# Semanta – Semantic Email Made Easy

Simon Scerri[1], Brian Davis[1], Siegfried Handschuh[1] and Manfred Hauswirth[1]

[1] Digitial Enterprise Research Institute,
National University of Ireland Galway, IDA Business Park, Galway, Ireland.
{simon.scerri, brian.davis, siegfried.handschuh, manfred.hauswirth}@deri.org

**Abstract.** In this paper we present Semanta – a fully-implemented system supporting Semantic Email Processes, integrated into the existing technical landscape and using existing email transport technology. By applying Speech Act Theory, knowledge about these processes can be made explicit, enabling machines to support email users with correctly interpreting, handling and keeping track of email messages, visualizing email threads and workflows, and extracting tasks and appointments from email messages. Whereas complex theoretical models and semantics are hidden beneath a simple user interface, the enabled functionalities are clear for the users to see and take advantage of. The system's evaluation proved that our experiment with Semanta has indeed been successful and that semantic technology can be applied as an extra layer to existing technology, thus bringing its benefits into everyday computer usage.

**Keywords:** Semantic Email, Speech Act Theory, Semantic Web Applications.

## 1 Introduction

In earlier work [1] we discussed the notion of *Semantic Email* – whereby the email process is semantically-enabled so that intelligent mail-user agents can support the user with their daily email chores like writing, reading, tracking emails and action items within, as well as managing personal information generated within. We all use email subconsciously to organise our work, but little do we realize that email is just the transportation layer supporting what effectively is a number of distributed, well-defined ad-hoc workflow processes. As a consequence of the inability to manage these implicit workflows, the productivity of knowledge workers that rely on email communication suffers. By adding semantics to the email process we can add new functionality, whereby machines can support the user in managing and keeping track of the ongoing workflows. If an email system is aware of the executing workflows, it can provide semantic ad-hoc process support.

In this paper, we will present *Semanta*, a fully-implemented system supporting Semantic Email, whereby we have lifted email processes to a semantic level via speech act theory and a formally-defined ad-hoc email workflow model. There is a lot of talk about semantic software, but these are seldom fully-implemented. In the cases when they are, a frequently forgotten golden rule is that semantics should be the enabler of further functionality and it is the latter not the former which the user should be aware of. Our achievements with regards to Semanta are that:

- we did not change the transport layer for email processes
- we did not create new Mail User Agents (MUAs) but provided lightweight extension to existing email applications
- we abstracted a complex theoretical model beneath a very simple Graphical User Interface (GUI)
- the extensions will be immediately useful by everyone who uses email

We would like to stress the final point, because we believe that such a viral approach is what is really required by the Semantic Web community to get semantics into everyday computer usage. The main results of the evaluation of the implemented system have confirmed that its support for semantic email processes is applicable to everyday email usage and that the chances for Semanta's take-up are very high.

## 2 Semantic Email

The notion of Semantic Email was originally introduced by McDowell et. al. [2] to refer to an email message consisting of a structured query (or an update to the query) coupled with a corresponding explanatory text. Their approach was 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). Implemented within Mangrove [3] 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.

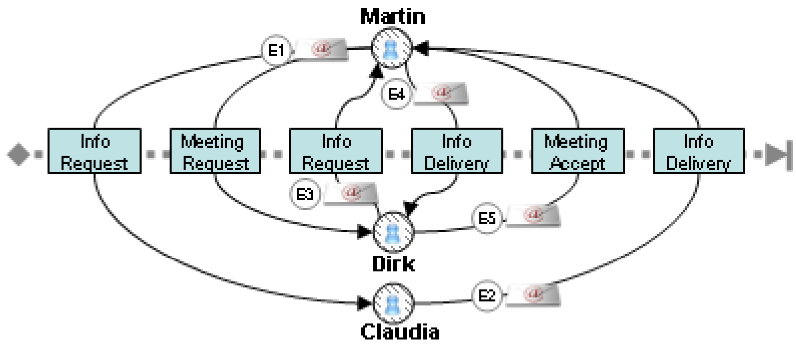



**Fig. 1.** An Email Workflow between Martin, Dirk and Claudia

Although we also handle commonly occurring email workflows, we take a different approach [1] whereby the only real similarity is that we also expose all the available knowledge to machines for querying, reasoning and eventually integration.

For this purpose, we employ the Resource Description (RDF)[1] to represent information extracted from email messages as machine-understandable metadata. In our vision we consider the fact that an email has one or more purposes, or *Action Items*. The content of an email message can be summarized into a number of such items (e.g. *Meeting Request*, *Task Assignment*, *File Request* etc.). Once exchanged, every single action item can be seen as the start, or continuation of a separate workflow. The *sMail Conceptual Framework* [4] applies *Speech Act Theory* [5] to the email communication process, in order to provide a formal structure and semantics for these action items and their workflows. Email action items like the ones above can be represented by a number of speech act instances provided in the sMail ontology[2] (in this paper we use the terms speech acts and action items interchangeably).

At the design stage, we learned a lot from a debate over the use of technology for the coordination of individual actions in organisations, sparked off by the Coordinator [6] – a heavily criticised system based on speech act theory. This work highlighted the need for a more sophisticated coordination of interactions among individuals, so as to avoid breakdowns in an organisation's efficiency. As documented in [7], a major concern was that shaping conversation to constrain behaviour reduces the potential of electronic communication to support ubiquitous interaction. The system we designed is based on an acceptable compromise between *Computability* and *Communication Efficiency*. We are aware that email workflows are flexible, evolve unpredictably with time, and that one can only predict what is *likely* to occur after certain action items are received or sent. For example, on receiving a *Meeting Request*, the user tends to accept the request, decline it or propose changes to the requested meeting. However, the user might also want to pursue other paths spontaneously. This is demonstrated in Fig.1. Martin sends an email to Claudia and Dirk (E1). The email contains a *Meeting Request* addressed to Dirk, and an *Information Request* addressed to Claudia. When Claudia receives the information request from Martin, she immediately sends an email (E2) back with the required information. On the other hand, when Dirk receives the meeting request, he decides to send an email (E3) with a request for more information regarding the meeting, and waits for a reply (E4) before also approving the meeting via another email (E5). We believe that this flexibility is an advantage of email communication and consequently should be retained. Therefore, although we believe that the option of fixed templates taken in [2] can in some cases be useful, our approach is more oriented towards the handling of what we termed *Email Ad-hoc Workflows*. In earlier work we presented the *Email Speech Act Workflow* model, explicitly modeled via standardised Workflow Patterns [8], to support exactly these kinds of email workflows. While it outlines the most common reactions to incoming email action items, it simultaneously allows for the characteristic spontaneity of email as demonstrated in our example. In this paper we will not go again into the details and complexity of the workflow model. Instead, we will demonstrate its usefulness in supporting semantic email processes as experienced by the user when using Semanta.

---

1 http://www.w3.org/RDF/

2 http://ontologies.smile.deri.ie/smail

## 3 Semanta

Semanta is a system that supports the handling and tracking of common email processes (e.g. *Meeting Scheduling*, *Task Delegation, Information Request*) (co)executing in email threads, and manages resulting artefacts (e.g. events, tasks). This implementation follows a limited prototype presented earlier [9]. Additionally, Semanta is now fully-integrated within the Social Semantic Desktop [10]. After an overview of the implemented system's architecture we will demonstrate its features.

### 3.1 Architecture

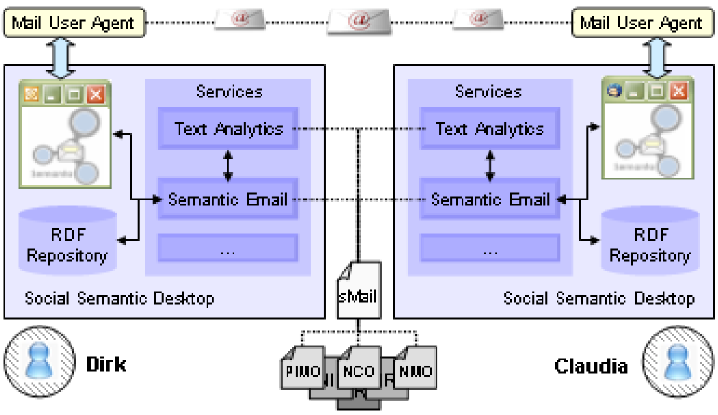



**Fig. 2.** Semanta within the Social Semantic Desktop

Fig. 2 illustrates the general system architecture. A simple intuitive user interface is crucial for the take-up of a semantic email application. Hiding the underlying complex workflow model beneath such a UI was not straightforward, but as we demonstrate in the next section the user is exposed only to the additional functionalities provided by the hidden semantics, and not the semantics themselves. Semanta's interface has been implemented as an add-in/extension to two popular Mail User Agents – Microsoft Outlook 2003 and Mozilla Thunderbird. The MUAs are still responsible for creating and sending email messages using the existing, time-tested transport technologies. Semanta is integrated within the Social Semantic Desktop (SSD) and since it employs Semantic Web technologies to represent generated data, not only is this data exposed to semantic operations such as queries and inference, but it is also immediately available for re-use by other intelligent SSD applications, and vice versa. On the SSD, personal information such as address book data, calendar data, email data, folder structures, etc. is lifted onto an RDF representation. These information items are treated as Semantic Web resources and ontologies allow the user to express such desktop data formally. The NEPOMUK Project[3] developed an infrastructure of ontologies[4] to handle different aspects of the SSD - from the Personal Information Model Ontology (PIMO) [11] to represent the structures and concepts within a knowledge workers mental model; to the Information Element Ontologies (NIE) to represent files (NFO), contacts (NCO), messages (NMO) etc. At the semantic level, integrating Semanta on the SSD required the modification of the sMail Ontology to extend existing concepts in the SSD ontologies (e.g. nmo:Email). At the practical level, Semanta was integrated by enabling the access and storage of generated metadata on the desktop's RDF repository.

The SSD provides a number of underlying services serving specific functions. Two crucial architectural components are the *Semantic Email* and the *Text Analytics* services. Both access the knowledge of the sMail Framework via the sMail Ontology, providing the Business Logic, which is separate from the Graphical User Interface (GUI). Thus only the GUI is dependent on the targeted MUA, i.e. Semantic Email can be exchanged between different users using different MUAs on different platforms. The semantic email service acts as an invisible layer beneath the GUI performing:

- Semi-automatic content annotation (via the use of the text analytics service)
- Writing/reading RDF statements into/from a specific ‘x-smail’ email header
- Reasoning over which options a user is given when reacting to action items (given the sMail Email Speech Act Workflow model)
- Detecting tasks or events generated within email
- Storing metadata in the SSD’s central RDF Repository
- Querying data in the repository to provide the user with information regarding action items, emails, tasks, events, people and their relationships.

The semantic email service utilises the text analytics service to provide semi-automatic annotation of email content. This service uses Ontology-Based Information Extraction (OBIE) techniques to elicit speech acts (action items) in email bodies. The information extraction is based on a declarative model which classifies text into speech acts based on a number of linguistic features like sentence form, tense, modality and the semantic roles of verbs. The service deploys a GATE [12] corpus pipeline consisting of a tokeniser, modified sentence splitter, POS tagger, keyphrase lookup via Finite State gazetteers and several JAPE [13] grammars. Some of these grammars are conditionally run based on the outcome of previous JAPE annotations and are ordered in priority to consume the longest matching annotation. Earlier work [14] implemented similar Knowledge Based (KB) approaches using earlier versions of GATE. The current service is at the beta testing stage whereby JAPE grammars are iteratively tuned based on the outcome of each test cycle. The results of this technology’s initial evaluation, where a corpus of manually annotated email was compared against automatic annotation, gave an f-measure of 0.648. However, when these were averaged by speech act category, the average f-measure went down to 0.468. This low score is due to a deficit of dictionary entries and we intend to improve upon our performance by extending our gazetteer lists through the use of the GATE

3 http://www.nepomuk.semanticdesktop.org/

4 http://www.semanticdesktop.org/ontologies/

*Annotation Diff* tool, which also allows us to test and tune rules that do not fire. At the current development stage, the service is suitable for providing semi-automatic classification, given that fully automated classification is not yet very reliable.

### 3.2 Supporting Semantic Email Processes

We will now demonstrate the added functionalities of Semantic Email as provided by Semanta by going through the example provided in Section 2 and simultaneously showing the support provided via Semanta's GUI (as implemented for Outlook). The email workflow demonstrated in Fig. 1 can be represented as a timeline depicting *Action Item Threads* within an *Email Thread*. Fig. 3 shows how an email (E1) containing two action items (marked '[?]') is conceptually broken down into two 1-1 transactions between Martin-Dirk (P1) and Martin-Claudia (P2). In the remainder of this section, we trace the paths undertaken by these workflows while demonstrating how Semanta supports Martin, Dirk and Claudia with their semantic email processes.

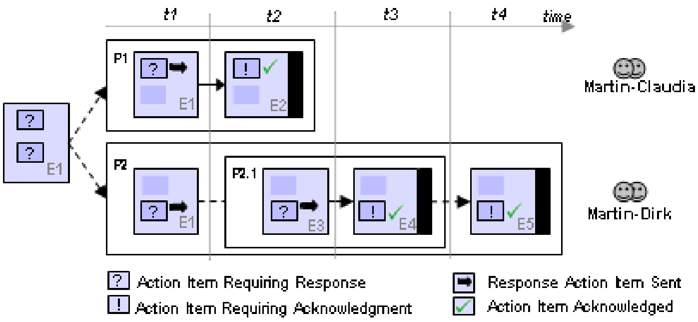



**Fig. 3.** Email breakdown into two separate workflows

#### 3.2.1 Initiating Email Processes

After writing email E1, Martin annotates it via the Semanta toolbar. Specific action items in the content are annotated, rather than the email as a whole. After reviewing the automatically annotated action items, he can remove, change or create new annotations via a right click which brings up Semanta's *Annotation Wizard* (Fig. 4). Rather than providing a list of predefined action items to the user, the wizard supports the user with building annotations by selecting the:

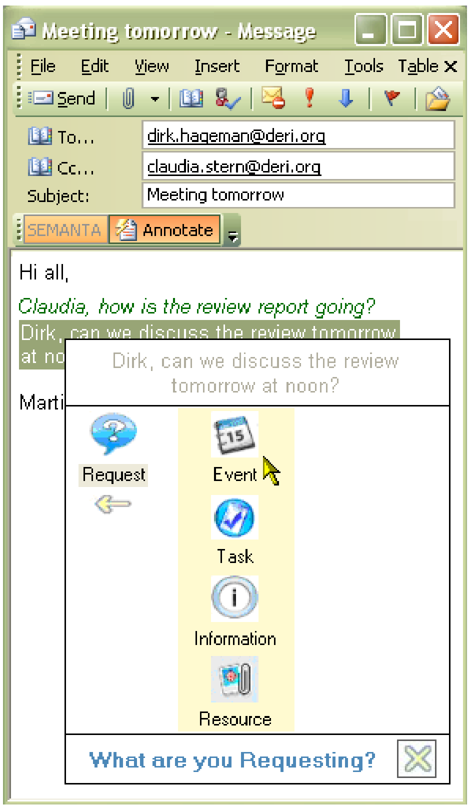



**Fig. 4.** Modifying and creating annotations

1. *Action* that represents the textual statement (e.g. Request, Assign, Suggest)
2. *Object* of the action (e.g. Event, Task, Information, Resource)
3. *Subject(s)* of the action, if applicable (e.g. who is implicated in a Task/Event)
4. *Target(s)* of the action – i.e. to whom the action is directed to.

These are the four defining parameters for a speech act in the sMail Speech Act Model. This process takes up between 3 clicks for simpler speech acts, e.g. the first annotation in E1 – "Request (action) Information (object) from Claudia (target)"; up to 4 or more clicks, depending on the amount of recipients, for more complex speech acts e.g. the second speech act in E1 - "Request (action) a Meeting (object) between Yourself and Dirk (subjects) and direct the request to Dirk (target)". A dynamic sentence (bottom of the wizard) guides the user in constructing the annotation. Objects populating the wizard are dynamically loaded from the sMail ontology via the semantic email service. When Martin sends the email, these annotations together with other metadata are invisibly transported alongside the email content in the custom 'x-smail' email header.

### 3.2.2 Executing Email Workflows

At time *t1* both Dirk and Claudia receive Martin's email. Only one of the action items is addressed to Claudia – the *Information Request*. The other action item – the *Meeting Request*, is addressed to Dirk. Semanta flags incoming email if they possess action items addressed to the current user. Although the email here is flagged as a whole, the flagging is based on how many, if any, action items in the message are addressed to the user. When viewing email with Semanta, users can react to *each* individual action item separately. Depending on its type, they are given a number of appropriate options, as determined by the sMail Email Speech Act Workflow. In Claudia's case, when she right-clicks the *Information Request*, a number of appropriate options are given whereby she can choose to deliver or decline the requested information. She can also ignore the action item (can be undone). More importantly Claudia is allowed to react in any other way. Claudia selects the 'Deliver Information' option, upon which she is prompted to input the required information - "So far, so good". This is automatically annotated as an *Information Delivery* and sent back in a reply email E2 to Martin, who receives and acknowledges it at time *t2*. Below we produce snippets from the RDF generated for this email reply, as transported within the email headers and stored on Claudia's SSD (some URI's have been simplified).

```
<EMAIL2> a <nmo#Email>;
  nmo:from :ClaudiaStern;
  smail:hasPrecedingEmail :EMAIL1;
  smail:hasSpeechAct :EMAIL2SA0;
  pimo:isDefinedBy
    <http://nepomuk.semanticdesktop.org/users/claudias/pimo>;
  nmo:messageId "0000000094A633229025F385F0";
  nmo:sentDate "2008-10-03T13:33:53";
  nmo:to :MartinWilliams .
```

```
<EMAIL2SA0> a <smail#SpeechAct>;
  smail:hasAction <smail#Deliver>;
  smail:hasNoun <smail#Information>;
  smail:hasPrecedingSpeechAct :EMAIL1SA0;
  smail:hasRole <smail#Completive>;
  smail:hasSpeechLength "16";
  smail:hasSpeechStart "46";
  smail:hasStatus <smail#Completed>;
  smail:hasTarget :MartinWilliams;
  smail:recipientExpectation <smail#Acknowledge>;
  smail:senderExpectation <smail#None> .

<MartinWilliams> a <nco#PersonContact>;
  nco:hasEmailAddress :MartinWilliamsEmail;
  <rdfs:label> "Martin Williams" .
<MartinWilliamsEmail> a <nco#PersonContact>;
  nco:emailAddress "martin.williams@deri.org" .
```

The first definition describes the email as a whole, providing an RDF representation of customary email metadata, e.g. sender, recipients, etc.; as well as information specific to Semanta, e.g. references to the speech act within, to the SSD PIMO where this email is relevant (i.e. Claudia), the system (Outlook) message ID as well as a reference to the URI of the preceding email message. The second definition represents the annotation of an information delivery for the text "So far, so good". This annotation consists of an *Information* (noun) *Delivery* (action) addressed to Martin (target). This speech act is in response to the preceding *Information Request* from Martin ("Claudia, how is the review report going?", represented by '#EMAIL1SA0') in the preceding email ('#EMAIL1'). The following properties are defined on the fly by the semantic email service. Given the nature of this action item and of the one preceding it, the role of the former is set to *Completive* – since it is completing the *Information Request* process. Since Claudia is expected to do nothing else on sending this action item (senderExpection), the status of this action item within its representation on her SSD is set to *Completed* right after the email is sent (as shown below). However, since the recipient is expected to acknowledge this action item, the status within its representation in the email sent to Martin will be set to *Pending*, and will remain so until it is acknowledged. The last two definitions describe an email participant. They make sole use of the NCO ontology (and RDF/S) whereas the previous use PIMO, NMO and the sMail ontologies (see 3.1).

Fig. 5 shows Dirk considering the *Meeting Request* he receives from Martin at *t1* - "Dirk, can we discuss the review tomorrow at noon?". On right click, Dirk is given the option to approve, decline or amend the proposal, ignore it or do something else. Dirk chooses the latter option ("Other..") since he decides that he

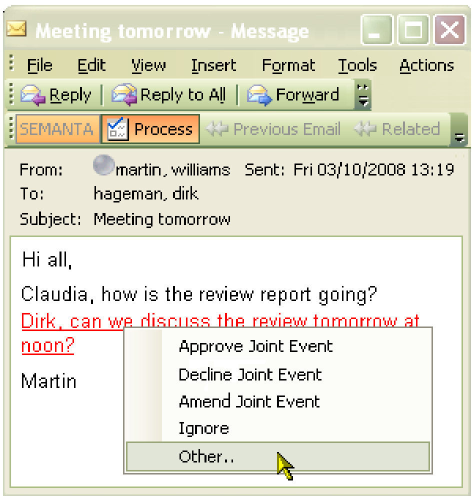



**Fig. 5.** Processing Incoming Email

wants to request more information before making a decision on the meeting request. The annotation wizard is shown right away and Dirk inputs the desired text - "What is there to be discussed?", and annotates it right-away as an *Information Request*. Once Dirk has finished, a reply (E3) with this information request is automatically sent back to Martin. This initiates the sub-workflow P2.1 shown in Fig.3. Martin receives E3 at time *t2* and promptly provides the required information in E4. When Dirk receives E4 and acknowledges this action item at time *t3*, P2.1 is terminated.

### 3.2.3 Tracking Email Workflows

So far we have shown how Semanta can support the user with finding and exchanging action items within emails. We have also shown how by reacting to individual action items within email messages the user is effectively executing implicit ad-hoc workflows with the support of Semanta. We will now demonstrate how Semanta can visualise these implicit email workflows. In the Outlook main window, Semanta's toolbar provides the *Action Items* button for viewing and tracking email workflows. The action item tracker has three main views:

1. Pending Incoming – this view shows all action items (e.g. requests, assignments, suggestions) which require action from the user.
2. Pending Outgoing – this view shows all outgoing action items (e.g. requests) for which the user is still awaiting a reply.
3. All Items – this view shows all incoming and outgoing action items, regardless whether they have been tackled by the user or the user's contacts accordingly.

By viewing the pending incoming action items, the users can tackle those email workflows they are personally stalling. By viewing the pending outgoing action items, the user can view those workflows which have been stalled by someone else and decide whether to send a reminder or otherwise. The all items view shows all exchanged action items within ongoing and terminated workflows.

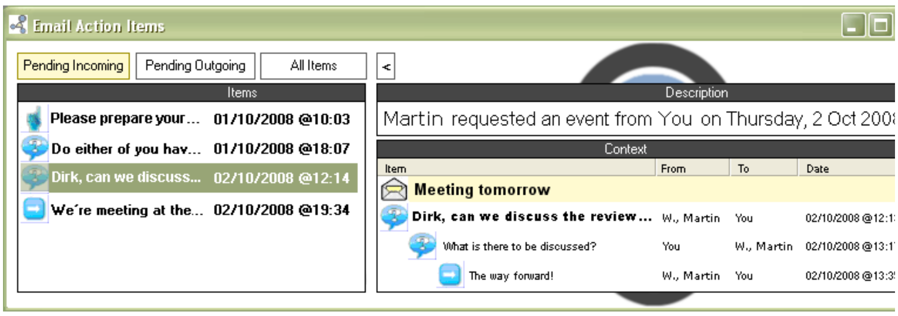



**Fig. 6.** Tracking Email Action Item Threads

For items in all three views, the user can expand the window to view the whole thread of action items exchanged for that workflow. This is presented to the user as the *context* of that action item within its thread. The options for visualising email action items threads are limitless and can be improved. The current design mimics the one for message folders in the host application, without being too similar so as not to confuse the user. Back to our example Dirk chooses to view more information about one of his four pending incoming action item (Fig. 6). He can see that this action item represents the event requested by his manager ("Dirk, can we discuss the review tomorrow at noon?") in email E1. In the context panel Dirk can see that he reacted to this request by asking for more information regarding the event. He can also see that the manager already replied to this request ("The way forward!"). Dirk therefore decides to reconsider the *Meeting Request* and on double-clicking this item he is again shown email E1. He now selects the 'Approve Joint Event' option upon which an automatic reply ("Yes") is embedded as reply text (the text can be changed). This text is automatically annotated as an *Event Assignment* where the event involves both Martin and Dirk, thus binding both of them to the now scheduled meeting.

### 3.2.4 Handling Workflow Artefacts

When Dirk is done viewing email E4, the task assignment is automatically sent in E5 to Martin. Semanta detects that an event involving Dirk and Martin has been generated and presents the item to Dirk for review (Fig. 7), where he can either dismiss it or add it to his Outlook Calendar (an analogous process takes places for detected tasks which can be added to the Outlook Tasklist). On choosing to add the event to his calendar, the native Outlook Appointment Item window is brought up. Semanta populates this item up to a certain degree, and presents it to Dirk for review. Currently Semanta can provide the item subject as extracted from the action item requesting the activity and the one approving

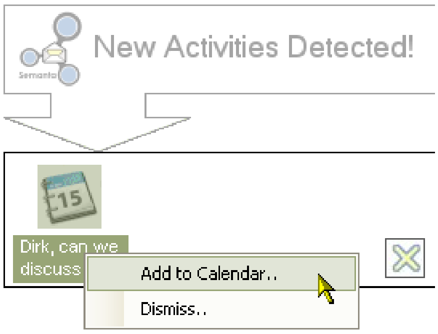



**Fig.7.** Detecting new events/tasks

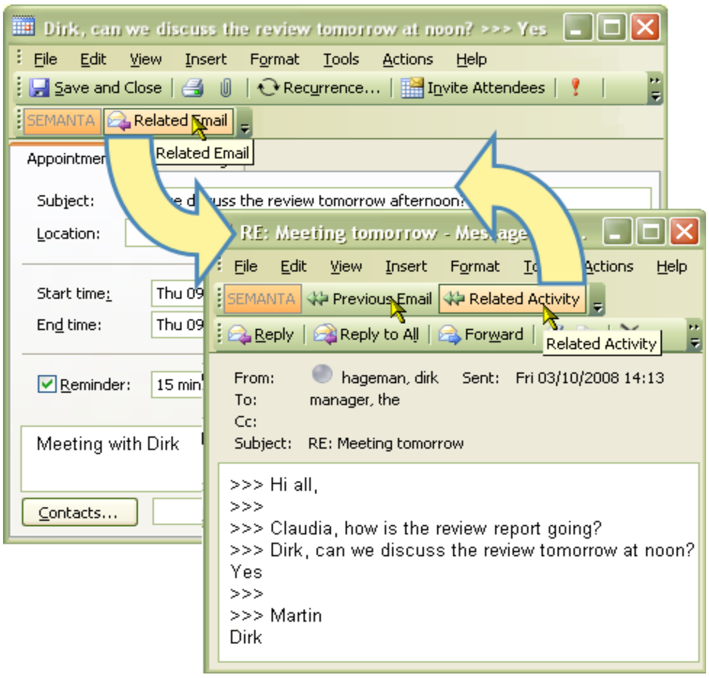
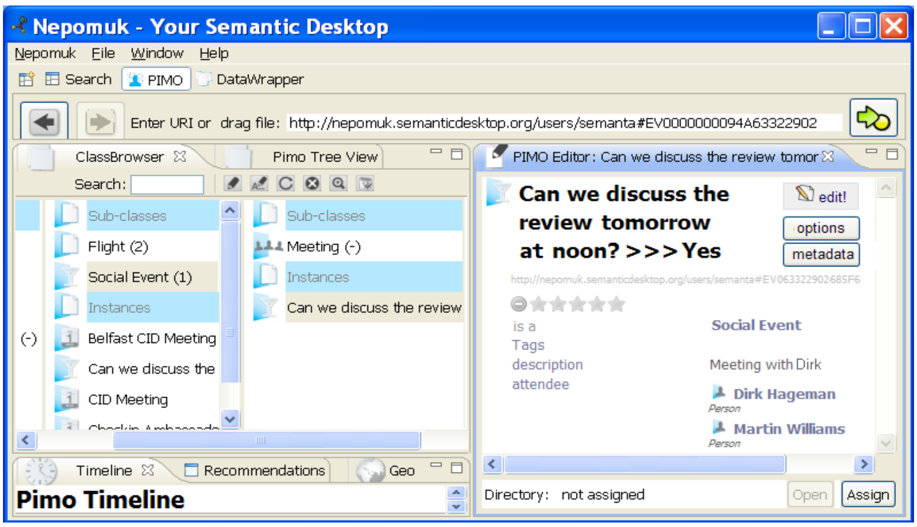



**Fig. 8.** Linking related Email, Task or Event items

it (“Dirk, can we discuss the review tomorrow afternoon? >>> Yes”). Semanta is also aware of the meeting participants and lists them as the appointment attendees. At *t4*, Martin acknowledges the *Event Assignment* in the automatic reply (E5). Since the event assigned also involves Martin, he is similarly prompted to review and/or add the generated event to his Calendar. At this point all the email workflows generated by the first email in our example (E1) are terminated.

Links between event/tasks created via Semanta and the email messages wherein they were generated are stored in the users’ SSD RDF Repositories. Figure 8 shows two buttons provided in the Semanta command bar for this purpose – the ‘Related Email’ button loads the email source for stored tasks and events; whereas the ‘Related Activity’ button points back to any generated tasks/events from an email message. The same linking functionality provided between emails and the activities generated within is also provided for email messages within one email thread. In Fig. 8 the reader can also note that alongside the ‘Related Activity’ button the user is given the possibility to traverse up the email thread via the ‘Previous Email’ button. All these links are stored and retrieved from the SSD's RDF Repository by Semanta’s semantic email service. Semantic interoperability between the various items is thus enabled.

### 3.2.5 Semantic Email on the Social Semantic Desktop




Si

**Fig. 9.** Sharing metadata generated by Semanta across the SSD

nce Semanta is integrated within the SSD, metadata generated by Semanta is also immediately available to other SSD applications and not just to itself. Fig. 9 shows the event generated in our example at time *t4*, as viewed from Martin’s desktop via the P2P Semantic Eclipse Work--bench (PSEW)[5] - an integrated environment based on the SSD architecture allowing the users to browse, query, share and annotate resources to which they have access. Other applications like SSD calendar and task management tools will also have access to these resources. Likewise it must be pointed out that metadata generated by these applications is available to Semanta. In the future we would like to have a system in place where any proposed meetings (for example) prompt Semanta to check for any time clashes with other user commitments as know by the SSD, irrespective of the application wherein they were generated.

## 4 Evaluation

The evaluation process involved 6 computer science researchers. Although this is not large number, previous experiments [15] concluded that 5-12 users are considered as an acceptable number for a system-usability study. In particular Nielsen et. al. found that 5 users can already find 75% of existing problems [16]. Once we improve our system we intend to test Semanta against an even wider audience.

### 4.1 Experimental Setup

The evaluators averaged 32.3 years of age and they had above average familiarity with the MUA used for the evaluation – Outlook. Five typical email situations (workflows) were given to the user for consideration - Task Delegation, Task Acceptance, Data Request, Appointment Scheduling and Event Announcement. To carry out these tasks, the users had to write emails to a number of people with the assistance of Semanta. Whereas some scenarios required only one email to be sent, other scenarios included multiple exchanges, i.e. writing and reading email replies. Other scenarios included the generation of tasks and events from email. Throughout the experiment, the time taken by the user to complete different tasks was recorded. The users' behaviour was also recorded (filmed) during the evaluation and comments, suggestions, problems and errors were noted down. Finally the users were provided with a questionnaire where they could rate different aspects of Semanta with respect to consistency, functionality and design. The evaluation process took an average of 01:23 hours. All material used for the evaluation, including the presentation, scenario outlines, questionnaire, time sheets, videos and the results are available online[6].

### 4.2 Main Findings

The users were quite satisfied with both the automatic and semi-automatic annotation functionalities provided by the system. Of the action items detected by the text

5 http://nepomuk-eclipse.semanticdesktop.org/xwiki/bin/view/Main/PSEW
6 http://smile.deri.ie/projects/semanta/evaluation/

analytics service, 54% were approved by the users without any changes, 32% were improved (i.e. although the automatic annotations made sense, the user felt that the level of sophistication of the annotations could be improved, e.g. a *Request Task* rather than the more generic *Request Information*) via the annotation wizard, and only 13% were either removed or changed completely (i.e. wrong annotations). Of the final number of exchanged annotations, 27% where manually created by the users (i.e. not detected by the service). Echoing the latter was the general feeling amongst users that automatic annotation had higher precision than recall. This reflected a decision taken at the implementation stage; whereby we acknowledged that it is better for action items to pass undetected, rather than having Semanta trying to support the users with inexistent workflows bound to incorrectly annotated action items. With regards to the Annotation Wizard, although users demanded more meaningful tooltips for items within, they found the semi-automatic annotation experience easy and intuitive. Thus we feel that the effort to abstract the sMail Speech Act Model and semantic annotation beneath a simple, user-friendly interface was successful. On the down side, 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 is not satisfactory given that this means that users were still taking around as much time to annotate the email as to write it. However, it is a fact that throughout the evaluation many users were giving suggestions and 'thinking-aloud', thus considerably increasing the time spent on each annotation task.

With respect to reading and processing incoming email, the evaluators were satisfied with the options given for different incoming action items, i.e. Semanta's support for email processes, and there were no cases when the users were unsure of which option to select. The users liked the event/task detection and were satisfied with the partial automatic population provided when adding these items to their task list/calendar. They also appreciated the links generated between detected tasks/events and their source email messages as well as the fact that one can traverse up the email thread from the email message window.

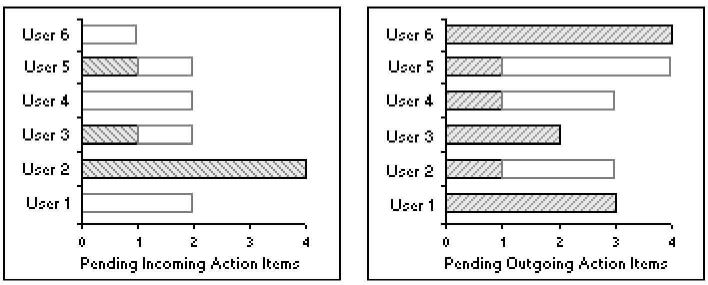



**Fig. 10.** Tracked action items with (outline) and without (shaded) Semanta

After the users carried out the scenarios, they were asked to enumerate all exchanged pending outgoing action items, i.e. action items they have sent for which they were still awaiting a reply; and pending incoming action items, i.e. action items they have received and still needed to consider; that they could remember. The users were allowed the use of Outlook for this task, i.e. looking at exchanged emails in the inbox and sent items folders. Although this process took the users an average of 6.66 minutes, they still could not point them all out. The same information can be instantly and fully retrieved by Semanta. On average, after browsing exchanged messages, the users could recall just 33% of total pending incoming email action items (Fig. 10). The recall rate for the outgoing email action items amounted to 65%. 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. In conclusion, the users were highly appreciative of Semanta's action item tracking function – which effectively provides visualisation for ad-hoc email workflows.

### 4.3 Improving Semanta

After the evaluation we made a number of improvements, both in terms of functionality as well as the user interface. With regards to the text analytics service we are still improving the recognition of persons involved in the email (taking advantage of existing structured sources such as Address Books) but have since included simple co-reference resolution. We will investigate the use of ML techniques to improve both precision and recall of automatic annotation. Eventually we want to bring the ratio of time required to annotate an email against the time required to write, annotate and send it to at most 25%. We also plan to extend the text analytics service in order to capture date and time information for detected tasks and events. Although we criticised the system implemented in Mangrove [3], we acknowledge that in some cases the option of using fixed templates can save the user's time. We therefore added QuickShot email function, offering a selection of email templates where the purpose is predefined and the user is required only to input the content.

We will provide the following additions to Semanta's email action item tracking. When viewing pending outgoing action items, users will be able to quickly send reminders to their contacts to inform them that they still await a response. Reminders will also be incorporated in the sMail Workflow Model, as special kinds of email messages. Action item thread visualisation will be extended such that, when a workflow ends with a generated event/task, these are also incorporated in this view and in the case of tasks, their status will be dynamically updated. For example, after Claudia accepts a *Task Request* from Dirk, it will be shown at the end of the workflow in both Claudia's and Dirk's action item tracker. When Claudia eventually ticks the task done, the task will be marked completed in both systems.

## 5 Conclusion

In this paper we have presented a fully-implemented system supporting Semantic Email Processes – Semanta. We have shown the reader how Semanta can assist the user with the execution of previously implicit ad-hoc email workflows. The main contribution of this paper is the successful implementation of a system whereby we have seamlessly integrated semantic technology supporting semantic process management into the existing technical landscape, used existing transport technology, and hidden a complex workflow model and semantics under an intuitive user interface that can be used by anyone on their own computer. In contrast, the additional functionalities enabled by these semantics are clearly visible to the end-user. We believe that this is exactly what the Semantic Web needs to be taken-up by the computer-literate society at large. We verified the acceptance and applicability of the user interface by an experimental case-study, where one of the goals was to find out how well we can actually hide the semantic technology. The results of the evaluation have confirmed our success in these respects.

**Acknowledgments.** The work presented in this paper 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.

## References

1. Scerri, S., Handschuh, S., Decker, S.: Semantic Email as a Communication Medium for the Social Semantic Desktop. In Proc. European Semantic Web Conference 2008. Tenerife, Spain (2008).
2. McDowell, L., Etzioni, O., Halevey, A., Levy, H.:Semantic Email. In Proc. World Wide Web, (2004).
3. McDowell, L., Etzioni, O., Gribble, S., Halevey, A., Levy, H., Pentney, W., Verma, D., Vlasseva, S.: Evolving the Semantic Web with Mangrove. UW Tech Report, (2003).
4. Scerri, S., Mencke, M., Davis, B., Handschuh, S.: Evaluating the Ontology underlying sMail – the Conceptual Framework for Semantic Email Communication. In Proc. Language Resources and Evaluation 2008, Marrakech, Morocco (2008).
5. Searle, J.: Speech Acts. Cambridge University Press (1969).
6. Winograd, T., Flores, F.: Understanding Computers and Cognition: A New Foundation for Design, Norwood, NJ: Ablex, pp. 220. (1986).
7. Curtis, B.: Can speech acts walk the talk? In Proc. of Comput. Supported Coop. Work 3(1): pp. 61-64. (1994).
8. Voorhoeve, M., Van der Aalst, W.: Ad-hoc workflow: problems and solutions. In: 8th International Workshop on Database and Expert Systems Applications (DEXA07) (1997).
9. Scerri, S.: Semanta – Your Personal Email Semantic Assistant. Demo Session at the Intelligent User Interface Conference 2009, Gran Canaria, Spain (2008).
10. Decker, S., Martin, F: The Social Semantic Desktop. Technical Report (2004).
11. Sauermann, L., van Elst, L., Dengel, A.: PIMO - a Framework for Representing Personal Information Models. In Proc. I-Semantics' 07, pp. 270--277. (2007).
12. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In Proc. 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02). Philadelphia, (2002).

13. Cunningham, H., Maynard, D., Tablan, K.: JAPE: a Java Annotation Patterns Engine (Second Edition). Technical report CS--00--10, University of Sheffield, Department of Computer Science (2000).
14. Khosravi, H., Wilks, Y.: Routing email automatically by purpose not topic. Natural Language Engineering, Vol. 5: pp. 237–250. (1999).
15. Dumas, J.S., Redish, J.C.: A practical guide to Usability Testing (revised edition). Intellect, Exeter. (1999).
16. Nielsen, J.: Heuristic Evaluation. In J. Nielsen and R.K. Mack (editors), Usability Inspection Methods. John Wiley & Sons, New York, (1994).