



Provided by the author(s) and University of Galway in accordance with publisher policies. Please cite the published version when available.

Title	Supporting Email-based Collaborative Work across a Social Semantic Space
Author(s)	Scerri, Simon
Publication Date	2010-10-31
Item record	http://hdl.handle.net/10379/1981

Downloaded 2024-04-24T10:59:41Z

Some rights reserved. For more information, please see the item record link above.





Supporting Email-based Collaborative Work
across a Social Semantic Space

Simon Scerri

Submitted in fulfillment of the requirements for the degree of
Doctor of Philosophy

SUPERVISOR:
Dr. Siegfried Handschuh
National University of Ireland Galway

INTERNAL EXAMINER:
Prof. Dr. Stefan Decker
National University of Ireland Galway

EXTERNAL EXAMINER:
Prof. Dr. Shahram Dustdar
Vienna University of Technology

Digital Enterprise Research Institute,
National University of Ireland Galway.
October 2010

Abstract

The *Social Semantic Desktop* adopts Semantic Web technology on the desktop to provide a universal platform for personal – and distributed – information management, social networking and community creation. The social semantic desktop extends the personal desktop across two dimensions: socially and semantically. The semantic aspect relies on the lifting of desktop information onto a semantic representation. A set of high-level ontologies, including a novel representational language, was engineered to enable such representation. The social aspect relies on semantically-enabled, inter-desktop, communication and information exchange media. As a crucial business communication tool, electronic e-mail was immediately identified as a medium that can undergo such a transformation. Moreover, e-mail is more than just a means of communication, often serving as a virtual extension to the user’s working environment, wherein they collaborate; generating and sharing new information in the process. However, although e-mail is a valuable source of desktop information, it also suffers from a number of information management problems. The overall effect of these problems is termed *e-mail overload*, and results in the users becoming overwhelmed by the amount and diversity of incoming information, eventually losing track of important tasks and commitments. Furthermore, the ensuing information

(mis)management issue at hand is of an interpersonal nature, with one's failures affecting the productivity of an entire collaborative network. Therefore, before enabling e-mail technology with semantics, this problem needed to be adequately addressed.

In comparison to a myriad of other approaches, this thesis targets what is considered to be the *source* of e-mail overload: the lack of support for the underlying *ad-hoc e-mail workflows*. An attempt is made to structure the underlying communication processes, in view of the possibility of enabling machines to support them. The conceptual building blocks for e-mail workflows – *action items* – derive from computationally-treatable aspects of *speech act theory*. Models for representing different kinds of e-mail action items, and ensuing workflows, are introduced. Knowledge in these models is exposed to machines via an ontology, itself deriving from the higher-level social semantic desktop ontologies. The latter are used to represent additional shared *workflow artefacts*, such as messages executing the workflows, contacts (participants) involved, resulting tasks and events, attached files, etc. Information extraction techniques are employed for the semi-automatic elicitation of action items, whereas additional information about the ensuing workflows is elicited through user interaction, either indirectly or at a minor cognitive cost. Powered by these technologies, an intelligent user interface provides on-the-fly e-mail workflow support, without exposing the user to the complexity of the underlying models.

The proof of concept for the proposed solution is provided via Semanta – a semantic communication support system that enables semantic e-mail, to assist the user with the better management of e-mail-based, cross-desktop, collaborative work. Thus, Semanta

also fosters data integration across a network of social semantic desktops. With semantic technology seamlessly-integrated within the existing technical landscape, Semanta does not require users to familiarise with new e-mail clients or transport technologies. Whereas workflow management and visualisation is provided *alongside* that conventionally provided for e-mail messages, workflow processing is performed *in parallel* to habitual e-mail use, such as reading and replying to messages. An evaluation of the system indicates that, although there is further room for improvement, Semanta already provides additional support to the collaborative e-mail user. The results imply that, coupled with semantically-enabled desktops, there is a huge potential for semantic communication support systems to improve cross-desktop collaboration taking place over internet-based, written forms of communication, such as e-mail.

Acknowledgments

There are many individuals who have, in their many ways, supported me and the work covered by this thesis, and to whom I would like to express my sincere gratitude. I would first of all like to thank Dr. David O’Sullivan for giving me the opportunity to pursue my postgraduate studies in DERI, under the umbrella of the National University of Ireland, Galway. My research directions during the past five years have been influenced by three key DERI figures: my supervisor Dr. Siegfried Handschuh, DERI’s director Prof. Dr. Stefan Decker and vice director Prof. Dr. Manfred Hauswirth. Whereas my interest in email collaboration support was spurred by Prof. Decker, my workflow management approach was adopted following discussions with Prof. Hauswirth. Apart from trusting me with the knowledge representation tasks in the Nepomuk project, Dr. Handschuh has constantly made sure that my evolving research directions remained relevant and innovative. Other fellow DERI researchers to whom I would like to express my personal thanks are: Brian Davis, for sharing his expertise in text analytics; Dr. Armin Haller for his valued opinion on my workflow model; Alison Muscat for sharing her mathematical background to help me provide the model’s formal definitions. Of the developed email client plugins, I owe the thunderbird counterpart to two dedicated interns: Ioana Giurgiu

and especially Gerhard Gossen. Short of mentioning all individuals who have in one way or another helped, e.g. carrying out evaluations, reviewing personal paper submissions and thesis chapters, etc., I will stop at just two more: Dr. Knud Moeller and Dr. Tudor Groza, for occasionally sharing their views on my research, and for providing technical help with some implementation and authoring tasks.

Apart from the above direct support, I am also grateful to those who, since my childhood, have given me a reason to explore my potential, as well as provided me with the most welcome personal challenges. In particular I would like to thank all friends and relatives who have provided me with their continuous support, especially the ones in Malta, who were always there despite the distance. Most of all, I am forever grateful to my parents, Anna and John, who together with my elder brothers, have allowed me the privilege to be raised in an environment which instilled in me a desire to grow and learn, a need to aim high, a sense of personal discipline and a belief in a successful future.

Declaration

I declare that the work covered by this thesis is composed by myself, and that it has not been submitted for any other degree or professional qualification except as specified.

The research contributions reported by this thesis was supported(in part) by the Lion project supported by Science Foundation Ireland under Grant No. SFI/02/CE1/I131 and (in part) by the European project NEPOMUK No FP6-027705.

Table of Contents

Table of Contents	x
List of Figures	xvi
List of Tables	xx
I Prelude	3
1 Introduction	5
1.1 Motivation, Problem Statement and Requirements	7
1.2 Research Questions	9
1.3 Thesis Overview	12
1.3.1 Document Structure	12
1.3.2 Research Map	14
II Background	19
2 The Semantic Web	21
2.1 The Evolution of the Web	21
2.1.1 Origins	22
2.1.2 World-Wide Web	24
2.1.3 Web 2.0	26

2.1.4	Semantic Web	27
2.2	Data Representation on the Semantic Web	29
2.2.1	RDF/S	31
2.2.2	Ontologies	34
2.2.3	Syntax and Semantics	39
2.2.4	Named Graphs	45
3	Towards a Social Semantic Desktop	51
3.1	Motivation	51
3.1.1	Semantic Desktop	52
3.1.2	Social Semantic Spaces	55
3.1.3	A Source and End-point of the Semantic Web	56
3.2	Knowledge Representation Requirements	59
3.2.1	Enabling Semantic Interoperability	59
3.2.2	Effective handling of multiple knowledge modules	60
3.2.3	Supporting Multiple Views and Semantics	61
3.2.4	Unified, epistemologically-adequate modelling primitives	62
4	Email and Personal Information Management: Problems and Approaches	65
4.1	Email Overload	66
4.2	Search-based Solutions	70
4.3	Visualising Email Communication	71
4.3.1	Visualising Email Conversations	72
4.3.2	Visualising email collaboration	72
4.4	Email Message Classification	73
4.4.1	Rule-based Classification Techniques	74
4.4.2	Information Retrieval-based Classification Techniques	75
4.4.3	Machine Learning-based Classification Techniques	76
4.4.4	Miscellaneous Classification Techniques and Approaches	78

4.5	Structuring Email Processes	79
4.5.1	Task-centric views of Email	80
4.5.2	Email Workflow and Business Process Modeling	82
4.5.3	Semantic Email	85
4.6	Advancing the State-of-the-art	87
5	Pragmatics and Electronic Communication	89
5.1	Origins of the Pragmatic Web	89
5.2	Supporting Electronic Communication	93
5.2.1	The Coordinator	94
5.2.2	Applications to Digital Communication Systems	96
5.2.3	Speech act-based Systems of Categorisation	98
5.2.4	Speech Acts as the building blocks of Digital Workflows	101
III	Core	105
6	Distributed Knowledge Representation on the Social Semantic Desktop	107
6.1	The Nepomuk Representational Language	108
6.1.1	Motivation	109
6.1.2	Named Graphs – Supporting Data Modularisation	111
6.1.3	Graph Views – Supporting Multiple Views and Semantics	113
6.1.4	Decoupling Syntax and Semantics in NRL	116
6.1.5	RDF/S re-use best practice for the Social Semantic Desktop	119
6.1.6	Enabling richer Semantics via Constraint Extensions	122
6.1.7	NRL in use: An example	123
6.2	Modelling the Building Blocks of the Desktop	128
6.2.1	NEPOMUK Ontologies Stack	128
6.2.2	The Nepomuk Annotation Ontology	131
6.2.2.1	Vocabulary Overview	132

6.2.2.2	Graph Metadata Vocabulary	134
6.2.2.3	Support for Web 2.0 Technology in NAO	134
6.2.3	Correlating with Related Work	136
6.3	Conclusions	138
7	A Conceptual Framework for Semantic E-mail	141
7.1	Modeling Email Speech Acts	142
7.1.1	The Speech Act Model	143
7.1.2	Evaluation and Improvements	150
7.1.2.1	Methodology	150
7.1.2.2	Results and Observations	153
7.1.2.3	Improvements	156
7.1.3	Defining Semantic E-mail in terms of Speech Acts	157
7.2	Modeling E-mail Workflows	165
7.2.1	Semantic E-mail as a Workflow Execution Environment	166
7.2.1.1	Scenario I: Meeting Scheduling	168
7.2.1.2	Scenario II: Task Assignment	170
7.2.1.3	Scenario III: File Acquisition	170
7.2.1.4	Scenario IV: Task Delegation	172
7.2.2	Defining E-mail Workflows in terms of Speech Act Processes	173
7.2.3	A study into Speech Act Sequentiality	176
7.2.4	A Behavioural model for E-mail Workflows	179
7.3	The Semantic E-mail Ontology	188
7.3.1	Speech Act Representation	189
7.3.2	Semantic E-mail Representation	190
7.3.3	E-mail Workflow Representation	190
7.4	Conclusions	191

8	Towards Automatic Speech Act Classification	193
8.1	A model for Rule-Based Information Extraction	193
8.2	Implementation	199
8.3	Evaluation	201
8.4	Conclusions	202
9	Semanta	205
9.1	E-mail Overload Revisited	206
9.2	Implementation	209
9.2.1	A multi-tier Architecture	210
9.2.2	Hiding complexity beneath an Intelligent User Interface	212
9.3	An Intelligent Workflow Management Application	216
9.3.1	Eliciting Workflow Knowledge	216
9.3.2	Supporting Workflow Execution	218
9.3.3	Supporting Email-generated Activity Management	219
9.3.4	Integrating Workflow Artefacts	220
9.3.5	Intelligent Secondary Features	221
9.3.6	Visualising E-mail Workflows for Improved Management	223
9.4	A Communication Medium for the Social Semantic Desktop	227
9.4.1	Transporting Semantics across Desktops	228
9.4.2	Representing E-mail Workflows	228
9.4.3	Supporting Desktop Knowledge Integration and Retrieval	230
9.5	Evaluation	233
9.5.1	Formative Evaluation	233
9.5.1.1	Method	234
9.5.1.2	Results & Discussion	235
9.5.1.3	Design Improvements	237
9.5.2	Summative Evaluation	237
9.5.2.1	Method	238

9.5.2.2	Results & Discussion	240
9.6	Conclusions	243
IV	Conclusion	246
10	Summary	247
11	Future Work	253
11.1	The Social Semantic Desktop Ontologies	253
11.2	Semantic E-mail Conceptual Framework	254
11.3	Speech Act Classification	255
11.4	Semanta	257
11.5	Web-Science: Inducing take-up of Semantic Technology	259
	Bibliography	261
A	Summative Evaluation Questionnaire	289
A.1	System Usability (USE Questionnaire)	289
A.1.1	Usefulness	289
A.1.2	Ease of Use	290
A.1.3	Ease of Learning	290
A.1.4	Satisfaction	291
A.2	Task Comparison	291
A.2.1	Writing New Email	291
A.2.2	Reading & Replying to Incoming Email	291
A.2.3	Email Action Item Tracking	292
A.2.4	Task/Calendar Integration (Skip if features not used)	292
A.2.5	General	292
A.2.6	Comments & Feedback	292

List of Figures

1.1	Research Map: The major spanned research areas and methodologies . . .	15
2.1	The Semantic Web Stack	31
2.2	An RDF Graph with three Statements	33
2.3	The ABox and TBox for Semantic Web data	37
2.4	Using Named Graphs as an alternative to statement reification	48
3.1	Sharing tags with specific meanings across different users and platforms .	57
3.2	From a syntax-based personal desktop to a Web-integrated SSD	58
5.1	From shared grammar on the Syntactic Web, towards shared meaning and socially-defined context on the Pragmatic Web	92
6.1	NRL Named Graph class hierarchy	114
6.2	Graph Views in NRL	116
6.3	Overview of NRL - Abstract Syntax, Concepts and Semantics	117
6.4	NRL Export for a Protégé RDF/S Project	119
6.5	NRL dataflow for Anna's task	124
6.6	The NEPOMUK Ontologies Stack	131
6.7	Basic Annotation Properties	132
6.8	More specific annotation properties	133
6.9	NAO Graph Metadata Vocabulary	135
6.10	Representing Tagging practices via NAO	135

7.1	Speech Act Actions, organised by Discourse Roles	144
7.2	Speech Act Objects and Subjects	147
7.3	Instances of the Speech Act Model	148
7.4	Confusion matrix for annotation disagreement in the speech act model . .	155
7.5	A Semantic E-mail from Martin to Claudia and Dirk	161
7.6	An email thread consisting of 7 emails carrying 11 speech acts	166
7.7	Visualisation of the first workflow in the example as per definitions	168
7.8	Visualisation of the second workflow in the example as per definitions . .	170
7.9	Visualisation of the third workflow in the example as per definitions . . .	171
7.10	Visualisation of the fourth workflow in the example as per definitions . . .	172
7.11	Bayesian network for the observed workflow speech act transitions	177
7.12	The Semantic E-mail Workflow (SEW) Model	182
7.13	<i>Other</i> and <i>Ignore</i> choices in simplified XOR	185
7.14	The dependant paths of a Propose speech act	186
7.15	Representing Activity Delegations and Forwarding	187
7.16	Classes and Properties in the Semantic E-mail Ontology	189
8.1	Graphical representation of the simplified classification model, classifying clauses from textual email content into five broad speech act categories . .	195
8.2	The Speech Act Classification Pipeline	199
8.3	Evaluation results for the rule-based classification task	202
9.1	Martin's Conceptual vs Desktop Workflow View	207
9.2	Semantic Email Implementation across the different levels	210
9.3	Constructing an Action Item annotation with the Annotation Wizard . .	214
9.4	Same GUI look and feel, same underlying functionality, different email clients	215
9.5	Semi-automatic action item annotation	217
9.6	Supporting individual e-mail action items and associated workflows	219

9.7	Detecting Generated Events and Tasks	220
9.8	Integrating E-mail Workflow Artefacts (Messages, Tasks/Events)	222
9.9	File Attachment Reminders	223
9.10	Screencast showing Semantas workflow-based email visualisation. The user navigates from an initial action item (top right) to the ensuing workflow (middle right) and finally a specific email (bottom right).	225
9.11	Semanta's Workflow-based E-mail Visualisation in Outlook	227
9.12	Metadata generated by Semanta as viewed on the SSD	232
9.13	Evaluation Process Timeline	234
9.14	Pending action items retrieved (shaded) against total (outlined)	236
9.15	USE Questionnaire results	241
9.16	Results of summative question	242

List of Tables

1.1	Personal publications covering the research contributions of this thesis . . .	18
5.1	Comparison of Surveyed Speech Act Categorisations	101
6.1	NRL Constraint Vocabulary	123
7.1	E-mail text annotated based on the speech act model	152
7.2	Workflow Patterns in use in the SEW model	180
8.1	Examples of text as classified by rules based on the classification model . .	197
9.1	Average Usage Statistics	240
9.2	Hypothesis Testing for Semanta's added support over Thunderbird	242

“Any intelligent fool can make things bigger and more complex... It takes a touch of genius - and a lot of courage to move in the opposite direction.”

– **Albert Einstein**

Part I

Prelude

1 Introduction

The *Social Semantic Desktop* initiative [Decker and Frank, 2004] adopts Semantic Web technology on the shared personal desktop, to provide for a universal platform for both personal and distributed information management, social networking and community creation. The social semantic desktop is required to support the integration and retrieval of heterogeneous data across application and desktop boundaries, and provide a means for semantic interoperability.

The semantic dimension of the social semantic desktop relies on the lifting of all the information on the user's personal desktop, onto a semantic representation, based on adequate knowledge representation schemas. Part of this thesis will discuss the knowledge modelling techniques employed to engineer a set of ontologies for this purpose.

The social dimension of the social semantic desktop relies heavily on inter-desktop communication. In a digital environment, there is more to communication than the word might initially suggest. *Electronic Mail* (e-mail, or email) [Resnick, 2001] for example, doubles as an extension to the collaborative user's working environment, serving as a virtual workplace within which they collaborate; generating and sharing new information in the process. E-mail therefore, is a source of valuable new information which should be tapped into, in order to enrich the knowledge representations on the personal desktop. Furthermore, this information is frequently intrinsically shared between the collaborators (e.g. shared events, tasks, documents etc.), and can thus be exploited to enhance the semantic interoperability between the respective social semantic desktops.

The use of email is however afflicted with widespread information management prob-

lems, which are collectively referred to as *email overload* [Whittaker and Sidner, 1996, Whittaker and Hirschberg, 2001]. This thesis presents an approach to counteract this problem via the structuring of the underlying email communication processes, in order to integrate distributed knowledge pertaining to specific workflows executing within email threads. To enforce such a structure, computationally-treatable aspects of *Speech Act Theory* [Austin, 1962, Searle, 1969] are considered as the building blocks of *Ad-Hoc Email Workflows*. For the purpose, a speech act model, and a workflow model that strings together sequences of speech acts into workflows, will be presented. Also covered by this thesis is a technique for the semi-automatic elicitation of speech acts, and the ensuing workflows.

The ontologies, models and techniques developed over the course of this dissertation have been integrated within a semantic communication support system – *Semanta*, that enables *Semantic Email*. *Semanta* has been implemented as an extension to existing email clients, reusing existing email transport technology. Coupled with a simple and intuitive user interface, this approach ensures that conventional email practices are retained, such that email users are not required to undergo any major changes in their email practices. In a nutshell, this thesis investigates whether the email model can be extended with semantics to successfully:

- function as a semantics-enabling communication channel for the social semantic desktop;
- support email users with better managing their email-generated information.

An evaluation of *Semanta* shows that semantic technology can indirectly reduce the negative effects of email overload, by providing automated support for the management of email workflows. The capabilities of semantic email as a means for transporting and integrating shared desktop information will also be demonstrated, via practical scenarios involving the exchange of semantic email across multiple users, and their social semantic desktops.

1.1 Motivation, Problem Statement and Requirements

Despite the rise of competing technologies, email remains a crucial business communication tool and an important source of enterprise information and knowledge. Email's successes are attributed to a very simple, yet effective, protocol, whose asynchronicity frees the participants from the constraints of time and space. However, not being designed for the current times, and uses, email is also afflicted with many renowned deficiencies. The majority of these shortcomings can be attributed to the many different ways in which people now use the technology, with most of them going beyond the intended design. The term *email overload* refers to the use of email for such functions, and describes how this unintended use results in email users being faced with a serious information overload problem. This problem is especially relevant in the context of the *Social Semantic Desktop* initiative (SSD), where a great deal of new information is continuously reaching a user's desktop via email. In fact, the vast amount of heterogeneous information reaching the desktop tends to outstrip the user's ability to correctly manage and exploit it. This situation is further worsened by the fact that the exchanged information is *interpersonal*, rather than personal, in nature. This effectively demands a group, rather than a personal, effort at proper email information management, as one's failure in staying in control of their email tasks can directly affect one or more collaborating users, and vice-versa.

Many research efforts reviewed in this thesis have targeted the email overload problem by enabling machines to support the users with better managing their email data. Some have taken a direct approach, e.g., through the development of technologies for automatic email classification, enhanced search and retrieval; whereas others have taken less direct approaches to solving the problem, e.g., by facilitating email visualisation. Most of these efforts however, offer only a somewhat superficial solution that does not target the *source* of the problem – which lies in email technology being utilised not only as a simple communication means, but also to effectively perform collaborative work. From this perspective, the email overload problem can be projected as a workflow management problem where, users become overwhelmed with the increasing amount (and complexity) of co-executing

workflows, resulting in a loss of control over their email-based collaborative work. The source of this problem lies partly in the lack of structure imposed by the email model, and partly in the fragmented way in which these workflows are ‘represented’ on the user’s conventional desktop. In effect this ‘representation’ amounts to nothing but a number of physically-unrelated, albeit workflow-related, resources such as messages, contacts, documents, events, tasks, etc. At one stage or another, all these different types of data abstractions participate in the execution of the workflow, and can thus be considered as *workflow artefacts*. Unfortunately for the user, these artefacts are stored separately in different desktop data silos such as email folders (messages executing the workflow), system folders (documents related to the workflows), contact lists (workflow participants), calendars, task managers, etc., with no links or associations being retained in between.

An objective of the SSD initiative is to provide means to semantically relate information items scattered around the desktop, which would be of special relevance in this case. However, before defining such relationships, the resources in question need to themselves be semantically lifted [Volz et al., 2002] onto a semantic desktop representation. To correctly model these resources and their relationships, this lifting needs to be performed *at source*. Therefore, a semantics-enabling email system is required to directly tap into email communication as carried out on the desktop, in order learn about the structure of the implicit workflows, and their artefacts, as they evolve. However, so as not to jeopardise the prospects of its take-up by the existing large email user-base, any such enhanced system needs to respect popular emailing conventions, so as to let users carry on with their email tasks unobtrusively and without any form of hindrance. Email users must neither be expected to sacrifice considerable amounts of their time to semantically annotate email, as this would be at odds with the motivation behind this thesis. Therefore, an intelligent user interface is expected to, at least partially, automate the elicitation of new workflows, and provide automated support thereafter. Furthermore, given the existing email transport technologies, a semantically-extended email must be backward compatible with existing email transfer protocols and mail user agents.

1.2 Research Questions

From the above introduced requirements, the following research questions are derived:

1. **Can a common knowledge representation, with the adequate level of expressivity, be established across multiple social semantic desktops?**

The SSD requires extensive representation schemes that cover anything (physical items as well as abstract concepts) that is created, processed, managed and shared on and across a network of desktops. This presents for a challenge, considering the heterogeneity of data (and data structures) within the existing plethora of desktop applications. More problematic is the SSD requirement which foresees the need to let knowledge modules be interpreted against multiple semantics, where necessary. The end-target is then to express both complex application-level annotations, as well as simple, end-user oriented annotations, within one coherent model.

2. **How to comprehensively target the Email Overload problem?** As already discussed, most efforts that attempt to alleviate this problem do not target the source of the problem – which in this thesis is considered to be the lack of support for the underlying email workflows. Although ad-hoc in nature, these workflows are conceptually well-formed. This thesis therefore, will investigate whether by structuring email workflows and providing for their automated support, the overload problem and the ensuing information management hardships can be reduced.

3. **How can one model and represent email workflows?** As a starting point, it is required to determine the nature of the building blocks for structuring email workflows. Once this conceptualisation is in place, the stringing together of these elements into a representation of an evolving workflows presents another challenge, given the ad-hoc nature of email workflows. In fact, analogously to natural conversations, email is spontaneous and the next conversational move is largely unpredictable. However, it also manifests repeated patterns of communication; therefore the email workflow modelling pursued needs to investigate whether, and to which

extent, email conversational moves can be predicted. Nevertheless, email's flexibility must at all times be considered as an intrinsic feature, as despite its obvious modelling disadvantages, it remains one of email's most favourable features.

4. **Can machines be enabled to work with workflow representations?** In order to eventually support email workflows, the conceptualisations provided by the envisaged modelling need to be fully and formally exposed to machines. Semantic Web representation schema offer a possibility in this respect, and the engineering of a dedicated (set of) ontology(ies) can sufficiently represent the necessary email workflow concepts and their relationships. Since email is just a medium for the execution of these workflows, email workflows need to be represented separately than the email messages per se. Instead, links between email thread structures, and the underlying email workflows, will need to be semantically established and retained. This will then allow machines to work directly with the underlying email workflows, while also being aware of the context of the email thread within which they execute.
5. **To what extent can new email workflows be automatically elicited?** To support with the execution of email workflows, an enhanced email system needs to first become aware of their initiation. This makes for a knowledge-acquisition bottleneck problem, especially since to ensure usability, the users cannot be burdened with this task. Therefore, techniques for the automatic, or semi-automatic recognition of initiating workflows need to be investigated.
6. **Can workflow support be provided without disrupting conventional email practices and user habits?** One of the earliest design criteria for the envisioned email communication support system is that the way users use email must change minimally, if at all. Furthermore, any superficial (i.e., at the interface level) changes must be unobtrusive, with any added tasks remaining optional (e.g. semi-automatic email annotation). In this way, while the user's emailing practices remain largely

unchanged, the system can provide additional workflow support. This is a challenging requirement, as it entails the integration of semantic technology within the existing technical landscape, which includes the use of popular Mail User Agents (MUAs). Here, an intelligent user interface can play a major role in mediating between the introduced semantic technology and the conventional email user, who must never be directly exposed to the former.

7. **Can semantic email serve as an adequate communication channel for the Social Semantic Desktop?** The social dimension of the SSD relies heavily on the data transportation channels in between networked desktops. In order for semantic data to also be shared between the SSDs, transportation channels such as email are required to transport metadata, alongside data. An enhanced email technology can thus offer the possibility to transport metadata between a network of semantic desktops, semantically integrating shared information items in the process. Additionally, email is not only responsible for transporting data, but it is also one of the major communication media in use between desktops. As already stated, email offers more than simple communication, serving as a virtual collaboration environment within which new, useful data is continuously being generated and shared. Thus, the right technology for a semantically-enhanced email would also be able to enrich the semantic knowledge within the involved SSDs with useful representations of items generated as artefacts of email workflows.
8. **Can the envisaged communication support system aid desktop users with the better management of their email-based collaborations?** Alternatively, one can instead pose the following question: If all the other research questions above prove to have satisfactory outcomes and technically-feasible solutions, can the semantic, workflow-oriented approach to reduce email overload achieve the intended results? In other words, can all the foreseen desktop and email workflow knowledge modelling, applicable semantic technologies, techniques for the (semi-) automatic

elicitation of workflows, and intelligent user interface designs, be combined to assist users with the management of email-based collaborations? An answer to this question can only be provided following a successful implementation, and evaluation, of the communication support system envisioned by this thesis.

1.3 Thesis Overview

To prepare the reader for the rest of the document, I provide an overview of its structure and contents below. In addition, a ‘research map’ highlights the main areas and methodologies investigated by this thesis, and includes references to the scientific publications covering this work, to which I have also contributed.

1.3.1 Document Structure

This book is structured as follows. The remaining chapters in Part I introduce the reader to the research problems targeted by this effort; providing an insight into these problems, the state-of-the-art dealing with them, and other relevant related work.

Chapter 2 then introduces the Semantic Web, starting with a historical account of how the existing World Wide Web came to be, and how it is foreseen to develop into a fully-fledged Semantic Web (section 2.1). Section 2.2 presents an overview of Semantic Web data representation formats and standards, and a discussion of specific paradigms and concepts which are of special relevance to this thesis, e.g. syntax, semantics, ontologies and named graphs.

In section 3.1, chapter 3 introduces the motivation for a social semantic desktop, as a vision for a personal desktop that is extended within two dimensions: semantically – to enable machine-processable representation and processing of desktop knowledge, and socially – to enable the sharing of this knowledge between a network of semantic desktops. The chapter explains how the current personal desktop can thus be transformed into a source and end-point of the Semantic Web, and, in consideration of this vision, proceeds to lay out the requirements in section 3.2.

Chapter 4 first delves into the information management problems associated with email (section 4.1). A number of approaches attempting to provide automated management support are then surveyed in sections 4.2 – 4.5 . Finally, in view of the results of this survey, a set of guidelines for advancing the state-of-the-art is outlined in section 4.6.

Chapter 5 discusses the prospects of the Pragmatic Web [de Moor et al., 2002, Te’eni, 2006] as a next stage in the evolution of the (Semantic) Web. The origins of this relatively new research area are discussed in section 5.1. In view of the context of this thesis, section 5.2 then demonstrates how concepts from this area have been applied to electronic communication with the intent of providing different forms of automated support.

The core contributions of this thesis are presented in Part III, starting from the knowledge modelling approaches pursued during the design stages of the SSD, followed by an attempt at structuring and representing semantic email and email workflows, and concluding with a demonstration and discussion of the implemented workflow-supportive semantic email client extensions.

Chapter 6 starts with a presentation (section 6.1) of the concepts and paradigms behind the modelling of a novel representational language – NRL [Sintek et al., 2007b], in light of the requirements presented in section 3.2. Section 6.2 presents some of the additional ontologies engineered for the platform, with a particular focus on the annotation ontology (NAO) [Scerri et al., 2007b].

Chapter 7 is the most significant chapter of this thesis. It starts by proposing how the content of an unstructured email message can be conceptualised into email speech acts (section 7.1). Arguments are then put forward to explain how related sequences of these speech acts can subsequently be considered as independent ad-hoc email workflows (section 7.2). Formal and informal definitions are provided for speech acts, speech act processes, semantic email, and email workflows; including conditions for their satisfiability. An ontology that exposes the knowledge in this modeling to machines, sMail [Scerri, 2008b], is finally introduced in section 7.3.

Chapter 8 describes the information extraction techniques employed to attempt the automatic elicitation of a first speech act in an email workflow. A rule-based technique is presented in section 8.1, followed by its implementation details (section 8.2) and evaluation (section 8.3).

The final chapter in the core part of the thesis (chapter 9) presents Semanta [Scerri et al., 2009a], the implemented semantic communication support system. Section 9.1 starts off by revisiting the email overload problem (section 4.1), in view of the workflow-oriented approach presented in the preceding chapters. Implementation details are then presented in section 9.2. Sections 9.3 and 9.4 serve as a proof of concept for the research questions posed by this thesis; via a few scenarios they demonstrate how Semanta doubles as both an intelligent workflow management application, as well as an adequate, semantics-enabling, communication medium for the SSD. The design, process and results of the extensive evaluation of Semanta are detailed in section 9.5.

Part IV wraps up this thesis, starting with a summary of the research contributions entailed (chapter 10), vis-a-vis the research questions laid down in section 1.2. Finally, Chapter 11 discusses a number of possible future directions for the continuation and extension of the models and technologies developed over the course of this thesis.

1.3.2 Research Map

To give an overview of the research areas and directions covered by the work in this thesis, a *Research Map* in the form of a Venn diagram is provided in Fig. 1.1 below. This map shows the overlapping topics and methodologies, with pinpointed references to the associated scientific publications to which I have contributed, mapped to Table 1.1. This helps to show these publications in the context of the overlapping areas and methodologies. In addition, the sets featured in Fig. 1.1 are briefly introduced below:

1. **Semantic Web** :- The advancements leading to this machine-interpretable extension of the World Wide Web are relayed in chapter 2. This includes the emergence of the Web itself, it's evolution to the Social Web (although Web 2.0 and associated

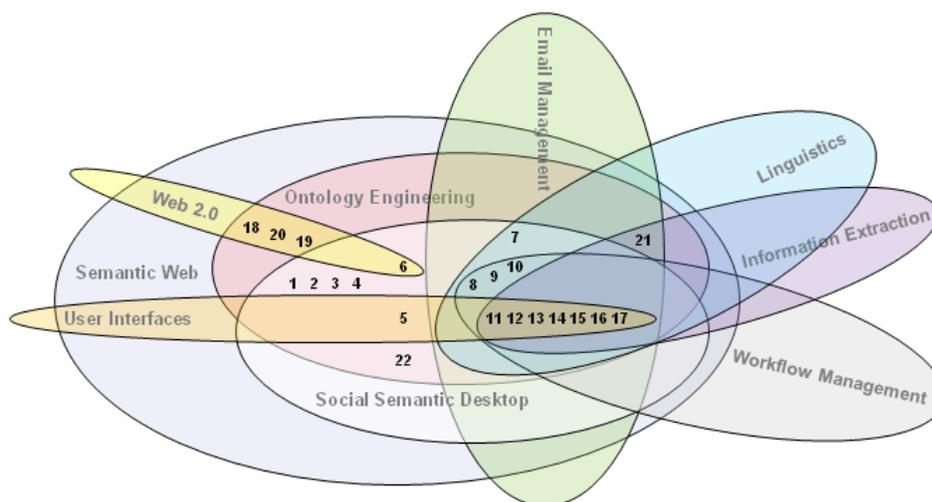


Figure 1.1: Research Map: The major spanned research areas and methodologies

paradigms are represented separately) and eventually to the architecture foreseen for the Semantic Web itself. Representation formats developed for the Semantic Web have been used to enhance the major novel technologies discussed in this thesis with semantics, i.e., semantic email and the SSD.

2. **Ontology Engineering** :- Given the need to expose knowledge arising from the conceptualisations and models presented in this thesis to machines, ontologies feature as an important part of the architecture for both the SSD as well as semantic email. Since I have contributed to the engineering of ontologies for both, this methodology has been explicitly included in the diagram, as a subset of the Semantic Web research area. An introduction to ontologies is provided in section 2.2.2, and a description of a number of relevant engineered ontologies ensues in sections 2.1.3, 6.1 (a representational language), 6.2 (a set of ontologies), and 7.3.
3. **Social Semantic Desktop** :- This area encompasses a number of efforts at augmenting the personal desktop with Semantic Web technology, in order to enable machine-processable representations of its contents. Its social dimension then en-

ables these representations to be shared across a network of semantic desktops. Requirements for the SSD are discussed in chapter 3, whereas chapter 6 is dedicated to the engineering of the required ontologies and the design of the semantic aspect of its architecture. One of the motivations for a semantic email lies in the need for a semantics-enabling communication medium that is able to transport metadata, alongside data, in between multiple social semantic desktops. Concepts from the SSD thus feature heavily in this thesis, culminating in a portrayal of the implemented semantic email client extensions as an adequate communication medium for the platform (section 9.4).

4. **Web 2.0** :- Parts of this thesis deal with technologies related to this area, often considered as an intermediary stage on the way to the realisation of a Semantic Web (section 2.1.3). In particular, concepts like tagging and folksonomies are discussed, and modelled, in engineered ontologies such as SCOT [Kim et al., 2008b] (also in section 2.1.3) and NAO [Scerri et al., 2007b] (section 6.2.2). The latter is particularly relevant as it introduces Web 2.0 concepts to the modelling primitives of the SSD. Future work also foresees the introduction of tags as part of the annotations provided for Semantic Email (section 11.1).
5. **Email Management** :- A second major motivation for a semantic email is to enable machines to support the long-suffering email users with email information management. The effect of email overload on the user's personal information management, and approaches that attempt to counteract the resulting problems, are discussed at length in Chapter 4. In the context of email, which is exchanged between multiple desktops, these problems extend beyond the user's personal sphere. The indirectly necessitates a group effort at *interpersonal* information management.
6. **Workflow Management** :- The approach at reducing the email management problems described in this thesis is centered around a workflow concept. In fact, the research contributions are driven by the hypothesis that, email-generated infor-

mation can be better managed once the underlying implicit workflows are elicited and represented in a machine-processable format, which would in turn enable automated support. Although, as explained in section 4.5.2, this idea is not entirely new, the approach pursued here is rather novel. Chapter 7, and in particular section 7.2, deals with the modeling of email workflows, grounded in the area of control-flow workflow patterns [Voorhoeve and van der Aalst, 1997]. An (ad-hoc) workflow model for email is presented in section 7.2.4. This workflow-oriented approach culminates in Section 9.3, where the implemented semantic email client extension(s) is portrayed as an intelligent workflow management application.

7. **Linguistics** :- As already hinted out, the building blocks for modelling email workflows are derived from speech act theory and the philosophy of language [Searle, 1969]. Following in the steps of earlier research efforts, this theory is reviewed in the context of digital, typed conversations, and from a workflow perspective. The workflow modelling in chapter 7 is based on a taxonomy of speech acts (section 7.1.1). This approach builds on earlier initiatives such as the Language-Action Perspective [Flores and Ludlow, 1980, Te'eni, 2006], which eventually evolved into the emergent area of the Pragmatic Web [de Moor et al., 2002]. Discussed at length in chapter 5, the Pragmatic Web is a candidate parallel progression to the Web (alongside the Semantic Web), that takes into account the purpose, alongside the semantics, of information.
8. **Information Extraction** :- Once the workflow models are in place, approaches at semi- or automatically eliciting email workflow knowledge are considered. A number of information extraction techniques for the recognition of an initial speech act in an email workflow are investigated. For the purpose, text mining/analytics techniques are used to recognise computationally-treatable aspects of speech act theory. A rule-based approach is pursued (Chapter 8), whereby a number of complex hand-coded rules execute in a pipeline that performs an automatic classification of speech

acts (based on the speech act model in section 7.1.1). Additional information about the evolving workflow is then elicited indirectly through the user’s interaction.

9. **User Interfaces** :- Chapter 9 introduces Semanta: semantic extensions to existing email clients, as a tool for computer-supported collaborative work. The success of Semanta relies on suitable user interface components which provide the user with the added functionality, without weighing them down with additional burdens. Thus, an insight into the design of a user interface which hides the semantics, while conveying the feeling of an intelligent communication support system, was required at the design stage. The implementation of an adequate user interface was then succeeded by a usability study that took into consideration the guidelines set out by research pioneers in the area such as [Gediga and Hamborg, 2001, Nielsen, 2000].

Table 1.1: Personal publications covering the research contributions of this thesis

Number	Publication	Number	Publication
1	[Sintek et al., 2009]	12	[Scerri et al., 2008b]
2	[Sintek et al., 2007a]	13	[Scerri et al., 2009a]
3	[Sintek et al., 2007c]	14	[Scerri et al., 2009b]
4	[Sintek et al., 2007b]	15	[Scerri et al., 2009c]
5	[Caires et al., 2007]	16	[Scerri et al., 2008a]
6	[Scerri et al., 2007b]	17	[Scerri et al., 2010b]
7	[Scerri et al., 2008c]	18	[Hak Lae Kim, 2009]
8	[Scerri, 2008b]	19	[Kim et al., 2008b]
9	[Scerri et al., 2007a]	20	[Kim et al., 2008a]
10	[Scerri, 2007]	21	[Scerri et al., 2010a]
11	[Scerri, 2008a]	22	[Reif et al., 2008]

Part II

Background

2 The Semantic Web

The World-Wide Web (WWW) is arguably the singlemost technological and sociological advancement of the late 20th century. In the space of a couple of years, the Web saw an exponential growth that has of yet not shown any signs of decline despite the passing of by of almost two decades.

In the haste of exploiting the myriad academic and, especially, industrial benefits of a global real-time information sharing platform, the many contributors all adopted hastily-drawn data publishing standards, widely ignoring the original vision and guidelines laid down for the realisation of the Web. Despite the initial success achieved by the Web, the person credited with its invention – Sir Tim Berners-Lee, has since embarked on a drive to bring back the Web into alignment with what he had originally envisioned. The Semantic Web, as it is now known, is considered to still be in its infancy, but an increasingly strong interest in the technology expressed by industrial and governmental quarters in recent years, after years of academic promotion, augurs well for its future as the futuristic Web.

In this Chapter I will provide a brief history of the Web and the envisioned gradual evolution into the Semantic Web. In addition, I will also explain why the Semantic Web is cognitively superior to the Web, by demonstrating the power of expressivity provided by its data representation standards.

2.1 The Evolution of the Web

The development of the Web as it stands today came to being in different stages and although largely uncoordinated, the process got underway in a seemingly natural series

of cognitive progressions and their subsequent implementations.

2.1.1 Origins

Its origins lie in a 1945 speculative article by Vannevar Bush [Bush, 1945] – “*As we may think*” wherein a theoretical proto-hypertext computer system is proposed. The so called *memex* (a blend of the terms memory and index) was described as a prototypical photo-electromechanical device capable of serving as an “intimate supplement to one’s memory”. An individual would be able to store all their books, records, and communications within and subsequently consult the mechanised memex with exceeding speed and flexibility. The memex could also double as a self-contained research library whereby an individual could follow associative trails created by themselves or by other researchers. The end-result was envisaged to give researchers access to a “huge, indexed repository of knowledge – any section of which can be called up with a few keystrokes” [Wardrip-Fruin, 2003]. This vision of the memex is widely considered as the first for a hypertext system, i.e. a system supporting data containing *hyperlinks*. However it did not foresee the creation of hyperlinks in the modern sense, where a hyperlink connects an entity within a document to a local or remote destination. Instead the memex associative trails would create chained (linear) sequences of related microfilm frames, rather than the contemporary practice of information indexing, which was not analogous to mental associations in the human brain. Moreover, Bush described the possibility of *sharing* these personally-created associative trails with others. This idea has only been brought to fruition on the WWW very recently, with the advent of Web 2.0 and the Social Web – whereby (groups of) individuals are able to collaboratively create, publish and follow these links (associative trails) collectively (e.g. Wikipedia).

The ideas relayed by Bush’s article inspired the actual inventors of hypertext, Douglas Engelbart and Ted Nelson. In 1962 Engelbart presented a research proposal: “*Augmenting Human Intellect: A Conceptual Framework*” [Engelbart, 1962], influenced by the memex system and targeted at “improving the intellectual effectiveness of the indi-

vidual human beings”. His vision for human-computer cooperation also referred to terms which are now central to the idea of the Semantic Web, like concepts and terminologies. In discussing co-operative team work he actually describes cognitive alignment processes which are now effectively known as ontology mapping and mediation. This proposal culminated in his oN-Line computer System (NLS), which however was not completed until 1968. Engelbarts demonstration of this system in the same year is now known as *The Mother of all Demos* due to the amount of, now commonplace, experimental technologies presented for the first time. Besides items like the mouse, and concepts like screen windows, Engelbart is credited with demonstrating the first hypertext interface ever. Meanwhile, it was Nelson who had coined the term *hypertext* in his 1965 book: “*A File Structure for the Complex, the Changing, and the Indeterminate*” [Nelson, 1965]. Nelson also founded the Xanadu project, considered the first hypertext project. It was based on an outline for a word processor that is capable of storing, and displaying the differences between multiple versions. This was extended to facilitate non-sequential writing, such that individuals could choose their own path through an electronic document, leading to so-called zippered lists that allow new compound documents to be formed from excerpts of multiple documents. Nelson extended the concept of hypertext to beyond text and dubbed these kind of media *hypermedia*. The term is still in use today and its modern definition reflects Nelsons ideas whereby different types of media such as text, images, etc. are compounded to form a non-linear medium of information.

The next stage in this evolution was the commissioning of the ARPANET network by the Advanced Research Projects Agency (United States Department of Defense) in 1969 to conduct research on networking. Regarded as the predecessor of the global Internet, this network adopted the concept of packet switching, rather than circuit switching, whereby data could be disassembled into datagrams and gathered into packets which in turn could be posted to different destinations and routed independently of other packets. ARPANET continued overseeing the progress towards the WWW in the following decade. In 1972, Ray Tomlinson extended his e-mail program over ARPANET, so that users could utilise

to exchange messages over the network. Electronic mail (e-mail or email) first appeared just one year earlier, in 1971, where Tomlinson created the first email program. However, the email system in ARPANET was the first that saw email exchanged between users on different hosts in a network, rather than users using the same machine. This separation between the user and their machine was what necessitated the '@' character in the email address, still in use today.

In 1973, ARPANET foresaw the definition of the File Transfer Protocol (FTP) specifications and its implementation to enable the transfer of files over the network. The final step in this progression towards the WWW was the specification of the Transmission Control Protocol (TCP) in 1974, followed by the Internet Protocol (IP) four years later in 1978. TCP first appeared in Cerf and Kahns "*A Protocol for Packet Network Interconnection*" [Cerf et al., 1974b]. TCP was designed to alleviate conceptual and practical issues like scalability with the existing protocols in use over ARPANET. The resulting "*Specification of Internet Transmission Control Program*" [Cerf et al., 1974a] is the first to carry the term Internet, as a shorthand for the term *internetworking*. TCP was extended with IP in 1978 to split the responsibilities of end-to-end host communication (TCP) and the routing of packets and device-to-device communication (IP). TCP and IP were always considered as parts of the same protocol, or a protocol suite, which thus became known as *TCP/IP*. Essentially, TCP/IP provided the mechanism to implement the internet. The setting up of the internet ushered the realisation of the WWW, as an information-sharing model built on top.

2.1.2 World-Wide Web

By the 1980s most of the foundational technology for the WWW was already in place. The Internet had been implemented, and the notion of hypertext had been well established. All that remained was for someone to pick up the pieces of the puzzle and introduce and realise the WWW. Sir Tim Berners-Lee is credited with this breakthrough, although his idea to marry hypertext to the Internet was not immediately considered

as such. Berners-Lee recounts his futile approaches to members of both the hypertext and Internet technical communities, suggesting the combination of the two technologies [Berners-Lee and Fischetti, 1999] to target the problem of information sharing and presentation. His 1989 proposal [Berners-Lee, 1989], in which he talks of “a large hypertext database with typed links”, also generated little interest. The proposal was for a more elaborate information management system than ENQUIRE, a database and software project he had built in 1980 at the CERN institute. Although essentially a personal database of people and software models, each new page in the system needed to be linked to another existing page – thus utilising hypertext.

Eventually, given the lack of enthusiasm in the respective communities, Berners-Lee decided to take matters into his own hands, with some help from Robert Cailliau. Their work culminated in an improved proposal: “*WorldWideWeb: Proposal for a HyperText Project*” in November, 1990 [Berners-Lee and Cailliau, 1990]. This proposal envisaged the WWW as a web of “hypertext documents” viewable through “browsers” via a client-server based architecture. By the end of that year Berners-Lee had produced the remaining required tools necessary for a fully-functioning Web, namely the Universal Resource Locator (URL), initially known as the Uniform Document Identifier; the HypertText Markup Language (HTML) as well as the Hypertext Transfer Protocol (HTTP) 0.9. Berners-Lee had also implemented the first server software (CERN httpd) and the first Web browser and editor (also called WorldWideWeb). The NeXT computer Berners-Lee use to write this Web browser also doubled as the worlds first Web server.

The Web became a publicly available service on the Internet on the 6th of August, 1991. On this day, Berners-Lee posted a summary of the WWW project to the *alt.hypertext* newsgroup claiming that the WorldWideWeb project was setup to enable fellow physicists to “share data, news, and documentation” and that as a project it “aims to allow all links to be made to any information anywhere”. In his post, Berners-Lee also expresses his interest in spreading the web to other areas, and having gateway servers for other data. Paul Kunz, from the Stanford Linear Accelerator Center (SLAC) in the United States,

did just that – after being introduced to the Web during a visit to CERN. After adapting the NeXT software to SLACs needs, it was utilised to display SLACs online document catalog, in the process becoming the first Web server outside of Europe. Although the basic features that enable the WWW to function were implemented, not all of Berners-Lees ideas were brought to fruition. Two particular aspects were largely ignored until the relatively recent history of the Webs evolution, namely:

- the flexibility and expressivity in its data representation, where resources as opposed to just *information resources* (i.e. documents, but also people, concepts, etc.) could be related in specific ways (i.e. typed links rather than mere links with no semantics);
- the interactive collaboration aspect, where users could actively contribute to its content, thus increasing the rate of information sharing and enabling the funnelling of mass knowledge to provide answers to users queries.

In the next two subsections I will describe how these aspects have been resurrected in recent years, and discuss their potential or already-established degree of success in the present and especially the future Web.

2.1.3 Web 2.0

In his 1990 proposal for the Web [Berners-Lee and Cailliau, 1990], Berners-Lee also conceived of “the creation of new links and new material by readers”, such that the information’s “authorship becomes universal”. He also wrote about the possibility to provide “automatic notification of a reader when new material of interest to him/her has become available”. The reader can easily create parallels between these proposed characteristics and the concepts and technologies which fall under what we now refer to as Web 2.0, particularly to the 3rd and 6th categories in the *SLATE* Web 2.0 features and techniques – Search, Links, Authoring, Tags, Extensions and Signals [McAfee, 2006]. The term Web 2.0 was coined by Darcy DiNucci in her 1999 article “*Fragmented Future*” [DiNucci, 1999].

However her use of the term was within a fairly different context, which was more concerned with a Web design that respects the fragmenting of the Web due to an increasing variety of hardware tapping into it. However, the term resurfaced in 2003 in an article by Kingsley Idehen [Idehen, 2003], followed by a string of a number of other articles in 2003, this time in a context associated with the current meaning. The term started its way to mainstream popularity with the hosting of the first Web 2.0 conference, in whose opening John Batelle and Tim OReilly discussed the Web, or rather the individual desktops, as a *platform* on which software applications can be built and utilised. This migration would enable the harnessing of the *users content generation* to create value. The popularity of the term reached its peak in 2006, when the TIME magazines person of the year was announced to be “You” [Grossman, 2006], acknowledging the power of the mass users which participate in the Web 2.0 phenomenon through their interactive content creation and sharing over numerous Web 2.0 media such as blogs, wikis, and social fora.

A major concern that arose with Web 2.0 technologies is trust. The extension of authoring of data on the Web to practically anyone relies on a radical trust in the users to collaboratively and asynchronously create content – thus moving away from an architecture that is based on expertise to one based on mass knowledge. However, the amount of participating users means that the generated content’s reliability remains relatively high¹. Eric S. Raymond sums up this effect very well in what he calls the Linus’ Law – expressed in a collaborative software development setting [Raymond, 1999] but applicable also to collaboratively-generated content: “given enough eyeballs, all bugs are shallow”.

2.1.4 Semantic Web

Berners-Lee’s 1989 proposal [Berners-Lee, 1989] also suggested a more expressive representation scheme for the Web than the one which remains widespread today. In fact, Berners-Lee’s vision for nodes (i.e. objects) and arcs (i.e. relations) on the Web was reduced to a very limited form, whereby nodes can only consist of documents and arcs

¹This of course excludes highly-sensitive online fora and social services such as E-government

can only consist of simple, semantically-equal links. Originally, it was proposed that both nodes and arcs on the Web could be arbitrarily typed to represent anything. This meant that nodes would consist of not just documents but also represent people, concepts, etc. Arcs would similarly be able to provide specific relationships in between nodes, including relations such as dependencies, subsumption, referral, etc. Thus, from the start the Web was envisioned as a web of typed resources, rather than a web of documents; related to one another with a wider variety of typed relations having varying semantics.

In 1999, ten years after the WWW's conception, Berners-Lee elaborated on his original vision of the Web [Berners-Lee and Fischetti, 1999]. He termed this extended web the *Semantic Web*, where computers “become capable of analysing all the data on the Web – the content, links, and transactions between people and computers”. The Semantic Web paradigm is based on the extended expressivity originally proposed by himself for the Web, with a renewed emphasis on the machine-processability of its data. In fact, Berners-Lee claimed that the Semantic Web will be the means by which intelligent agents can “finally materialise”. He also described it as a component of Web 3.0 [Shannon, 2006], suggesting that it becomes the next natural step in the evolution of the WWW. In contrast to data on Web 2.0, knowledge generated on the Semantic Web needs to be fully processed by machines, and not just retrieved by humans. The main challenge facing the realisation of the Semantic Web is that, as opposed to Web 2.0 technology, the underlying vocabularies and ontologies driving it are generally designed by experts and not by regular Web users. Initially, most of the research in the Semantic Web community focused on the theoretical foundations of what would allow automatic agents to realise it as envisioned in [Berners-Lee et al., 2001]. This included ontologies, data modelling, logic and reasoning.

Eventually, the generation of a useful amount of Semantic Web data itself became more challenging than its modelling and processing. *Linked Data* refers to the methods used to expose, share and connect such data on the Web. To make this data as useful as possible, Berners-Lee defined [Berners-Lee, 2006] four principles for Linked Data:

1. All things should be identifiable by URIs.
2. The protocol used for URIs should be HTTP to ensure they are *dereferencable* (i.e. agents can trace and look them up).
3. Metadata should be in place for each dereferenced URI.
4. Data about a URI should contain links to other relevant URIs to prevent isolated islands of data.

The *Linked Open Data* community project [Bizer et al., 2007] picked up on these principles in an effort to oversee the generation of a sufficient amount of relevant open datasets on the Semantic Web, interconnected by means of links between nodes within the different datasets. For the Semantic Web to become mainstream, semantic web applications need to also be in place to process this data for the users benefit.

So far this chapter has focused on the differences between the Web and the Semantic Web and the resulting requirements, from a conceptual point of view. In the next section, the representation challenges faced by the extended expressivity required for data on the Semantic Web and the resulting envisioned architecture, terminologies and paradigms, as well as the available standard data representation formats, will be discussed.

2.2 Data Representation on the Semantic Web

Initially, HTML provided a sufficient means of representation for data on the WWW. The markup language allows for the creation of documents consisting of a combination of text and other objects, such as images and forms. Although HTML enables the provision of basic unidirectional links between documents and of shallow metadata (e.g. keywords, author), it falls short of the expressive representation required for a machine-processable organisation of knowledge, such as the one required to realise the Semantic Web. At best, HTML provides a means by which machines can categorise the content of web documents.

A number of attempts aimed at the extension of HTML to enable the machine-processable semantic markup. Microformats² re-use existing (X)HTML tags to convey metadata, such that information intended for human users can also be processed by software. However, a major breakthrough in the development of the Semantic Web took place with the publication of the *Resource Description Frameworks*³(RDF) data model and XML syntax specifications in 1999. RDF and the associated schema⁴ (RDFS) are a major component of the W3C's Semantic Web activity. RDF/S, as they are collectively referred to, have always featured in the *Semantic Web Stack* (Fig. 2.1). Originally referred to as the Semantic Web Layer Cake, the stack illustrates the architecture of the Semantic Web as envisioned by Berners-Lee himself. The stack consists of a bottom-up succession of increasingly more powerful standardised languages and technologies, where each successive layer exploits the capabilities of the preceding ones. However, each layer remains independent of the lower ones and although RDF, for example, is often serialised using XML syntax, other serialisations can also be used (more about this later). Below the syntax layer, URIs and UNICODE are established hypertext technologies. The two layers at the basis of the architecture thus demonstrate that the Semantic Web is an extension of the hypertext Web and not its replacement. RDF/S is depicted as the standard framework for data interchange over the Semantic Web, with RDFS providing the basic vocabulary required to create taxonomies. The Web Ontology Language (OWL) is shown as a standard language that extends the capabilities of RDF with reasoning power, by providing additional constraints based on description logic. Querying over RDF data (including RDFS and OWL) is provided by query languages for the Semantic Web, such as the now standardised SPARQL [Prud'hommeaux and Seaborne, 2008]. The remaining blocks and layers have no associated standardised technologies yet, although RIF and/or SWRL are shown as the candidates for extending the capabilities of OWL by providing

²<http://microformats.org/>

³<http://www.w3.org/RDF/>

⁴<http://www.w3.org/TR/rdf-schema/>

support for rules. In the latest versions of the Stack, Berners-Lee has explicitly incorporated Semantic Web user interfaces (UI) and applications to highlight the importance of a layer which will essentially allow humans to exploit the Semantic Web.

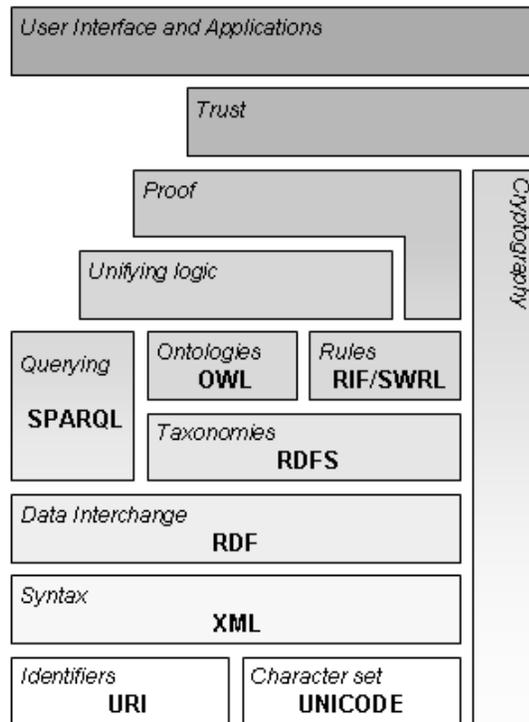


Figure 2.1: The Semantic Web Stack

In the following subsections I will discuss Semantic Web languages and technologies which are most relevant to the work done in this thesis. In particular I will discuss the capabilities and limitations of RDF/S and OWL, an introduction to ontologies, the role of syntax and semantics in the representation languages, and more sophisticated means of Semantic Web data modularisation and identification.

2.2.1 RDF/S

Although originally designed as a metadata model, RDF now represents a mechanism for the modelling of information that is implemented in web resources, using a variety

of syntax formats. The original RDF specifications were based on extending the existing URI and XML technologies, re-using the former for resource naming and the latter for syntax. An improved, successive set of specifications⁵ whereby RDF is not restricted to the XML syntax was published in 2004.

As earlier discussed, a Semantic Web resource can represent any knowledge, including that for abstract entities. RDF in fact does not place any limitation on the type of a Resource being described. RDF's uniformity and simplicity make RDF's statements generic and applicable to natural-language statements, particularly for the encoding of object-oriented models. The RDF model encodes meaning in a set of statements in the form of *Triples*. Triples have a uniform structure that consists of three nodes – a *Subject*, consisting of a resource identified by a URI on which an assertion is being made; a *Predicate*, consisting of a property representing some kind of relationship; and an *Object*, which related to the subject given the relationship specified by the predicate. An *RDF Graph* in turn is a set of RDF triples, where nodes consist of the subjects and objects in the triples and the arcs consist of the predicates.

Whereas the subject of a statement must always consist of a URI (except in the case of blank nodes, discussed below), objects can also consist of a literal value. Literals can additionally be typed with language identifiers or XML Schema datatypes [Biron and Malhotra, 2004]. Fig. 2.2 demonstrates a simple example of an RDF graph consisting of three statements, having the unique subject `http://nepomuk.semanticdesktop.org/users/claudias/pimo#EMA3644F2000`, shortened to `claudia:EMA3644F2000` via the use of namespace abbreviations (QNames [Bray et al., 2006]). The first two statements have a URI for an object node, whereas the third's object contains a literal value. The first statement defines a type for `claudia:EMA3644F2000` via the use of the standard `rdfs:type` property and a reference to a URI representing the concept of `nmo:Email`. The second triple uses the property `nao:isRelated`, defined in another vocabulary, to relate the resources

⁵<http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/>

claudia:EMA3644F2000 and *claudia:Task176765*. The third and last statement does not relate two resources. Instead, the message subject (a literal value) of *claudia:EMA3644F2000* is defined via the *nmo:messageSubject* property. Contrary to objects consisting of a resource, no further statements can be made about the literal valued object itself (as is the case with the objects of the first two statements). This is because literals can only occur as objects of RDF statements.

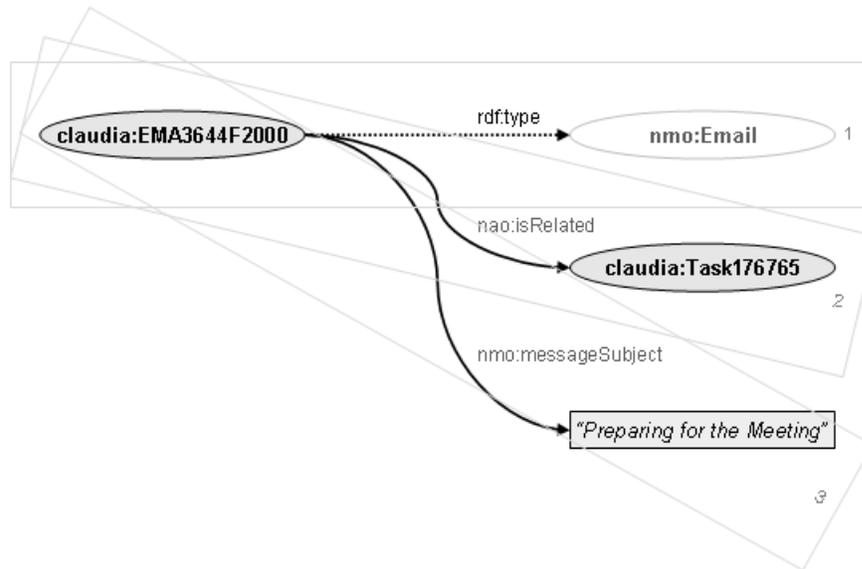


Figure 2.2: An RDF Graph with three Statements

Besides resources and literals, RDF specifies an additional type of node which can be used for both subject and object. *Blank Nodes* are very aptly called so because they are anonymous resources which are not identified by a URI. The use of blank nodes is usually employed to make further statements about an object node without the need to create a specific URI to represent it. However, the use of blank nodes can be the source of misidentification and additional problems. In fact, blank nodes are one of a number of problems identified with the RDF model and discussed later in this thesis (e.g. Section 3.1).

While RDF is used as the data model to express statements, the organisation of these

statements is possible via the RDFS extensions. The RDF Schema adds a number of features to RDF, such as classes (*rdfs:Class*) in addition to properties (*rdf:Property*); the ability to organise them within hierarchies (via *rdfs:subClassOf* and *rdfs:subPropertyOf*); the possibility of anchoring properties as predicates to specific types of subjects and objects (via *rdfs:domain* and *rdfs:range* respectively). RDFS also provides some basic terminology for human-readable annotations which however carry no formal semantics (eg. *rdfs:comment* and *rdfs:label*). Although RDF serves as the meta-language for the Semantic Web, RDFS is equally important as it allows for the definition of all RDF-based *Taxonomies* (as shown in Fig. 2.1), where a taxonomy consists of hierarchical categorisations of concepts based on their interrelated (and/or their interdependencies). The simplest *Vocabularies* make direct use of the *rdfs:subClassOf* property to define taxonomies, whereas more elaborate vocabularies are not restricted to this type of relationship between the hierarchical resources. For example, a thesaurus is a controlled vocabulary where relationships such as synonyms and antonyms can be defined. However, RDFS is still required to define these special kinds of relationships and their domain and range. In addition, although RDFS lacks the expressive power to define RDF-based *Ontologies*, all the languages that provide this required expressivity are themselves based on RDFS.

2.2.2 Ontologies

Whereas taxonomies consist of simple hierarchical organisations of concepts, ontologies model a little piece of existence or knowledge. Within the context of the Semantic Web, where the focus is on machine-understandable data, ontologies have been defined as “formal, explicit specifications of shared conceptualisations” [Gruber, 1993]. For this purpose, ontology languages and vocabularies provide the required definition sets to represent this knowledge. However, the meaning of the term ontology is broader than this context, and as such, formal vocabularies are only one way of representing them.

The main ontological components of an ontology are *Classes* and *Relationships*, formally represented by the standard RDF/S vocabularies (*rdfs:Class* and *rdf:Property*). In

essence, the definition of relationships is what constitutes an ontology, as opposed to a controlled vocabulary. A taxonomy can thus be considered as an ontology where only hyponym (and hypernym) relationships are allowed. More expressive ontology languages enable additional components, such as *Constraints* on what must hold in order for asserted statements to be valid. RDFS already introduces basic constraints in the form of relationship domain and range restrictions, thus constraining the class types that can be related by a relationship, or seen from another point of view, specifying *attributes* for a class. More complex constraints have been provided by ontology languages, such as restrictions on the cardinality of an attributes range and on actual pairs of resources that an attribute should or should not, through inference, relate. Other components provided by such languages include *Function terms*, as complex terms formed by combining existing attributes; *Rules*, as logical statements with an antecedent and a consequent that describe logical inferences that can be drawn from other assertions; as well as other features like dynamic attributes, termed *Events*.

There are different kinds of ontologies serving different needs. Usually, one distinguishes between *Representational* ontologies, *Foundational* (upper-level) ontologies, and *Domain* (lower-level) ontologies, in order of decreasing generality, abstraction and stability [Reif et al., 2008]. These three layers have also been reflected in the ontology classifications used in [van Heijst et al., 1995] and [Semy et al., 2004]. Representational ontologies, of which RDFS and OWL are an example, are themselves Ontology languages – formal languages that are employed to encode ontologies. The relationship of a representational ontology to other ontologies is quite different from the relationship between the other ontologies themselves. While upper-level ontologies generalise lower-level ontologies, all these ontologies can be seen as *instances* of the representational ontology. Concepts occurring within the latter include abstract classes and properties, constraints, etc. Foundational ontologies are high-level, domain-independent ontologies. They provide a framework by which disparate systems may utilise a common knowledge base and from which more domain-specific ontologies may be derived. They are characterised by

their representation of common sense concepts, i.e., those that are basic for human understanding of the world. Concepts expressed in foundational ontologies are intended to be basic and universal to ensure generality and expressivity for a wide area of domains. These ontologies do not cater for the details of each specific type of information and its relationships. This is instead provided by the domain ontologies, each of which is dedicated to an individual domain. Domain ontologies consist of group and personal ontologies. Group-level ontologies (e.g. an organisational ontology) are domain-specific and provide more concrete representations of abstract concepts found in the upper ontologies. They serve as a bridge between abstract concepts defined in the upper-level ontologies and concepts specified in personal ontologies at the individual level. Personal ontologies arbitrarily extend (personalise) group-level ontologies to accommodate requirements specific to an individual or a small group of individuals. Using common group-level and upper ontologies is intended to ease the process of integrating or mapping personal ontologies.

The domain knowledge modelled within an ontology is utilised with the realisation of *Individuals*, or instances of the classes represented within, and their relationships as per the defined ontological relationships. This effectively generates Semantic Web data. Fig. 2.3 shows the distinction between statements defining ontological classes, relationships, constraints etc. (domain knowledge, top), and statements associated with instances of those classes (data, bottom). The combination of the two types of knowledge results in a *Knowledge Base*. In Description Logic (DL) terms the former is identified as the TBox, whereas the latter is identified as the ABox [Baader and Nutt, 2003]. DLs are a family of formal knowledge representation languages which have been utilised by the Semantic Web community to provide a logical formalism for Ontologies. In fact, one can draw a parallel between the ontological components just described and DL modelling constructs. DL models *concepts* (classes), *roles* (relationships) and *individuals* via *axioms* – logical assertions relating roles and/or concepts.

DLs have been explicitly incorporated in OWL, a standard (set of) ontology languages [Dean and Schreiber, 2004]. On its own, RDFS lacks the expressive power required to de-

fine anything other than simple taxonomies. However it allows for the definition of custom properties, and subsequently for more expressive vocabularies, which can in turn be used for the authoring of ontologies. The OWL languages in fact, use vocabulary from RDFS to define their definition sets. As a group of knowledge representation languages endorsed by W3C (a W3C recommendation since 2004) and represented within the *Ontology* component of the Semantic Web stack (Fig.2.1), OWL is considered as one of the fundamental Semantic Web technologies.

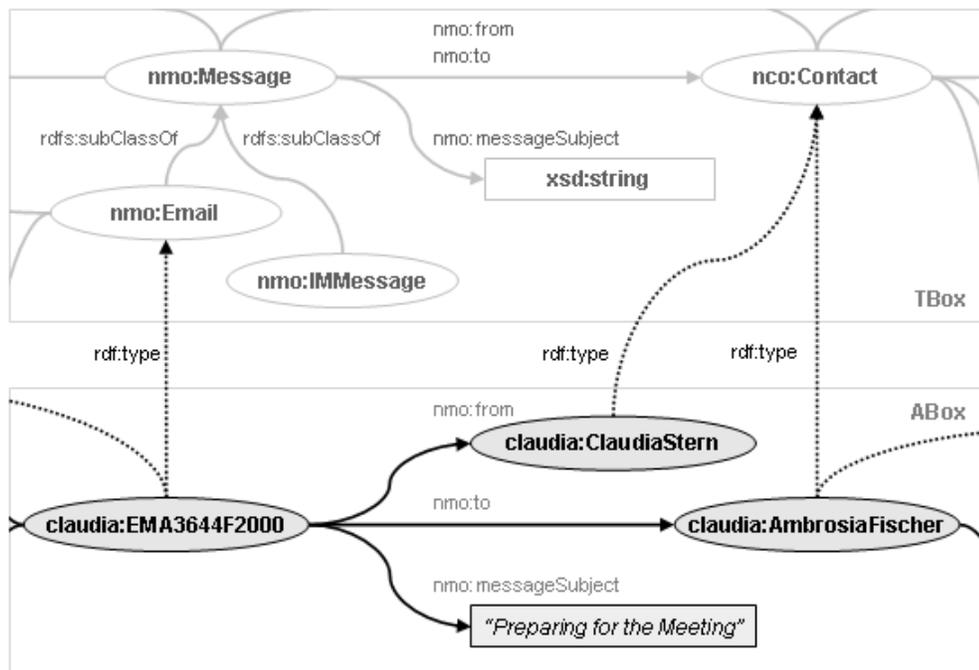


Figure 2.3: The ABox and TBox for Semantic Web data

OWL adds another level of expressivity on top of RDFS by introducing new vocabulary in order to be able to define more precise semantics. In particular, it introduces its own class concept (*owl:Class*) and new, distinctive, top-level properties that differ between properties linking individuals to individuals (*owl:ObjectProperty*) and properties linking individuals to data (literal) values (*owl:DatatypeProperty*). OWL also introduces vocabulary to define complex classes as an enumeration of specific

resources or as the union, intersection or complement of existing classes; constraints on the cardinality of a property's range (*owl:cardinality*, *owl:minCardinality* and *owl:maxCardinality*) and on the actual property values (*owl:hasValue*, *owl:allValuesFrom* and *owl:someValuesFrom*); relations between individuals (*owl:sameAs*, *owl:differentFrom* and *owl:AllDifferent*); relations between properties (*owl:inverseOf* and *owl:equivalentProperty*); as well as additional property restrictions and logical characteristics (*owl:FunctionalProperty*, *owl:InverseFunctionalProperty*, *owl:TransitiveProperty* and *owl:SymmetricProperty*). OWL also provides vocabulary for importing ontologies (*owl:import*) and a number of definitions for their annotation (e.g. *owl:priorVersion*).

OWL comes in three flavours – the unrestricted use of the language is termed *OWL Full*, whereas an additional two sublanguages impose some restriction on its use [Patel-Schneider, 2004]. *OWL DL* is based on description logics, harnessing DLs characteristics of computational completeness and decidability while imposing restrictions like the disjointness of classes, properties and individuals (e.g. a class cannot be a property, etc.). *OWL Lite* imposes even more restrictions and is intended for the extension of hierarchical classifications as provided by RDF/S with some added simple constraint features. The three sublanguages offer a trade-off between the very expressive but undecidable (OWL Full) and the less expressive but decidable (OWL DL and OWL Lite). The majority of a number of ontologies surveyed on the Web in 2006 were found to be on the lower-end of the expressivity vector [Wang et al., 2006]. OWL DL and OWL Lite both have desirable computational properties for reasoning systems, although OWL DL is subject to higher worst-case complexity. On the other hand reasoning support for OWL Full is less predictable.

As the first standard Web ontology language, OWL has been instrumental in advancing the Semantic Web paradigm, by providing a higher level of expressivity in the design of ontologies. Consequently, it enabled the generation of Semantic Web data with a rich enough semantics so that it can be processed by dedicated reasoning systems.

However, since its endorsement by W3C as a recommendation, a number of concerns have been raised about the language, mostly in relation to its syntax and semantics [Motik and Horrocks, 2006, Grau et al., 2008]. In particular, these identified inconsistencies between the interpretation of its three different syntaxes, ambiguous interpretations resulting from OWLs reliance on the separation between object and data property names, loss of information when translating OWL Abstract Syntax into OWL RDF syntax, the unclear relation between OWL and RDF semantics and limitations in its expressivity. These problems have a negative effect on the implementation of OWL Application Programming Interfaces (APIs) as well as applications processing OWL data. This is to be avoided, as applications using Semantic Web data are crucial to the realisation of the Semantic Web, as shown by the inclusion of the application layer in the Semantic Web Stack, Fig. 2.1.

A number of publications have since proposed solutions to remedy these problems, culminating in the 2009 proposal for a newer version of OWL: OWL2 [Grau et al., 2008] and a subsequent W3C recommendation in October of the same year [Hitzler et al., 2009]. A metamodel based on the Meta Object Facility⁶ (MOF) ensures that OWL2 is defined unambiguously, solving the above-identified ambiguity issues due to OWL's multiple syntaxes. This in turn simplifies the implementation of OWL APIs and eases the interoperability in between the various language implementations. In addition OWL2 introduces some new useful features, such as disjoint classes and properties, and composite properties to also enable propagation along properties and further improve expressivity.

2.2.3 Syntax and Semantics

The validity of a statement in a language depends on two definitions: *syntax* and *semantics*. A syntactically valid statement is one that is constructed using a correct grammatical structure, as specified by a language. A semantically valid statement on the other hand is one that respects the constraints and restrictions of a language structure being used

⁶<http://www.omg.org/mof/>

such that it is well defined and thus carries meaning. The two are independent, so that a syntactically valid statement does not imply a semantically valid one and vice versa. However, they are not separate enterprises and every syntax needs to have clear underlying semantics, whereas every semantics needs a syntactic form to clearly express the underlying model [Robie et al., 2001].

In computer languages, a statement is checked for syntactic validity via syntactic analysis, which determines whether components in the statement form an allowable expression given the syntax rules in the language syntax specifications. The original *serialisation* syntax defined for the RDF representation model is based on XML (RDF/XML) [Beckett, 2004]. The formal grammar for this syntax is annotated with actions generating triples of the RDF graph. The triples are in turn encoded using the N-Triples serialisation format [Grant and Beckett, 2004]. The RDF/XML syntax representation of the RDF graph represented in Fig. 2.2 are shown in Listing 1 whereas its N-triples encoding is shown in Listing 2 in the form:

`<subject> <predicate> <object> .`

Whereas N-Triples require the definition of the full URIs for resources, RDF/XML allows namespace prefixes at the beginning to improve readability.

As explained earlier, as a data model RDF/S is independent of any serialisation format such as XML. In fact there are other formats that can encode RDF/S data, such as Notation3 (N3) [Berners-Lee and Connolly, 2008] which is an extension of N-Triples. Although not as well-known as XML, N3 is a more compact and readable alternative to the RDF/XML syntax while also allowing greater expressiveness. Additionally, Turtle [Beckett and Berners-Lee, 2008] is a subset of the N3 language (and a superset of the N-Triples format) which takes the most useful and appropriate N3 extensions to N-Triples but unlike N3 does not go beyond the RDF model. The same graph represented in Fig. 2.2 is again shown in Listing 3, this time serialised as Turtle. Prefixes are again provided at the start, resulting in a more compact serialisation. Turtle (and N3) allows for a number of shorthand notations which are not permissible in Notation 3. In particular, the

Listing 1.

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns/#"
        xmlns:nao="http://www.semanticdesktop.org/ontologies/nao/"
        xmlns:nmo="http://www.semanticdesktop.org/ontologies/nmo/"
        xml:base="http://nepomuk.semanticdesktop.org/users/claudias/pimo">
  <rdf:Description rdf:about="#EMA3644F2000">
    <rdf:type rdf:resource="nmo#Email"/>
    <nao:isRelated rdf:resource="#Task176765"/>
    <nmo:messageSubject>Preparing for the Meeting</nmo:messageSubject>
  </rdf:Description>
</rdf:RDF>
```

Listing 2.

```
<http://nepomuk.semanticdesktop.org/users/claudias/pimo#EMA3644F2000>
  <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
  <http://nepomuk.semanticdesktop.org/users/claudias/nmo#Email> .

<http://nepomuk.semanticdesktop.org/users/claudias/pimo#EMA3644F2000>
  <http://www.semanticdesktop.org/ontologies/nao/isRelated>
  <http://nepomuk.semanticdesktop.org/users/claudias/pimo#Task176765> .

<http://nepomuk.semanticdesktop.org/users/claudias/pimo#EMA3644F2000>
  <http://www.semanticdesktop.org/ontologies/nmo/messageSubject>
  "Preparing for the Meeting" .
```

‘;’ symbol allows for the grouping of statements that share a subject (as in the example shown), and allows the ‘a’ in place of the *rdf:type* predicate.

Listing 3.

```
@prefix nao: <http://www.semanticdesktop.org/ontologies/nao/> .
@prefix nmo: <http://www.semanticdesktop.org/ontologies/nmo/> .
@prefix claudia: <http://nepomuk.semanticdesktop.org/users/claudias/pimo#> .

claudia:EMA3644F2000
    a nmo:Email ;
    nao:isRelated claudia:Task176765 ;
    nmo:messageSubject "Preparing for the Meeting" .
```

Over time, new useful ways to encode RDF graphs have been devised. A notable example is RDFa [Adida et al., 2008], which allows the encoding of RDF data alongside human-readable data directly into Web pages, thus making them also to an extent machine-readable. In the period between 2008 to 2009, RDFa was adopted by both Google⁷ – to enable ‘rich snippets’ in web pages, as well as Yahoo⁸ – to enhance search via semantically-enriched content.

Employing the correct syntax for RDF/S, OWL and any other Semantic Web language being used means that the generated data can successfully be parsed. However, this does not guarantee the semantic validity for the data. Semantic resolution works out the implications of a syntactically valid expression against a set of conditions (semantic rules) in a corresponding formal semantics specified for a language. The role of formal semantics is to expose the meaning of assertions to machines. Such formal notions of meaning can be captured in mechanical inference rules. The formal semantics of the RDF language [Hayes, 2004] are specified through model theory – a formal semantic theory which relates

⁷<http://googlewebmastercentral.blogspot.com/2009/05/introducing-rich-snippets.html>, 16/05/2009

⁸<http://www.ysearchblog.com/2008/03/13/the-yahoo-search-open-ecosystem/>, 16/05/2009

expressions to interpretation. Model-theoretic semantics assumes that the language in questions refers to a *world*, and an asserted sentence amounts to a *constraint* on that world, or alternatively, that the world is so arranged as to be an *interpretation* which makes the assertion true. The theory describes the minimal conditions that a world must satisfy in order to assign an appropriate meaning for every expression in the language. RDFS and OWL are considered to be *semantic extensions* of RDF because they both impose extra semantic conditions on the meaning of terms in the RDF vocabulary. The semantics of RDF are defined via a set of axioms and corresponding *entailment* rules. In model theory, the notion of entailment enables the definition of valid inference rules, which state that if a given pattern of triples is present in a graph, this *entails* the existence of additional triples in that same graph. Two RDFS entailment rules are provided in Def. 1 and Def. 2 below.

Definition 1.

```
uuu rdfs:subClassOf xxx .
vvv rdf:type uuu . ? vvv rdf:type xxx .
```

Definition 2.

```
aaa rdfs:domain xxx .
vvv aaa yyy . ? vvv rdf:type xxx .
```

Interpreting the first rule, one can say that the semantics of RDFS imply that every instance of a class **uuu** is also an instance of **uuu**'s superclass **xxx**. A realisation of this rule is provided as an example in Listing 4. The second rule states that if a resource **vvv** is related to an object via a property whose domain is class **xxx**, the assertion that **vvv** is an instance of **xxx** is implied. An example of such an inference is shown in the example in Listing 5.

The axioms and entailment rules defined for a language are subject to some assumptions, which determine how the rules are interpreted. In particular, both RDF and OWL semantics apply the open-world assumption (OWA). OWA presumes that its knowledge of the world is incomplete and that this lack of knowledge does not imply falsity. On

Listing 4.

```
<nmo:Email> <rdfs:subClassOf> <nmo:Message> .
<http://nepomuk.semanticdesktop.org/users/claudias/pimo#EMA3644F2000>
  <rdf:type> <nmo:Email> .
-->
\begin{verbatim}
<http://nepomuk.semanticdesktop.org/users/claudias/pimo#EMA3644F2000>
  <rdf:type> <nmo:Message> .
```

Listing 5.

```
<nmo:messageSubject> <rdfs:domain> <nmo:Message> .
<http://nepomuk.semanticdesktop.org/users/claudias/pimo#EMA3644F2000>
  <nmo:messageSubject> <"Preparing for the Meeting"> .
-->
<http://nepomuk.semanticdesktop.org/users/claudias/pimo#EMA3644F2000>
  <rdf:type> <nmo:Message> .
```

the other hand, the closed-world assumption (CWA) presumes that what is not currently known to be true, is false. Whereas the OWA states that everything that is not known is undefined, the CWA implies that everything that isn't explicitly specified is false. While OWA is more flexible, and more likely to generate new statements based on ones that already exist in the models, its non-monotonic nature hinders computability and largely increases the complexity of RDF data. On the other hand, while CWA is much more prone to generate errors, it is totally deterministic and one can always compute whether any statement under the assumption is true or false. The implications of a CWA versus an OWA were also extensively discussed in the context of relational databases, which also raised the same concerns with respect to indefinite data [Minker, 1982].

The assumption of a closed- or an open-world has therefore a huge impact on how the truth-value of asserted statements can be inferred. A deductive reasoner operating

on the CWA will infer that a statement is false from its absence, whereas one operating on the OWA will not be able to make any inference. Applied to the RDFS entailment rules provided above, whereas the default OWA interpretation meant that new statements could be inferred from other statements, the CWA interpretation considers the RDFS definitions (e.g. *rdfs:subClassOf*, *rdfs:domain*) as strict *constraints* on their instances. Therefore in the absence of the consequent statement in Listing 5, stating that the resource in question is an instance of *nmo:Message*, the preceding statements use of the *nmo:messageSubject* property is considered *invalid*.

This enforces the strict use of an ontology, and prevents anyone from being able to say anything about anything. However this also has its advantages, as it prevents the incorrect use of *nmo:messageSubject* to assert something about a non-message, e.g. an *nco:PersonContact* instance. Therefore, although a reasoner operating on a CWA is better suited to detect inconsistent data, the CWA is only practical within a closed system that has complete control over its data. Within the context of the Semantic Web, where the essential dogma is that anybody can state anything about anything, the use of the CWA is practically impossible. Always within the context of the Semantic Web, it makes sense for both RDF and OWL Semantics to assume an open-world. However, their explicit reliance on the OWA means that semantic reasoners based on the languages within other (closed) systems are also limited to an OWA, thus allowing the possible generation of conflicting data.

2.2.4 Named Graphs

Sometimes it is desirable to be able to make assertions about other asserted *statements*, rather than a resource. This equates to the provision of metametadata, i.e. data about metadata. This is useful in order to be able to say things about a statement such as who stated it, when it was made, under which conditions it is true, conditions on its reuse, and additional information pertaining to provenance and trust. For this purpose, RDF provides for the *reification* of statements, a process through which a previously un-

addressable statement is made available for conceptual and computational manipulation via a proxy, defined as an instance of *rdf:Statement*. An example of how the third triple in the graph in Fig. 2.2 can be reified to specify its creation date and ownership (using vocabulary from the Dublin Core Metadata initiative⁹) is given in Listing 6 below in Turtle format:

Listing 6.

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix dc: < http://purl.org/dc/elements/1.1/> .
@prefix claudia: <http://nepomuk.semanticdesktop.org/users/claudias/pimo#> .

claudia:statement3
    rdf:type rdf:Statement ;
    rdf:subject claudia:EMA3644F2000 ;
    rdf:predicate nmo:messageSubject ;
    rdf:object "Preparing for the Meeting" ;
    dc:creator claudia:ClauiaStern ;
    dc:date "2010-01-26" .
```

Although useful, reification was intended to provide metadata about individual statements. However, it would be more useful if it was possible for such metadata to be defined over *entire* graphs. The data model for RDF defines a graph \mathbf{g} as a collection of triples $\mathbf{t} = (\mathbf{s}, \mathbf{p}, \mathbf{o})$ such that:

$$\mathbf{g} = \{ \mathbf{t} \mid \mathbf{t} \in \mathbb{N}, \mathbf{t} > 0 \}$$

Given the RDF model, although individual triples within a graph can be reified and managed, it is not possible to do the same for graphs. The limitations of RDF reification for this purpose are well known [Carroll and Stickler, 2004]. *Quads* have been proposed as a solution, where a quad consists of an RDF triple and an additional URI. The nature of

⁹<http://dublincore.org/>

this proposed additional subpart varies from the representation of an information source to ‘contexts’.

A variation of quads that was most well-received by the research community is that put forward by the *Named Graphs* [NGs] approach, where every RDF Graph is identified by a unique name [Carroll et al., 2005]. NGs are an extension on top of RDF, providing a solution for issues such as data handling, provenance and trust by enabling the modularisation and identification of RDF graphs to make data handling easier. Essentially NGs provide for RDF graphs what reification provides for RDF triples, without any need at all for reifying the individual triples. The notion of NGs in fact, renders reification redundant. A named graph is defined as a pair (\mathbf{n}, \mathbf{g}) , where \mathbf{n} is a unique URI reference denoting the assigned name for graph \mathbf{g} . Such a mapping fixes the graph \mathbf{g} corresponding to \mathbf{n} in a rigid, non-extensible way. The URI representing \mathbf{n} can then be used from any location to refer to the corresponding set of triples belonging to the graph \mathbf{g} . Consequently graph names, like URIs, must be globally unique. A graph \mathbf{g}' consistent with a different graph \mathbf{g} named \mathbf{n} cannot be assigned the same name \mathbf{n} . Two different datasets asserting graphs \mathbf{g} and \mathbf{g}' and having the same URI for a name are in a contradictory state. The graph presented in Fig.2.2 is below shown (Fig.2.4) as a named graph (*claudia:Graph01*) with the same additional descriptive metadata provided for one statement via the reification in Listing 6.

Although named graphs are a widely-popular notion, they are not yet supported by any of the standard Semantic Web representational languages, i.e. so far, no standard named graph-specific vocabulary has been provided. However, a number of the serialisation formats presented earlier have been extended to support their representation, e.g. Notation3 and Turtle. The TriG format [Bizer and Cyganiak, 2004] is a straight-forward extension of Turtle, created for the serialisation of NGs and RDF Datasets, whereas TriX¹⁰ is its XML-based alternative. The named graph depicted in Fig.2.4 is below shown serialised as TriG.

¹⁰<http://sw.nokia.com/trix/TriX.html>

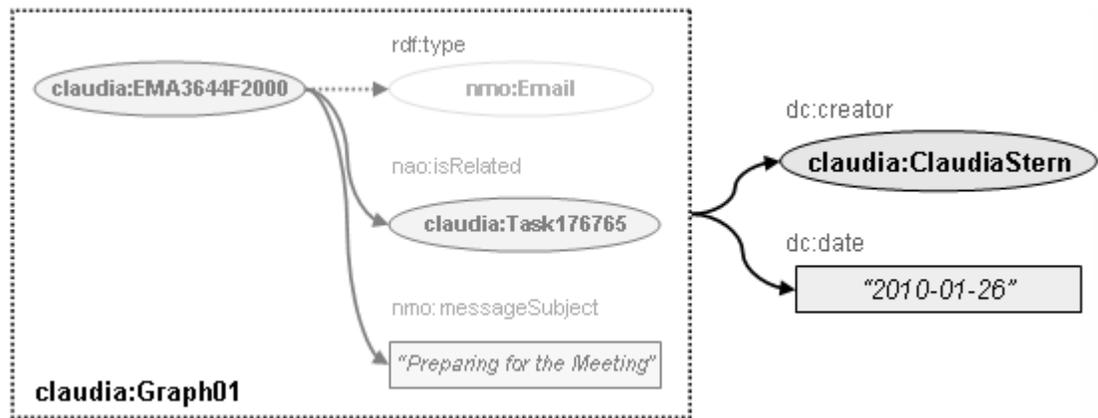


Figure 2.4: Using Named Graphs as an alternative to statement reification

TriG introduces new syntax over Turtle, such as the use of the ‘{’ and ‘}’ symbols to group triples into a named graph. Once the RDF model has been extended to the concept of named graphs, all triples are required to form part of some graph. The so called *Default Graph* contains all triples which are not defined to be within any other graph by default. In the TriG serialisation above, the triples required to assert the two statements about the graph in question (*claudia:Graph01*) are represented within the default graph – denoted as an unnamed graph.

Listing 7.

```
@prefix nao: <http://www.semanticdesktop.org/ontologies/nao/> .
@prefix nmo: <http://www.semanticdesktop.org/ontologies/nmo/> .
@prefix dc: < http://purl.org/dc/elements/1.1/> .
@prefix claudia: <http://nepomuk.semanticdesktop.org/users/claudias/pimo#> .

claudia:Graph01 {
    pimo:EMA3644F2000
        a nmo:Email ;
        nao:isRelated claudia:Task176765 ;
        nmo:messageSubject "Preparing for the Meeting" . }
{
    claudia:Graph01
        dc:creator claudia:ClaudiaStern ;
        dc:date "2010-01-26" . }
```


3 Towards a Social Semantic Desktop

In this section I will introduce the motivation for a *Social Semantic Desktop* (SSD) as a solution to the problem of information management within one's personal sphere, extended across a web of spheres of collaborating individuals. While also discussing related work on the topic, the focus will be on the requirements set-out by the consortium responsible for transforming the SSD from a prototype-supported vision to a widely-implemented reality within the NEPOMUK project [Decker and Frank, 2004].

3.1 Motivation

The minds responsible for the progression of ideas, visions and paradigms leading to the development of both the WWW and the Semantic Web were essentially driven by their perceived need for non-human support in the management and organisation of one's information. Bush's Memex and Engelbart's NLS (Section 2.1) can in fact be considered as solutions for personal information management. Since then, the information management problem that motivated these scientists has largely remained unsolved; not because renewed attempts or efficient solutions haven't been proposed, but rather because both amount and diversity of digital personal data has increased exponentially. This is further exacerbated with the ever-increasing ways of sharing and storing personal data. At a time were humans are increasingly dependent on digital information, this problem plagues everyone owning or having access to a digital medium; especially people working in a collaborative setting, for whom the efficient knowledge organisation, sharing and retrieval is crucial.

Focusing on the personal desktop, each application is an isolated island of data, managing its own data and unaware of other applications harbouring related or relevant data [Sintek et al., 2007a]. Although individual vendors may decide to allow their applications to interoperate (e.g. email client is aware of contacts in an address book) there is still no consistent approach for allowing a system-wide exchange of data between applications. Similarly, each individual desktop is itself a bigger isolated (group of) data island(s), with no standardised architecture for data exchange in between. Only support for low-level communication is provided, limited through just a handful of media such as email, instant messaging or shared server upload. As a result, sending out a file to a number of people, multiplies the effort of managing the file by the same number of contacts it is being sent to, as information stored on different desktops is contained within different data islands [Decker et al., 2005].

Moreover, data exchanged over desktops can generally only be interpreted by the human recipient(s). This is due to insufficient metadata support for desktop data, limiting interoperability between both applications on a single desktop, as well as on multiple desktops. Consequently machines cannot provide additional support, e.g., by recognising that an incoming email is related to file folder, both of which are related to a project that the user is participating in.

3.1.1 Semantic Desktop

In theory, the approach for data on the Semantic Web is not limited to the Web, and the above-mentioned desktop problems have the same root cause as the one afflicting the Web. Semantic Web technologies can be employed to provide a means to build the semantic bridges necessary for data exchange and application integration on and across multiple desktops. Ontologies are employed to express personal knowledge formally, enabling personal information on the semantic desktop to be lifted onto an RDF representation. These represent both conceptualisations of desktop data that can be shared in between multiple applications and desktops (e.g. files, folders, email, contacts, and their relationships),

as well as the desktop user's personal mental models for the data within (e.g. photos, projects, people, groups, and their relationships).

Thus a Semantic Desktop possesses an additional semantic layer providing rich metadata about both the physical (digital) information entities on the desktop as well as the abstract (conceptual) non-information entities [Jacobs and Walsh, 2004] in the user's conceptualisation of their personal information sphere. This metadata does not only provide the standards required to ensure interoperability between the various components in a semantic desktop setting, but it also makes the desktop content available to automated processing, thus enabling machines to support the user with the organisation of their data. In even more sophisticated scenarios, machines can employ reasoning techniques over the stored metadata to enable the discovery of additional data that is not directly available, or visible, to the user. For example, the machine could assist the user with their calendar scheduling by warning them if they commit to two unrelated events at the same date and time.

To date, a number of research attempts to design and implement a Semantic Desktop have been made. The Haystack [Quan et al., 2003] project's aim was to bring the Semantic Web to end-users by leveraging Semantic Web technology to support them with managing personal information on their desktop. An individual's 'haystack' is then a personal repository of all the information they have come across on both the Web and the desktop. In this way, separate pieces of information about the same resource that used to require browsing through different desktop folders and/or Web sites can be unified into one display. The DeepaMeetha networked semantic desktop [Richter et al., 2005] is a personal knowledge management tool that integrates information items on the desktop into a coherent user environment, based on the Topic Map paradigm [Pepper, 2010]. IRIS [Cheyer et al., 2005] (Integrate, Relate, Infer, Share) is another semantic desktop application framework that enables users to create a *personal map* of information objects on their desktop. The framework is supported by an ontology-based knowledge store powered by ontologies which capture every aspect of a user's working environment while

ensuring semantic interoperability with other IRIS semantic desktop installations. Both IRIS and Haystack focus not just on the personalisation of information but also on the personalisation of its management. In fact, one of the set requirements for Haystack's implemented interface was to let users manage their information in ways that make most sense to them. Similarly the IRIS semantic desktop emphasises its ability to let users organise their information in ways that suit their individual needs.

The largest challenge faced by these semantic desktop implementations was their adoption. Their entire infrastructure was built on top of that of the underlying desktop operating system, replacing existing applications and functionality, instead of integrating the new technology into the existing architecture. This makes for a serious drawback in terms of adoption prospects, as this requires a significant investment by the users to adapt to the new working environment. Alternatively, semantic desktop technology can be integrated within existing operating systems and end-user applications, extending and making use of existing technology rather than replacing it. Gnowsis [Sauermann et al., 2006] is an example of an approach in this direction, and its goal is to *enhance*, rather than replace, existing desktops (and applications) with Semantic Web functionalities to assist with personal information management. The Gnowsis semantic desktop employs Aperture¹¹ adapters to extract metadata from existing desktop objects, and mechanisms to allow the user to generate further metadata through their linking. The metadata is stored into a central RDF store and forms part of the Personal Information Model Structures (PIMOs) of each user. Another similar approach is described in [Möller and Handschuh, 2007], reporting specifically on the possibilities of harnessing the power of metadata-enabled file systems to search indices and link desktop data to easily create a light-weight Semantic Desktop system.

¹¹<http://aperture.sourceforge.net/> 05/02/2010

3.1.2 Social Semantic Spaces

Web 2.0 (Section 2.1.3) has revolutionised the Web by harnessing the power of human collaboration to interactively generate further valuable data and share it over over numerous media such as blogs, wikis, and social networking technologies. The Semantic Web now promises to revolutionise Web 2.0 technologies, by exposing the collaboratively-generated data to machines, to attain a *Social Semantic Space*. Web 2.0 technologies have started to deal with metadata, leading to terms such as semantic blogs and semantic wikis. Augmenting these entities with machine-interpretable metadata enable new ways of querying and also navigating the information contained within. Conventional blogging tools for example, are not able to add any semantics to the weblog posts apart from free-text keywords, topics or *Tags* (with the exception of semantic blogging approaches such as [Möller and Decker, 2005]). Tagging is in fact a prime example of how Web 2.0 concepts and technology can benefit from the integration of metadata into their architecture. The collaborative creation and use of tags to categorise Web 2.0 content generates a *Folksonomy* (folk taxonomy) – a system of classification resulting from the social tagging performed within a particular platform or website. Folksonomies have become popular because they are relatively easy to generate, they do not necessitate either a controlled vocabulary or any top-down mediation. Despite being a free-for-all system, it has been shown that a consensus around stable distributions and shared vocabularies emerges nevertheless [Halpin et al., 2007]. However, being collections of free-text tags restricts their machine-processability to a dependency on arduous natural language processing techniques such as string matching. This unearths problems such as dealing with language ambiguities, grammatical errors and acronym resolution. There were numerous attempts to augment folksonomies and tags with some form of semantics, supported by the design of models and ontologies that capture the involved concepts and processes. In one particular instance, the tripartite tagging model (*User, Resource, Tag*) behind an existing tagging ontology (Tag Ontology [Newman et al., 2005]) was extended to (*User, Resource, Tag, Meaning*) to also express meanings of tags via URI's (MOAT Ontology

[Passant and Laublet, 2008]).

I have also contributed to the design of an additional tagging ontology – SCOT (Social Semantic Cloud of Tags); to support the re-use of tags across collaborative tagging spaces [Hak Lae Kim, 2009]. SCOT promotes the federation of existing ontologies by reusing many of their definitions, as some of them are more concerned with the concept of tags and their meaning, whereas others are more adequate to describe folksonomies and their use. The new vocabulary provided in SCOT supports semantic collaborative tagging, enabling the semantic alignment of tags generated by multiple users as well as their re-use. In theory, SCOT also enables personal tag clouds to be exported and re-used across different tag-supportive platforms. This is illustrated in Fig. 3.1, where a user called Claudia can re-use a SCOT representation for a tag generated from a folksonomy on one platform (Blogger) to tag a resource on another (Flickr) via *scot:hasTag*. The figure is split across three dimensions: (shared) resources at the top, shared tag metadata in the middle, and their shared meaning at the bottom. This meaning gives tag instances a particular semantics by tying them to specific URIs. In Fig.3.1, although Claudia and another user called Dirk use different names for a tag (Semantic Web, SemWeb), defining a unique meaning via *moat:hasMeaning* ensures that machines know that the tags are referring to the same concept. The URI representing the concept in this case is defined in DBpedia [Lehmann et al., 2009] – one of the datasets related to the Linked Open Data initiative (Section 2.1.4) which contains data extracted from Wikipedia.

3.1.3 A Source and End-point of the Semantic Web

The social semantic tagging approach described in the previous section is an example of a social technology being extended with semantics towards a social semantic space. As explained in Section 3.1, the user’s conventional desktop is neither semantic, nor social, in the sense that no standardised architecture for representation and interoperation is provided. To realise an SSD thus requires the extension of the user’s personal (desktop) information environment with semantics, at the same time ensuring that these semantics

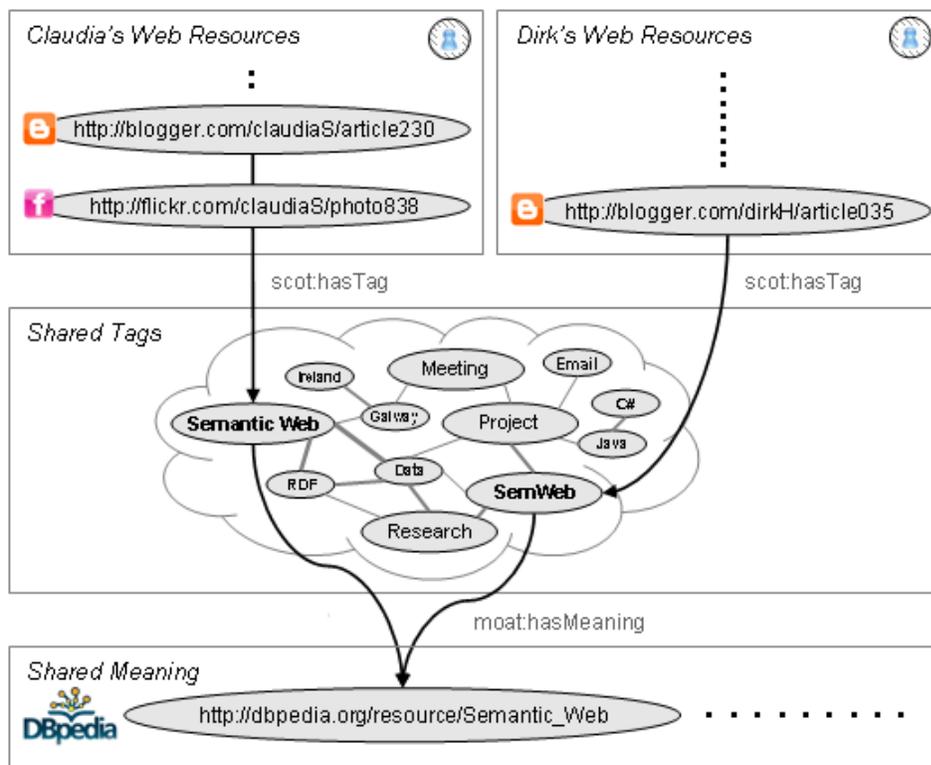


Figure 3.1: Sharing tags with specific meanings across different users and platforms

can be interpreted across multiple user information spaces. This necessitates the provision of standard knowledge models for the representation and generation of structural and content-related metadata. This was also a requirement for the Gnowsis semantic desktop (Section 3.1.1). However, that approach was largely confined to just the user's personal information space, and did not address the integration of multiple information spaces.

The NEPOMUK Social Semantic Desktop [Groza et al., 2007] approach has foreseen the next logical step, and its ambition is to effectively integrate the desktop within the global social semantic space. This idea is not entirely new, having had already seen research in this direction at the application level, an example of which is an attempt to integrate desktop data into a semantic blog [Möller and Decker, 2005]. However the vision of the SSD is broader, aiming to bootstrap the power of the Semantic Web to create

an extended working environment, where information items on the desktop are treated as Web resources. By aligning the SSD and Semantic Web paradigms, the user's semantic desktop can be seen as source and end-point of the Semantic Web [Sintek et al., 2009], where the borders between individual applications and the physical workspace of different users are loosened to transform the conventional desktop into a seamless, networked working environment.

The path from a conventional syntax-based personal desktop to a Web-integrated SSD is illustrated¹² in Fig.3.2 . Enhanced with standardised knowledge representation models, multiple desktops can interoperate, enabling collaboration in between SSDs. The integration of Semantic Web data and personal data on multiple semantic desktops into one extended information space transforms the user's SSD into a semantic collaborative space.

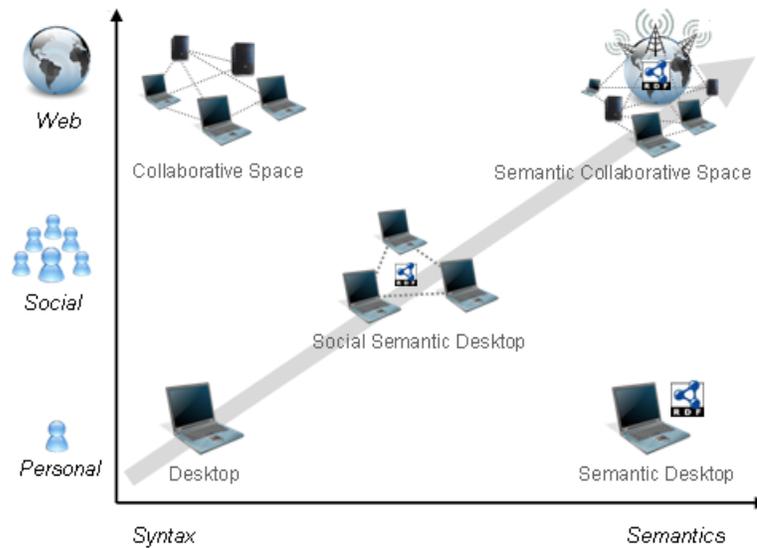


Figure 3.2: From a syntax-based personal desktop to a Web-integrated SSD

¹²This figure is an adaptation of a similar illustration presented in [Decker et al., 2005]

3.2 Knowledge Representation Requirements

Knowledge representation is a fundamental aspect of the SSD. Knowledge models are required to serve as standards for the representation of heterogeneous data within the desktop, across a network of desktops, as well as on the Web. The RDF Model and ontologies served as a good starting in the design of these standards. However, a number of research questions with respect to the required high-level knowledge representation were immediately raised. In particular, the design of the semantic layer of the SSD faced the following challenges:

- i. How can a common knowledge representation with the required level of expressivity be established across multiple desktop systems?
- ii. How to cope with the heterogeneity of knowledge models, especially knowledge modules with potentially different interpretation schemes?
- iii. How to support the tailoring of knowledge models towards different needs in various exploiting applications?
- iv. How to represent existing legacy data on the desktop and express both complex application-level annotations and simple end-user oriented annotations in one coherent model?

To tackle these challenges, four fundamental requirements for knowledge representation on the SSD were outlined, as discussed in the following subsections.

3.2.1 Enabling Semantic Interoperability

By its very nature, the SSD depends on a shared understanding of the underlying conceptualisations, i.e., the semantic interoperability of heterogeneous information in between different applications, desktops and Semantic Web end-points. Semantic Web languages and protocols have been employed to formalise these conceptualisations and to co-ordinate

local and global information access. Semantic Web technology is intended to serve as a middleware for a wide variety of applications in domains such as e-business, e-government, and personal/organisational knowledge management. The distributed datasources for these applications are typically generated and maintained by autonomous entities (e.g. applications, online databases, etc.). The SSD architecture is required to provide protocol standards as well as data and information representation standards for federated information access and services across multiple sources and desktops. Ontologies provide the necessary representation schemes to be shared between several information providers and consumers. However, a clean ontology design for SSD scenarios is required, in order for diverse applications with different goals and intended user groups to be able to access the ontologies and knowledge bases and (re)present them adequately.

3.2.2 Effective handling of multiple knowledge modules

In order to correctly represent the social dimension of distributed knowledge generation and usage, the SSD requires a module concept which supports encapsulation of statements and the possibility to refer to such modules. This is also largely relevant to the issues of provenance and trust information, with regards to imported and exported data. Given the basic RDF model, importing external RDF data from a remote application presented some difficulties, mainly revolving around the fact that there are no standard means of retaining provenance information of imported data. This means that data is propagated over multiple applications, with no information regarding the original provider and other crucial information like the context under which that data is valid. This results in various undesirable situations – such as outdated RDF data with no means to update it, as well as redundant RDF data which cannot be entirely and safely removed. The Named Graphs (Section 2.2.4) paradigm introduced earlier is suitable to address this deficiency, and therefore the SSD’s underlying RDF model needs to be extended with the named graphs concept.

3.2.3 Supporting Multiple Views and Semantics

A central principle of the Semantic Web is that it is an open world where anyone can add new information about existing resources. Since the Web is a huge information space where everything can link to everything else, it is impossible to rule out that a statement is true, whether it can become true in the future or under which context. However, there can be logical inconsistencies. Hence, as is the case with the official OWL and RDF/S semantics, the global Semantic Web relies on open-world semantics, with no unique-name assumption. On the other hand, within a single application using Semantic Web technology people would find it very difficult to understand the logical meaning and consequences of the open-world assumption; the closed-world and unique-name assumptions are much easier to understand for most users, as has been argued e.g. by [de Bruijn et al., 2005]. Also, as explained in Section 2.2.3, a reasoner operating on a CWA is better suited to detect inconsistent data. Thus, whereas on the Web the OWA is necessary, the comparably smaller amounts of data present in a desktop environment do not pose the same requirements on robustness as the global Web does, and the OWA is therefore not necessary. The CWA is more practical for the personal desktop, the latter being a closed system with complete control over its data. However, these complete control over data does not extend to between multiple desktops, on which an OWA, as on the Web, is more practical. This calls for the SSD's data handling to be able to distinguish between data per se and the semantics or assumptions on that data. If these are handled analogously, it becomes possible to build applications that handle local data as a closed world while processing external data with open world (or other) semantics.

Secondly, the social aspect of the SSD brings about the need to support multiple data views, since different agents (both humans and machines) can be interested in different aspects of the data in different circumstances. Looking at the contents of an image folder for instance, a user might wish to see related concepts for an image (e.g. Galway, Project), or other files related to it (e.g., Galway Meeting Agenda), but not necessarily both concurrently, even if the information is extracted from the same dataset. Addition-

ally, advanced users might wish to see data that is not usually visible to regular users, like additional concepts and instances related to the file (e.g. other cities, projects). In order to retrieve such additional results, the viewing application is required to *realise* the RDF/S semantics over the data, to generate custom, expanded *views*. A view in this context consists of dynamic, virtual data computed or collated from the original data, where the best view for a particular purpose depends on the information that a user needs. For example, from a number of data modules describing the profiles of the user's contacts (e.g. via FOAF [Brickley and Miller, 2005]), the user might only be interested in who knows who (i.e. the social network). For the purpose, a restrictive view over the profiles can omit all personal information, bar their name and social connections. This view concept needs to be taken into consideration in the design of the SSD's architecture, in order to support views at the lowest knowledge representation level.

3.2.4 Unified, epistemologically-adequate modelling primitives

Modelling knowledge about existing desktop resources is not straightforward, given the multitude of existing applications and native representation formats in use. The knowledge modelling needs to tie in as much as possible into the existing operating system technology and behaviour, as this allows the design of user-friendly application interfaces that do not require the users to change their environment or habits. Whereas some attempts at standardising knowledge about the desktop resources and environment have been made, no definite standard was available. A broad array of utilities has already been developed for the extraction of RDF metadata from desktop sources given a number of domain ontologies. However, these ontologies lack the required expressivity to model the users' desktop activities as required by the SSD. Therefore the approach is to build upon these experiences in order to provide a unified vocabulary for the description of typical native resources that are of interest to the users. Furthermore, in knowledge-intensive scenarios like the SSD, knowledge modelling is also performed directly by the end-user who can continuously invent new vocabulary for describing their information

items through activities like annotation, tagging, etc. Even if much of the complexity of the underlying representation formalism can be hidden by adequate UIs, it is desirable to minimise the epistemological gap between the way end-users would like to express their knowledge and the way it is represented on the system. Thus, in order to ensure that this gap does not become too wide, the vocabularies and modelling primitives are required to be epistemologically adequate (i.e.. more akin to how the user would perceive them).

4 Email and Personal Information Management: Problems and Approaches

Since the establishment of ARPANET (Section 2.1.1), electronic mail has persisted as the most popular digital communication media in use within digital collaboration infrastructures. Despite the rise of competing technologies, email remains a crucial business communication tool and an important source of enterprise information and knowledge. Suffices to say that according to a 2003 survey about 80% of users prefer email as a business communication medium¹³, surpassing even the use of voice over phone as their preferred means of communication. This is because unlike face-to-face or phone conversations, email frees the participants from the constraints of time and space, allowing them to communicate at the most convenient times and places [Clark and Brennan, 1991]. Email's successes are attributed to a very simple protocol, based on a communication model that is fundamentally right for a communication system [Armstrong, 2000]; naming asynchronicity, the concept of threading and the fact that it is a command-central system as its major design advantages. However, the email model has also many disadvantages, the majority of which are attributed to the many unintended ways in which people use email, as opposed to the functions supported by its original design [Whittaker et al., 2006].

The term *Email Overload* [Whittaker and Sidner, 1996] is a specialisation of the more general *Information Overload* term, and it refers to the use of email for such functions. Since email is one of the channels via which a great deal of new information continuously reaches the user's desktop, email overload is therefore a threat to the user's prospectives

¹³<http://www.mariosalexandrou.com/technology-trends/2003/80-percent-of-users-prefer-email.asp>

of being in control of their information.

In this chapter I first provide an overview of the possible causes for email overload, and the guidelines and recommendations suggested by top researchers in the area. I then provide an overview of the various approaches at reducing the problem via the better structuring – at a conceptual (model) and/or practical (implementation) level; and management of email-generated information. After reviewing all existing techniques and approaches to counteract this problem, I present a number of requirements for a more comprehensive and improved approach to email overload.

4.1 Email Overload

I will start this section with a few vital statistics¹⁴ compiled by the *International Data Corporation*¹⁵ and the *Information Worker Productivity Institute Research*. The number of person-to-person emails sent annually worldwide in 2004 was estimated at 7.8 trillion and projected to reach 10.4 trillion by 2008. The average office worker (based only on data from the USA) sends and receives up to 200 email messages a day. It has been calculated that the average information worker spends more than 90 minutes of their working time (20%) to deal with email. In an office setting, email is not just a major channel of data exchange, but is a veritable virtual working environment, serving as an indispensable digital extension of the user's physical collaborative environment. As a main conduit through which personal work and information is distributed, email is a critical site for *Personal Information Management* (PIM) [Whittaker et al., 2006]. A common tendency observed in knowledge workers is to embed PIM directly into their favorite workspaces. In fact, many desktop users have co-opted email as a PIM tool [Ducheneaut and Belotti, 2001]. In this context, email's adequacy as a PIM tool has a direct bearing on the office worker's productivity. The problem has been discussed a large number of times, most notably by Whittaker et. al., in a number of publications covering

¹⁴<http://office.microsoft.com/en-us/outlook/HA011751391033.aspx>

¹⁵<http://www.idc.com/>

over ten years of research. This research was sparked by Whittaker's feeling that despite email's popularity and widespread use, research had not addressed how people organise and manage their email information. Together with Sidner, Whittaker is credited with coining the term 'email overload' to refer to all the unintended ways in which email is used and their effects on the user's PIM [Whittaker and Sidner, 1996]. The problem is not the use of email for unintended functions per se, but the fact that both the email model as well as popular email systems are ill-equipped to support them.

Studies of email usage have documented many unintended functions for which email is used, such as its doubling as a task manager, as a document delivery system, and even as an archive and contact manager [Mackay, 1988, Bellotti et al., 2003]. Most of these additional functions are in fact PIM functions. Furthermore, given that email is interpersonal, the PIM problem is extended to one's social network; thus becoming an *Interpersonal Information Management* (iPIM) problem [Whittaker et al., 2006]. Group information management brings about new challenges, as collaborating people have different priorities and expectations about how shared information will be acted upon. Thus, failure to correctly manage iPIM and keep track of both message status and obligations, by all persons involved, may directly jeopardise one's or another person's work.

In the earlier work, Whittaker and Sidner discuss how these problems result from the breakdown of what they call the *email one-touch model*. This simplistic model assumes that on the receiving side, incoming messages fall into two categories – *informational* or as part of a *correspondence*. The former are read and then either deleted or filed, whereas the latter are answered and then either deleted or filed. Thus according to this model, email information can only be in two states, i.e. unread and filed. When investigating what leads to the breakdown of this model, Whittaker et. al. found two major problems. Firstly, although users try to process information at once, immediate reading or responding of email is not always possible/appropriate. Secondly, filing is renowned to be a cognitively difficult task [Lansdale, 1988], which does not always serve the intended purpose, since filed information is less available to remind users

about the topic [Whittaker and Sidner, 1996]. Additionally, many users find that automatic searching for an email in the inbox is faster than manually looking for it within a folder structure. However feelings of email overload increase as message volumes increase [Dabbish and Kraut, 2006], and thus having an unmanageable inbox only worsens the problem.

To counteract the two weaknesses with the one-touch model, Whittaker and Sidner proposed the ‘redesign’ of email to fit its functions. In particular this redesign was aimed at the better support of asynchronous communications, such that the user can keep track of the conversational status; and of email task management such the user can view and manage action items separately. In later research work [Whittaker et al., 2006] Whittaker et. al. discuss whether email systems should pursue centralisation to include all the required iPIM functions (e.g. calendaring functions, reminders, scheduling, etc.); or whether ways to synchronise data with dedicated iPIM applications should instead be provided. In the second scenario, iPIM *Fragmentation* is highly likely to occur as there is a lack of support for email data migration into PIM applications. As a result, useful contextual information (e.g. sender of the email) is often lost. Inversely if the effort required to export, say email tasks to an appropriate task manager, proves to be too much, with the email system not doubling as the main task manager, tasks risk being set aside and forgotten. Thus the recommended approach is to have a centralised email system coupled with better information extraction techniques to ensure the migration and synchronisation of information over additional dedicated PIM applications. Furthermore, they outline the following five recommendations for new email systems:

1. New visualisations that allow the viewing and organisation of related information in multiple related messages should be provided.
2. Techniques can be employed to detect obligations within email and their level of urgency.
3. Organisation needs to be allowed at the individual email task rather than at the

message level.

4. Implement a technology to automatically propose additional actions based on email content and ease their initiation.
5. Employ techniques to anticipate message importance and prioritise them accordingly.

Most of the iPIM issues raised by the email research community have been taken into consideration in the design of many recent email systems. To fulfil these as well as additional ideas, a number of techniques and approaches were used to support iPIM in email. These attempts will be discussed in detail in the remaining sections of this chapter. They can be categorised as going into the two following directions:

- Ensuring better retrieval of interpersonal email information from email archives.
- Ensuring that the user always remains in control of all concurrent email to-do's and be constantly able to switch conversational contexts.

The employed techniques range from search-based and information extraction techniques to more radical approaches such as the application of solutions from other domains, e.g. workflow management. Solutions aimed at the first problem are largely superficial and include improving search facilities (Section 4.2), providing chronographic visualisation of email communication (Section 4.3), and a better organisation of email messages and conversations via automatic and semi-automatic classification (Section 4.4). Solutions aimed at the second problem provide more of a challenge, as they attempt to give a better structure, and even semantics, to the underlying email model; with the aim of providing for the separate management of email action items (Section 4.5) and entire email workflows (Section 4.5.2). Many of the systems discussed in the following sections do not stick to just one approach but apply a combination of techniques.

4.2 Search-based Solutions

Email information management is only an issue because without it, email information retrieval becomes strenuous. If required information could be easily retrieved, such management would be less crucial. Thus *Search* has in the past been proposed as a solution to email iPIM [Dumais et al., 2003]. Search-based solutions vary from basic text-based indexing to ones assisted by minor organisational efforts such as tagging. A number of initiatives have explored the latter possibility, such as *Tag the Bird*¹⁶, a Thunderbird addon which performs automatic tagging of email. The tags, which are stored in the email headers, can then be searched via the use of an external search engine. A more comprehensive tag-based search solution has been employed by the Zimbra desktop¹⁷, an e-mail client with characteristics of both Web-based and conventional desktop applications. Zimbra automatically assigns tags to email, increasing the odds of the right results being returned via its advanced search which also provides for conventional search criteria e.g. folder, date, person, etc. Search queries can also be saved as virtual folders, returning dynamic results each time they are run.

The most popular system employing a search-based solution to emerge in recent years is Gmail¹⁸ – a webmail, POP3 and IMAP service provided by Google. Perhaps Gmail’s most attractive feature is its search-oriented interface, which allows users to easily find an email by adapting their search engine-style ‘googling’ to within their email archives. Gmail also provides for the manual retrieval of items from the customisable categorisation and organisation of email conversations via its ‘Labels’. Users can create labels on the go and drag and drop conversations from the inbox or from within other labels for their better organisation.

Many other initiatives have attempted to extend popular email clients with a more ad-

¹⁶<https://addons.mozilla.org/en-US/thunderbird/addon/1832>

¹⁷<http://www.zimbra.com/products/desktop.html>

¹⁸<http://mail.google.com/>

vanced index-based search facility, naming LookOut¹⁹, Lookeen²⁰ and Xobni²¹ (discussed in the next section) as examples for Microsoft Outlook. However, despite the relief provided by search-based approaches, search can at most be a partial solution to problems like task and event management [Whittaker et al., 2006]. Because it is a user-initiated process, it is not of much use when the user needs to be *reminded* about obligations, commitments, etc. To provide this kind of support, one needs to investigate methods to automatically highlight critical tasks [Horvitz et al., 2003].

4.3 Visualising Email Communication

Asynchronous communication is concerned with interaction in a permanent medium across space and time [Whittaker and Sidner, 1996]. Although email communication is asynchronous, the email model is not best suited to replace face-to-face office communication. Research carried out in [Kraut et al., 1993] characterises such communication as consisting of repeated brief interactions that are required to complete one or more tasks. Moreover, each person can be involved in multiple concurrent conversations, thus requiring to constantly switch contexts while keeping track of all running conversations [Whittaker et al., 1994]. Thus the email model is not suited to replicate these characteristics – mostly because it is message-, rather than thread-oriented. Therefore, the stringing together and display of email asynchronous conversations is left entirely to the email clients. The first of the iPIM-supportive recommendations for the design of email system in Section 4.1 refers to the need of adequate visualisations of related email information. The coming subsections will discuss existing approaches that target the better visualisation of email conversations per se as well as the underlying collaboration.

¹⁹<http://email.about.com/od/outlookaddons/gr/lookout.htm>

²⁰<http://www.lookeen.net/>

²¹<http://www.xobni.com/>

4.3.1 Visualising Email Conversations

A number of initiatives strived to provide a thread-oriented visualisation of email. *ReMail* is a ‘reinvented email prototype’ that incorporates new ways to visualize, manage and interact with threads and other groups of related messages [Rohall et al., 2004]. ReMail’s solutions are based on three constructs: showing message context, marking messages, and selective display via the introduction of the *Collection* organisational paradigm as an alternative to folders [Kerr and Wilcox, 2004] (more about this in Section 4.4.4). ReMail automatically highlights related messages when selecting an email item, allowing the user to gather them in one email thread. It also provides *thread map visualisations* (Thread Arcs) consisting of a chronologically ordered tree with nodes for messages and arcs denoting replies. In addition, *message maps* provide a visualisation of messages in a particular folder, highlighting messages in the same thread as well as ones from the same author.

Some of the systems introduced in Section 4.2 also provide some support for email conversation visualisation. In order to better organise the inbox, Zimbra allows the user to collapse email threads into one single *Conversation View*. Gmail is designed around a ‘threaded conversation view’ that is similar to that traditionally characteristic of Internet forums.

4.3.2 Visualising email collaboration

Additional ways of visualising email collaboration, rather than conversations, have been implemented in systems like Xobni (Section 4.2). Rather than visualising email conversations/threads, Xobni is contact-oriented such that; when clicking on an email, a graphical representation of the collaboration that’s been taking place between the user and the contact in question is provided. This includes the times when they have exchanged email, other people who have exchanged email with the contact, files that have been exchanged, ongoing conversations, as well as other personal information about the contact. Xobni

also ranks contacts in terms of the frequency of email collaboration.

ReMail (Section 4.3.1) also provides *correspondence maps* alongside the other visualisations. These group messages by sender and orders them by the number of messages they have sent. A similar people-based visualisation of email is provided in *Inner Circle* – an Outlook plugin that focuses on improving email collaboration [Turski et al., 2005]. Inner Circle displays ‘tiers’ of people, organised by frequency of message exchange, and displays messages related to a selected contact.

More elaborate email collaboration visualisation and organisation schemes than the ones provided in Gmail (Section 4.3.1) were investigated in Google Wave²². Described as a personal communication and collaboration tool, this web-based service was under development between 2009 and 2010. Wave’s demise was perhaps caused by an overly ambitious attempt at providing a real-time collaboration management and visualisation system that caters for a combined use of e-mail messaging, instant messaging, wikis, and social networking.

Although noteworthy, when employed on their own the solutions presented in the whole of this Section remain very superficial, as their knowledge of the complex underlying conversational structures – whereby one single email can contain a multitude of tasks requiring separate management; is severely limited.

4.4 Email Message Classification

One of the ways of ensuring the better retrieval of email messages is to have comprehensive message filing schemes. Apart from the possibility of creating folder structures, the only means by which email filing is supported in conventional email clients is via the definition of *Email Rules* that automatically sort and organise incoming email into the existing folders. However these rules are very superficial, operating on the textual matching of the sender address or text in the message subject and/or content. Otherwise, message organisation is largely manual. Before filing an email away, users are required to think

²²<https://wave.google.com/wave/>

where they would look for that particular item in advance. The average user thus finds standard foldering techniques too cognitively taxing and time consuming. Although they put a huge effort in creating and maintaining email folders, they still face difficulties when retrieving messages and to keep the folder structure alive. This approach is so unsuccessful that few people tend to file messages right away. The majority are discouraged from doing any filing, resulting in a highly unmanageable inbox.

One solution to this problem is that of assisted filing, whereby machine learning techniques are employed to make recommendations to users about where best to file an incoming email message. Users are more receptive towards assisted, rather than automatic filing, with most users being inherently distrustful of automatic classification schemes [Whittaker and Sidner, 1996]. The major problem with automatic filing is that users may never become aware of automatically filed messages. In this section I will provide an overview of systems that attempt to reduce email overload by supporting the user with organising their email, mostly by way of some form of email classification/filtering technique. The majority of techniques in question draw from classic text classification solutions, and can be broadly categorised into *rule-based*, *information retrieval-based* and *machine learning-based*. These, as well as additional miscellaneous techniques, will be covered in the following subsections.

The majority of these techniques assume an existing folder structure, but some go a step further to automatically generate an enhanced folder structure, or even entirely different organisational paradigms.

4.4.1 Rule-based Classification Techniques

Email Rules are a very primitive form of rule-based techniques, relying solely on the strict definition specified by the user. Apart from lacking flexibility, rules also require maintenance. As the number and characteristics of incoming mails change, the rules in the rule set may have to be modified to reflect the same [Aery and Chakravarthy, 2005]. Besides the need to create new rules over time, specified rules may required updates, or

cease to be relevant completely.

Some have attempted to provide more flexible rule-based classification systems. Cohen introduced a method [Cohen, 1996] for learning sets of “keyword-spotting rules” based on the RIPPER rule learning algorithm [Cohen, 1995]. Evaluation of this system suggested that learned keyword-spotting rules combined with user-defined rules are a viable architecture for a personalised email filtering system.

Iems [Mccreath and Kay, 2003] is an “Electronic Mail Sorter” which learns rules based on sender information and content keywords, and suggests a destination folder accordingly. The authors found that there are better results when user-adjusted learnt rules are used in combination with hand-crafted rules. However, the authors also warned that the classification scheme predicted by the system directly effects the users’ manual classification.

4.4.2 Information Retrieval-based Classification Techniques

Information Retrieval (IR) techniques commonly consider a set of predefined keywords as features. Feature extraction is the act of looking for these keywords, in an email message in this case. IR-based classification was employed in *SwiftFile* (formerly MailCat), an intelligent personal email assistant implemented as an add-on to Lotus Notes²³. It reduced the cognitive burden and the time required for the organisation of e-mail into folders [Segal and Kephart, 1999, Segal and Kephart, 2000]. SwiftFile bootstraps a TF-IDF [Baeza-Yates and Ribeiro-Neto, 1993] text-based classifier over email messages already stored in the existing folder structure. TF-IDF is a text mining-based approach that is frequently used in IR. In this case, TF-IDF weighs how relevant a keyword is to an email message; by factoring the frequency of the keyword in the email against the frequency in the whole corpus. SwiftFile employs TF-IDF to predict the three most likely destination folders wherein an incoming email message should be stored. Suggesting three folders raised the observed classification accuracy from an accuracy rate of between 60% and

²³<http://www.lotusnotes.com/>

80%, to a more acceptable range of 80% to 98%. This system only works for existing folder structures, and it cannot identify and propose or create additional folders automatically. However the classifier is not static and supports incremental learning such that it takes into account a changing folder structure.

4.4.3 Machine Learning-based Classification Techniques

Rule-based classification systems learn to generate rules for the classification of email based on keywords. In contrast, systems based on Machine Learning (ML) perform iterative learning via the extraction of a whole set of features from empirical training sets. Feature extraction is handled by specific ML algorithms. Among the most successful of these algorithms are Naïve-Bayes classifiers [Lewis, 1998], which use a simple probabilistic model coupled with training sets to perform classification. These classifiers have successfully been used against email spamming to filter junk email [Sahami, 1998]. They have also been employed by a number of ML-based email classification systems, such as the iFile e-mail filtering system [Rennie, 2000]. iFile uses the Naïve-Bayes approach to perform training, build a classification model and perform iterative learning. The observed classification accuracy was of between 86% and 91%, a range that is likely to be acceptable for a large number of e-mail users. Dredze et. al. developed an unsupervised learning framework for selecting summary keywords from e-mails using latent representations of the underlying topics in a users mailbox [Dredze et al., 2008b]. The resulting keywords are then use for two iPIM-supportive automated foldering and recipient prediction. In another approach, Turenne presented a clustering algorithm to learn semantic classes for the automated construction of filters to classify email messages [Turenne, 2003]. The algorithm extracts graph patterns of terms in the training corpus and generates user profiles containing the user's areas of interest (folders) each with an associated term set learned via a clustering technique. An incoming email is then routed to an applicable folder given the distance between the terms it contains and the term clusters associated with that folder. An evaluation of the system gave a satisfactory 94% precision and 92%

recall.

The bottleneck of ML is the requirement of having a suitable amount of data for training. *Co-training* was introduced by Blum and Mitchell [Blum and Mitchell, 1998] as a technique which greatly decreases the effort required in applying ML on real-life data, by allowing classifiers to learn with fewer classified examples while taking advantage of the more abundant unclassified documents. The authors have demonstrated how the technique helps reduce the error rates of classifiers. A number of approaches have applied this technique to email classification [Kiritchenko and Matwin, 2002, Chan et al., 2004]. Kiritchenko and Matwin have shown that the co-training approach can be successfully applied for email classification, although the results varied depending on the learning algorithm used. In this case, the use of Support Vector Machines (SVM) [Cortes and Vapnik, 1995] performed better than Naïve-Bayes. One requirement of co-training is for the underlying dataset to be described by two disjoint sets of redundantly sufficient natural features, such that a reasonably accurate classifier can be built using just one of the feature sets. In many practical scenarios, no two such feature sets are available to describe the dataset. Chan et. al. investigated whether this technique is thus really viable when applied to email, and concluded that co-training is only beneficial when only a single natural feature set is used, opting instead for a random split of these feature set.

Some approaches have combined ML with other approaches. The *Re:Agent* intelligent email agent [Boone, 1998] combines an IR and ML approach. The TF-IDF classifier is used to extract useful features from email messages to create a feature vector on which an ML algorithm can then operate. The system thus first learns concepts present within the email via IR, and then uses them as features to learn actions that can be performed on messages via ML. Action prediction is based on a neural network. The learned actions in Re:Agent are not limited to automatic message filtering, but include additional email actions such as prioritising and forwarding.

Other approaches have combined ML with rule-based approaches. An example is the *Magi* Mail Agent Interface [Payne, 1994], which employs machine learning techniques to

discover rules for filtering email. Magi uses the CN2 rule induction system [Clark, 1996] to generate a user-profile from observations of user interactions with e-mail. ML techniques are used to help automate the classification task and move messages into different folders.

4.4.4 Miscellaneous Classification Techniques and Approaches

Most of the approaches described so far have applied text classification techniques to email. However, pure text-classification methods might not be well-suited for this taskclassification, as pointed out by Aery et. al. in their motivation for their *EmailSift* mail classification system [Aery and Chakravarthy, 2005]. Firstly, personal preferences in email management vary greatly, and there is no ‘one-size-fits-all’ solution. Secondly, the email environment is very volatile, thus classification systems require adaptive and incremental re-training. Thirdly, email messages may not be as rich as other textual documents, thus relying solely on text from their context is not very appropriate. Thus, providing a solution for email classification may require looking beyond conventional text classification techniques. The approach pursued in EmailSift employs an adaptation of graph-based mining techniques to mine structural data, in addition to email content; in order to classify email into a corresponding folder structure. Representative common and recurring structures are extracted from a pre-classified email folder, which are then used to classify incoming e-mail. The resulting classification system performed better than text classification systems such as Naïve-Bayes.

All of the approaches above considered classical email folder structures. This is however not the only organisational system explored, especially due to its known limitations [Pazzani, 2000]. Whereas systems such as Gmail (Section 4.3) have introduced improved versions of the foldering system, others provide email organisation via tagging (Tag the Bird and Zimbra, Section 4.2). Additionally, some other approaches sought alternative organisational paradigms, like ReMail’s (Section 4.3) *Collections* - containers of pointers to related messages. Another example is the *CEM* email management system [Cole and Stumme, 2000], which stores its email in a *concept lattice* based on formal

mathematical structures, rather than in the usual tree structure. Catchwords (natural language phrases) are used to classify documents into the resulting conceptual multi-hierarchy. The authors of CEM claim that the lattice-based view of e-mail provides more flexibility in retrieving email items while simultaneously aiding the process of knowledge discovery in large email collections.

Another approach looked beyond storing messages in organised collections, and focused instead on organising email according to the user's activities. Dredze et. al. [Dredze et al., 2006] pursued an activity management approach to effectively monitor individual activities rather than messages. A classification task is performed for the purpose via two algorithms which find similarities between message content and the ongoing activities and compare a message's contacts to the activities' participants. Their evaluation concludes that the use of the algorithms perform better (0.81 F-measure) than a 'baseline method' that uses a Naïve-Bayes classifier and information pertaining to message reply-to threads to determine activity membership (0.60). In later work, the authors evolved their methods to beyond classifying whole emails, focusing instead on classifying *individual* email activities/tasks. Classification approaches such as this one are discussed in the following section.

4.5 Structuring Email Processes

Dredze et. al. report that attempting to classify entire email messages into activities is not a straightforward task [Dredze et al., 2006]. Yet, one of the problems undermining this approach could very well be the fact that email messages tend to contain information related to more than one activity. The very flexibility of the email model allows multiple topics to be simultaneously discussed in the same message. Furthermore, even discussing just one topic can result in multiple e-mail activities, or *E-mail Action Items*. For example, in an email related to a forthcoming event, questions might be raised about who will be attending, tasks related to preparing for the event may be assigned, etc.

One way of tackling the email overload problem would thus be to provide individual

support for these activities, over and above any support provided for the management of email message threads. This is very much in line with the third of the five iPIM-supportive recommendations for email systems as provided in Section 4.1 – “organisation needs to be allowed at the individual email task rather than at the message level”. Thus a number of approaches have strived to identify and place patterns of email communication into a structured form. An overview of these approaches is provided in the following subsections

4.5.1 Task-centric views of Email

Many of the above approaches have acknowledged the importance of individually supporting activities executing *within* email. Gmail for example (Section 4.2) has developed Gmail Tasks²⁴ as an enhancement and, although the technology is still evolving, it already supports the conversion of email messages into tasks for separate management. However, this is performed at the message, not at the task level. Google Wave (Section 4.3.1) supported the organisation of specific tasks rather than the entire message or conversation.

Some other approaches have sought to transform e-mail into a full-blown Task Manager. One must point out that the term ‘tasks’ as used in this sense, could also refer to the management of appointments. The creators of the *TaskMaster* email client argue that life in the email habitat should be rethought not in terms of messaging, but in terms of the various activities users are trying to accomplish within [Bellotti et al., 2003]. In their approach, Bellotti et. al. attempt to recast email as task management and to embed task-centric resources directly into the email client [Bellotti et al., 2002]. Aside from tasks in individual messages, TaskMaster also supports semi-automatic collections of *Thrasks* – threads of messages containing interdependent tasks. An evaluation of TaskMaster reveals that embedding task management features directly in the inbox, had a significant positive effect on the overloaded email user. A similar approach was taken by Gwizdka

²⁴<http://gmailblog.blogspot.com/2008/12/new-in-labs-tasks.html>

in his *TaskView* task-based email interface [Gwizdka, 2002], which aims at supporting the management of pending email tasks. The UI was compared to that for Microsoft Outlook’s inbox with regards to the effectiveness of retrieving email information. Gwizdka observed that whereas the former was more efficient in finding information related to task dates, times and overviews, the latter was faster for finding information from subject lines, senders or content. Therefore a combined approach was favoured.

Although the above approaches allow for activity-centric views of email, activity support and visualisation remains at the message- or thread-level. A more effective approach is to instead focus on the identification and separate management of the individual activities within. In this regard, Khoussainov and Kushmerick presented an approach targeting the high-level support for email activity management [Khoussainov and Kushmerick, 2005], whereby email activities *within* an email message can be identified and managed as separate entities. They employ ML techniques to try and identify email tasks distributed over multiple messages and the relations in between these messages. Their method is also aimed at semantic message analysis, whereby metadata about how a message within one email task relates to the overall task progress. Their approach, like a significant number of others, attempts to classify email content into *Speech Acts* – communicative acts with clear but frequently implicit semantics. Speech Act-based research efforts dealing with the elicitation of specific patterns of communication will be discussed more thoroughly in Section 5.2. In this particular approach the elicitation of speech act information helps find related messages and to group them into tasks. In later work, the authors join forces with Dredze et. al. [Dredze et al., 2006] to adopt the activity classification technique presented by the latter in (Section 4.4.4). As examples of how their technology can benefit the use of email, they mention functions such as reply prediction, detection of completed activities, message prioritisation based on the activity status, identifying dependencies, version control over attached data, and automated. They also addressed email activity management from an interface perspective, designing ‘intelligent’ UIs to provide

activity-centric rather than message-centric tools.

What many in this Section have considered as singular e-mail tasks, can be considered as just another step in an evolving *Collaborative Process*. These kind of processes are not discrete but iterative and thus in the context of email, to get the full picture users are required to combine task-related information in incoming messages with prior relevant information [Whittaker, 2005]. Thus the management of individual email activities is rather futile unless they are taken in the context of the entire process of which they form just a part.

4.5.2 Email Workflow and Business Process Modeling

The succession of e-mail tasks in an evolving e-mail-based collaborative process can be interpreted as a sequence of *operations* in an *E-mail Workflow*. The use of this term in this sense is to be differed from the one used in approaches that consider superficial e-mail workflows, such as the approach taken in [Venolia et al., 2001]. In that context e-mail workflow referred to the e-mail management tasks the e-mail user is expected to carry out when receiving e-mail; such as deciding whether to save and file a message, delete it, reply or leave it in the inbox. In this section, I introduce some attempts striving to elicit and structure email's *underlying* workflows. In fact, e-mail has been considered as not just a task management system but as a complete workflow management system that defines and supports the user with a series of tasks to produce a final outcome. Here, business process modelling has been sought as a means of modelling the workflows as a structured sequence of related activities or tasks. Driven by Van der Aalst, the *Workflow Patterns*²⁵ initiative was established with the aim of delineating the fundamental requirements that arise during business process modelling on a recurring basis. The main contribution of this work is a set of 40 patterns describing the control-flow perspective of workflow systems.

Van der Aalst et. al. also surveyed a number of workflow mining techniques

²⁵<http://www.workflowpatterns.com/>

[van der Aalst et al., 2003]. Similar techniques have been tailored for e-mail workflows and applied accordingly. Kushmerick and Lau [Kushmerick and Lau, 2005] (introduced earlier) acknowledge that many email messages are a manifestation of a user’s participation in a structured process, or workflow. They formalise these processes as finite-state automata, where the state corresponds to the status of the process, and the transitions represent messages exchanged between participants. ML algorithms are employed to automatically generate the process model, so as to perform automated activity management of e-mail workflows. Their technology was implemented in *Nectar*, an email activity assistant²⁶ provided as a Thunderbird extension that leverages email structure to organise messages by workflows.

Some other approaches are not exclusive to email and offer a more comprehensive workflow mining approach. Shen et. al. [Shen et al., 2009] investigate ways to discover work procedures from user desktop activities, including e-mail exchange. They present graph mining algorithms to detect repeated sequences of actions by observing user desktop activities. The algorithms are executed on top of an *information flow graph* based on information captured by their *TaskTracer*, which tracks resource movement and provenance on the desktop. Files (including e-mail messages) constitute nodes of the constructed graphs whereas information flow actions (including e-mail exchange) constitute their edges. The system’s evaluation shows that the approach can accurately discover many known workflow instances while introducing few false instances.

A major limitation affecting these approaches was pointed out by Whittaker in [Whittaker, 2005]. Workflow systems assume that collaborative tasks have a structure which is to an extent or another predictable. The lack of predictability is not entirely a bad thing, as it directly results from the e-mail model being so flexible and thus allowing for spontaneity, which in itself is an advantage of the email model. However, most collaborative email tasks lack the required predictability as most have an evolving structure requiring iterative negotiation to complete. This makes email very unpractical

²⁶<http://www.kushmerick.org/nick/research/ea/>

as a workflow tool. In order for e-mail to be effectively used as a workflow management system, such an approach must be tailored for email's less predictable nature. E-mail based workflow systems need to support *Ad-hoc Workflows*, which allow for a better management of the less-structured collaborative e-mail processes than via traditional, rigid workflow systems. Support for ad-hoc processes allows for the required flexibility in day-to-day, business processes. General (i.e not specific to email) ad-hoc collaborative process management has been targeted by a number of initiatives.

Stoitsev et. al. [Stoitsev et al., 2008] target the user-driven business process modeling via the bridging of ad-hoc and formal workflow models. Weakly-structured process models are generated by web services capturing personal task management data. These models can then either be reused for ad-hoc process support or exported and integrated as formal workflows by software developers. The interconnection of ad-hoc and formal workflows results in enhanced process flexibility and allows for deviations from the otherwise rigid formal workflows, via on-demand, ad-hoc task hierarchies. An evaluation of their *Collaborative Task Management* prototype has shown that apart from enabling user-driven process model refinement, deviations effectively reduce the cognitive distance between collaborative work tasks and the workflow modeling task. This approach is relevant also when taken solely in the context of e-mail workflow modelling. However, it remains somewhat limited, as ad-hoc moves are treated as *deviations*, implying that a shared execution of ad-hoc workflows between the participants is not really supported unless formalised as non-ad-hoc workflows.

Dustdar et. al. [Dustdar, 2004] present conceptual, design and implementation issues for ad-hoc collaborative work management systems. As a proof of concept for their work, they provide Caramba, a system that supports virtual teams with the ad-hoc collaborative processes. Caramba enables links between workflow artifacts (e.g. resultant documents), the business processes themselves, and additional resources. To support Caramba, Dustdar et. al. also proposed their own techniques for ad-hoc business process [Dustdar et al., 2005] and interaction pattern [Dustdar and Hoffmann, 2007] mining.

In order for Caramba to be able to support ad-hoc processes, the underlying workflow modelling is tailored towards ‘runtime’, rather than pre-established collaboration control flow. Thus, the control of business processes supported by Caramba is not restricted by the system and remains in the users’ hands. As discussed by Voorhoeve et. al. [Voorhoeve and van der Aalst, 1997], this added flexibility can be considered a problem, as ad-hoc workflow support systems require the users to participate in the selection and modification of a process on a case by case basis. However, this is not a problem in the context of e-mail business process modelling, where the user is anyway expected to drive a workflow, rather than strictly adhere to a predefined business process. Therefore, in trying to structure and support conventional email-based collaboration, the added flexibility as provided by systems like Caramba is not only desirable, but constitutes a requirement.

4.5.3 Semantic Email

Studies on lexical densities of email discourse showed that despite being a written form of communication, email texts are closer to spoken rather than written discourse [Khosravi and Wilks, 1999]. In fact email has been regarded as a new computer-mediated communication genre [Goldstein and Sabin, 2006], where a genre is a “patterning of communication which structures communication by creating shared expectations about the form and content of the interaction, thus easing the burden of production and interpretation” [Erickson, 2000]. This *shared understanding* is also the backdrop that makes any collaboration possible, as explained in Clark and Brennan’s *Theory of Common Ground*, which elaborates the underlying processes involved in collaboration [Clark and Brennan, 1991]. Common ground entails both the shared understanding and a protocol for maintaining that understanding. Whittaker et. al. report [Whittaker et al., 2005] how this theory was employed to examine the interplay of communication and shared structures to organise synchronous collaboration. They also discuss how common ground can also be supported by external representations, as persistent, integrative, and shared means of constraining communication so as to

structure and co-ordinate activities.

A shared understanding of the form and content of e-mail interactions can be achieved by sharing elicited email communication structures. However, a lot of useful semantics pertaining to these structures is lost in the communication process itself. Thus it would be handy if the email model could support the transport of such semantics, in order for elicited patterns of communication to be more reliable and offer their shared interpretations. The use of some well-defined language to enhance the structure and purpose of e-mail is not a novel idea. *Computational Email* was a concept introduced by Borsentein [Borenstein, 1992] back in 1992, and had at its heart the idea of sending an executable e-mail rather than plain text or multimedia files. Enhanced e-mail clients then executed the computational e-mail rather than presenting it for the user to read.

The more interesting, and more recent, concept of *Semantic Email* was originally introduced by McDowell et. al. [Mcdowell et al., 2004] as an attempt to place email communication within well-defined structures whose semantics can be interpreted in the same way by all communicating parties. It is to be differed from the concept of *Semantic Email Addressing* as introduced in [Kassoff et al., 2009], which targets the well-defined, semantic definition of e-mail recipient groups. In contrast, Semantic Email refers to an email message consisting of a structured query (or an update to the query) coupled with a corresponding explanatory text. McDowell's approach is based on the provision of a broad class of *Semantic Email Processes* that represent commonly occurring workflows within email (e.g. collecting RSVPs, coordinating group meetings) [Mcdowell et al., 2004]. Implemented within Mangrove [McDowell et al., 2003] the system provided templates which exposed structured knowledge about these scenarios to both humans and machines. The ultimate goal was to support the user with common email related tasks such as collecting information from a group of people, handling schedule/event information and reminding others about previous unanswered emails etc.

Giving the email process a semantics via a formal standard enables machines to guide and support the user with the communicative process, automate the flow between trivial

states of the communication, and in general participate actively in the communication alongside the user. The problem with McDowell's approach, is again the lack of flexibility afforded for the email collaboration. In fact, email processes need to be pre-meditated and follow fixed templates (e.g. a template for meeting scheduling). Additionally, the system can only support one process per email, which again was a major limitation of earlier-presented approaches.

4.6 Advancing the State-of-the-art

In this chapter I have presented the most relevant attempts at reducing the effect of e-mail overload and providing support for iPIM in e-mail. In attempting to enable the desktop to effectively support the end-user with their email-based collaborations, I will go on to establish the following requirements:

1. Multiple email action items, or tasks, should be managed independently of the email messages and threads within which they are executing.
2. Email action items should be considered within the context of the e-mail process, or workflow, of which they form part.
3. It is possible to elicit communicative patterns to create well-defined and frequently occurring e-mail workflows.
4. Due to the email's advantageous flexibility, such e-mail workflows need to be less rigid and allow for additional and spontaneous courses of actions.
5. Such spontaneous courses of actions should not be considered as deviances, but taken into account in the very fabric of the resulting in ad-hoc workflows.
6. As email collaboration relies on a shared understanding, semantics pertaining to the structure of ad-hoc workflows need to be transported alongside email.

Given these requirements are taken into account, an ensuing implementation will be able to support the user with better email interpersonal information management. A system that is aware of machine-processable ad-hoc e-mail workflows will not only be able to enhance the organisation, visualisation and retrieval of e-mail data, but also provide additional intelligent functions such as task reminders, detection of required attachments and assisted e-mail task handling.

5 Pragmatics and Electronic Communication

In their attempt to elicit individual action items from within general electronic forms of communication, many researchers have relied on computational linguistics. In particular, many have sought to ground their techniques within the philosophy of language and more specifically to theories and research initiatives such as Speech Act Theory, the ensuing Language-Action Perspective, and various spin-off research areas such as Dialog-Act Modeling. From an information systems point of view, these research contributions are now considered to be part of the arising *Pragmatic Web* – a vision for a Web that takes into account the context, or the *purpose*, of information; rather than solely the semantics [de Moor et al., 2002]. As such, the Pragmatic Web can be considered as a likely extension of the Semantic Web, with a focus on shared interpretation rather than shared meaning.

In this chapter I will cover the origins of the Pragmatic Web, and discuss how its founding ideas have been employed to better make sense and structure electronic communications.

5.1 Origins of the Pragmatic Web

Whereas syntax is concerned with the grammatical structures used to convey information and semantics with the meaning of that information (Section 2.2.3), pragmatics are concerned with the actual purpose of information. Thus pragmatics are an important aspect of information exchanged over a collaborative scenario, as they are what individuals concerned perceive as commitments and expectations.

As a school of thought, the Pragmatic Web is considered to be still in its infancy. Its origins lie within John L. Austin's doctrine of locutionary, perlocutionary and illocutionary acts as introduced in "*How to do things with words*" [Austin, 1962]. Here Austin argued that truth-evaluable sentences form only a small part of the range of utterances. One form of utterances which does not belong in that category are *Performatives*, whereby the very act of their utterance constitutes an action. Austin goes on to differ between *locution* – the very act of saying something and the novel concepts of *illocution* and *perlocution* – respectively the act performed *in* uttering something, and the act performed *by* an utterance. An illocutionary act is the use of a locution with a certain force, e.g. asking questions, making assertions, giving orders, etc. To perform an illocutionary act, the act being carried out must be made clear to the hearer such as to ensure the production of conventional consequences, such as commitments and obligations. The ensuing consequences, or effects of an illocutionary act on the hearer, are in themselves perlocutionary effects. Thus, eliciting an answer from a question (an illocutionary act) is the perlocutionary effect resulting from it. Perlocutions are external to the performance of an utterance, and usually entail getting someone to do or realise something.

In later work, John R. Searle focused particularly on illocutionary acts, also referred to as *Speech Acts* by Austin himself. In his book "*Speech Acts: An Essay in the Philosophy of Language*" [Searle, 1969], he discusses at length the distinction between the propositional content of an utterance and the illocutionary force of a speech act (i.e. the intended purpose). To substantiate his ideas, he provides a number of examples, e.g., whereas the propositional content of "Sam smokes habitually" and "Does Sam smoke habitually" is identical, the illocutionary force of the former is an assertive whereas the latter is a question. In later work [Searle, 1975b], Searle categorises illocutionary acts as follows:

- *Representatives/Assertives* commit the speaker to the truth of the expressed proposition (include statements and opinions)
- *Directives* cause the hearer to take a particular action (includes requests, orders

and suggestions)

- *Commissives* commit the speaker to some future action (includes promises and self commitments)
- *Expressives* express attitudes towards a proposition (includes excuses and thanking)
- *Declaratives* change the reality in accord with a proposition (includes events like verdicts, resignations, marriage pronouncements)

Searle’s most important contribution, especially in the context of the Pragmatic Web, is his theory of *Indirect Speech Acts* [Searle, 1975a] whereby he introduces *context* to the notion of speech acts, such that it determines the real purpose or illocutionary act. He describes indirect speech acts as those where “the speaker communicates to the hearer more than he actually says by way of relying on their mutually shared background information, both linguistic and non-linguistic, together with the general powers of rationality and inference on the part of the hearer”. Searle goes on to distinguish between a primary (indirect) and a secondary (direct) illocutionary act for speech acts. Given this distinction, the primary illocutionary act is the only one performed when uttering an indirect speech act.

By the early 1980s, Austin’s and Searle’s contributions had already become a very influential topic in computer science, particularly in the design of artificial languages for agent-based communication. Eventually this saw the rise of the *Language/Action Perspective* (LAP), first introduced by Flores et. al. in 1980 [Flores and Ludlow, 1980], and which considers language not only as a means for exchanging information but also as a way of performing actual actions. However, LAP is mostly based on Terry Winograd’s notion that in designing information systems, expert behaviour requires an exquisite sensitivity to context, which sensitivity is more in the realm of the human than in that of the artificial [Winograd, 1986]. Winograd provides an insight on the pragmatics of language action and it’s role in evoking and interpreting actions. LAP also draws significantly from Jürgen Habermas’s *Theory of Communicative Action* [Habermas, 1984]. From the LAP

perspective communication is primarily action which, in turn, facilitates coordination and interaction [Ljungberg and Holm, 1995]. Thus, the design of effective information systems should explore linguistic communication as a basis for the design of information systems.

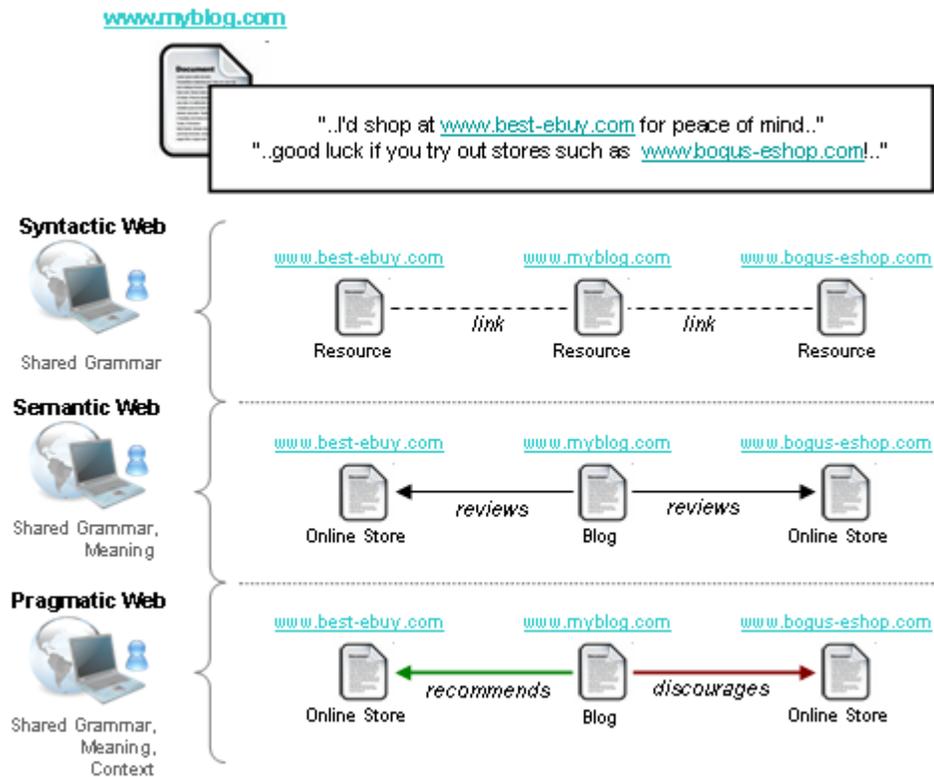


Figure 5.1: From shared grammar on the Syntactic Web, towards shared meaning and socially-defined context on the Pragmatic Web

Enhanced with pragmatics, the Semantic Web becomes context-aware and more oriented towards social interaction. This is illustrated in Fig. 5.1, which depicts how within the Pragmatic Web, the actual intended purpose can be pragmatically imparted given knowledge of the socially-defined context. This is much aligned to the envisaged progression of a Semantic Desktop into a Social Semantic Desktop as represented back in Fig. 3.2.

5.2 Supporting Electronic Communication

The value of speech act theory was first investigated within the context of spoken conversation. However, these ideas were quick to be applied to computer-based communication, where most of the dialogue is subject to digitalisation. In this more controlled scenario, machines could be enabled to operate on speech acts with clearly defined semantics once they are elicited from the information exchanged in the undergoing communication. In fact many have attempted to define appropriate speech act models with varying semantic constraints to enable agents to perform automated communication-supportive features such as consistency checking, negotiation support and status resolution.

Winograd introduced LAP as a means to assist with human-computer interaction in electronic conversations [Winograd, 1986], whereby speech act knowledge can be used to structure entire conversations. Based on his LAP model he categorised a number of conversations types by the underlying intention. He explains how “there is no sharp line between them, but they are accompanied by different moods”. The most interesting categories of purposeful conversations are:

1. *Conversations for Orientation* – where the mood is that of “creating a shared background as a basis for future interpretation of conversations”.
2. *Conversations for Action* - where the participants initiate and negotiate actions to achieve an appropriate result.
3. *Conversations for Possibility* – where the mood is that of “speculation, anticipating the subsequent generation of conversations for action”.
4. *Conversations for Clarification* – where the participants “cope with or anticipate breakdowns concerning interpretations of the conditions of satisfaction”.

The focus of LAP is on conversations for action, dubbed by Winograd as the central coordinating structure for human organisation. Conversations for action are made up of

a network of speech acts (specifically *requestives* and *commisives*) that are directed to explicit coordinated action.

Winograd demonstrated how machines can support users with the management of coordinated work by considering conversations for action via the *Coordinator* system. In the process Winograd’s research generated intensive discussions about LAP’s suitability for the given purpose. An overview of these ideas, criticism and ensuing discussions is provided in the first sub-section below. The second sub-section will present an overview of techniques that focus specifically on how speech act theory has been applied to classify electronic forms of communication. The following subsection will then provide an overview and comparison of the speech act categorisation systems used in these techniques. These categorisation systems have been considered as the basic units of knowledge for structuring written forms of digital discourse, as discussed in the final Section.

5.2.1 The Coordinator

Based on Flores et. al’s observation [Flores and Ludlow, 1980] that speech acts are not unrelated events, but form part of entire conversation structures, Winograd presented a speech act-based approach to structure entire conversations, implemented within the Coordinator – a conversational system that provides facilities for managing “records of moves in conversations”. He outlines the difference to email systems, which consider individual messages and information as their starting points, rather than entire conversations.

As such, the style and features of the Coordinator were unlike any other system in use at the time, and although it served the purpose of demonstrating his LAP-based conversation theory, it also attracted a lot of criticism. Bowers [Bowers, 1992] describes the Coordinator as a system in the area of computer-mediated communication that relies too heavily on formal representations, which can be a problem if rigidly enforced on the users. Based on Bower’s criticism, Lucy Suchman particularly challenged the validity of LAP as a basis for coordination support in computer systems in a number of publications [Suchman, 1993, Suchman, 1994]. Her main concern was that, applied in

the style of the Coordinator, the explicitness of the theory imposes an a priori structure on the undergoing conversations, leading to an undue discipline on the users and thus limiting the potential of electronic communication in supporting ubiquitous interaction [Curtis, 1995]. Subsequently Winograd responded to these concerns [Winograd, 1994], arguing that the explicit speech act theory enforces a necessary uniformity in online communications, where “ambiguity and vagueness cannot be routinely resolved through direct personal contact and knowledge”, and that therefore such shared structuring is a precondition for effective cooperation over online communication. Bill Curtis argues [Curtis, 1995] that Winograd’s rationale behind the defined speech act categories is biased towards computability rather than efficient communication patterns.

Although many consider the Coordinator to have failed as an online communication system, many others have acknowledged the suitability of the underlying LAP approach as a theoretical foundation for the design of information technology. In [Ljungberg and Holm, 1995], the authors state that importing speech act theory from an other discipline to use it as a basis in active design can be useful, but only if the theory’s limitations are kept in mind. In surveying criticism levelled at LAP, they attempt to identify where speech act-based methods breakdown in this context. They point out that further adaptations, such as the establishment of alternative classification criteria and a new focus on organisational (rather than personal) commitments, are required.

A number of important lessons were learnt from the Coordinator. Most importantly that a system designed with LAP principles in mind must be less rigid, such that effective communication is not compromised for the sake of computability. Also, in order not to limit take-up and provoke instant scepticism it’s also more appropriate to target existing communication media and systems, rather than attempt to design something entirely new. In his survey of communication support systems that capitalise on speech act components [Te’eni, 2006], Dov Te’Eni points out that the key to success of such systems will be “the clever organisation of extant communication to support future communication”.

5.2.2 Applications to Digital Communication Systems

The application of LAP to support with the coordination of computer collaborative work focused mainly, although not exclusively, on email. Some of the approaches attempting to structure email processes already described in Section 4.5.1 pursued an activity-centric view of email based on speech act theory. The approach in [Khoussainov and Kushmerick, 2005] in fact considered speech acts as the basic units of knowledge for structuring email such that activities within can be managed separately. A similar approach was presented in [Corston-Oliver et al., 2004], via the SmartMail system, which automatically identifies specific action items in email by consulting an ML-based classifier. The results are used by the system to present task-focused summaries of email, message prioritisation and to perform actions appropriate to the speech acts (e.g. add tasks to a task list). In [Khosravi and Wilks, 1999] the authors attempt to route email based on the automatic extraction of pragmatic content. The extraction was based on a phrase matching approach operating on the empirical collection of speech act patterns from email corpora. Through their implemented system – Pyam, they conclude that given their knowledge-based approach most of the considered speech acts can be successfully recognised, whether expressed directly or indirectly. Another approach [Lampert et al., 2006] presents another speech acts-based method that indicates the dialogic function of email utterances. A statistical classifier for the automatic identification of speech act categories in email messages achieved an accuracy of up to 79.8%.

Innovating his earlier rule-based email classification approach as described in Section 4.4.1, Cohen et. al. [Cohen et al., 2004] propose the use of ML to classify email into one speech act. They describe how their proposed system, as opposed to systems like Winograd’s Coordinator, passively observes email and automatically classifies it by intention. This approach retains the “flexibility and socially desirable aspects of informal, natural language communication”. In an evaluation of their classifier they conclude that many categories of messages can be detected using existing text-classification learning meth-

ods. In a collaboration with Carvalho, Cohen [Carvalho and Cohen, 2006] introduced Ciranda – a toolkit for email speech act prediction, based on a combination of n-gram sequence features that is found to be highly effective for the desired classification task. This approach reduced the error rate in their earlier work by an average 26.4%.

In [Goldstein and Sabin, 2006] email categorisation was attempted, based on the belief that email as a genre is more akin to spoken communication. Although the authors acknowledge that an email is a sequence of one or more utterances, or speech acts, they attempt to characterise an email by the most important act and its genre. This, they argue, enables the categorisation of email in terms of the intended action of the sender and expected action on the part of the recipient.

The techniques employed by Goldstein and Sabin are based on methods used for dialog analysis. Thus, their technology can in theory be applied to other forms of electronic communication. The reversal of this argument is also true – speech act-based approaches focusing on other forms of electronic communication are still relevant to this dissertation. Twitchell and Nunamaker [Twitchell and Jr., 2004] for example, present a method for visualising electronic conversations by creating *Speech Act Profiles* for participants. Visualised as radar graphs, the profiles provide an overview of a conversation in terms of the underlying intentions, and insights into the roles of the individual participants. Their approach employs a hidden Markov model (HMM) to obtain different probabilities for speech acts underlying a sentence. The profiles are created by aggregating the probabilities for each participant. In a slightly different approach, Feng et al. [Feng et al., 2006] consider the use of the term speech act to refer to a relationship between pairs of email messages, such that a speech act represents negative, positive or neutral responses to a previous message. Speech acts were here considered to perform conversation analysis in determining the focus of threaded discussions.

5.2.3 Speech act-based Systems of Categorisation

In her criticism of the Coordinator [Suchman, 1993], Suchman also pointed out the difficulty of setting up appropriate systems of categorisation. This is a major issue faced by all approaches striving to categorise electronic dialogue into speech acts. I will now provide an overview and comparison of the most relevant categorisation systems employed, many of which derive from Searle’s original speech act categories as defined in [Searle, 1975b]. However, not all of these categories were deemed relevant in the context of the given problem, with researchers choosing to focus mostly on *Directives*, *Assertives* and *Commissives*. An example is the *Conversational Roles* (COR) model – a generic, application independent model of human-computer information-seeking dialogue [Sitter and Stein, 1992].

Additionally, the incidence of misclassified speech acts is bound to increase proportionally to the amount of speech act classes or instances. Approaches such as [Khosravi and Wilks, 1999] have kept the speech act categories to a minimum, focusing solely on three types of *Requests* (for Information, for Action and for Permission) and an *Inform* speech act category. [Feng et al., 2006] propose a slightly more elaborate taxonomy, adding an *Interpersonal* category to the two provided by Khosravi et. al. Examples of speech acts in this category are acknowledgments, suggestions and objections. [Jose, 1987] addressed the problem of coherence in conversations by proposing a number of speech act-based models of communicative interaction.

A more comprehensive approach was pursued by Cohen et. al. [Cohen et al., 2004] in the design of their *Email Act Ontology*. Here, a speech act is represented by a *Verb-Noun* dichotomy, both of which are organised as a hierarchy reflecting the illocutionary intents for verbs and specific categories for nouns. Their final ontology provides for 5 verbs (*Propose*, *Request*, *Amend*, *Commit*, *Deliver*) and 6 nouns (*Data*, *Opinion*, *Ongoing Activity*, *Meeting*, *Other Event*). Although the ontology still draws significantly from Searle’s illocutionary points, the resulting verb-noun pairs are not linearly comparable. Following an analysis of real email corpora, the authors noted that the observed linguistic behaviour does not always reflect the conventional abstract speech acts categories. It is

possible, for example, to have speech acts which are both directive and commissive. Thus proposing a joint task entails an implicit commitment by the sender to perform the task as well as a directive request for the recipient to collaborate.

Cohen et. al's ontology also takes into account non-linguistic uses of electronic communication, such as the use of email to deliver files (represented by the *Deliver* verb). Such uses have also been given due credit by [Goldstein and Sabin, 2006], who went as far as adding three new e-mail-specific categories to the five provided by Searle. These are *Transmissives* which cover the forwarding of information, including files; *Self* – covering the email-specific habit of sending mail to oneself as reminders; and *Non-Personal* – to cater for indirectly addressed messages such as newsletters and list-originating e-mail. Their resulting system of categorisation provided for 12 categories, further refined into 30 subcategories which consisted of 23 'traditional' speech acts plus '7' e-mail-specific acts. The same ontology was employed in the approach in [Khoussainov and Kushmerick, 2005].

A considerable amount of research efforts have looked at *Dialogue Acts* as 'alternatives' to speech acts²⁷. Dialogue acts (also referred to as Dialog Acts) describe basic elements of human communication, rather than words or sentences [Alexandersson et al., 2000]. [Traum, 2000] presents 20 questions that need to be asked in the set-up of dialogue act taxonomies in order to facilitate their shared understanding and use. Dialogue acts mark important characteristics of utterances, indicate the role of an utterance in a specific dialogue and make the relationship between utterances more obvious. For example, in the SWBD-DAMSL dialogue act tag set [Jurafsky et al., 1997] a 'Question' speech act can be further refined into a 'Yes_No_Question', 'WH_Question', 'Rhetorical_Question', etc. [Stolcke et al., 1998] used this system of categorisation to label transcripts of the Switchboard telephone speech corpus with 42 available dialogue acts. The speech act profiling approach in [Twitchell and Jr., 2004] extended the latter dialogue act model by combining it with Alston's idea of illocutionary act potential [Alston, 2000]. Dialogue acts

²⁷The use of the apostrophes is because dialogue acts are nothing but specialised speech acts

were also at the centre of the Verbmobil project, which sought to develop a system that is able to recognise, translate and produce natural utterances in speech using a hierarchy of 32 dialogue acts [Alexandersson et al., 1998]. Inspired by the Verbmobil and the SWBD-DAMSL tag sets, [Corston-Oliver et al., 2004] employed 15 application-specific speech acts.

The fact that dialogue act schemes are more system-specific than generic speech acts, can be more of a drawback rather than an advantage. As pointed out in [Lampert et al., 2006], such systems of categorisation can be overly specific for the wide range of acts that is found e.g., in email conversation. Instead, Lampert et. al. employ the *Verbal Response Modes* (VRM) taxonomy of speech acts [Stiles, 1992], which categorises discourse on two dimensions – the literal meaning and the pragmatic meaning. This taxonomy is distinguished by its construction from crosscutting principles of classification, ensuring universal applicability across any domain of discourse. The end-categories are *Disclosure, Edification, Advisement, Confirmation, Question, Acknowledgment, Interpretation* and *Reflection*. Their later work [Lampert et al., 2006] focused on just two activity-focused speech acts – Requests (for action) and Commitments (to act), demonstrating the practical difficulty in categorising just two speech acts. They identified email-specific phenomena that makes this task difficult and explain how these are subject to context-based ambiguities which in some cases can only be resolved by the author of the email. In their annotation task, the authors instructed the annotators to also assign a different ‘strength’ (*Strong, Medium, Weak, None*) to the two available annotations implying different strength of intention (e.g. a suggestion versus an order for a Commitment). Thus, just like Cohen et. al.’s ontology, this categorisation system also somewhat represents the end-speech acts as a pair, rather than a singleton.

Table 5.1 sums up the surveyed speech act-based systems of categorisation, providing a comparison of what are considered the most relevant features – in the context of this dissertation – namely the type of modelling used, the abstraction used for the repre-

sentation of the speech act, the context of application, whether non-linguistic features specific to digital communication were considered, and finally the amount of speech (or dialogue) acts that were effectively used for the described approaches.

Table 5.1: Comparison of Surveyed Speech Act Categorisations

<i>Categorisation System</i>	<i>Type</i>	<i>Representation</i>	<i>Application Context</i>	<i>DC-Specific Features</i>	<i>Acts</i>
Searle	Taxonomy	Singleton	Speech	No	5
Winograd	Taxonomy	Singleton	Coordinated Action	No	9
COR	Taxonomy	Singleton	Information Seeking Dialogue	No	11
Khosravi	Taxonomy	Singleton	E-mail	No	4
Feng	Taxonomy	Singleton	Threaded Discussions	No	13
Jose	Taxonomy	Singleton	Speech	No	19
Cohen	Ontology	Pair	E-mail	Yes	5
Goldstein	Taxonomy	Singleton	E-mail	Yes	30
SWBD-DAMSL	Tag Set	Singleton	Telephone	No	42
Twitchell	”	”	Instant Messaging	”	”
Verbmobil	Taxonomy	Singleton	Speech	No	32
Corston-Oliver	Tag Set	Singleton	Email	No	6
VRM	Taxonomy	Singleton	Speech	No	8
Lampert	Taxonomy	Pair	E-mail	Yes	8

5.2.4 Speech Acts as the building blocks of Digital Workflows

Apart from message classification, activity identification and enhanced visualisation of digital collaboration, the other major application of LAP with respect to digital commu-

nication systems as presented in Section 5.2.2 is the structuring of entire ‘conversations for action’. This implies that, to one extent or another, it is frequently possible to predict the next speech act in the underlying sequence. With regards to speech, one school of thought believes that predicting the next conversational move is rather impossible, as language is so large that it cannot be described by any number, finite or transfinite [Alm et al., 1992]. Although this might be true, it does not necessarily imply that speech is unpredictable. In fact others have pointed out that in practice, “a significant percentage of conversational language is highly routinised into prefabricated utterances” [Stubbs, 1983]. The same can be said for human digital communication, which again must be pointed out, is more similar to spoken than to written discourse [Khosravi and Wilks, 1999]. Knowledge of the most likely set of subsequent conversational moves, during any point in a conversation, presents an opportunity to model different kinds of conversations. This subsection will discuss attempts exploiting this opportunity.

In a collaboration with Flores, Winograd depicted entire conversations as a network of speech acts [Winograd and Flores, 1986]. Later, Winograd adapted this idea to represent such conversations as state transition networks [Winograd, 1986], with nodes representing conversation states, and arcs representing speech acts. An investigation into speech act sequentiality in conversational structure was also performed by Jose [Jose, 1987] in his attempt to improve coherence in discourse. Here, sequential patterns of speech acts occurring in natural speech were presented following an experiment that saw the manual annotations of spoken child-adult conversations with speech acts.

One of the concluding observations for Chapter 4 (Section 4.6) was that it is possible to model entire e-mail workflows, once the appropriate underlying communicative patterns are elicited. Winograd’s ‘conversations for actions’ and their subsequent modeling can in fact be considered to be nothing other than workflows and workflow modelling, respectively. In fact, Winograd’s own Coordinator system was designed around a workflow concept, and its promise to provide facilities for managing records of moves in conversations can be interpreted as that of a workflow management system. From his LAP

perspective, Winograd introduces notions which are very related to workflow management. He describes how a “conversation is initiated by a request” and how “at each point in the conversation there is a small set of possible action types”.

Winograd’s speech act-based state transition networks have inspired a lot of research in the area. Carvalho & Cohen extended their earlier work (Section 5.2.2) to also factor in the *context* of a message [Carvalho and Cohen, 2005]. They introduced a dependency network-based collective classification method that takes into account sequential correlation between messages (and associated speech acts) within an email thread. They also discussed how these networks can aid with the prediction of speech acts from surrounding acts and provide evidence for the sequential correlation of e-mail speech acts.

Some of the resulting theories and models have been used as foundations for negotiation protocols for autonomous agents. [Smith and Cohen, 1995] developed a semantics for a speech act-based communications language to model task-oriented dialogue in distributed agent systems. In a similar approach [Barbuceanu and Fox, 1995], a coordination language that integrates speech act communication in a structured conversation framework was presented as an attempt to capture coordination mechanisms between collaborating agents. The language caters for the representation of conversation objects, conversation rules, continuation rules, conversation nesting as well as error recovery rules. Other approaches have focused on the use of speech act communication models as a basis for negotiating meaning, e.g. by identifying and repairing misunderstandings (e.g. [McRoy and Hirst, 1995]).

Alternatives to state transition diagrams for visualising successions of communicative acts were investigated by Parunak, who suggested the use of enhanced Dooley Graphs [Parunak, 1996]. These graphs allow for nodes to represent both agent participants as well as states through which they pass. Parunak distinguishes between four ways in which successive communicative actions can be related, adding two additional kinds of relations – *Response* and *Completion*, to another two discussed by Longacre [Longacre, 1986]. The latter distinguished a successive speech act as either constituting a *Reply* or a *Resolution*

to the preceding act.

Winograd's work also discussed how conversations have different "states of completions, in which it is mutually recognised that neither party is waiting for further action by the other" and how "all other states represent an incomplete conversation". He also talks about communicative acts having "certain conditions of satisfaction which characterise a future course of action" by the person(s) at the receiving end. He goes on to say that such conditions of satisfaction are not independent of interpretations, but are subject to ambiguities which can lead to a breakdown of his conversations for action. Some researches have focused primarily on the problem of determining such conditions of satisfaction for individual speech acts. For example Singh, as a formal theory of communication for multi-agent systems [Singh, 1991], presented a formal semantics for speech acts that included conditions for their satisfiability. This is a very relevant issue, as in order to determine the general status of conversation for actions, or *speech act-based e-mail workflows*, one needs to be able to determine the state of each individual communicative act.

Part III

Core

6 Distributed Knowledge Representation on the Social Semantic Desktop

The vision for a Social Semantic Desktop (SSD) as introduced in Chapter 3 envisages a universal platform that enables both personal and distributed information management, social networking and community creation. In order to be realised, the knowledge representation methodology beneath this platform is bound by the four fundamental requirements laid out in Section 3.2. In this Chapter I will present my contributions to the knowledge representation modeling aspect of this project. A lot of material included in this Chapter thus draws from a number of related publications, e.g., [Sintek et al., 2007a, Sintek et al., 2007c, Sintek et al., 2009].

The first and most important requirement laid out for the SSD is driven by the need to provide for a common knowledge representation that supports the integration and retrieval of heterogeneous data across application and desktop boundaries. In other words, the SSD needs to allow for *semantic interoperability* between the semantic desktops, with each doubling as individual source- and end-points of the Semantic Web. Semantic Web languages and protocols are the obvious candidates for formalising the conceptualisations of just about any desktop knowledge as well as for the coordination of local and global information access. Thus, the NEPOMUK knowledge modeling team pursued an approach whereby the user's personal, desktop-based information environment is enhanced with technologies adopted from the Semantic Web paradigm, such as ontologies and the RDF data model.

Semantic Web technology has been adapted, rather than re-used, because the SSD

requirements differ from that for the Semantic Web in a number of ways, most notably the decision to consider each personal desktop as a closed world. The social aspect of the semantic desktop further increases the need for an effective handling of multiple models (second fundamental SSD requirement) as well as multiple interpretations of these models (third requirement). These special requirements make existing semantic representational languages inappropriate, eventually leading to the design of a novel representational language for the distributed environment of the SSD.

The specific requirements and modelling techniques used to design the resulting Nepomuk Representation Language (NRL) [Sintek et al., 2007a] will be covered in the first section of this chapter. The gradual design of the resulting multi-tiered set of ontologies will be covered in the second section. Central to the pursued ontology engineering approach is a knowledge representation scheme that provides adequate expressivity for representing legacy data and common operations occurring on and between desktops, while at the same time respecting the users' mental models (fourth SSD requirement).

6.1 The Nepomuk Representational Language

The first major KR modeling decision taken was the selection of an appropriate representational language that fulfils the requirements set out in Section 3.2. Representational ontologies (Section 2.2.2) serve as the language required to define the vocabulary with which other ontologies are represented. Concepts occurring at this level of abstraction include high-level classes and properties, constraints, etc. The use of representational languages such as RDF/S and OWL as the foundation for all the required knowledge models immediately raised a number of conflicts with respect to the given requirements – especially the second (Subsection 3.2.2) and third (Section 3.2.3). Given the inappropriateness of both languages and the unavailability of any other better alternative, within the KR-modeling team we set out to design a novel and improved representational language. Although still based on the RDF model, this language successfully targets the latter's shortcomings when applied to the SSD scenario. In this section I will explain how

the resulting NEPOMUK Representational Language (NRL) serves as a foundation for the ensuing knowledge modeling, covering all the relevant aspects of the SSD.

6.1.1 Motivation

The first shortcoming of the RDF model (and subsequently RDFS, OWL) with respect to the SSD requirements, is its lack of imposed structure beyond the triple space composed of all existent RDF data. The RDF model does not provide for any modularisation of its data, leading to practical problems such as dealing with invalid or outdated data as well as issues of provenance and trust. These problems can be addressed if the RDF model supported Named Graphs (NGs) [Sintek et al., 2007a] (Section 2.2.4). The RDF recommendation itself does not provide suitable mechanisms for describing or define relations between graphs [Beckett, 2004] [Brickley and Guha, 2004] [Hayes, 2004] [F. Manola, 2004]. Although the extension of RDF with NG support has been proposed [Carroll et al., 2005, Prud'hommeaux and Seaborne, 2008, Sintek and Decker, 2002], and the motivation and ideas clearly stated, none of the available representational languages provide specific support for NGs. A basic syntax and semantics that models minimal manipulation of NGs was however presented by participants of the Semantic Web Interest Group, with the intent of introducing the technology to the W3C process²⁸.

The second shortcoming of both RDF/S and OWL is their strict adherence to the open-world assumption (OWA) (Section 2.2.3), imposing this semantics on all generated RDF instance data. As explained in Section 3.2.3, it is more natural for the semantic desktop to operate on a CWA. Therefore, SSD data must not be restricted to just the OWA. From another perspective, the knowledge modeling in place for the Semantic Web needs to be adjusted to take into account the somewhat different realities of the semantic desktop. Ideally, SSD data should not innately carry any realised semantics or assumptions, meaning that different semantics – or interpretations, can be realised by different desktop applications in different contexts, as required. Given this scenario, although

²⁸<http://www.w3.org/2004/03/trix/>

the semantic desktop itself operates in a closed-world, it is also possible to work with open-world interpretations of particular data modules. This enables importing data with predefined open-world semantics from the Web onto the semantic desktop, while retaining the distinction between the data itself and its semantics. If required, closed-world applications can then work with a closed-world semantics interpretation of the imported graph.

There are additional RDF/S features which are unsuitable for the KR problem at hand, e.g., blank nodes (refer to Section 2.2.1). With regards specifically to OWL, of particular concern are its complex predicate-logic concepts (especially `owl:Restriction` and the associated inference). Due to the performance-oriented design of the SSD, the supported semantic modelling constructs need to be computable in linear effort, with perhaps an exception for transitivity.

Given the unavailability of an adequate representational language for the SSD, we undertook the design and development of a novel language that satisfies all the requirements outlined in Section 3.2. NRL, the resulting language, is an extension to RDF/S that in particular imposes no specific semantics on data and provides support for NGs. NRL addresses several limitations of current Semantic Web languages, especially with respect to data modularisation and customisation aspects. Thus NRL is also of relevance to the general Semantic Web, particularly because of its support for NGs.

In the remaining subsections I will focus on the data modeling concepts introduced and adopted by NRL, particularly on the use of NGs for the modularisation aspect and a graph view concept for the custom interpretation of graphs. I will also discuss how NRL respects the separation between syntax and semantics. As NRL extends RDF/S but, unlike the latter, does not imply any semantics on its use, another discussion focuses on RDF/S vocabulary best practices for use within NRL. In addition I will introduce the constraint extensions provided by NRL before finally demonstrating how NRL's knowledge modeling can be put to good use both conceptually and practically via a prototypical scenario.

6.1.2 Named Graphs – Supporting Data Modularisation

In view of the SSD’s second requirement (Section 3.2.2), it was decided all data handling (including storage, retrieval and exchange) should be carried out in terms of NGs. NRL thus extends RDF to support NGs, creating an intermediate representation layer that enables the management of identifiable, modularised sets of data. The approach is based on the work described in [Carroll et al., 2005], excluding however the open-world assumption, such that NGs do not innately carry any (assumptions on) semantics. These can instead be realised through designated views on graphs (introduced in Section 6.1.3). The following are the core NRL concepts supporting NGs as defined in the NRL vocabulary [Sintek et al., 2007b].

nrl:Graph and nrl:DocumentGraph Instances of these classes represent NGs, such that the name of the instance coincides with the name of the graph. The graph content for an `nrl:DocumentGraph` is located at a URL serving as the URIref for its instance. This allows for existing RDF triples to be re-used as NGs, avoiding the need of a syntax like TriG (Section 2.2.4) to define NGs.

nrl:subGraphOf, nrl:superGraphOf, and nrl:equivalentGraph These relations between named graphs have the obvious semantics: they are defined as \subseteq , \supseteq , and $=$ on the bare triple sets represented by these graphs.

nrl:imports is a subproperty of `nrl:superGraphOf` and models graph imports. Apart from implying the \supseteq relation between the triple sets, it also requires that the semantics of the two graphs is compatible.

nrl:DefaultGraph This `nrl:Graph` instance represents an abstract graph containing all triples external to any user-defined NG. This is provided for consistency and to ensure backward compatibility with triples that are not defined under NRL, since all triples must be assigned to some NG. This approach gives rise to the term RDF Dataset

[Prud'hommeaux and Seaborne, 2008], which is composed of a *default graph* plus a finite number of distinct NGs.

Alongside provenance data (e.g. creator, creation date and time, etc.) it is also possible to attach other useful information to NGs. In particular, for better data management NGs can be distinguished by their roles, e.g., an ontology, an instance-base, a knowledge-base, etc. For this purpose the concept of *Graph Roles* was introduced to distinguish between graphs and their roles. This ensures orthogonal modelling primitives for defining graphs and for specifying their role. A graph role refers to the characteristics and content of an NG and how the data is intended to be handled. Since it is more intuitive to annotate an ontology, for example, rather than the underlying graph, graph metadata is attached to the roles rather than to the graphs themselves. Also, roles are more stable than the graphs they represent, and while the graph for a particular role can change constantly, evolution of the role itself is less likely. An instantiation of a role will consist of a specific type of graph and the corresponding triple set data. The following are the core concepts for the definition of graph roles in NRL.

nrl:Data This is an abstract subclass of **nrl:Graph**, representing an untyped graph role, to enable roles to be used as marker classes through its subclasses.

nrl:Schema and **nrl:Ontology** are graph roles that represent data in a conceptualisation model. **nrl:Ontology** is a subclass of **nrl:Schema**. As marker classes, no specific semantics are assigned to these roles. However, graphs that also contain instance data in addition to ontological/schematic data should be marked as a **nrl:KnowledgeBase**

nrl:InstanceBase marks a NG as containing instances from schemas or ontologies. The properties **nrl:hasSchema** and **nrl:hasOntology** relate an instance base to the corresponding schema or ontology.

nrl:KnowledgeBase marks a NG as containing a conceptual model plus instances from schemas or ontologies. Therefore a **nrl:KnowledgeBase** is a combination of a **nrl:Ontology/nrl:Schema** and a **nrl:InstanceBase**.

nrl:GraphMetadata is used to mark graphs whose sole purpose is to store metadata about other graphs. Data about a graph – *Graph Metadata* – is thus stored in a corresponding graph having this role. The property **nrl:graphMetadataFor** binds a metadata graph to the graph being annotated. Although a graph can have multiple metadata graphs describing it, there can only be one unique metadata graph which defines the graph’s important core properties, e.g. whether it is updatable (through **nrl:updatable**) or otherwise. NRL provides the **nrl:coreGraphMetadataFor** property for this purpose, as a subproperty of **nrl:graphMetadataFor**, to identify the core metadata graph for a graph.

nrl:Configuration is used to represent technical configuration data that is irrelevant to general semantic web data within a graph.

nrl:Semantics Declarative semantics for a graph role can be specified by referring to instances of this class via **nrl:hasSemantics**. These will usually link (via **nrl:semanticsDefinedBy**) to a document specifying the semantics in a human readable or formal way (e. g., the RDF Semantics document [Hayes, 2004]).

The class hierarchy supporting NGs and Graph Roles in NRL, including the core vocabulary presented in this section, is presented in Fig. 6.1.

6.1.3 Graph Views – Supporting Multiple Views and Semantics

In many situations it is desirable to work with an extended or restricted interpretation of specific NGs. To preserve their integrity, NG interpretations should never replace the original. To model this functionality and retain the separation between an original NG

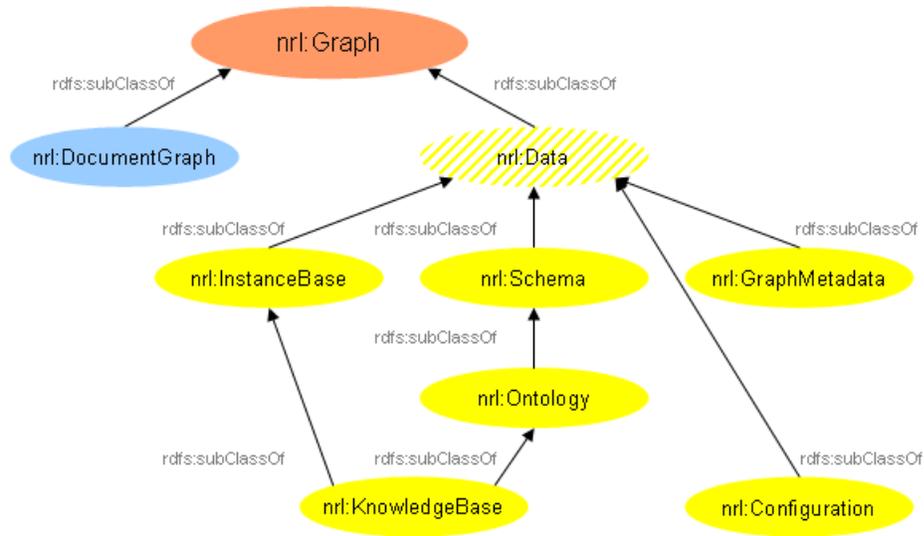


Figure 6.1: NRL Named Graph class hierarchy

and any number of interpretations, a view concept from database theory was adopted to introduce *Graph Views*. Views are defined as an executable specification of an input graph into a corresponding output graph. Informally, views can be seen as arbitrary wrappings for a NG. Views can be realised by applying some algorithm (e.g., specified through rules) to enhance an NG with entailment triples, to return a restricted form of the triple set, or even an entirely new triple set. *View Specifications* execute such realisations, via a set of queries/rules in a query/rule language or via an external application. An example of a rule language that can be employed for the purpose is the SPARQL Query Language for RDF [Prud'hommeaux and Seaborne, 2008]. A way of using SPARQL to realise view definitions (called Networked Graphs) was described in [Schenk and Staab, 2008]. While Networked Graphs allow views to be defined in a declarative way (in contrast to NRL's somewhat procedural way), they lack many of the ideal features for a view language. For example, they do not allow access to the underlying RDF graphs without any interpretation, and they only allow views to be defined via SPARQL. This excludes languages with more advanced semantics like OWL and also languages that do not have a

declarative semantics.

The introduced view concept also provides the required mechanism to impose different semantics on one syntactical structure. NRL therefore also supports *Semantic Views* – view realisations that realise the implicit semantics of a graph according to the underlying language or schema. For example, a view generated by an application that returns the transitive closure of `rdfs:subClassOf` for a specific NG, will indirectly carry the corresponding RDFS semantics. In contrast to graph roles, which have only declarative semantics, semantic views also carry procedural semantics, since the semantics of these graphs are also realised (through `nrl:realizes`) and not solely implied. The following are the core concepts for the definition of graph views, view specifications and semantic views in NRL.

nrl:GraphView represents a view, modeled as a subclass of a NG. A view is realised through a view specification, defined by an instance of `nrl:ViewSpecification` via `nrl:hasSpecification`. The NG on which the view is being generated is linked by `nrl:viewOn`.

nrl:ViewSpecification General view specifications can currently take one of two forms, modeled as the two subclasses `nrl:RuleViewSpecification` and `nrl:ExternalViewSpecification`. As discussed earlier, semantic views realise procedural semantics and are linked to some semantics via `nrl:realizes`. This is however to be differentiated from `nrl:hasSemantics`, which states that a NG carries (through a role) declarative semantics which is not necessarily (explicitly) realised by way a view specification.

nrl:RuleViewSpecification Views can be specified by referring to a rule language (via `nrl:ruleLanguage`) and a corresponding set of given rules (via `nrl:rule`). These views are realized by executing the rules, generating the required output NG.

nrl:ExternalViewSpecification Instances of this class map to the location (via `nrl:externalRealizer`) of an external application, service, or program that is executed to create the view.

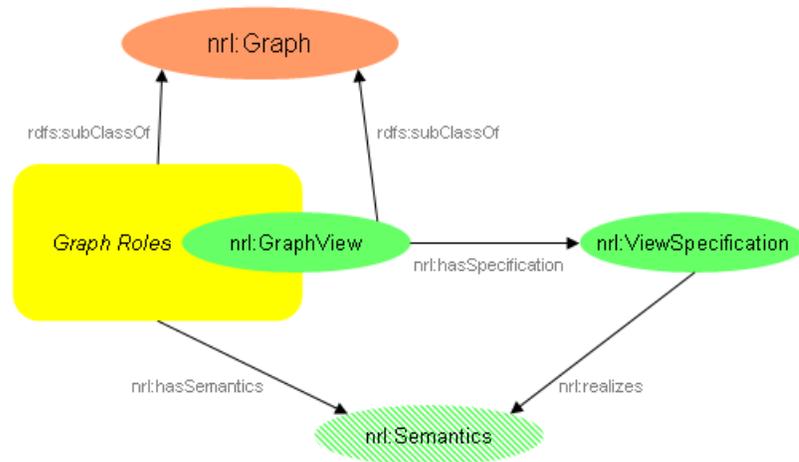


Figure 6.2: Graph Views in NRL

Fig. 3 provides an overview of NRL’s graph view vocabulary. As views are themselves NGs (represented via the subclass relation), one can have a NG that is a different interpretation, or view, of another NG. This modelling can be applied recurrently, yielding a view of a view and so on and so forth. Semantic Views are represented by the intersection of a `nrl:GraphView` and graph roles – whereby the procedural semantics are realised by the former whilst the declarative semantics are defined through the latter.

6.1.4 Decoupling Syntax and Semantics in NRL

NRL’s design enables a complete separation between syntax and semantics. As discussed in Section 2.2.3, these are independent definitions which determine the validity of a statement expressed in a language. Thus, it is possible to express knowledge using NRL syntax but bind it to e.g. RDF, rather than the default personal desktop CWA Semantics (NRL-CW).

Fig. 6.3 depicts this separation in NRL, showing both the syntactical and the semantic

blocks of the language. The syntax block contains the NRL Schema language (upper part) as an extension of a large subset of RDFS (the apostrophe denotes the various best practices recommended for the re-use of this vocabulary, as discussed in Section 6.1.5). The conditions of this extension will be covered in the next two subsections and include specific recommendations on the use of RDFS constructs plus additional constraints. The lower part of the syntax block shows the relationship between representations for NGs, Graph Roles, and Graph Views. Semantic Views are again depicted as an overlap between Graph Roles and Graph Views (as in Figure 6.2).

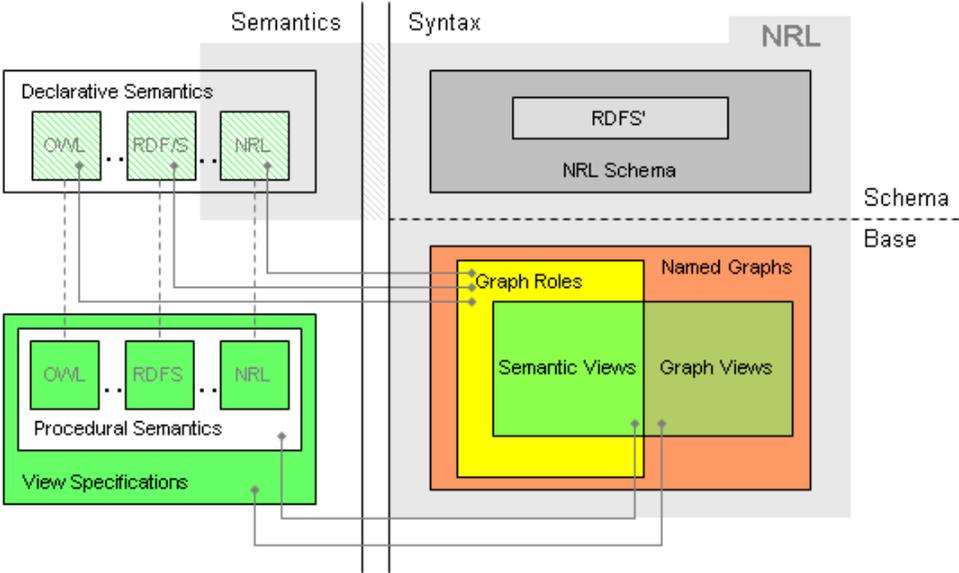


Figure 6.3: Overview of NRL - Abstract Syntax, Concepts and Semantics

The left half of Fig. 6.3 illustrates the semantic block of NRL. In turn, the semantics aspect is split into *declarative* and *procedural* Semantics. Declarative semantics are linked to graph roles, as roles are used to assign meaning to named graphs. A procedural semantics is expressed via view specifications that realise a particular declarative semantics, generate semantic views in the process. This link between procedural and declarative semantics is depicted using the dotted lines. However not all view specification realise a particular semantics, and in this case the views are not tied to any procedural semantics.

Views (and NGs) need neither be assigned a declarative semantics, e.g., in cases when the semantics is not yet known or simply not relevant.

The approach taken in NRL is useful especially in view of the distributed nature of the SSD, which thrives also on semantic knowledge that is available on the Semantic Web. Thus, importing global knowledge from the OWA-based Semantic Web onto the CWA-based local desktop does not result in conflicting semantics, given that it is adapted to the knowledge representation scheme of the SSD. Normally, this requires Semantic Web data to be represented in NRL on import. As one way of addressing this problem, I have contributed in a development effort for a tool that converts RDF/S data to NRL. This tool is provided both as a web service and as an export plug-in for Protégé [Caires et al., 2007] to export RDF/S ontologies to valid NRL data with custom semantics.

The Protégé NRL export window is shown in Fig. 6.4. Options provided by the interface include the specification of the name for the generated NG, its namespace and an abbreviation for the latter. Generated data consists of two graphs: an NG containing the actual imported/exported statements, and a metadata graph describing it. The user is also allowed to define the name of this metadata graph by way of a suffix that can be appended to the input name for the main graph. The metadata graph will contain the various provenance and descriptive metadata attached to the main graph, such as its role (e.g. knowledge base), its status (e.g. testing), a version, a creator/contributor(s), and whether it is updatable.

Most relevant to this subsection is the ability to choose a declarative semantics for the generated graph(s). In Fig 6.4 the user has chosen the NRL-closed world semantics (NRL/CW), which are the standard semantics adopted on the local desktop. The interface also allows the user to serialise the generated graphs in either TriG or RDF/S. As the latter does not support NGs, the conversion will result in two files containing the NG and its metadata graph respectively. Other options allow the user to define a file as a Document Graph, if they intend to keep the generated NRL data within one. The user can also request the automatic replacement of RDFS containers with RDF Lists. This

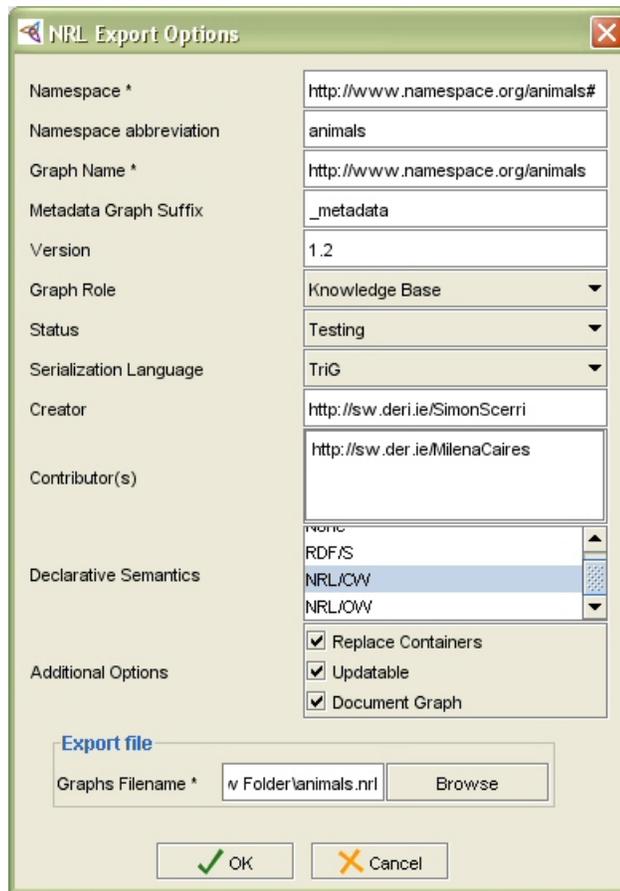


Figure 6.4: NRL Export for a Protégé RDF/S Project

replacement may be required to satisfy one of NRL’s recommendations on the use RDFS, introduced next.

6.1.5 RDF/S re-use best practice for the Social Semantic Desktop

NRL data generated on the desktop is attributed the NRL/CW semantics, unless otherwise specified. As NRL extends RDFS, this semantics is bound by a number of recommendations, or best practices, on the use of RDFS. Disregarding these recommendations results in NRL data with unclear semantics, since this is not compatible with the NRL/CW semantics. Expressed in another way, not following the recommendations will

result in *invalid*, albeit *legal*, NRL data. In this case, unless other types of declarative semantics are specified by the creator, the NRL data will have unclear semantics. Although a machine is still able to process legal NRL data, irrespective of whether it is valid or not, it won't be able to draw conclusions or perform reasoning unless the data fully conforms to a specific semantics (See 2.2.3 for a discussion of conflicts arising from data with unclear semantics). Standard NRL data on the desktop is thus subject to the following RDFS recommendations.

rdfs:domain, rdfs:range In Section 2.2.3 I explained how a CWA interpretation considers RDFS definitions as strict constraints on their instances. This is especially true for the `rdfs:domain` and the `rdfs:range` elements. Under an OWA, when using a property to relate two resources, one is implicitly casting the type of those resources to the types specified in the property's domain and range definitions. In other words the use of a property evokes additional implicit statements about the types of the objects being related, even if these types are different than the types that have been predefined for the objects (if at all). In a CWA scenario such as that on the semantic desktop, this is not possible as in order to relate two resources, their types must fit the expected domain and range in the first place. This also means that untyped resources cannot be related under a CWA. Therefore, valid NRL (with NRL/CW semantics) enforces a strict adherence to domain and range constraints for properties, implying that untyped resources cannot be related through a property.

Blank Nodes Section 2.2.1 introduced blank nodes – anonymous resources created to make further statements about other resources without the need of having a unique URI. However, blank nodes are one of the most problematic concepts in the RDF model. They often pose problems in graph aggregation, because there is no real way to determine which blank nodes from two different graphs should be considered identical. Blank nodes are discouraged by many Semantic Web practitioners such as in the Linked Data context [Bizer et al., 2007]. Blank nodes are semantically difficult to handle and

as a result it is very difficult to implement them correctly, if at all. Thus, the use of blank nodes in conjunction with NRL elements is strongly discouraged, and NRL data containing blank nodes is also legal but invalid.

Collections and Containers [Bizer et al., 2007] also questions the need for RDFS collections and containers, when this information can be expressed using multiple triples sharing a predicate. Application-wise, handling the latter option is more straightforward. RDF containers are resources that contain a set of other resources, or members. They can be used to describe groups of resources, by way of three different concepts (`rdf:Bag`, `rdf:Seq`, `rdf:Alt`) each having different implications on membership (unordered, ordered or alternatives). RDF collections similarly describe groups that can contain a specified list of members. However, whereas an RDF container only states that the contained resources are members, an RDF collection implies that only the specified resources can be members of that collection. Being non-exclusive, from a CWA point of view, the semantics of containers are unclear and therefore difficult to handle. As the sole use of either containers *or* collections is sufficient to model grouped resources (at least within the KR envisaged for the SSD), the use of containers in NRL is discouraged in favour of collections.

Reification As discussed in Section 2.2.4 the main motivation for modelling RDF data using quads was driven by the limitations of RDF reification [Carroll and Stickler, 2004]. Thus, the purpose of reification is made redundant by NGs, which provide for entire graphs what reification provides for individual statements. With NRL's NG-based data modelling metadata can easily be attached to one or more statements contained in a graph. NRL even provides a specific metadata graph role for the purpose. Furthermore, the semantics of RDF reification are unclear, and application-wise the handling of reified statements is cumbersome. These problems are also cited by Bizer et. al. in their best practices for Linked Data (refer to Section 2.1.4). Thus, NRL strongly recommends against the use of RDF reification.

rdfs:isDefinedBy The use of this element is strongly discouraged as its semantics are too vague. Allowing any two resources to be related via this property irrespective of their type, results in the generation of statements that are difficult to interpret. In a CWA setting, this unrestricted use renders any such statements useless. Therefore NRL data containing statements using this property is also legal but not valid. Furthermore, the range of more specific properties made available by the rest of the NEPOMUK ontologies renders this property itself redundant.

6.1.6 Enabling richer Semantics via Constraint Extensions

The RDFS vocabulary provides only two elements that impose some kind of constraint on which resources can be related and by which property. *rdfs:domain* and *rdfs:range* provide a means of defining *Class-Type Constraints* on RDF data. OWL's vocabulary provides for more constraints on the knowledge that can be expressed through its use, e.g., limitations on the amount of values that a property can take and on actual pairs of resources that the property should or should not, through inference, relate. OWL also provides a way of specifying local domain and range constraints through OWL Restrictions that restrict the way in which properties can be used by instances.

During the design phase of NRL, it was decided that additional constraint vocabulary such as those provided by OWL should be provided. However the more complex predicate-logic concepts of OWL, such as OWL Restrictions, are undesirable. Also, the direct re-use of the desirable constraint extensions from OWL was not as straightforward as re-using RDFS vocabulary, where specific recommendations for their use were outlined. Given OWL's underlying semantics, we instead decided to redefine the required constraints anew in NRL.

To enable more expressivity in the representation of desktop knowledge while retaining clear semantics and not jeopardising the SSD's performance, NRL provides two categories of constraint extensions in addition to RDFS's class-type constraints. The three resulting NRL constraint categories are summarised in Table 6.1. The provided cardinality

Table 6.1: NRL Constraint Vocabulary

Class-Type Constraints	Cardinality Constraints	Resource Relation Constraints
rdfs:domain	nrl:cardinality	nrl:TransitiveProperty
rdfs:range	nrl:minCardinality	nrl:SymmetricProperty
	nrl:maxCardinality	nrl:AsymmetricProperty
		nrl:ReflexiveProperty
		nrl:inverseProperty
		nrl:FunctionalProperty
		nrl:InverseFunctionalProperty

constraint and resource relation constraint extensions are inspired by OWL, with some added syntactic sugar such as `nrl:AsymmetricProperty` (although this has now been incorporated in OWL2 [Motik et al., 2009]).

NRL’s recommendation on the use of RDFS class-type constraints (Section 6.1.5) is naturally extended to NRL’s constraint extensions. Given a possible closed-world view, in order to generate valid NRL data an agent needs to be able to check – prior to using a property – whether the resources are indeed valid candidates that satisfy the selected constraints. This called for a NRL Validator to check the validity of desktop data against the implicit local NRL/CW semantics. A prototype validator was developed as a proof of concept, to provide practical guidelines for the developers of the SSD. The Validator checked the validity of input NGs against the RDFS recommendations presented in the previous subsection and the semantics underlying the constraint extensions provided by NRL.

6.1.7 NRL in use: An example

This Subsection will demonstrate how the knowledge modeling concepts and paradigms underlying NRL enable an SSD application to realise custom interpretations, or perspectives, of semantic data modules on the desktop.

The example, outlined in Fig. 6.5, involves a senior biologist, Anna, who would like

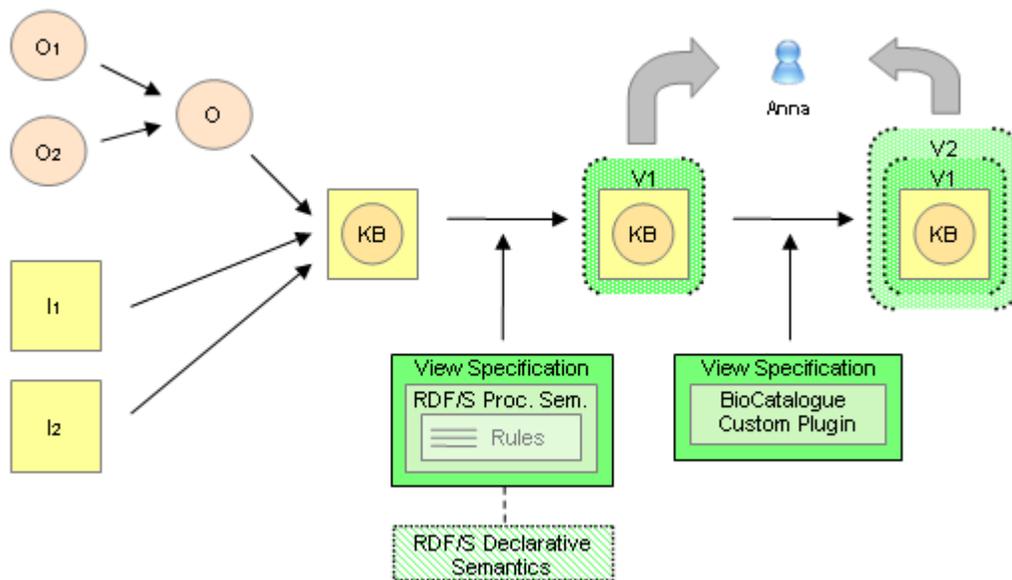


Figure 6.5: NRL dataflow for Anna's task

to compile an online knowledge base describing animal species for her students to access. She knows that a rather generic ontology describing the domain is already available on the Web. On downloading this ontology onto her desktop its statements are stored within a NG, O_1 , and an amount of core metadata is automatically generated. In particular, this includes provenance information and a role for the graph, determined to be that of a *nrl:Ontology*. Anna also acquires a vast amount of instances for this ontology via a fellow biologist. On importing this data from the other SD it is stored within a second NG, I_1 , whose role is this time automatically determined to be an *nrl:InstanceBase*.

However, the expressivity provided by the ontology does not satisfy Anna's requirements – O_1 fails to cover some attributes of special interest to Anna's work, namely those relating to the animals' diet and food chain. Therefore an ontology engineering tool (such as Protégé) is employed to extend O_1 with these concepts and attributes. Unbeknown to Anna, this additional knowledge is stored as a separate ontology, within a third NG, O_2 . The knowledge in both O_1 and O_2 is however combined into a pseudo ontology O , which

imports both ontologies and presents the combined knowledge to Anna as the extended animal species ontology. Similarly, the instance data in I_1 is extended with an additional NG containing knowledge pertaining to the included animals' diet, I_2 .

Anna loads the extended ontology and both of the associated datasets in BioCatalogue²⁹— a semantic application to manage and visualise catalogued species information. The application combines the concepts and attributes in O (i.e. O_1 and O_2) with the knowledge expressed in both instance datasets (I_1, I_2), into a union graph KB . Given the mixed roles of its subgraphs, this graph is assigned the role of a `nrl:KnowledgeBase`. Therefore, KB is the supergraph of O , which in turn is the supergraph of O_1, O_2 , and I .

BioCatalogue generates interactive and explorative graph structures from loaded data. These visualisations are based on the hierarchies defined for a given dataset, such as KB . In this example, each end-leaf denotes a different species. When clicking on a node or a leaf, BioCatalogue presents the available knowledge for that group or species. This knowledge includes all the data available for a particular node, but also that available for its parent nodes. This information is not present in the loaded data by default. KB 's triples for example, do not go beyond defining *direct* subclass relationships between groups of species and their *direct* attributes. The indirect superclass and subclass relationships and inherited attributes are instead *inferred* through realising the semantics of `rdfs:subClass` and `rdfs:subProperty`. NRLSail [Sintek et al., 2007c] is a prototypical open-source implementation of NRL³⁰, that performs this kind of inferencing. In BioCatalogue, this inference is executed via specific rules that augment the KB dataset with entailment triples. The rules are defined within a custom `nrl:ViewSpecification` that computes the procedural semantics for the two RDFS properties named above by executing their transitive closure. To retain both the underlying model and the model with the required semantics, the augmented data does not replace KB . Instead, a semantic view is generated over it, resulting in a separate graph $V_1(KB)$.

²⁹The application is only provided as a conceptual proof of concept and is therefore fictitious.

³⁰<https://dev.nepomuk.semanticdesktop.org/wiki/NRLSail>

Anna feels overwhelmed by the volume of information that she needs to sift through in order to find a specific point of interest in this graph. As she is currently mostly interested in the dietary habits of species, she does not need to see all the other attributes when exploring the data. BioCatalogue also offers the possibility of generating custom views over a dataset via plugins, which serve as external view specifications and can also be shared online. Anna is supported with creating a custom plugin that only shows the attributes she is interested in when selecting a node in dataset $V_1(KB)$. Of course, Anna does not want to permanently discard all the other information from this graph, just to generate this custom view. Its view specification is instead applied on top of $V_1(KB)$ to generate an additional view, $V_2(V_1(KB))$, that contains only the relevant information. All seven named graphs on which this last view is generated upon are still intact and have not been affected by any of the operations along the way. If one of the underlying graphs is changed, the views can be regenerated, thus avoiding data consistency issues and the possibility of working with outdated data that can't be updated because links to underlying models have been lost.

Listing 8 demonstrates how the NRL vocabulary can be employed to represent the data for the above example. Below is a summary of the TriG serialisation:

- [1] namespace declarations
- [2-5] ontology graphs (`ex:o1` and `ex:o2` are defined and imported into `ex:o`)
- [6-8] instance/knowledge base definitions
 - [9] contents of ontology `ex:o2`, defining extended animal domain
 - [10] contents of instance base `ex:i2`, defining instances of animals in `ex:o2`
- [11-13] `ex:v1kb` is defined as a view over `ex:kb` via view specification `ex:rvs`, which realises RDFS semantics for `rdfs:subClassOf` and `rdf:type`, through SPARQL-inspired CONSTRUCT queries (for this to work, a real rule language is required)

Listing 8.

```
[1] @prefix nrl: <http://www.semanticdesktop.org/ontologies/2007/08/15/nrl/> .
    @prefix ex: <http://www.example.org/vocabulary/> .
[2] ex:o2 rdf:type nrl:Ontology .
[3] <http://www.domain.com/o1.rdfs> rdf:type nrl:Ontology ,
                                     nrl:DocumentGraph .
[4] ex:o1 rdf:type nrl:Ontology ;
        nrl:equivalentGraph <http://www.domain.com/o1.rdfs> .
[5] ex:o rdf:type nrl:Ontology ;
        nrl:imports ex:o1, ex:o2 .
[6] ex:i2 rdf:type nrl:InstanceBase ;
        nrl:hasOntology ex:o2 .
[7] http://www.anotherdomain.com/i1.rdf> rdf:type nrl:InstanceBase ,
                                          nrl:DocumentGraph .
[8] ex:kb rdf:type nrl:KnowledgeBase ;
        nrl:imports ex:o, ex:i2, <http://www.anotherdomain.com/i1.rdf> .
[9] ex:o2 {
    ex:Animal rdf:type rdfs:Class .
        ## further Animal Ontology definitions here ## }
[10] ex:i2 {
    ex:CandyCaneWorm rdf:type ex:Flatworm ;
        ## further Animal Instance definitions here ## }
[11] ex:v1kb rdf:type nrl:KnowledgeBase, nrl:GraphView ;
        nrl:viewOn ex:kb ; nrl:superGraphOf ex:kb ;
        nrl:hasSpecification ex:rvs .
[12] ex:rvs rdf:type nrl:RuleViewSpecification ;
        nrl:realizes ex:RDFS semantics ; nrl:ruleLanguage "SPARQL" ;
        nrl:rule "CONSTRUCT {?s rdfs:subClassOf ?v} WHERE ..." ;
        nrl:rule "CONSTRUCT {?s rdf:type ?v} WHERE ..." .
[13] ex:RDFS semantics rdf:type nrl:Semantics ; rdfs:label "RDFS" ;
        nrl:semanticsDefinedBy "http://www.w3.org/TR/rdf-mt/" .
[14] ex:v2v1kb rdf:type nrl:GraphView, nrl:KnowledgeBase ;
        nrl:viewOn ex:v1kb ; nrl:hasSpecification ex:evs .
[15] ex:evs rdf:type nrl:ExternalViewSpecification ;
        nrl:externalRealizer "BioCataloguePlugin1" .
```

[14-15] similar to [11-13], but here `ex:v2v1kb` is realised with the help of an external BioCatalogue plugin, as designed by Anna

6.2 Modelling the Building Blocks of the Desktop

An overview of the approach taken for the modelling of the less abstract desktop information items and their relationships is hereunder presented. In order for the SSD to function, it requires knowledge about practically everything that it contains, maintains, generates and shares. One of the major goals of the SSD is to allow users to organise this information and enrich it with annotations. The heterogeneous nature of the various data structures and formats of desktop native resources, the amount of desktop applications consuming them, and the many ways in which this data can be interpreted and used in different scenarios, necessitated a standardised knowledge representation scheme that is both sufficiently flexible and expressive.

A layered approach to ontology engineering was pursued, with lower-level ontologies deriving from higher-level ontologies, and NRL serving as the representational language. Details about this approach are presented in the first subsection, including an overview of the most important ontologies developed. Although I didn't directly contribute to all of the ontologies' design, I participated in the coordination of their modeling. In the second subsection I then focus on a high-level ontology whose design I personally lead. The last part of this section discusses how we synchronised all the new (or extended) knowledge modelling, in view of the SSD-specific requirements, with the existing state-of-the-art in the spirit of re-use promoted by the Semantic Web.

6.2.1 NEPOMUK Ontologies Stack

To correctly manage the creation and integration of the numerous required ontologies we pursued a serial approach, starting with the design of the higher (general, abstract) knowledge models, and moving on to the lower-level (more detailed) ontologies. This

layered approach is illustrated in the NEPOMUK Ontologies Stack (Fig. 6.6) – a top-down conceptual representation of the engineered ontologies and their interdependencies, which also served as the ‘roadmap’ for their gradual design. The Stack is horizontally divided into the three standard ontology layers introduced in Section 2.2.2, i.e. the Representational, the Upper-level (Foundational) and the Lower-level (Domain) Layer.

This multi-layered approach is similar to design principles undertaken by other ontology-engineers. Alexakos et al. described “a multilayer ontology scheme for integrated searching in distributed hypermedia” [Alexakos et al., 2006], differentiating between an *Upper Search Ontology Layer*, a *Domain Description Ontologies Layer*, and a *Semantic Metadata Layer*. Xiao and Cruz also described a multi-ontology approach for the representation of personal resources and mappings between them [Xiao and Cruz, 2005], differentiating between an *Application Layer*, a *Domain Layer* and a *Resource Layer*.

NRL is shown at the representational layer, and was the first to be developed. Subsequently, the project’s knowledge modelling team moved on to the next stage: representing actual information items and their relationships. Three main components constitute the upper-level ontology layer:

- The NEPOMUK Information Elements (NIE) [Mylka et al., 2007] is an ontology framework composed of seven different, but unified, ontologies. These collectively provide vocabulary for describing native resources which are commonly present on the SSD. The NIE-core ontology provides the basic vocabulary for desktop information elements, whereas a number of more specific ontologies aim to represent legacy data in its various forms, namely the Contact Ontology (NCO) for contact information, Message Ontology (NMO) for messaging information, File Ontology (NFO) for basic file metadata and expressing file system structures in RDF, NEXIF for image metadata, NID3 for audio metadata and NCAL for calendaring information.
- Of the ontologies in this layer, the NEPOMUK Annotation Ontology

(NAO)[Scerri et al., 2007b] is the most high-level, defining trivial relationships between desktop entities as perceived by the user. As I provided a major contribution to the design and development of NAO, its in-depth coverage will be presented in the next subsection.

- The Personal Information Model Ontology (PIMO) [Sauermann et al., 2007] combines knowledge in the rest of the ontologies to express an individual's unified personal information model, i.e. it is able to represent the comprehensive mental model of all the user's personal data on and beyond their desktop.

Whereas NIE is concerned with native data sources and describes concepts like 'File', 'Email', 'Contact', etc., NAO and PIMO add value to this data by expressing tags, people, places, projects and anything else the user might be interested in. Once the high-level ontologies were considered stable, the development of the lower-level ontologies took place. The design of the lower-level ontologies was not directly coordinated at the project level, but by groups or individuals developing domain-specific applications for the SSD (e.g., the Task Modelling Ontology for semantic task managers [Brunzel and Grebner, 2008] or the Semantic Email Ontology for semantic email clients introduced in this thesis (Section 7.3)).

Given the nature of the SSD a large number of ontologies are either related to, or meant to be used for, personal knowledge management. An individual/group of users is free to create or modify existing concepts for their collective (shared) or individual personal information models. This personal user aspect of the ontologies is highlighted accordingly in the Ontologies Stack (Fig. 6.6). Conceptually, it includes all concepts and relationships that the desktop user deals with, as opposed to all concepts and relations required to model every aspect of the SSD.

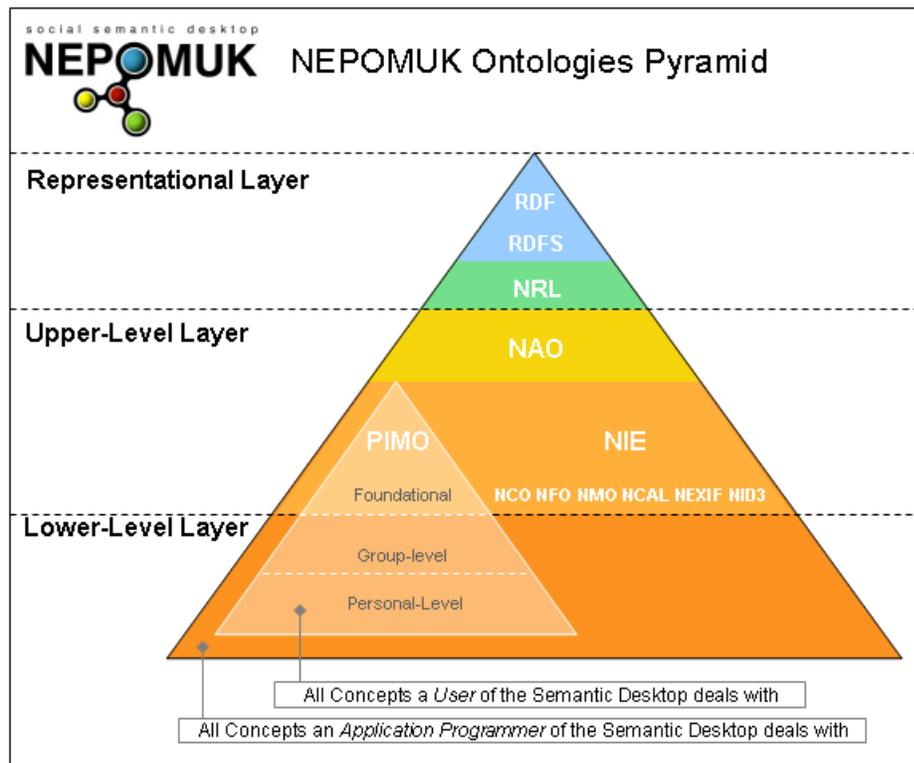


Figure 6.6: The NEPOMUK Ontologies Stack

6.2.2 The Nepomuk Annotation Ontology

One of the initial discussions related to the design of this ontology was in regards to the meaning of what constitutes an annotation. The meaning of the term *semantic annotation* is highly contextual [Uren et al., 2006] [Handschuh, 2007], also because one needs to consider both annotation as the process (of annotating), as well as the end-result (an annotation) [Oren et al., 2008]. Within the SSD, we decided to call any description that goes further than creating resources and defining their elementary relationships (as laid out by domain ontologies) as *annotation*. Thus, NAO provides for the high-level, as opposed to domain-specific, annotation of information objects on the SSD. For example, a user can create an instance of a ‘Person’, and provide values for all the elementary properties that an instance of ‘Person’ can have (e.g. name, address, knows, etc.). Via

vocabulary in NAO, the user can then further enrich semantic knowledge and annotate the resources with more information, e.g. personalised and user-friendly identifiers, labels, descriptions, tags, ratings.

NAO also provides vocabulary for representing high-level relationships between information items. Generic relationships may exist between resources across multiple domains, and via NAO the user is able to make these trivial relationships more explicit. Thus NAO can be used to concretely connect any two resources which are related under some context in the user’s mental model. For example, a user wants to state that a ‘Document’ is about some instance of ‘Person’. This relationship is too general to be applied at the domain ontology level, since such a relationship may exist between other concepts in other domains e.g. between ‘Conference’ and ‘Technology’. Although this information is optional and does not reflect the elementary nature of a ‘Document’, it contributes to improved data unification. The more such relationships are exposed to machines, the higher the knowledge cohesion on the desktop, and the easier it becomes to perform information retrieval, inference, reasoning etc.

6.2.2.1 Vocabulary Overview

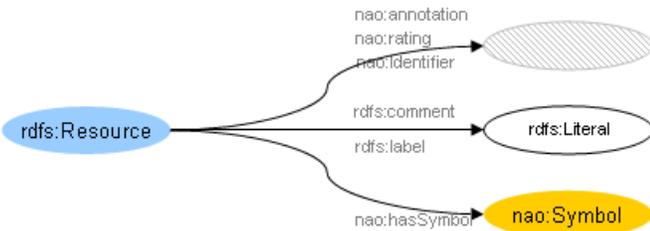


Figure 6.7: Basic Annotation Properties

Given the high-level status of NAO, it can be used to describe and relate virtually any information items present on the platform. Annotation was modelled via properties, rather than classes, as it was agreed that annotation can always be translated to some form

of relationship, rather than a concept per se. This was also the idea for the `rdfs:label` and `rdfs:comment` properties in the RDFS vocabulary, which have been re-used in NAO as a form of textual annotation. Generic annotation (Fig. 6.7) is represented at its highest level with the abstract `nao:annotation` property. Although it is not meant to be used, it is the superproperty of many other annotation properties. Just like `nao:rating` and `nao:identifier`, the range for this property is unspecified, because at this high-level, an annotation's nature can be both textual (literal) or non-textual (concept or instance).

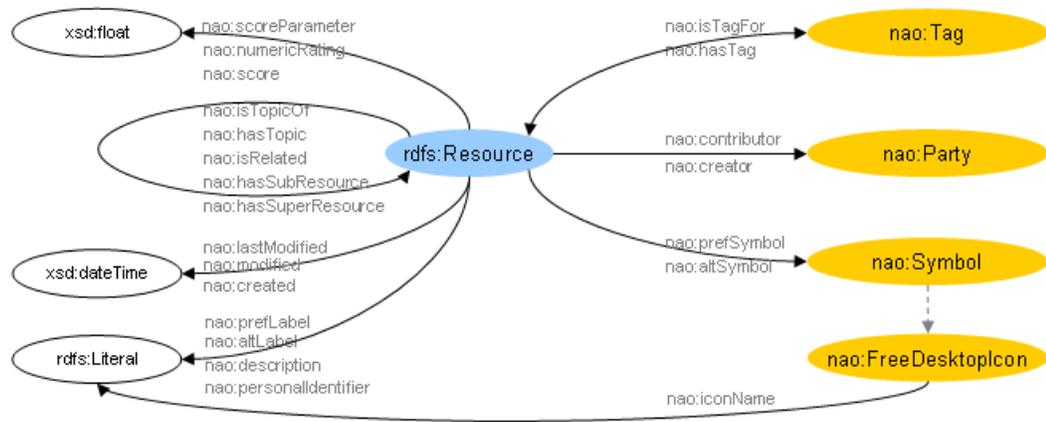


Figure 6.8: More specific annotation properties

Fig 6.8 introduces richer annotation relationships that derive from the basic annotation properties presented in Fig. 6.7. Amongst these are a number of properties that provide for simple, yet useful, links between resources. `nao:isRelated` annotates a resource with pointers to related resources, e.g. a blog entry for an event may be linked to the logo for the same event. This property is semantically symmetric in nature. `nao:hasTopic` and its inverse can be used to state that a resource is about some concept, or vice versa. `nao:hasSuperResource` and its inverse are an example of annotation that offers practical use in the SSD context. Through these properties, super-sub relationships can be defined between any two resources, whereby one resource can be then treated as dependent on another. Thus if a resource is deleted or removed from the SSD, all its subresources can also be automatically removed, unless other superresources still exist on the system.

A number of properties in Fig. 6.8 directly extend those in Fig. 6.7, e.g., `nao:numericRating` extends `nao:rating` to restrict the range to a specific format. Similarly, NAO provides subproperties of `rdfs:label` (`nao:prefLabel` and `nao:altLabel`) to differ between a unique compulsory label for all SSD items and alternatives. Within the SSD, `rdfs:comment` is reserved for technical users whereas its subproperty `nao:description` was provided for use by the end-users.

6.2.2.2 Graph Metadata Vocabulary

A subset of NAO's vocabulary is dedicated to provenance-related metadata, such as `nao:creator`, `nao:contributor`, `nao:created`, `nao:modified` and `nao:lastModified`. These properties can not only be used to attach metadata to information elements, but also to entire NGs. NRL's special *Graph Metadata* graph role (Section 6.1.2) was envisaged to carry this kind of annotation pertaining to specific NGs. NRL already provides some vocabulary that is used to define essential graph metadata, including the graph role specification itself (Fig. 6.1), e.g. whether a graph is updatable and pointers to the declarative semantics for a graph. NAO provides more NG-specific annotation, via properties applying to all NRL graph roles (Fig. 6.9), or as in the case of `nao:serializationLanguage`, specifically to `nrl:DocumentGraph` (documents that encode NGs).

6.2.2.3 Support for Web 2.0 Technology in NAO

Another aspect of NAO focuses on Web 2.0 technology, especially tagging, which can be considered a form of annotation. Given its popularity and wide-spread use, the concept of tagging was adopted in the data modelling. NAO enables users to attach custom descriptions, identifiers, tags and ratings to resources on their desktop. Fig.6.10 shows how NAO can be used to represent tagging practices on the SSD. The user can tag resources conventionally, automatically creating an instance of `nao:Tag`. The tag's creator (an individual/organisation as an instance of `nao:Party`) is linked via `nao:creator`. A unique default

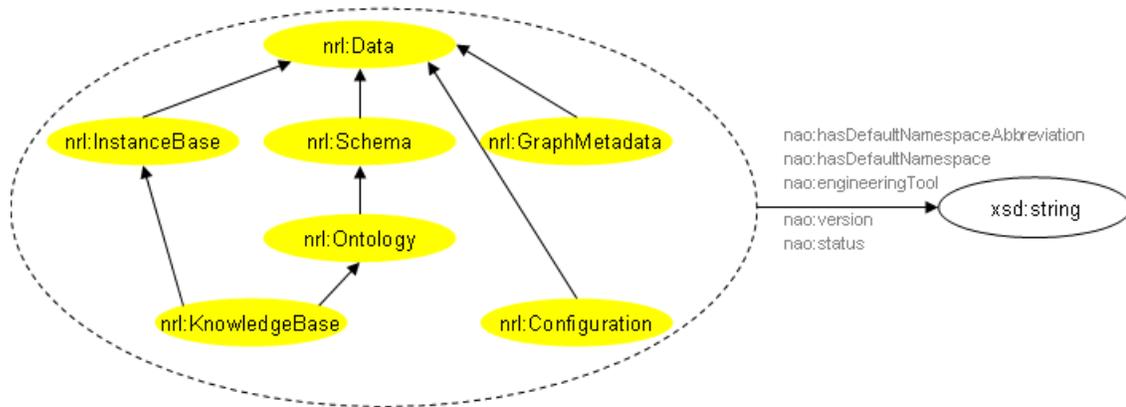


Figure 6.9: NAO Graph Metadata Vocabulary

name is defined by the creator via `nao:prefLabel`, whereas additional custom names can be defined using `nao:altLabel`. A description can be provided via `nao:description` and an icon/image can be attached to the tag via `nao:prefSymbol`. The creation and modification times of the tag can be retained via `nao:created`, `nao:modified` and `nao:lastModified`. A newly created tag is linked to the tagged resource via `nao:hasTag`. An automatic inverse relationship is created to link the new tag with the resource via `nao:isTagFor`.

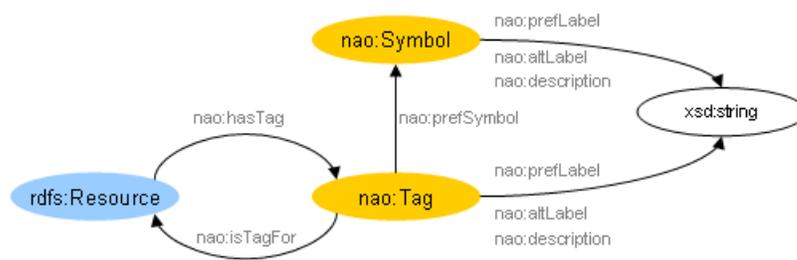


Figure 6.10: Representing Tagging practices via NAO

The relevance of NAO with respect to the social semantic tagging efforts discussed in Section 3.1.2 was discussed in a collaboration with Kim et. al. [Kim et al., 2008a, Kim et al., 2008b]. These contributions compare NAO to other ontologies which aim to

provide a semantic model for tagging and folksonomies. In particular, NAO was found to be one of only two ontologies (the other being SCOT, Section 3.1.2) whose knowledge model is able to empower a subsequent semantic tagging system to provide for all of the following tagging features:

- **Related Tags:** the ability to present related tags to the user;
- **Tag Variant Support:** the ability to store/link equivalent tags (based on language, synonyms, etc.) or provide alternative labels for the same tag;
- **Social Network:** the provision for social networking functionality, i.e. the possibility for users to connect and share personal data
- **Tag Meaning:** the system can provide for a representation of a tag's meaning
- **Multiple Resources:** the system allows for tagging multiple resource types
- **Multiple Taggers:** multiple users can tag the same resource on the system
- **Tag Data Portability:** tagging data can be exported and re-used elsewhere
- **Remote Tag Linking:** the system can relate tags to tags in external systems

6.2.3 Correlating with Related Work

Although NAO was created from scratch, it re-employs concepts from various ontologies such as the Gnowsis ontologies [Sauermaun et al., 2006], and various tagging ontologies such as the *Common Tag Format*³¹. Properties such as `nao:altLabel`, `nao:symbol` and the symmetric `nao:isRelated` property are directly inspired from the *Simple Knowledge Organisation System Ontology* (SKOS) [Miles, 2004]. A subset of NAO's graph metadata vocabulary is based on the *Ontology Metadata Vocabulary* (OMV) [Palma et al., 2009]. However, after NRL's distinction between graphs and roles, NAO's graph metadata vocabulary is applicable to these roles, rather than ontologies.

³¹<http://www.commontag.org/>

Vocabulary elements needed to be recreated with different specifications in NAO, rather than reused, as the semantics of the existing elements were not compliant with the SSD requirements. The vocabulary reuse versus recreation problem was not unique to NAO, but a general issue in the development of all ontologies. As explained in Section 6.1.6, NRL provides constraint extensions that are at face value identical to elements provided by OWL. The other upper-level ontologies have also been inspired by existing ontologies and models, including those developed for desktop information domains within projects such as Haystack [Adar et al., 1999], IRIS [Cheyer et al., 2005] and Gnowsis [Sauermann et al., 2006]. Other examples of these ontologies are FOAF [Brickley and Miller, 2005], Dublin Core ³², as well as more domain-specific ontologies such e.g. ICAL for calendar entries [Dawson and Stenerson, 1998] and the EXIF ontology [Kanzaki, 2004] for image metadata.

Recreating new vocabulary when vocabularies and ontologies covering certain domains already exist, seems to contradict the very idea of knowledge sharing on the Semantic Web. However, in some cases this was necessary, as in certain aspects, the existing vocabulary elements did not conform to the requirements. Faced with this problem, we had three options:

1. Redefine the semantics of elements within vocabularies for use within the SSD
2. Recreate new elements with the required semantics and ignore existing ones
3. Recreate new elements with the required semantics while providing a mapping between them and existing vocabularies

The first option would have created a major problem when it comes to heterogeneity issues. One cannot redefine an element if it already has a defined semantics, because when encountering such an element, it would not be possible to decide in which context to interpret it. Restrictions are a more subtle form of redefinition. This option was

³²<http://dublincore.org/>

pursued in the re-use of sections of RDFS constructs within NRL (Section 6.1.5). Here, this was considered sufficient since it did not violate the predefined semantics defined by RDFS. Option two goes against the knowledge sharing concept of the Semantic Web, that is, to have shared conceptualisations, promote the reuse of ontologies and discourage the recreation of data. The third option is a variant of the second option, where although new elements are re-created, the relation between the new and the existent elements is modelled using mappings (e.g. subclass, hyponym, meronym, etc.). Although new elements satisfying the SSD requirements are created, existent elements are not ignored and therefore the shared conceptualisation ideology is respected. This latter option was perceived to be good practice when designing the required ontologies, when existing elements having SSD-compatible semantics were required.

6.3 Conclusions

This chapter presented my contributions to the Social Semantic Desktop's knowledge modeling task, in the light of the requirements discussed in Section 3.2. More specifically, the pursued approach took into account the following paradigms and modelling methodologies:

- *Semantic Web Technologies and Ontologies*: to enable the use of a standard data representation format offering the required level of expressivity to support semantic interoperability and application-data integration across multiple desktop systems.
- *Named Graphs*: to enable the integration of an intermediate representation layer that enables the management of identifiable, modularised sets of data.
- *Graph Views*: to support flexible and malleable interpretations of existing data sets such that they do not modify the originals, geared towards arbitrary semantic end-user applications.
- *Multi-tiered Ontology Design*: To enable a modelling that reflects different levels

and aspects of the SSD and ensure easy maintenance and extensibility.

The most important design decisions undertaken was the requirement for a novel representational language to regulate and constrain the general expressivity of the whole system. NRL fulfils the requirements arising from the distributed knowledge representation and heterogeneity aspects of the SSD, which were not satisfactorily supported by the existing state of the art. NRL's major design principles are its support for NGs, multiple interpretations of these graphs (views) and the strict decoupling of data (sets of triples, graphs) and their interpretation (semantics).

Once the representational backbone was in place, focus was turned to the design and development of the other ontologies in the layered Ontologies Stack, starting from the more general (abstract) layers, down to the more specific ontological extensions. Although I have contributed to the coordination and design of most of the ontologies, in this thesis I focused mainly on NAO, having taken the lead for its design and development. NAO represents basic attributes of entities on the semantic desktop (defined at both application and user-level) that go beyond the essential information defining them. NAO thus provides for the explicit semantic representations of common operations that take place on the desktop, abstract concepts, and their relationships to physical desktop entities.

Although not covered in this thesis per se, the NIE Ontologies provide adequate semantic representation of legacy desktop data, including existing structures, information elements and their relationships, whereas the PIMO Ontology provides modelling primitives that reflect the desktop user's mental models and usage habits. This modelling was applied throughout and lifted up to the highest (abstract) levels of desktop knowledge representation.

Aligning knowledge representation on the SSD with the general Semantic Web approach enables the comprehensive use of data and schemas and transforms the desktop into an active, personalised access point to the Semantic Web. In this scenario, ontologies from the most general to the personal vocabularies, play a crucial role. The development of the resulting ontologies followed an in-depth review of existing vocabularies, including

representational languages like RDF/S and OWL. Although the requirements for the SSD meant that these could not be directly reused, our approach strives to re-use and extend existing concepts and attributes, in the spirit of the Semantic Web's vision for networked knowledge.

Our approach to knowledge representation is extensible and the vocabularies have been designed with standardisation in mind. While originally designed as internal standards for the SSD, the arguments behind our approach also hold true for the Semantic Web at large, especially in the light of the current trends which increasingly show a shift from the view of the “Semantic Web as one big, global knowledge base” to “a Web of (machine and human) actors” with local perspectives and social requirements such as provenance and trust technology will be discussed in Section 11.3. Section 11.1 will then explain how the knowledge representation schemes presented in this Chapter have been provided as standards, and how their re-use and extension are being promoted.

7 A Conceptual Framework for Semantic E-mail

The social aspect of the Social Semantic Desktop is very much dependent on data exchange and communication. One of the key requirements for this collaboration infrastructure is for communication and data exchange channels in between desktops to support the transport of metadata alongside data. This ensures semantic interoperability, which is one of the four fundamental requirements for the SSD (Section 3.2.1). The extension of communication technologies to enable them to send semantic messages in between SSDs was first suggested in [Sauer mann et al., 2005]. The realisation of *Semantic E-mail* is also intended to address the (inter)personal information management issues arising from email use, as discussed in Chapter 4. Thus Semantic Email serves two parallel, but complementary purposes, i.e.:

- Support email-based collaboration by addressing email model limitations
- Fulfil the needs of a semantically-enabled communication technology for knowledge exchange and integration on, and across, the SSD platform.

This Chapter will reintroduce the concept of Semantic E-mail in the context of this thesis, as opposed to earlier approaches (refer to Section 4.5.3). Given the two above purposes, the presented vision for Semantic E-mail draws significantly from the requirements outlined for the SSD (Section 3.2) and the related work presented in Chapter 4, especially the conclusions of their survey in Section 4.6.

This approach attempts to structure email communication, not only at the email thread level, but also within individual messages. The approach is comparable to related

work presented in Section 4.5, which also attempted to give a better structure and semantics to the email model by eliciting specific concepts (e.g. tasks) and patterns (e.g. reply to's). As in this other work, the aim is to enable the separate management of email action items. Subsequently, the management of entire email workflows is also targeted.

Sound theoretical foundations for representing semantic email were required to carry on with this task. The *sMail Conceptual Framework* was designed for this purpose, and it encapsulates all the resulting concepts and models. The framework focuses on *Speech Acts* as the basic units of knowledge for structuring e-mail communication (Section 5.2). The first challenge was therefore the selection of a speech act-based system of categorisation that is appropriate for the task at hand. Although some of the existing taxonomies under comparison in Section 5.1 were close to providing a required system of categorisation, these were still not deemed satisfactory. The first section in this Chapter will present a newly-designed speech act model that is used as a basis for structuring e-mail. The results of its evaluation suggest that it outperforms the state-of-the-art with respect to expressivity.

The next step was to investigate the nature and strength of relationships between successive, related speech acts in email threads. A study in email speech act sequentiality led to the design of a speech act-based email workflow model, introduced in the second section of this Chapter. This model is considered as a formal representation of the ad-hoc workflows taking place within email communication.

To enable machines to work with these models, as well as to integrate this knowledge on the SSD, a domain ontology that represents all elicited concepts, patterns and relationships was engineered; introduced in the third section.

7.1 Modeling Email Speech Acts

Of the surveyed speech act models, the one that best served our purpose was that used by Carvalho & Cohen in their attempt to trace intents behind email messages [Carvalho and Cohen, 2006]. In brief, this model represents hierarchies of *Verbs* and

Nouns, whose conjunction forms a speech act as a pair: (v, n) . The verbs, or *Actions* as we prefer to call them, are organised by way of two of Searle’s speech act categories (Commissives and Directives). The nouns, or *Objects* (of the action) as we prefer to call them, are organised at the highest levels into *Activities* (including ongoing and upcoming events) and *Deliveries* (of opinions or data).

Apart from the addition of a new useful parameter to the two above, our differences to Carvalho & Cohen’s model lies mostly in the hierarchical organisation.

The rest of this section will start with the presentation of the resulting speech act model, which first appeared in [Scerri, 2007], followed by the results of its comparative evaluation and formal representation.

7.1.1 The Speech Act Model

Although we also focus on commissive, directive, as well as assertive speech acts, our action hierarchy (Fig. 7.1) is more discourse oriented. It differs between different *Discourse Roles*, whereby an action can double for one or more such roles. These roles are differentiated mostly with respect to the expectations of their sender, to reflect the sender’s real intent. The seven ensuing actions serve multiple roles in different contexts, represented by multiple parent nodes in the graph in Fig. 7.1. At the highest conceptualisation level, the model consists of two categories: *Initiatives* – actions that are initiating discourse (e.g. a question); and *Continuatives* – actions addressing previous actions (e.g. a reply to an earlier communicative act).

Initiatives are refined into *Requestives* – covering communicative acts that involve directives addressed to the recipient(s), and to which the latter is expected to provide some form of reply; and *Informatives* – covering acts that involve assertives requiring no further communication from the recipient’s end. In addition, *Imperatives* cover acts that are both requestive and informative since their behaviour corresponds to both definitions above. Handing out an order, for example, entails a directive in the sense that something is being asked out of the recipient, but as replying to it is optional, it can also be considered

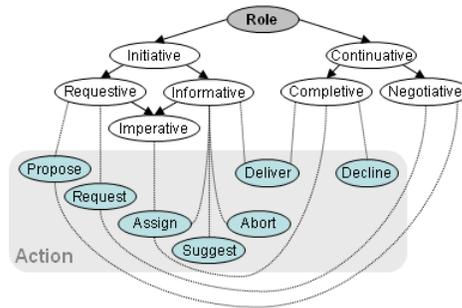


Figure 7.1: Speech Act Actions, organised by Discourse Roles

as an informative. Continuatives are refined into *Completives* – covering communicative acts that satisfy a former request; and *Negotiatives* – for acts which are also part of (but do not complete) an ongoing communication process where an exchange of further acts is expected. Deriving from the given roles, the model provides for the following seven actions:

- Request – this action entails a directive request which sees the email sender, in the role of a speaker, asking for something (e.g. file, information, permission for an event/task, the recipient(s) attendance to a (joint) event/task, etc.). The pragmatics (intent and purpose) of a request are such that it will remain pending until answered via a successive communicative act. Requests can have an initiative communicative role, i.e. it is not succeeding an earlier one; or a negotiative role when responding to an earlier request with some modified characteristics (e.g. negotiating time or place of a requested meeting).
- Propose – this action also entails a directive request. However it is only relevant to activities having multiple recipients, whereby a consensus between all participants is required before a further action is considered (e.g. event scheduling and task discussions). The pragmatics of a Propose are similar to that for a request, with the difference that the action will remain pending until all of its recipients provide a successive communicative act. This action can have both initiative and negotiative

roles. The exact implications of a Propose in comparison to a Request will be clarified in Section ??.

- Assign – entails assertives, or strong directives, that commit either the sender or the recipient(s) to some activity (e.g. orders, personal activity announcements). The pragmatics of an assignment are such that no successive act from the recipient is required. Instead, the undertaking of the activity in question is immediately assumed. This action has an imperative role when the recipient is involved in the activity, or informative if only the sender is involved. Additionally, an assign action might be in response to other communicative acts (e.g. meeting permission request) and can thus have a completive role.
- Suggest – entails assertives, or weak directives, that can potentially result in the commitment of either the sender or the recipient(s) to some activity (e.g. volunteering, suggesting someone’s participation in an activity). These actions are comparable to Assign actions with unassumed consequences. As further communicative acts are not expected, their role is always informative.
- Deliver – entails assertives that relay some kind of information (e.g. special notifications, announcements) or email-specific information deliveries (i.e. attachments). As successive communicative acts are not expected, this action can take the role of an informative. However, when it is in reply to a request action (e.g. a request for data), it’s role is completive.
- Decline – this action rejects a preceding directive communicative act (e.g. declining requests). As this action can only take place following a preceding communicative act and no further acts are expected, it can only manifest a completive role.
- Abort – This action cancels a pre-established activity (e.g. cancelled meeting notification). As this action does not need to follow a preceding communicative act and no further acts are expected, its only possible role is informative.

The actions above can concern different things. Whereas some are related to activities, others involve only the exchange of information. To further represent communicative acts, a second parameter was introduced to the model, as in [Carvalho and Cohen, 2006], to represent specific *objects* of the above actions. Objects (Fig. 7.2) are categorised into two major concepts: *Data*, representing something which occurs strictly within the boundary of Email (e.g. information); and *Activity*, representing something that is either conceptually or physically external to Email (e.g. carrying out a task, attending an event). Possible data objects include:

- Information – Represents textual data corresponding to opinions and facts.
- Resource – Represents non-textual data, e.g. files or URIs.
- Feedback – Introduced following the modeling of the semantic email workflow model (Section ??).

The Possible activities include:

- Events – covering all email-generated activities taking place at a specific time and (virtual or non-virtual) place, e.g. a meeting, a talk, conference, social event. Typically these items are stored in a Calendaring application.
- Tasks – covering all email-generated tasks which might need to be performed within a specific time span. Typically these items are stored in a Task Manager. Although there can be an overlap between the two, ultimately it's the user who decides whether an activity constitutes an event or a task.

While scrutinising the eventual speech act instances in existing speech act models such as [Carvalho and Cohen, 2006], it was noted that the complexity of speech acts relevant for the given task can be further reduced by considering a third parameter – i.e. the *Subject* of a speech act. Some of the previous taxonomies differed, at the object level, between a speech act requesting permission from the recipient to attend an event (i.e.

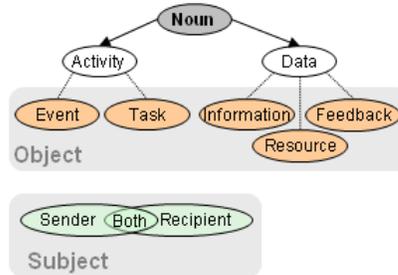


Figure 7.2: Speech Act Objects and Subjects

Request for Permission) and another requesting the recipient to attend (i.e. Request for Event). In reality however, these speech acts share both the action and the object, i.e. Request Event. What they differ in is who is actually being implied in, or bound to, the event being requested: the *sender* in the former and the *recipient* in the latter. Another example is announcing one’s assumed responsibility for a task, versus assigning the task to another person. In both cases, a task is being assigned to a subject: the sender in the first case and the recipient in the second. Speech acts can also have *both* sender and recipient(s) tied to the activity, as is the case for joint event requests or task announcements. These three options make for instances of the subject parameter, as included in the speech act model designed for this thesis (Fig. 7.2). Thus as opposed to earlier work, in our model a speech act is modelled as the triple: (a, o, s) . The additional third subject parameter makes for less actions (e.g. Permit and Announce and Amend become redundant) and objects (e.g. Permission) and thus reduces the complexity of speech act representations. Given its purpose, the subject parameter is only applicable to speech acts with activity-noun objects (i.e. tasks and events).

Fig. 7.3 depicts the 38 speech act instances resulting from valid combinations of the three parameters of the speech act model. The two object categories (Fig. 7.2) have been included for a better presentation. Specific descriptions for each is given on the right-hand side. Furthermore, the sender’s *Expected Action* on sending (SEA) and the recipient(s)’s *Expected Reaction* on receipt (RER), are included for each. These two characteristics are

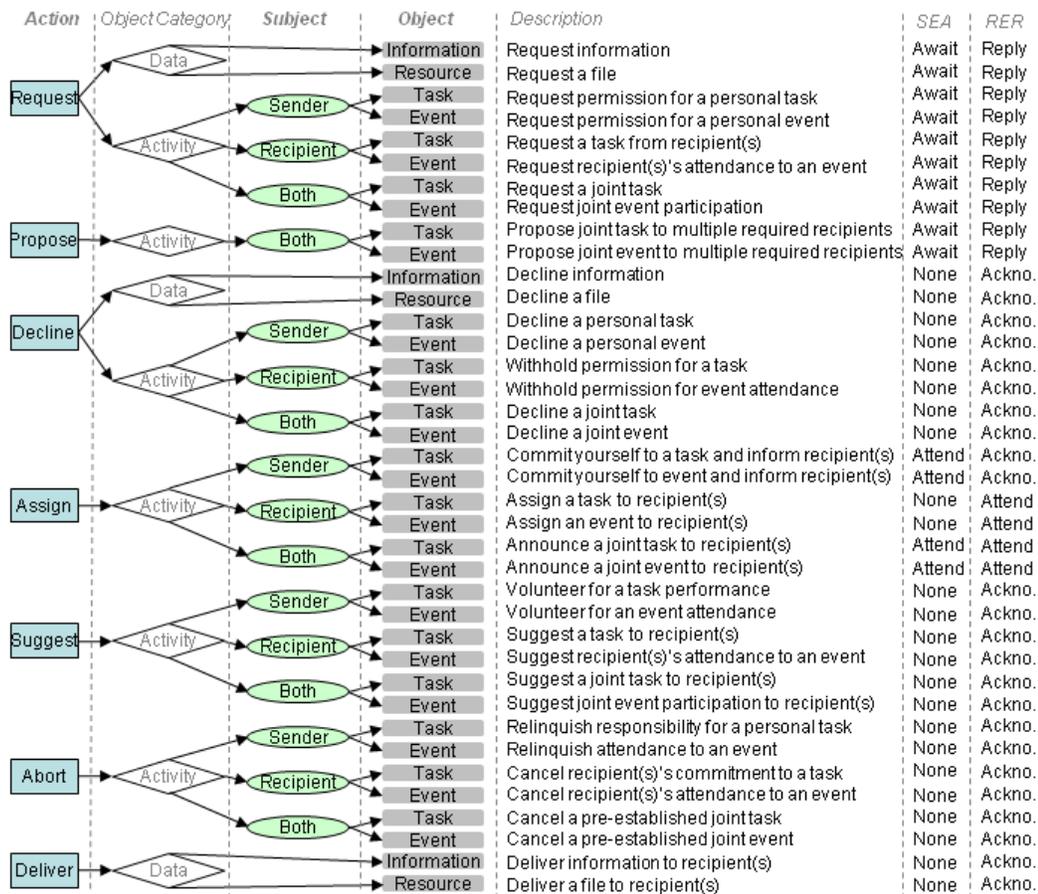


Figure 7.3: Instances of the Speech Act Model

very relevant to the concept of speech act satisfiability, as introduced in Section 5.2.4 and elaborated on in Section 7.1.3. Based on the pragmatical nature of the speech act, there are 5 different expectations:

- Await – This mode is specific to the speech act sender. After sending out some speech acts, the sender will be awaiting a reply, e.g. Questions, Proposals, File Requests, etc. Pending this reply, the speech acts will also remain pending.
- None – This mode is also specific to the sender of a speech act. The very act of sending out some speech acts is all that is required on the sender's side, e.g.

Notifications, Suggestions, Handing out orders, etc.

- Acknowledge – This mode is the equivalent of the former, but concerns the recipient. When receiving some speech acts, the only expected reaction from the recipient is to acknowledge their receipt.
- Reply – This mode is only specific to speech act recipients, and is bound to speech acts where the sender will go on await mode. Thus the expected reaction from the recipient is to reply.
- Attend – This action is applicable to both speech act sender and recipient. It involves the performance of an activity that is external to email by the sender, on sending a speech act, and/or the recipient, on receiving a speech act. E.g. attending to a task or to a joint event after committing or approving it, etc.

An aspect of the speech act model that is not represented in Fig. 7.3 is their flexibility with respect to the discourse role they are performing. In fact, the descriptions, as shown in Fig. 7.3, for the speech acts that can serve multiple roles, can vary slightly. For example, used in reply to a request event with the sender as subject, i.e. a (*Request, Event, Sender*) (hence an event permission request), the best way to describe an ensuing (*Assign, Event, Recipient*) (the recipient being the former sender) would be as a ‘Permit recipient to attend an event’ rather than the corresponding description in the figure above. Another example is treating an (*Assign, Task, Sender*) as an *approval* of an earlier request, i.e. (*Request, Task, Recipient*), and as a personal task *announcement* otherwise. Yet another example is replying to an event request (any subject) via another event request (same subject). In this case a better description for this speech act would be ‘Amend event’. For these two particular cases, previous work required the specific definition of *Amend* and *Approve* actions. Thus the possibility of actions to perform under different roles also reduces the complexity of the representation model.

7.1.2 Evaluation and Improvements

An evaluation was carried out to determine whether the designed speech act model, enhanced with a third parameter, offered a better representation of speech acts in e-mail than the state-of-the-art, which at the time was best represented by Carvalho and Cohen’s model (henceforth referred to as the C&C model). This section is based on the evaluation reported in [Scerri et al., 2008c].

7.1.2.1 Methodology

An appropriate statistical method serving the following two purposes was required:

1. To measure the speech act model’s goodness of fit when applied to real data.
2. To compare this measure with that for the C&C model.

Such a measure could be obtained by calculating the inter-annotator agreement between human annotators, annotating segments of a corpus of email messages with one or more speech acts. Of the available methodologies we chose the Kappa coefficient, introduced by J. Cohen [Cohen, 1960], to measure chance-corrected nominal scale agreement between two raters. The κ statistic may be computed as:

$$\kappa = \frac{P(A) - P(E)}{1 - P(E)}$$

where κ is the Kappa coefficient, $P(A)$ is the total agreement, and $P(E)$ is the percentage of agreement which occurs by chance alone. The value of Kappa ranges from -1 to +1, with 1 signifying complete agreement. The use of this statistic in computational linguistics and cognitive science to measure inter-annotator agreement in discourse and dialogue work was proposed by [Carletta, 1996]. The statistic has received a lot of criticism and a number of problems in its application have been exposed. Since then, various extensions and generalisations of the coefficient have been proposed [M. and W.C., 1987]. The standard κ statistic was however deemed appropriate for the evaluation task at hand.

Despite that the same statistic was used to evaluate the C&C model, this could not be directly compared to the one being measured, as the email corpus in use was different and was not available. Therefore two local annotators were instructed to carry out two separate annotation experiments, using the same corpus, to calculate κ for both the devised model and the earlier C&C model. The selected subjects were one male and one female postgraduate student, with a computer science and linguistics backgrounds respectively. The corpus for the experiment was comprised of a random selection of 50 email threads from the Enron corpus³³ which discussed social, academic, and corporate issues. The random selection was subject to one constraint, i.e. that the number of messages in a thread should not exceed 8. At 3.5 messages per thread, the number of messages in the resulting 50 threads totalled 174, with each being between 1 and \sim 500 words in length.

The annotation tasks required annotating multiple text segments within an email. The segments were not pre-agreed upon by the annotators and where necessary, it was agreed to assign more than one speech act to a single segment. If one annotator assigned one speech act to a sentence whereas the other assigns none, or multiple ones, the extra annotations were considered as a disagreement. At the time of the evaluation, the speech act model provided four actions (Request, Commit – subsequently renamed to Assign, Deliver and Decline), four objects (Event, Task, Information, Resource) and three subjects (Sender, Recipient, Both). These made for 24 valid speech act combinations, whereas the C&C model only had 16. These two speech act sets were used as categories for calculating two separate κ statistics. The two annotators were instructed to select text segments for annotation exclusively according to the model being evaluated, to avoid any undue influence of one model on the other. Given our emphasis on context retention, the annotation task was thread-oriented in order to facilitate the appropriate assignment of speech acts. Without context, a text segment may be assigned an entirely different speech act than the one otherwise intended. For example, the text “No problem!” out of

³³<http://www.cs.cmu.edu/enron/>

Table 7.1: E-mail text annotated based on the speech act model

Segment	Annotator	Action	Object	Subject
1	A,B	Deliver	Resource	∅
2	A,B	Commit	Task	Recipient
3	A,B	Deliver	Information	∅
4	A	Request	Information	∅
	B	Request	Task	Recipient
5	A,B	Deliver	Information	∅

context, could be assigned no speech act, or just be classified as a (*Deliver, Information, ∅*)³⁴. However if this was in reply to “Lets you and me try and talk today” in an earlier email, it’s more appropriately classified as(*Commit, Event, Both*).

Listing 9.

```

Seg1: "Here is C.'s resume."
Seg2: "We would appreciate any help you could give him."
Seg3: "He is available to come by and meet anyone you would
think appropriate in the intern process."
Seg4: "Please advise us what we should do next."
Seg5: "He's here for experience but very interested in any
prospects you might have at Enron."

```

Table 7.1 shows an example of a speech act annotation, based on the presented speech act model, for an email after greetings and signatures where removed. Both annotators considered the five separate segments in Listing 9 above. They agreed on the assigned speech acts in all but the fourth case, which shows a disagreement between a (*Request, Information, ∅*) and a (*Request, Task, Recipient*). This data was compiled for all email messages, and for both models under review, in order to calculate an inter-annotator agreement and study the main causes for disagreement.

³⁴∅ serves as a place holder for the subject parameter in speech acts with data category objects

7.1.2.2 Results and Observations

The total number of annotations produced by each annotator differed slightly for both models. Where the annotators produced different amounts of annotations for a text segment, each pair was considered for (dis)agreement. The resulting number of pairs considered for the speech act model was 419, whereas 444 were considered for the C&C model. The larger amount of annotations for the C&C model can be attributed to the fact that its Deliver-Opinion and Deliver-Data categories are roughly equivalent to the single (*Deliver, Information, \emptyset*) speech acts in our model. Thus a delivery of some information followed by an opinion were regarded as one annotation, as opposed to two in the C&C model.

The value of κ for each model was calculated with regards to the full number of parameters and categories, as well as with some categories merged or omitted, in order to be able to make more accurate observations. The κ for the speech act model was calculated at 0.811, in contrast to a κ of 0.75 for the C&C model. The latter compares well with their earlier inter-annotator agreement experiment [Carvalho and Cohen, 2005], which gave a value of between 0.72 and 0.85. To gather more insight into the causes for disagreement, and make better comparisons with the C&C model, we decided to calculate further κ 's for both models. These were recomputed after the following considerations (results summarized in Table 2):

1. C&C (Deliver speech acts merged): As the speech act model's (*Deliver, Information, \emptyset*) is equivalent to both the C&C Deliver-Data and Deliver-Opinion speech acts, comparing the above κ 's might be considered unfair. Thus we recomputed κ after having merged these two C&C categories. Although the result, at 0.83, is comparable to ours, it is still satisfactory as it means that the introduction of a third parameter in the model to improve the knowledge representation's expressivity did not affect the inter-annotator agreement.
2. sMail (Object parameters merged): Disregarding the object parameter in the speech

act model, in order to examine its effect on κ . The result suggests some difficulty in deciding whether a speech act concerns an event or a task.

3. sMail (Subject parameters merged): Disregarding the subject parameter in the speech act model for the same reason. In this case κ shows a negligible improvement, suggesting that the parameter’s introduction was successful.
4. C&C (Assertives omitted): All speech acts in the C&C model have a clear expected action for the user (reply or perform) except for Deliver-Data and Deliver-Opinion, which require simple acknowledgment and constitute 57.1% of the total annotations. In the context of this thesis, directives and commissives have a heavier weight than assertives. Thus these two categories were here omitted, obtaining a κ of just 0.511.
5. sMail (Assertives omitted): Similarly (*Deliver, Information, \emptyset*), accounting for 57.2% of total annotations, was disregarded for the speech act model. A κ of 0.623 demonstrates significant improvement over its equivalent in test 4 above.

In order to improve the speech act model, special attention was given to the causes for disagreement. For this purpose, a confusion matrix showing disagreements between the two annotators’ speech act assignments was computed (Fig. 7.4). The matrix highlights the most significant counts (Outlined: 5), disagreements attributed to the speech act subject (Dark Gray: 2), and disagreements attributed to the speech act object. In turn, the latter are classified into disagreement on the type of activity, i.e. task versus event (Gray: 4) and disagreement on the type of data, i.e. information versus resource (Shaded Gray: 7) noun objects.

(*Commit, Task, Recipient*) and (*Request, Information, \emptyset*) tops the list of pairs in high disagreement with 14 counts. The rough distinction between these two is that the first expects the recipient to perform something, whereas the second expects the recipient to send information back to the sender. However there are cases where they are both applicable e.g., “Please go through the list and determine if you want the following information”. The second highest disagreeing pair: (*Commit, Task, Recipient*) versus

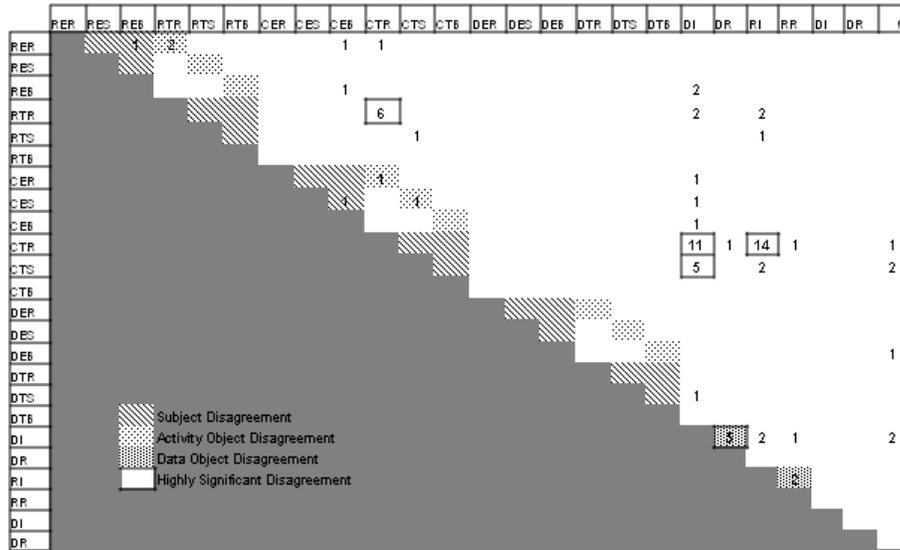


Figure 7.4: Confusion matrix for annotation disagreement in the speech act model

(*Deliver, Information, Ø*); with 11 counts, is attributed to conditional statements e.g. “If we can hold off until next week to set the new offices up I’m sure that D. and D. would be very grateful!” and non-binding statements e.g. “I recommend that you visit with M.T. on this.” In both these cases, the commissive force is too weak for a (*Commit, Task, Recipient*) and thus results in one annotator considering it as a (*Deliver, Information, Ø*). (*Commit, Task, Recipient*) has also the third most significant disagreement, at 6, with (*Request, Task, Recipient*) and this is also attributed to cases where both speech acts apply, e.g. “Please look at the attachment and inform me if you believe we may be perceived in an advisory style role”. The fourth highest disagreeing pair was also related to the Commit action, making it the most ambiguous. There were 5 disagreements for (*Commit, Task, Sender*) versus (*Deliver, Information, Ø*). Most of these are also attributed to conditional or non-binding statements, e.g. “I dont mind helping out - let me know if you need a hand.”. This example falls short of being a personal commitment by the sender. After disambiguating some definitions with the annotators, they were asked to re-annotate the disagreements for the five highest ranking disagreeing category pairs

(numbering 41). The resulting κ for these annotations was 0.609. Although this is a big improvement from the total disagreement of the first round, it shows that some ambiguities cannot be resolved even upon discussion. Some other issues that were not apparent in the results were also brought forward and taken into account in the improvement of the speech act model.

7.1.2.3 Improvements

After examining the obtained results, some modifications were deemed necessary. The most important change was to include *Suggest* as an additional action, to cater for all weak-commissive, conditional and non-binding statements, which were the cause of a large proportion of the disagreements. “I recommend that you visit with M.T. on this.” can now be treated as a *(Suggest, Task, Recipient)*, “ok. I don’t mind helping out - let me know if you need a hand.” as a *(Suggest, Task, Sender)* and “If we can hold off until next week to set the new offices up I’m sure that D. and D. would be very grateful!” as a *(Suggest, Task, Both)*.

Another added action is *Abort*. Although occurrences of this action were very low, it is included for completeness, as the envisaged semantic email client should also support users with cancelling pre-established activities. Other superficial changes include renaming the *Commit* action into *Assign*, as this proved more intuitive to the annotators. The *Propose* action and the associated *Feedback* object were only added at a later stage, in order to accommodate the speech act workflow model introduced in Section ??.

After fine-tuning the model, it was immediately re-evaluated by way of instructing the annotators to reconsider the five highest-disagreeing categories with the adjusted model in mind. The result was a κ of 0.732, significantly higher than the 0.609 achieved after the attempted disagreement resolution. In conclusion, results prove that the speech act model constitutes an improvement over earlier attempts tackling the classification of speech acts within digital text-based communication.

7.1.3 Defining Semantic E-mail in terms of Speech Acts

The exchange of a speech act in a semantic email constitutes a *Speech Act Process*. I will now provide more formal definitions of what constitutes a semantic e-mail in terms of these processes, starting from the following definition of a speech act.

Definition 3. A *Speech Act* sa from the sender s to a recipient r is defined as follows:

$$sa = (a, o, S) \text{ where:}$$

- action $a \in \{Request, Commit, Propose, Suggest, Deliver, Abort, Decline\}$
- object $o \in Data \cup Activity$ where:

$$Data = \{Information, Resource, Feedback\}$$

$$Activity = \{Event, Task\}$$

- subjects S s.t.

$$S \subseteq \{s, r\}, S \neq \emptyset \quad \text{if } o \in Activity$$

$$S = \emptyset \quad \text{otherwise}$$

That is, a speech act is a triple consisting of an action, an object and a set of subjects if the object is of activity subtype; in which case the subject is either the sender, the recipient, or both. I will now define the characteristics of speech acts, namely their various roles, the sender's expected actions on sending a speech act (SEA) and the recipient's expected reaction on receiving it (RER).

Definition 4. If SA is the set of all speech acts then:

- *role* is a function from SA to the set

$$roles = \{Requestive, Imperative, Informative, Negotiative, Compleitive\}$$

- *sea* is a function from SA to the set

$$actions = \{Await, None, Attend\}$$

- *rer* is a function from SA to the set

$$reactions = \{Reply, Acknowledge, Attend\}$$

Speech acts are exchanged over email. When an email containing speech acts is sent, a speech act process is initiated for each, as follows:

Definition 5. *The exchange of a speech act sa in an e-mail sent by a sender s to a set of recipients R , with $|R|=n$, initiates a **Speech Act Process**:*

$$sa \xrightarrow{i} sap \text{ where:}$$

- *speech act process $sap = (sa, T)$ where:*

- *$sa =$ speech act initiating the process*

- *$T =$ set of targets s.t.*

$$T = S \cap R \quad \text{if } ((S \neq \emptyset) \text{ and } (S \neq \{s\}))$$

$$T \subseteq R, T \neq \emptyset \quad \text{otherwise}$$

That is, a speech act process is augmented with a set of speech act *targets* – these being the specific contacts to whom it is addressed. This reflects email’s multi-recipient modelling, such that different speech acts can be addressed to different recipients (e.g. an information request to one of the recipients and a joint task request to all recipients can be exchanged within the same email). The definition of the speech act targets reveals that in the case of speech acts having a subject (i.e. those having an activity object e.g. task or event requests/assignments), the speech act target coincides with the speech act subject(s), excluding the e-mail sender. If the sender is the sole speech act subject, or if the speech act has a data object, the target can instead constitute any of the recipients.

The concept of a speech act process here differs in definition than that for a semantic email process provided by McDowell et. al. [Mcdowell et al., 2004] (see Section 4.5.3), which entailed a continuous process spanning an entire email thread. In this thesis, an independent speech act process is initiated with *each* exchanged speech act, whose state and completion may depend on successive speech act processes. The state of a speech act process is defined as the following function:

Definition 6. *state is a function from SAP to the set States where:*

- $States = \{completed, pending\}$
- $\forall sap \in SAP.$

$$state(sap) = pending$$

Thus, a speech act process can have two states – completed or pending, and by default all speech act processes are initially set to pending.

I will next define a semantic email in terms of the speech act processes it entails.

Definition 7. Semantic E-mail $m = (s, R, SAP)$ where:

- sender s
- $R =$ set of recipients r_j
- $SAP =$ set of entailed speech act processes sap_x s.t. $SAP \subseteq SA \times 2^{\bar{T}}$ where:
 - $SA =$ set of embedded speech acts $sa_x = (a_x, o_x, S_x)$ with $|SA| = z$
 - $2^{\bar{T}} =$ power set of union \bar{T} of all target sets T_x in each process sap_x s.t.

$$\forall sap_x \in SAP, sap_x = (sa_x, T_x). \bar{T} = \bigcup T_x$$

When a semantic email is exchanged, a semantic email transaction is generated for each of the included recipients. These are defined as follows:

Definition 8. A semantic e-mail m generates n Semantic E-mail Transactions mt_j for each e-mail recipient r_j . Each transaction carries a subset of the speech act processes initiated by m , specifically those that are relevant to the current recipient, i.e. the recipient is a member of the speech act target set. The generation of transactions from m is denoted as follows:

$$m \xrightarrow{g} MT \text{ s.t.}$$

- $|MT| = |R| = n.$
- For $j = 1..n$, $mt_j = (s, r_j, SAPt_j)$ where:
 - sender s
 - recipient r_j

- *speech act processes bound to transaction* $SAPt_j \subseteq SA \times 2^{\bar{T}}$ *s.t.*
 - $\forall T_x \in \bar{T}. r_j \in T_x$
 - $\forall sa_x \in SA. sa_x \xrightarrow{i} sap_x$ *where* $sap_x = (sa_x, T_x), r_j \in T_x$

Additionally, when a semantic email is sent, a number of instances are generated for each of the entailed speech act processes, with the number of instances corresponding to the number of targets defined for each process.

Definition 9. $|T_x|=l$ **Speech Act Process Instances** are generated for each speech act process $sap_x \in SAP$. Each of these l instances needs to belong to one of the speech act process sets $SAPt_j$ within exactly one semantic e-mail transaction mt_j :

$$sap_x \xrightarrow{g} SAP_x \text{ s.t.}$$

- $|SAP_x| = |T_x| = l$
- For $p = 1..l$, $sap_{xp} \in SAP_x$ where $sap_{xp} = sap_x = (sa_x, T_x)$
- $\forall sap_{xp} \in SAP_x. \exists! mt_j \in MT \text{ s.t. } sap_{xp} \in SAPt_j$

In order to facilitate the identification of speech act process instances within their rightful transaction, I provide for the following alternative notation.

Definition 10. An individual instance sap_{xp} of speech act process sap_x within transaction mt_j is referred to as $sap_{x,j}$. Thus the set of speech act processes SAP entailed by a semantic e-mail m can be defined in terms of all the respective generated process instances $sap_{x,j}$ within all generated e-mail transactions mt_j as follows:

$$\begin{aligned}
 SAP &= [sap_{x,j}]_{x=1,\dots,z;j=1,\dots,n} \text{ s.t.} \\
 sap_{x,j} &= \{sap_{xp}\} \quad \text{if } r_j \in T_x \\
 sap_{x,j} &= \emptyset \quad \text{otherwise}
 \end{aligned}$$

The non-empty entities for this $z \times n$ matrix consist of a set of zero or one speech act process instances generated for each of the x speech act processes in SAP in m . An entity

is empty if the recipient r_j of the current mt_j is not a member of the target set T_x for the current speech act process sap_x .

To help digest these definitions I will now give a concrete example of a semantic email and its initiated speech act processes. Fig.7.5 provides for its graphical representation, with the left-hand side (a) providing a more visual representation and the right-hand side (b) tying in more with the actual definitions via more formal graphical notations. This example will be extended in later sections.

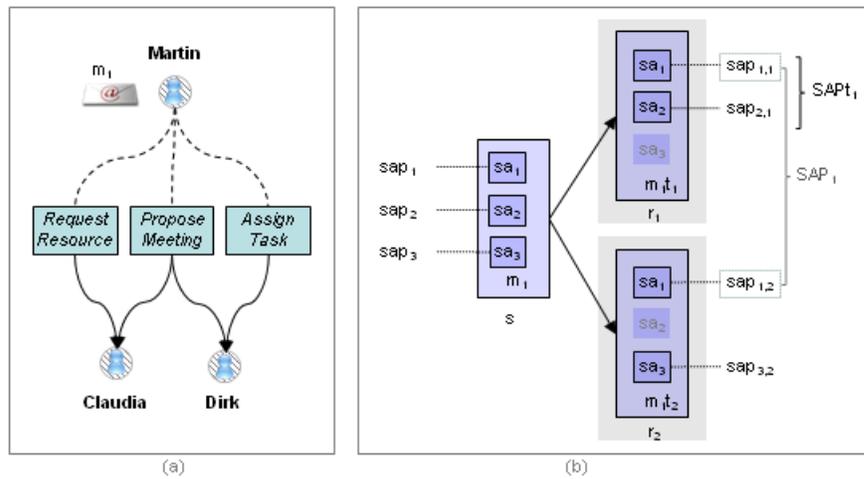


Figure 7.5: A Semantic E-mail from Martin to Claudia and Dirk

Martin, in the role of the sender s , sends a semantic e-mail m_1 to Claudia (r_1) and Dirk (r_1). m_1 contains three speech acts – a proposal for a meeting between all three and addressed to both recipients: $sa_1 = (Propose, Meeting, \{s, r_1, r_2\})$; a request for a resource addressed to Claudia: $sa_2 = (Request, Resource, \emptyset)$; and a task assignment addressed to Dirk: $sa_3 = (Assign, Task, \{r_2\})$. Fig 7.5b shows how when Martin sends the e-mail, three speech act processes (sap_1, sap_2, sap_3) are initiated for each of the speech acts, and two transactions ($m_1 t_1, m_1 t_2$) are generated for each recipient. The speech act processes within the transaction involving Martin and Claudia ($m_1 t_1$) are denoted with $SAPt_1 = \{ (sa_1, \{r_1\}), (sa_2, \{r_1\}) \}$. Similarly (not shown in Fig.7.5b) the transaction between Martin and Dirk ($m_1 t_2$) includes $SAPt_2 = \{ (sa_1, \{r_2\}), (sa_3, \{r_2\}) \}$.

The instances generated for the first process (sap_1) are denoted by $SAP_1 = \{sap_{11}, sap_{12}\}$. Of these two instances, the first is entailed within the first transaction $m_1 t_1$ and is referred to as $sap_{1,1} = sap_{11} = (sa_1, \{r_1\})$, whereas its instance within $m_1 t_2$ is denoted by $sap_{1,2} = sap_{12} = (sa_1, \{r_2\})$ (as shown in the figure). The similarity in notation numbers between the left-hand-side and the middle variable in both of the past two equations is coincidental. In fact, the third speech act process, which generates only one instance $SAP_3 = \{sap_{31}\}$ entailed within the second transaction $m_1 t_2$, is referred to as $sap_{3,2} = sap_{31} = (sa_3, \{r_2\})$.

Recalling the related discussion at the end of Section 5.2.4, I will now define conditions of satisfiability for a semantic email m as a whole, and for each individual speech act process sap_x . For the purpose I will first extend the *state* function defined in Definition 6 above as follows:

Definition 11. *Function state also operates from the following to the set States:*

- $M =$ the set of all semantic email m
- $MT =$ the set of email transactions generated by a semantic e-mail m
- $SAP_x =$ the set of instances generated for process sap_x in a transaction mt
- $SAPt_j =$ the set of process instances entailed by a transaction mt_j

Then, the satisfiability conditions for semantic email m is defined as follows:

Definition 12. *Satisfiability Conditions for semantic e-mail m*

$$\forall m \in M.$$

$$(state(m) = completed) \text{ if } (state(sap_x) = completed \forall sap_x \in SAP)$$

$$\forall sap_x \in SAP \text{ where } sap_x \xrightarrow{g} SAP_x$$

$$(state(sap_x) = completed) \text{ if } (state(sap_{xp}) = completed \forall sap_{xp} \in SAP_x)$$

That is, the state of a semantic email is completed if all associated speech act processes (sap_x) are completed. In turn, a speech act process is completed if the state of all its generated instances (sap_{xp}) is also completed. Alternatively the state of a semantic e-mail

can also be expressed in terms of the speech act processes instances $sap_{x,j}$ generated by that email as follows:

Definition 13. *Satisfiability Conditions for semantic e-mail m (alternative)*

$\forall m \in M.$

$(state(m) = completed)$ if $(state(mt_j) = completed \forall mt_j \in MT)$

$\forall mt_j \in MT.$

$(state(mt_j) = completed)$ if $(state(sap_{x,j}) = completed \forall sap_{x,j} \in SAPt_j)$

That is, the state of a semantic e-mail is completed if the state of each generated semantic e-mail transaction (mt_j) is also completed. In turn, an individual semantic e-mail transaction is completed when the state of each speech act process instance it entails ($sap_{x,j}$) is also completed.

Recall however that each of the non-empty matrix entities $sap_{x,j} \neq \emptyset$ represent exactly one instance of one of the speech act processes initiated by m , i.e. sap_{xp} . Therefore, both of the above-provided conditions of satisfiability, and ultimately the state of both a speech act process and a semantic e-mail, depend on the state of individual speech act process instances within. As shown in Definition 6, this state is by default set to *pending*. I will now define change conditions for the state of $sap_{x,j}$, which depend on the characteristics of the underlying speech act sa_x :

Definition 14. *Change Conditions for a speech act process instance $sap_{x,j}$*

if $role(sa_x) \in \{Informative, Compleitive, Imperative\}$, $rer(sa_x) = \{Acknowledge\}$

then $state(sap_{x,j}) = completed$ if r_j acknowledges speech act sa_x
pending otherwise

if $role(sa_x) \in \{Informative, Compleitive, Imperative\}$, $rer(sa_x) = \{Attend\}$

then $state(sap_{x,j}) = completed$ if r_j attends to the speech act sa_x
pending otherwise

if $role(sa_x) \in \{Requestive, Negotiative\}$

then $state(sap_{x,j}) = completed$ if r_j replies to sa_x with a speech act sa_y denoted:

$$sa_x \xleftarrow{r} sa_y$$

in semantic e-mail m' s.t.

- $m \xleftarrow{r} m'$
- $m' = (r_j, R', SAP')$ where
 - $s \in R'$
 - $SAP' \subseteq SA' \times 2^{\bar{T}'}$ where
 - $\{sa_y\} \in SA'$
 - $s \in \bar{T}'$
- $m' \xrightarrow{g} MT'$ and $\exists mt'_k \in MT'$ s.t.
 - $mt'_k = (r_j, s, SAPt_k)$
 - $mt_j \xleftarrow{r} mt'_k$
- $role(sa_y) \in \{Negotiative, Compleitive\}$
- $sa_y \xrightarrow{i} sap_y$

pending *otherwise*

Therefore, changing the state of a speech act process instance in a semantic email transaction depends on both the role and the recipient's expected reaction of the underlying speech act. For informative, completive and imperative speech acts requiring only an acknowledgment, the corresponding speech act process instance between s and r_j , i.e. $sap_{x,j}$ is completed when r_j flags them as read. For speech acts having the same role but requiring r_j to attend to a task or an event, $sap_{x,j}$ is set to completed when the recipient commits to the activity's performance.

For requestive and negotiative speech acts (whose recipient's expected reaction is always *reply*), the satisfiability conditions require the issuing of a reply speech act sa_y . This generates a corresponding speech act process sap_y , having s as its target, in another semantic e-mail m' . The semantic email transactions generated for m' include (or are

equal to, if $|MT'| = 1$) that exchanged between r_j and s , i.e. mt'_k . The latter is generated in reply to the incoming semantic email transaction mt_j . One last but crucial condition for the completion of $sap_{x,j}$ is that the role of the generated reply speech act needs to be continuative, i.e. either negotiative or completive. It is possible to issue initiative reply speech acts (more about this in the next section), but in this case the state of $sap_{x,j}$ remains unchanged.

7.2 Modeling E-mail Workflows

In Chapter 4 I argue that the lack of support for email workflows is a major contributing factor to Email Overload. Section 4.5.2 deals specifically with related work that considers email as a workflow management system. One of that Chapter's concluding points is that it is possible to elicit well-defined and frequently occurring email workflows. In this Section I will show how the concepts of speech acts, speech act processes, semantic email and email transactions can be used as a basis for defining email workflows as they already occur in standard email use.

As outlined earlier, a speech act process more often than not does not occur in isolation. On the contrary they tend to take place in sequence, with new speech act processes being generated in response to preceding ones. In fact, reply speech acts initiating such processes are in many cases required for the state of a preceding process to change to completed. This thesis considers these elaborate sequences of speech act processes as complex workflows taking place over email, and this section is dedicated to their study, definition and representation.

I will first explain how email can serve as a workflow execution environment. In the second subsection I will then extend the notations presented in the previous section to also cater for email workflows. The third subsection includes the results of a study into speech act sequentiality. This was then taken into consideration in the design of an email workflow model, presented in the last subsection.

7.2.1 Semantic E-mail as a Workflow Execution Environment

Each speech act process can be considered to be either the start, or the continuation of an existing workflow executing over an email thread. The ambition of semantic email is to be able to elicit such process modules to string together workflows and represent them in a standard machine-processable way. I will first illustrate this idea through an extension to the example in Fig. 7.5a, that demonstrates how, while an email thread evolves, workflows within it progress over time.

Recall the email (m_1), carrying three speech acts, which Martin sent to Claudia and Dirk. The first is a resource request addressed to Claudia – “Claudia, can you please forward me the financial report?”, the second is a meeting proposal addressed to both Claudia and Dirk – “can you let me know what day best suits for an urgent meeting re: the financial situation?”, whereas the third is task assignment addressed to Dirk – “Dirk I need you to prepare a report about your group’s spending”.

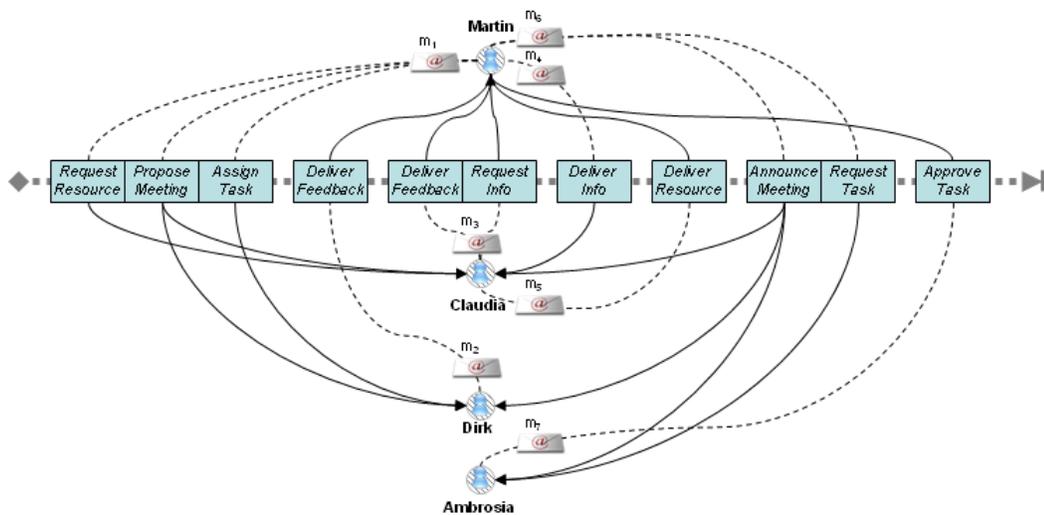


Figure 7.6: An email thread consisting of 7 emails carrying 11 speech acts

When Dirk receives the email he immediately takes note of the task assigned to him. He also creates an email reply to provide his preferred dates and times for the proposed

meeting, i.e. he delivers his ‘feedback’ to Martin – “anytime during working hours this week except wednesday after 3”, by way of an email reply m_2 .

When Claudia receives Martin’s email, she considers the two speech acts addressed to her. She also creates an email reply that includes her feedback regarding her availability for the proposed meeting. She then considers the second speech act, the file request, but before choosing to disclose this file Claudia wants to make sure Martin is aware of the disclosure conditions. Therefore, she replies to Martin’s request by way of another question (information request) – “are you aware of the confidentiality agreements for the report?”. Claudia’s reply email (m_3) is then sent back to Martin with both reply speech acts.

Martin receives both replies, and starts by considering Claudia’s email. He replies to the information request in her email to say that he is aware of the conditions (information delivery) – “I’m aware of the agreement, won’t disclose”, by way of another email (m_4). When Claudia receives this email, she immediately delivers the report to Martin – “Attached” in email m_5 .

Once Martin acquires the report, he decides to organise the meeting on the most appropriate date given the received feedback. After rereading Claudia’s and Dirk’s earlier emails to see their availability, he creates a new email (m_6) to announce the meeting – “Urgent meeting re: financial situation Friday @11am at the director’s office”. Apart from Dirk and Claudia, the announcement is also addressed to Ambrosia, who had requested the meeting in the first place during an office visit. In the same email, Martin also requests a task from Ambrosia – “can you please take care of inviting any other relevant parties?”. When Claudia, Dirk and Ambrosia receive this email, they add the meeting to their calendaring tool. Additionally, Ambrosia also approves Martin’s task request and sends back a confirmation of her approval – “yes no problem”, via m_7 .

By way of this example I have shown how independent sequences of related speech acts can (simultaneously) occur within an e-mail thread. I will now explain, always through the same example, how in this thesis each of these sequences is considered as a

separate email workflow. The speech acts and messages in Fig. 7.6 can in fact be visually organised as to more correctly represent the following four separate but concurrently executing email workflows within.

7.2.1.1 Scenario I: Meeting Scheduling

Fig 7.7a shows the first workflow – meeting scheduling – executing within the example email thread. The workflow is initiated by Martin’s meeting proposal in m_1 , progressed by Claudia and Dirk’s feedback delivery in m_2 and m_3 and by Martin’s meeting announcement to Claudia, Dirk and Ambrosia in m_6 . The workflow terminates when all three recipients acknowledged the meeting. As a product of the workflow, an event was generated for all four participants.

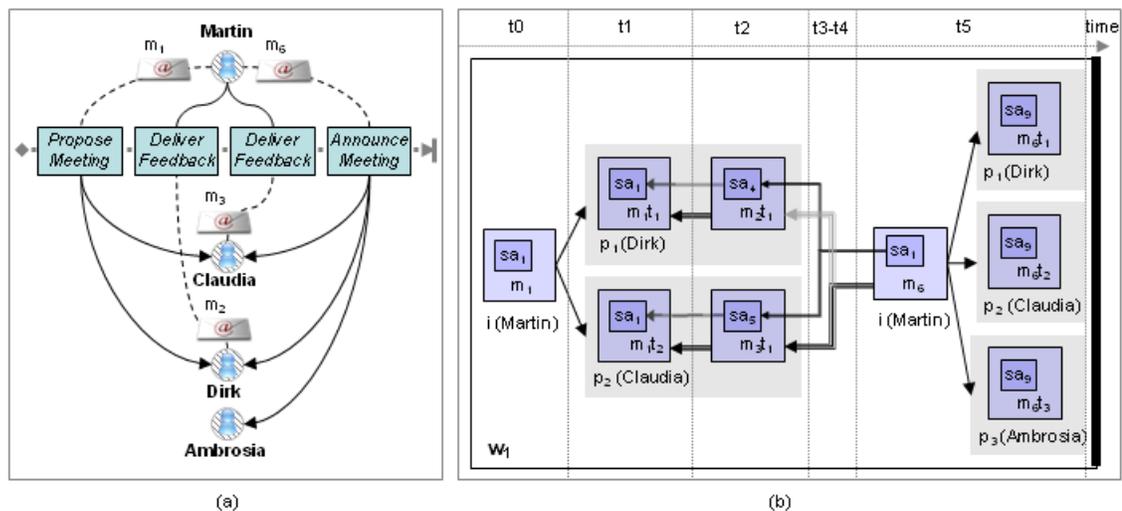


Figure 7.7: Visualisation of the first workflow in the example as per definitions

Fig 7.7b utilises the alternative graphical notations introduced in Fig.7.5b, which are based on the definitions of semantic email, transactions and speech act processes. This representation shows how the workflow – w_1 (outer black frame), can be represented in terms of speech act process instances within multiple email transactions. The workflow starts at t_0 when the workflow *initiator* $i = \text{Martin}$ sends the first e-mail m_1 which

includes $sa_1 = (Propose, Meeting, \{Martin, Dirk, Claudia\})$. Two instances of the initiated speech act process sap_1 are generated within the two transactions m_1t_1 and m_1t_2 for each speech act target, which by default become also workflow *participants*, $p_1 = Dirk$, $p_2 = Claudia$. Then the two instances of sap_1 within m_1t_1 and m_1t_1 respectively are $sap_{1,1}$ and $sap_{1,2}$, both of which are completed with the issuing of two complete reply speech acts – $sa_4 = (Deliver, Feedback, \emptyset)$ from Dirk to Martin in m_2 , and $sa_5 = (Deliver, Feedback, \emptyset)$ from Claudia to Martin in m_3 ; such that $sa_1 \xleftarrow{r} [sa_4, sa_5]$ and $role(sa_4) = role(sa_5) = Complete$. Also, the only transaction generated for m_2 , m_2t_1 is in reply to transaction m_1t_1 , denoted $m_1t_1 \xleftarrow{r} m_2t_1$. Similarly, $m_1t_2 \xleftarrow{r} m_3t_1$.

Between t_1 - t_2 the workflow is split in two parallel sessions – one between Martin and Dirk, the other between Martin and Claudia. When Martin receives both sa_4 in m_2 and sa_5 in m_3 , the two workflow sessions merge back to one, with Martin regaining its control. At time t_5 , i (Martin) resumes the workflow by acting upon both sa_4 and sa_5 and initiating another speech act process for $sa_9 = Assign, Event, \{Martin, Claudia, Dirk, Ambrosia\}$ such that $[sa_4, sa_5] \xleftarrow{r} sa_9$. However, the standard e-mail model only allows an email reply to be generated from exactly one other email, and therefore the email carrying sa_9 is in practice only connected to one of the preceding emails in the workflow, i.e. $m_3 \xleftarrow{r} m_6$. Similarly, the three transactions generated for m_6 can only point back to one earlier transaction in the workflow, in this case $m_3t_1 \xleftarrow{r} [m_6t_1, m_6t_2, m_6t_3]$. Note that at this stage, Ambrosia has also become a workflow participant p_3 . As the recipient's expected reaction for the meeting announcement $rer(sa_9) = Attend$, neither of the three workflow participants need reply to this speech act. Instead, when they acknowledge the generated event (and optionally add it to their calendar), the status of each speech act process instance for sa_9 in each transaction is set to completed. At this point, w_1 terminates (depicted via the vertical black bar at the edge of the workflow frame) as it no longer contains any pending speech act processes.

7.2.1.2 Scenario II: Task Assignment

Fig. 7.8a shows how the second workflow is initiated by Martin’s task assignment to Dirk in m_1 , and terminated shortly afterwards upon Dirk’s receipt and acknowledgment. A task is generated for Dirk as a result of the workflow.

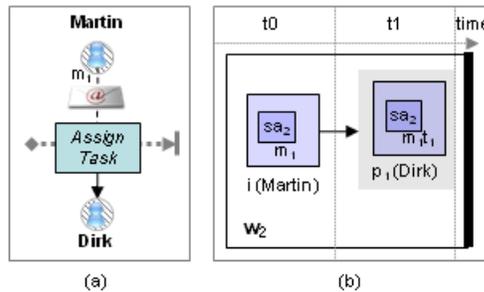


Figure 7.8: Visualisation of the second workflow in the example as per definitions

Fig. 7.8b shows how speech act $sa_2 = Assign, Task, \{Dirk\}$ in m_1 initiates a separate workflow w_2 , with $i = Martin$ and one participant $p_1 = Dirk$. As $rer(sa_2) = Attend$, when Dirk receives the email at t_1 , he only needs to acknowledge (and optionally add the task to a task list) for the speech act process initiated by sa_2 to be set to completed. w_2 thus terminates.

7.2.1.3 Scenario III: File Acquisition

Martin’s first email also included a resource request, which initiates this third workflow, as shown in Fig 7.9a. This workflow is special in that it contains a nested workflow, or a *subworkflow*. The latter consists of the information request sent by Claudia in m_3 , which was in reply to the resource request in m_1 ; and the successive information delivery by Martin in m_4 . This constitutes a subworkflow because although Claudia’s information request was sent in reply to the earlier resource request, it did not change the latter’s state to completed (from pending) as its role is neither negotiative nor complete (refer to satisfiability conditions for speech act process instances in the previous Section). After the

subworkflow's termination, Claudia resumes the parent workflow by sending a resource delivery in m_5 . The workflow terminates upon Martin's receipt of the resource.

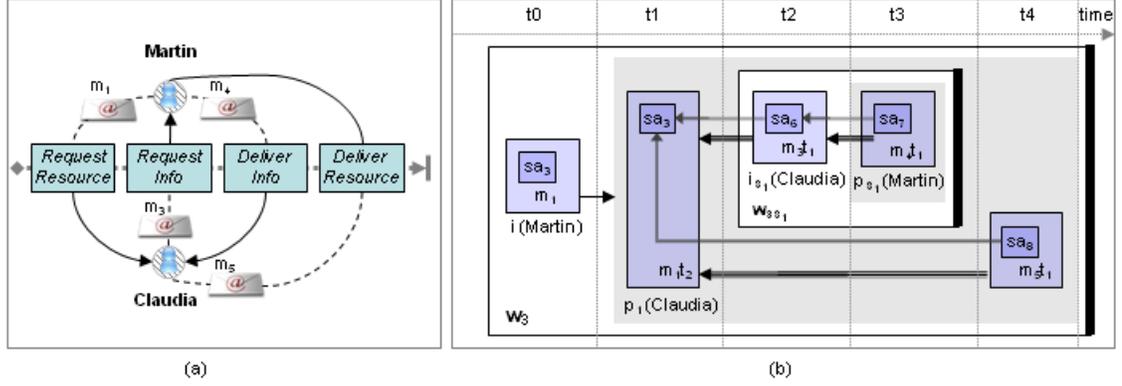


Figure 7.9: Visualisation of the third workflow in the example as per definitions

Fig.7.9b's alternative representation shows how Claudia reacts to $sa_3 = (Request, Resource, \emptyset)$ at time t_2 . Although Claudia reacts to sa_3 by way of $sa_6 = (Request, Information, \emptyset)$ in email m_3 , the state of instance of the speech act process for sa_3 in m_3t_1 remains pending because $role(sa_6) = Requestive$. As a result, sa_6 is considered the start of a subworkflow of w_3 , i.e. w_{3s_1} , with $i_{s_1} = Claudia$ and one participant $p_{s_1} = Martin$. At t_3 , Martin replies to sa_6 via $sa_7 = (Deliver, Information, \emptyset)$ in e-mail m_4 . Because $role(sa_7) = Completive$ and the only transaction carrying sa_6 is m_3t_1 , the state of the speech act process generated for sa_6 is set to completed. Also, as $role(sa_7) = Completive$ and $rer(sa_7) = Acknowledge$, when Claudia acknowledges this speech act, the state of its process is also set to completed. As now all the speech act processes in w_{3s_1} are completed, the subworkflow terminates. At time t_4 , Claudia resumes w_3 by providing the requested resource, represented by a resource delivery speech act $sa_8 = (Deliver, Resource, \emptyset)$ in email m_5 . When Martin receives the file (and thus acknowledges the speech act), w_3 also has no remaining pending speech act processes, and thus it also terminates.

7.2.1.4 Scenario IV: Task Delegation

As shown in Fig.7.10a, this workflow is initiated by Martin’s task request in m_6 . It is then progressed by Ambrosia’s task approval in m_7 , and it terminates upon Martin’s acknowledgment of this approval.

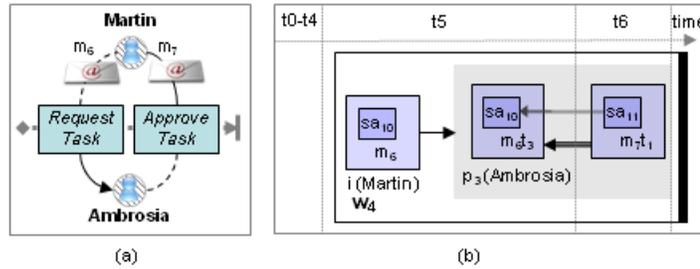


Figure 7.10: Visualisation of the fourth workflow in the example as per definitions

As shown in Fig.7.10b this workflow is initiated at time t_5 by one of the two speech acts exchanged within m_6 , i.e. $sa_{10} = (Request, Task, \{Ambrosia\})$. Although m_6 was sent to three recipients, sa_{10} is only relevant to Ambrosia, who is thus the sole workflow participant p_1 . As $rer(sa_{10}) = Reply$ Ambrosia replies at t_6 with $sa_{11} = (Assign, Task, \{Ambrosia\})$, i.e. a personal commitment to the task, in m_7 addressed back to Martin. This sets the state of the speech act process for sa_{10} to completed. As $role(sa_{11}) = Completive$ and $rer(sa_{11}) = Acknowledge$, its speech act process is set to completed when Martin receives and acknowledges this speech act at time t_6 . At this stage, w_4 terminates as all its speech act processes have completed.

The extensive example presented above demonstrates how fairly well-structured ad-hoc email workflows execute continuously over a user’s email threads. Emails can carry multiple workflows, and as an email thread evolves, additional workflows can be initiated; all of which can regard different participants and some of which produce artefacts such as tasks or events. After explaining what constitutes email workflows in terms of the speech act paradigm by way of graphical examples, I will next provide formal definitions for email workflows and their satisfiability conditions.

7.2.2 Defining E-mail Workflows in terms of Speech Act Processes

The following definitions are based on Definitions 3 -14 in Section 7.1.3. An email workflow is defined in terms of semantic email and speech act processes as follows:

Definition 15. An **E-mail Workflow** $w = (i, P, Mw, SAPw, w_s)$ where:

- $i =$ workflow initiator s.t.

$$i = s_1 = \text{sender of initial semantic e-mail } m_1$$

- $P =$ set of workflow participants s.t.

$$P = \bigcup \{T_x \setminus \{i\} \mid T_x \in \text{sap}_x, \forall \text{sap}_x \in \text{SAP}w\}$$

- $Mw =$ set of exchanged semantic e-mails m within w s.t.

- initial semantic e-mail $m_1 \in M_1$ s.t. $m_1 \xrightarrow{g} MT_1$

- $\forall m' \in Mw, m' \neq m_1.$

- $\exists! m \in Mw$ s.t. $m \xleftarrow{r} m',$

- $m' \xrightarrow{g} MT'$ s.t. $\forall mt'_k \in MT', mt'_k \notin MT_1.$

- $\exists! mt_j \in m$ s.t. $mt_j \xleftarrow{r} mt'_k$

- $\text{SAP}w =$ set of speech act processes entailed by w s.t.

- $\text{SAP}w \subseteq \text{SAP}Mw \subseteq \text{SAM}w \times 2^{\bar{T}Mw}$ where

- $\text{SAP}Mw =$ set of all speech act processes entailed by Mw

- $\text{SAM}w = \bigcup \{SA \mid SA \in m, \forall m \in Mw\}$

- $\bar{T}Mw = \bigcup \{T_v \mid T_v \in \text{sap}_v, \forall \text{sap}_v \in \text{SAP}Mw\}$

- initial speech act process $\text{sap}_1 \in \text{SAP}w$ s.t

- initial speech act $\text{sa}_1 \xrightarrow{i} \text{sap}_1 = (\text{sa}_1, T_1)$

- $\text{role}(\text{sa}_1) \in \{\text{Requestive}, \text{Imperative}, \text{Informative}\}$

- $\text{sap}_1 \in \text{SAP}_1 =$ speech act processes bound to m_1

- $\forall \text{sap}_y \in \text{SAP}w, \text{sa}_y \xrightarrow{i} \text{sap}_y, \text{sap}_y \neq \text{sap}_1.$

- $\exists \text{sap}_x \in \text{SAP}w$ s.t. $\text{sap}_x \xleftarrow{r} \text{sap}_y$

- $w_s =$ set of sub-workflows occurring within w s.t.

- $\text{SAP}w_s =$ set of speech act processes entailed by w_s

- *initial speech act process* $sap_{s1} \in SAPw_s$
- $\exists! sap_x \in SAPw$ s.t. $sap_x \stackrel{r}{\leftarrow} sap_{s1}$
- $sa_{s1} \stackrel{i}{\rightarrow} sap_{s1}$ with $role(sa_{s1}) \notin \{Completive, Negotiative\}$

In other words, an email workflow can be defined in terms of the set of semantic email messages required to execute the entailed sequence of speech act processes, a number of possible sub-workflows originating from within, a workflow initiator and all other participants. The initiator is the sender of the first email m_1 , whereas all other contacts involved become the workflow participants, i.e. all the targets of speech act processes initiated in w , excluding the workflow initiator i .

Bar the first exchanged email m_1 , any additional messages (m') in Mw need to be in-reply-to, or generated as a forwarded email, from exactly one previous message (m) which is also in Mw , leading all the way back to m_1 . Additionally, each individual transaction generated for each of the m' messages (mt'_k) must be preceded by exactly one transaction (mt_j) generated by the preceding email m . This reflects the standard email model, whereby a reply-to or a forwarded message cannot be generated upon multiple messages (i.e. transactions), even if they are all related. However, the underlying generated speech act process can still be in reply to multiple earlier speech act processes that formed part of the same workflow – as shown in Fig. 7.7b.

The workflow itself is generated by the first speech act process (sap_1) in the first email (m_1), which must also have a initiative role (i.e. requestive, imperative or informative). The set of speech act processes relevant to w is defined as a subset of the set of all speech act processes in the associated set of e-mail message Mw , such that, bar sap_1 , all other relevant processes can be traced to an earlier process, leading all the way back to sap_1 . This allows for the fact that workflows as such function independently of the email messages within which they are executed. Thus an email can execute multiple independent workflows, and an email thread can contain multiple speech act process sequences carrying them out.

Workflows can also generate related workflows, which despite executing independently

of the original workflow, are still essential to its completion. I refer to these workflows as subworkflows w_s of w . A subworkflow is generated when a speech act process is initiated in reply to an earlier process sap_x without changing the status of the latter. This is only true for speech acts with initiative roles. If such a speech act was completive or negotiative, this would not constitute a subworkflow but just another speech act process in w . Subworkflows function independently of their parent workflow, although the state of the latter depends on the state of the former. Subworkflows can themselves initiate additional subworkflows, recursively.

I will now also extend Definitions 6,11 in Section 7.1.3 to also cover workflows. Therefore the function *state* also operates from the set of all workflows W to the set *States*. The conditions of satisfaction for w are then defined as follows:

Definition 16. *Satisfiability Conditions for workflow w*

$\forall w \in W.$

$(state(w) = completed)$ if $(state(sap_x) = completed \ \forall \ sap_x \in SAPw)$

$\forall sap_x \in SAPw$ where $sap_x \xrightarrow{g} SAP_x$

$(state(sap_x) = completed)$ if $(state(sap_{xp}) = completed \ \forall \ sap_{xp} \in SAP_x)$

Building on the conditions of satisfiability for a speech act process provided in Definition 12, an email workflow is completed if all speech act processes it entails are completed. Note that w does not require its subworkflows w_s to terminate in order to terminate itself. Therefore, a parent workflow can terminate even if its subworkflows are still executing. This reflects conventional email use, where at any point in time users are still free to continue handling an original speech act (e.g. the resource request in Fig.7.9) without waiting for some other directly-related initiated communicative process (e.g. the information request in Fig.7.9) to be over.

As a final note, the state of an email workflow does not depend on the state of the semantic emails, $m \in Mw$, required for its execution. This is because email m (and its transactions mt_j) can execute multiple workflows by way of carrying multiple speech act

processes sap_x attributed to each. Therefore, although the conditions of satisfiability for a semantic email require the completion of all the speech act processes within, the inverse is not true.

7.2.3 A study into Speech Act Sequentiality

Following the above definitions, a study into the frequency and occurrence of speech act sequences within email workflows was necessary. Similar studies have been carried out within different contexts, e.g. by Jose for natural language speech and by Carvalho and Cohen for email threads (refer to Section 5.2.4). These sequences are also comparable to Winograd and Flores's communicative act networks (also Section 5.2.4).

For this purpose, the inter-annotator agreement experiment presented in Section 7.1.2 was extended so that the annotators did not only highlight speech acts within individual email, but also related sequences of speech acts within email threads. The speech acts considered for this task however, were agreed-upon by the two annotators. The stringing together of email speech acts into workflows was then performed by both annotators at the same time, following discussions. When a disagreement about which sequence constitutes a workflow could not be resolved, both workflow interpretations were marked separately. The annotators reported on the difficulty of this task, particularly because of the way an email sometimes evolves and changes topic, subjects and also recipients. The hardest aspect of the task was to determine whether some speech acts were initiating subworkflows, whether they were a continuation of an existing workflow, or if they were simply independent workflows that happened to initiate in an existing threaded email discussion. Another problem is that although in reality the workflows could have progressed further, at acquisition time, many threads/workflows in the corpus were incomplete. Additionally, some of the less common speech acts were under-represented in the study.

Despite these limitations, the results still provided for some interesting observations, presented in Fig.7.11 as a Bayesian network depicting the observed transitions between speech act sequences in email workflows. This directed acyclic graph represents the

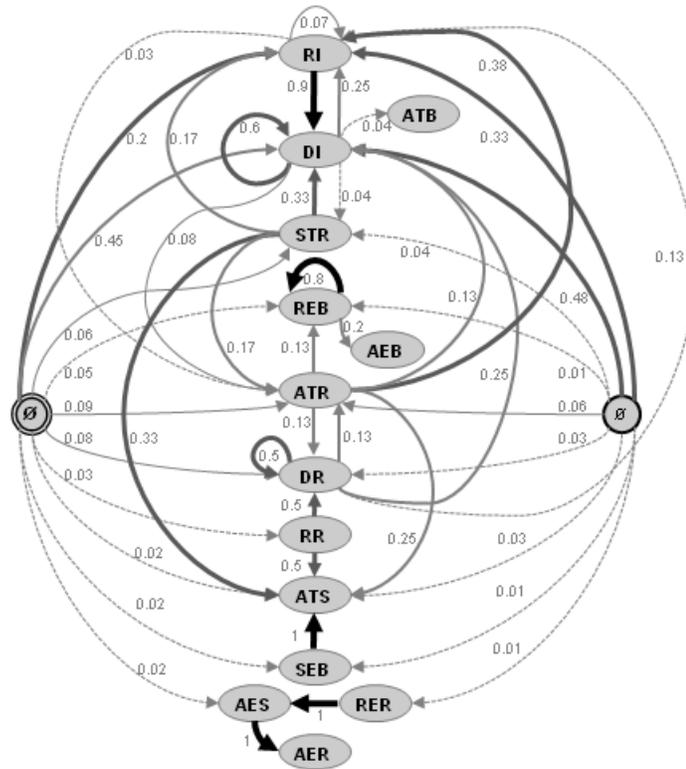


Figure 7.11: Bayesian network for the observed workflow speech act transitions

transition diagram generated by the two annotators, with nodes representing individual instances from the speech act model, and arcs the observed probabilities of transition from one speech act to another – as a percentage of instances where transitions were observed. A similar transition diagram was provided in [Carvalho and Cohen, 2005], inspired by Winograd & Flores’ *Conversation for Action* structures [Winograd and Flores, 1986].

Nodes in Fig.7.11 include speech acts which have been observed to precede or succeed others plus two others representing the start of a sequence of speech acts (workflow). The double outlined \emptyset represents the start of a new workflow within a new email thread, whereas the single outlined \emptyset represents the start of a new unrelated workflow (i.e. a separate email topic) within an existing email thread. Whereas some of the results were expected, e.g. 90% of (*Request, Information, ∅*) speech acts (shown as RI) are answered

with a (*Deliver, Information, ∅*) (shown as DI) and another 7% with an additional information request (another RI); others appear to be less logical, e.g. the remaining observed 3% were answered with a task assignment (ATR). This state of affairs was observed across the board, with various expected results occurring at higher frequencies, but also a diverse amount of less expected speech act reactions occurring at substantial frequencies. Another example of the former are joint event requests (REB), which tend to be answered with further negotiative event requests (80%) or with approvals of the event (AEB, 20%). Half of the resource deliveries (DR) are answered with further resource deliveries, suggesting files being sent back and forth. However, a substantial 13% and 25% were answered with task assignments (ATR) and information deliveries (DI) respectively. The remaining ~13% were succeeded by an information request (RI). In fact, many of the observed speech acts have a substantial number of information requests as their successor. These constitute the best example of the earlier defined sub-workflows. That is, the information request in these cases was in relation to a preceding speech act (parent workflow), and is likely to be itself succeeded by the delivery of information, which is then taken into account in resuming the parent workflow. Many of the observed examples of what Fig. 7.11 depicts as transitions from DI to ATR (8%) and ATB (4%) were cases where the email user assigned a task (to someone else in the former, and to oneself in the latter case) after getting the requested information related to an earlier task request.

Despite the limitations of this study, it does provide a glimpse of the nature and complexity of speech act sequencing in email. To an extent this view confirms, although only visually, the hypothesis cited by various (computational) linguists in Section 5.2.4, i.e. that to an extent or another, it is frequently possible to predict the next communicative act in an underlying sequence. Or that, again quoting Stubbs in his *Discourse Analysis* contribution [Stubbs, 1983], “a significant percentage of conversational language is highly routinised into prefabricated utterances”.

7.2.4 A Behavioural model for E-mail Workflows

So far it has been explained how, sequences of speech acts within email messages in email threads can be interpreted as independent but concurrently executing e-mail workflows, and how a significant percentage of studied sequences in actual email conversations were routinely organised into specific ad-hoc email workflows, e.g. Meeting Scheduling, File Request, Task Delegation, etc. A behavioural model for email workflows is next presented, based on the speech act and workflow definitions provided in Section 7.1.3, 7.2.2 that can:

- Model frequently occurring email workflows
- Also support the ad-hoc, spontaneous, aspect of these workflows

This ties in with the conclusions for Chapter 4, especially point 5 (Pg. 87), which states that “such spontaneous courses of actions should not be considered as deviances, but taken into account in the very fabric of the resulting ad-hoc workflows”. This Section is mostly based on the work presented in [Scerri et al., 2008b], although a preliminary email workflow model appeared earlier in [Scerri et al., 2007a].

The *Semantic E-mail Workflow* (SEW) model presented hereunder is modelled explicitly in a standardised workflow language. Although the ‘Semantic’ in SEW only refers to the paradigm on which it is based, workflow patterns within can be given semantics through their translation to YAWL³⁵ and subsequently Petri Nets³⁶. Thus, by breaking down semantic email into a number of speech act processes, each of which executes within a formal workflow, the email process is given a semantics.

The SEW is based on the earlier defined email speech act and workflow definitions. Every email is conceptually broken down into a number of 1-to-1 transactions between the sender and each recipient, where each transaction can carry zero or more speech acts. If the role of an embedded speech act is initiative, it initiates a new workflow, i.e. a

³⁵<http://www.yawl-system.com/>

³⁶<http://www.informatik.uni-hamburg.de/TGI/PetriNets/>

Pattern	Description	Notation
Exclusive Choice (XOR-split)	The thread of control is immediately passed to exactly one outgoing branch based on the outcome of a logical expression associated with each.	
Simple Merge (XOR-join)	Convergence of two or more exclusive branches into a single branch. The active incoming branch passes control over to the outgoing branch.	
Multi-Choice (OR-split)	The thread of control is passed to one or more of the outgoing branches based on the outcome of logical expressions associated with each.	
Parallel Split (AND-split)	Branch diverges into two or more parallel branches each of which executes concurrently.	
Structured Synchronising Merge	The thread of control is passed to the subsequent branch when each active incoming branch has been enabled.	
Multi-Merge	The thread of control is passed to the subsequent branch immediately when just one of the active incoming branches is enabled.	
Structured Loop: Post Test	Executes an activity or sub-process repeatedly. Post-test is evaluated at the end of the loop to determine whether it should continue.	
Recursion	The ability of an activity to invoke itself during its execution or an ancestor in terms of the overall decomposition structure with which it is associated.	
Persistent Trigger	An activity is triggered by a signal from the external environment. These triggers are persistent and are retained by the workflow until they can be acted on.	
Simplified XOR-Split	The majority of exclusive choices in the workflow have two common choices. These are abstracted with this symbol and expanded later in Fig. 2.	

Table 7.2: Workflow Patterns in use in the SEW model

new sequence of successive speech acts. Otherwise the speech act is progressing a pre-established workflow. Each of the workflows can be modeled using a distinguishable SEW, where the sender of the first speech act takes the role of its initiator, and all other contacts implicated in the first and remaining speech acts, that of participants. Both initiator and participant(s) can interchangeably take the roles of both the sender and recipient of an email in the ensuing thread. This concept was demonstrated in the example involving Martin, Dirk, Claudia and Ambrosia, and the illustrations provided in Section 7.2.1.

The SEW model is grounded on key research in the area of control flow workflow patterns. The model uses 9 out of an available standard set of 40 workflow patterns describing the control-flow perspective of workflow systems (refer to Section 4.5.2). The 9 patterns in use are enlisted in Table 7.2 (the 10th is a customised version of the 1st, as explained shortly) together with a brief description and a graphical notation. Of the two graphical process modeling notations compared in [White, 2004], we have opted to use UML 2.0 Activity Diagrams ³⁷.

The workflow model is graphically represented in Fig. 7.12. A swimlane splits the figure vertically to distinguish between the workflow initiator and each individual workflow participant. Whereas the initiator is always the agent that starts the workflow, its termination can depend on the initiator, the participant, or both. This is reflected by the position of the termination symbol, which is external to both agent swimlanes. A walkthrough description of the SEW is provided below.

A workflow is initiated when an agent initiates a speech act process by sending one of the following sets of *initiative* speech acts, (as marked in Fig. 7.12) :

1. (*Suggest*, $o \in \{Task, Event\}$, $S \subseteq \{s, R\}$): An activity suggestion involving the sender (the workflow initiator), one or more recipient(s) (workflow participants) or a combination of which. As the sender's expected action (*sea*) on sending the speech act is *None* and the recipient's expected reaction (*rer*) is *Acknowledge*, neither a further action from the initiator, nor a reply from the participant(s) is required.

³⁷<http://www.uml.org/>

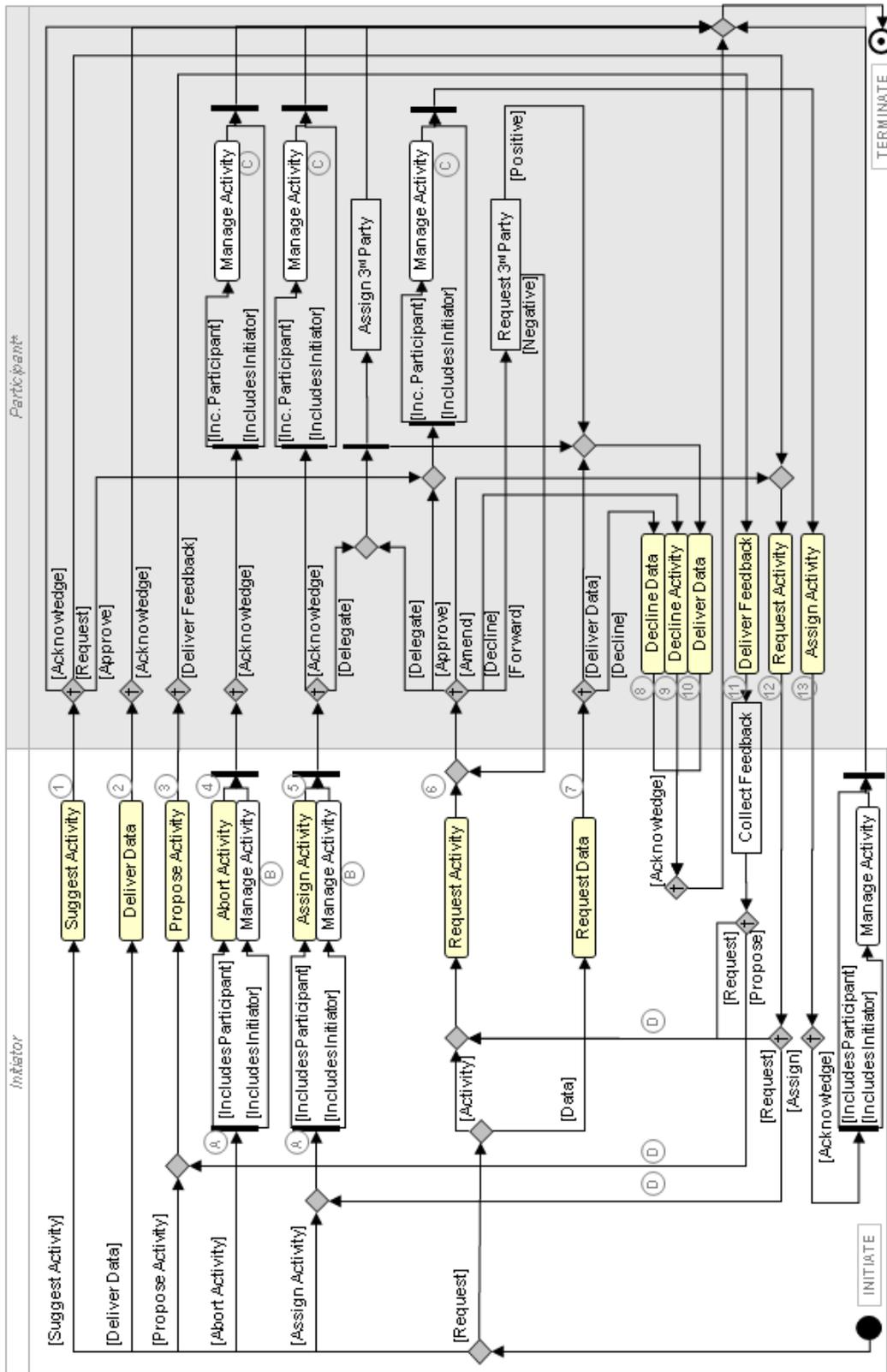


Figure 7.12: The Semantic E-mail Workflow (SEW) Model

2. (*Deliver*, $o \in \{Resource, Information\}, \emptyset$): Delivery of unrequested data. Again, given the *rer* and *ser* function for these speech acts, no further action from the initiator and neither a reply from the participant(s) is required.
3. (*Propose*, $o \in \{Task, Event\}, S \subseteq \{s, R\}$): Proposal of an activity involving a subset of all the contacts involved. Given the *rer* and *sea* for these speech acts, the initiator will be awaiting a reply. The *Propose* speech acts differs from *Request* (activity), such that the formers' outcome depends on the *collective* participants' feedback, i.e., the individual replies are not independent. The semantics for these speech acts is explained in more detail shortly.
4. (*Abort*, $o \in \{Task, Event\}, S \subseteq \{s, R\}$): Notification of a cancelled activity. Given the *rer* and *ser* for these speech acts, the initiator and/or participant need(s) to attend to the activity's management, depending on whom it implies, e.g. remove it from a calendar or task list.
5. (*Assign*, $o \in \{Task, Event\}, S \subseteq \{s, R\}$): Notification of a commitment or assignment of an activity. The implied initiator and/or participant need(s) to attend to the activity's management, e.g. add it to the calendar/task list.
6. (*Request*, $o \in \{Task, Event\}, S \subseteq \{s, R\}$): Request for an activity involving the initiator, the participant(s), or a subset of both. The initiator will expect a reply from all speech act subjects. Contrary to *Propose* speech acts with activity objects however, the activity's outcome will not depend on the collective responses, i.e. the individual replies are independent.
7. (*Request*, $o \in \{Resource, Information\}, \emptyset$): Request for information. The initiator will await a reply from the participant(s).

The SEW shows that the choice of some of these initiative speech acts, marked **A** in Fig. 7.12, result in a multi-choice split. If the activity's subject only includes the initiator (e.g. "I will do something! – a personal task commitment) the respective path

is executed. If it only includes the current participant (e.g. “You will do something!– a non-personal task assignment) the alternative path executes. If both are included (e.g. “We will do something! – joint task assignment), both paths execute simultaneously. If the activity implies an action by the participant, the speech act process (and ensuing workflow) continues simply by sending the speech act. If the activity implies an activity by the initiator, the latter is expected to manage the generated activity (labelled **B**), e.g. store or take note of it, so as to eventually carry it out. On receiving a speech act (refer to the participant swimlane, i.e. grey half of Fig. 7.12), the participant is presented with a choice of activities relevant to its type. Whereas some of these choices lead to the termination of the SEW, e.g. Acknowledge’, others execute lengthier paths. Some lead to activity management, this time on the participant’s side (labelled **C**), whereas others require the participant to send one of the following six sets of *continuative* speech act as replies to the initiator:

8. (*Deliver*, $o \in \{Resource, Information\}$, \emptyset): Deliver requested data in reply to (i.r.t) an earlier data request.
9. (*Decline*, $o \in \{Resource, Information\}$, \emptyset): Decline requested data i.r.t an earlier data request.
10. (*Decline*, $o \in \{Task, Event\}$, $S \subseteq \{s, R\}$): Decline participation or permission for an activity i.r.t. an earlier activity request.
11. (*Deliver*, *Feedback*, \emptyset): Deliver feedback i.r.t an earlier activity proposal.
12. (*Request*, $o \in \{Task, Event\}$, $S \subseteq \{s, R\}$): Request activity i.r.t. activity request, signifying an amendment to an existing activity (activity negotiation).
13. (*Assign*, $o \in \{Task, Event\}$, $S \subseteq \{s, R\}$): Commit to, or assign someone an activity i.r.t. an earlier activity request.

On sending one such speech act as a reply, control of the workflow is returned to the initiator (white half, Fig. 7.12). At this point, some paths lead to the termination of the

workflow (e.g. via ‘Acknowledge’) whereas others restart the loop, returning control to the participant through another speech act process (labelled *D*).

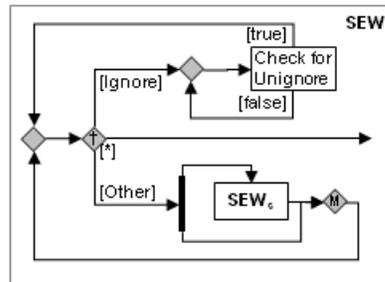


Figure 7.13: *Other* and *Ignore* choices in simplified XOR

The SEW as shown in Fig.7.12 has been simplified, and not just by grouping together speech acts by object category (i.e. activities and data). In effect, each exclusive choice that an agent must consider after receiving a speech act (the custom simplified XOR-split in Table 7.2) has two extra default choices - ‘Ignore’ and ‘Other’. Fig. 7.13 shows the specific behaviour of these two choices within the SEW. The ‘Ignore’ path leads to a structured loop pattern that uses a post test which continuously checks for a reactivation signal. If detected, it leads back to the start of the subactivity. The ‘Ignore’ option is not strictly speaking part of the formal workflow model. In fact, an ignored state was not included alongside the two standard speech act states (i.e. *Completed* or *Pending*). However, when implementing the SEW in real applications, an option to ignore speech acts was deemed necessary. Thus, although this option is not part of the formal speech act and workflow definitions, it is still included in the SEW. The second default option, i.e. the ‘Other’ path, uses the recursion pattern to enable initiative speech acts to be sent in reply to other speech acts – thus initiating a subworkflow SEW_s as defined in Subsection 7.2.2. Meanwhile, control is returned to the start of the sub activity via a multi-merge. This means that any number of independent subworkflows can be initiated and the parent SEW does not need to wait for any of them to terminate in order to regain control. Therefore, a workflow can technically terminate even before its subworkflow(s)

has terminated – if the agent decides to do so.

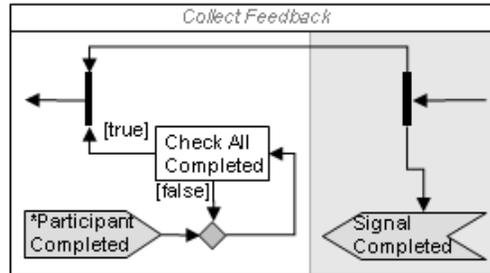


Figure 7.14: The dependant paths of a Propose speech act

The *Propose* speech acts have a special characteristic, such that when they are addressed to n recipients (and participants), they do not branch into n independent workflow instances. On the contrary, the workflow instances are *dependent*, and the workflow will stall until the initiator gets the responses from *all* n participants. In Fig. 7.12 this is represented with the abstract ‘Collect Feedback’ activity, which is expanded in Fig. 7.14, showing how, when each of the n participants delivers the required feedback, a signal is fired. On the initiators side, each time a signal fires, a post-check structured loop checks whether all participants have submitted their feedback. If this is the case the initiator can decide on which action to take, depending on the desired level of consensus in the collective feedback.

Two other abstract components in Fig.7.12 support two special functions – activity forwarding (Request *3rd* party) and activity delegation (Assign *3rd* party). These two specific paths have also been observed to occur frequently and are therefore specifically included in the SEW³⁸. The difference between these two is their speech act ‘force’. An activity delegation has the same *rer* as an *Assign* speech act, i.e. the recipient is not expected to negotiate, but simply carry out the assigned task. On the other hand an activity forwarding has the same *rer* as a *Request*, i.e. the recipient needs to approve the forwarded activity as their own, or otherwise.

³⁸Less frequent paths are still covered by way of the ‘Other’ option, even if not included in the model.

These two components have been expanded in Fig. 7.15. The activity delegation path leads to a parallel split into two paths. The first involves the initiation of a subworkflow – consisting of an initiative activity assignment speech act. Given the use of the multi-merge pattern, right after sending this speech act (and initiating the subworkflow) the SEW regains control, i.e. it does not wait for the termination of the subworkflow, which occurs when the third party acknowledges the activity assignment. Fig. 7.12 also shows how the parallel split preceding the 3rd party assignment leads to an information delivery addressed to the initiator, i.e. the participant is expected to notify the original requestor of the activity delegation.

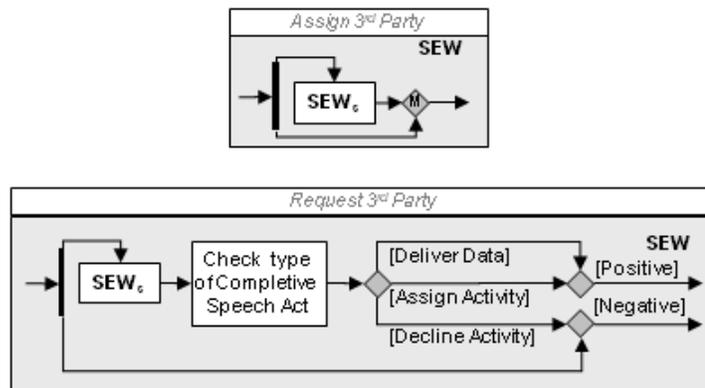


Figure 7.15: Representing Activity Delegations and Forwarding

Similarly, the activity path also splits in two parallel paths, the first of which leads to a new subworkflow, which in this case contains an initiative activity request speech act. Since the *ser* of an activity request is to await and the *rer* is to reply, the forwarding agent will keep waiting for the third-party's reply within SEW_s . However, through the second path, the outgoing *Negative* path is enabled right away, which means that the agent will still, at any time, be able to reconsider their options with respect to the received activity (see output of *Negative* path in Fig. 7.12). This also means that it could lead to attempts at additional delegations and/or forwardings of the same activity – which is just a reflection of the current, although not necessarily logical, possibilities in email

usage. When the activity forwarding (SEW_s) eventually terminates, the outgoing path that is enabled depends on the nature of the completive speech act terminating it. This is represented as an exclusive choice between the only three possible completive speech acts for an activity request, i.e. an activity assignment, an activity declining, or an information delivery (to notify of a successful, further, activity forwarding/delegation). If the third party declines the activity, the path merges into the *Negative* outgoing path – again leading the participant to reconsider the original activity request. The *Positive* outgoing path is enabled if the third party either accepts (assigns the activity to oneself), or is successful in forwarding/delegating it to yet another party (and notify the current agent through a deliver information). As shown in Fig. 7.12, the enabling of this path requires the current agent (workflow participant) to notify the original requestor (initiator) of the forwarding by means of an information delivery.

7.3 The Semantic E-mail Ontology

To enable machines to work with and support email workflows, it was required to provide machine-processable representations of the conceptualisations presented in this Chapter. For this purpose a domain ontology that represents all the elicited concepts, patterns and relationships in the semantic email paradigm was engineered. The ensuing Semantic E-mail (sMail) Ontology³⁹ enables and provides for:

- Extensible semantic email conceptualisation standards via the creation, manipulation and sharing of representations beneath different applications/UIs.
- Interoperable support for email workflows over multiple machines/platforms.
- Integration of desktop data from one domain (email) with data from other domains, e.g. calendaring, task management, folder structures, topic maps, etc.; via the re-use of vocabulary from other Socail Semantic Desktop Ontologies.

³⁹Online at: <http://ontologies.smile.deri.ie/smail#>

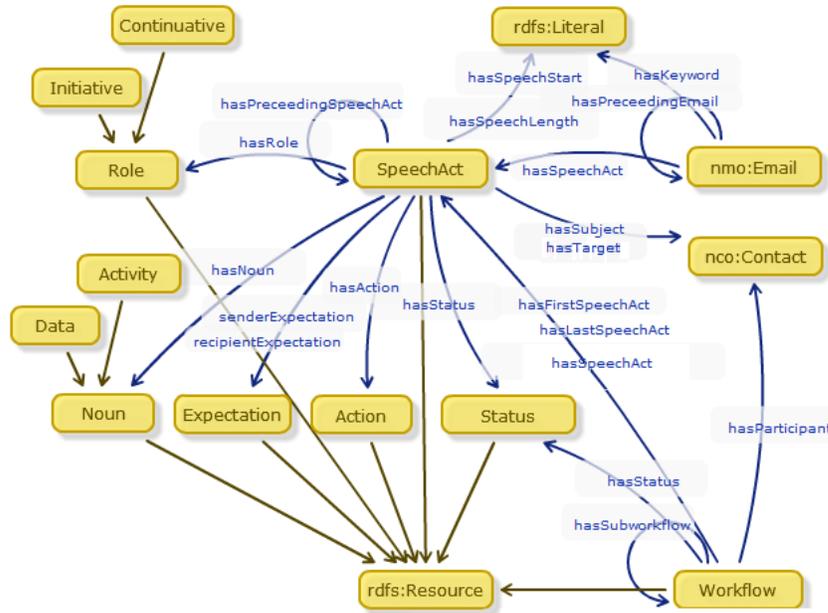


Figure 7.16: Classes and Properties in the Semantic E-mail Ontology

In the remainder of this Section a description of the vocabulary, as shown in Fig. 7.16, will be provided. Note the reuse of some concepts from the higher-level SSD Ontologies (Section 6.2), specifically *nco:Contact* to represent email users and *nmo:Email* to represent emails. This way, knowledge harvested from email workflows can also be exposed and linked to other knowledge on the SSD.

7.3.1 Speech Act Representation

The most important descriptions for a *smail:SpeechAct* point to a *smail:Action*, a *smail:Noun* and zero or more *nco:Contact*'s to represent its subject(s). The latter class is also used to define the speech act's target(s). Actual actions (i.e. *smail:Request*, *smail:Propose*, *smail:Deliver*, *smail:Suggest*, *smail:Assign*, *smail:Decline*, *smail:Abort*) are provided as instances of *smail:Action*. Nouns are subclassed into *smail:Data* and *smail:Activity* and actual nouns (i.e. *smail:Information*, *smail:Resource*, *smail:Feedback*, *smail:Event*, *smail:Task*) are then provided as instances of these subclasses.

Speech act roles are represented via *smail:Role*, which can be either of type *smail:Initiative* or *smail:Continuative*, which are its two subclasses. Actual roles (i.e. *smail:Imperative*, *smail:Informative*, *smail:Requestive*, *smail:Completive*, *smail:Negotiative*) are provided as instances for each. The speech act's sender expected action and recipient expected reaction are defined by a number of instances for *smail:Expectation* (i.e. *smail:Expect* – for the Await action, *smail:Acknowledge*, *smail:Reply*, *smail:None* and *smail:Perform* – for the Attend re/action).

Furthermore, a *smail:SpeechAct* is assigned a specific state through one of the provided instances for *smail:Status* (i.e. *smail:Pending*, *smail:Completed* and *smail:Ignored* – included only at the implementation level for practical purposes). Finally, to determine the location of a speech act within a chunk of text, *smail:SpeechAct* provides for pointers to the physical start and end positions within the text.

7.3.2 Semantic E-mail Representation

A Semantic Email is simply represented as a standard *nmo:Email*⁴⁰ extended with pointers to speech act instances within (via *smail:hasSpeechAct*) and the possibility to chain successive messages together (via *smail:hasPrecedingEmail*). Additionally, in view of the future plans for allowing semantic email to carry semantic tags, a corresponding property was introduced (*smail:hasKeyword*).

7.3.3 E-mail Workflow Representation

A *smail:Workflow* is defined as a set of successive speech acts (via *smail:hasSpeechAct*). The workflow must point to its first and last speech act (*smail:hasFirstSpeechAct*, *smail:hasLastSpeechAct*). The ordering of any other speech acts in between the first and the last is handled by a property which links a speech act to the preceding one (via *smail:hasPrecedingSpeechAct*). A workflow can also point to any

⁴⁰See appropriate vocabulary at <http://www.semanticdesktop.org/ontologies/nmo/#Email>

number of subworkflows (*smail:hasSubworkflow*), and all involved contacts in the encapsulated speech acts are automatically considered as workflow participants (via *smail:hasParticipant*).

The design of the ontology echoes the conceptualisation envisaged for semantic email, speech acts and workflows. Whereas both semantic email and workflow instances point to a number of speech acts within (unlinked and unordered in the former case, and linked and ordered in the latter), there is no actual direct semantic link between email and workflow instances. That is, although email workflows require email messages to be carried out, they are otherwise independent and can thus be visualised and represented outside the email message thread context.

7.4 Conclusions

The research presented in this Chapter reflects the core concepts and theoretical contributions of this thesis. These targeted the vision of a Semantic E-mail that serves two parallel purposes:

1. to provide an appropriate communication medium for data (and metadata) exchange on the Social Semantic Desktop (Chapter 6),
2. and to address well-known limitations of the email model (Chapter 4);

via the provision of rich semantic information pertaining to the structure, content and context of each email message. The *Semantic E-mail Conceptual Framework* provides for the better structuring of email communication in terms of specific email speech acts and the ensuing workflows, based on derivations from concepts within the Pragmatic Web (Chapter 5). The two main components of the framework are:

1. *The Speech Act Model* (Section 7.1.1) represents an improvement over the state-of-the-art in the modelling of specific types of email speech acts.

2. *The Email Workflow Model* (Section ??) interprets sequences of speech acts exchanged over email as independent, concurrent ad-hoc workflows.

Both models are extensible, and their modelling is supported by formal definitions (Section 7.1.3 and 7.2.2) of what constitutes semantic email, speech acts and workflows, including their characteristics and satisfiability conditions. In order to expose the knowledge embodied within the conceptual framework to machines, the relevant conceptualisation was captured within the Semantic E-mail Ontology (Section 7.3), which derives from pre-established concepts in the Social Semantic Desktop's other ontologies. This enables the management of interpersonal information generated via email over and across multiple Social Semantic Desktops. A number of future directions to improve the technology introduced in this Chapter will be discussed in Section 11.2.

8 Towards Automatic Speech Act Classification

The Semantic Email ontology (Section 7.3) enables machines to work with the speech act and workflow conceptualisations introduced in the semantic email conceptual framework, in order to enable semantic applications to support the users with handling email workflows. To provide the envisioned support any such applications need first become aware of different types of speech acts embedded in exchanged email. As users should not be burdened with manually recognising, classifying and annotating each single speech act, this makes for a knowledge-acquisition bottleneck problem. Thus a significant part of this thesis concerns the application of computationally treatable aspects of speech act theory for automatic classification. This Chapter will present a rule-based classification model, based on the presence and form of a number of linguistic features, employed to classify email segments into the predetermined set of speech acts (Section 7.1.1). The results of its evaluation suggest that whereas complete automation is not very reliable, the results are satisfactory for the presentation of user-reviewable suggestions. This Chapter is based on a publication describing this work [Scerri et al., 2010a].

8.1 A model for Rule-Based Information Extraction

The model is based on a rule-based classification model that considers the following five linguistic, grammatical and syntactical features:

- **Modality** - Sentence modality deals with different phrase types, of which the model considers three: *Declaratives*, *Imperatives* and *Interrogatives*. Whereas most interrogative sentences/clauses are easily recognised by the presence of a question

mark, to differ between the two remaining types, the model considers modal verbs (e.g. must, will), especially those expressing concepts of *Possibility* and *Necessity* (roughly equal to *Suggest*, *Assign* speech act actions respectively (See Fig. 7.1)).

- **Verb Category** - Verbs are used to express an action, an occurrence or a state of being. As the aim here is to recognise speech acts, the main interest lies with action verbs. For this purpose, the model differs between two self-categorised groups of action verbs: *Activity Verbs* – representing events and tasks (e.g. go, prepare); and *Communicative Verbs* – implying actions specific to electronic communication (e.g. send, forward, attach).
- **Grammatical Tense** - The tense morpheme specifies the time at/during which the descriptive content of the sentence in question holds [Ogihara, 2007]. Of the different tense categorisation schemes in English [Comrie, 1985], the model adheres to the two-tense approach, i.e. *Past* versus *non-Past*, as the interest generally centers around actions that occur in the non-past.
- **Negation** - From a pragmatic point of view, negation usually expresses the exact opposite of what otherwise the statement would convey, i.e. impossibility instead of possibility, prohibition instead of permissibility [Möschler, 1992]. Nouns and verbs can be negated via negative adjectives, pronouns or adverbs.
- **Semantic Role** - The speech act model (Section 7.1.1) introduces the subject as a very important parameter that represents the person implied in an activity. The classification model is concerned with the semantic rather than the grammatical roles, i.e. the *Agent* and the *Patient* for an activity. The *Grammatical Person* for both roles, specifically – *First*, *First Plural*, *Second* or *Third Person* has also a bearing on the classification task.

The model depicted in Fig. 8.1 provides an insight into how these features can be factored in to determine which speech act best represents a text clause. The illustration

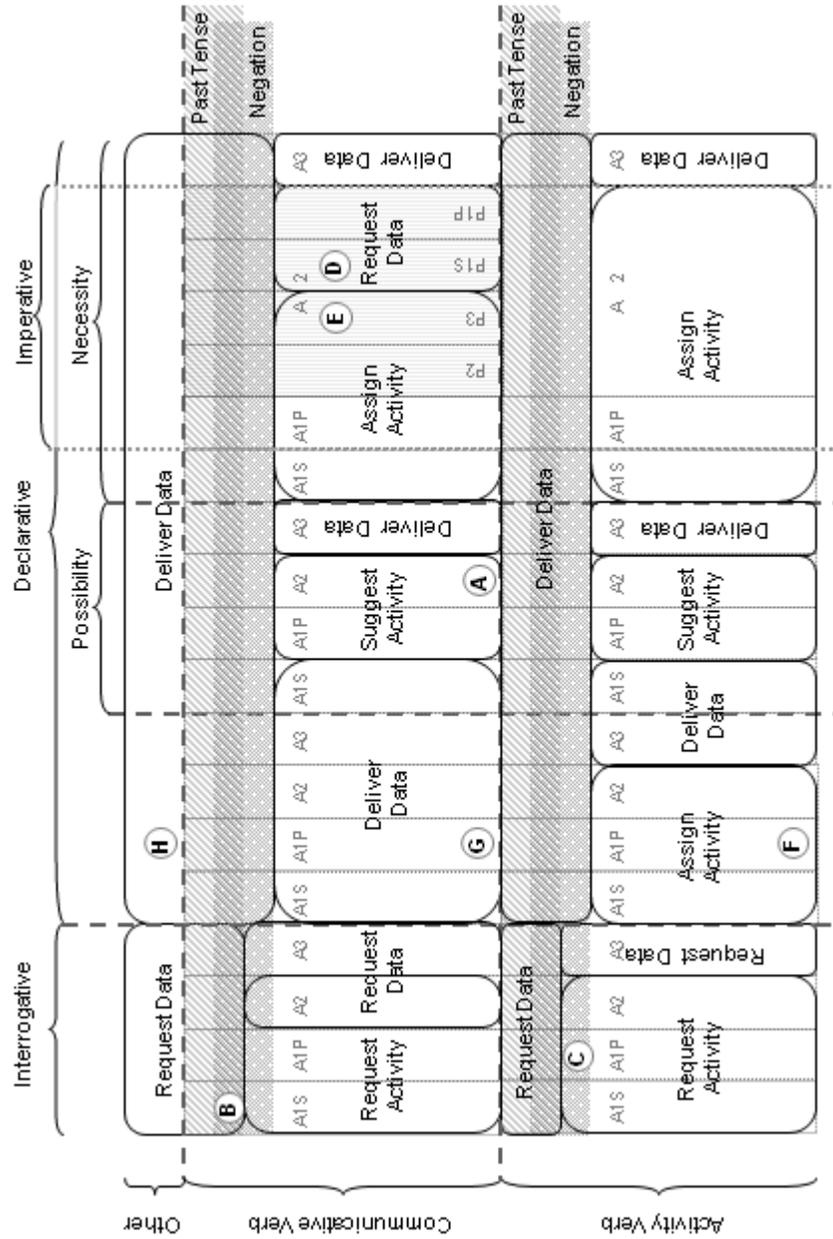


Figure 8.1: Graphical representation of the simplified classification model, classifying clauses from textual email content into five broad speech act categories

breaks down the ‘linguistic space’ into a number of dimensions, given the presence or form of the above features. Grammatical clauses are classified into exactly one of the resulting classification classes, which coincide with 22 out of the 38 possible types of speech acts depicted in Fig. 7.3, i.e. only those 22 instances which can assume an initiative role. This is because the aim here is only to try and elicit the first speech act in an email workflow, as the ensuing sequence of speech acts can be elicited through other means, e.g. via user interaction (more about this in the next Chapter). For simplification purposes, the figure does not include all 22 classes, but abstracts over five broader categories which entirely disregard both the different object types and the subject parameter. The resulting five categories are equivalent to the following five pairs of ‘Action - Object Category’: *Request Data*, *Request Activity*, *Suggest Activity*, *Assign Activity* and *Deliver Data*.

As for the classification itself, an individual textual clause is narrowed to exactly one speech act (category) as follows. Modality splits the space vertically into interrogatives and declaratives. Declarative statements are further differentiated between those having a possibility modal, those having a necessity modal (which subsume imperative statements) and the rest. Horizontally, the space is split between communicative verbs, activity verbs and their complement. Statements having activity or communicative verbs are then segmented given the agent semantic role where *A1S* stands for agent 1st person singular; *A1P* for agent 1st person plural; and *A2*, *A3* for agent 2nd, 3rd person respectively. In order to keep the figure as simple as possible yet be able to provide the required examples, the patient role (similarly; *P1S*, *P1P*, *P2*, *P3*) only features in one quadrant (the intersection for Necessity modals and Communicative Verbs). Negation and grammatical tense (past tense) are represented as (overlapping) horizontal shades of grey across statements with communicative and activity verbs.

I will now demonstrate how the classification model can in theory be employed to classify clauses into a specific speech act by way of eight practical examples (Table 8.1). The examples are matched against one of the implemented JAPE pattern rules [Cunningham, 1999] (introduced in Section 8.2). The bold tokens in example **A** are

Table 8.1: Examples of text as classified by rules based on the classification model

A	<p>“ You should forward it to me.”</p> <p>(([A1] [A2]) [PosMod] ([TaskV] [EventV] [CommV]) : SuggestTaskRecipient</p>
B	<p>“Hadn’t I sent you the file ?”</p> <p>([PastAux] [PastAuxNeg]) ([A1S] [A1P] [A2] [A3]) [CommV] (Person)? [Q] : RequestInformation</p>
C	<p>“Can we not meet to plan ahead today?”</p> <p>([Modal] ([ModalNeg]))? ([A1S] [A1P] [A2]) [Neg]? ([TaskV] [EventV]) [Q] : RequestJointEvent</p>
D	<p>“you still have to send me the document!”</p> <p>[A2] ([NecMod])? [CommV] ([P1S] P1P)? : RequestResource</p>
E	<p>“You must email them the document.”</p> <p>[A2] [NecMod] [CommV] ([P2] [P3] [Entity]) : AssignTaskRecipient</p>
F	<p>“We are going to attend the meeting,”</p> <p>([A1S] [A1P] [A2]) [NecMod]? ([TaskV] [EventV]) : AssignJointEvent</p>
G	<p>“We are sending you the files”</p> <p>([A1S] [A1P]) [NecMod]? [CommV] [P2]? : DeliverResource</p>
H	<p>“We are happy.”</p> <p>Catcher rule all for unclassified declarative clauses : DeliverInformation</p>

matched against the left hand-side (LHS) of the rule below to classify as a task suggestion speech act (*Suggest, Task, Recipient*); were “You” is recognised as a 2nd person agent, “should” as a possibility modal, and “forward” as a communicative (electronic communication context) verb. The classification can be mapped to Fig. 8.1 by focusing on the intersection between the horizontal communicative verb segment and the vertical declarative/possibility modal segment. The presence of a second person agent in the text places it in its shown position in Fig.8.1-A. As the speech act categories in the figure disregard the subject parameter, the statement is shown within its broader category, *Suggest Activity*.

The question mark at the end of example **B** indicates an interrogative statement. A 1st person agent preceding the “sent” verb (of type ‘communicative’) would normally have placed this in the *Request Activity* quadrant in Fig.8.1-B. However the question’s past tense, identified by the verb and the past auxiliary “Hadn’t” reduces it to a simple information request (*Request, Information, ∅*) within the *Request Data* category. Although the past auxiliary is negated, negation alone would not have effected the statement. Thus “Won’t I send you the file?” or “Will I not send you the file” would still have classified as a personal task request.

Example **C** is similar to **B**, with the difference that the verb “meet” is recognised as a non-past activity verb. Being an interrogative, this places the statement in the lower horizontal segment. Although “Can” is recognised as a modal verb, these verbs only effect noninterrogative statement. Negation neither effects this classification, and the presence of the 1st person plural agent “we” results in a joint event request (*Request, Event, Both*) shown as a *Request Activity* in Fig. 8.1-C.

Examples **D** and **E** are both declarative statements with a necessity modal, and they differ only with respect to the patient role. Whereas **D** is a request for the recipient (2nd person agent – “you”) to perform a communicative action verb (“send”) to the sender (1st person singular patient – “me”); **E** is a request for the recipient to perform a communicative action (“email”) to a third party (3rd person patient – “them”). Whereas **D** is classified as a resource request (*Request, Resource, ∅*), **E** is classified as a task assignment speech act (*Assign, Task, Recipient*). The resulting difference in classification is illustrated in Fig. 8.1, mapped to the broader categories of *Request Data* (Fig. 8.1-D) and *Assign Activity* (Fig. 8.1-E).

Example **F** and **G** differ mostly due to the verb type. The presence of a 1st person plural agent (“We”) followed by a non-past activity verb (“attend”) classifies as a joint event assignment (*Assign, Event, Both*) as shown by Fig. 8.1-F. **G** includes a non-past communicative verb (“sending”) followed by a 2nd person patient (“you”). This classifies the statement as a resource delivery speech act (*Deliver, Resource, ∅*), shown in Fig.

8.1-G within the *Deliver Data* category.

All non-interrogative clauses which remain unclassified by the implemented pattern rules, e.g. example **H**, are classified as an information delivery (*Deliver, Information, ∅*), also shown as a *Deliver Data* in Fig. 8.1-H.

8.2 Implementation

The classification model has been implemented as a rule-based classifier in GATE [Cunningham, 2002]. The classifier consists of an Information Extraction (IE) pipeline based on ANNIE – a Nearly-New IE system that is distributed with GATE.

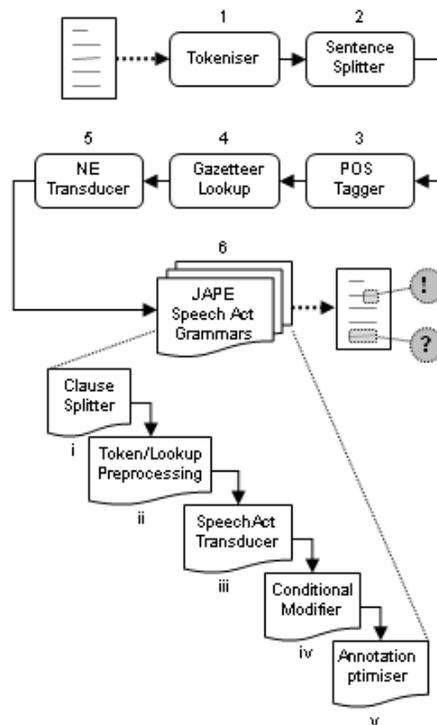


Figure 8.2: The Speech Act Classification Pipeline

The pipeline, shown in Fig. 8.2, consists of

1. Standard GATE English Tokeniser

2. Standard Sentence Splitter
3. Standard Hepple POS Tagger
4. ANNIE Gazetteer Lookup
5. Standard Named Entity Transducer
6. JAPE Speech Act Grammars Set

After the text is tokenised and split into sentences, it is forwarded to the *POS tagger*, which assigns a part of speech category to each token. In particular, the POS tagger recognises past tense verb inflections. The gazetteer lookup performs customised finite state lookup for key-phrases, including trigger words/phrases for linguistic features such as negation, modal verbs, grammatical person and the two verb categories introduced in Section 8.1. The *Named Entity Transducer* performs named entity identification and is particularly useful in the recognition of person references in the text. The set of hand-coded *Java Annotation Patterns Engine* (JAPE) [Cunningham, 1999] grammars is the most important component in the pipeline, as they provide pattern rules such as the ones provided in Table 8.1. The rules match combinations of the linguistic/semantic annotations output by the previous pipeline components, to classify clauses into a speech act. The grammars constitute a cascade of finite state transducers over patterns of annotations, such that the output of one transducer becomes the input of the next, as follows:

- i The *Clause Splitter* is a custom modification of the sentence splitter that splits sentences into individual clauses, on which classification is performed.
- ii The *Token and Lookup Preprocessor* binds tokens/gazetteer entries to intermediate annotations (e.g. groups modal verbs, grammatical persons by category)
- iii The *Speech Act Transducer* matches combinations of intermediate annotations to one speech act class. This is where most of the pattern matching is performed.

- iv The *Conditional Modifier* changes some identified speech acts based on preceding/-succeeding conditional modifiers, e.g. an ‘if-then’ clause before a task assignment changes it to a task suggestion/request, depending on the context.
- v The Annotation Optimiser extends speech acts to cover whole sentences, and groups together consecutive identical speech acts.

Each transducer consists of a collection of phases containing pattern/action rules (e.g. ones shown in Table 8.1). The LHS of the rule is written in BNF style⁴¹ whereas the right hand side consists of annotation-binding variables within a block of JAVA code, which can be manipulated as desired. JAPE rules are set to fire according to the desired behavior e.g., based on textual ordering, priority or longest match. *The Speech Act Transducer* alone consists of 58 rules within 14 different phases, such that text matched in the initial phases is not considered later.

8.3 Evaluation

To evaluate the classification rules, twelve people were employed to review automatic annotations generated for at least 10 email messages. The evaluators were introduced to the available classification classes prior to the task and the reviewing consisted of rating the classified speech acts, and annotating the missing ones manually. Each returned speech act could be rated using a 4-point Likert scale, two for correct annotations and two for false positives. The reason for multiple positive and negative ratings is that the classes for the classification task are not always outright good or bad, and there are cases where multiple speech acts can apply to a clause or phrase, with different levels of suitability. For example, although for one user “Please make sure to give me the document” is ideally classified as a $(Request, Resource, \emptyset)$, it can also be classified as a $(Request, Task, Recipient)$. Ultimately, it is the author of the email who is in the best position to determine whether the classified speech acts apply, and to which degree.

⁴¹<http://foldoc.org/?Backus-Naur+Form>

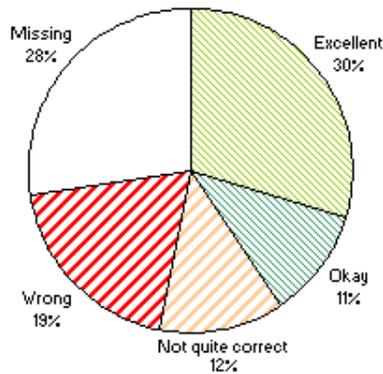


Figure 8.3: Evaluation results for the rule-based classification task

The evaluators ran the classification rules over a total of 116 emails, rating 194 speech act classifications. A further (missing) 74 speech acts were manually annotated. Positive ratings, representing correct classifications, amounted to 41%, negative ratings (representing false positives) to 31%, and missing speech acts to 28% (Fig. 8.3). An F-measure of 0.58 was obtained, weighing precision (0.56) and recall (0.60) equally. The result needs to be interpreted in the light of the result obtained for the inter-annotator agreement experiment in Section 7.1.2, which at 08.11 indicates the difficulty of the classification task, even when performed by humans. Also, both results are well within the reasonable performance range described in related literature for IE tasks of a similar complexity [Cunningham, 2005]. However, an F-measure of 58% means that although the rule-based classification fares very well, it is still not reliable enough to be fully automated. This was taken into account in the implementation of the semantic email support system presented in the next Chapter, which only provides for semi-automatic email annotation, in the form of speech act annotation suggestions to the user.

8.4 Conclusions

This Chapter targeted the knowledge-acquisition problem associated with enabling machines to recognise the initiation of email workflows, in view of the envisioned computer-

mediated workflow support for end-users. The approach entails an attempt at the automatic classification of textual email content into speech acts.

Considering the score achieved for speech act agreement between humans (Section 7.1.2) – 81%, the evaluation results for the presented classification system – 58% is encouraging. Although it is not suitable for full automation it is satisfactory for partial automation, thus greatly reducing the added burden of speech act annotation expected out of humans when writing new email messages. A number of future directions to improve this technology will be discussed in Section 11.3.

9 Semanta

The vision behind the work presented in this thesis is to provide for an appropriate metadata-enabling communication medium for the Social Semantic Desktop (SSD) (Chapters 3) that also addresses email overload and the ensuing interpersonal information management problems (iPIM) (Chapter 4). This Chapter will revisit these problems from a workflow perspective, to demonstrate how the implemented semantic communication support system – *Semanta*, provides better support of email-based collaboration and email-generated iPIM on the SSD.

Semanta has been implemented as add-ons to two of the most popular email clients in use: *Microsoft Outlook 2003*⁴², *Mozilla Thunderbird 2.0*⁴³; serving as a proof of concept for the Semantic E-mail Conceptual Framework (Chapter 7). This chapter will show how *Semanta* serves as an intelligent workflow management system, as well as an appropriate communication medium for the SSD; thus showing how the two main objectives of this research have been fulfilled. Also provided are an overview of the system’s multi-tier architecture, an insight into the UI’s intelligent features, a description of the system’s gradual evaluation, and a discussion of the main findings and the drawn conclusions. Parts of this Chapter are based on a number of publications regarding *Semanta* [Scerri et al., 2008a, Scerri et al., 2009a], including demonstrations portraying the evolving intelligent UI [Scerri, 2008a, Scerri et al., 2009c, Scerri et al., 2009b]. The two-staged evaluation was also covered in [Scerri et al., 2008b, Scerri et al., 2010b].

⁴²<http://office.microsoft.com/en-us/outlook-help/introducing-microsoft-office-outlook-2003-HA001071498.aspx>

⁴³<http://en-us.www.mozillamessaging.com/en-US/thunderbird/2.0.0.23/releasenotes/>

9.1 E-mail Overload Revisited

In Chapter 4, Email overload was attributed to a poorly structured email model that is used in a multitude of ways which go beyond its intended design. As an extension to the collaborative workers' working environment, email serves as a virtual workplace where they collaborate, carry out tasks, etc., generating and sharing new personal information in the process. With email persisting as the most popular digital communication medium, desktop users are constantly faced with a difficulty in handling email-generated interpersonal information. One can say that the vast amount of heterogeneous information reaching the users' desktops outstrips their abilities to correctly manage and exploit it. This results in widespread information management problems, that especially affect those users that thoroughly depend on electronic collaboration to carry on with their daily work. Thus, this thesis explores the possibility of enabling machines to support the handling, visualisation and integration of email-generated information, as a way of alleviating the underlying desktop information management problem.

Numerous research efforts have targeted the email overload problem. Some have taken a direct approach, such as through the development of technologies for email classification (Section 4.4), search and retrieval (Section 4.2). Others have taken less direct approaches, e.g. by facilitating email visualisation (Section 4.3) or even structuring email processes (Section 4.5). Central to the comprehensive approach described in this thesis, is the idea that email overload can be reduced by providing automated support for the underlying email workflows. From this perspective email overload can be considered as a workflow management problem where, faced with an increasing amount (and complexity) of co-executing workflows, users become overwhelmed and lose track of them.

The Semantic Email approach described in Chapter 7 aims to identify and place patterns of email communication into a structured form, without changing the email experience for the end-user. From an end-user point-of-view, the speech acts introduced in

the framework (Section 7.1.1) can be considered as *E-mail Action Items*⁴⁴ (e.g. Task Assignment, Meeting Proposal). As in the presented email workflow model (Section 7.2.4), sequences of related action items exchanged in email are then treated as implicit, but well-defined, ad-hoc *E-mail Collaborations* (e.g. Task Delegation, Meeting Scheduling). The way these collaborations (or rather their implicit action item components) are represented on the user's conventional desktop system is in no way similar to how the users would conceptualise them in their mind. In fact, one can say that there is a huge epistemological gap between the way users perceive email workflow knowledge and the way it is represented on their desktop. Such an epistemological gap hinders information management on the desktop, and in fact, its avoidance was a matter of great concern in the laying out of the requirements for the SSD (Section 3.2.4).

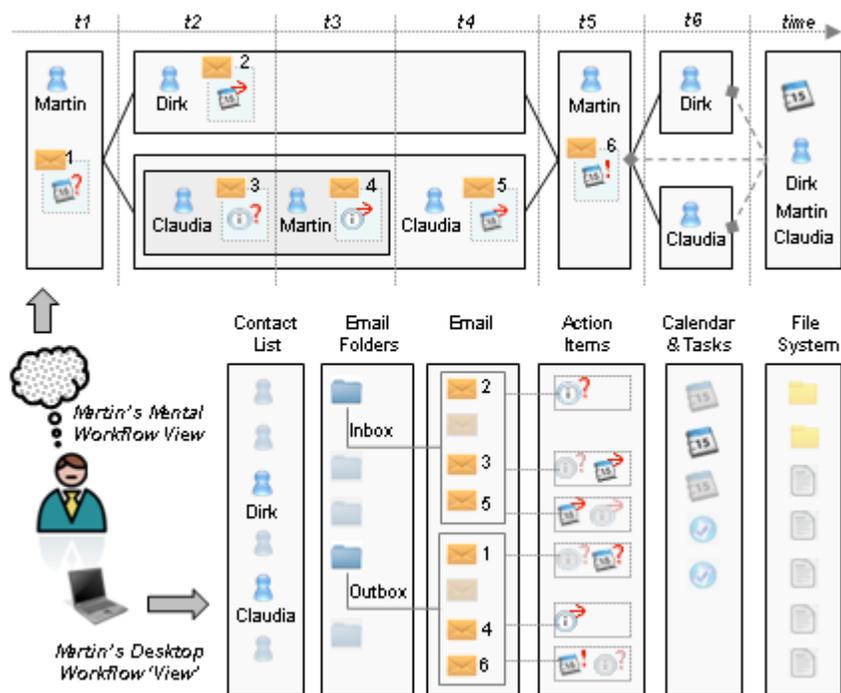


Figure 9.1: Martin's Conceptual vs Desktop Workflow View

⁴⁴Consequently the terms Speech Act and Action Item are used interchangeably in this Chapter

To highlight this problematic situation, I provide an example that illustrates how Martin perceives an email collaboration (workflow) in his mind and how he can physically see the corresponding fragmented information on his desktop (Fig. 9.1). At time t_1 , Martin writes an email (1) to Dirk and Claudia, which amongst others, contains a *Meeting Proposal* action item asking about their availability for a group meeting. This initiates an implicit *Meeting Scheduling* workflow, which splits in two co-executing paths at time t_2 ; control of which is passed to Dirk and Claudia individually. Dirk reacts to the meeting proposal immediately by sending an email reply (2) with his feedback (a *Feedback Delivery* action item). Being unsure of the meeting's purpose, Claudia replies (email 3) with an inquiry (*Information Request*). As explained in Section 7.2.4, this can be considered as a sub-workflow of the currently executing workflow. Martin deals with the sub-workflow at time t_3 , by replying with an *Information Delivery* in email 4, thus terminating it. Claudia can now get back to the initial workflow, and at time t_4 , she also sends her feedback to Martin (email 5). At this point (t_5), Martin has all the required information to make a decision regarding the meeting proposal in email 1. In other words, the workflow's two parallel paths merge back together and Martin regains its control. After deciding on the meeting's date and time, he sends another email (6), containing an *Event Notification* addressed to Dirk and Claudia. On sending this email, an event involving Dirk, Martin and Claudia is generated for Martin. When Dirk and Claudia acknowledge the *Event Notification* at time t_6 , the same shared event has been generated for all of them.

Unfortunately for Martin, the workflow view presented above is in no way similar to what he can visually gather through a conventional desktop email client. Unless the email collaboration is still fresh in mind, there is no straightforward way for him to quickly retrieve its overview. Worse still, there are multiple workflows running at the same time, within hundreds of email messages, with varying priorities and complexity. The bottom part of Fig. 9.1 shows the fragmented physical workflow view with which Martin has to make-do. The main workflow artefacts are scattered within a number of separate, largely unconnected, data 'islands'. Action items are obscurely strewn across a number

of usually (physically) unrelated email messages, which are in turn stored in different email folders. People in the contact list are only associated with these emails, and their roles in the contained workflows remain unspecified. The generated event is stored in a Calendar, with little or no connection to the email, or even the email thread, wherein it was generated. Workflow artefacts can also be dispersed in additional data islands, such as generated tasks which end up in a separate task list, or attached documents which are propagated onto the file system without any connection to their source email messages.

The example above demonstrates how the lack of visibility for these workflows is a major source of email overload, and the resulting management problems constantly faced by email users. The concepts, models and technologies introduced in Chapter 7 promise to provide for the elicitation, support and visualisation of email action items and ensuing workflows. Semanta's ambition is to realise this promise and provide this support.

9.2 Implementation

This section provides an overview of the design and implementation of Semanta, in view of two major established requirements described below:

- To take advantage of email's large userbase, it was required to integrate semantics into the existing technical landscape and to use existing email transport technology. Thus the favoured approach was to extend popular email clients rather than recreate a semantic email client. This called for a complete separation between the business logic and the required multiple UIs dedicated to each extended client, which was addressed by a multi-tier architecture (Section 9.2.1).
- While letting the users fully benefit from the additional intelligent functionalities, the semantics themselves must not be exposed to them. This supports complexity while reducing difficulty. Thus, the complex models and all generated metadata need to be completely hidden from the users, beneath a simple and intuitive UI, as discussed in Section 9.2.2. This fully supports the complexity of the underlying mod-

els, while minimising the difficulty of user interaction with the UI [Norman, 2002].

9.2.1 A multi-tier Architecture

Fig. 9.2 depicts the decoupling between the business logic (fifth level from top) and the UI (second layer). The role of each of the six levels/layers beneath the user level in the architecture, is explained below, bottom-up.

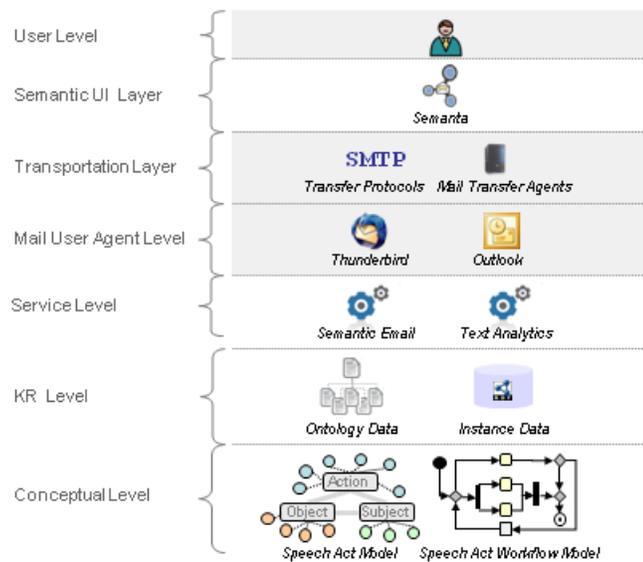


Figure 9.2: Semantic Email Implementation across the different levels

1. **Conceptual Level** :- This abstract level contains knowledge encapsulated within the Semantic Email Conceptual Framework, i.e. the speech act and workflow models.
2. **Knowledge Representation (KR) Level** :- The knowledge in the conceptual level is here exploited level via the semantic email ontology (Section 7.3), which itself re-uses concepts from additional SSD ontologies. Generated instances of these ontological concepts, e.g. semantic email, speech acts (action items), workflows,

contacts, tasks, events, etc.; also belong here, and they form the basis for all workflow operations performed by Semanta. Generated metadata is stored in the SSD's RDF store, meaning that Semanta's metadata is integrated with that on the SSD. This integration is not only at the practical level, but also at the conceptual (semantic) and levels. This means that, concepts like email contacts, events, tasks etc. can be directly related to other equivalent concepts on the desktop, e.g. contacts stored in instant messaging applications, tasks in a task manager, etc.

3. **Service Level** :- This level provides for all the business logic of Semanta by way of two services, implemented as a Java servlets. Both are included in the NEPOMUK Middleware [Reif et al., 2008], meaning that they also form part of the SSD's architecture. The *Text Analytics Service* employs the GATE Pipeline (Section 8) to perform action item classification on email content. The *Semantic Email Service* is responsible for most of Semanta's underlying technology, including the generation, retrieval, querying of metadata. Elicited and generated metadata is expressed in RDF format and stored in the RDF store within the KR level.
4. **Mail User Agent Level** :- This level is incorporated to stress that Semanta is not a stand-alone semantic application. Instead, it extends two widely used email user agents (MUAs): Microsoft Outlook and Mozilla Thunderbird. Whereas the former is built for the Windows operating system, the latter is platform-independent. Thus Semanta is available to most existing email users, and builds on their familiarity with conventional email clients.
5. **Transportation Layer** :- This layer is included in the picture to emphasise Semanta's reuse of standard email transport technology. The shaded levels in Fig. 9.2 stress Semanta's reliance on existing MUAs and transport technology. Metadata exchanged over email is embedded in a specific RDF MIME (Multipurpose Internet Mail Extensions) email header extension and transported alongside email content. When viewed with a standard email client, a semantic email appears as a normal

email. Thus non-semantic email users remain unaffected.

- 6. Intelligent User Interface Layer** :- The semantic UI is what the user perceives as Semanta, the extension. Powered by the knowledge in the conceptual level, which is exposed by the KR level and operated-upon by the the services layer, the UI delivers the intelligent workflow-supportive functionalities to the end-user.

Aside from the functionality provided by Semanta’s UI extensions, this architecture design ensures that the user’s email experience remains relatively unchanged.

9.2.2 Hiding complexity beneath an Intelligent User Interface

Semanta’s UI design is in keeping with the established guidelines set out in the human-computer interaction (HCI) area, particularly in [Shneiderman and Plaisant, 2009]. The design of a high quality interface that is easily understood, predicted and controlled by end-users is especially crucial in applications like Semanta. Here, users need to be encouraged to get more out of their habitual email practice, without being discouraged by invasive UIs and additional tasks.

The new functionality is provided either in addition and in similar style to that already provided by the MUAs, e.g. a workflow item view complementing the conventional email item (folder) view; or as suggestions and notifications via non-invasive window popups, e.g. task and event detection. The invisibility of the underlying models and semantics, coupled with this design, convey the feeling of an intelligent email support system. Most of this intelligent support is user-driven, via interactions with the system’s enhanced UI. Once an email workflow has been recognised, the UI enables Semanta to interact with the user in a way that only serves to further elicit workflow knowledge, dynamically and on the fly. The only, yet crucial, point where Semanta depends almost completely on the user is during an initiation of a workflow. As explained in Section 7.2.2, a workflow is initiated when action items that are not bound to any previous workflow (i.e. initiative speech acts) are included in an *outgoing* e-mail, e.g. sending a new meeting request, rather than

an amendment to an existing one. This accounts for one of the very first design decision, which considers the email sender as being in the best position to determine the intended types of embedded action items. As a result, an email is annotated before it is sent. Metadata pertaining to that email will then contain information about the action items and the corresponding generated workflows.

Therefore, once initiative (Pg. 143) action items in outgoing email are recognised, Semanta can provide seamless email workflow support, with little to no disruption of the user's conventional email use. To counteract this knowledge-acquisition bottleneck, the automatic classification technology presented in Chapter 8 was implemented within the text analytics service in Fig. 9.2. The latter performs semi-automatic action item classification in textual content of an outgoing email, such that the considered text is not already bound to a workflow. This includes, but not exclusively, all the text in the first email in a thread. Although given the difficulty of the task, the achieved $\sim 60\%$ F-measure is deemed fairly successful, as pointed out in the evaluation of the Swiftfile system introduced in Section 4.4.2, an error rate of over 20% is completely unacceptable in automated processes [Segal and Kephart, 1999]. Thus the classification performed by the text analytics service is meant to facilitate, rather than completely automate action item annotation. For the purpose, classification results are presented to the user as suggestions, requiring individual review. Therefore the UI's design was required to let users modify incorrectly classified speech acts, as well as create some anew. This threatened the second requirement in the start of this section, i.e. not to expose the user to the underlying conceptualisation and models. In fact, the most challenging aspect of Semanta's UI was the design of a suitable *Annotation Wizard* that facilitates the annotation creation/modification task, while hiding the complexity of the speech act model. As the latter provides for up to 38 unique speech acts (growing further with the number of email recipients, as possible speech act subjects), it was not acceptable to just enlist all the possible speech act instances and let the user select the most appropriate action item. Instead, the implemented wizard supports the user with *constructing* an annotation by

way of a few clicks. It guides the user by asking a simple question at a time, each time providing a set of relevant options, dynamically loaded from the underlying ontology.

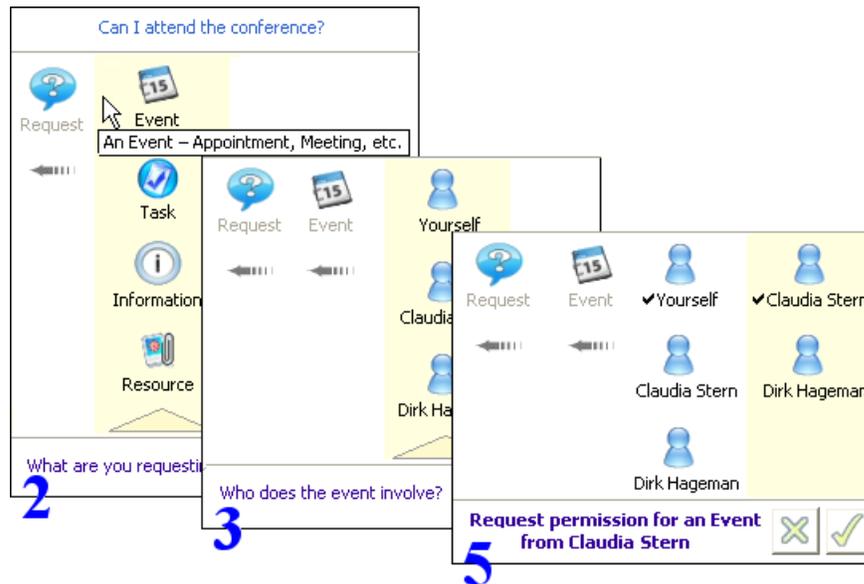


Figure 9.3: Constructing an Action Item annotation with the Annotation Wizard

Fig. 9.3 shows snippets (step 2,3,5) from a five-step construction of an action item annotation, which is presented to the user either when highlighting new email text for annotation, or modifying incorrectly classified ones. The text to be annotated in the example is “Can I attend the conference?” and it originates from an email being sent by Martin to Claudia and Dirk. Initially the wizard asks the user which (step 1, not shown in Fig. 9.3), of the available actions, best applies to this action item – ‘*What action is being performed?*’. After selecting a ‘*Request [Ask a question – reply required]*’, the user is then asked ‘*What are you requesting*’ and presented with a number of options (Fig. 9.3, step 2). When Martin decides for an ‘*Event [An Event – Appointment, Meeting, etc.]*’, he is asked ‘*Who does the event involve?*’ (Fig. 9.3, step 3). Since the user is actually asking for permission to attend an event, Martin selects himself only. Given there are multiple recipients, Semanta then asks Martin who he is requesting permission from (step 4, not shown in Fig. 9.3). Of the two email recipients, this action item is only addressed to

Claudia, whom Martin selects as the target. Following this last decision, the annotation is complete – ‘Request permission for an Event from Claudia Stern’ (Fig. 9.3, step 5). Once Semanta is aware of this initial action item in the workflow, its knowledge of the workflow model enables it to keep track of subsequent action items (updates to the workflow) and to support with their management on both the email sender and recipient(s) sides. The methods behind this functionality will become evident in the next Section, where actual examples and screenshots of additional components of the implemented intelligent UI will be presented.

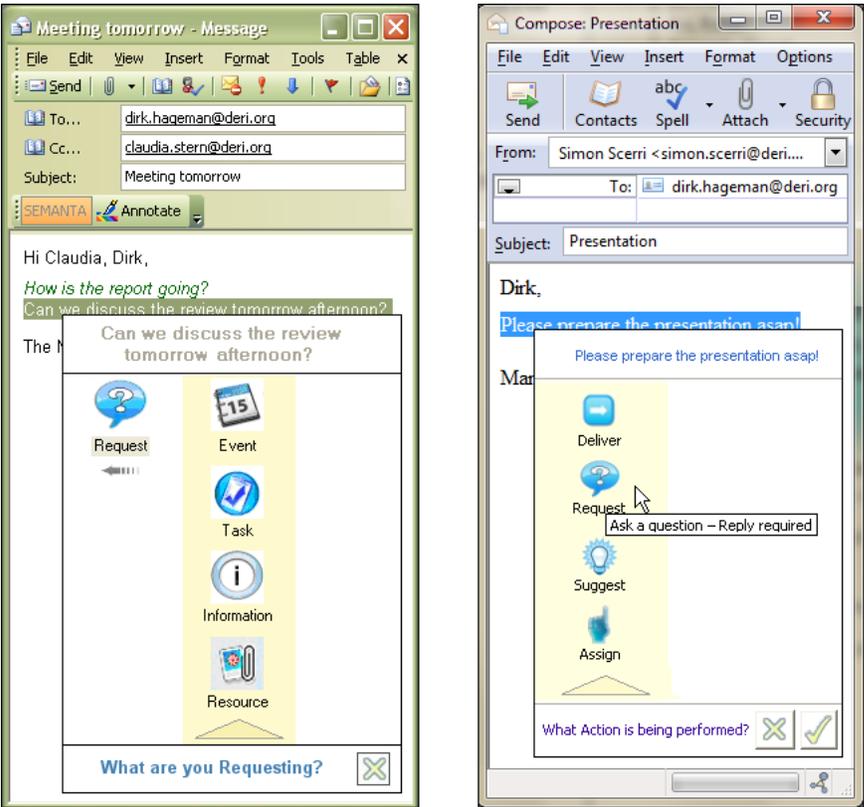


Figure 9.4: Same GUI look and feel, same underlying functionality, different email clients

The annotation wizard also serves to demonstrate the decoupling between Semanta’s business logic and UI. Although the screenshots in Fig. 9.3 were obtained from the Outlook plug-in, as shown in Fig. 9.4, the wizard appears and functions the same in the

Thunderbird add-on. Apart from obtaining the model-related knowledge dynamically from the semantic email ontology, all functionality, apart from the GUI behaviour itself, is provided by the semantic email service, which is common to both extensions. These include the generation of the UI questions posed to the user, themselves dynamically constructed given the user's selections; the decisions whether to prompt for the selection of a speech act subject (who is/would be performing a given task/event), depending on the speech act type; and the generation of the final annotation description. These functions are performed continuously, since the annotation can be changed at all times via the 'Back' option in the above screenshots (gray left-arrows).

9.3 An Intelligent Workflow Management Application

This section shows how Semanta addresses the first of the two complementary purposes of the Semantic Email paradigm conveyed by this thesis (introduction, Chapter 7), i.e., *to better support email-based collaboration by addressing limitations of the email model*, via the provision of rich semantic information pertaining to the structure, content and context of each message. Given the pursued workflow-oriented approach, I will show how Semanta fulfils this purpose through its email workflow management support. This intelligent support will be demonstrated via real examples and screenshots from both email extensions, although I will generally stick to the Thunderbird add-on.

9.3.1 Eliciting Workflow Knowledge

Section 9.2.2 explains how Semanta's knowledge-acquisition bottleneck (eliciting the first action item in an email workflow) is addressed via a classifier providing action item suggestions, and the intuitive annotation wizard presented in Section 9.2.2. These components come into play when the user finishes composing an email, right before it is sent. As action item annotation constitutes an additional task which the email user is not accustomed to, the GUI makes this task as easy as possible. The user is shown a reviewable set of classified action items, which can be removed, changed, or accepted (Fig. 9.5). The latter

option ensures that no misclassified speech acts result in incorrectly supported workflows. Changing a classified action item brings up the wizard, showing the current annotation, for modification. New action item annotations can also be created by highlighting the relevant text, performing a right-click, and following the option in the context menu.



Figure 9.5: Semi-automatic action item annotation

Action item annotation is required not just for messages starting a new email thread, but also for unbound text segments in email replies. By unbound, I mean text which has not already been automatically annotated by Semanta as an action item that is in reply to an earlier item in an existing workflow (details about this in the coming subsections). Alternatively, new workflows are determined in a straightforward fashion via Semanta's *Quickshots* - e-mail with *one* predefined action item, e.g. a *Meeting Request*. Annotation of quickshot content is thus unnecessary. Quickshots can be created via the respective 'Compose new e-mail' drop-down menu, and the user can pick from the most common workflows, e.g. joint meeting proposals, event permission requests, task assignments, etc.

9.3.2 Supporting Workflow Execution

Alongside the other email information, when a semantic email reaches the inbox, Semanta displays the number of pending action items (last column, Fig. 9.6). This adjusts dynamically when action items are taken care of. When an email is selected, action items within are highlighted in the content (red italic). Users can interact with each, whereby they are presented with a number of relevant options, based on the workflow model and the action item’s type. For the *Task Request* shown in the example – “can you prepare the agenda?”, Claudia has the following options, also traceable along the workflow model (Fig. 7.12). Firstly, Claudia can either approve the task, disapprove/decline it, or alternatively amend its properties. Given that action items having a *Request* action always require a reply, all of these options will result in an email reply being generated. The first option requires a reply to relay a *Task Approval* action item, the second a *Task Declinement* and the third a *Task Amendment*. In each case, the reply action item constitutes the next item in the workflow. A reply action item (and email) is however not always necessary. For example, upon receiving a *Task Assignment*, the user can simply acknowledge the assignment without the need for a reply, as per the semantics of the *Assign* action (particularly the recipient’s expected reaction).

As illustrated in Fig. 7.13, the semantic email workflow model augments each exclusive choice which an agent must consider after receiving an action item, with two default choices. In Fig. 9.6, through the first of these choices Claudia can ‘Ignore’ (with an undo possibility) the task request, if she desires. This is the only place where the practical implementation deviates slightly from the theoretical definitions, as the only two possible states for a speech act were defined to be either *Pending* or *Completed*. Users are provided the option to ignore a speech act only for practical reasons, such as when a user wants to ignore action items from a particular user, or an action item has been resolved outside the world of email (e.g. handing a printed requested document to a colleague). However, although in their own view an ignored speech act process is akin to have been completed, it might not always be the case for the sender of the speech act. In the example, if Claudia

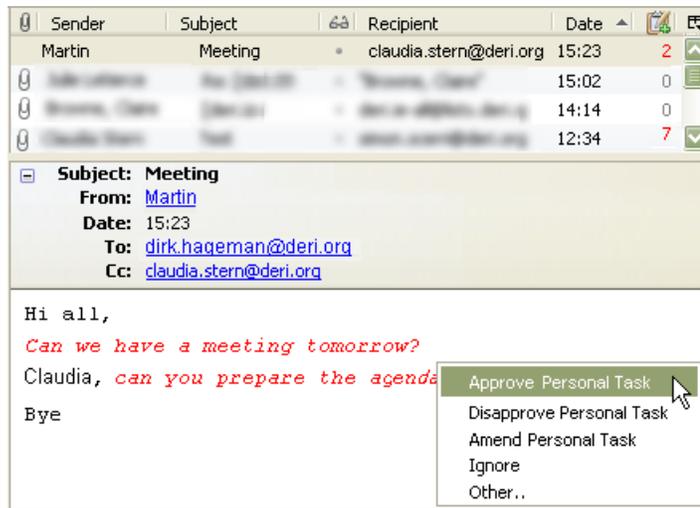


Figure 9.6: Supporting individual e-mail action items and associated workflows

ignores the task request, she will no longer be shown the action item as pending. This does not mean however, that Martin will not keep awaiting her reply.

The second of the default choices is what makes Semanta an *ad-hoc* workflow support system. The ‘Other’ option in Fig. 9.6 allows the user to react to an action item in additional, less-predictable ways. For example, before submitting her availability for the proposed meeting, Claudia decides to question its purpose. After selecting the ‘Other’ option, Claudia writes her question (e.g. “I thought the meeting was cancelled?”) with which she is again assisted to annotate, via the wizard, as an *Information Request*. This action item is embedded within an automatically generated email. When it is sent, a subworkflow of the original one is initiated, and its control is passed on to Martin.

9.3.3 Supporting Email-generated Activity Management

If a task or event in which users are involved (i.e. they are a subject of the underlying speech act) has been announced or approved, Semanta prompts them to directly export

them to the associated Task/Calendar management system⁴⁵. For example, if Claudia approves the meeting request shown in Fig. 9.6 (first action item), Semanta will suggest exporting this generated activity directly to the associated Calendar, as shown in Fig 9.7. Additionally, Semanta auto-completes some of the properties of the generated tasks/events. The contacts implicated in the activity are automatically included as the contacts involved. In this example, since they were both subjects of the meeting request speech act, Martin and Claudia are both expected to attend the meeting. The subject of the calendar item will also carry the textual excerpts from the workflow right away (e.g. *Martin wrote: “can you prepare the agenda?”: You replied: “Yes”*).

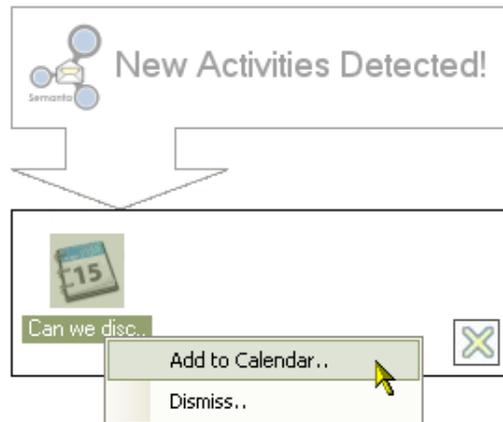


Figure 9.7: Detecting Generated Events and Tasks

9.3.4 Integrating Workflow Artefacts

Email messages, contacts, action items as well as generated tasks and events can all be considered workflow artefacts. Semanta ensures that artefacts related to a single workflow remain linked, providing workflow artefact integration. Links between them are stored in the underlying RDF store, later exploited to enable the user to navigate from one artefact to the next such that they are able to easily link and retrieve workflow data.

⁴⁵The default Outlook Task List and Calendar are used to store Outlook tasks and calendar items respectively, whereas the Lightning add-on is required for Thunderbird

This support is quite novel, and it counteracts the problem illustrated by Fig 9.1, where workflow artefacts remained scattered around the user’s desktop with no easy means of inter-navigation.

Fig. 9.8 extends the example above to illustrate Semanta’s workflow artefact integration for the three items related to the workflow which ensued Martin’s task request in Fig. 9.6. The request (Fig. 9.8-1) was answered via a task approval in an email reply (Fig 9.8-2), which also contained the information request belonging to the other workflow (initiated by the meeting request). The two emails shown are both artefacts of the same workflow, and are linked accordingly via the ‘Previous Email’ and Next Email’ buttons. The last item (Fig 9.8-3) is the task generated at the end of the workflow, specifically from the second email. The user can jump to this task from the email via the Related Activity’ button. Additionally, Semanta extends the standard display of the task item by a Conversation’ panel which shows the history of the workflow up until the generation of the task itself. Before the task generation, the workflow in the example consisted of the two action items shown (i.e. a task request send by Martin to Claudia, followed by a task approval sent in reply). The user can also directly jump from these two items to the emails within which they were exchanged, i.e. Fig. 9.8-1 and Fig. 9.8-2 respectively.

Through this integration, Semanta rectifies the information fragmentation problem in described Section 9.1, ensuring a better chance for users to stay in control of their email.

9.3.5 Intelligent Secondary Features

The comprehensiveness of the workflow-based approach pursued in this thesis allows for the implementation of secondary features which have been at the centre-stage of other efforts. An example is Semanta’s ability to prevent users from forgetting to attach files, which is a specific research problem targeted in efforts such as [Dredze et al., 2008a]. Semanta offers an indirect but effective solution - users are notified if they fail to attach any file to email with detected *Resource Delivery* action items (Fig. 9.9). A similar functionality for action reminders is in the pipeline (refer to Section 11.4).

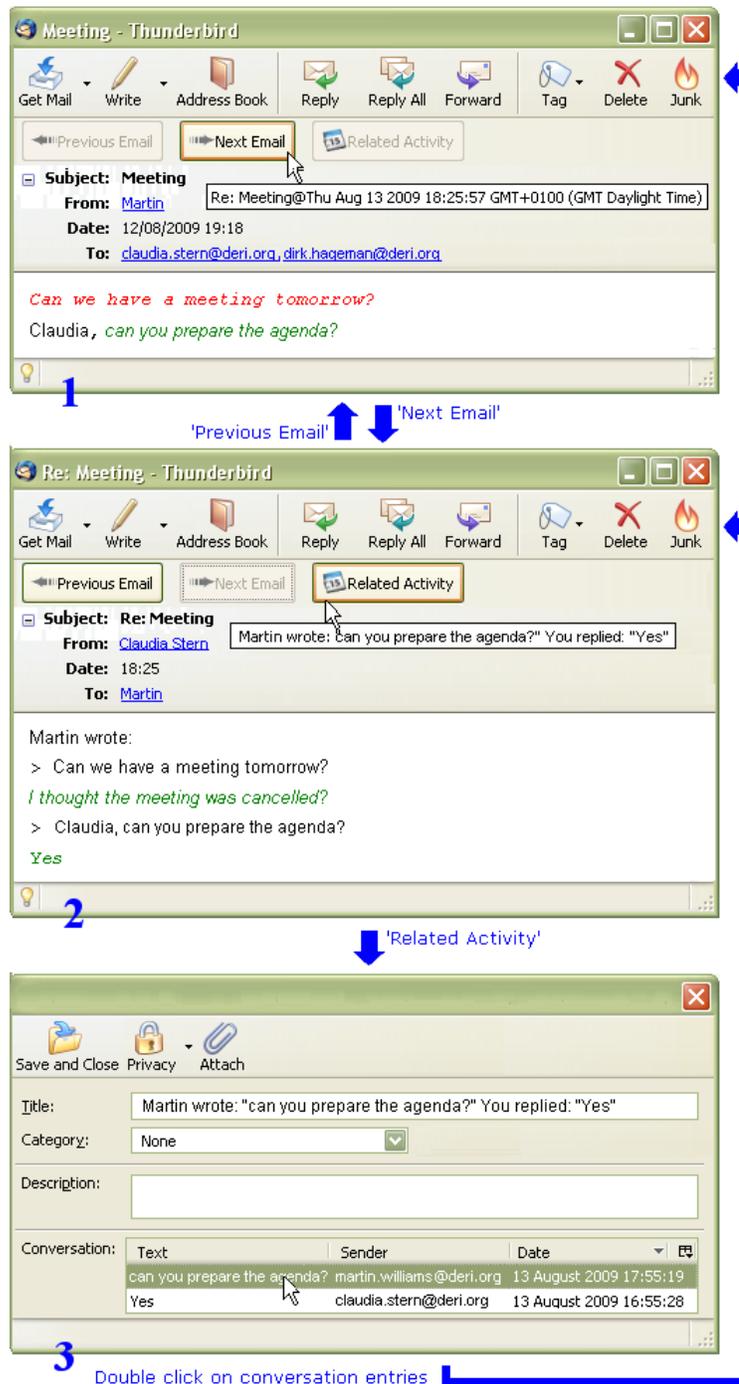


Figure 9.8: Integrating E-mail Workflow Artefacts (Messages, Tasks/Events)

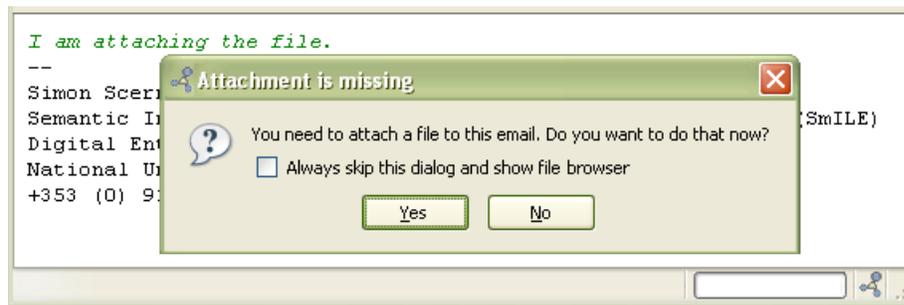


Figure 9.9: File Attachment Reminders

9.3.6 Visualising E-mail Workflows for Improved Management

Semanta’s most novel and exciting feature is the workflow-based email visualisation. This is to be differed from thread-based ones, of which a few have made their mark, e.g. the *Thread Arcs* integrated in TaskMaster (refer to Section 4.3.1). The workflow-related knowledge that is continuously elicited and gathered by Semanta, makes the system aware of all exchanged action items, their position within a workflow as well as their status. Semanta’s UI exploits this knowledge to visualise workflows and enable users to easily retrieve emails carrying the individual action items within. In Thunderbird, Semanta’s *Workflow Treeview* (Fig. 9.10) is available alongside the default email treeview on the LHS. The treeview provides three views, one selection of which enables the UI components on the RHS. In turn, these offer a form of visualisation that functions like faceted-search, enabling users to restrict the field-of-view to a particular email, starting from a workflow.

The main view (‘All’) displays a list of all the workflows that have taken place or are still running/pending (displayed in bold) in the *Workflow List*, ordered by start date. When a workflow in the list is selected, its details are shown in the *Workflow Details* component below. The details consists of the sequence of action items within. When one of the action items is selected, Semanta retrieves the email within which it has been exchanged and displays it to the user in the *Email Message* component below.

The example shown in Fig. 9.10 is more akin to Martin’s mental chronographic con-

ceptualisation of the workflow depicted in Fig. 9.1. In fact, the shown workflow originates from the *Meeting Proposal* he sent to Dirk and Claudia. The workflow details below show that whereas Dirk provided his availability right away (4th action item), Claudia asked for further information before providing hers. This sub-workflow is represented by the two indented action items: the *Information Request* (2nd item), followed by the *Information Delivery* (3rd item). The email within which this action item was exchanged (in Martin's Outbox folder) is displayed in the email message view below. The workflow is still marked as pending in the workflow list, because although Martin had received the feedback from both the other two meeting participants, he has yet to announce the meeting.

The following are the two additional views provided by the workflow treeview. The 'Incoming' view shows all incoming action items (e.g. requests, assignments, suggestions) which remain pending. In this case, rather than displaying a list of workflows, Semanta displays a list of pending action items, which are however shown in the context of their workflow (in the workflow details). The user can then resume the workflow by directly reacting to the pending item. In contrast, the 'Outgoing' view shows all outgoing action items (e.g. requests) for which the user is still awaiting a reply. Thus, the user can consider sending email reminders, in order for the correspondents to resume stalled workflows.

Although the Outlook workflow visualisation is similar (Fig. 9.11), the UI is inferior to that implemented in Thunderbird, due to the limitations faced by the non-open source nature of the software and the non-straightforward email data access. As a result, the UI does not provide for a counterpart of the *Email Message* view. Otherwise, all other views and functionality are present, even though the look-and-feel in this case differs.

The main view ('All') displays a list of all the workflows that have taken place or are still running/pending (displayed in bold) in the *Workflow List*, ordered by start date. When a workflow in the list is selected, its details are shown in the *Workflow Details* component below. The details consists of the sequence of action items within. When one of the action items is selected, Semanta retrieves the email within which it has been exchanged and displays it to the user in the *Email Message* component below.

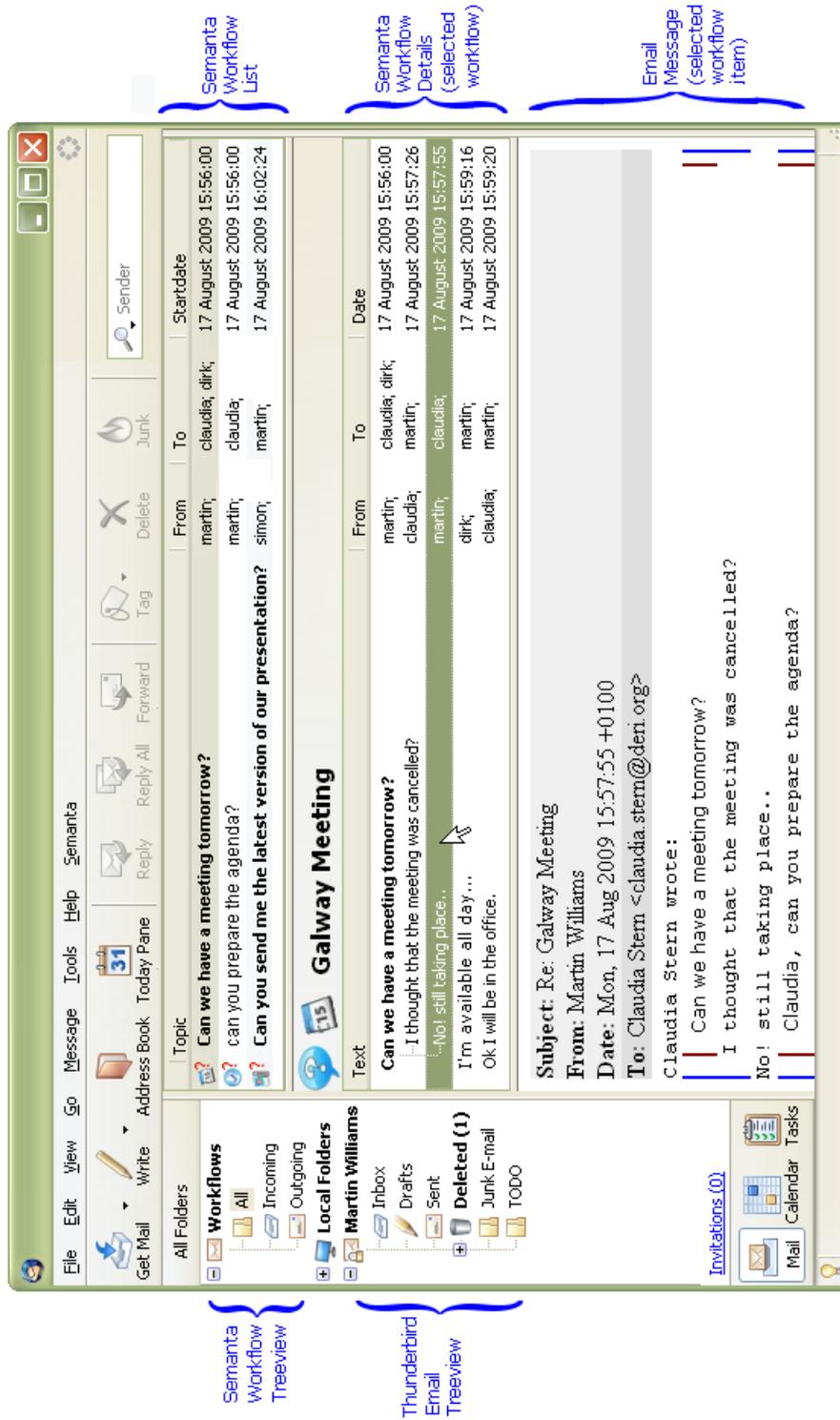


Figure 9.10: Screenshot showing Semantas workflow-based email visualisation. The user navigates from an initial action item (top right) to the ensuing workflow (middle right) and finally a specific email (bottom right).

The example shown in Fig. 9.10 is more akin to Martin’s mental chronographic conceptualisation of the workflow depicted in Fig. 9.1. In fact, the shown workflow originates from the *Meeting Proposal* he sent to Dirk and Claudia. The workflow details below show that whereas Dirk provided his availability right away (4th action item), Claudia asked for further information before providing hers. This sub-workflow is represented by the two indented action items: the *Information Request* (2nd item), followed by the *Information Delivery* (3rd item). The email within which this action item was exchanged (in Martin’s Outbox folder) is displayed in the email message view below. The workflow is still marked as pending in the workflow list, because although Martin had received the feedback from both the other two meeting participants, he has yet to announce the meeting.

The following are the two additional views provided by the workflow treeview. The ‘Incoming’ view shows all incoming action items (e.g. requests, assignments, suggestions) which remain pending. In this case, rather than displaying a list of workflows, Semanta displays a list of pending action items, which are however shown in the context of their workflow (in the workflow details). The user can then resume the workflow by directly reacting to the pending item. In contrast, the ‘Outgoing’ view shows all outgoing action items (e.g. requests) for which the user is still awaiting a reply. Thus, the user can consider sending email reminders, in order for the correspondents to resume stalled workflows.

Although the Outlook workflow visualisation is similar (Fig. 9.11), the UI is inferior to that implemented in Thunderbird, due to the limitations faced by the non-open source nature of the software and the non-straightforward email data access. As a result, the UI does not provide for a counterpart of the *Email Message* view. Otherwise, all other views and functionality are present, even though the look-and-feel in this case differs.

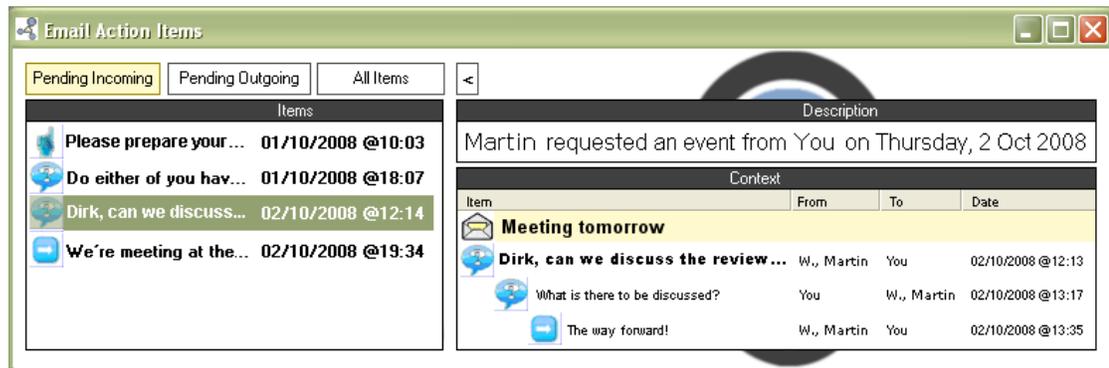


Figure 9.11: Semanta's Workflow-based E-mail Visualisation in Outlook

9.4 A Communication Medium for the Social Semantic Desktop

Although Semanta still functions on a normal desktop, the SSD has the added benefit of desktop data integration, where machine-processable data generated by multiple semantic applications is shared across multiple applications and desktops. The rich knowledge representation models provided by the SSD mean that Semanta's representations can be extended onto the entire personal information model of the user. Thus, workflow artefacts can be linked to other physical and abstract personal concepts, e.g. to a related folder, or project. Additionally, the social aspect of the SSD enables workflow artefact instances to be shared between all the involved persons (and desktops).

This section will discuss the merits of Semanta as a semantic application on the SSD, particularly its role as a *semantics-enabling communication technology for knowledge exchange and integration on and across the platform*, which is the second of the Semantic E-mail vision targets presented in Chapter 7. Through another example, I will demonstrate Semanta's support for semantic email exchange on the desktop, and how it fosters knowledge integration between different applications and desktops. The use case considers an average user – Claudia, who is planning a working trip to Belfast.

9.4.1 Transporting Semantics across Desktops

Using Outlook, Claudia writes an email to Martin asking whether she and Dirk can attend the Belfast meeting. The automatic annotation of the content returns one action item represented by: “*Can we attend the meeting in Belfast?*”. This is represented as a ‘Request for a Meeting to be attended by Claudia, Dirk, addressed to Martin’, using elements in the sMail Ontology, which is one of the developed application-level SSD ontologies.

Excerpts of the metadata generated and transported alongside the email content, when Claudia sends the e-mail, are shown in the named graph (`emailGraph`) in List. 9.1. The graph, serialised in N3 (Section 2.2.4), provides for an alternative RDF view of conventional email properties (e.g., from, to, etc.), as well as richer information about specific action items within (as instances of `smail:SpeechAct`) and their nature. The use of a number of SSD ontologies is evident (NMO, NCO, PIMO; refer to Section 6.2). Also, note that the namespace prefix “`claudia:`” refers to the URI identifying Claudia’s PIMO (which contains all of Claudia’s personal (semantic) desktop information). The PIMO’s URI is fully defined for the graph via `pimo:isDefinedBy`).

When Martin receives the email in his Thunderbird client, Semanta provides him with a number of automated options for reacting to the embedded *Meeting Permission Request*, represented by speech act `claudia:EMA3644F2000SA0 = (Request, Event, {Claudia, Dirk})`. When Martin eventually approves the meeting, Semanta sends an automatic semantic email notifying both Claudia and Dirk that they can indeed attend the meeting.

9.4.2 Representing E-mail Workflows

Claudia’s meeting permission request generated a workflow, which was completed upon Martin’s approval and subsequent notification of his decision to Claudia and Dirk. The metadata generated for the workflow is shown in List. 9.2. In particular, this includes a

```

:emailGraph{
  claudia:EMA3644F2000 a nmo:Email;
    nmo:from :ClaudiaStern;
    smail:hasSpeechAct :EMA3644F2000SA0;
    pimo:isDefinedBy <http://nepomuk.semanticdesktop.org/users/clauidias/pimo>;
    nmo:messageId "000000094A633229025F385F0";
    nmo:sentDate "2010-09-24T13:33:53";
    nmo:to claudia:MartinWilliams .

  claudia:EMA3644F2000SA0 a smail:SpeechAct;
    smail:hasAction smail:Request;
    smail:hasNoun smail:Event;
    smail:hasRole smail:Requestive;
    smail:hasSpeechLength "16";
    smail:hasSpeechStart "53";
    smail:hasStatus smail:Pending;
    smail:hasSubject claudia:ClaudiaStern ,
      claudia:DirkHageman;
    smail:hasTarget claudia:MartinWilliams;
    smail:recipientExpectation smail:Reply;
    smail:senderExpectation smail:Await .

  claudia:MartinWilliams a nco:PersonContact;
    nco:hasEmailAddress claudia:MartinWilliamsEmail;
    rdfs:label "Martin Williams" .

  claudia:MartinWilliamsEmail a nco:EmailAddress;
    nco:emailAddress "martin.williams@deri.org" . }

```

Listing 9.1: E-mail and Speech Act metadata graph in N3 serialisation

reference to the workflow participants (Claudia, Dirk and Martin), to its completed state, and to the sequence of speech acts it entails (via `hasSpeechAct`). As the later consists of only two speech acts, these coincide with the references to the the first and last speech act in the workflow. The workflow also includes a subworkflow, stored separately in another NG. Representations of this workflow will also be found on Dirk’s and Martin’s SSDs.

```

:workflowGraph{
  claudia:EMA3644F2000WF0 a smail:Workflow;
  pimo:isDefinedBy <http://nepamuk.semanticdesktop.org/users/clauidias/pimo>;
  smail:hasParticipant claudia: ClaudiaStern ,
                      claudia: DirkHageman ,
                      claudia: MartinWilliams;
  smail:hasFirstSpeechAct claudia:EMA3644F2000SA0;
  smail:hasLastSpeechAct claudia:EMA3656F2121SA0;
  smail:hasSpeechAct claudia:EMA3644F2000SA0 ,
                    claudia:EMA3656F2121SA0;
  smail:hasStatus smail:Completed;
  smail:hasSubWorkflow claudia:EMA2342F0812WF0. }

```

Listing 9.2: Email workflow metadata in `workflowGraph`

Adhering to the modeling presented in Chapter 7, which stressed the independence of email workflows in relation to email messages, there is a strict separation between representations for the two even at the metadata level. However, this independence does not imply that the two are unrelated. In fact, as shown in List. 9.1 and List 9.2, the metadata for a workflow and the metadata for all the related emails, refers to the same speech acts instances. Abiding by the workflow satisfiability conditions defined in Def. 16, by the time that the metadata for the workflow in List. 9.2 is updated as shown, the status of the speech act (process instance) in List. 9.1 will also have changed to completed.

9.4.3 Supporting Desktop Knowledge Integration and Retrieval

After Claudia, with Semanta’s support, adds the detected appointment to her Calendar, a semantic representation of the event is stored on her SSD. This representation also

includes all the relevant information transported by the semantic email, as well as a link to the email itself. A graphical representation of (a subset of) the triples within the (`eventGraph`) for the generated event is shown in List. 9.3:

```

:eventGraph {
  claudia:EV00966322902 a smail:Event ,
                        nfo:FileDataObject ,
                        ncal:Event ,
                        pimo:SocialEvent ;
  nfo:fileName "A3405BF6640420" ;
  ncal:attendee claudia:ClaudiaStern ,
               claudia:DirkHageman ;
  ncal:dtstart claudia:EVF6640420Start ;
  nao:description "Project meeting in Belfast" ;
  nao:isRelated claudia:EMA3656F2121 ;
  nao:prefLabel "Belfast CID Meeting" ;
  pimo:isDefinedBy <http://nepomuk.semanticdesktop.org/users/claudias/pimo>.

  claudia:EVF6640420Start a ncal:NcalDateTime ;
  ncal:dateTime "10/10/20010 09:00:00" . }

```

Listing 9.3: Partial graph for `eventGraph`

In preparation for the meeting, Claudia creates additional related items, such as documents and tasks. For example, via KASIMIR [Grebner et al., 2008], Claudia creates a dedicated related task. KASIMIR is another SSD application, providing for personal task management. It utilises the Task Management Ontology (TMO), another domain-specific SSD Ontology [Brunzel and Grebner, 2008], that is dedicated to the conceptual representation of tasks. Metadata for this task is stored within `taskGraph`. Once again, this graph includes a link to the event described by `eventGraph` via `nao:isRelated` (refer to Section 6.2.2 for NAO vocabulary).

Like all other information elements represented on Claudia’s SSD, the email-generated event can be viewed from the peer-to-peer Semantic Eclipse Workbench (PSEW)⁴⁶ – an

⁴⁶<http://nepomuk-eclipse.semanticdesktop.org/xwiki/bin/view/Main/PSEW>

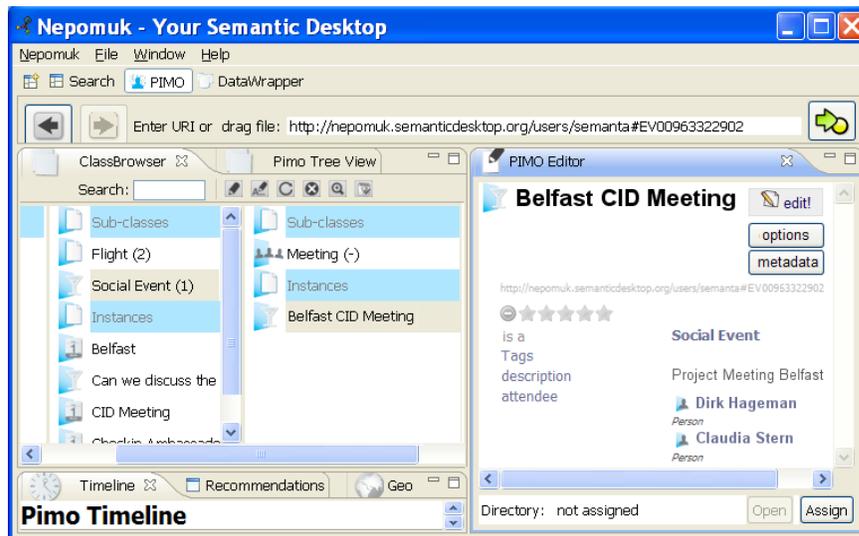


Figure 9.12: Metadata generated by Semanta as viewed on the SSD

integrated environment based on the SSD architecture that allows users to browse, query, share and annotate resources to which they have access. Fig. 9.12 shows how the event represented within `eventGraph` is seen through PSEW. Projects, topics, persons, places, and other important things represented in Claudia's PIMO are inter-linked, enabling her to find relevant information more easily. Relations expressed through the SSD ontologies allow Claudia to step from a task to a connected document, or from a document to an email. This enables knowledge integration and helps users with finding alternative paths to information, using contextual knowledge as a guide. This information retrieval technique, called *Orienteering*, has been found to be the one favoured by users when navigating in between information items [Teevan et al., 2004]. For example, if Claudia remembers the place of a meeting but not the date, she can start from the city *Belfast* and progress to related meetings that way. The technique was also applied by the Gnowsis Semantic Desktop [Sauermann, 2009] (refer to Section 3.1.1).

9.5 Evaluation

The evaluation methodology employed follows the guidelines and recommendations outlined in [Gediga and Hamborg, 2001]. The experiments can be reproduced as all material used for the evaluation, including presentations, scenario outlines, questionnaires, time sheets, videos as well as the results themselves are available online⁴⁷. The evaluation process consisted of two stages:

- In the **Formative** stage, the initial system prototype was improved following a controlled study based on Semanta’s Microsoft Outlook add-in.
- In the **Summative** stage, the improved prototype was under trial by three different groups of collaborating users in their actual day-to-day email-based work. Given the GUI differed slightly for the two extensions, and in view of Thunderbird’s platform independence, this evaluation is based on the latter, to ensure uniformity of results.

Fig. 8 depicts a timeline for the period in between the two stages. The Thunderbird add-on (T) was implemented in full only after the Outlook add-in (O) was improved after the formative evaluation. The implementation included a synchronisation stage for the two different MUA extensions, and a suitable testing period. As a result, the summative evaluation took place a year after the formative evaluation, and its emphasis was on Semanta’s functionalities as provided through its GUI.

9.5.1 Formative Evaluation

This evaluation was concerned with the question “Why is it bad?”, where the goal was to pinpoint the weaknesses of the system in order to improve it [Gediga and Hamborg, 2001]. It employed *behaviour-based* (thinking-aloud, recording, timing) as well as *opinion-based* (questionnaire) descriptive evaluation techniques. The evaluation process and findings reproduced in this Section were published in [Scerri et al., 2008b].

⁴⁷<http://smile.deri.ie/projects/semanta/evaluation/>

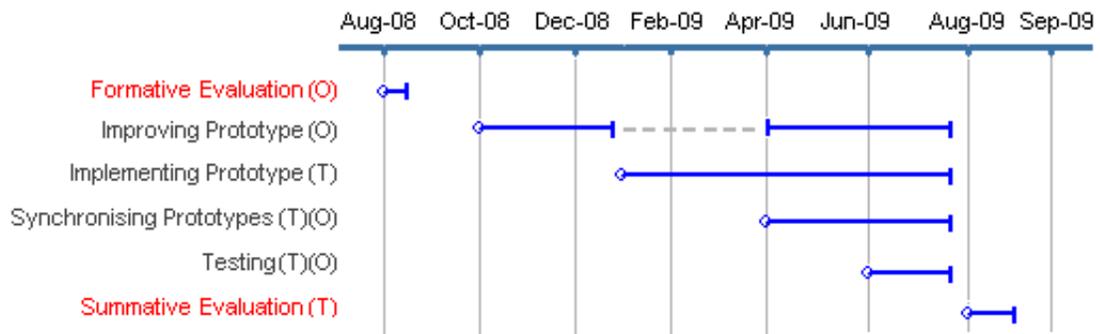


Figure 9.13: Evaluation Process Timeline

9.5.1.1 Method

Participants The evaluation process involved 6 computer scientists who depend on email-based collaboration for their day-to-day work. Although this is not a large number, previous experiments [Dumas and Redish, 1993] conclude that 5-12 users are an acceptable number for a system-usability study. In particular Nielsen et. al. report that 5 users can already find 75% of existing problems [Nielsen, 2000]. Evaluators averaged 32.3 years of age and had above average familiarity with Outlook.

Material Throughout the experiment, the users were timed and their behaviour recorded. This included comments, suggestions, problems encountered and errors. For the purpose, timesheets were designed to support with the evaluation task (available at the evaluation web page provided above). Finally the users were provided with a questionnaire in order to rate Semanta as a whole, as well as specific GUI components (also available online).

Design & Procedure The evaluation took the form of a controlled study in which users faced five characteristic email situations (workflows):

1. Task Delegation
2. Task Notification

3. Data Request
4. Appointment Scheduling
5. Event Announcement

To carry out these tasks, users were instructed to write emails to a number of people, with Semanta’s assistance. Whereas some scenarios required only an email to be sent, others included multiple exchanges, i.e. writing and reading replies, and task and event generation. After this procedure, the users were handed out a copy of the questionnaire.

9.5.1.2 Results & Discussion

Quantitative The evaluation process took an average of 01:23 hours per user. In the questionnaire, users could rate the system as a whole and different aspects of the GUI with respect to consistency, functionality and design. The following are the main findings. The evaluators were quite satisfied with the automatic action item classification. Of the items detected by the text analytics service, 54% were approved by the users without changes, 32% were adjusted via the annotation wizard, and only 13% were either removed or changed completely (i.e. wrong annotations). Finally, of the final number of exchanged annotations, 27% were manually created by the users (i.e. these action items were not detected by the service).

The ratio of time the users required to review and/or add annotations, against the total time required to create and send the email only went down to just above 50% by the end of the experiment. This was not satisfactory as it implies that, on average, users were taking around as much time to annotate the email as to actually write it. However, one must take into account that throughout the evaluation users were giving suggestions and thinking-aloud’, thus considerably increasing the time spent on each annotation task.

As Semanta supports the user with action items tracking, the evaluation included an experiment to gauge how well users could keep track of them without any support. After the users carried out the scenarios, they were asked to list the number of action items

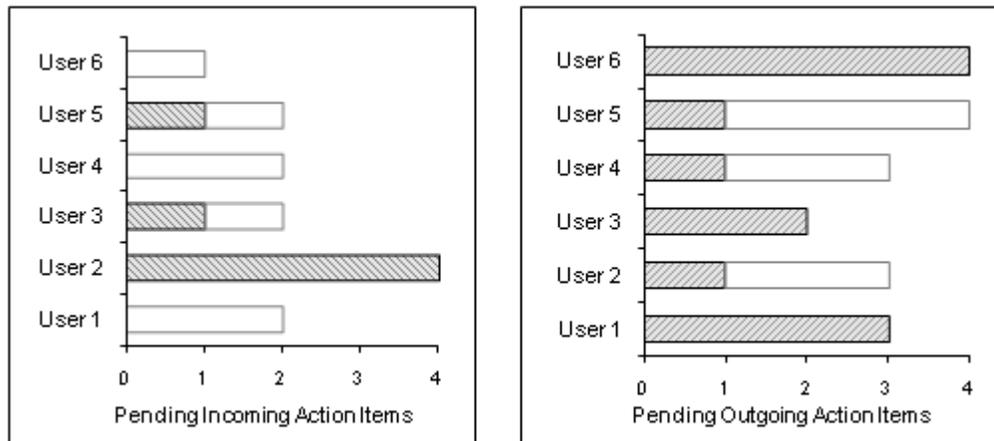


Figure 9.14: Pending action items retrieved (shaded) against total (outlined)

exchanged in the experiment that remain pending, by looking at the exchanged email items in the inbox and sent items folder. The results are interesting, and show that on average, the users retrieved just 33% of total *pending* incoming email action items, i.e. action items that they still needed to take care of (Fig. 9.14), and 65% of the *pending* outgoing email action items, i.e. sent action items for which they are still awaiting a reply. This suggests that users tend to remember pending action items that they have sent to their contacts with twice the accuracy that they remember pending action items addressed to them.

Qualitative The general feeling amongst the evaluators was that the automatic annotation had higher precision than recall. This reflects a decision taken at implementation stage; whereby it was decided that it is better for action items to pass undetected, rather than having Semanta try and support users with inapplicable workflows bound to incorrectly annotated action items. With regards to the annotation wizard, although users demanded more meaningful tooltips for items within, they found its use rather easy and intuitive.

The evaluators were satisfied with the options provided for different incoming action

items (i.e. Semanta’s support for speech act processes and email workflows), when reading incoming semantic email. No cases of users being unable to decide which option to select were observed. Users were very appreciative of the event/task detection, including their partially automatic descriptions, when adding them to their Task list/Calendar. Also appreciated were the links generated between detected tasks/events and their source email messages and the possibility of traversing up an email thread from an email message.

9.5.1.3 Design Improvements

After this evaluation a number of improvements were implemented. The GUI was enhanced with informative tooltips and non-invasive notifications that allows users to understand what is going on while they carry on with their tasks unhindered. New functionalities were also introduced, including the addition of QuickShots (Section 9.3.1), as well as the redesign and implementation of the novel, interactive, faceted-browsing styled email workflow visualisation (Section 9.3.6).

9.5.2 Summative Evaluation

This evaluation was concerned with the questions “How good is it?” and “Which one is better?” [Gediga and Hamborg, 2001]. The first is concerned with the usability of the improved system, whereas the second compares Semanta with an alternative, the latter being the standard MUA in question (Thunderbird). The comparison is relevant within a collaborative setting, as Semanta’s most worthy use-case is the exchange of email in a business, rather than a social environment. This evaluation employed *Usability Testing* descriptive evaluation techniques, including retrieval of user usage statistics and a questionnaire. The evaluation process and findings reproduced in this section were also presented in [Scerri et al., 2010b].

9.5.2.1 Method

Participants The evaluation involved 18 users, collaborating in groups of between 2 and 6 people. The users consisted mostly of researchers in three European universities (including our own) where English is used as the first language; but also included a few industrial partners with whom they frequently collaborate. The background of the researchers included Semantic Web technology, Natural Language Processing, Artificial Intelligence and Social Science.

Materials The evaluators were introduced to the evaluation process via a web page⁴⁸ and supported by a detailed user manual⁴⁹. The evaluation process was wrapped up with a questionnaire (Appendix A) composed of two main parts, each dedicated to the two questions in the introduction to this section. Part one of the questionnaire is in fact a reproduction of the standard USE questionnaire⁵⁰ (Appendix A.1), which measures the usability of the system across four dimensions: *usefulness*, *user satisfaction*, *ease of learning* and *ease of use*. Part two of the questionnaire (Appendix A.2) is dedicated to the second goal question, and tries to quantify the advantages and disadvantages of Semanta over the standard functionalities of Thunderbird. The questionnaire, based on a 7-point Likert scale, allows for three levels of both positive and negative ratings; plus a 'No opinion' option to ensure better results. The two questionnaire sections were succeeded by an important summative question (discussed later), two required fields for Semanta's worst and best features, a field for additional desired features, and another for general feedback.

Design The setup proved to be non-trivial, as a controlled study would have compromised the realistic use of an enhanced email client. Having opted for an uncontrolled

⁴⁸<http://smile.deri.ie/projects/semanta/semantaevaluation2009>

⁴⁹<http://smile.deri.ie/projects/semanta/usermanualthunderbird>

⁵⁰www.stcsig.org/usability/newsletter/0110_measuring_with_use.html

evaluation, another limitation surfaced right away: up to three quarters of Semanta's features can only be appreciated during the exchange of semantic email between Semanta users. Therefore, the hypothesis in question - *that the use of Semanta improves the email experience over the use of a standard, conventional email client*; needed to be tested in an environment where all evaluators were using Semanta. As a result, it was necessary to recruit groups, rather than individual users, who collaborate regularly via email. As only email exchanged between established Semanta users is semantic, the system learns of other users through an identifier in the email headers. This also means that the evaluation process did not interfere with email exchanged outside of the evaluation groups.

Although summative evaluations tend to include a lot of interactive behaviour-based scenarios, this was not practical in this case, due to the following reasons:

- For the reason given earlier, the nature of the study was uncontrolled;
- Half of the evaluators were using Semanta remotely, making it hard to assist them physically and record their behaviour;
- Privacy issues.

The latter reason was a matter of concern for many, which we addressed by letting the evaluators themselves view and send back their usage statistics, in an email at the end of the evaluation. This email and its contents were automatically generated through a set of queries to Semanta's RDF store. The statistics helped to understand the results of the survey better, including e.g. how many email workflows Semanta supported the user with, the number of incoming and outgoing action items which remained pending, the number of semantic email exchanged, the number of users with whom they exchanged semantic email, how long they used Semanta for, the number of events and tasks generated, etc.

Procedure The resulting user groups were instructed to use Semanta for at least 10 days. We carried out the evaluation within the group from our university first, to

detect any problems with the process and be able to discuss the experiences in person. Any detected problems, issues and bus were reported accordingly

After a total of 14 days, the remaining users were asked to also submit their usage statistics (via the automatic email).

9.5.2.2 Results & Discussion

Quantitative A summary of the averaged usage statistics is presented in Table 9.1. The user group size averaged 4 (2.83 plus the user). As noted, a few users opted to keep using Semanta after day 10, resulting in an average total usage duration of almost 12 days. An average of 40 action items (speech acts) were exchanged in 30 emails. Of these, 7 of the incoming, and 9 of the outgoing remained pending. An average of 3 tasks and 2 events were recognised and exported by Semanta to the task list/calendar, as appropriate.

Table 9.1: Average Usage Statistics

Semantic Email exchanged	29.29
Speech Acts Exchanged	40.43
Incoming Action Items that remain pending	6.57
Outgoing Action Items that remain pending	9.29
Corresponding Semanta users	2.83
Duration of use (statistics collected on day 14)	11.85
Email extracted tasks	3.29
Email extracted Events	2.14

The results of the questionnaire's first part are averaged in Fig. 9.15. Although users appeared to have been satisfied with Semanta's features, their ratings for its usefulness are a bit disappointing, albeit still positive. The other main finding here is that although it is not always straightforward to use Semanta, it is quite easy to adjust to.

The second part of the questionnaire tried to quantify the performance of Semanta over the standard Thunderbird. Questions were based on a 7-point Likert scale, here ranging from -3 (Predominantly worse) to 3 (Predominantly better), with 0 signifying no

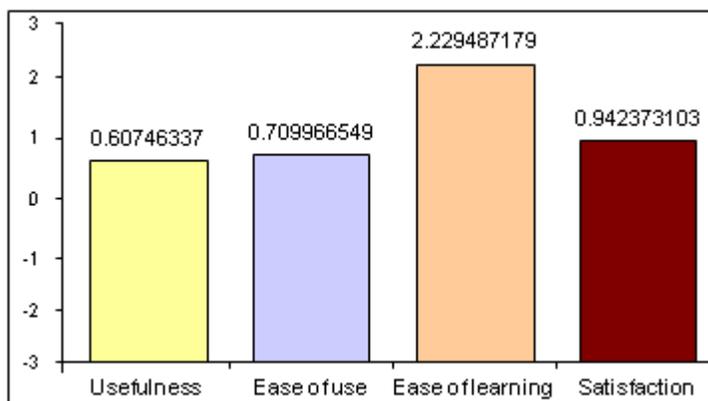


Figure 9.15: USE Questionnaire results

perceived changes (thus, ratings for Thunderbird are zero by default). One-sample t-tests (two-tailed, 99% confidence interval) were then performed to interpret the ratings. The following are the highlights⁵¹, summarised in Table 9.2. The first result (Table 9.2-H1) rejects the hypothesis that the same amount of time is required to write email with or without Semanta. A lengthier writing time was expected due to the annotation reviewing stage. However, the users feel that this stage does not harm the email writing process, as shown by the acceptance of H2. Optional user comments suggest that although some see it as an annoyance, others like the idea of annotating email, as it helps getting things done. H3 was rejected, showing that the flexibility of email replies was somewhat jeopardised by Semanta. In fact, in an additional best and worst feature fields in the questionnaire, the email reply interface got the highest number of negative votes. H4-H5 were rejected in Semanta's favour, concluding that keeping track of both incoming and outgoing action items is significantly easier with Semanta. Finally H6's rejection implies that the workflow-based email visualisation was perceived as useful. Additional results confirmed that users appreciate Semanta's ability to link tasks and events to the email threads wherein they were generated and the possibility to traverse individual messages in a thread. This was expected, given that the standard Thunderbird lacks such features.

⁵¹Full results are available via the earlier provided evaluation page

Table 9.2: Hypothesis Testing for Semanta’s added support over Thunderbird

<i>Null Hypothesis H_0</i>	<i>Mean</i>		<i>T-test</i>	<i>Outcome</i>
	<i>T.B.</i>	<i>Semanta</i>		
H1 : No change in time required to write email	0	-0.75	-3.000	Rejected
H2 : Annotation process doesn't effect email writing experience	0	-0.08	-0.173	Accepted
H3: Flexibility of email replies is not effected	0	-0.85	-3.091	Rejected
H4: Difficulty of keeping track of pending received action items is unchanged	0	2	11.015	Rejected
H5: Difficulty of keeping track of pending sent action items is unchanged	0	2	11.832	Rejected
H6: No effect on the mental visualisation of email workflows	0	2	6.36	Rejected

Qualitative The questionnaire ended by posing the following summative question: “*Are Semanta’s functionalities worth the effort to review automatic annotations or manually create them?*”.

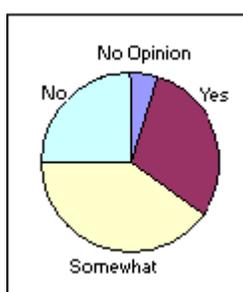


Figure 9.16: Results of summative question

Results, shown in Fig. 9.16, suggest that whereas the majority of users felt that the time sacrificed reviewing email annotations was worth the subsequent email support, to one extent or the other, around 25% thought otherwise. This can perhaps be summed up by the following general comment provided by an evaluator: “leaving aside the fact that Semanta is a research prototype, for a new email tool to be accepted by a broad set of users, it will need to provide benefits that are at multiple orders of the additional cost it imposes on them”.

9.6 Conclusions

This Chapter provides a proof of concept for the vision projected in this thesis via Semanta: a user-supportive extension for popular email clients. This proof of concept is supplemented by a portrayal of the system's capabilities as:

- an intelligent workflow management application that supports email-based collaboration and reduces e-mail overload
- an appropriate communication medium for the Social Semantic Desktop enabling the exchange and integration of semantic knowledge, on and across the platform.

The pursued approach strives to identify and place patterns of email communication into a structured form, in order for machines to support users with email (ad-hoc) workflow management. In turn, this knowledge is employed to reduce the epistemological gap between the way users perceive collaborative workflows, and the fragmented way in which these are currently displayed' on the desktop. The semantic representation of email workflows exchanged over semantically-enabled email technology, allows for machine-processable representations of the same information items across multiple desktops, thus enabling the integration of shared knowledge on the SSD⁵².

Through its multi-tiered architecture, Semanta successfully decouples the business logic from the user interface, meaning that both current (and any future) email client extensions are able to provide the same functionality, albeit maybe via a slightly different UI. The UI itself provides for intelligent supportive features, powered by the underlying metadata pertaining to the structure, content and context of email messages. Whereas the introduced functionalities are clearly visible to the end-user, the complex models and

⁵²In reality, this is true only in concept. In practice, for representations to be truly shared, a unique URI must be used. In this case, although the URI fragment is unique, the scheme-specific part of the namespace is bound to each desktop wherein it is shared. For this issue to be addressed, a standard Desktop URI Scheme that also takes into account inter-desktop URIs is necessary. Although a request for comments (RFC) was drafted by the NEPOMUK project, at the time of writing this thesis this scheme remains a draft, retrievable from <http://aperture.wiki.sourceforge.net/SemdeskUri>

theory behind are successfully hidden behind the intuitive UI. Semanta seamlessly integrates semantic technology into the existing technical landscape, using existing transport technology and email clients, thus largely eliminating the need for users to get used to new methods and/or applications. Thus the introduced semantic email technology is within the reach of anyone using a personal computer.

The two-staged evaluation verified the acceptance and applicability of the UI by way of an experimental case-study. The hypothesis under test was that if average email users are willing to sacrifice a little extra time to the email writing process, in order to review automatic email action annotations, they are in return supported by a system that:

- is aware of the existence and status of action items within email,
- supports users with reviewing incoming action items and a semi-automatic provision of their replies,
- provides an alternative email visualisation, based on action item threads within email workflows, that is more akin to what the users conceptually perceive when carrying out their email tasks,
- detects tasks and events generated within email workflows,
- enables the integration of workflow artefacts scattered on and across multiple social semantic desktops, and
- provides other useful secondary features such as file attachment reminders.

The results of the summative stage of the evaluation suggest that although Semanta provides additional support to the collaborative email user over standard email clients, this support comes at a cost, i.e. the annotation reviewing process for a first email in a thread. The extent of this cost is subjective – whereas some were more than willing to spend a little extra time reviewing and adjusting annotations, others considered it as yet another email chore.

In conclusion, Semanta demonstrates how semantic technology can enable automated support for email-based collaborative work across multiple desktops. The tool was widely appreciated, although it has to be acknowledged that in order for Semanta to jump over the research fence into the real world, the extra cost imposed on the user needs to be further reduced. It is hoped that the future work outlined in the next Chapter, especially Section 11.4, will amongst others, enable Semanta to reach this goal.

Part IV

Conclusion

10 Summary

This chapter outlines the major contributions of this thesis by revisiting the research questions provided in Section 1.2.

Can a common knowledge representation, with the adequate level of expressivity, be established across multiple social semantic desktops? Within the ontology engineering team of the Social Semantic Desktop [Reif et al., 2008], I have contributed to the engineering of comprehensive knowledge representation models. These models enable the level of expressivity required to support semantic interoperability and application data integration across a network of semantic desktops. The adopted modelling primitives reflect the desktop user’s mental models and usage habits, applied throughout and lifted up to the highest (abstract) levels of desktop knowledge representation. The most important contribution was the engineering of a novel representational language – NRL (Section 6.1), to regulate and constrain the general expressivity of the whole system. This language fulfils all the requirements arising from the distributed knowledge representation, and heterogeneity aspects, of the SSD scenario, which were not satisfactorily supported by the existing state-of-the-art.

How to comprehensively target the Email Overload problem? Email can be considered as a virtual extension to the collaborative workers’ working environment, wherein they carry out shared tasks, etc., generating and sharing new personal information in the process. From this perspective email overload can be considered as a workflow management problem where, faced with an increasing amount (and complexity)

of co-executing workflows, users become overwhelmed and lose track of their email-based collaborative work. This thesis demonstrates how, by adopting a workflow-approach to email management, the *source* – rather than the *effects* – of most problems attributed to email overload can be effectively reduced.

How can one model and represent email workflows? The pursued approach identifies and places patterns of email communication into a structured form. Email messages are broken down into a set of *action items* (e.g. task assignment, meeting proposal, etc.). Implicit sequences of related action items exchanged over an email thread are then exposed as well-defined *ad-hoc email workflows* (e.g. task delegation, meeting scheduling). Representations of specific action items, as well as the entire ensuing workflows, are provided by the *Semantic E-mail Conceptual Framework* (chapter 7). The two, extensible, major components of the framework are:

1. A ***Speech Act Model***, which models specific email action items, based on conceptualisations deriving from speech act theory [Austin, 1962].
2. An ***Email Workflow Model***, which interprets sequences of exchanged email action items as independent, but concurrently executing ad-hoc email workflows. The model is based on a study of real action item sequences in an email corpus. The study attests to the hypothesis that, although generally unpredictable, a significant percentage of email conversational patterns are routinely exchanged between participants. However, whilst catering for frequently occurring email communication processes, the designed workflow model also respects email’s characteristic flexibility – by supporting any other conversational move during all stages of an email workflow.

Can machines be enabled to work with workflow representations? The conceptualisations provided by the above framework are captured within the semantic email ontology *sMail* (section 7.3). The use of semantic technology ensures that a shared

understanding of the underlying collaboration is established between both humans and machines. Furthermore, concepts in the ontology derive from the ontologies engineered for the *Social Semantic Desktop* (chapter 6). Grounding the sMail ontology within existing knowledge models enables desktop data integration - whereby machine-processable representations of email workflows and their artefacts (e.g. messages, action items, contacts, tasks, etc.) can be shared across multiple desktop applications, and entire networks of social semantic desktops.

To what extent can new email workflows be automatically elicited? To be able to support the user with executing workflows, the implemented communication support system needs to first become aware of those action items which initiate them. This knowledge-acquisition bottleneck problem is addressed by applying computationally treatable aspects of speech act theory, to enable the automatic classification of email action items. A rule-based classification model classifies email segments into exactly one of the action items (or speech acts) provided by the speech act model (section 7.1.1). An evaluation of this information extraction technique suggests that whereas complete automation is not very reliable, the results are satisfactory for the presentation of user-reviewable suggestions requiring a minor cognitive cost. Once Semanta is aware of the first action item in a workflow, further workflow knowledge is indirectly elicited through the user's interaction with the user interface (which is powered by the workflow model in section 7.2.4), on the fly as the workflow progresses.

Can workflow support be provided without disrupting conventional email practices and user habits? The employed semantic technology is integrated seamlessly with existing email transport technology. Additionally, a semantic user interface providing the intelligent features, is implemented on top of existing email clients. Whereas workflow management and visualisation is provided *alongside* that conventionally provided for email messages, workflow processing is performed *in parallel* to habitual email use, such as reading and replying to email. When eliciting workflow knowledge solely

through user interaction is not satisfactory, the user interface ensures that the required knowledge is obtained at the minimal cost to the user. At all stages, user interface interaction is user-driven, ensuring that workflow support is provided only when desired, and never perceived as disruptive.

Can semantic email serve as an adequate communication channel for the Social Semantic Desktop? The email workflow model presented in section 7.2.4 has a concrete manifestation in the creation and organisation of semantic desktop data. This is shown in section 9.4, which demonstrates the capabilities of semantic email (using *Semanta* as a medium) as a means for integrating shared desktop items that are generated *within* email (e.g. tasks and events). Thus, semantic email targets a previous vacuum, whereby email data was only (and less accurately) semantically lifted and represented at a later stage, by means of techniques such as data crawling. Additionally, these representations are instantly and automatically shared between all workflow participants, abolishing the need to map different representations of the same (shared) items on multiple desktops.

Can the envisaged communication support system aid the users with the better management of their email-based collaborations? The proof of concept for the vision of an intelligent workflow management application that supports cross-desktop, email-based collaboration is provided via *Semanta* – a tool that extends popular email clients to enable semantic email (chapter 9). Evaluation results indicate that *Semanta* provides additional support to the collaborative email user.

The models and technology embodied by the research in this thesis enable *Semanta* to identify and structure email communication processes. This knowledge is then employed to reduce the epistemological gap between the way users *perceive* their collaborative workflows, and the fragmented way in which these are currently *presented* on their desktop. The semantic, machine-processable, representation of workflows and their artefacts are shared between multiple desktop applications and entire networks of semantic desktops. This enables knowledge integration both on and across SSDs. As a semantic communi-

cation support system, Semanta:

- semi-automatically becomes aware of the existence and status of email action items;
- supports the user with handling incoming action items and provides for their semi-automatic replies;
- introduces a novel, supplementary, workflow-oriented visualisation of email, which is based on action item sequences within email threads, and that is more akin to the user's mental model of the email collaboration;
- automatically detects tasks and events generated within email (workflows) messages;
- enables the integration of workflow artefacts scattered across multiple personal desktop applications, as well as entire networked desktops;
- provides other useful secondary features, such as file attachment reminders.

Whereas the introduced functionalities are clearly visible to the end-user, the complex modelling behind is successfully hidden underneath an intuitive user interface. Furthermore, because Semanta seamlessly integrates semantic technology into the existing technical landscape, it is within the reach of anyone using a personal computer. Evaluation results also suggest that although Semanta provides additional support over standard email clients, this support comes at a slight cost, i.e., the action item suggestion reviewing process, for a first email in a thread. The extent of this cost is subjective – whereas some people are happy to spend a little more extra time reviewing action item annotations, in view of the ensuing support, others considered it as yet another email chore.

In conclusion, the research covered by this thesis demonstrates how semantic technology can enable automated support for digital collaborative work taking place across a network of desktops, reducing the negative effects of email overload by easing interpersonal information management. The results also imply that, given the right resources, there is a huge potential for semantic communication support systems to break out of the lab and become a beneficial part of everyday use.

11 Future Work

There are numerous possibilities for future work to be undertaken as a continuation of the research contributions presented in this thesis. These possibilities fall under a number of different directions, such as knowledge modeling representation on the Social Semantic Desktop and ontology dissemination, extended e-mail workflow modeling, improved methods for workflow information extraction, Web science, as well as the addition of more functionality and an enhanced UI in Semanta.

11.1 The Social Semantic Desktop Ontologies

Within the Nepomuk project's knowledge modeling team, we have concerted our efforts towards the dissemination and standardisation of the engineered ontologies presented in Chapter 6. The Open Semantic Collaboration Architecture Foundation⁵³ (OSCAF) was setup in order to keep the ball rolling after the end of the project (December, 2008). It's aim is to bring together organisations and individuals interested in ensuring interoperability between desktops and collaborative environments, and to establish common standards in collaborative software. With the development and dissemination of ideas for enabling advanced personal information management and collaboration at its heart, OSCAF presents a number of opportunities for future work. The published ontologies can be further extended to cover other aspects which were not relevant for the SSD, e.g., complex concepts like active contexts and privacy modelling, which are crucial in ubiquitous environments involving mobile devices. Additionally, the ontologies can be integrated

⁵³<http://www.oscaf.net/>

with emerging ontologies, e.g., conceptualisation in the annotation ontology (NAO) can be integrated with the tagging ontologies introduced in Section 3.1.2, to provide for an even more powerful semantic expression of knowledge in this domain. Semantic Email would then be one of the first technologies to take advantage of this extended modeling, to provide for the semantic tagging of email, with the intent of further enhancing the email experience via the introduction of (semantic) tag-based information retrieval.

11.2 Semantic E-mail Conceptual Framework

Although the framework as a whole was originally targeted at the email communication domain, it can easily be extended to apply to collaborative work that takes places over other forms of electronic communication, such as instant messaging. In fact, instant messaging (IM) is very similar to email communication, except for the fact that it is synchronous in nature and thus ambiguities can be cleared instantly. Otherwise, the differences are mostly superficial, with people engaging in ‘conversational turns’ rather than email message exchanges, and a more widespread use of informal ‘text-speak’. However, the workflow modelling in place for email still applies. Questions posed over IM can still be ignored or not given due attention; tasks are assigned and events are planned in the same way as over email. Thus, there is a good possibility to experiment with the presented models so that they can also be applied to this domain.

The SEW workflow model is completely extensible and new patterns can be introduced without considerable effort, to, for example, accommodate custom workflow patterns that or of use to a particular user or team. Given its flexible nature, it can also be implemented on top of any existing protocol and email client. The integration of the workflow model within widely-used email services, such as Gmail, is of particular interest.

From a more formal point of view, another possibility for the future is the mathematical analysis of the SEW model’s properties, to determine the complexity of various patterns. An analysis of particular email scenarios and their complexity was performed in the earlier semantic email research by McDowell [McDowell et al., 2004]. In addition, formal

methods can also be applied to determine which of the model's properties can be proved, particularly the defined satisfiability conditions.

The presented models can also be combined with other established ideas and methods in relation to email personal information and workflow management. One such method is *Getting Things Done* (GTD) – a time/action management method devised by David Allen, which places particular attention on the context of tasks to be managed [Allen, 2002]. Driven by the stress and anxiety experienced by knowledge workers trying to keep up with their daily work, GTD offers recommendations about how to handle incoming information and efficiently carry out any ensuing tasks. This method is particularly relevant to this work given that it includes a workflow process [Heylighen and Vidal, 2008], used to gain control over all tasks and commitments which one needs, or wants, to get done. GTD has already been applied to the email domain⁵⁴, and the contributions of this thesis might benefit such initiatives.

11.3 Speech Act Classification

The evaluation of the implemented speech act classifier exposed a lot of room for improvement. Whereas the majority of these would only achieve a negligible, or very slight, increase in the classification score, two specific linguistic problems are of specific interest and have accordingly been singled out as high-priority future work. The first is the effect of *Conditional Modifiers*, i.e. clauses which change the semantics of a preceding (or succeeding) speech act in the text. This is a problem of context, and requires more than the (majorly) shallow, employed parsing techniques. Although the existing GATE speech act pipeline (Fig. 8.2) includes grammar for the purpose, these are very basic and can only handle the most obvious ‘if-then’-type clauses. The second problem that needs immediate attention is that of *Coreference Resolution*, because, for example, if an email is addressed to multiple recipients, for each elicited speech act it is required to determine who the subject (if an activity is implied) and/or target (to whom it is addressed) is. This

⁵⁴<http://chandlerproject.org/>

is not always straightforward, as person names are not used throughout, and it might be required to match a ‘she’ or a ‘you’ to specific contacts. This problem has already been given some consideration, and a basic grammar which chains anaphora to a subject (i.e. a person name) in the text is also included in the pipeline. However, this does not perform well enough and needs to be improved, as the extraction of the correct 3rd speech act parameter – the speech act subject – is highly dependant on it.

The Information Extraction (IE) approach employed to construct the speech act classifier is Knowledge-Based (KB). The counterpart to this method is Machine Learning (ML). The choice of the former was driven by the absence of sufficient gold standard training data. As the manual annotation of a vast corpus of email data with speech acts is very time consuming, the more interesting language engineering path was chosen. In general KB systems are perceived as achieving high precision, while ML approaches are more oriented towards higher recall. However, mixing both approaches is considered best practice. Thus one of the future plans is to improve the automatic classification, by using the existing KB classifier to bootstrap an ML system. For this purpose, a gold standard training dataset can be generated from the rule-based annotations, and subsequently manually corrected to ensure higher quality. An ML-trained classifier can then be applied to annotations derived from the existing linguistic processing resources. This includes both the shallower resources (e.g. Tokeniser, POS tagger, NE-lookup; refer to Chapter 8) as well as the intermediary annotations (e.g. Gazetter-lookup entries). If this ML classifier proves successful, the existing rule-based approach will only execute as a fallback, when it fails. Candidate ML solution that can be employed include the GATE ML plugin, which can load different ML engines such as the LibSVM package⁵⁵; the PAUM algorithm [Li et al., 2002]; as well as the open source ML package, Weka [Witten and Frank, 2000].

One of the future work possibilities outlined in the previous section involves the extension of the scope of this research to also apply to IM. The classification system will there-

⁵⁵A library for support vector machines. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>

fore need to cater for speech acts within electronic chats. There have already been a number of contributions in this area, mostly based on ML and statistical approaches. In particular, Fišel discusses ML techniques for dialogue act recognition [Fišel, 2007], whereas Forsythand and Martell [Forsythand and Martell, 2007] build a chat corpus tagged with lexical, syntactic and discourse information; for classification via statistical-based NLP. Although, in principle, the same classifier can be equally applied on textual content in both email and chat conversations, one problem which already effects speech act classification in email is more pronounced in IM. In fact, the form of language used in IM is significantly less formal than that for email [Quan-Haase et al., 2005]. As a result, the phenomenon of text speak', i.e. the use of non-standard English words and language, limits the performance of the classification rules. This was also observed in a preliminary experiment in which the current rule-based classifier was executed over chat content. Although the envisaged ML support will partly solve this problem, there is still a need to investigate how the flexibility of our pipeline can be extended for the purpose.

Another future enhancement to the employed IE technique will focus on the recognition of additional useful information, with the better integration of workflow artefacts in mind. For example, person names in the conversational text can be matched to contacts in the address book, specific dates and times associated with event or task deadlines, etc. A similar IE method for identifying event dates and times was implemented in Horvitz's LookOut mixed-initiative system [Horvitz, 1999]. The extracted relationships can then also be stored semantically (e.g. via the use of NAO, Section 6.2.2) for a better integration of workflow artefacts, which can then be utilised later for an improved retrieval of related information elements.

11.4 Semanta

After the summative evaluation of Semanta, a number of future improvements were outlined. Firstly, new ways to improve the least attractive features, as rated in the evaluation, need to be found. For example, the current presentation of applicable operations for an

action item (given the workflow model) in an incoming email, needs be improved, so as to ‘fit’ more naturally in the email replying task. Also, the workflow visualisation needs to be extended so as to enable users to quickly send reminders to their contacts, when specific outgoing email action items remain pending for a period of time. This process could also be automated, as per the user’s preference. Email reminders can themselves be incorporated in the SEW model, as a special type of email message. Furthermore, workflow views can also be extended such that, when a workflow ends with a generated event/task, these are also incorporated in the view. In the case of tasks, their status would be dynamically updated when the implicated workflow participant(s) updates it as such. For example, after Claudia accepts a *Task Request* from Dirk, it would be shown as a pending task at the end of the workflow, in both Claudia’s and Dirk’s workflow visualisation. When Claudia eventually ticks the task done, the task is then marked as completed not only on Claudia’s desktop, but also on Dirk’s, via an automatic workflow update sent over email.

Secondly, the realisation of future work outlined in the previous sections will enable the further integration of information elicited by Semanta with that stored on the SSD. The enrichment of semantic email with more metadata, such as references to specific tags/user topics, contacts, etc., can enable more cohesion between workflow artefacts and other items on the SSD. For example, an email workflow between Claudia and Dirk, tagged with ‘Belfast Meeting’ and resulting in the generation of multiple tasks and event, will enable intelligent desktop applications to link these items to others on the SSD. A number of documents attributed the same tag, as well as their respective folders, can in this way become automatically associated with the email and its artefacts, presenting a more ‘complete’ desktop information view to the user. Apart from improving both personal information management and retrieval, this also extends the vision, discussed in Section 9.1 and illustrated by Fig. 9.1, to reduce the epistemological gap between the way users mentally perceive their digital personal information, and the fragmented way in which related information items are currently scattered around on their desktops.

Thirdly, a major required improvement of the system is to enable it to work with email workflows which involve participants that are not using Semanta. As things stand, although any (semantic or otherwise) email client can receive semantic email, only ones for which Semanta is provided can operate on it. If the second of two corresponding authors is not using Semanta, they will be provided with very limited support. To rectify this limitation, incoming (non-semantic) email will need to also be mined for action items. This introduces new challenges as, lacking any semantics associated to an incoming (non semantic) email, Semanta will need to work out to which thread this message belongs to. Even more arduous, will be the task of determining whether any of the detected incoming speech acts are initiating new workflows, or whether they are updates to ones that are already executing. This problem is perhaps worthy of an entire separate dissertation.

11.5 Web-Science: Inducing take-up of Semantic Technology

Web Science is an interdisciplinary field dedicated to the understanding of the evolving World Wide Web and engineer its future in a way that ensures its social benefit [Hendler et al., 2008]. As one of the foreseen future directions for the Web, the Semantic Web plays an important role in this new field. That said, the Semantic Web only remains relevant if a social benefit is guaranteed. As opposed to the simpler-to-grasp and more hands-on Web 2.0 technology, proving the benefits of Semantic Web technology to the world out there is a much harder task, due to its higher complexity and the feeling of a lack of instant gratification that is frequently involved. This problem has also affected Semanta, and is the basis for the summative question provided at the end of its evaluation - is the required (semi-automatic) semantic annotation stage worth the ensuing support?

That said, semantic applications such as Semanta are the Semantic Web's best odds to be taken-up by the computer-literate society at large. By enhancing and supporting everyday computer usage through semantics, the benefit can quickly be experienced by a large number of the existing user base. However, as shown by this experience, this is still not a straightforward task. Additionally, the challenge lies beyond getting one's

semantic applications adopted and widely used. From the point of view of the Linked Open Data (LOD) initiative [Bizer et al., 2007], the challenge also lies in generating a sufficient amount of relevant data that is interconnected to external datasets, such that other semantic web applications are able to process the entire data cloud for the user's benefit. For this to be possible, ontologies underlying different applications (such as the sMail ontology in Semanta) are required to be recognised as standards. The dissemination and standardisation (refer to Section 11.1) of all engineered ontologies covered by this thesis is here also crucial, to give a social dimension to the valuable scientific and engineering contributions presented in this thesis. Yet, this remains a challenging but known problem, which has been faced by other similar earlier initiatives with mixed results. One ontology whose dissemination and use is deemed to have been successful is the one behind the Semantically-Interlinked Online Communities (SIOC) initiative [Bojars et al., 2008], whose aim is to enable the integration of online community information. The SIOC ontology achieved a significant adoption rate, and has become a standard (endorsed by W3C) way for expressing user-generated content in online communities, thus allowing for semantic applications to be built on top of the existing Social Web. We would like to emulate this success, and target the adoption of the sMail ontology not only by our 'private' semantic plugins, but also by, for example, other existing public email services on the Web.

Another future objective related to this discussion is the further extension of the SSD's social aspect, such that generated desktop data is linked directly to external datasets, thus also playing a part in increasing the Semantic Web's cohesion. As it stands, metadata generated by semantic applications such as Semanta can easily be retrieved and operated upon by other semantic applications across the SSD. In the future we also want to consider unidirectional links from the desktop to the Semantic Web, e.g. linking a private meeting scheduling workflow to an existing public representation of a conference in the LOD cloud.

Bibliography

- [Adar et al., 1999] Adar, E., Karger, D., and Stein, L. (1999). Haystack: Per-user information environments. In *Proceedings of the 1999 Conference on Information and Knowledge Management, CIKM*.
- [Adida et al., 2008] Adida, B., Birbeck, M., McCarron, S., and Pemberton, S. (2008). Rdfa in xhtml: Syntax and processing. Recommendation, World Wide Web Consortium. Available at: <http://www.w3.org/TR/rdfa-syntax>.
- [Aery and Chakravarthy, 2005] Aery, M. and Chakravarthy, S. (2005). emailsift: Email classification based on structure and content. In *ICDM*, pages 18–25. IEEE Computer Society.
- [Alexakos et al., 2006] Alexakos, C. E., Vassiliadis, B., Votis, K., and Likothanassis, S. D. (2006). A multilayer ontology scheme for integrated searching in distributed hypermedia. In Sirmakessis, S., editor, *Adaptive and Personalized Semantic Web*, volume 14 of *Studies in Computational Intelligence*, pages 75–83. Springer.
- [Alexandersson et al., 1998] Alexandersson, J., Buschbeck-Wolf, B., Fujinami, T., Kipp, M., Koch, S., Maier, E., Reithinger, N., Schmitz, B., and Siegel, M. (1998). Dialogue Acts in VERBMOBIL-2 - Second Edition. Verbmobil Report 226, German Research Center for Artificial Intelligence (DFKI), Saarbruecken.
- [Alexandersson et al., 2000] Alexandersson, J., Engel, R., Kipp, M., Koch, S., Küssner, U., Reithinger, N., and Stede, M. (2000). Modeling negotiation dialogs. In Wahlster,

- W., editor, *Verbmobil: Foundations of Speech-to-Speech Translation*, pages 441–451. Springer.
- [Allen, 2002] Allen, D. (2002). *Getting Things Done: The Art of Stress-free Productivity*. Piatkus Books.
- [Alm et al., 1992] Alm, N., Arnott, J. L., and Newell, A. F. (1992). Prediction and conversational momentum in an augmentative communication system. *Communications of the ACM*, 35(5):46–57.
- [Alston, 2000] Alston, W. P. (2000). *Illocutionary Acts and Sentence Meaning*. Cornell University Press, Ithaca, N.Y.
- [Armstrong, 2000] Armstrong, E. (2000). What’s wrong with email? Available at: <http://www.treelight.com/software/collaboration/whatsWrongWithEmail.html>.
- [Austin, 1962] Austin, J. L. (1962). *How to do things with words*. Harvard U.P., Cambridge, MA, USA.
- [Baader and Nutt, 2003] Baader, F. and Nutt, W. (2003). Basic Description Logics. In Baader, F., Calvanese, D., McGuinness, D., Nardi, D., and Patel-Schneider, P., editors, *The Description Logic Handbook: Theory and Implementation and Applications*, pages 47–100. Cambridge University Press.
- [Baeza-Yates and Ribeiro-Neto, 1993] Baeza-Yates, R. A. and Ribeiro-Neto, B. (1993). *Modern Information Retrieval*. Addison-Wesley, New York.
- [Barbuceanu and Fox, 1995] Barbuceanu, M. and Fox, M. S. (1995). Cool: A language for describing coordination in multi agent systems.
- [Beckett, 2004] Beckett, D. (2004). RDF/XML syntax specification (revised), W3C recommendation. *World Wide Web Consortium*.

- [Beckett and Berners-Lee, 2008] Beckett, D. and Berners-Lee, T. (2008). Turtle - terse RDF triple language. Team Submission, World Wide Web Consortium. Available at: <http://www.w3.org/TeamSubmission/turtle/>.
- [Bellotti et al., 2002] Bellotti, V., Ducheneaut, N., Howard, M., and Smith, I. (2002). Taskmaster: recasting email as task management. In *Proceedings of the CSCW Re-designing Email for the 21st Century Workshop*.
- [Bellotti et al., 2003] Bellotti, V., Ducheneaut, N., Howard, M., and Smith, I. (2003). Taking email to task: the design and evaluation of a task management centered email tool. In Cockton, G. and Korhonen, P., editors, *CHI*, pages 345–352. ACM.
- [Berners-Lee, 1989] Berners-Lee, T. (1989). Information management: A proposal. <http://www.w3c.org/History/1989/proposal.html>. Available at: <http://www.w3c.org/History/1989/proposal.html>.
- [Berners-Lee, 2006] Berners-Lee, T. (2006). Linked data - design issues. Available at: <http://www.w3.org/DesignIssues/LinkedData.html>.
- [Berners-Lee and Cailliau, 1990] Berners-Lee, T. and Cailliau, R. (1990). Worldwideweb: Proposal for a hypertext project. Proposal, CERN.
- [Berners-Lee and Connolly, 2008] Berners-Lee, T. and Connolly, D. (2008). Notation3 (n3): A readable rdf syntax. Team submission, World Wide Web Consortium. Available at: <http://www.w3.org/TeamSubmission/n3/>.
- [Berners-Lee and Fischetti, 1999] Berners-Lee, T. and Fischetti, M. (1999). *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web by its Inventor*. Harper, San Francisco.
- [Berners-Lee et al., 2001] Berners-Lee, T., Hendler, J., and Lassila, O. (2001). The Semantic Web. *Scientific American*, 89.

- [Biron and Malhotra, 2004] Biron, P. V. and Malhotra, A. (2004). XML schema part 2: Datatypes.
- [Bizer and Cyganiak, 2004] Bizer, C. and Cyganiak, R. (2004). The trig syntax. Available at: <http://www4.wiwiw.fu-berlin.de/bizer/TriG/>.
- [Bizer et al., 2007] Bizer, C., Cyganiak, R., and Heath, T. (2007). How to publish linked data on the web. Available at: <http://sites.wiwiw.fu-berlin.de/suhl/bizer/pub/LinkedDataTutorial/>.
- [Blum and Mitchell, 1998] Blum, A. and Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. In *Proceedings of the Workshop on Computational Learning Theory*, pages 92–100. Morgan Kaufmann Publishers.
- [Bojars et al., 2008] Bojars, U., Breslin, J. G., Finn, A., and Decker, S. (2008). Using the semantic web for linking and reusing data across web 2.0 communities. *Web Semantics*, 6(1):21–28.
- [Boone, 1998] Boone, G. (1998). Concept feature in re: Agent, an intelligent email agent. In *Agents*, pages 141–148.
- [Borenstein, 1992] Borenstein, N. (1992). Computational mail as network infrastructure for computer-supported cooperative work. In *CSCW 92*, pages 67–74.
- [Bowers, 1992] Bowers, J. (1992). The politics of formalism. In Lea, M., editor, *Contexts of Computer-mediated Communication*, pages 232–261. Harvester Wheatsheaf, London.
- [Bray et al., 2006] Bray, T., Hollander, D., Layman, A., and Tobin, R. (2006). Namespaces in xml 1.0 (second edition). Available at: <http://www.w3.org/TR/2006/REC-xml-names-20060816>.
- [Brickley and Guha, 2004] Brickley, D. and Guha, R. (2004). RDF Vocabulary Description Language 1.0: RDF Schema. Available at: <http://www.w3.org/TR/rdf-schema/>.

- [Brickley and Miller, 2005] Brickley, D. and Miller, L. (2005). FOAF Vocabulary Specification. Available at: <http://xmlns.com/foaf/spec/20071002.html>.
- [Brunzel and Grebner, 2008] Brunzel, M. and Grebner, O. (2008). Nepomuk task management ontology. Available at: <http://www.semanticdesktop.org/ontologies/2008/05/20/tmo/>.
- [Bush, 1945] Bush, V. (1945). As we may think. *The Atlantic Monthly*, 176(1):101–108.
- [Caires et al., 2007] Caires, M., Scerri, S., Handschuh, S., Sintek, M., and van Elst, L. (2007). A protege plug-in development to support the nepomuk representational language. In *Proceedings of the 10th International Protege Conference*.
- [Carletta, 1996] Carletta, J. (1996). Assessing agreement on classification tasks: The kappa statistic. *Computational Linguistics*, 22(2):249–254.
- [Carroll et al., 2005] Carroll, J. J., Bizer, C., Hayes, P., and Stickler, P. (2005). Named graphs, provenance and trust. In *WWW '05: Proceedings of the 14th international conference on World Wide Web*, pages 613–622, New York, NY, USA. ACM Press.
- [Carroll and Stickler, 2004] Carroll, J. J. and Stickler, P. (2004). Trix: Rdf triples in xml. Technical Report HPL-2003-268, HP Labs.
- [Carvalho and Cohen, 2005] Carvalho, V. R. and Cohen, W. W. (2005). On the collective classification of email “speech acts”. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 345–352, New York, NY, USA. ACM.
- [Carvalho and Cohen, 2006] Carvalho, V. R. and Cohen, W. W. (2006). Improving “email speech acts” analysis via n-gram selection. In *ACTS '09: Proceedings of the HLT-NAACL 2006 Workshop on Analyzing Conversations in Text and Speech*, pages 35–41, Morristown, NJ, USA. Association for Computational Linguistics.

- [Cerf et al., 1974a] Cerf, V., Dalal, Y., and Sunshine, C. (1974a). Specification of internet transmission control program.
- [Cerf et al., 1974b] Cerf, V. G., Robert, and Icahn, E. (1974b). A protocol for packet network intercommunication. *IEEE Transactions on Communications*, 22:637–648.
- [Chan et al., 2004] Chan, J., Koprinska, I., and Poon, J. (2004). Co-training with a single natural feature set applied to email classification. In *Web Intelligence*, pages 586–589. IEEE Computer Society.
- [Cheyer et al., 2005] Cheyer, A., Park, J., and Giuli, R. (2005). Iris: Integrate. relate. infer. share. In Decker, S., Park, J., Quan, D., and Sauermann, L., editors, *Proceedings of Semantic Desktop Workshop at the ISWC, Galway, Ireland, November 6*, volume 175.
- [Clark and Brennan, 1991] Clark, H. and Brennan, S. (1991). Grounding in communication. In Resnick, L., Levine, J., and Teasley, S., editors, *Perspectives on Socially Shared Cognition*, pages 127–149. American Psychological Association.
- [Clark, 1996] Clark, H. H. (1996). Using language. *Cambridge University Press*.
- [Cohen, 1960] Cohen, J. (1960). A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37.
- [Cohen, 1996] Cohen, W. (1996). Learnig rules that classify email. In *Proceedings of the IEEE Spring Symposium on Machine learning for Information Access*.
- [Cohen et al., 2004] Cohen, W., Carvalho, V., and Mitchell, T. (2004). Learning to classify email into Speech Acts. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*.
- [Cohen, 1995] Cohen, W. W. (1995). Fast effective rule induction. In *ICML*, pages 115–123.

- [Cole and Stumme, 2000] Cole, R. and Stumme, G. (2000). Cem - a conceptual email manager. In Ganter, B. and Mineau, G. W., editors, *ICCS*, volume 1867 of *Lecture Notes in Computer Science*, pages 438–452. Springer.
- [Comrie, 1985] Comrie, B. (1985). *Tense*. Cambridge University Press.
- [Corston-Oliver et al., 2004] Corston-Oliver, S., Ringger, E., Gamon, M., and Campbell, R. (2004). Task-focused summarization of email. In Marie-Francine Moens, S. S., editor, *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 43–50, Barcelona, Spain. Association for Computational Linguistics.
- [Cortes and Vapnik, 1995] Cortes, C. and Vapnik, V. (1995). Support vector networks. *Machine Learning*, 20:273–297.
- [Cunningham, 1999] Cunningham, H. (1999). JAPE: a Java Annotation Patterns Engine. Research memorandum, Department of Computer Science, University of Sheffield.
- [Cunningham, 2002] Cunningham, H. (2002). Gate, a general architecture for text engineering. *Computers and the Humanities*, 36(2):223–254.
- [Cunningham, 2005] Cunningham, H. (2005). Information extraction, automatic. *Encyclopedia of Language and Linguistics*.
- [Curtis, 1995] Curtis, B. (1995). Can speech acts walk the talk? *Computer Supported Cooperative Work*, 3(1):61–64.
- [Dabbish and Kraut, 2006] Dabbish, L. A. and Kraut, R. E. (2006). Email overload at work: an analysis of factors associated with email strain. In *CSCW '06: Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*, pages 431–440, New York, NY, USA. ACM.
- [Dawson and Stenerson, 1998] Dawson, F. and Stenerson, D. (1998). RFC 2445: Internet Calendaring and Scheduling Core Object Specification (iCalendar).

- [de Bruijn et al., 2005] de Bruijn, J., Lara, R., Polleres, A., and Fensel, D. (2005). Owl dl vs. owl flight: conceptual modeling and reasoning for the semantic web. In Ellis, A. and Hagino, T., editors, *WWW*, pages 623–632. ACM.
- [de Moor et al., 2002] de Moor, A., Keeler, M., and Richmond, G. (2002). Towards a pragmatic web. In Priss, U., Corbett, D., and Angelova, G., editors, *Proceedings of the 10th International Conference on Conceptual Structures (ICCS 2002)*, volume 2393 of *Lecture Notes in Computer Science*, pages 235–249. Springer.
- [Dean and Schreiber, 2004] Dean, M. and Schreiber, G. (2004). OWL web ontology language reference. Recommendation, World Wide Web Consortium. Available at: <http://www.w3.org/TR/owl-ref/>.
- [Decker and Frank, 2004] Decker, S. and Frank, M. (2004). The social semantic desktop. In *Proc. of the WWW2004 Workshop Application Design, Development and Implementation Issues in the Semantic Web*.
- [Decker et al., 2005] Decker, S., Park, J., Quan, D., and Sauermann, L., editors (2005). *The Semantic Desktop - Next Generation Information Management & Collaboration Infrastructure. Proc. of Semantic Desktop Workshop at the ISWC, Galway, Ireland*, volume 175 of *CEUR Workshop Proceedings ISSN 1613-0073*.
- [DiNucci, 1999] DiNucci, D. (1999). Fragmented future. *Print*, 53(4).
- [Dredze et al., 2006] Dredze, M., Lau, T., and Kushmerick, N. (2006). Automatically classifying emails into activities. In *IUI '06: Proceedings of the 11th international conference on Intelligent user interfaces*, pages 70–77, New York, NY, USA. ACM.
- [Dredze et al., 2008a] Dredze, M., Wallach, H. M., Puller, D., Brooks, T., Carroll, J., Magarick, J., Blitzler, J., and Pereira, F. (2008a). Intelligent email: Aiding users with ai. In Fox, D. and Gomes, C. P., editors, *AAAI*, pages 1524–1527. AAAI Press.

- [Dredze et al., 2008b] Dredze, M., Wallach, H. M., Puller, D., and Pereira, F. (2008b). Generating summary keywords for emails using topics. In Bradshaw, J. M., Lieberman, H., and Staab, S., editors, *IUI*, pages 199–206. ACM.
- [Ducheneaut and Belotti, 2001] Ducheneaut, N. and Belotti, V. (2001). Email as habitat: an exploration of embedded personal information management. *Interactions*, 8:30–38.
- [Dumais et al., 2003] Dumais, S., Cutrell, E., Cadiz, J. J., Jancke, G., Sarin, R., and Robbins, D. C. (2003). Stuff i’ve seen: a system for personal information retrieval and re-use. In *SIGIR ’03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 72–79. ACM Press.
- [Dumas and Redish, 1993] Dumas, J. F. and Redish, J. C. (1993). *A Practical Guide to Usability Testing*. Greenwood Publishing Group Inc., Westport, CT, USA.
- [Dustdar, 2004] Dustdar, S. (2004). Caramba - a process-aware collaboration system supporting ad hoc and collaborative processes in virtual teams. *Distributed and Parallel Databases*, 15(1):45–66.
- [Dustdar and Hoffmann, 2007] Dustdar, S. and Hoffmann, T. (2007). Interaction pattern detection in process oriented information systems. *Data and Knowledge Engineering*, 62(1):138–155.
- [Dustdar et al., 2005] Dustdar, S., Hoffmann, T., and van der Aalst, W. (2005). Mining of ad-hoc business processes with teamlog. *Data and Knowledge Engineering*, 55(2):129–158.
- [Engelbart, 1962] Engelbart, D. C. (1962). *Augmenting Human Intellect: A Conceptual Framework*.
- [Erickson, 2000] Erickson, T. (2000). Making sense of computer-mediated communication: Conversations as genres, cmc systems as genre ecologies. In *HICSS*.

- [F. Manola, 2004] F. Manola, E. Miller, E. (2004). Rdf primer. W3C Recommendation, W3C, <http://www.w3.org/TR/2004/REC-rdf-primer-20040210/>. Available at: <http://www.w3.org/TR/rdf-primer/>.
- [Feng et al., 2006] Feng, D., Shaw, E., Kim, J., and Hovy, E. (2006). Learning to detect conversation focus of threaded discussions. In *Proceedings of the Joint Human Language Technology/North American Chapter of the ACL Conference (HLT-NAACL)*, pages 208–215.
- [Fisel, 2007] Fisel, M. (2007). Machine learning techniques in dialogue act recognition.
- [Flores and Ludlow, 1980] Flores, F. and Ludlow, J. (1980). Doing and speaking in the office. In Fick, G. and Sprague, R., editors, *Decision Support Systems: Issues and Challenges*, pages 95–118. Pergamon Press, New York.
- [Forsythand and Martell, 2007] Forsythand, E. N. and Martell, C. H. (2007). Lexical and discourse analysis of online chat dialog. In *ICSC '07: Proceedings of the International Conference on Semantic Computing*, pages 19–26, Washington, DC, USA. IEEE Computer Society.
- [Gediga and Hamborg, 2001] Gediga, G. and Hamborg, K.-C. (2001). Evaluation of software systems. *Encyclopedia of Computer Science and Technology*, 45.
- [Goldstein and Sabin, 2006] Goldstein, J. and Sabin, R. E. (2006). Using speech acts to categorize email and identify email genres. In *HICSS*. IEEE Computer Society.
- [Grant and Beckett, 2004] Grant, J. and Beckett, D. (2004). Rdf test cases. Available at: http://www.w3.org/TR/2004/REC-rdf-testcases-20040210.
- [Grau et al., 2008] Grau, B., Horrocks, I., Motik, B., Parsia, B., Patel-Schneider, P., and Sattler, U. (2008). Owl 2: The next step for owl. *Web Semantics: Science, Services and Agents on the World Wide Web*.

- [Grebner et al., 2008] Grebner, O., Ong, E., and Riss, U. (2008). Kasimir - work process embedded task management leveraging the semantic desktop. In Bichler, M., Hess, T., Krcmar, H., Matthes, U. L., Florian, Picot, A., Speitkamp, B., and Wolf, P., editors, *Multikonferenz Wirtschaftsinformatik*, pages 715–726. GITO-Verlag, Berlin.
- [Grossman, 2006] Grossman, L. (2006). Person of the year: You. Time Magazine. Available at: <http://www.time.com/time/covers/0,16641,20061225,00.html>.
- [Groza et al., 2007] Groza, T., Handschuh, S., Moeller, K., Grimnes, G., Sauermann, L., Minack, E., Mesnage, C., Jazayeri, M., Reif, G., and Gudjonsdottir, R. (2007). The NEPOMUK Project — On the way to the Social Semantic Desktop. In Pellegrini, T. and Schaffert, S., editors, *Proceedings of I-Semantics' 07*, pages pp. 201–211. JUCS.
- [Gruber, 1993] Gruber, T. R. (1993). A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220.
- [Gwizdka, 2002] Gwizdka, J. (2002). Taskview: design and evaluation of a task-based email interface. In Stewart, D. A. and Johnson, J. H., editors, *CASCON*, page 4. IBM.
- [Habermas, 1984] Habermas, J. (1984). *The Theory of Communicative Action Reason and the Rationalisation of Society (Vol I)*. Beacon Press, Boston, MA.
- [Hak Lae Kim, 2009] Hak Lae Kim, Simon Scerri, J. B. (2009). Analysis and representation of tagging practices in online communities. In *Proceedings of the AAAI 2009 Spring Symposium on Social Semantic Web: Where Web 2.0 meets Web 3.0*.
- [Halpin et al., 2007] Halpin, H., Robu, V., and Shepherd, H. (2007). The complex dynamics of collaborative tagging. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 211–220, New York, NY, USA. ACM.
- [Handschuh, 2007] Handschuh, S. (2007). Semantic annotation of resources in the semantic web. In *Semantic Web Services: Concepts, Technologies, and Applications*, chapter 5, pages 135–155.

- [Hayes, 2004] Hayes, P. (2004). RDF Semantics. Recommendation, World Wide Web Consortium. Available at: <http://www.w3.org/TR/2004/REC-rdf-mt-20040210>.
- [Hendler et al., 2008] Hendler, J., Shadbolt, N., Hall, W., Berners-Lee, T., and Weitzner, D. (2008). Web science: An interdisciplinary approach to understanding the web. *Communications of the ACM*, 51(7):60–69.
- [Heylighen and Vidal, 2008] Heylighen, F. and Vidal, C. (2008). Getting things done: The science behind stress-free productivity. *Long Range Planning*, 41(6):585 – 605.
- [Hitzler et al., 2009] Hitzler, P., Krtzsch, M., Parsia, B., Patel-Schneider, P. F., and Rudolph, S. (2009). Owl 2 web ontology language primer. Recommendation, World Wide Web Consortium.
- [Horvitz, 1999] Horvitz, E. (1999). Principles of mixed-initiative user interfaces. In *CHI '99: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 159–166, New York, NY, USA. ACM.
- [Horvitz et al., 2003] Horvitz, E., Kadie, C., Paek, T., and Hovel, D. (2003). Models of attention in computing and communication: From principles to applications. *Communications of the ACM*, 46(3).
- [Idehen, 2003] Idehen, K. (2003). Rss: Injan (it's not just about news). Blog Data Space. Availabel at: <http://www.openlinksw.com/dataspace/kidehen@openlinksw.com/weblog/kidehen@openlinksw.com>
- [Jacobs and Walsh, 2004] Jacobs, I. and Walsh, N. (2004). Architecture of the world wide web, volume one.
- [Jose, 1987] Jose, P. E. (1987). Sequentiality of speech acts in conversational structure. *Psycholinguistic Research*, 1:65–88.
- [Jurafsky et al., 1997] Jurafsky, D., Bates, R., Coccaro, N., Martin, R., Meteer, M., Ries, K., Shriberg, E., Stolcke, A., Taylor, P., and Ess-Dykema, C. V. (1997). Switchboard

- discourse language modeling project report. Technical report, Johns Hopkins University, Center for Speech and Language Processing, Baltimore, MD.
- [Kanzaki, 2004] Kanzaki, M. (2004). Exif vocabulary workspace - rdf schema. Technical report, RDF Interest Group, World Wide Web Consortium. Available at: <http://www.w3.org/2003/12/exif/>.
- [Kassoff et al., 2009] Kassoff, M., Petrie, C., Zen, L., and Genesereth, M. (2009). Semantic email addressing. *IEEE Internet Computing*, 13:48–55.
- [Kerr and Wilcox, 2004] Kerr, B. and Wilcox, E. (2004). Designing remail: reinventing the email client through innovation and integration. In Dykstra-Erickson, E. and Tscheligi, M., editors, *CHI Extended Abstracts*, pages 837–852. ACM.
- [Khosravi and Wilks, 1999] Khosravi, H. and Wilks, Y. (1999). Routing email automatically by purpose not topic. *Nat. Lang. Eng.*, 5(3):237–250.
- [Khossainov and Kushmerick, 2005] Khossainov, R. and Kushmerick, N. (2005). Email task management: An iterative relational learning approach. In *CEAS*.
- [Kim et al., 2008a] Kim, H. L., Passant, A., Breslin, J., Scerri, S., and Decker, S. (2008a). Review and alignment of tag ontologies for semantically-linked data in collaborative tagging spaces. In *Proceedings of the 2nd IEEE International Conference on Semantic Computing*, volume 0, pages 315–322, Los Alamitos, CA, USA. IEEE Computer Society.
- [Kim et al., 2008b] Kim, H. L., Scerri, S., Breslin, J. G., Decker, S., and Kim, H. G. (2008b). The State of the Art in Tag Ontologies: A Semantic Model for Tagging and Folksonomies. In *Proceedings of the 2008 International Conference on Dublin Core and Metadata Applications*, pages 128–137, Berlin, Deutschland. Dublin Core Metadata Initiative.

- [Kiritchenko and Matwin, 2002] Kiritchenko, S. and Matwin, S. (2002). Email classification with co-training. Technical report, University of Ottawa.
- [Kraut et al., 1993] Kraut, R., Fish, R., Root, R., and Chalfonte, B. (1993). Informal communication in organizations: Form, function, and technology. *Groupware and Computer-Supported Cooperative Work*, pages 287–314.
- [Kushmerick and Lau, 2005] Kushmerick, N. and Lau, T. A. (2005). Automated email activity management: an unsupervised learning approach. In Amant, R. S., Riedl, J., and Jameson, A., editors, *IUI*, pages 67–74. ACM.
- [Lampert et al., 2006] Lampert, A., Dale, R., and Paris, C. (2006). Classifying speech acts using verbal response modes. pages 34–41.
- [Lansdale, 1988] Lansdale, M. (1988). The psychology of personal information management. *Applied Ergonomics*, 19(1):55–66.
- [Lehmann et al., 2009] Lehmann, J., Bizer, C., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., and Hellmann, S. (2009). DBpedia - a crystallization point for the web of data. *Journal of Web Semantics*, 7(3):154–165.
- [Lewis, 1998] Lewis, D. D. (1998). Naive (bayes) at forty: The independence assumption in information retrieval. In Nedellec, C. and Rouveirol, C., editors, *ECML*, volume 1398 of *Lecture Notes in Computer Science*, pages 4–15. Springer.
- [Li et al., 2002] Li, Y., Zaragoza, H., Herbrich, R., Shawe-Taylor, J., and Kandola, J. (2002). The perceptron algorithm with uneven margins.
- [Ljungberg and Holm, 1995] Ljungberg, J. and Holm, P. (1994 (published in 1995)). Speech acts on trial. In *Proceedings of Third Decennial Conference: Computers in Context - Joining Forces in Design*.
- [Longacre, 1986] Longacre, R. E. (1986). *An Anatomy of Speech Notions*. Walter De Gruyter Inc, Thousand Oaks, CA, US.

- [M. and W.C., 1987] M., M. and W.C., W. (1987). Misinterpretation and misuse of the kappa statistic. *American journal of epidemiology*.
- [Mackay, 1988] Mackay, W. E. (1988). More than just a communication system: Diversity in the use of electronic mail. In *CSCW*, pages 344–353.
- [McAfee, 2006] McAfee, A. P. (2006). Enterprise 2.0: The dawn of emergent collaboration. *MIT Sloan Management Review*, 47(3):21–28.
- [Mccreath and Kay, 2003] Mccreath, E. and Kay, J. (2003). Iems: Helping users manage email. In *User Modeling 2003*.
- [McDowell et al., 2003] McDowell, L., Etzioni, O., Gribble, S. D., Halevy, A. Y., Levy, H. M., Pentney, W., Verma, D., and Vlasseva, S. (2003). Evolving the semantic web with mangrove. In *WWW (Posters)*.
- [Mcdowell et al., 2004] Mcdowell, L., Etzioni, O., Halevy, A., and Levy, H. (2004). Semantic email. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 244–254, New York, NY, USA. ACM Press.
- [McRoy and Hirst, 1995] McRoy, S. W. and Hirst, G. (1995). The repair of speech act misunderstandings by abductive inference. *Comput. Linguist.*, 21(4):435–478.
- [Miles, 2004] Miles, A. (2004). Simple Knowledge Organisation System (SKOS). Available at: <http://www.w3.org/2004/02/skos/>.
- [Minker, 1982] Minker, J. (1982). On indefinite databases and the closed world assumption. In Loveland, D., editor, *6th Conference on Automated Deduction*, volume 138 of *Lecture Notes in Computer Science*, pages 292–308. Springer Berlin / Heidelberg.
- [Möller and Decker, 2005] Möller, K. and Decker, S. (2005). Harvesting desktop data for semantic blogging. In Decker, S., Park, J., Quan, D., and Sauermann, L., editors, *Proc. of Semantic Desktop Workshop at the ISWC, Galway, Ireland, November 6*, volume 175.

- [Möller and Handschuh, 2007] Möller, K. and Handschuh, S. (2007). Towards a lightweight semantic desktop. In *Proceedings of the Semantic Desktop Design Workshop (SemDeskDesign 2007) at ESWC2007, Innsbruck, Austria*, Innsbruck, Austria.
- [Möschler, 1992] Möschler, J. (1992). The pragmatic aspects of linguistic negation: Speech act, argumentation and pragmatic inference. *Argumentation*, 6:51–76. 10.1007/BF00154259.
- [Motik and Horrocks, 2006] Motik, B. and Horrocks, I. (2006). Problems with owl syntax. In Grau, B. C., Hitzler, P., Shankey, C., and Wallace, E., editors, *OWLED*, volume 216 of *CEUR Workshop Proceedings*.
- [Motik et al., 2009] Motik, B., Patel-Schneider, P. F., and Parsia, B. (2009). Owl 2 web ontology language structural specification and functional-style syntax. Recommendation, World Wide Web Consortium.
- [Mylka et al., 2007] Mylka, A., Sauermann, L., Sintek, M., and van Elst, L. (2007). Nepomuk information element ontology. Available at: <http://www.semanticdesktop.org/ontologies/2007/01/19/nie/>.
- [Nelson, 1965] Nelson, T. H. (1965). Complex information processing: a file structure for the complex, the changing and the indeterminate. In *ACM '65: Proceedings of the 1965 20th national conference*, pages 84–100, New York, NY, USA. ACM.
- [Newman et al., 2005] Newman, R., Ayers, D., and Russell, S. (2005). Tag ontology. Available at: <http://www.holygoat.co.uk/owl/redwood/0.1/tags/>.
- [Nielsen, 2000] Nielsen, J. (2000). Why you only need to test five users. Available at: <http://www.useit.com/alertbox/20000319.html>.
- [Norman, 2002] Norman, D. A. (2002). *The Design of Everyday Things*. Basic Books, New York, reprint paperback edition.

- [Ogihara, 2007] Ogihara, T. (2007). Tense and aspect in truth-conditional semantics. *Lingua*, 117(2):392 – 418. Approaches to tense and tense construal.
- [Oren et al., 2008] Oren, E., Mller, K. H., Scerri, S., H, S., and Sintek, M. (2008). What are semantic annotations?
- [Palma et al., 2009] Palma, R., Hartmann, J., and Haase, P. (2009). Omv ontology meta-data vocabulary for the semantic web. Technical Report v2.4.1, OMV Consortium. Available at: <http://ontoware.org/frs/download.php/669/OMV-Reportv2.4.1.pdf>.
- [Parunak, 1996] Parunak, H. V. D. (1996). Visualizing agent conversations: Using enhanced dooley graphs for agent design and analysis.
- [Passant and Laublet, 2008] Passant, A. and Laublet, P. (2008). Meaning Of A Tag: A collaborative approach to bridge the gap between tagging and Linked Data. In *Proceedings of the WWW 2008 Workshop Linked Data on the Web (LDOW2008), Beijing, China*.
- [Patel-Schneider, 2004] Patel-Schneider, P. F. (2004). What is owl (and why should i care)? In *Proceedings of the ninth International Conference on the Principles of Knowledge Representation and Reasoning*, pages 735–737.
- [Payne, 1994] Payne, T. (1994). Learning email filtering rules with magi, a mail agent interface.
- [Pazzani, 2000] Pazzani, M. J. (2000). Representation of electronic mail filtering profiles: A user study. In *International Conference on Intelligent User Interfaces.*, New Orleans, LA USA.
- [Pepper, 2010] Pepper, S. (2010). Topic maps. In *Encyclopedia of Library and Information Science*. Taylor & Francis, 3 edition.

- [Prud'hommeaux and Seaborne, 2008] Prud'hommeaux, E. and Seaborne, A. (2008). SPARQL query language for RDF. Recommendation, World Wide Web Consortium. <http://www.w3.org/TR/rdf-sparql-query/>.
- [Quan et al., 2003] Quan, D., Huynh, D., and Karger, D. R. (2003). Haystack: A platform for authoring end user semantic web applications. In *International Semantic Web Conference*, pages 738–753.
- [Quan-Haase et al., 2005] Quan-Haase, A., Cothrel, J., and Wellman, B. (2005). Instant messaging for collaboration: A case study of a high-tech firm. *J. Computer-Mediated Communication*, 10(4).
- [Raymond, 1999] Raymond, E. S. (1999). *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*. O'Reilly, Cambridge, Mass, 1st edition edition.
- [Reif et al., 2008] Reif, G., Groza, T., Scerri, S., and Handschuh, S. (2008). Final NEPO-MUK Architecture Deliverable D6.2.B. EU Project Deliverable D 6.2.B, NEPOMUK Consortium. Contributors: Francesco Lelli, Edith Felix, Rosa Gudjonsdottir, Leo Sauermann, Ann Johnston, Martin.
- [Rennie, 2000] Rennie, J. (2000). iFILE: an application of machine learning to email filtering. In *Proceedings of the SIGKDD Text Mining Workshop*.
- [Resnick, 2001] Resnick, P. (2001). RFC 2822: Internet message format. Technical report, IETF. Available at: <http://www.rfc-archive.org/getrfc.php?rfc=2822>.
- [Richter et al., 2005] Richter, J., V'olkel, M., and Haller, H. (2005). Deepamehta – a semantic desktop. In Decker, S., Park, J., Quan, D., and Sauermann, L., editors, *Proc. of Semantic Desktop Workshop at the ISWC, Galway, Ireland, November 6*, volume 175.

- [Robie et al., 2001] Robie, J., Garshol, L. M., Newcomb, S., Biezunski, M., Fuchs, M., Miller, L., Brickley, D., Christophides, V., and Karvounarakis, G. (2001). The syntactic web. *Markup Lang.*, 3(4):411–440.
- [Rohall et al., 2004] Rohall, S. L., Gruen, D., Moody, P., Wattenberg, M., Stern, M., Kerr, B., Stachel, B., Dave, K., Armes, R., and Wilcox, E. (2004). Remail: a reinvented email prototype. In Dykstra-Erickson, E. and Tscheligi, M., editors, *CHI Extended Abstracts*, pages 791–792. ACM.
- [Sahami, 1998] Sahami (1998). A Bayesian approach to filtering junk e-mail. In *AAAI workshop on learning for text categorization*.
- [Sauermann, 2009] Sauermann, L. (2009). *The Gnowsis Semantic Desktop approach to Personal Information Management*. PhD thesis, Fachbereich Informatik der Universitt Kaiserslautern. Supervisors: Prof. Andreas Dengel, Prof. Mehdi Jazayeri.
- [Sauermann et al., 2005] Sauermann, L., Bernardi, A., and Dengel, A. (2005). Overview and Outlook on the Semantic Desktop. In Decker, S., Park, J., Quan, D., and Sauermann, L., editors, *Proceedings of the 1st Workshop on The Semantic Desktop at the ISWC 2005 Conference*, volume 175 of *CEUR Workshop Proceedings*, pages pp. 1–19.
- [Sauermann et al., 2007] Sauermann, L., Elst, L. V., and Mööller, K. (2007). Personal Information Model (PIMO) Ontology Guide. NEPOMUK Specification. Available at: <http://www.semanticdesktop.org/ontologies/pimo>.
- [Sauermann et al., 2006] Sauermann, L., Grimnes, G. A., Kiesel, M., Fluit, C., Maus, H., Heim, D., Nadeem, D., Horak, B., and Dengel, A. (2006). Semantic Desktop 2.0: The Gnowsis Experience. In *Proc. of the ISWC Conference*, volume 4273/2006 of *Lecture Notes in Computer Science*, pages 887–900. Springer Berlin / Heidelberg. ISSN 0302-9743 (Print) 1611-3349 (Online).

- [Scerri, 2007] Scerri, S. (2007). Aiding the workflow of email conversations by enhancing email with semantics. In *Proceedings of the PhD Symposium at 2007 European Semantic Web Conference*.
- [Scerri, 2008a] Scerri, S. (2008a). Semanta - your personal email semantic assistant. In *Proceedings of the 2008 International Conference on Intelligent User Interfaces*.
- [Scerri, 2008b] Scerri, S. (2008b). Semantic email ontology. Available at: <http://ontologies.smile.deri.ie/smail>.
- [Scerri et al., 2007a] Scerri, S., Davis, B., and Handschuh, S. (2007a). Improving email conversation efficiency through semantically enhanced email. In *Proceedings of the Web Semantics Workshop at the 18th International Conference on Database and Expert Systems Applications (DEXA '07)*, pages 490–494.
- [Scerri et al., 2008a] Scerri, S., Davis, B., and Handschuh, S. (2008a). The path towards semantic email: Summary and outlook. In *Proceedings of the AAAI 2008 Workshop on Enhanced Messaging*.
- [Scerri et al., 2009a] Scerri, S., Davis, B., Handschuh, S., and Hauswirth, M. (2009a). Semanta - semantic email made easy. In *Proceedings of the 6th European Semantic Web Conference*.
- [Scerri et al., 2009b] Scerri, S., Davis, B., Siegfried, and Handschuh (2009b). Semanta - supporting email workflows in business processes. In *Proceedings of the 11th IEEE Conference on Commerce and Enterprise Computing (CEC09)*.
- [Scerri et al., 2009c] Scerri, S., Giurciu, I., Davis, B., and Handschuh, S. (2009c). Semanta semantic email in action. In *Demo at the 6th European Semantic Web Conference*.
- [Scerri et al., 2010a] Scerri, S., Gossen, G., Davis, B., and Handschuh, S. (2010a). Classifying action items for semantic email. In *Proceedings of the seventh conference on*

- International Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).
- [Scerri et al., 2010b] Scerri, S., Gossen, G., and Handschuh, S. (2010b). Supporting digital collaborative work through semantic technology. In *Proceedings of the International Conference on Knowledge Management and Information Sharing*, Valencia, Spain.
- [Scerri et al., 2008b] Scerri, S., Handschuh, S., and Decker, S. (2008b). Semantic email as a communication medium for the social semantic desktop. In *The 5th European Semantic Web Conference (ESWC2008)*, Tenerife, Spain, pages 124–138.
- [Scerri et al., 2008c] Scerri, S., Mencke, M., Davis, B., and Handschuh, S. (2008c). Evaluating the ontology powering smail - a conceptual framework for semantic email. In *Proceedings of the sixth International Conference on Language Resources and Evaluation*.
- [Scerri et al., 2007b] Scerri, S., Sintek, M., van Elst, L., and Handschuh, S. (2007b). Nepomuk annotation ontology. Available at: <http://www.semanticdesktop.org/ontologies/2007/08/15/nao>.
- [Schenk and Staab, 2008] Schenk, S. and Staab, S. (2008). Networked graphs: a declarative mechanism for sparql rules, sparql views and rdf data integration on the web. In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pages 585–594, New York, NY, USA. ACM.
- [Searle, 1975a] Searle, J. (1975a). Indirect speech acts. In Cole, P. and Morgan, J., editors, *Syntax and Semantics 3: Speech Acts*, pages 59–82. Academic Press, New York.
- [Searle, 1975b] Searle, J. (1975b). A taxonomy of illocutionary acts. In Gunderson, K., editor, *Language, Mind and Knowledge*, pages 344–369. University of Minnesota Press.

- [Searle, 1969] Searle, J. R. (1969). *Speech Acts: An Essay in the Philosophy of Language*. Cambridge University Press, Cambridge, London.
- [Segal and Kephart, 1999] Segal, R. and Kephart, J. (1999). Mailcat: An intelligent assistant for organizing mail. In *Autonomous Agents*.
- [Segal and Kephart, 2000] Segal, R. and Kephart, J. O. (2000). Incremental learning in swiftfile. In Langley, P., editor, *ICML*, pages 863–870. Morgan Kaufmann.
- [Semy et al., 2004] Semy, S. K., Pulvermacher, M. K., and Obrst, L. J. (2004). Towards the use of an upper ontology for U.S. government and military domains: An evaluation. Technical report, MITRE.
- [Shannon, 2006] Shannon, V. (2006). A ‘more revolutionary’ web. The New York Times Technology. Available at: <http://www.nytimes.com/2006/05/23/technology/23iht-web.html>.
- [Shen et al., 2009] Shen, J., Fitzhenry, E., and Dietterich, T. G. (2009). Discovering frequent work procedures from resource connections. In Conati, C., Bauer, M., Oliver, N., and Weld, D. S., editors, *IUI*, pages 277–286. ACM.
- [Shneiderman and Plaisant, 2009] Shneiderman, B. and Plaisant, C. (2009). *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Pearson Addison-Wesley, Upper Saddle River, NJ, 5. edition.
- [Singh, 1991] Singh, M. P. (1991). Towards a formal theory of communication for multi-agent systems. In *IJCAI*, pages 69–74.
- [Sintek and Decker, 2002] Sintek, M. and Decker, S. (2002). Triple - a query, inference, and transformation language for the semantic web. In *ISWC '01: Proceedings of the First International Semantic Web Conference on The Semantic Web*, pages 364–378, London, UK. Springer-Verlag.

- [Sintek et al., 2007a] Sintek, M., Elst, L., Scerri, S., and Handschuh, S. (2007a). Distributed knowledge representation on the social semantic desktop: Named graphs, views and roles in nrl. In *ESWC'07: Proceedings of the 4th European conference on The Semantic Web*, pages 594–608, Berlin, Heidelberg. Springer-Verlag.
- [Sintek et al., 2007b] Sintek, M., Elst, L. V., Scerri, S., and Handschuh, S. (2007b). Nepomuk Representational Language Specification. Nepomuk specification. Available at: <http://www.semanticdesktop.org/ontologies/nrl/>.
- [Sintek et al., 2009] Sintek, M., Handschuh, S., Scerri, S., and van Elst, L. (2009). Technologies for the social semantic desktop. In Tessaris, S., Franconi, E., Eiter, T., Gutierrez, C., Handschuh, S., Rousset, M.-C., and Schmidt, R. A., editors, *Reasoning Web*, volume 5689 of *Lecture Notes in Computer Science*, pages 222–254. Springer.
- [Sintek et al., 2007c] Sintek, M., van Elst, L., Grimnes, G. A., Scerri, S., and Handschuh, S. (2007c). Knowledge representation for the distributed, social semanticweb - named graphs, graph roles and views in nrl. In *Proceedings of the 2nd International Workshop on Modular Ontologies (WoMO-07)*.
- [Sitter and Stein, 1992] Sitter, S. and Stein, A. (1992). Modelling the illocutionary aspects of information-seeking dialogues. *Information Processing and Management*, 28(2):165–180.
- [Smith and Cohen, 1995] Smith, I. and Cohen, P. R. (1995). Toward a semantics for a speech act based agent communications language. In *Proceedings of the CIKM 95 Workshop on Intelligent Information Agents*.
- [Stiles, 1992] Stiles, W. B. (1992). *Describing talk: A taxonomy of verbal response modes*, volume 12 of *Sage series in interpersonal communication*. Sage Publications, Inc., Thousand Oaks, CA, US.
- [Stoitsev et al., 2008] Stoitsev, T., Scheidl, S., Flentge, F., and M'uhlh'ausser, M. (2008). From personal task management to end-user driven business process modeling. In

- Dumas, M., Reichert, M., and Shan, M.-C., editors, *BPM*, volume 5240 of *Lecture Notes in Computer Science*, pages 84–99. Springer.
- [Stolcke et al., 1998] Stolcke, A., Shriberg, E., Bates, R., Coccaro, N., Jurafsky, D., Martin, R., Meteer, M., Ries, K., Taylor, P., and Van Ess-Dykema, C. (1998). Dialog act modeling for conversational speech. In Chu-Carroll, J. and Green, N., editors, *Applying Machine Learning to Discourse Processing. Papers from the 1998 AAAI Spring Symposium*. Tech. rep. SS-98-01, pages 98–105, Stanford, CA. AAAI Press.
- [Stubbs, 1983] Stubbs, M. (1983). *Discourse Analysis*. Blackwell.
- [Suchman, 1993] Suchman, L. A. (1993). Do categories have politics? the language/action perspective reconsidered. *Computer Supported Cooperative Work (CSCW)*, 2(3):177–190.
- [Suchman, 1994] Suchman, L. A. (1994). Speech acts and voices: Response to winograd et al. *Computer Supported Cooperative Work*, 3(1):85–95.
- [Te’eni, 2006] Te’eni, D. (2006). The language-action perspective as a basis for communication support systems. *Communications of the ACM*, 49(5):65–70.
- [Teevan et al., 2004] Teevan, J., Alvarado, C., Ackerman, M. S., and Karger, D. R. (2004). The perfect search engine is not enough: a study of orienteering behavior in directed search. In *CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 415–422, New York, NY, USA. ACM.
- [Traum, 2000] Traum, D. (2000). 20 questions for dialogue act taxonomies. *Journal of Semantics*, 17(1):7–30.
- [Turenne, 2003] Turenne, N. (2003). Learning semantic classes for improving email classification. In *Biomtrie et Intelligence*.

- [Turski et al., 2005] Turski, A., Warnack, D., Cheng, L., Farnham, S., and Yee, S. (2005). Inner circle: people centered email client. In van der Veer, G. C. and Gale, C., editors, *CHI Extended Abstracts*, pages 1845–1848. ACM.
- [Twitchell and Jr., 2004] Twitchell, D. P. and Jr., J. F. N. (2004). Speech act profiling: A probabilistic method for analyzing persistent conversations and their participants. In *Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04)*.
- [Uren et al., 2006] Uren, V., Cimiano, P., Iria, J., Handschuh, S., Vargas-Vera, M., Motta, E., and Ciravegna, F. (2006). Semantic annotation for knowledge management: Requirements and a survey of the state of the art. *Web Semantics: Science, Services and Agents on the World Wide Web*, 4(1):14 – 28.
- [van der Aalst et al., 2003] van der Aalst, W. M. P., van Dongena, B. F., Herbst, J., Marustera, L., Schimm, G., and Weijters, A. J. M. M. (2003). Workflow mining: A survey of issues and approaches. *Data Knowledge Engineering*, 47(2):237–267.
- [van Heijst et al., 1995] van Heijst, G., Falasconi, S., Abu-Hanna, A., Schreiber, G., and Stefanelli, M. (1995). A case study in ontology library construction. *Artificial Intelligence in Medicine*, 7(3):227–255.
- [Venolia et al., 2001] Venolia, G., Dabbish, L., Cadiz, J., and Gupta, A. (2001). Supporting email workflow. Technical report, Microsoft.
- [Volz et al., 2002] Volz, R., Oberle, D., Staab, S., and Studer, R. (2002). OntoLift prototype. Deliverable D11, WonderWeb Project.
- [Voorhoeve and van der Aalst, 1997] Voorhoeve, M. and van der Aalst, W. M. P. (1997). Ad-hoc workflow: Problems and solutions. In *DEXA Workshop*, pages 36–40.

- [Wang et al., 2006] Wang, T. D., Parsia, B., and Hendler, J. (2006). A survey of the web ontology landscape. In *Proc. of the 5th Int. Semantic Web Conference (ISWC 2006)*, Athens, Georgia.
- [Wardrip-Fruin, 2003] Wardrip-Fruin, N. (2003). *Introduction to “Computer Lib/Dream Machines”*, pages 301–302. The New Media Reader.
- [White, 2004] White, S. A. (2004). Workflow patterns with bpmn and uml. *IBM, January*.
- [Whittaker, 2005] Whittaker, S. (2005). Supporting collaborative task management in e-mail. *Human-Computer Interaction*, 20(1):49–88.
- [Whittaker et al., 2006] Whittaker, S., Bellotti, V., and Gwizdka, J. (2006). Email in personal information management. *Communications of the ACM*, 49(1):68–73.
- [Whittaker et al., 1994] Whittaker, S., Frohlich, D., and Daly-Jones, O. (1994). Informal workplace communication: what is it like and how might we support it? In Adelson, B., Dumais, S. T., and Olson, J. S., editors, *CHI*, pages 131–137. ACM.
- [Whittaker and Hirschberg, 2001] Whittaker, S. and Hirschberg, J. (2001). The character, value, and management of personal paper archives. *ACM Transactions on Computer-Human Interaction*, Vol. 8(No. 2):Pages 150–170. ATT Labs—Research.
- [Whittaker et al., 2005] Whittaker, S., Moran, T. P., and Farrell, S. P. (2005). Why email is not enough: Combining communication and shared representation to support activity management.
- [Whittaker and Sidner, 1996] Whittaker, S. and Sidner, C. (1996). Email overload: exploring personal information management of email. In *CHI '96: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 276–283, New York, NY, USA. ACM.

- [Winograd, 1986] Winograd, T. (1986). A language/action perspective on the design of cooperative work. In *Proceedings of the 1986 ACM conference on Computer-supported cooperative work*, pages 203–220. ACM Press New York, NY, USA.
- [Winograd, 1994] Winograd, T. (1994). Categories, disciplines, and social coordination. *Computer Supported Cooperative Work*, (2):191–197.
- [Winograd and Flores, 1986] Winograd, T. and Flores, F. (1986). *Understanding Computers and Cognition: A New Foundation for Design*. Ablex, Norwood, NJ.
- [Witten and Frank, 2000] Witten, I. H. and Frank, E. (2000). *Data mining: practical machine learning tools and techniques with Java implementations*. Morgan Kaufmann series in data management systems. Morgan and Kaufmann, San Francisco, CA.
- [Xiao and Cruz, 2005] Xiao, H. and Cruz, I. F. (2005). A multi-ontology approach for personal information management. In *Semantic Desktop Workshop at ISWC2005, Galway, Ireland*.

A Summative Evaluation Questionnaire

This questionnaire is also available live online ⁵⁶.

A.1 System Usability (USE Questionnaire)

A.1.1 Usefulness

	1	2	3	4	5	6	7
It helps me be more effective.							
It helps me be more productive.							
It is useful.							
It gives me more control over the activities in my life.							
It makes the things I want to accomplish easier to get done.							
It saves me time when I use it.							
It meets my needs.							
It does everything I would expect it to do.							

⁵⁶http://www.surveymonkey.com/s.aspx?sm=hwJLbdf_2bZdyUL6hXw4dhiQ_3d_3d

A.1.2 Ease of Use

	1	2	3	4	5	6	7
It is easy to use.							
It is simple to use.							
It is user friendly.							
It requires the fewest steps possible to accomplish what I want to do with it.							
It is flexible.							
Using it is effortless.							
I can use it without written instructions.							
I don't notice any inconsistencies as I use it.							
Both occasional and regular users would like it.							
I can recover from mistakes quickly and easily.							
I can use it successfully every time.							

A.1.3 Ease of Learning

	1	2	3	4	5	6	7
I learned to use it quickly.							
I easily remember how to use it.							
It is easy to learn to use it.							
I quickly became skillful with it.							

A.1.4 Satisfaction

	1	2	3	4	5	6	7
I am satisfied with it.							
I would recommend it to a friend.							
It is fun to use.							
It works the way I want it to work.							
It is wonderful.							
I feel I need to have it.							
It is pleasant to use.							

A.2 Task Comparison

A.2.1 Writing New Email

	1	2	3	4	5	6	7
Overall experience.							
Time required to write and send email.							
Effect of the annotation creation/review requirement.							
Effect on email-writing style.							

A.2.2 Reading & Replying to Incoming Email

	1	2	3	4	5	6	7
Overall experience.							
Time required to open and view the email.							
Time required to act upon an incoming email.							
Identifying Email Action Items.							
Effect on email-reading style (going through email).							
Time required to reply to an email.							
Flexibility in replying.							
Effect on your email-replying style.							

A.2.3 Email Action Item Tracking

	1	2	3	4	5	6	7
Remembering incoming email items that require attention.							
Remembering the amount of action items within email items in the inbox.							
Keeping track of unanswered outgoing requests.							
Remembering the context of pending action items.							
Visualising specific conversational threads within email threads.							

A.2.4 Task/Calendar Integration (Skip if features not used)

	1	2	3	4	5	6	7
Adding email-generated tasks/events to the respective Calendar.							
Visualising links between email items and related task/calendar items.							

A.2.5 General

	1	2	3	4	5	6	7
Remembering to attach files to email.							
Finding the previous/next email in a thread.							
User control over email data.							
Managing Email Information.							

A.2.6 Comments & Feedback

Are Semanta's functionalities, including the workflow visualisations and action item tracking, worth the effort to review automatic annotations or manually create them?

No	Somewhat	No Opinion	Yes

Best feature of Semanta is.. (required)

Worst feature of Semanta is.. (required)

Would you like to see any new feature(s) added?

General comments, concerns & criticism