it did not have correct— or the right level of detail of navigation of the seismic data. So our data loaded refused to load it, because one of the corner points could be up to 10 meters off. This was data from 1980s. It was a 3D survey but it covered an area which we had only 2D lines over. I said to them: “for what we want to do with it, it is significantly better spatially located than the 2D lines. Please load it and just put it with a flag saying that it is not QC'd and not planed a well on it.” They do not understand that there is a different fit for purposed level of data that we want to look at. To me, it is much better to have a 3D survey, even if they have uncertainty, your software or your architecture would be able to flag that and hold that with anything that is made out of this data whereas now the only way I am able to do that is if I look back at the name that was given to the data and it says not enough QC'd. So in cases like that I can see that your architecture would good at tracking that kind of things. So nothing gets lost. Somebody in 5 years would know what this is, and there is an interpretation over this area, let's use that. But you need to know that this was generated on something that had uncertainty that needs to be understood.
- I1** [52:41] That answered my second question, which is what cases you think the architecture would be useful at.
- R1** Particularly dealing with “vintage data” where modern level of accuracy are no longer able to be upheld because element of the data has been lost or mislead. Or they were known but they were never documented, because we did not have a thought of people will want to know ... in x years from now. That might be useful when we are processing it, but now what you get is a stack of the data that has inbuilt navigation; there is no separate navigation file. It would be good then to flag things up with a load uncertainty. Now, we can do that by putting it in the name of the cube (seismic data file); just add text to the name of the cube or text to the header. But then some end users— if I was given a cube, and got a week to interpret this, I would not go diving into the EBCDIC header. I would just load it into Petrel, see if it roughly came— does it come as I would expected to be? Is it the right way around. That would be looking at it in relation to a cultural data; may be comparing a paper map to what I see on my screen. I would not go and look at the header and see if there is a flag.
- I1** [55:45] What about the different roles— people with different roles, how this would benefit them? Let's say you are an interpreter, you want to pass your interpretations to a modeler, and then to the reservoir engineer to decide on where to drill. Would this type of generic and expandable metadata help them?

- R1** It should do. Because they should in theory want to interrogate different queries on the database, because different things are important to them. If for example automatically extracted computer petrophysical information onto the surface everywhere you had a well penetration, if automatically—so if a well comes through here on my petrophysics logs tells me that my calculated porosity at that point there ... is 20%, and if it then on the other wells does that, they could then query: show me the porosity distribution on the surface. That might not be something that the basin modeler would want to look at, but it would be something that the geological modeler or the reservoir engineer would want to look at. They will all be querying different things at the database. So if you could auto-generate as much metadata, but it depends if any of the metadata element would ever be looked at.
- I1** For a modeler, what metadata they have about the interpretation of the seismic?
- R1** All what they will know, from the way our files are named—they will know who did the interpretation, or who's name on that interpretation—That interpretation may be a merger of numerous people's interpretations and then updated. They will know, if the person named it correctly, the age of the horizon or the name of the horizon, and what they have picked. They will also know the domain that it was picked in: time or depth. They will know the year that it was done. They may know the project associated with that. They would know the seismic survey, not specifically the seismic cube. Depending how people have put in. We have these metadata generators for that name; the ability of imposing naming conventions. So you pick the ...; you put your interpreter ID in; you put the year; you tell it which seismic survey; and it put all of this into a profile name. Then that is sent at the simplest level. That is all the modeler will know—very high level simple things. They will have no confidence factor. Even if that is generated in Petrel, it might not be generated in OpenWorks or SeisWorks. They will get this data from numerous sources depending on where the project has been done. All what they will expect is some interpretation or a gridded horizon based on some interpretation. The gridded horizon will have the same name as the interpretation. But, for example, they won't know, unless they looked the files, the grid increments, the rotation of the grids—. You have to look the files to know this. So one thing that is evident in our company—and I am sure it is evident in another companies—some people—when I grid in Petrel, I grid based on the rotation and on the grid of the seismic nodes. Because then when I make an attribute map, I have a point on my horizon that is relevant or geospatially located exactly on a node in seismic. If my seismic was shot 12.5x12.5 but I gridded 25x25—If I can attribute my map on 25x25 grid of a 12.5x12.5 seismic dataset, I am essentially applying it ... smooth filter on the attribute map. Now, ... they don't understand that. They are making an attribute maps that are smoothed because of a choice they made when they gridded. Then you could say this map is smooth, but that map is not. You got a point every second point of the seismic data. Therefore, you got to interpolate between these two points and not using all the data you have. When you QC'ing others' people map, you could say: this map looks funny compared to the rest. That is because they put in different—. I am sure this happens every—.
- I2** [1:02:03] That sort of thing should be regardless of the right or wrong; ... needs to be applied But that is the sort of thing that needs to be persisted right through. At the end of the day, a production engineer who then has to say of all the possible well interventions: I am doing for the next weather window, this one is the most ... risk, because I have the best understanding of the uncertainties and risks around it, and going into this compartment has the highest reward for the best insight and tightest uncertainty around it. It helps to prioritize their work over.
- I1** So let's assume we built a plug-in to a mature application like Petrel, and to another modeler application. They talk to a single database. How useful would be for the modeler or the reservoir engineer to have all the rich or extra metadata?

- R1** I think it would defiantly be useful because then it should save the modeller time if they could query the database and say for example: are all these gridded horizons gridded on the same grid? Does that grid match the seismic data that I am going to potentially extract ... from, to populate on my modeled petrophysical data? Am I essentially going to train my model based on seismic input? If I am, I want my grid to be the same as my horizons. I want that to be a known grid spacing and consistent grid spacing— let's say a known grid spacing for each zone. If someone want to grid differently, that's their choice. They want to know that the inputs are what they expect them to be and not have to check each one them individually every time, if you can just query: show me for all these horizons what is the rotation angle and grid spacing, without having to look at each one of them individually. You may have 50 horizons; you don't want every time you build a model, or update a model, ... grid incorrectly. It should flag up; there is something wrong here. That is where I see the importance, or one level of importance there (i.e. the work of this PhD).
- I1** I think we talked a lot about my third question, which is about how you would expect this work into production. I think we covered that.
- R1** It is difficult, unless you make an interface with each application that ... going to start using metadata. People like Schlumberger are making this slightly easier for you. Because, let's take the Petrel platform as an example, they are making it easier and easier for the other applications to run as plug-ins into Petrel. Then once I no longer have to go from ... to ... (application names), I can do it on a plug-in in Petrel. Suddenly, as long as you got something that is capturing all of the metadata changes in Petrel, you are capturing all the metadata changes in RokDoc (petrophysical modeling application) through that plug-in.
- I1** Are you saying that—.
- R1** The advent of these Ocean plug-ins to Petrel actually makes it easy for your to capture the metadata. Because no longer I will take my data into RokDoc, I will use the RokDoc plug-in in Petrel to do the process I would have done in RocDoc natively in Petrel without taking the data out. So if you are just watching what is happening in Petrel, you will see that this data has been taken through the Petrel plug-in for RocDoc, this and this have been done, and you will capture all of that metadata. Whereas before you had have written an interface with RocDoc. Then capture all of that metadata
- I2** You then just need to talk to the Ocean layer because Ikon RocDoc has access to the API.
- I1** [1:10:30] I think you covered my questions. Anything you would like to add?
- R1** I think it is definitely an interesting subject and quite rightly it does not have a simple answer. If it did, someone would have do it.
- I1** Does it fill a gap in the workflow?
- R1** I think it does fill a gap in the workflow; it fills an important gap in the workflow. But back in my mind I cannot help and think unless it is semi-automated in what is capturing. If it requires the user every step to fill in every element of the metadata, it is not going to happen. But if it is automatically, say this was picked at this volume, tag, this was picked at this time, tag, when it is pushed back, the user usually has to select the decision gate that is in relation to but we have to do that now. But simple things is OK. This data has gone into this plug-in; this is ... tagged. At the moment, Petrel captures a history. If I run an operation on a surface it captures that in a history. But I then have to go into the history; it does not outputted in a searchable spreadsheet, but in a big list; have to open the columns really wide to actually read what has happened. It is usable but not user friendly. If someone copied and paste that horizon, everything is lost. So you have no idea where they came from. Simple things, say these two bits of data in my database are identical but got different names. Once they got different names, people thinks they are unique. It is difficult as an end user to go back and make different maps and things like that. Then they look the same. So your architecture should be able to flag up that they are identical.
- I1** We can flag, not just per point, but also the whole horizon since all of these points share a single ID. So we can flag the horizon ID.
- R1** We got loads of data that is just duplications; or duplication plus one point. That is part of my daily work when I saw a new project, I try to find out what actually the data is, what it represents and which is the more completed dataset. It is digging through. Now (with this PhD work) it is interesting.
- I1** Thanks a lot for your time. [end of interview]

C.2 Interview 2

Interviewee 2 has a background in geology and more than 12 years of a mixture experience between academia and the oil and gas industry. Symbols in below dialog are: I1 – the interviewer (author of this thesis); R2 – the respondent of interview 2; and I2 – a third person (advisor to this research). Text of respondent is shaded.

- I1 So when I mention the word provenance here we mean the computer science provenance, not the geological provenance.
- R2 That's good, yeah, I could imagine the [inaudible 00:25]. If you say that word to geologists it means something quite different, provenance, that's funny. Well, you look at second order of... [Voices overlap 00:33]
- I1 And, as I said, we don't just capture the metadata of who and when this piece of interpretation was created but also we allow other users, like senior geologists, to evaluate the limitations.
- R2 Okay.
- I1 So we have two cases where we think this would be valuable is, let's say, an interpretation was done ten years ago then you want to investigate exactly who and how this interpretation was done through.
- R2 Okay, yeah.
- I1 And also let's say that you've done an interpretation and pass it to modellers then I understand from the people, from the domain people I work with, that modellers don't get enough information on the interpretation.
- R2 Okay.
- I1 So with this work we believe that modellers will have enough information to make more accurate decisions.
- R2 Okay, yeah, yeah.
- I1 So what I'm going to show you here is a prototype application, I built it from ground up so don't expect anything complicated like [inaudible 02:00].
- R2 Good.
- I1 It's a small dataset just to serve the purpose of this research and we have implemented a few actions just to show that the actions are captured back to the dataset and can be then read back.
- R2 Okay.
- I1 So this is seismic cue.
- R2 So you wrote this viewer?
- I1 Yes.

- R2** Oh right, it's way beyond what I would be able to do in itself.
- I1** So we have some interpretations, they are imported from Petrel. So what you would be able to do here is read the metadata and get each point as you walk.
- R2** Okay.
- I1** So here we can see that this was created by user B on this date and it was assessed by...I think they [inaudible 03:14] the information, yeah, it was clicked.
- R2** So that's for each XYZ point contains its own...
- I1** Yeah, and this was assessed by Duncan with a confidence level of 70 per cent and, of course, that 70 per cent can be just high, medium or low, it's just different representations of...
[Voices overlap 03:30]
- R2** But how do you assign the confidence levels?
- I1** Yes, I'll go through this. And also we have different values here, so let's highlight this area just to...yeah, so we have highlighted [inaudible 03:53] because we have all the secondary data so we can say that, show me any assessments of, let's say, 50 per cent and above, and it shows that this area is 50 per cent or above.
- R2** Okay.
- I2** So we're quite clear to [inaudible 04:19] not to get too hung up on exactly how you do it, it's more a framework than an [inaudible 04:23] to capture and then disclose confidence data, the idea that an application would then deal with rules by reducing [inaudible 04:32].
- R2** A sign of confidence, alright, okay, because I know bits of software do ascribe confidence but I've never been that clear how they do it.
- I2** Yes, so we're sitting somewhere between the application layer and the vapour layer doing all the productivity...
[Voices overlap 04:45]
- R2** Yeah, yeah.
- I2** ...way between it.
- R2** Yeah.
- I1** So basically I just want to...30 per cent so that's a 30 per cent [inaudible 04:56] then we're moving it from the scene, so let me clear the highlight and say that's for a loading filter just to show me any interpretation, 50 to 100, that particular [inaudible 05:15] because it's...
- R2** Okay.
- I1** It doesn't have a high confidence level. This bit is 100 per cent, this bit is 70 per cent so you can do all the filtering. Now to do...let me load a different dataset to show how it's done.

- I2 Also you can do it saying where is all the stuff that [name of R2] is talking about where their confidence is really high or where all the stuff that's come from the [inaudible 05:45] model, confidence is low, it's not quite the [inaudible 05:50] type of scenario.
- R2 Okay, okay, yeah, yeah.
- I1 And because we have all this rich data you can do all types of filtering, like if we have...you know that Duncan is a senior geologist. Then I can just select to show me anything that was assessed by Duncan.
- R2 Okay.
- R2 Right, okay, that's been through peer review or something like that?
- I1 Something like that, yeah, yeah.
- R2 A competent person.
- I1 So this is a different piece of horizon.
- I2 Sorry, we're constrained quite tightly on not doing anything about the bare minimum of prototyping the front end because that's not...
- R2 Really what it's about, no, no, no.
- I2 Really what it's about.
- I1 So these areas were assessed, I forget why they analyse a lot of [inaudible 07:03] assessment. So we can...let's say that you're not happy with this part or you're happy with it, then you can get a selection of [inaudible 07:21] tiers or just...
[Voices overlap 07:23]
- R2 Yeah.
- I1 ...make another selection, you can then write it and say, okay, delete this bit. And when you say delete it's just tagged as deletion and the change of the activities you've done, so you can always go back in history, and I can show you this, or you can say, time shift. And time shift is just a simple correction.
- R2 Okay, and milliseconds up or down.
- I1 Milliseconds, yeah, up or down. This is just...it might not make sense geologically but it's just to show that...
- R2 It's one of the trackable perimeters.
- I1 Yeah, we can track the changes when a geologist [inaudible 08:02] make any changes then that goes back to being [inaudible 08:06] you can pull up, or you can make a confidence assessment.
- R2 Okay.
- I1 And, again, this is just one example of annotation.

- R2** So you could go in there manually and put your confidence assessment around an area of complex faulting and say, confidence level is 30 per cent, whereas outside of that area it's 100 per cent or 90 per cent.
- I1** Yeah.
- I2** So if this was ever used in a commercial sense there would be a [inaudible 08:34] that we'd feel comfortable about and there'd be a [inaudible 08:35] that would allow you to...
- R2** Just digitise around it, okay.
- I2** [Inaudible 08:39] through that and...
[Voices overlap 08:43]
- I2** ...say, right, all of that is higher confidence.
- R2** Yeah.
- I2** Or you could go into the [inaudible 08:50] and get the performance out of there.
- R2** Yeah, it does something like that, doesn't it?
- I2** Yeah. And then if you look at the application [inaudible 08:58] we have to have a nod in that direction of the application [inaudible 09:02] from an asset team's point of view to help you put it all back...
[Voices overlap 09:05]
- R2** Yeah, back together, yeah.
- I1** Yeah, I clicked on time shift [inaudible 09:11] correct here, let me create an ID for you. So you're [person name], right?
- R2** Yeah.
- I1** Okay, your new ID is [number], keep a record of this.
- R2** Yeah.
- I1** So in milliseconds let's say that we want to shift it and it might take a while because I'm using the local database version which is slower than the one I have in my office there which is a real database, this is just a virtual client.
- I2** [Inaudible 09:54]
- I1** Sorry?
- I2** [Inaudible 09:57] machine on here.
- I1** Okay, yeah, I think it's...
[Voices overlap 10:00]
- R2** I think it's shifted it.

- I1 As you see it goes into...
[Voices overlap 10:10]
- R2 Well, five milliseconds is very small.
- I1 So it was really small.
- R2 Yeah, yeah, five milliseconds will be imperceptibly small.
- I1 But you can see it here.
- R2 So these are the time stamps?
- I1 Yeah, so if you did basically any correction, so you did some deletion and then you did some insertion, you can go back, so let's just go back to the previous...
[Voices overlap 10:50]
- R2 Okay, so that was a deletion?
- I1 Yeah, so any correction is basically a deletion of that.
- R2 How long is the memory of the...well, not particularly this piece of software that you developed but how far back do you think provenance should go or what do you think is...ten years or something?
- I1 Yeah.
- R2 Yeah, yeah, yeah.
- I2 Yeah, so [inaudible 11:19]
- R2 Okay, 20, 30 years, okay, yeah.
- I1 So it's there so when you did the correction, when I did the correction, it [inaudible 11:34] then inserted an input so that's why you see that really. So it's just a [inaudible 11:41]. And also not just highlight confidence but you can also highlight, show me the bit that was done by [name of R2].
- R2 My user ID was [number].
- I1 Yes, so it shows me any bit that was created by...
[Voices overlap 12:04]
- R2 Okay, that was mine here.
- I1 Yes, that was yours. So this is the kind of...
- R2 Yeah, yeah, yeah, so you can conduct a witch hunt on this, it's very good.
- I1 So now...
[Voices overlap 12:20]
- R2 Yeah, basically, it's automatic blame gaming, I like it.

- I1** So now this horizon when it is extracted or when it is transferred the axis of this horizon only [inaudible 12:33] is transferred to models, they can see all the rich information of who did this, then they can go back to [name of R2] and call them.
- R2** I guess one useful thing in that would be is the interpretations that I've seen most change, I guess, because that presumably are the horizon levels which have most uncertainty and they're the ones, therefore, that you'd want to be at least most aware of or have flagged coming in, so top reservoir might be a problem but [inaudible 13:07] might not be. So presumably when you bring it into a different piece of software the reservoir model should be able to say, well, this is the one that we might want to have...
- I1** A question about.
- R2** Yeah, or a variety of statistic realisations of it rather than let's not bother with this one. Yeah, no, sorry, it's just me, I'm not...
[Voices overlap 13:24]
- I2** The other one that started springing up was working with faults as they start producing from wells...
[Voices overlap 13:32]
- R2** Oh god, yeah.
- I2** ...you get a better idea of where faults are, especially once you start reading faults.
- R2** Yeah.
- I2** You can see how the ideas have developed over the...
[Voices overlap 13:40]
- R2** Well, there's a very famous picture, isn't there [inaudible 13:44], I think it is, the paper in 1993 where they show three fault maps, one from 75, one from 85 and one from 92 or something, and it shows the updating of the fault map for the reservoir, top reservoir, and cormorant field, and they showed the changing in the well placements for the injectors and the producing wells. It's amazing, there's like this map and it's got two faults and then there's the next one that's got four faults and they're all striking in different ways, and then there's another one, it looks like somebody's hit a piece of toffee with a hammer.
- I2** What paper's that?
- R2** I can write it, it's like [inaudible 14:16] I think it's [inaudible 14:19] 99, I can send it to you, if you remind me I can send it to you. But it's a very impressive series and there's another version...
[Voices overlap 14:24]
- R2** Yeah, there's another one as well, I'll ask [person name] about that as well, which show the criticality of taking some of that information through. I mean the simplest metadata tag on those horizons is the fact that the earlier ones were done on two dimensional data and the newer ones were done on three dimensional data.
[Voices overlap 14:45]
- R2** So that's the biggest meta type difference which is underpinning that change to the interpretation. But I can send you a couple of papers, they're nice graphics, I use them in class and things.

- I2** He's got a couple of papers where he's referred to this as how interpretations change...
[Voices overlap 15:04]
- R2** No, this one's very good, yeah, and there's two, there's this and there's a more recent one which is really, really cool, really nice, simple images.
- I1** I have a paper I think from [inaudible 15:16] where they'd done a study with 200 interpreters, I think, interpreting the scene and they found that there was a high percentage of answers [inaudible 15:30].
- R2** Yeah, there's a recent study...
- I2** That was the Midland Valley...
[Voices overlap 15:32]
- R2** That was the Midland Valley but that's in cross-section, so a lot of those I took part in, a lot of that was just you were given a 2D cross-section, seismic section on paper and you had to interpret it, but these are cool because they're actually in map view which I think is even more interesting because the 2D thing is 2D and there's a lot of [inaudible 15:52] but if you take it out to the third dimension the Lord knows how difficult it would be if you had a series of paper sections you were trying to complete a 3D model for, yeah.
- I1** I think this one I referenced, I have reference to this.
- R2** Yeah, Bonder Tower 2000.
- I1** So let me ask you before we run out of time, so how would you evaluate the usefulness of this research of the oil and gas exploration?
- R2** I think it's...how do I think bits of it, well, some of it I've eluded to already, I guess. I guess flagging and pinpointing uncertainty in a more quantitative way rather than word of mouth, because I guess word of mouth would normally be, oh yeah, the top reservoir was always difficult to pick in block four. Whereas actually having something which had a quantitative measure of that uncertainty, an automated way of flagging up that uncertainty and where within that interpretation, geographically where there's the biggest problem will then allow you to focus your effort. Rather than saying, well, let's reinterpret top reservoir everywhere if there was some way of coding up on this grid where the highest uncertainty was then you'd focus your efforts there. Because if something's time sensitive you don't want to be spreading your effort across the whole of the time you have available to you, you want to say, well, I'm going to use all of my time just on this and everything else is 80 per cent and above confidence.
- I1** So do you mean since we have this model and architecture has the ability to flag or pinpoint then we could add some automatic mechanism to detect where the high uncertainty would be?

- R2** Yeah, yeah, exactly, yeah, so detecting it and quantifying it because you might look at it and say, well, this bit's less...is this bit more confident than this bit but even in the bit where we're less confident we're still 80 per cent confident so do we even need to bother, do you know what I mean, let's go and make tea and eat biscuits and not worry about it. Because it's all relative but there might be a threshold, 80 per cent and above we just don't bother reinterpreting this particular surface.
- The reservoir modellers might have a specific threshold they're happy with, I mean, God, I don't even know what the reservoir modellers [inaudible 18:29]. I mean in my experience, which is not vast, but in terms of working with production they will take the horizons as read, deterministically they will just have them as top and base reservoir or reservoir layers, they'll basically do whatever they need to do. The only time they'll ever try and update those is basically without any discussion with the geologists about whether some of the updates are geologically reasonable. So they'll say, well, the history match doesn't work so we're going to put a fault in here, or, we need to lose some volume here so we're going to bring the top of the reservoir down a little bit because the volumes that we're seeing in the wells is not that same.
- And I'm trying to think for where the production would just benefit from having bits of metadata flagged in here so that rather than having to go back to geologists they can actually look at it themselves, because if a piece of the interpretation's 100 per cent you shouldn't be changing it.
- I2** So from where they're working on the 4D work flow...
[Voices overlap 19:33]
- I2** ...they're all in the same room anyway so you don't have that disconnect that you...
[Voices overlap 19:40]
- I2** ...slower moving fields, they are in separate side lines.
- R2** Yeah, you're right, maybe if people are sitting together there's less of a problem for that, but still I think the power of what you're trying to do here is something to do with the...is quantifying confidence from certainly and making sure that gets passed down between an exploration unit, because one of the examples you used...Okay, so we appraised this acreage ten years ago, the political situation there was not good for us to proceed with the bid on the licence so we've come back ten years later, ten political situation changes, so we're still in the exploration stage but we still need to know something about the provenance, the interpretations we're either going to reuse or we're just going to scrap them and redo them again.
- I think most people would just map them again, people seem to be obsessed with making the new maps whereas actually if you could give them some confidence or there was some way of actually saying, well, actually this is done on the same data, there's high levels of confidence in it, don't bother, let's look...
- I1** And is designed by this person and he was well known and...
- R2** And it's designed by this person. Yeah, but that thing, I think is incorrect, I think that's probably the one...I think that would be where I disagree, this whole idea that there's some sort of quality assigned to an interpretation because of someone with seniority or experience I think is slightly dangerous. Because otherwise you come to a very hierarchical system which...I did my PhD in [country name] and it was the worst professional experience I've had, one of the worst professional experiences I've ever had where the whole system is based on the fact that somebody is older than you and, therefore, they know more than you, therefore, their interpretation is better than yours.
- And what that does is basically it means that you get this matriarchal interpretations which basically never change because it was done by the most senior guy, and then all these other interpretations are forgotten, they never are used because the most senior guy's already done it so why would you do it. So I think you'd have to be really careful in the meta tagging of the...

- I1 No, I think that...
[Voices overlap 21:43]
- R2 Because, in theory, you could have a meta tag which says, how many years' experience did the person have who did this interpretation? And I would just think that's complete craziness, I mean you should be able to look at the data yourself and see whether you think it's any good or not, I guess.
- I1 Yeah, I think so, I think that's left to the [inaudible 21:58] of the environment.
- R2 Yes, yeah, yeah, but I don't see...
[Voices overlap 22:05]
- I1 So the mechanism is there so it depends on how they're using it or how they wish to...
- R2 Yeah. I'm not sure how...I mean the most extreme end member version of this is that you don't meta tag names on things at all. I mean Petrel does, you go into settings, look at the information, it says, created by, accessed by, edited by, autotracked by, blah, blah, blah. You could just get rid of all of that provenance and not worry about who did it. I guess I've never looked in any of those things and thought about it.
- I2 I think that [inaudible 22:45] many people abstract it will say, it has been reviewed by, and then the [country name] case, it's been reviewed by [inaudible 22:53] or in the region...
- R2 It's the peer review, peer reviews, yeah.
[Voices overlap 22:58]
- I2 ...it has been signed [inaudible 23:03] or whatever.
- R2 Yeah.
- I2 However you do that that's not really your business, you just have to capture the number.

R2 Yeah, and I wonder as well when you get down to...okay, so you go through all this, you do the...I mean let's not think about [inaudible 23:14] let's think about exploration, so you basically do this appraisal, you have this interpretation that's bouncing around, you sit in a room, you're trying to calculate the chance of success in calculating volumes and risk and so forth, and you're sitting in that room and there's a decision to be made. But that point I just...again, my experience, which is not vast, by that point the uncertainties in the interpretation of it are long gone, they either want to drill that well or they don't.

It's like strategically they want to drill that well or they've got a rig or there's some external driver to make this happen, that all of that nuancing in terms of the agony you probably spent over the initial interpretation which is all couched in your meta tagging and has been thought a little bit about by the people lower down, by then it's not quantitative, it's more of a qualitative sensing, well, top reservoir's a risk because of the depth conversion, because the depth conversion in terms of velocities might be a problem. And by the time you get there it's just those things just feel like they're so far away from that final decision.

It might have some power then if everybody's talking about where you're going to place the well and everything, if then you could show people, well, actually where you're going to place that well is where there's a high degree of uncertainty on the interpretation, you code it up, here's the 30 per cent, this is only a 30 per cent confidence area and it can move by this and this and these two people said that. So just be aware of that as you make the decision, it's not a showstopper, you can't really put up these numbers in to give you a yes or no answer but you can certainly use them to change your gut feeling about how you proceed in the exploration sense.

And, likewise, in the production sense, you'd think it would be the same way, but there's nothing numerical you could really use to make you to do or not do something.

I2 I think what this would help with is it's going to be less of an impact where you're looking \$200 million exploration, but where you've got 20 options for \$20 million infill wells how would you rank them?

R2 Yeah, exactly.

I2 And what am I going to do for the next six months [inaudible 25:31] I've got 20 options and I can do four of them, this will then help sort them out.

R2 Yeah, yeah, even the exploration one though, it's still the same thing if you think about it, because \$100 million is a lot of money, right, and you've got ten \$100 million wells globally, some of which are in...let's say they're not all in frontier acreage but some of them are, so some of them could be big players, they could change the world's perception of this particular type of drilling system, or some of them are in the basin next door to one which is already producing. That sort of...I think it would be hugely important then still, it would still change or influence a decision knowing where the interpretation activity has come from.

I1 We should all [inaudible 26:18].

R2 Yeah.

I1 Yeah, but I'm not sure if I highlighted that [inaudible 26:26], it's not just a confidence or assessment or an evaluation that is tagged but in multiple versions could that be any help like, for example...

R2 Yeah, I mean, say, if you've got multiple versions...

I1 Yes, it's an A1 user who did this started doing some horizon on our...
[Voices overlap 26:42]

R2 Yeah.

- I1** Did this bit and it is limiting...
[Voices overlap 26:47]
- R2** So could you...and this is a question then of how you take it in, so the example in this case in production or exploration work flow you might want to capture that statistically. So if you're running volumetrics to try and work out how big a structure rather than just using this horizon to depth confirm CO volume, or this horizon which is yours, you could actually run multiple...I guess you sort of...do you capture that normally when you're doing volumetric calculations or do you use different input grids? We can, because normally what you do is you put in one grid and then you calculate uncertainties around that, well, those uncertainties wouldn't be vaguely based on statistical range, they'll be based on a range and you'd sample within that.
But you wouldn't necessarily take in another physical surface and do the calculations, again, statistically around that, which would make sense if you could do that, if you could take out ten different interpretations and do that assessment to see how this variability impacted the decision.
- I2** That's just about shell [inaudible 27:59] another few days of machine time to put it all in.
- R2** Yeah, because if you think about this in terms of a map view, like this difference in here might only be done in this corner, right? So, in essence, it doesn't really matter around here, this is where we have 100 per cent and this might be where we only have 30 per cent because there's a bunch of faults or something, let's say, in this area. So you might then want to be testing the depth conversion using this version, this version, there might another intermediate one as well that you will then test on.
I mean simply letting somebody know...here's an idea, here's an idea, letting somebody know that there's an alternative version might be useful, even if when somebody opens a horizon to do their reservoir modelling or their volumetric calculation and it flashes up saying, this is one of five versions of this grid, click here to see the other versions.
- I2** [Inaudible 28:55] next to it as well.
- R2** Well, yeah, even have the same person's name on it, that would be hugely valuable.
- I1** Because at the moment what's available is the first example I showed you is exactly this bit, so if one piece of the horizon was there then two divergences or two...
[Voices overlap 29:19]
- R2** Yeah, yeah.
- I1** ...two versions. Now in terms of storage, this is just technical, technically, whenever you have a new version it's not a wholly new...it's not a whole copy of the whole...
[Voices overlap 29:38]
- R2** Okay, yeah, yeah.
- I1** ...it's just the differences, so it's just this bit is tagged under your name, for example, and then this bit's tagged on [inaudible 29:45] my name.
- R2** Okay.
- I1** Yeah, and this will at the same time will have Duncan's name there. We can then all sit together and see all the other versions of the...
[Voices overlap 29:56]
- R2** Yeah, yeah.

- I1 ...this class and then prioritise them or bank them.
- R2 Yeah, I think that would be hugely useful because imagine going from Petrel to RMS or some reservoir modelling software, I've got all this reservoir model grids, I then export as XYZ, lose all the information associated with that grid and the nine other versions we have, I'll send it to you, you load it up into RMS, you start building your reservoir model, putting faults in, doing whatever. I mean just having some architect linking back from RMS to Petrel, well, I'm not being specific about those softwares but having one way of looking back and knowing there's all those other versions. Does that already exist, I don't think it does, does it?
- If you have something like what's that database called, Duncan, that Kingdom suite we're trying to use to [inaudible 30:54], what's it called? The one that we're trying to use to make their software more portable, what's it called?
- [Voices overlap 31:06]
- I2 OpenSpirit.
- R2 OpenSpirit, yeah, I mean does that do anything like that?
- I2 It allows you to connect to and export things like Open Works or Petrel but you have to know what you're [inaudible 31:22] for.
- R2 Okay.
- I2 Certainly it wouldn't be easy, certainly it wouldn't be the same functionality across all the applications, on an application to be able expose that information to OpenSpirit in the first place.
- R2 Yeah, I think that's the usefulness is definitely having...one is the confidence, the other one is the mapping of the changes...well, it's not even the mapping of the changes because they can get really granular with it, can't you, things like going down to the...but actually just telling somebody that there's four different top reservoir versions.
- I remember in [company name] there was a project I worked on once where there was 1,062 base crustaceous and conformity picks in a well and the standard deviation must have been about 20 centimetres on those picks. So the 1,00 picks [inaudible 32:17] 20 centimetres, why make that many, why have that on...it was never clear to me why so many people had taken...or why they'd done that [inaudible 32:24] of it all the way into that. It would have been good to know why that was done then to delete 1,061 of them.
- I2 [Inaudible 32:34] half an hour each, the number of man hours that are wasted.
- R2 Have got no idea what they were doing, yeah.
- I1 So I think you already included cases where you think the work would benefit the [inaudible 32:48] work flow, because that was my second question.
- R2 Yeah, one if the exploration, the other's the production side of things, but just the uncertainty would be good to track.
- I1 And how do you think this work can be extended to go into production, I think you mentioned...
- R2 Into production?
- I1 Yeah, because you mentioned the [inaudible 33:11] to calculate the...

- R2** Of volumes?
- I1** Of volume, yeah.
- R2** Yeah, I mean faults are...I mean we're talking about horizons a lot because you show the horizon, but in terms of fault position and fault uncertainty and whether a fault is sealing or not I think...again, knowing what data the faults were interpreted on. I think that would be useful in the production sense because in production when you are producing from the reservoir and that dynamic data is coming up to your desk and you're trying to make sense of it in terms of what you can't see in the subsurface, your job is then to manipulate the other horizons and grids to try and fit what you think is going on in the subsurface.
- I think one of the important things there is that those updates are done across all the platforms, because if you're the production geologist updating those grids and so forth you need to make sure that they're always being updating in the seismic domain and I don't think that ever happens. So what normally happens is I'll make an interpretation as a geophysicist say, I give it to the Duncan as the production geologist or the guys [inaudible 34:20] the reservoir model, he goes to the reservoir model, he updates it but there's not...because that data then is in depth, in metres or feet rather than milliseconds to a time, if he updates it there's no automation to actually use the velocities to go back to the time domain and update the initial seismic grid.
- So presumably you end up with a flourishing of data because you end up with an initial one or numerous time inputs and then out of that is born Lord knows how many depth grids, but none of that comes back into the time domain. I wonder if it should, maybe it should because if your well proves there's a fault somewhere that should be going back to the geophysicist who's then going to reinterpret the data.
- I2** Firstly, just going back to my point originally, going back in just to QC and make sure it makes sense... [Voices overlap 35:12]
- R2** ...logically, yeah, yeah, yeah, exactly.
- I2** And if it does then, yeah, you'd ought to revisit the original... [Voices overlap 35:16]
- R2** Yeah, exactly. So maybe there's some power in there in terms of...like it's a bit more two way than between the time and the depth domain, so between the two physical and geological domain would be quite useful. So in that respect a production geologist has some value for the production geophysicist because he's updating based on real geological data.
- I1** Yeah.
- R2** But, again, I'm saying this based on my experience, some companies might do this by default, the guy updates the reservoir model automatically, the geophysicist updates his, because they should be exactly the same, right, because the surface in the time domain should be the same as the surface in the depth domain given a certain set of velocities so they should never be different.
- I1** Physical... [Voices overlap 36:10]
- R2** Yeah, they should know the differences, yeah, they can have multiple versions themselves but the ones you're using to make the volume calculation should be the same, I think.

I1 Those were my main questions, so just a final point is about the different roles of [inaudible 36:42] so we assume that having the ability to capture multiple versions of [inaudible 36:53] interpretations would allow more collaboration, so more than one geophysicist can be working on interpreting the same datasets because of the [inaudible 37:09], would this help if you have...

R2 I don't know if you'd ever want two geophysicists interpreting the same data at the same time, personally I think it's a bit of a waste.

I2 They'd fight over it.

R2 I think it's a bit of a fight, I always make this point when we do an exercise here in class, we do this project on a field called [field name], and it's quite a fair sized seismic dataset and normally the teams assign two interpretatives to the dataset. So I always say to them, well, you have two options over how to split this because it's a very faulted reservoir, there's a few [inaudible 37:57] horizons in the top and base reservoir. Now what do you do, do you split it geographically, so you'd make one person do the east and one do the west and then they both have to meet in the middle with some degree of structural consistency?

Or...and that causes me problems, right, so what do you do, if you go that way you say, well, let's start in the middle together or let's map this bit together and then let's have this little stand where you've done it together and then we can push outwards. Because if you start outwards and work inwards you could imagine...

I2 [Inaudible 38:25]

R2 ...or the other option is you split it stratigraphically, so you map all the [inaudible 38:31] horizons and I'll map the reservoir horizons and, okay, that'll be fine. But that causes some other problems because there's you've got faults, right, and the faults are not stratigraphically bound, they go all the way down to the reservoir and all the way up to the [inaudible 38:45]. So everybody has to have consistency in terms of where their horizons are against the faults.

And neither things are particularly elegant solutions, there are ways of probably limiting some of the problems but I think in terms of sharing data between two interpreters I think it's difficult anyway. Your question being about whether this would help that collaboration, I can't really see how it would, I'm not really sure how it would help that collaboration. It might make people feel a bit closer to the...so if ten years on I get something from you and I look at it and I see all these tags of confidence and certainty [inaudible 39:33] it was done on, I might feel like I knew you.

I1 [Inaudible 39:36].

R2 Yeah, I'd feel like I knew you even though you've now left or done something else since then, you can get a window into what the person who was interpreting at the time was thinking. That might be useful in terms of almost like a long range collaboration but I don't know if it'll get the people sitting in the same room together, is that what you meant?

I1 Yeah.

R2 Getting people to sit in the same room is as much about having open plan offices and making people sit in sensible sized teams, and I've never been too sure about whether software can do that. Those people who just come into work, they sit at their computer, they go home normally, although having spaces where they all do that together is at least useful.

What would be cool is if you somehow had in your control project it was like Facebook where you were interpreting and updating, setting up a status update, [person name] has changed his [inaudible 40:33], see it live now, and you press the button and then you saw like a dashed version of their reservoir horizon come in or something. Or if you could think of some sort of...

- I2 You may laugh, I know someone who's actually written that.
- R2 No, I know, I am taking the .. slightly but something which was actually trying to bring in that more dynamic...that would maybe make people feel like they're collaborating and interacting more because everybody wants that kind of immediate fix of...if you've got that ping on screen, and it is like...there's experiments on this, when people have their email ping or the little envelope appears their adrenalin is released, their heart rate goes up, even if...
- I2 [Inaudible 41:18]
- R2 Exactly, even if it's not good.
- I2 He's moved it again.
- R2 Yeah.
- I1 I don't think what you've just mentioned is difficult because we already tried some [inaudible 41:30] machines, making different changes on the same horizon, because we're using the single database in the middle then the second user can really just refresh the screen and see if it changes...
[Voices overlap 41:43]
- R2 Yeah, but that would be cool though because if you're sitting on a production job doing something you could have a stream so there's somebody...
[Voices overlap 41:51]
- R2 Yeah, so one really good example I've seen lots and lots of time with this is...so if somebody's in the production environment so, let's think, some troll field in the North Sea, somebody's [inaudible 42:04] for quite a while, they're doing their thing, doing the reservoir modelling, they've got a reservoir geophysicist, they may be tweaking interpretation but simultaneously in another building or in another city or in another country even there's a bunch of explorationists looking at the same data, reinterpreting it for something shallower than the existing reservoir or something deeper. Even when they're doing that they're probably making an interpretation of the reservoir itself because it's just any horizon you pick.
Now if you could basically tell that person not to bother because there's a bunch of people in production who have done this 100 times already, or it flagged on this data stream for the production guys that somebody in exploration in Houston was updating their interpretation or doing an interpretation that was useful for you, that would be quite cool. But that would require a very, very rigorous set of names for particular horizons which, again, is a huge problem.
Because [company name] tried this when I was in [country name], there was like a standard way of writing horizon names, so it was like top Jurassic, it would be like T underscore Jurassic with a capital J, underscore, then you had to put your personal identification number. So there was a load of different codes and...
- I2 So this is like activating the file [inaudible 43:21].
- R2 And then basically nobody bothered doing it or some people did, some people didn't. But if the software had to force you to that I think that would be neat because then you could propagate that updating and those confidence factors and different things much more dynamically globally.
- . [End of transcript]

C.3 Interview 3

Interviewee 3 is a chief geologist with more than 25 years of experience in the oil and gas industry. His experience included oil and gas exploration data management. Symbols in below dialog are: I1 – the interviewer (author of this thesis); and R3 – the respondent of interview 3. Text of respondent is shaded.

I1 Okay. So can we start? So after I have demonstrated the prototype application which is an interface to the proposed provenance enabled interpretation pipeline how would you evaluate the concept and usefulness of this research to the oil and gas industry?

R3 I think the concept is very important for the oil and gas industry. Today several people are doing interpretations and not necessarily all of these interpretations are available for the end user. Sometimes part of them, sometimes they are not. The way they are stored in databases, the way they are stored in application databases can make them available but generally there is always changes in the interpretations applications which would, which makes the chance of having all the interpretations available for the user is limited, there's lots of limitations. So I think this type of work is going to help the industry and help the interpreters to find out who made the interpretation and he can even tap into it and he can see this interpretation and make use of it.

I1 Right, so can you provide us cases from the industry where you think it would benefit the end user? Maybe save time finding an interpretation, so I mean real cases, general cases I would say.

R3 As working in the industry, this kind of application, this kind of meter data availability is very important and almost it's in every company and every day the need is there. So what do we do? Sometimes we build models of seismic interpretations and they are in an X application.

I1 Yes.

R3 And then the company decided to use another application, Y application. So then if we did not move all the data and the interpretation from X application to Y application then there would be a gap, but who made an interpretation? Number two, several versions of interpretation, because the interpretation is an end interpretation, several versions of this interpretations are existing and interpreters normally lean to a certain side. They prefer this kind of interpretation for a reason or another. But it will pay them more benefits if they looked at all the interpretations, then they make a decision based on seeing everything. Looking at one side is forcing its bias in the interpreter to take that side. Seeing everything will give him a better judgement call. It will give him a better chance to make a decision. He may disagree with all of them and he'll start his own interpretation, but at least whatever questions in his head could be answered through seeing different interpretations.

I1 Okay, yeah. What do you think are the challenges that would face such an idea, research to be implemented in the industry?

R3 I think, I'm not sure

I1 The applications?

R3 they are, yeah, the applications. I'm not sure they are having similar system. They may have sort of systems but each of these applications, they just serve the users who use their application, use that data, that use their application, the meter reader for the user, or if the interpreter or the data manager meant to move other users into that application. Some of them just do not [inaudible 0:04:32] so what we need to do is either create a plug in, and the challenges to make this applicable and usable in all existing applications in the market. So you build in a plug in which can plug into all different applications, and number one, number two, the challenge, the real challenge is where is this data existing? So how can this application read meter data of pre-existing interpretations and make it available for the user? So

I1 So maybe the challenge is to migrate all data into this model?

R3 All data is open for this kind of, this application specifically

I1 Yes.

R3 brings the meter data of the users and brings the meter data of the users who made an interpretation and bring in their also interpretations at the same time. Now if a company who has been working for 30 or 40 years and maybe the past 20 years they have all of the interpretations existing in the computer and in several different interpretations, that, and how to collect all these interpretations and make it available in the current application is the challenge, some of the challenges, and I think it's a real challenge. So what we need to do is build a standard database where we can migrate all pre-existing data and meter data of the users, make available, put them in that database and then the application carried from that database, that would be a help.

I1 Oh okay, so migrate pre-existing interpretations into this model

R3 Mm-hmm.

I1 and then

R3 In another incident

I1 Yeah.

R3 the challenge is, companies moving from one application to another and not having all the data is a challenge by itself.

I1 So can you say a bit more about it?

R3 Which means if I take this application and I place it in Landmark product...

I1 Yes.

R3 and this company is using, moved from Landmark to Petrel

I1 Yes.

R3 then all they would have to migrate

I1 Yeah.

R3 in a form that this application can read, then it can be available in both applications and then the Petrel user can see all the work done progressively in...

I1 But once interpreted

R3 in terms of

I1 on the Landmark application.

R3 If you used dual frame then you move to Landmark then you move to Petrel then there is three different applications, and so on. If you used to use the geo frame, the mainframe and then you move to the applications, that's a third, another dimension of the challenge. All of this data has to be available to the system, and the system can read them.

I1 Okay, yeah, yeah. I get that, yeah. Yeah, I think that's it, that's the, thank you very much.

R3 Sure. [End of transcript]

Appendix D

Background on Rendering Techniques

This appendix starts with a brief comparison between surface and direction volume rendering techniques. Then it discusses the *texture mapping* technique which is adopted in the implementation of this work. Finally, it gives general background on the programming library used in the implementation.

D.1 Surface Rendering vs. Direct Volume Rendering

Objects with changing properties, such as computer tomography (CT) scans, fluid dynamics and seismic data), are rendered using either *surface rendering* or *direct volume rendering* techniques [139].

Surface rendering models an object with a surface description such as points, lines and triangles as illustrated in Figure D.1. Isosurfacing, one method of surface rendering, extracts points of equal values to render them as a surface of polygonal mesh. For seismic visualization, the *isosurface* rendering technique is used for visualizing as well as extracting and picking surfaces, mainly horizons and faults [80]. Marching cube technique is commonly used for generating the triangular mesh of isosurface objects. However, when dealing with a massive volume, issues with polygonal representation are encountered [140]. The generation and rendering of a large number of polygons, or triangles, are computationally expensive.

Direct volume rendering (DVR), on the other hand, displays a 2D projection by rendering a 3D object directly as a block of data without extracting a geometry. This method might be considered to be computationally intensive if a large volume is to be rendered as a whole, since mapping is recalculated for each change in the viewing angle. However, with the support from recent advances in graphics cards, direct

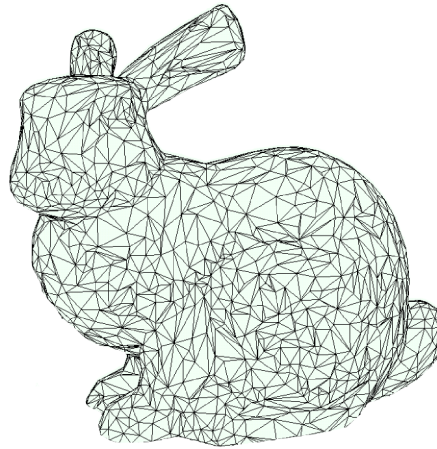


Figure D.1: A polygonal mesh representing a bunny. More triangles are required for a smoother surface and thus higher resolution. (*illustration by Floriani et al. [141]*)

volume rendering techniques have become highly efficient and interactive. Several techniques are used to conduct direct volume rendering including volume ray casting [136], splatting [137, 138] and texture mapping [128]. For seismic data visualization, *texture mapping* rendering techniques are commonly used and are further discussed in Section D.2.

D.2 Texture Mapping

Texture mapping is a direct volume rendering technique that has been very commonly used in visualizing scientific data, such as CT scans and seismic data. Its common use is due to its simplicity and fast computation that is supported by most graphics hardware. As illustrated in Figure D.2, 3D texture slices can be used to render a volume [128].

Texture slicing is originally implemented in 2D and later in 3D. In both cases, data is first sampled in 2D polygonal slices. These planes are then textured with the volume data and rendered in order from back to front. While rendering, data is interpolated by the GPU. The differences between 2D and 3D texture slicing are as follows [142]; refer to Figure D.3. In *3D texture slicing*, the slices are image-aligned, which means that they are aligned with the viewing angle. Interpolation in 3D texture slicing is performed in trilinear fashion, by the GPU as a built-in feature of 3D texture mapping. On the other hand, slices in *2D texture slicing* are aligned with the object regardless of the viewing angle. To cover all possible viewing angles, three stacks of textures are

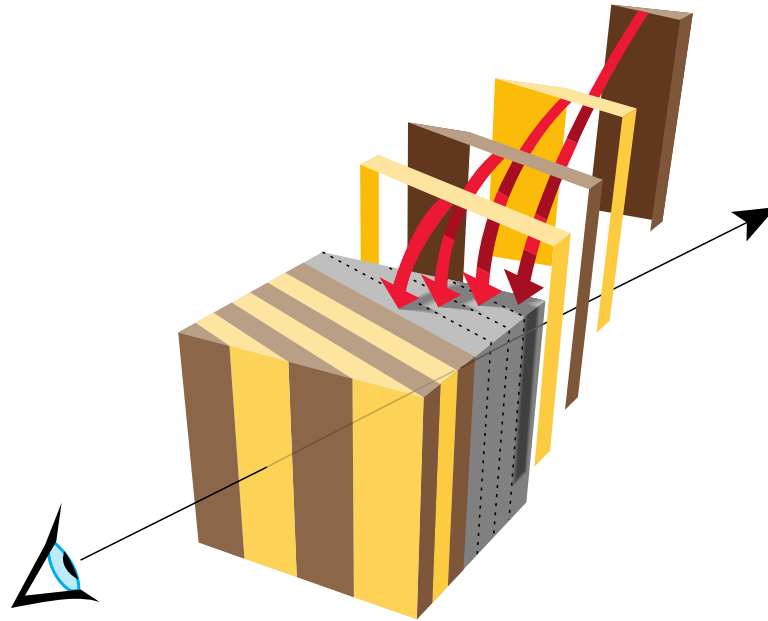


Figure D.2: An illustration of the *3D texture volume rendering* technique (*illustration by McCreynolds et al. [128]*).

generated, one for each coordinate axis. However, only one stack enters the rendering stage. Because of the three stack generation, 2D texture slicing requires three times the memory required by 3D texture slicing. Interpolation in 2D texture slicing is bilinear which is faster than the trilinear interpolation of 3D texture slicing. Due to this, 2D texture slicing is supported by more graphics hardware than 3D texture slicing.

D.3 Programming Libraries

D.3.1 OpenGL

Open Graphics Library (OpenGL) is a cross-platform and cross-language library used to write applications displaying 2D and 3D objects [126, 127]. OpenGL is managed by Khronos¹, a not-for-profit group. OpenGL is widely applicable in various disciplines including virtual reality, scientific visualization and gaming.

OpenGL is a low-level application programming interface (API). Basic operations in OpenGL accept primitives such as points, lines and polygons, converting them into pixels via a graphics pipeline. Such operations were traditionally performed using fixed functions. However, OpenGL 2.0 released in 2004, added support for shaders

¹<http://www.khronos.org/>

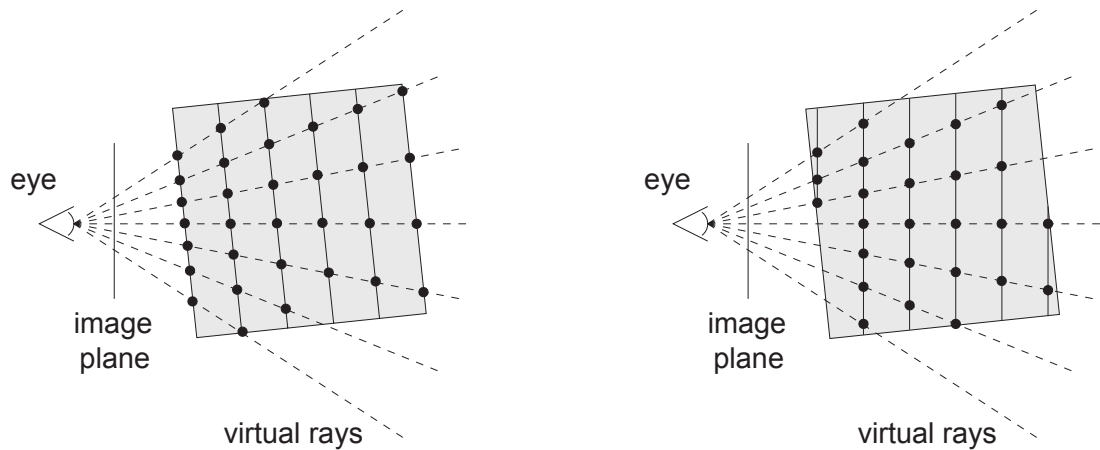


Figure D.3: Slices of 2D texture slicing (left) are object aligned while slices of 3D texture slicing (right) are view-aligned (*illustration by D. Weiskopf [142]*).

which allows full programming stages to replace the fixed functions using OpenGL Shading Languages (GLSL). Shaders are further explained in the next section.

D.3.2 Shading Languages

A shading language is a relatively recent concept that allows attributes of individual vertices and pixels to be described. Attributes of a vertex include a position, texture and colour. Attributes for a pixel include a colour, z depth and alpha value.

A single compiled unit produced by a shading language is called a *shader*. A shader replaces the traditional *fixed function pipeline (FFP)* which hard-codes the processing of effects such as lighting and texture mapping. Thus, a shader allows a programmable lighting and texture mapping. Shading was implemented in assembly language till *OpenGL Shading Language (GLSL)* [129] was introduced in 2002 and formally became part of OpenGL 2.0 in 2004. Another common shading language is *High Level Shading Language (HLSL)* for Microsoft's Direct3D, part of DirectX API [143]. There are three types of shaders as follows.

1. **Vertex shaders** run once per vertex transferring its 3D position to a 2D coordinate that appears on a screen. The output of a vertex shader is either fed into the *rasterizer stage* or *geometry shader*.
2. **Geometry shaders** remove or add vertices from a mesh. The output of a geometry shader is fed into the *rasterizer stage* from which an output is fed into the *pixel shader*.

3. ***Pixel shaders*** calculate colours of individual pixels. Pixel shaders are also known as *fragment shaders* as the term “fragment” is used for “pixel” in OpenGL.

Appendix E

SEG-Y Reader for Voreen

Voreen is an open source and cross-platform volume rendering engine built and maintained by the Visualization and Computer Graphics Group at the Department of Computer Science of the University of Münster, Germany [144]. It allows users to interactively visualize volumetric datasets through GPU-based renderer techniques, such as ray-casting. Similarly to *AVS*, *Voreen* allows users to build a network of processors.

GPU-based ray casting is the primary volume rendering technique in *Voreen*. With the introduction of shading languages, it became possible to perform the ray-casting technique directly on a supporting graphics card. Each volume in *Voreen* is handled as a single object. An object of volume is identified by a dimension, spacing (for scaling) and the bits stored forming the dataset.

Voreen originally supports several formats as an input source such as RAW. However, since it is open-sourced, developers can extend it. During the timeframe of this PhD, the author developed a SEG-Y reader into the core library of *Voreen*; this is to read and visualize a seismic dataset from the standard seismic file format (SEG-Y). Figure E.1 shows screenshots of two rendered 3D seismic volumes. This extension was released as part of *Voreen*'s version 2.6.1¹.

¹The source code of the developed SEG-Y volume reader header can be found at this link: http://www.voreen.org/doc/2.6.1/segyvolumereader_8h_source.html.

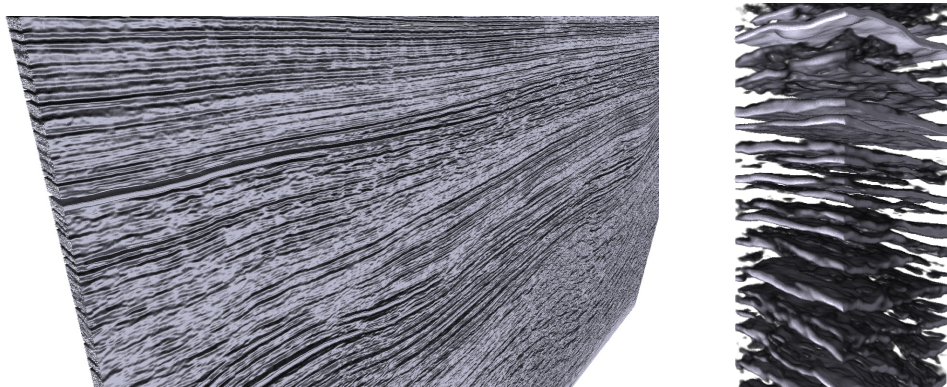


Figure E.1: These are screenshots of the output of the SEG-Y reader, which was developed as an extension to *Voreen*. The rendered outputs illustrate some geological layers of two different datasets.