The Case for Explicit Knowledge in Documents

Leslie Carr
Intelligence, Agents, Multimedia Group
University of Southampton, UK
lac@ecs.soton.ac.uk

Timothy Miles-Board
Intelligence, Agents, Multimedia Group
University of Southampton, UK
tmb@ecs.soton.ac.uk

Arouna Woukeu
Intelligence, Agents, Multimedia Group
University of Southampton, UK
aw1@ecs.soton.ac.uk

Gary Wills
Intelligence, Agents, Multimedia Group
University of Southampton, UK
gbw@ecs.soton.ac.uk

Wendy Hall
Intelligence, Agents, Multimedia Group
University of Southampton, UK
wh@ecs.soton.ac.uk

ABSTRACT

The Web is full of documents which must be interpreted by human readers and by software agents (search engines, recommender systems, clustering processes *etc.*). Although Web standards have addressed format obfuscation by using XML schemas and stylesheets to specify unambiguous structure and presentation semantics, interpretation is still hampered by the fundamental ambiguity of information in `#PCDATA` text. Even the most easily distinguishable kinds of knowledge such as article citations and proper nouns (referring to people, organisations, projects, products, technical concepts) have to be identified by fallible, post-hoc extraction processes. The WiCK project has investigated the writing process in a Semantic Web environment where knowledge services exist and actively assist the author. In this paper we discuss the need to make knowledge an explicit part of the document representation and the advantages and disadvantages of this step.

Categories and Subject Descriptors

I.2.1 [**Artificial Intelligence**]: Applications and Expert Systems—*office automation*; I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods; I.7.2 [**Document and Text Processing**]: Document Preparation—*markup languages*

General Terms

Human Factors

Keywords

Document Structure, Semantic Web, Knowledge Writing

1. BACKGROUND

The initial development of the World Wide Web drew from an established body of research into document processing; the current development of the Semantic Web is assimilating work from the Artificial Intelligence and Knowledge Management communities.

1.1 Document Processing 101

The computer model of documentation has evolved from 80-column ASCII files, through various kinds of presentational markup (TEX, troff) to so-called structural markup (LATEX and SGML) [2]. The aim has been to enable computers to provide as rich a presentation style as possible for information (that uses fonts, graphics, colour and layout structures as effectively as possible to implement visual design) and then to make that presentation specification as independent of the document contents as possible. This is often achieved through the use of separately stored stylesheets referred to by the attributes of content elements.

The result is a document that can be interpreted efficiently by human eyes. The effect of the style separation (or parameterisation) is to allow the document to be reproduced in different display contexts (with different publication regimes, such as A4 report, gatefold booklet, web page, poster) or reprocessed for different information environments and databases. The use of XML on the Web reinforces this separation between content and presentation, and allows both human authors and software agents to focus on the information that is to be communicated, rather than on the way in which it is to be delivered.

1.2 Semantic Web 101

The aim of the Semantic Web [3] is to build on this content-neutral platform to provide an environment in which computational agents can unambiguously determine the meaning of a resource, to make the Web an environment in which software agents and humans can make better (reasoned) use of the available resources.

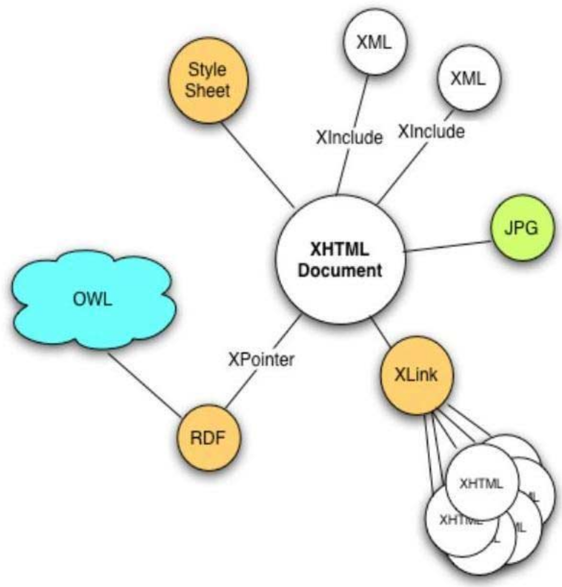


Figure 1: The context in which a document is read.

The key components of the Semantic Web are (a) agreed models (ontologies) of the objects and relationships contained in the documents (b) formally specified ontology languages for unambiguously codifying these agreed models and (c) an annotation mechanism for identifying (parts of) Web documents with concepts from relevant ontologies.

1.3 Annotating for Semantics and Annotating for Presentation

Presentational- and semantic- processing are not dissimilar. The fundamental concepts involved with independent layout and formatting processes as developed through ODA and DSSSL and which are now embodied in CSS and XSL-FO are parts of community-agreed ontologies, expressing the concepts of a page, margin, border, block, paragraph etc. The particular languages in which they are expressed (CSS, XSL) are loosely analogous to the ontology languages of the Semantic Web in that they describe the composition and useful conjunction of the (display) concepts. RDFs mechanism for conjoining concept and content (via structure addressing and namespace reference) has one counterpart in the document processing: the external link (XLink or HyTime).

Stylesheets either achieve this implicitly by subverting particular content nodes (CSS uses `style` or `class` attributes) or by declaring a generalised content-visiting process (XSLT). It would not be impossible to interchange mechanisms and to identify individual elements for stylesheet processing with RDF. Conversely, it would be possible to define a concept attribute for content elements to explicitly add ontological semantics to the document content. In fact, this is an approach undertaken by the earliest Web ontology systems such as SHOE [14].

1.4 Adaptive and Active Documents

A document is often considered as structure plus content, and constructed at a low level from a number of different storage entities or XIncluded fragments (figure 1). Presentation specifications are mapped onto structure or specified in it. Hypertext links are specified similarly, but some link semantics may affect the content inclusion (e.g. XLink's `actuate="auto" show="embed"`).

All of these devices are available to the author, the visual designer or the web site manager. Typically, the author is considered to have control over the content of the document rather than its display. However, the content itself may include directives which generate or manipulate document components. These may simply add timestamps, navigational panels or adverts, but they can also be used to *adaptively* insert or delete relevant subject material [4].

Active documents are documents which contain or are directly associated with executable program code or scripts. Placeless documents [7], for example, have *active properties* in addition to regular properties (title, author, *etc.*). As well as a name and value, each active property includes a third component — code, which runs in response to various actions upon the document (e.g. readContent, writeContent, deleteDocument). Active properties take functionality normally associated with specific applications, such as workflow or content conversion, and associate them directly with the document so that they travel around with it (e.g. by email). Microsoft's "smart documents" are an example of active documents: "available in Microsoft Office Word 2003 and Microsoft Office Excel 2003, smart documents contain programming logic that defines the way documents are used, and controls the way the data in the documents can be manipulated" [19].

2. DOCUMENT PARADIGM FOR THE SEMANTIC WEB

However a document is constructed, and whatever flexibility is provided in its storage and construction, the defining feature of a document is *an author*; a document is the bounded communication of a human individual. It is the content identified at a particular time, with a particular meaning for a particular purpose. The way in which this meaning and purpose is represented within the document has a direct impact on the successful (or otherwise) interpretation of the content by its consumers.

In the case of traditional plain ASCII documents, although easily processable by both human and software agents, knowledge is represented implicitly and has to be extracted by analysing the text (e.g. by using NLP techniques); subsequently interpretation of the document content is open to error and ambiguity. In the case of Postscript or Adobe PDF documents, software agents have to extract the textual content from drawing code before processing it. $\LaTeX$macros and HTML markup may give software agents some clues as to the structural semantics of the text, but are ultimately ambiguous.

XML reduces ambiguity in documents by making the document structure explicit through a schema, for example DocBook[1]. The interpretation of the document structure is therefore unambiguous, but the concepts and ideas within the text itself — the `#PCDATA` — are unlikely to have schematic equivalents. These need to be in place to facilitate the interpretation of the document at a finer granularity than its structural blocks.

[1]`http://docbook.sourceforge.net/projects/schema/`

RDF facilitates this by associating knowledge with text spans at the character level via XPointers. However, this pointing mechanism becomes a problem when the content of the target document is changed, potentially causing XPointers to reference the wrong text span, or to become unresolvable ('broken'). This problem has been termed the *editing problem* by the Open Hypermedia community [6]. A straightforward solution to this problem is to *embed the knowledge markup in the content itself*, at least while the documents is being authored or undergoing editorial changes, so that the knowledge survives (e.g., by moving with the text as content is inserted, deleted, or copied and pasted to a new location in the document).

SHOE and XMP are examples of formats for embedding knowledge in documents. SHOE [14] extends HTML with a handful of extra tags for describing the document *as a whole* in relation to a domain ontology. Adobe's XMP [1], allows RDF constructs to be embedded in HTML, PostScript and PDF documents, as well as TIFF, JPEG, GIF, PNG and all Adobe formats (Photoshop, Illustrator *etc.*). XMP-compliant applications provide built-in support for a functional set of XMP schemas including Dublin Core, rights management, and EXIF, although new schemas can also be defined. However, the XMP portion within the document representation is separate from the content, so in the case of a PDF for example, a fragile pointer mechanism would still be required to associate knowledge with specific spans of text during authoring/editing. One solution offered by a number of knowledge writing tools is to use an ad-hoc representational format for embedded knowledge during the authoring or editing of a document, which can be extracted to a formal representation (e.g. SemanticWord [21] — DAML+OIL, SemTalk [8] — RDFS/DAML) when the document is published. Such tools are the subject of the next section.

3. CREATING SEMANTIC WEB DOCUMENTS

For knowledge markup to become an explicit part of Semantic Web document representation, we also need to consider how this step can be supported during the process of authoring new documents.

In an attempt to make parts of the Web corpus amenable to machine-processing, much emphasis within the Semantic Web community has been on building tools for the manual annotation of *existing* documents [11]. However, recognising that post-hoc manual annotation is difficult, time-consuming, and error prone, researchers have turned to producing automatic or semi-automatic methods for adding knowledge annotations to existing documents in order to make this process more feasible [17, 23, 12]. Amilcare [5], for example, uses the generalised NLP rules learnt from an annotated test corpus to create new knowledge annotations on an unseen corpus. Amilcare has been used as a fully automatic process, and also as a 'suggester' where the annotations are revised by a human annotator (these revisions may in turn fine tune the extraction rules).

However, while this post-hoc mining exercise works well as an approach for 'enabling' a legacy corpus, it is not a paradigm that should be used in the case of creating *new* documents. Semantic annotation, overseen by the author, should be an essential and active component of the writing process itself, shifting the responsibility for knowledge markup from system/annotator to author.

The Writing in the Context of Knowledge (WiCK) project aims to produce tools which capture the knowledge and intentions of the author rather than just capturing the author's keystrokes and attempting to guess afterwards what they actually mean — in short, to facilitate the author in communicating his or her ideas clearly and unambiguously to human and machine interpreters in the Semantic Web. A number of other approaches have also investigated this *simultaneous* authoring of content and semantic markup, which we describe briefly before turning to the contributions of the WiCK project itself.

CREAM [10] allows an author to build new documents by dragging and dropping knowledge fragments from an ontology browser into a text editor — for example a dropped instance slot inserts a text rendering of the slot value; a dropped relationship slot inserts a short sentence complete with links to each of the related instances. In both cases, knowledge is embedded in the document alongside the inserted text.

Knowledge markup at authoring-time does not preclude the use of information extraction techniques such as that demonstrated by Amilcare. Extracted knowledge can be used to offer relevant services to the author in order to assist the writing task. For example, ARIA [18] supports email or web page authoring based on a semantically annotated photo database. By continuously monitoring the text typed by the author against a domain ontology, ARIA recommends photos from the database that seem appropriate to illustrate the various facets of the unfolding narrative.

The potential research and commercial benefits of bringing these knowledge-aware processes into the office arena have not gone unnoticed. Microsoft Word, for example, is the most often adopted product for authoring text documents [21]; authors can therefore adopt new knowledge-aware extensions without learning a new production environment and without sacrificing familiar features [22]. SemanticWord [21], a Microsoft Word-based environment, adds several toolbars to the standard interface which support the creation of semantic annotations according to selected ontologies (local or imported from the Semantic Web). Using these toolbars, authors can associate a text region with an instance of a class (an *instance reference*), or describe the content of a text region with a collection of triples — both types of annotation are embedded in the text and can be directly manipulated. Annotations are "carried over" in text cut/copy and paste operations, facilitating a level of knowledge reuse between documents. As well as knowledge-rich documents, the Semantic Word toolbars can also be used to create *annotated templates*, thus speeding up content and annotation production in frequently created documents.

SemanticWord also offers a more proactive information extraction feature which the author experiences through the Microsoft Smart Tags interface. As with ARIA, the author's keystrokes are monitored by an information extraction process which relates named entities in the text to ontology instances and types, visually highlighting the recognisedd text in the document. The author can then examine the highlighted entities and convert them into instance reference annotations.

Although provoking a range of reactions upon its release [15], Smart Tag technology has also been adopted by other office-based knowledge writing initiatives, including SemTalk [9] and OntoOffice [20]. As with SemanticWord, recognised concepts and instances are highlighted with Smart Tags. However, the kinds of action offered differs between systems: in SemTalk, for example, the author can access and edit the underlying ontological model; in OntoOffice, a search for context-relevant documents can be initiated.

Going beyond simply supporting knowledge writing in the context of an underlying ontology, the WiCK project has attempted to build on these initiatives by considering an office environment in which several knowledge-bases and knowledge-aware services exist and actively assist the author by providing targeted knowledge that would otherwise need to be searched for both manually and individually.

4. WICKOFFICE: A KNOWLEDGE WRITING ENVIRONMENT

As we have argued in previous sections, authors of new Semantic Web documents face an additional responsibility of explicitly embedding knowledge markup in their documents. However, just as author's worries about linking responsibilities when writing for the Web were alleviated by search engines such as *Google*, the presence of such knowledge-aware services (for example, SemanticWord's knowledge markup as-you-type) and resources (for example, OntoOffice's searchable document store) is essential. The WiCK project has investigated the writing process in a Semantic Web environment where knowledge repositories and services exist and actively assist the author in producing a document with explicit embedded knowledge. In order to demonstrate our approach, we consider a business-type scenario where an author is tasked to produce a funding proposal for a project. By analysing and subsequently modelling the knowledge 'flow' in this scenario we can demonstrate the benefits that a knowledge-aware office environment can provide.

4.1 Scenario

The task of writing a funding proposal is common in industrial and commercial environments; here, we consider a hypothetical funding proposal for a research project in an academic environment. The proposal is directed at the UK's Engineering and Physical Sciences Research Council (EPSRC), which has a well-defined procedure for submitting, reviewing, and selecting proposals for funding, and provides a standard form[2] (the Je-SRP1) and a comprehensive guidance document[3] on how to fill out the form, create the supplementary documentation, and submit it for consideration (table 1).

The Je-SRP1 form itself serves as an administrative summary of the research proposal, collecting together the relevant information about the hosting organisation, project investigators, project partners (for joint proposals), referees, staff (including visiting researchers), and travel and equipment costs. The 'meat' of the proposal is contained in the supplementary document — the Case for Support — the composition of which is tightly defined in the guidance notes. The rules for the Case define the formatting (constraints on page length, font sizes *etc.*), the information content, and the structure of parts and sections where each of these pieces of information should be placed. The content of the Case for Support includes previous research track records, proposed research programme and methodology, proposed dissemination routes, and justifications for each of the resources requested in the Je-SRP1 form.

[2]http://www.epsrc.ac.uk/website/commonpages/downloads.aspx?CID=4482

[3]http://www.epsrc.ac.uk/website/commonpages/downloads.aspx?CID=8621

1. Je-SRP1 Form — Administrative summary of research proposal.
2. Guidance Notes — Instructions for filling out the Je-SRP1 form and creating the supplementary Case for Support documentation.

(a) Documents provided by the EPSRC.

1. Completed Je-SRP1 Form.
2. Case for Support — Supplementary documentation containing 'meat' of proposal.

(b) Documents submitted to the EPSRC.

Table 1: Research proposal documents provided by, and subsequently submitted to, the EPSRC.

In order to properly model the Je-SRP1 form and Case for Support document and the knowledge they contain, and hence be able to deploy it usefully in a computational environment more complex than a search engine, our scenario requires a number of ontologies. Firstly, we need to understand and model *what is being written about.* To meet the requirements for our scenario, we propose a *research ontology* to describe the stakeholders and activities who participate in research — the researchers, their publications, research interests, conferences and journals, and a *subject ontology* to describe the area in which we wish to conduct research, the problems that we wish to address and the methods, systems and approaches which have been described in the literature.

Having modelled the subject domain of the writing task, we next need to understand the 'design specification' for the writing task itself — *what needs to be written.* We therefore propose a *document ontology* to make explicit the semantic structure of the proposal documents — the pages, sections, paragraphs, forms, and fields. In order to explicity model the *type* of information that the author must enter into each part of this structure, a *project ontology* capturing the activity of undertaking work — the ideas of work package, budget, personnel, milestones *etc.* — and a *proposal ontology* — describing the objectives, beneficiaries, funding call, and programme of activity for the proposed project are proposed.

It follows that filling in the Je-SRP1 form is mainly a matter of choosing appropriate instances against the above ontologies from the knowledge-base. The initial fields on the form are for the host organisation's name and reference, and the name of the Principal Investigator. The explicit constraints on this information (according to the Guidance Notes) are that the host organisation must be of a specific type (e.g. UK Higher Education) and that the PI must be employed by the host organisation and must have a contract of the appropriate type (an academic, duration at least as

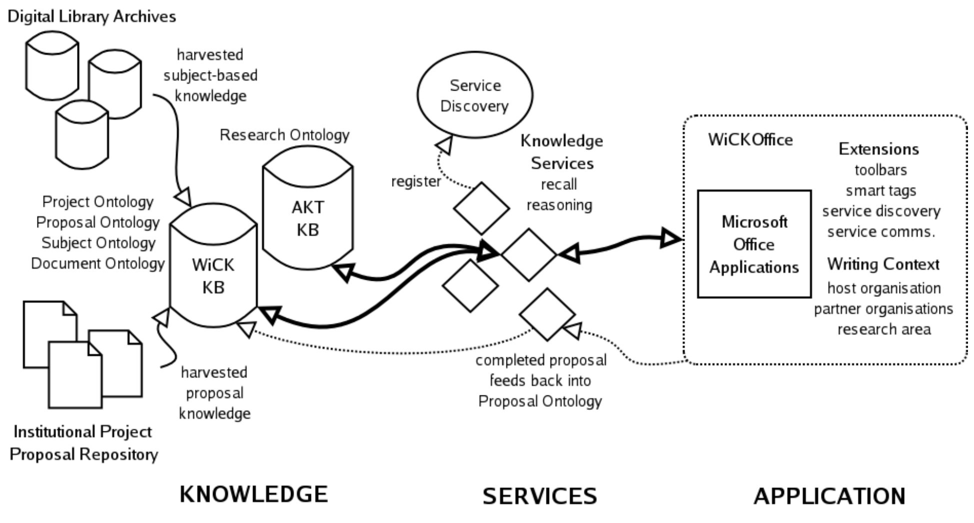


Figure 2: The proposed WiCKOffice knowledge writing environment.

long as the project lifetime). These simple constraints can easily be modelled as verification conditions on data entry, or as queries upon the knowledge-base to select an appropriate list of choices. De-constructing the form in this way therefore provides an outline proposal ontology, with the Guidance Notes document supplying the constraints.

Creating the Case for Support document is more involved, as the author is required to construct a text, rather than enter data into clearly labelled spaces on a form. However the Guidance Notes document indicates very clearly the kind of information that is expected in each part of the document. Examining the bullet points which give instructions for Part 1 of the Case for Support, we can see what basic information is required from the knowledge-base, in addition to the kind of processing and analysis which would need to be performed on it:

Provide a summary of the results and conclusions of recent work in the technological/scientific area which is covered by the research proposal. Include reference to both EPSRC funded work and non-EPSRC funded work. Details of relevant past collaborative work with industry and/or with other beneficiaries should be given... This specifies a literature review; the knowledge is described by the subject and research ontologies. A simple query of the knowledge-base or digital library would provide a list of potentially relevant papers, but a more advanced reasoning agent would be required in order to assist the author in evaluating the relative significance of the projects and papers.

Part 2 of the Case for Support requires a different kind of knowledge support, for instance within the *Program and Methodology* section: *Identify the overall aims of the project and the individual measurable objectives against which you would wish the outcome of the work to be assessed.* This information does not exist in the knowledge-base; it is invented as an integral part of the creation of a new research undertaking. However, authors may be assisted by seeing the aims and objectives of similar, recent or successful project proposals, especially if they do not have much experience of proposal writing to draw on. In other words a lack of personal experience could be supplemented by directed browsing of an institutional memory.

This brief examination of the EPSRC Guidance Notes for a project proposals shows how heavily the writing process (both apparently free-text content creation and information recall) is constrained and specified by the appropriate ontologies, opening the possibility of substantive help from a suitably equipped knowledge environment.

4.2 Proposed Architecture

Figure 2 illustrates our proposed knowledge-aware office environment, WiCKOffice, designed in response to the opportunities for functionality identified in the previous section. In this environment, knowledge is managed by two knowledge-bases, both based on the AKT 3Store platform [13]. The AKT knowledge-base models the UK Higher Education computer science community[4] (expressed using the AKT Reference Ontology[5]), and hence provides a suitable research ontology for our purposes. A WiCK knowledge-base hosts the additional ontologies. Instances for the project and proposal ontologies are acquired from previous EPSRC project proposals; we envision Semantic Web agents trawling digital library archives and automatically constructing and populating the subject ontology.

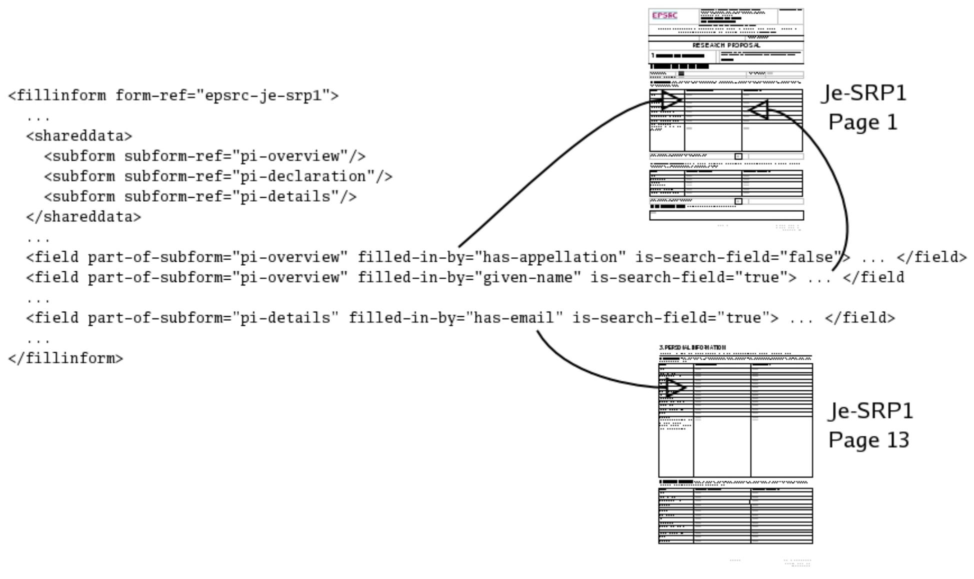
WiCK extensions to the Microsoft Office environment (both VBA and COM-based) utilise key computational knowledge services to assist the writing task (in accordance with the 'writing context'), and to update the knowledge-bases when the writing task is completed (for example, new proposals becoming part of the "institutional memory"). Explicit knowledge representation in the proposal documents makes the latter a straightforward process.

5. WICKOFFICE PROTOTYPE

Based on the opportunities for functionality identified in the previous section, our modelling and development efforts to date have produced a coherent WiCKOffice environment in which several knowledge services are available to authors. A *knowledge fill-in* service and *knowledge recall* service are motivated by the need to provide timely and convenient ac-

[4]http://www.hyphen.info/

[5]http://www.aktors.org/publications/ontology/



```
<fillinform form-ref="epsrc-je-srp1">
  ...
  <shareddata>
    <subform subform-ref="pi-overview"/>
    <subform subform-ref="pi-declaration"/>
    <subform subform-ref="pi-details"/>
  </shareddata>
  ...
  <field part-of-subform="pi-overview" filled-in-by="has-appellation" is-search-field="false"> ... </field>
  <field part-of-subform="pi-overview" filled-in-by="given-name" is-search-field="true"> ... </field
  ...
  <field part-of-subform="pi-details" filled-in-by="has-email" is-search-field="true"> ... </field>
  ...
</fillinform>
```

Figure 3: Augmenting Je-SRP1 template with explicit structural semantics facilitates assisted knowledge fill-in.

cess to knowledge, which would otherwise have to be manually 'looked up' on the institutional intranet. A third service, *in-line guidelines*, also assists recall by exposing guidelines and constraints captured from a design specification (in this case, the EPSRC guidance notes), that are relevant to the part of the proposal document currently being worked on, via the Microsoft Office Assistant interface.

5.1 Filling In Forms

The *knowledge fill-in* service assists the author in filling in the Je-SRP1 form. For example, the author can specify the (partial) name of the Principal Investigator and instruct the service to retrieve appropriate (in context) instances from the knowledge-base to automatically fill in the remainder of the required information.

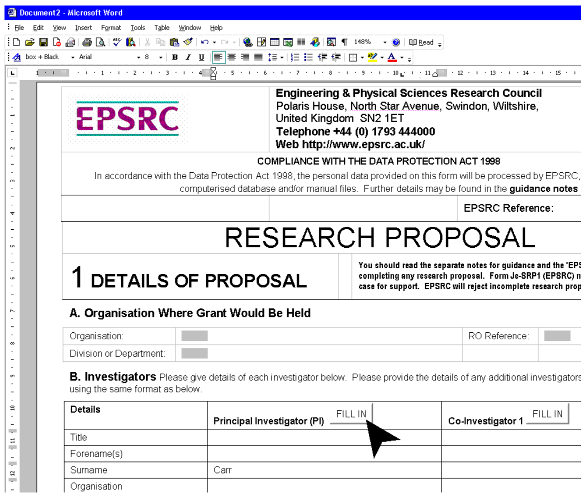
The majority of the information required to provide an assisted knowledge fill-in service for the Je-SRP1 form is already provided by the AKT Reference Ontology (the *research* ontology in our scenario). However, leveraging this service is not as simple as filling each part of the form with an appropriate instance selected from the research ontology — different parts of the Je-SRP1 form "share" data about the same concept. For example, information relating to the Principal Investigator must entered in three different locations: section 1B (page 1) requires the PI's title, name, organisation, department, and commitments to other projects; section 2B (page 12) requires the PI's name (for the proposal declaration); and section 3B (page 13) requires the PI's contact telephone number, email address, fax number, *etc*.

In line with our paradigm for Semantic Web documents (section 2), we have used Microsoft Office 2003's new "smart documents" feature to add *semantic structure* to the otherwise unstructured Je-SRP1 template in the form of an XML Schema derived from the document ontology. The XML Schema identifies each 'sub-form' of the Je-SRP1 and groups together related sub-forms (thus, for example, describing the fact that information about the PI is shared by sub-forms 1B, 2B, and 3B). Each individual form field is marked up with three attributes — the ID of the sub-form to which the field belongs, a boolean value indicating whether that field is a preferred search field (in the case of the Je-SRP1, the PI's first name and surname are good search terms for a person instance in the research ontology; knowing the PI's title may not so helpful), and finally a `filled-in-by` attribute which identifies the slot of the matching knowledge instance which should be used to actually provide a value for the field.

When the author partially fills in a sub-form (figure 4a) and presses the "Fill-In" button, the XML structure of the document is consulted to determine which fields are part of the current sub-form (and also which fields are part of other sub-forms that share data with the current sub-form). Fields in the current sub-form with an `is-search-field` attribute value of `true` are then used by the knowledge fill-in service to construct an RDQL query to extract matches from the research ontology. In the case that multiple instances match the query, these instances are presented to the author who chooses the appropriate match. Finally, the `filled-in-by` attribute is used to map the slot values of the returned instance to each associated field (figure 4b); the URI of the matching instance from the research ontology is also embedded.

Recently, the EPSRC rolled out its own assisted form filling system, the Je-S1 e-form[6], which provides some equivalent functionality to this service. Provided that each party has previously registered their details with the system, the author can select the host organisation, principal and co-investigators, referees and other staff from checklists and then download a partially completed JE-SRP1 form which contains all the required details of the selected parties, but still requires some unaided 'mandraulic' effort to complete in full. By contrast, we argue that the WiCKOffice approach of leveraging the functionality of multiple services operating over diverse *knowledge* sources (including, but not restricted to, employee data and information harvested from

[6]`https://je-s.rcuk.ac.uk/`


Document2 - Microsoft Word
File Edit View Insert Format Tools Table Window Help
148%
Read
box + Black
Arial
8
FILL IN
FILL IN

EPSRC

Engineering & Physical Sciences Research Council
Polaris House, North Star Avenue, Swindon, Wiltshire,
United Kingdom SN2 1ET
Telephone +44 (0) 1793 444000
Web http://www.epsrc.ac.uk/

COMPLIANCE WITH THE DATA PROTECTION ACT 1998
In accordance with the Data Protection Act 1998, the personal data provided on this form will be processed by EPSRC,
computerised database and/or manual files. Further details may be found in the **guidance notes**

EPSRC Reference:

RESEARCH PROPOSAL

1 DETAILS OF PROPOSAL

**You should read the separate notes for guidance and the 'EPS
completing any research proposal. Form Je-SRP1 (EPSRC) m
case for support. EPSRC will reject incomplete research prop**

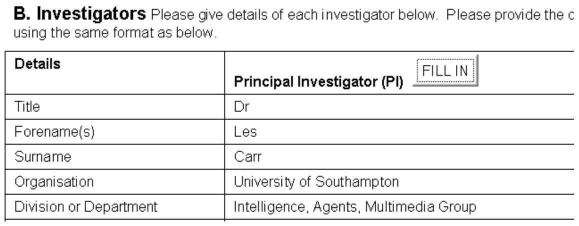
A. Organisation Where Grant Would Be Held

Organisation:		RO Reference:	
Division or Department:			

B. Investigators Please give details of each investigator below. Please provide the details of any additional investigators using the same format as below.

Details	Principal Investigator (PI)	Co-Investigator 1
Title		
Forename(s)		
Surname	Carr	
Organisation		

a. Author fills in partial details.



B. Investigators Please give details of each investigator below. Please provide the c
using the same format as below.

Details	Principal Investigator (PI)
Title	Dr
Forename(s)	Les
Surname	Carr
Organisation	University of Southampton
Division or Department	Intelligence, Agents, Multimedia Group

b. All sub-forms sharing data with current sub-form are populated from matching instance.

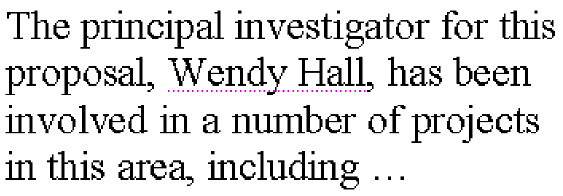
Figure 4: Using the knowledge fill-in service via the WiCKOFfice toolbar.

personal webpages and directories) not only allows authors to be aided in filling in *all* aspects of the Je-SRP1 form but also potentially offers wider applicability (adding new types of form requires only that form's semantic structure be elicited according the document ontology) than a *data*-based application.

5.2 Knowledge In The Right Place At The Right Time

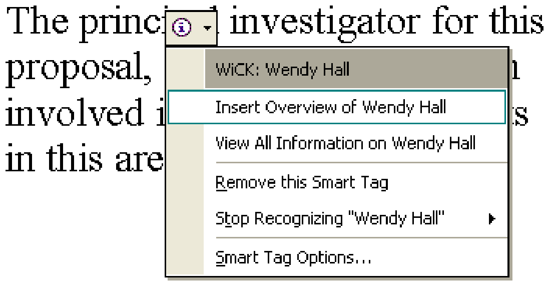
The *knowledge recall* service assists the author in quickly and conveniently recalling appropriate knowledge from the research environment. Example (contextual) queries include "what papers relevant to this proposal have been published recently?", or "what relevant projects has this person worked on?". In response to such queries, appropriate knowledge from the knowledge-bases is selected and inserted directly into the document in the form of 'potted' summaries.

As with the *knowledge fill-in* service, the AKT Reference Ontology provides the majority of knowledge utilised by this service. In the current implementation, given the name of a recognised person, project or place, the knowledge recall service assists the writer in recalling facts about it. We have seen that recent incarnations of Microsoft Office already provide a mechanism for recognising terms and presenting available "actions" associated with that term to the user in the form of Smart Tags. However, in Case for Support document, the author's information requirements depend on the section or part of the document currently being worked on. For example, the author might expect that typing "Les Carr" in the *Previous Research* section would

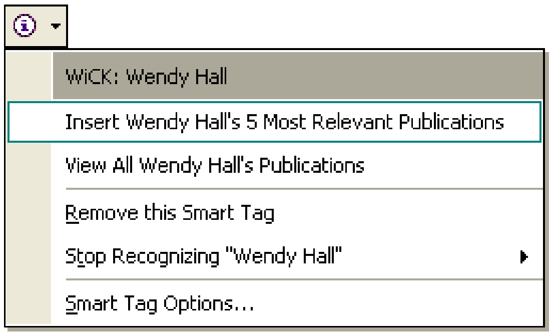


The principal investigator for this proposal, Wendy Hall, has been involved in a number of projects in this area, including …

a. Name recognised as author types.



b. Available actions in Previous Research section.


WiCK: Wendy Hall
Insert Wendy Hall's 5 Most Relevant Publications
View All Wendy Hall's Publications
Remove this Smart Tag
Stop Recognizing "Wendy Hall"
Smart Tag Options…

c. Available actions for recognised text "Wendy Hall" in References section.

Figure 5: Using the knowledge recall service, via the WiCKOffice Smart Tag.

make available options to "auto-summarise" or browse those facets of Les Carr's previous research history most relevant to the current proposal, whereas typing "Les Carr" in the *References* section would make available options to insert Les Carr's most recent and relevant publications, and typing "Les Carr" in the *Researcher Curriculum Vitae* section would make available options to insert a "mini CV" with information appropriate to the proposal (with appropriate embedded knowledge markup in each case). However, prior to the release of Microsoft Office 2003, the actions made available through Smart Tags have been static; Office 2003 allows the set of available actions to be determined dynamically when the author activates (clicks on) a Smart Tag [16].

An XML Schema derived from the document ontology is again used to make explicit the structural semantics of the Case for Support document. When the author activates a WiCK Smart Tag by clicking on a highlighted term in the text, the XML structure of the document is consulted to work out which part of the document the text appears in (e.g. Background, References) and the actions offered by available services which are appropriate to the type of knowledge required in that section are presented (figure 5).

We therefore describe this service as providing knowledge in the right *place* (the author's current location in the document) at the right *time* (when a name of a recognised person, place or project is typed by the author). Again we anticipate the wider applicability of this type of service beyond the specifics of our scenario; with appropriate knowledge sources, services, and discovery mechanisms in place this 'right place, right time' writing paradigm can be applied to other writing tasks.

5.3 Planned Future Services

Two further knowledge-based services are currently under development within the project proposal writing scenario. An *augmented experience* service provides the author with access to the "institutional memory" of previous research proposals, thereby augmenting the author's own experience of proposal writing ("what works? what doesn't work?"). For example, the author is assisted in evaluating the most important beneficiaries of the proposed research by being shown the beneficiaries put forward by other proposals (with an indication as to whether those proposals were subsequently approved or otherwise).

An *assisted writing* service attempts to assist the author in making higher-level decisions about relevant content to include in the proposal by suggesting appropriate instances from the subject ontology (for example, relevant projects, papers, resources) based on both the writing context and the text that the author has already written. For example, this service uses an internal reasoning engine to detect that although the author has referred to a number of *knowledge acquisition*-related projects in the *Background* section[7] of the Case for Support, one particularly 'significant' project has not yet been mentioned, and so offers to create a summary of the project from the relevant instances in the knowledge-base (gathering details of key personnel and publications) and inserts the knowledge-annotated information into the appropriate sections of the Case document.

6. CONCLUSIONS AND FUTURE WORK

In order to allow documents to be unambiguously interpreted by both human readers and software agents, knowledge should be an explicit part of document representation. Rather than being the result of an imprecise, after-the-fact activity, knowledge elicitation in Semantic Web documents can be an exact, author-assisted process. However, instead of manifesting this additional responsibility as an extra process of annotation, knowledge elicitation can be an indistinguishable part of the authoring process. In fact the knowledge elicitation process can actually help the author (or editorial staff) rather than adding an extra burden, by providing a synthesis of a range of targeted background material that would otherwise need to be searched for both manually and individually. This paper has introduced our contribution to this process, WiCKOffice, a knowledge-writing environment. In the context of a project proposal writing scenario, WiCKOffice demonstrates that with a suitable set of ontologies and a supportive knowledge-aware environment, an author can be assisted in producing explicit knowledge documents.

[7]Guidance notes: "Demonstrate a knowledge and understanding of past and current work in the subject area both in the UK and abroad."

As well as facilitating unambiguous interpretation of its content, the knowledge-augmented document can then be intelligently processed in further ways, for example by the proposed Assisted Writing service in the WiCKOffice environment. The documents are then used to update the RDF-based knowledge services, asserting the new facts that the author has created. In our example these are particularly easy to determine by taking advantage of the explicit knowledge representation in the completed proposal documents. Hence making knowledge representation an explicit and separate part of the document construction process makes the writing process easier to support.

Our future work plans, aside from continued implementation of our integrated office environment, include a more detailed focus on the processes and mechanisms by which the knowledge provided by the AKT and WiCK knowledge-bases can be updated and maintained as more and more research proposals are produced. We also plan to carry out a systematic user evaluation — academics who write research proposals as part of their day-to-day work are a readily exploitable human resource in our department. Lastly, we are also working on a writing methodology for creating more complex, knowledge-rich documents such as multi-faceted Web sites and hypertexts.

7. ACKNOWLEDGEMENTS

This work has been funded in part by the EPSRC Knowledge Writing in Context (KWiC) project (GR/R91021/01) — now known as *Writing in the Context of Knowledge (WiCK)* — and the EPSRC Advanced Knowledge Technologies IRC (GR/N15764/01) in the UK.

8. REFERENCES

[1] Adobe Systems Incorporated. XMP Adding Intelligence to Media. `http://www.adobe.com/products/xmp/pdfs/xmpspec.pdf`, 2004.

[2] D. W. Barron. Why use SGML? *Electronic PublishingOrigination, Dissemination, and Design*, 2(1):3–24, Apr. 1989.

[3] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, May 2001.

[4] P. Brusilovsky. Methods and techniques of adaptive hypermedia. *User Modeling and User-Adapted Interaction*, 6(2-3):87–129, 1996.

[5] F. Ciravegna and Y. Wilks. Designing Adaptive Information Extraction for the Semantic Web in Amilcare. In S. Handschuh and S. Staab, editors, *Annotation for the Semantic Web*, Frontiers in Artifical Intelligence and Applications. IOS Press, Amsterdam, 2003.

[6] H. C. Davis. Referential integrity of links in open hypermedia systems. In *Proceedings of the ACM Hypertext '98 Conference, Pittsburgh, USA*, pages 207–216, 1998.

[7] P. Dourish, W. K. Edwards, J. Howell, A. LaMarca, J. Lamping, K. Petersen, M. Salisbury, D. Terry, and J. Thornton. A programming model for active documents. In *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology*, pages 41–50, 2000.

[8] C. Fillies. On Visualizing the Semantic Web in MS Office. In *Proceedings of the 6th International Conference on Information Visualisation (IV'02), London, England*, pages 441–446, 2002.

[9] C. Fillies, G. Wood-Albrecht, and F. Weichardt. A Pragmatic Application of the Semantic Web using SemTalk. In *Proceedings of the Eleventh International World Wide Web Conference, Honolulu, Hawaii, USA*, pages 686–692, 2002.

[10] S. Handschuh and S. Staab. Authoring and Annotation of Web Pages in CREAM. In *Proceedings of the Eleventh International World Wide Web Conference, Honolulu, Hawaii, USA*, 2002.

[11] S. Handschuh, S. Staab, and A. Maedche. CREAM — Creating relational metadata with a component-based, ontology-driven annotation framework. In *Proceedings of the First International Conference on Knowledge Capture (KCAP 2001), Victoria, B.C., Canada*, Oct. 2001.

[12] S. Handschuh, S. Stabb, and F. Ciravegna. S-CREAM - Semi-automatic CREAtion of Metadata. In *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW'02)*, 2002.

[13] S. Harris and N. Gibbins. 3store: Efficient Bulk RDF Storage. In *Proceedings of the 1st International Workshop on Practical and Scalable Semantic Systems (PSSS'03), Sanibel Island, Florida*, pages 1–15, 2003.

[14] J. Heflin, J. Hendler, , and S. Luke. Reading Between the Lines: Using SHOE to Discover Implicit Knowledge from the Web. In *Proceedings of the 1998 Conference on Artificial Intelligence*, 1998.

[15] G. Hughes and L. Carr. Microsoft Smart Tags: Support, ignore or condemn them? In *Proceedings of the ACM Hypertext 2002 Conference, Maryland, USA*, pages 80–81, 2002.

[16] C. Kunicki. What's New with Smart Tags in Office 2003. *MSDN OfficeTalk*, 2003. Available from `http://msdn.microsoft.com/library/en-us/dnofftalk/html/office01022003.asp`.

[17] T. Leonard and H. Glaser. Large scale acquisition and maintenance from the web without source access. In *Proceedings of the Knowledge Markup and Semantic Annotation Workshop (K-CAP 2001)*, pages 97–101, 2001.

[18] H. Lieberman and H. Liu. Adaptive Linking between Text and Photos Using Common Sense Reasoning. In *Proceedings of the Conference on Adaptive Hypermedia and Adaptive Web Systems, Malaga, Spain*, pages 2–11, 2002.

[19] Microsoft Corporation. Smart Documents in Microsoft Office 2003: Summary Technical White Paper. `http://www.microsoft.com/technet/prodtechnol/office/office2003/operate/smrtdcsy.mspx`, 2003.

[20] ontoprise GmbH. OntoOffice Tutorial. `http://www.ontoprise.de/documents/tutorial_ontooffice.pdf`, 2003.

[21] M. Tallis. Semantic Word Processing for Content Authors. In *Proceedings of the Knowledge Markup & Semantic Annotation Workshop, Florida, USA*, 2003. Part of the Second International Conference on Knowledge Capture, K-CAP 2003.

[22] M. Tallis, N. M. Goldman, and R. M. Balzer. The Briefing Associate: Easing Authors into the Semantic Web. *IEEE Intelligent Systems*, 17(1), 2002.

[23] M. Vargas-Vera, E. Motta, J. Domingue, M. Lanzoni, A. Stutt, and F. Ciravegna. MnM: Ontology driven semi-automatic or automatic support for semantic markup. In *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW'02)*, 2002.