Journal of Intelligent Manufacturing manuscript No.
(will be inserted by the editor)

An Information System to support the Engineering Designer.

Richard Crowder · David Fowler · Quentin Reul · Derek Sleeman · Nigel Shadbolt · Gary Wills

Received: date / Accepted: date

Abstract Engineering companies are currently shifting their focus from selling products to providing services, hence the products' designers must increasingly consider life-cycle requirements, in addition to conventional design parameters. To identify possible areas of concern, engineers must consider knowledge throughout the life cycle of similar or related products. However, because of the size and distributed nature of a company's operation, engineers often do not have access to front-line maintenance data. In addition, the large number of documents generated during the design and operation of a product makes it impractical to manually review all documents thoroughly during a task. As a case study, this paper discusses the concept and development of a large hypermedia based *Knowledge Desktop* that has been developed to support the maintenance and future design of aircraft engines. As part of the development cycle, the performance of the software and its acceptance by the user community has been fully evaluated. The evaluation method considered in this paper focuses on the subjective opinion of the users and measures the ease with which users could retrieve the information required to perform specific tasks.

Keywords Aerospace Engineering, Hypermedia, User Evaluation, Semantic Web, Industrial Application.

Richard Crowder, David Fowler, Nigel Shadbolt and Gary Wills
School of Electronics and Computer Science, University of Southampton, Southampton.
E-mail: rmc, drf, nrs, gbw @ecs.soton.ac.uk

Quentin Reul
STARLab, Vrije Universiteit Brussels, B-1050 Brussels.

Derek Sleeman
Department of Computer Science, University of Aberdeen, Aberdeen.

1 Introduction

Within manufacturing industries it is now recognized that with the increased emphasis on innovation to maintain an organization's competitive edge, knowledge has become a key asset in many companies as opposed to capital and labor (Reed et al, 2009). Knowledge may be described as the additional contextual understanding of facts that provide a framework for decision making. Hence a loss or lack of knowledge to support a particular decision can lead to an increase in costs, additional design or manufacturing effort. Knowledge is therefore a valuable commodity. This paper considers the development of a knowledge management system that is able to support knowledge reuse within a high value engineering design process.

Our work will be discussed in the context of aero-engine manufacturing where a fundamental shift is occurring, from selling products to providing services. For example, Rolls-Royce set the target of making half its engine fleet subject to profitable long-term service agreements by 2010 (Harrison, 2006; Xu and Wang, 2009). Essential to success in this market shift is to ensure that designed new products are optimized for the aftermarket. In other words, new products must be designed to provide lower and more predictable maintenance costs. To minimize maintenance costs throughout the engine's life cycle, engineers must obtain and actively use knowledge gained from maintenance histories of similar products during the design phase of new products. This approach will help engineers identify parts most likely to raise issues throughout the engine's life cycle. However, in organizations which perform a substantial amount of engineering design, a very large number of documents are created. It is therefore impossible for any

single member of a design team to access more than a fraction of available documentation.

In understanding the design process and interaction between design engineers it must be recognized that the modern aircraft engine is considered to be one of the most complex machines ever designed, by any measures of complexity. The design incorporates a large number of advanced technologies including high temperature materials, complex fluid dynamics and high speed rotating components. For this reason aircraft engine design is typically undertaken by a number of engineering teams who are responsible for individual subsections of the engine, e.g compressor or turbine. As a result, it becomes increasingly difficult for engineers, either individually or as teams, to follow information trails through the design, maintenance and other related resources to provide the support that designers and front line activities require (Wills et al, 2004). The challenge for organizations is therefore to develop an information system that is both comprehensive and will satisfy the increasing demands from industry for up-to-date and easily accessible information. In the solution of large and complex engineering problems, it can be difficult to locate all the resources relating to a specific issue. This is especially difficult when these resources are stored in legacy systems.

The proposed solution presented in this paper is based on ontological hypertext, which we applied to a number of previous industrial applications (Fowler and Sleeman, 2004; Wills et al, 2004). Ontological hypertext is a variant of hypertext whose structure and links are derived from the relationships between individual objects in the application domain. In addition it is also related to schematics hypertexts that define a linking schema as a design mechanism for structuring document collections; ontological hypertext extends the schema into the domain of world knowledge, allowing inference mechanisms to deduce meaningful documents links (Carr et al, 2001b).

A further problem in an application such as this is the management of the segmentation of the information space (Maneewatthana et al, 2005). One possible solution is to take a modular approach as discussed in Crowder et al (2000). However, there is a move away from the monolithic or tightly coupled applications into a more loosely coupled Service Orientated Architecture (SOA) as discussed by Wilson et al (2004). A Service Orientated Architecture implements the functionality of the system using Web services, and provides an attempt to modularize large complex systems in such a way that they are constructed from independent software components that offer services to one another through well–defined interfaces. This approach allows any component to be replaced at a later date to improve the other functionality of the system, with minimal effort.

Ontological hypertext and SOA add additional complexity to hypermedia design (De Bra et al, 2004), hence this must be reflected in the design process as required. In particular ensuring that the underlying knowledge, behavior and presentational aspects are all tied together correctly. Ontological hypertext, overcomes this problem by generating the associations between resources based on the semantics of their content (Carr et al, 2004a). The ontologies can be derived from the document structure, the tasks being undertaken and the roles of the people using the application.

As is widely recognized hypertext aims to mimic the way people think by capturing the association between resources as links; this is of particularly relevance within the engineering design domain, where large systems bring their own set of problems as first identified by Malcolm et al (1991). Some of these challenges have been addressed, and reported in Heath et al (2000). This work demonstrated how structural links can be generated from the structure of documents. However, for the associative linking the solution proposed used human authored links and this is not considered practical for large industrial hypermedia applications.

The paper is structured as follows: the following section discusses the context of the work as applied to the engineering design domain. In section 3 we discuss related work both from the design reuse, and information system perspective. The design and implementation of the *Knowledge Desktop* is discussed in sections 4 and 5. The initial evaluation of the *Knowledge Desktop* is discussed in section 6, which is followed by our conclusions.

2 Context

The work presented in this paper proposes a solution to the challenges that many large engineering companies face — that of cost-effective collecting, maintaining and retrieving their corporate knowledge. As noted above, the shift from selling products to providing services is significant with many aircraft engine manufactures. Essential to the success of this market shift is to design new products with lower and more predictable maintenance costs. This approach to engine provision has considerable similarities to the "Total Care" concept described in Alonso-Rasgado and Thompson (2007) that identifies benefits to both the manufacturer and the customer. To minimize maintenance costs throughout the engine's life cycle, engineers must obtain knowledge gained from maintenance histories of similar products during the design phase of new products. This will

help engineers identify parts most likely to raise issues throughout the engine's entire life cycle. It should be recognized that the design of an aeroengine is typically undertaken by a number of teams who are responsible for individual engine modules, for example combustion or turbines. Therefore it is impossible for any single member of a specific design team to access more than a fraction of the available documentation. As noted by (Wills et al, 2004) information systems usually develop over time into a set of heterogeneous resources with ill-defined metadata, and as a result, it becomes difficult for the user to follow a specific trail through the resources. The challenge for organizations is therefore to develop an information system that is both comprehensive and will satisfy the increasing demands from industry for up–to–date and easily accessible information (Qiu, 2006).

Due to the importance of increasing operational reliability and minimizing maintenance costs in the new market environment of product support, information gained in the operation of a fleet of engines needs to be fed back to the designers of subsequent engines. However, the current information infrastructure makes this difficult as design engineers responsible for conceptual designs do not have full and easy access to maintenance knowledge. Similarly, design engineers should consult existing maintenance documents to help design parts with more predictable maintenance costs. As a result, we need to strengthen and help formalize the information flow between the company's aftermarket operations and the design teams. This would allow the knowledge gained during the design, production and operation of an engine to advise the design of the next variant.

Figure 1 summarizes the information flow during the life cycle of an engine, and illustrates the following key points:

- As an engine is designed, manufactured, and operated, there is a flow of knowledge between the design and production activities to support the current engine version (represented by the solid lines);
- Knowledge, including design rationale, for the design, production, and operation of one engine variant provides an invaluable resource to the designers of the next version of the engine (represented by the dotted lines);
- Knowledge derived from the operation of an engine, either from engine monitoring or from issues reported by the users is fed back to the designers of subsequent versions (represented by the dashed lines).

The development of the conceptual design is the initial stage of the process and results in a set of broad requirements, such as a specified amount of thrust and the target aircraft, from which engineers determine the approximate dimensions, weight, power and other costs for the engine design. In the design stage, engineers transform the preliminary abstract design into a set of designs that can be used in production. In the manufacturing stage, engines are built according to the issued design. Traditionally, after production and sales, responsibility for the engine passed from the manufacturer, to the airlines, who own the engine. The airlines are responsible for maintaining the engines, and are supported by the manufacturers through their aftermarket provision. To assist maintenance engineers to identify possible issues earlier, engines are commonly equipped with a large number of engine monitoring sensors. This monitoring information can be analyzed for abnormal operating conditions, such as excess temperature or pressure, which is valuable for the designers of subsequent versions of the engine (Jackson et al, 2005).

The potential use and benefit from such an intelligent document repository are considerable. The maintenance engineers are involved in the day-to-day servicing of the engines, and are thus responsible for the initial population of the document repository with maintenance reports and other similar documents. Firstly, the documents provide the maintainers with a set of lessons learned, allowing a reduction in maintenance time; secondly they provide trend information over the engine fleet, allowing the identification of underlying design or operational issues; and thirdly they provide knowledge resources for the development of subsequent engines. It should be noted that this intelligent document repository is just one iof a considerable number of tools that need to be brought together to design an aircraft engine.

3 Related Work

In this section we describe previous work relating to design reuse and industrial information systems. In addition, we discuss the underlying technologies and methodology which we used to deliver the demonstrator systems.

3.1 Design Reuse

Design reuse describes the application of past concepts to a new problem. Research in engineering design has long emphasized the important of knowledge sharing (e.g. (McMahon et al, 2004; Sim, 2004)). It has been

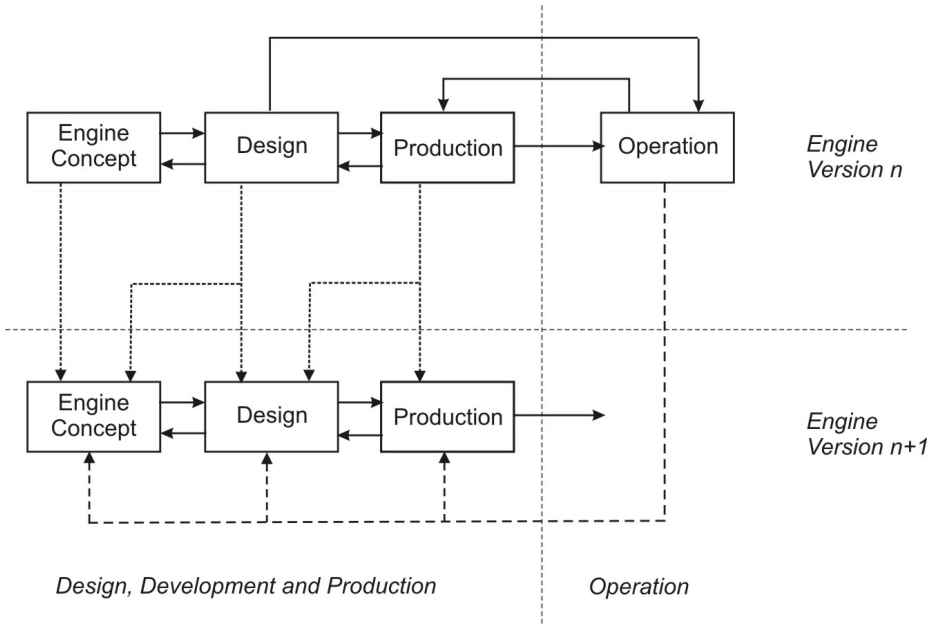


Fig. 1 The flow of information gained during the life-cycle of one engine variant (n) is used to inform the design of the next variant (n+1). The solid lines represent the information flows to support the original engine's operation, while the dotted arrows indicate the flow of design rationale and related knowledge to the new engine design team and the dashed line represents operational information being used to inform the design process.

estimated that 90% of the industrial design activity is based on variant design (Goa 1998), while during a redesign activity up to 70% of the information is taken from previous solutions (Khadilkar and Stauffer, 1996; Lanfranchi et al, 2007). As discussed by Poli et al (1992), the exchange of knowledge throughout the life cycle can improve process performance and increase the quality of the product's design. As is recognised, knowledge can be either tacit (intuitive, experimental, and based on heuristics) or explicit (coded or structured in a formal manner). In the application we have developed we provide the designer with a tool set that will allow the linkage of both types of knowledge in a semi-structured environment. The study by Ahmed and Wallace (2004) showed that the experienced designers rely heavily on their own previous designs with which they are familiar, i.e. reuse without dissemination.

The work by Busby (1998) identified a considerable number of challenges to reuse as well as the potential benefits. In the paper the key barriers to knowledge transfer and subsequent reuse were identified as:

- Lack of training and the belief that reuse leads to deskilling.
- The underlying design process inhibits reuse, through the loss of ownership of the design as it is modified.
- Organizational issues, including the incentive to reuse information.

Thus in order to encourage knowledge reuse, social as well as technical solutions are required. However if satisfactory reuse is obtained, the benefits to the overall manufacturing process are significant and include, (Busby, 1998):

- The use of an existing design avoids the reuse of resources that have already been spent on the original design.
- Helps to avoid the uncertainty and errors that are associated with a new design.

However despite these benefits to the overall business, design reuse at the organizational level appears to be remarkably low (Reed et al, 2009). In order to improve reuse the designer must be provided with the tools that permit easy reuse at the desktop, typically by easy-to-use search engines, this forms the underlying philosophy of the *Knowledge Desktop* presented in this paper.

3.2 Industrial Information Systems

The work described in this paper was informed by our previous work with Rolls-Royce. In Crowder et al (2003), we presented a future vision for the working practices of designers within a manufacturing organization. We have found that engineering design environments are highly distributed in nature and are characterized by

a large number of information sources, which together with the designers forms a complex sociotechnical system. It is concluded that a range of knowledge management tools would be required to support this future vision of engineering design environments (Wallace et al, 2001). Therefore one of the objectives of our current work is to define a future engineering design environment, with particular emphasis on the social and technical systems that will support designers in their day-to-day activities. One of the key components of this approach is the concept of the *Knowledge Desktop*, that gives the designer a single access point to an organization's knowledge repositories (Crowder et al, 1998, 2000).

In Wills et al (2004), we created a document repository from distributed and heterogeneous engineering document resources. When an engineer searches for documents within the repository, the system generates a list of documents ordered according to the engineer's role and the user's related concepts. Thus, the document retrieval process is intelligent and adapts to the user's role within the company. However, we found that engineers actually want the knowledge that is buried within the documents, instead of the actual documents themselves. This is due to the large volume of documents available within the repository which makes it very time consuming to peruse thoroughly. In other words, engineers prefer to see summary reports of documents archived. For example, when searching for engine part failures, engineers want to see how many times the failure of a particular part leads to engine removal, but not the list of original maintenance documents. Another example of useful knowledge that can be extracted from engineering documents is an expertise finder (Cohen et al, 1998; Crowder et al, 2002). The expertise can be obtained by integrating author information and the contents of documents. Thus, for our new engineering document repository, we aim to include the ability to provide analysis of information stored, in addition to simple document search.

Our work in Wills et al (2004) can be seen as a domain specific digital library, with the extension that information presented is adapted to the role of the user. Digital libraries concentrate on the problem of searching for documents distributed over multiple repositories. For example, Priebe and Pernul (2003) developed a portal over multiple document repositories by using an integrated metadata store. As a result, users can search on both the content of the documents and their metadata. In contrast, document index functionality does not form part of our proposed infrastructure. However, global document indexes can be provided to our knowledge repository as services that implement document indexes and metadata indexes. Another area that digital libraries concentrate on is dynamic links generation to relevant document resources (Nnadi and Bieber, 2004; Malcolm et al, 1991; Carr et al, 2001a). Dynamic links are injected into documents automatically during presentation time, and do not alter the original documents. These dynamic links can point to related documents, or even services such as searching annotation and peer reviews (Nnadi and Bieber, 2004). In comparison, our proposed system does not perform dynamic link injection on existing documents. However, it can provide a list of suggested documents as a service.

Carr et al (2004a) extended the concept of digital libraries to include the dissemination process that leads to publications, hence developing the Digital Review Journal (DRJ), which is a knowledge base where users from different roles can collaborate through the entire process on writing an academic paper. It includes support for experiment creation, execution to publication. In comparison, our work concentrates on extracting existing knowledge generated from outside the framework. In addition it should be noted that the DRJ does not handle the integrating knowledge from heterogeneous document sources.

A related problem that relates to large scale design knowledge reuse is that the social problems become significant due to the interaction between knowledge providers and knowledge users, this leads to the concept of social translucence (Erickson and Kellogg, 2000; El-Tayeh et al, 2008). This work concentrates on the codifying of the knowledge and the mechanism that will allow the users to come together and negotiate sharing. It can be argued that a purely technical solution, limits how tacit knowledge can be codified and shared. Other possible solutions include IDRAK (El-Tayeh et al, 2008) and the increasing use of wiki based technologies (Guan et al, 2009).

Finally, there are also related projects working on creating new semantically marked up data for large knowledge repositories, such as (Ciravegna and Wilks, 2003; Carr et al, 2004b). Creating new documents is outside the scope of our project, as we concentrate on the problem of delivering knowledge from existing documents. However, employing techniques to generate semantic information automatically for new documents to be deposited into the knowledge repository will improve document analysis and searching inside our framework.

3.3 Development process

The *Knowledge Desktop* application discussed in this paper was developed through an approach which draws significantly on current hypertext design models and

methodologies. As advances in web applications have forced the development of new methodologies to support the construction of complicated, data intensive Web applications, improved methodologies have contributed tremendously to the way in which systems are designed and implemented.

This work employed the agile approach to software development that was proposed in the mid 1990s as a reaction to the limitations of traditional software development methodologies and business change, as well as a result of people's experience and reusing ideas from history, (Abbas et al, 2008). The term *agile* is an umbrella for a number of well-defined methods, which vary in practice. However, these methods share common principles and values that were defined in the Agile Manifesto (Highsmith et al, 2001). This Manifesto gives more importance to: individuals and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation and to responding to change over following a plan. The Manifesto's values and principles can be used to form a definition of an Agile method as adaptive, iterative and incremental, and people oriented (Abbas et al, 2008):

- **Adaptive:** an agile method welcomes change, in technology and requirements, even to the point of changing the method itself. In addition, it responds to feedback about previous work.

- **Iterative and incremental:** The software is developed in several iterations, each from planning to delivery. In each iteration, part of the system is developed, tested, and improved while a new part is being developed. In each iteration, the functionality will be improved. The system grows incrementally as new functionality is added with each release. After each iteration, a release was delivered to the customer in order to obtain feedback.

- **People-oriented:** people are more important than any process. Good people with a good process will outperform good people with no process every time. In an Agile method, people are the primary drivers of project success. Therefore, the role of the process in an Agile method is to support the development team to determine the best way to handle work. Furthermore, an Agile method emphasises face-to-face communication within the team and with the customer who is closely involved with the development process rather than written documents.

The agile philosophy firmly believes in the abilities of people. The role of customers in the various phases of this model is much higher compared to the other models. The customers are allowed to participate in the decision making processes. It also provides equal opportunity for development of individuals involved in the process by continuous swapping of membership (Nerur et al, 2005).

The process relies heavily on end user involvement, that generally leads to user buy-in. Persona and scenario based modelling is commonly used in software engineering, but we have found that the end users can find the process a little confusing. However, by walking them through previous examples, giving them templates, and through the use of co-design workshops they can help produce detailed personas and scenarios.

3.4 Service Oriented Architectures

A Service-Oriented Architecture (SOA) enables large complex systems to comprise of independent software components that operate through well-defined interfaces. This is ideally suited to more loosely coupled systems, where individual parts may be developed by different people or organizations. Wilson et al (2004) discuss in detail the advantages of using a SOA; the ability to couple services dynamically, interoperability of services due to clearly defined standards, and as a result the ability to avoid technology "lock-in". Due to the nature of the loose coupling in a SOA, applications can be developed and deployed incrementally. In addition, new features can be added easily after the system is deployed. This modularity and extensibility makes an SOA especially suitable in situations with evolving requirements and standards.

A Web service is mainly made up of three components: Service provider, service broker and service requester. The integration of services is achieved through connection, communication, description and discovery. Web services make use of XML to connect and communicate with services using the Simple Object Access Protocol (SOAP). The description of facilities offered by various services in machine readable format is done by Web Services Description Language (WSDL). Universal Description, Discovery and Integration (UDDI) generates a registry of services available which helps the service brokers to choose the service required by them (Huhns and Singh, 2005). The features of a service oriented architecture include:

- **Loose coupling and implementation neutrality:** These features of SOA enable communication with services much more easily than other traditional architectures. This provides abstraction and does not deal with the details of transaction between

various platforms and services.

- **Granularity:** The detailed information regarding the transactions is not considered. The interactions are dealt with in an abstract fashion and the actors involved in a transaction are treated at coarse granular level (Huhns and Singh, 2005).

- **Reusability and maintenance:** SOA enables reusability of services and maintains services separate from the implementation platforms (Sprott and Wilkes, 2004). Reusability can be achieved only if there is certain amount of trust between the parties who may use the service at any point of time. Once achieved, it will lead to reduction of cost incurred during development and maintenance (Meta Group, 2003).

4 The Development of the Designer's Knowledge Desktop

In Sections 2 and 3.1 we introduced the problems and challenges facing organisations as they try to encourage knowledge reuse across the engineering design process. In this section we identified a process by which knowledge can be captured and searched across the organisation. The developed approach is them implemented in a *Knowledge Desktop* that will permit both the capture and subsequent retrieval of knowledge. The implementation of the Desktop is discussed in Section 5.

4.1 Personas and scenarios

The initial step in developing the application was the identification of, and defining the underlying personas and scenarios which are a lightweight method of capturing and recording the requirements of the system from an end user's viewpoint (Cooper and Reimann, 2003). A persona describes an end users in some detail, their background, job function, situation in the organization. Scenarios are textual descriptions of how the persona interacts with the system and with other personas. The scenarios are independent of any technology and they may represent either current practice or improved practice. Subsequently the scenario illustrates the potential use and benefit from such a knowledge support system, and is constructed around the people identified by the personas.

The operation of the *Knowledge Desktop* is based on the concept of a *report*. A report contains a summary of an issue, allowing easier searching at a later date, and is based on the ontology discussed later in the paper. Hence if the same issue arises at a later date, a new report is not raised, but information is appended to an existing record. The front-line service engineers are involved in the day-to-day resolutions of issues; it is their information that is used to initially populate the knowledge repository. A further use of the report is as part of the design process, where a folder containing a number of reports relating to a specific component provides an ideal starting point for subsequent designs.

Within this case study we identified three main stakeholders; a Service Engineer (A), a Design Engineer (B), and a Knowledge Builder (C). Following discussions with the stakeholders, the following short personas were developed:

- **Service Engineer:** A is an engineer working in a civil aircraft engine service team and is based either at a major airport or overseas service facility. A's role is to respond quickly to issues on aircraft by containing the problem and preserving evidence needed for subsequent root cause analysis.
- **Design Engineer:** B is an engineer working on the component design for both existing and new engine design, B is based in a design team, and is responsible to the team leader.
- **Knowledge Builder:** C is an experienced design engineer, who oversees the design direction for a group of engineers, and has considerable experience in all aspects of engine design.

For each persona a related scenario was created. The aim of these scenarios was to capture in the language of the stakeholders, including the organizations engineers, the detailed interactions of each persona with the proposed system. This proved to be a key step in the design process of the *Knowledge Desktop*, as the stakeholders begin to see what is possible with a new system, hence their understanding and therefore requirements tend to change and become more focused, leading to an iterative process. The individual scenarios are as follows:

> *During the regular pre-flight checks, a flight crew reported a problem with an engine. Subsequent inspection by A revealed an issue with a specific component. After replacement, the engine was returned to service, and a maintenance report submitted. On receipt of the report at the company's service center, a search reveals that this is the first occurrence of such an event, hence a new record is raised, Figure 2. As the issue was investigated by a group of engineers including B, the knowledge generated was entered into a record, with links provided to technical reports and supporting documentation, i.e. the solid line in Figure 1.*

This approach results in the record becoming a summary of that issue, allowing rapid retrieval of the knowledge in the future. The knowledge repository can then be used by engineers responsible for improving the performance of existing engines, and for monitoring trends that develop over a fleet of engines:

> *Following the receipt of report of an maintenance issue by A, a query of the knowledge repository reveals an existing record — allowing a rapid resolution of the problem. As the history of this specific issue builds up over time, it is also possible to recommend earlier intervention to reducing the issue occurring, the required procedures developed by B.*

As is well recognized, the use of past experiences and previously acquired knowledge, either from the designer's own experiences or from resources within their organization is an important part of the design process for future engines, as follows;

> *As part of the design of a new engine variant, C consolidates information relating to a specific component, including the records, into a folder, this provides the designers, including B, with a list of possible issues that need to be resolved in the redesign effort. As part of the initial review, C goes through all the records, and rejects a number based on his knowledge of the new application, this process is represented by the dashed lines in Figure 1.*

It should be recognized that throughout the search activities, the members of the teams has access to a wide range of material, some of which are discussed in related papers including Lanfranchi et al (2007) and Dadzie et al (2008).

4.2 Use case

The use cases consisted of a narrative together with supporting UML diagrams, this approach was used as it was abstract and implementation independent. A brief narrative description is held alongside the diagram as a whole, as well as for each individual use case. This description helps to disambiguate the use cases, explains the roles of the different actors associated with that use case and focuses at a high level on what each use case involves. From an agile point of view they are also effective because they are relatively informal, yet help to define and capture a problem space in enough detail and can be understood by the whole team, including the end users and stakeholders. In practice the use cases iteration was undertaken during a round table discussion with the engineers, software developers and stakeholders involved. This approach allowed all parties to have a overview of the complete development process.

4.3 Storyboard

Using the scenarios, a number of storyboards can be created to represent the user interface design, for the *Knowledge Desktop*. This is a standard technique used in HCI development and is effective when used in a participatory (or co-design) process. This brings together the end users and the HCI experts, to designing the user interface. During this process, the scenarios and use cases can be clarified and modified if required. In practice, the storyboard consisted of a PowerPoint presentation accompanied by a textual description of the slide and the available actions. Figure 3 shows a typical storyboard slide. One point that should be noted is that the development of the scenarios, use cases and storyboard was undertaken in parallel and was completely transparent to everyone involved in the project.

4.4 Definitions of the required web services

To move from the use cases to Web service design was a further iterative process. There are two methods used, the first is to iterate through the use cases until they each represent one service. Alternatively a method proposed by Millard et al (2006) termed *The Service Responsibility and Interaction Design Method* (SRI-DM) could be used. SRI-DM separates abstract representations of services from their implementation. Service profiles are thus modeled in an abstract way that does not prescribe a data model or dictate explicit methods. To do this SRI-DM uses Service Responsibility and Collaboration cards (SRCs) based on Class Responsibilities/Collaborations, a modeling technique first described by (Beck, 2000) for eXtreme Programming (XP). The SRCs do not show how services will be combined in a wider scenario, but do model possible collaborations with other services that might occur for this service to fulfill its own specific responsibilities.

At the scenario level, services must interact with each other to fulfill a wider purpose. These interactions are complex and include transactions, sequences and state. If the scenario modeling is to maintain the high level of abstraction necessary for Agile development then it would be inappropriate to declare a detailed data model, or to specify the logic of the communicating services. In this project we used UML 2.0 Sequence Diagrams to represent the interactions. This approach defines which services should communicate

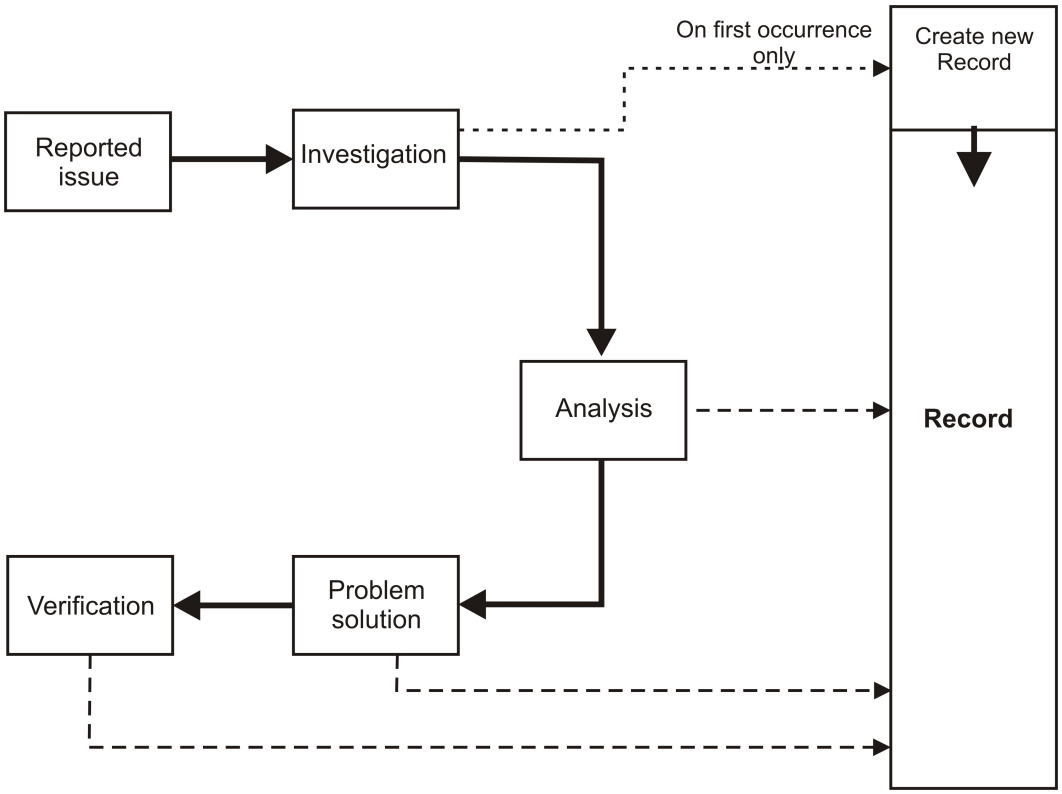
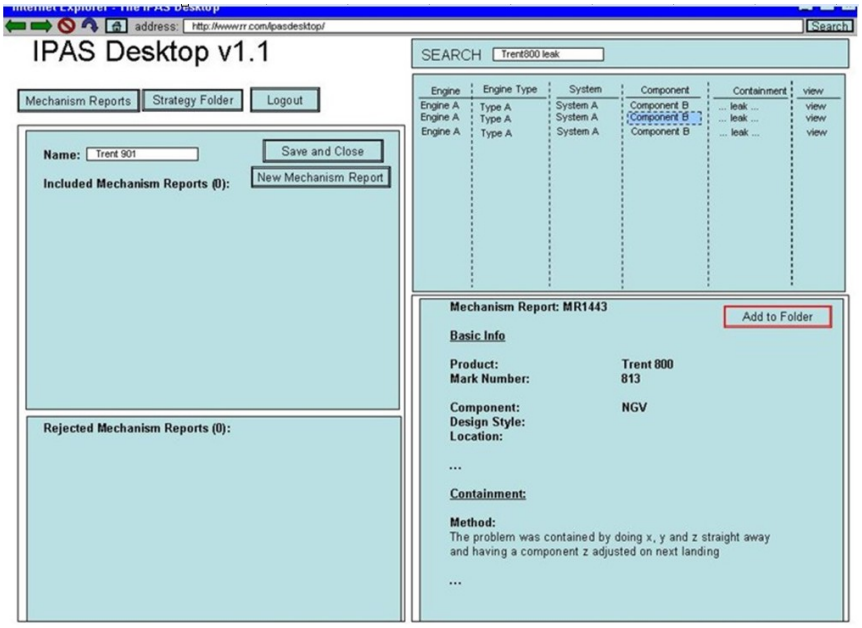


Fig. 2 A record is generated for a new issue. The solid line represents the steps taken for finding an acceptable solution, which are summarized in the record, the dotted lines the information being stored in the record.



Adding a new folder. The new name is entered on the left. Entering a term in the right hand search box allows you search for a record. Following selection, a short overview of the report is presented. Selecting **Add to Folder** will place this record into the folder.

Fig. 3 A typical storyboard slide for the *Knowledge Desktop* application. This particular slide discusses aspects of forming a folder, a task that would be undertaken by the Knowledge Builder.

and in which order, and containing enough description to show how the individual services are responsible for moving and processing data, specifying the detail of the data model or the decision making logic. The Web services provide processing functionalities for the system. The Web service interfaces are defined in WSDL (Christensen et al, 2001). WSDL is the standard interface language for defining Web service interfaces. This document is in XML format, and lists the operations a Web service supports. The document also defines the syntactic types of inputs and outputs for each operation. When specifying the service to be built it is not enough to give the programmers just the WSDL, a textual description of the operation of the service is also required. We have developed several Web services that perform common requests on the maintenance documents: for example, obtaining a list of parts that are involved in the highest number of unscheduled maintenance events, tracing the maintenance record of an engine or a part, or retrieving details of a maintenance event.

4.5 Resources and ontologies

Identifying and locating the multimedia resources is a significant component of the hypermedia authoring process (Lowe and Hall, 1998). The structure and use of the resource also inform the development of the ontologies. Both the identification of resources and ontology building need the knowledge of end users and stakeholders. During this process the scenarios can be embellished with details of how the resource may be used and information they supply. An aerospace engineering ontology defined in OWL is used to describe the objects and concepts that appear within this information set. OWL (McGuinness and v. Harmelen, 2004) is the standard Web ontology language for use between Web agents in the Semantic Web. The development of this ontology was greatly helped by the closely controlled language used across the aerospace industry. The ontology is created by analyzing existing documents and conducting knowledge acquisition interviews with engineers (Wills et al, 2003). The result of these interviews enabled us to identify, by specialism, the main concepts and the associated keywords for these concepts used by the particular type of engineer when searching for information.

The underlying information set is expressed as RDF triples (Manola and Miller, 2004). RDF is the language for representing information about resources in the Semantic Web. RDF is used to represent both the data itself, and its associated metadata. Documents in the repository will be in the form of RDF triples. Both the RDF triples and associated OWL ontologies are stored in Sesame[1], an open source RDF framework with support for RDF Schema inferencing and querying.

5 Architecture and Implementation

The intelligent document repository including the *Knowledge Desktop* was organized as a SOA, as shown in Figure 4. The user forms a query via a user interface. In the current system design, this user interface is in the form of Web pages provided by a Web portal. To answer the user's query, the Web portal finds and calls the Web services that can provide it with relevant knowledge. These Web services provide algorithms and functionalities that work on the underlying information set. There are no restrictions on the type of functionalities that the Web services can provide, and it can range from an index of available documents, to finding the frequency of maintenance events.

Existing Web standards are used wherever possible, to maximize tool reuse, compatibility and portability. A Web browser accesses the Web portal using the HTTP protocol. In turn, the Web portal supplies the user interface to the Web browser. The user interface is formed by a series of portlets. Portlets are reusable components that display relevant information to portal users. In general, they appear to end users as rectangular sections of a Web page offering a limited set of information. In our implementation, the portal conforms to the Java Portlet API standard, JSR 168[2]. To create the information for display, the portlets access one or more Web services. We envisaged that these Web services are provided by different departments within the company, and can be distributed across multiple sites. These Web services will perform more than document indexing that is already available in all document repositories. For example, it can provide an interface to existing manufacturing systems such as PDM (Product Data Management), or allow access to analysis tools such as Life-Cycle Cost Models and Finite Element Analysis.

It should be noted that the architecture does not require all Web services to access the underlying information sources via the triplestore. However, most, if not all, Web services will operate on the semantically enabled data within the triplestore instead of the original heterogeneous documents. The Web services construct SPARQL queries according to their inputs (Prud'hommeaux and Seaborne, 2006). They then obtain results from a triplestore using the constructed

[1] http://www.openrdf.org/

[2] http://developers.sun.com/portalserver/reference/techart/jsr168/

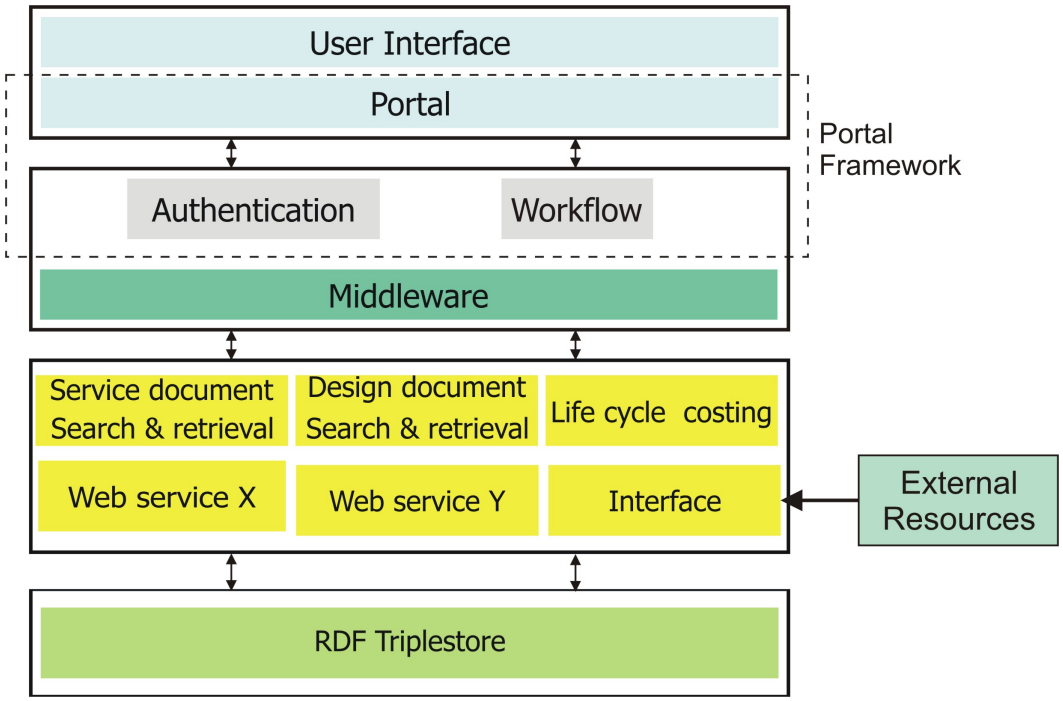


Fig. 4 A conceptual view of the *Knowledge Desktop*'s architecture, showing three typical Web services, two for search and retrieval, and the third being the interfacing to a external life cycle cost modeler

query. Thus, the Web services allow SPARQL queries to be reused across applications. Furthermore, they provide an abstraction over the data stored in the triplestore so developers who are unfamiliar with the ontology will still be able to perform queries. In our implementation the records were stored in the form of RDF triples, with both the RDF triples and associated OWL ontologies being stored in a Sesame 1.x triplestore.

6 Evaluation

We developed three incremental versions of the Knowledge Desktop over a period of a year. Each development cycle was approximately 90 days in length, followed by evaluation and development of the requirement specification for the next version of the application. A screenshot from the final version, Figure 5, can be compared with the original user interface design, Figure 3. While all three versions have considerable similarities, a number of changes resulted from the feedback process, both from the users and the developers. As the majority of technical issues have been resolved during the development of the scenarios or at the storyboard stage, most changes identified at the evaluation stages were either cosmetic (font size, accessibility issues) or related to company policy, for example naming conventions were not totally uniform between designers, manufacturing or service teams. The problems associated with this issue were in part addressed by the incorporation of sophisticated navigational tools using the underlying ontologies. All the contributors to the development process commented on the accessibility of the development approach being employed, and its ease of use, particularly allowing understanding by both the engine design community and the software developers.

While all versions of the application were evaluated, in this section we will focus on the final evaluation as this reflected the final deliverable with all the key features both for the designer and knowledge builder. The purpose of the evaluation was to consider the *Knowledge Desktop* and its ease of use, rather than the quality and quantity of the information stored. Due to the time and resource constraints present within an industrial organization the formal part of the evaluation was limited to six subjects, all of whom were design engineers (persona *B* in Section 4.1). To ensure that the results were not component dependent, three of the engineers had prior knowledge of the chosen part and the other three were drawn from across the company and had identical responsibilities but for different components.

Prior to the evaluation the *Knowledge Desktop* was populated by an experienced designer with knowledge on a number of related components; over 70% relating to the component that was used in the evaluation. Each subject was seen individually during the evaluation, and following an introduction to both the project and the

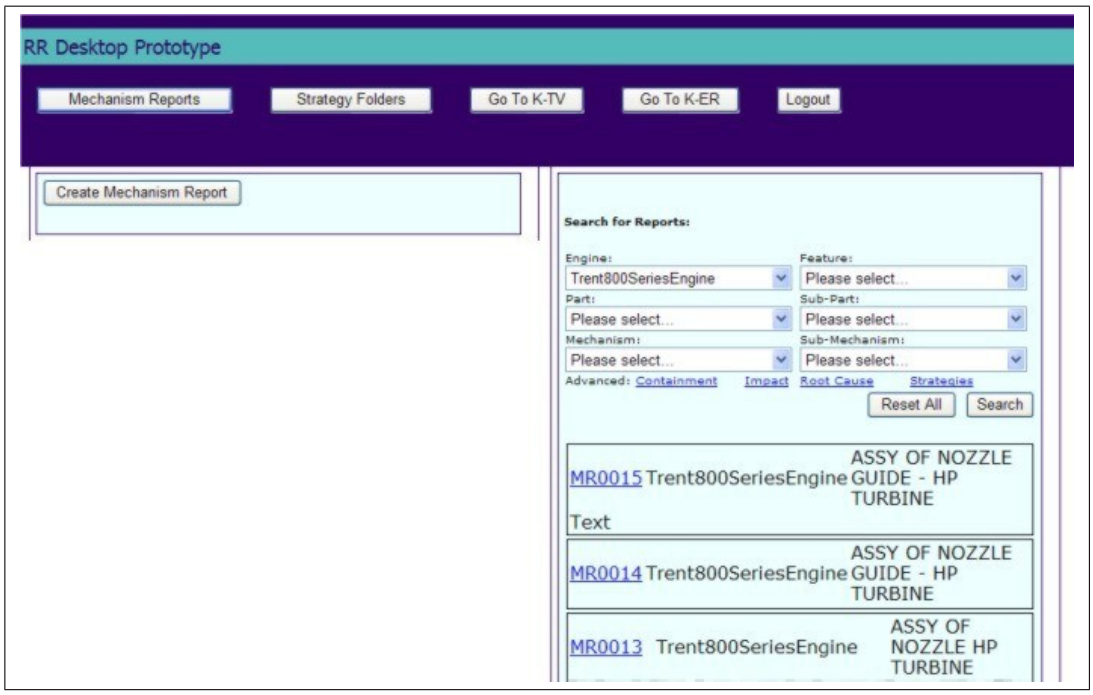


Fig. 5 A typical screenshot from the final application, showing the addition of search terms, and report summaries, that were requested by our customers as part of the development process.

application was asked to undertake a short task, which allowed then to address the following query:

> *What are the top three issues for component X when life–cycle costs are considered?*

To undertake the required task each participant had to search the repository for records related to the specific component using the *Knowledge Desktop*, analyze the information stored in the retrieved records, and then rank them in order of importance. This task was selected as it was considered representative of the activities that designers will be expected to undertake when using the repository. The user was then asked to select the top three records from the subset and to summarize the rationale for their choice. At the completion of this task, we asked the subject to complete a questionnaire about their experience. The questionnaire included questions about the structure of the records and the user's experience of the *Knowledge Desktop*. The results are summarized in Figure 6, which gives the responses from all six subjects on selected items from the questionnaire. From the responses to the first three items we see that the subjects found the system easy to use, and had a favorable overall impression. The other four items in the table indicate how useful the subjects found the information retrieved in the records and how understandable the records were, including consideration of the terminology used; as can be seen, for these four items, the response by the group was very positive.

During the evaluation and subsequent discussions we noted that all six participants found the system very intuitive and user friendly. Many suggested that they would have been able to use the system without any formal training. Although the subjects found the search mechanism very intuitive, many commented that it might be easier to start at an engine module level and then refine the search to specific part(s). However we did note that the designers who had a high level of familiarity with the component, as would be the case when the *Knowledge Desktop* is used in practice, did not need this feature.

During the study we observed that the understanding of the data in the system was highly dependent on the participants' familiarity with component *X*. Participants who were not familiar with the component tended to look primarily at numerical data, while participants who were familiar with the component used all the information available. Despite this problem, the participants found the structure of the records very useful to access service and design information in a single system. As expected, some participants mentioned that accessing external data sources would sometimes provide a better understanding of the problem. Additionally, the subjects suggested it would be useful to have some statistical data to support early decisions and the impact on related processes, these approaches were implemented in earlier versions of the system as discussed

in Wong et al (2006) and Wong et al (2007), and hence need to be reflected in future work.

7 Discussion and Conclusions

In looking at future systems to support designers, there is often a presumption that all processes should be based on IT systems; in our discussions with the designer engineers this was strongly resisted. They were only willing to move to automated systems if there was a clear advantage to themselves in doing so. In the case of the *Knowledge Desktop*, we believe that a system based on ontological hypertext did demonstrate such an advantage, for the following reasons. In many cases, initially the designer may not fully understand exactly what is required and therefore may not know what type of expertise or information is required to resolve the problem. In order to maintain the accuracy and reliability as the underlying information resources change, coupled with the need for engineers to use semantically related terms to locate information, made a strong case to use ontological hypertext in guiding the engineer towards the information they need. A key issue for the user is that the system has to be accurate and reliable. If this is not achieved, people will not trust the system, and hence it will not be used. This paper has presented the *Knowledge Desktop* that will allow the collection of engineering knowledge of a specific component to be collated and analyzed so that maximum use can be made, both on current and future aircraft engines.

With a human centered approach that we employed in the development cycle, the question and problem can be discussed and interpreted by the user, making it more likely to result in an approach that is beneficial. In our approach the inclusion of the stakeholders had a positive impact on the development process; firstly it ensured that possible problems were identified at the earliest opportunity, and secondly the stakeholders had a degree of ownership with the developed desktop. The approach we have taken has proved to be robust and easily understood by all members of the user community, this has ensured that the design is fit for purpose and will support the designer's day–to–day activities.

As we have discussed there is a move away from the tightly coupled systems into a more Service Oriented Architecture approach. The SOA allows the easy reuse of elements of the software, hence further reducing the length of the development cycle. It is recognized that a large number of models and methodologies can be applied to the design of the hypertext systems; however the area of requirement analysis is perhaps the least investigated. The use of the agile approach in this application has allowed us to use a lightweight method of capturing and recording the requirements for the ontological hypermedia application. This approach is recommended for similar applications as it ensures that the user community is involved with all aspects of the development cycle, and when the development is incremental following interim evaluations, that result in rapid design changes.

In the evaluation of the final version of the *Knowledge Desktop*, it was clear that the developed technology closely matched the requirements of the user community; we believe this was due to the approach of earlier and continuous customer involvement. In our discussion with all the designers evaluating the Desktop, it was clear that if this system was deployed, it would have a significantly beneficial effect on their day-to-day activities. It should be noted that the *Knowledge Desktop* is just one a range of tools that the designer will require in the future, some of which are discussed in Fowler and Sleeman (2004); Crowder et al (2003).

As discussed in Section 2 a fundamental shift is occurring in the aircraft engine suppliers, by moving away from selling products to providing services. This implies that new engines must be designed to provide lower and more predictable life cycle costs, and in turn, this requires reliable access to the organization's information repositories. To achieve this, engineers must obtain knowledge, gained from the entire life of an aircraft engine, which will be held in distributed and heterogeneous data sources. Integration is made possible with the use of the ontological hypertext, which allows software agents to reason over the different resources.

8 Acknowledgments

This research was undertaken as part of the *Integrated Products and Services* project funded by the UK Technology Strategy Board under grant number (`DTI Grant TP/2/IC/6/I/10292`).

The authors fully acknowledge the considerable support received from Rolls-Royce plc in undertaking this project, in particular Dave Knott and Colin Cadas. The authors would also like to thank the project partners for providing us with data and ontologies. Specifically, we would like to thank Aylmer Johnson and Santosh Jagtap from Cambridge EDC for the user requirement analysis.

References

Abbas N, Gravell A, Wills G (2008) Historical roots of agile methods: Where did "agile thinking" come from? In: Agile processes and eXtreme programming in Software Engineering

Ahmed S, Wallace K (2004) Identifying and supporting the knowledge needs of novice designers within the aerospace industry. Journal of Engineering Design 15(5):475–492

Alonso-Rasgado T, Thompson G (2007) A rapid design process for total care product creation. Journal of Engineering Design 18(1):509–585

Beck K (2000) Extreme Programming Explained: Embrace Change. Addison-Wesley

Busby J (1998) Effective practices in design transfer. Research in Engineering Design 10(3):178–188

Carr L, Hall W, Bechhofer S, Goble C (2001a) Conceptual linking: Ontology-based open hypermedia. In: Proceedings of 10th International World Wide Web Conference (WWW), Hong Kong, pp 334–342

Carr L, Kampa S, Miles-Board T (2001b) Metaportal final report: Building ontological hypermedia with the ontoportal framework. Tech. rep., IAM, ECS, University of Southampton

Carr L, Miles-Board T, Wills G, Grange S (2004a) Extending the role of digital library: Computer support for creating articles. In: Proceedings of The 15th ACM Conference on Hypertext and Hypermedia, Santa Cruz, USA, pp 12–21

Carr L, Miles-Board T, Woukeu A, Wills G, Hall W (2004b) The case for explicit knowledge in documents. In: Proceedings of ACM Symposium on Document Engineering, Milwaukee, Wisconsin, pp 90–98

Christensen E, Curbera F, Meredith G, Weerawarana S (2001) Web Services Description Language (WSDL) 1.1. W3C Note, `http://www.w3.org/TR/wsdl`

Ciravegna F, Wilks Y (2003) Designing adaptive information extraction for the semantic web in Amilcare. In: Handschuh S, Staab S (eds) Annotation for the Semantic Web, IOP

Cohen A, Maglio P, Barett R (1998) The expertise browser: How to leverage distributed organizational knowledge. In: Workshop on Collaborative Information Seeking at CSCW '98, Seattle

Cooper A, Reimann R (2003) About Face 2.0: The Essentials of Interaction Design. John Wiley & Sons.

Crowder R, Hall W, Heath I, Wills G (1998) Factory information provision using hypermedia. International Journal of Computer Applications in Technology 11:442–453

Crowder R, Hall W, Heath I, Wills G (2000) Industrial strength hypermedia: Design, Implementation and Application. International Journal of Computer Integrated Manufacturing 13(3):173–186

Crowder R, Hughes G, Hall W (2002) An agent based approach to finding expertise. In: Karagiannis D, Reimer U (eds) Proceedings Fourth International Conference on Practical Aspects of Knowledge Management, vol LNAI 2569, Springer-Verlang, Berlin Heidelberg, pp 179–88

Crowder R, Bracewell R, Hughes G, Kerr M, Knott D, Moss M, Clegg C, Hall W, Wallace K, Waterson P (2003) A future vision for the engineering design environment: A future sociotechnical scenario. In: Folkeson A, Gralen K, Norell M, Sellgren U (eds) Proceedings of 14th International Conference on Engineering Design, Stockholm, pp 249–250

Dadzie A, Bhagdev R, Chakravarthy A, Chapman S, Iria J, Lanfranchi V, Magalhes J, Petrelli D, Ciravegna F (2008) Applying semantic web technologies to knowledge sharing in aerospace engineering. Journal of Intelligent Manufacturing 20(5):611–623, DOI 10.1007/s10845-008-0141-1

De Bra P, Aroyo L, Chepegin V (2004) The next big thing: Adaptive web-based systems. Journal of Digital Information, article No. 247, 2004-05-27

El-Tayeh A, Gil N, Freeman J (2008) A methodology to evaluate the usability of digital socialization in virtual engineering design. Research in Engineering Design 19(1):29–45

Erickson T, Kellogg WA (2000) Social translucence: an approach to designing systems that support social processes. ACM Trans Comput-Hum Interact 7(1):59–83

Fowler D, Sleeman D (2004) The designers' workbench: Using ontologies and constraints for configuration. In: The Twenty-fourth SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence, Queens' College, Cambridge, UK.

Guan T, Fowler D, Bandara A, Zaluska E, De Roure D, Crowder R, Wills G (2009) Enhancing grid service discovery with a semantic wiki and the concept matching approach. In: Proceedings of the 5th International Conference on Semantic, Knowledge and Grid

Harrison A (2006) Design for service – harmonising product design with a services strategy. In: Proceedings of GT2006, ASME Turbo Expo 2006: Power for Land, Sea and Air, Barcelona, Spain

Heath I, Wills G, Crowder R, Hall W, Ballantyne J (2000) Towards a new authoring methodology for large–scale hypermedia applications. Multimedia Tools and Applications 12(2-3):129–144

Highsmith J, Beck K, Cockburn A, Jeffries R (2001) Agile manifesto. `www.agilemanifesto.org`

Huhns M, Singh M (2005) Service oriented computing: Key concepts and principles. IEEE Internet Computing 9(1):75–81

Jackson T, Austin J, Jessop M, Linag B, Pasley A, Ong M, Allan G, Kadirkamanathan V, Thompson H, Fleming P (2005) Distributed health monitoring for aero-engines on the grid: Dame. In: Proceedings of IEEE Aerospace, Montana

Khadilkar DV, Stauffer LA (1996) An experimental evaluation of design information reuse during conceptual design. Journal of Engineering Design 7(4):331–339

Lanfranchi V, Bhagdev R, Chapman S, Ciravegna F, Petrelli D (2007) Extracting and searching knowledge for the aerospace industry. In: Proceedings of 1st European Semantic Technology Conference, Vienna , Austria

Lowe D, Hall W (1998) Hypermedia Engineering: the Web and Beyond. Wiley

Malcolm K, Poltrock S, Schuler D (1991) Industrial strength hypermedia: Requirements for a large engineering enterprise. In: Proceedings Hypertext '91, Seattle, pp 13–24

Maneewatthana T, Wills G, Hall W (2005) Adaptive personal information environment based on the semantic web. In: Proceedings of Hypertext 2005 - Sixteenth ACM Conference on Hypertext and Hypermedia, Salzburg, Austria

Manola F, Miller E (2004) RDF Primer. W3C Recommendation, `http://www.w3.org/TR/rdf-primer`

McGuinness D, v Harmelen F (2004) OWL Web Ontology Language overview. W3C Recommendation, `http://www.w3.org/TR/owl-features`

McMahon C, Lowe A, Culley S (2004) Knowledge management in engineering design: personalization and codification. Journal of Engineering Design 15(4):307–325

Meta Group (2003) Practical approaches to services-oriented architecture. Meta Group White Paper

Millard D, Howard Y, Jam E, Chennupati S, Davis H, Gilbert L, Wills G (2006) FREMA method for describing web services in a service oriented architecture. Tech. Rep. ECSTR-IAM06-002, ECS, University of Southampton.

Nerur S, Mahapatra R, Mangalaraj G (2005) Challenges of migrating to agile methodologies. Communications of the ACM 48(5):72–78

Nnadi N, Bieber M (2004) Lightweight integration of documents and services. In: DocEng '04: Proceedings of the 2004 ACM symposium on Document engineering, ACM Press, New York, NY, USA, pp 51–53, DOI http://doi.acm.org/10.1145/1030397.1030408

Poli C, Dastidar P, Graves R (1992) Design knowledge acquisition for DFM methodologies. Research in Engineering Design 4(3):131–145

Priebe T, Pernul G (2003) Towards integrative enterprise knowledge portals. In: CIKM '03: Proceedings of the 12th international conference on Information and knowledge management, ACM Press, New York, NY, USA, pp 216–223, DOI http://doi.acm.org/10.1145/956863.956906

Prud'hommeaux E, Seaborne A (2006) SPARQL query language for RDF. Tech. rep., W3C Working Draft, `http://www.w3.org/TR/rdf-sparql-query`

Qiu R, (2009) Towards ontology-driven knowledge synthesis for heterogeneous information systems. Journal of Intelligent Manufacturing 17(2) 99–106

Reed N, Scanlan J, Wills G, Halliday S (2009) Providing value to a business using a lightweight design system to support knowledge reuse by designers. In: Proceedings of 17th International Conference on Engineering Design, Stanford, CA, USA, pp 135–145.

Sim YW (2004) Capturing organisational knowledge from documentation for expert finding. PhD thesis, School of Electronics and Computer Science, University of Southampton

Sprott D, Wilkes L (2004) Understanding service oriented architecture. `http://msdn.microsoft.com/architecture/soa/default.aspx?pull=/library/en-us/dnmaj/html/aj1soa.asp`

Wallace KM, Clegg C, Keane A (2001) Visions for engineering design: a multi-disciplinary perspective. In: Proceedings of 13th International Conference on Engineering Design, Glasgow, Scotland, pp 107–114

Wills G, Woukeu A, Carr L, Hall W (2003) The need for semantic in web design. In: Semantic Web workshop, Fourteenth Conference on Hypertext and Hypermedia

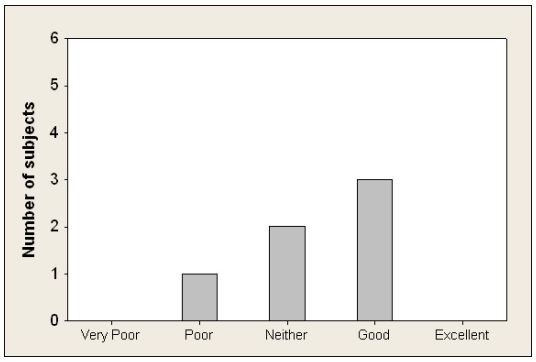
Wills G, Fowler D, Sleeman D, Crowder R, Kampa S, Carr L, Knott D (2004) Issues in moving to a semantic web for a large corporation. In: Proceedings of 5th International Conference on Practical Aspects of Knowledge Management (PAKM), Springer, Lecture Notes in Artificial Intelligence, vol 3336, pp 378–388

Wilson S, Blinco K, Rehak D (2004) Service-oriented frameworks: Modelling the infrastructure for the next generation of e-learning systems. In: JISC, Bristol

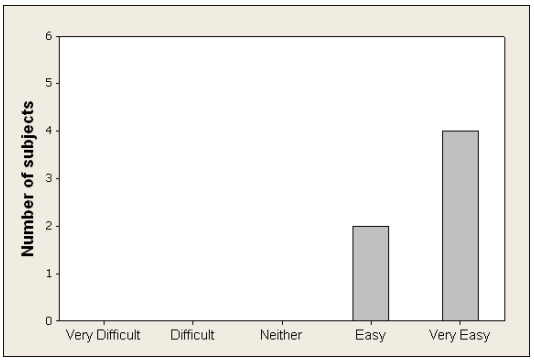
Wong S, Crowder R, Wills G, Shadbolt N (2006) Knowledge engineering - from front-line support to preliminary design. In: ACM Symposium on Document Engineering (DocEng)

Wong S, Crowder R, Wills G, Shadbolt N (2007) Lesson learnt from a large-scale industrial semantic web application. In: 18th ACM Conference on Hypertext and Hypermedia

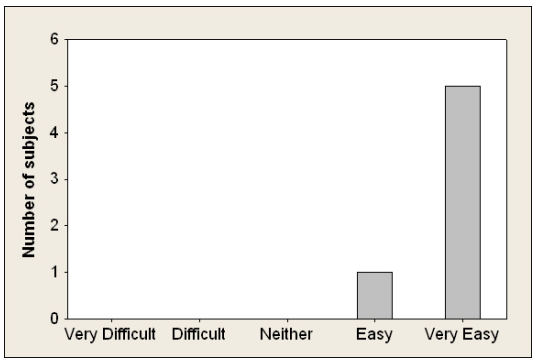
Xu X, Wang Z (2009) State of the art: business service and its impact on manufacturing. Journal of Intelligent Manufacturing DOI 10.1007/s10845-009-0325-3



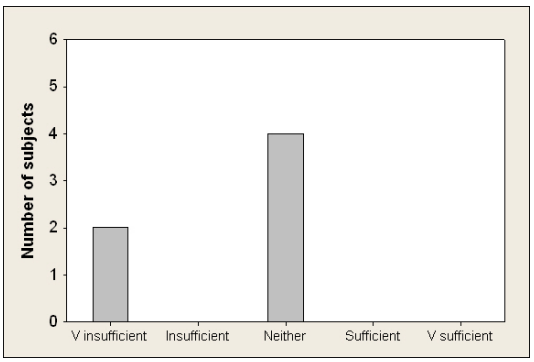
(a) Overall impression



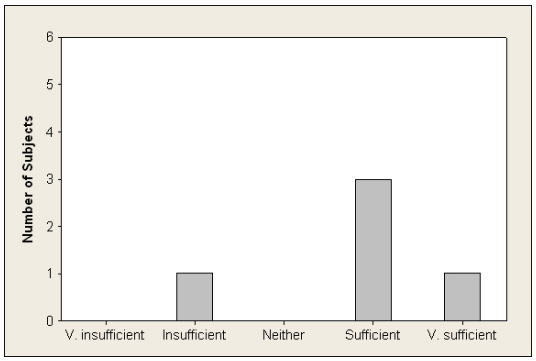
(b) Overall impression (ease of use)



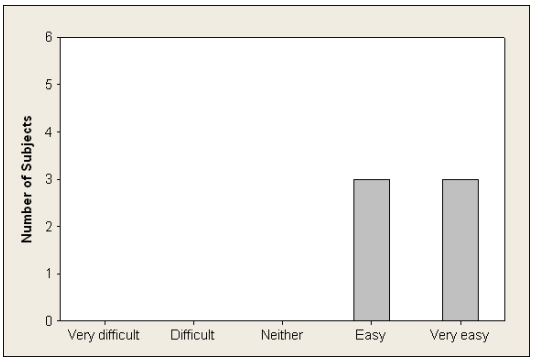
(c) Learning how to operate the system was easy



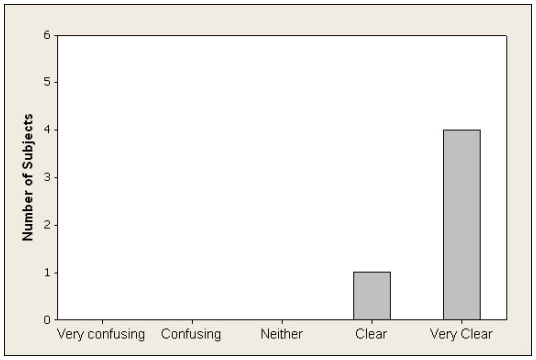
(d) My knowledge of the HPT NGV is sufficient?



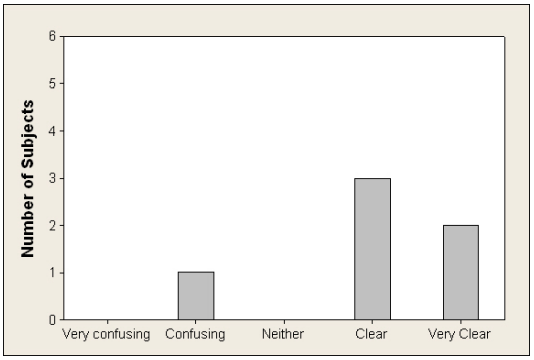
(e) The information provided in the mechanism records is sufficient



(f) Retrieving the mechanism records was easy?



(g) The names used for the fields in mechanism records are clear?



(h) The information presented was clear?

Fig. 6 Results from the evaluation questionnaire