Enabling through life product-instance management: Solutions and challenges

Damith C. Ranasinghe, Mark Harrison, Kary Främling, Duncan McFarlane

1. Introduction

The life of a product does not begin at a manufacturing plant and terminate at a retailer. Just as the human cycle of life, a product’s life evolves from its inception as an idea and transforms its way into design, testing, manufacture, usage and ultimately its retirement and eventual disposal. Ameri and Dutta (2005) and Stark (2004) highlighted the development of the concept of product lifecycle management (PLM, CIMdata, 2007) as knowledge management by capture, storage, processing and usage of product related knowledge throughout the life of a product. The rationale for PLM strategies are many (Stark, 2004), such as improved detection of deficiencies or failures in the design or manufacturing process (which may require comparison and correlation of data from multiple parts or products of the same type), as well as the ability to make better end-of-life decisions (Parlikad and McFarlane, 2004) to extract maximum residual value from the products and to ensure safe disposal of the components that cannot be re-used or re-manufactured. Even two objects of the same type, which are manufactured at the same time and location, may be subjected to completely different usage patterns during their lifetime. Other reasons for collecting such data include better management of warranties and tracking of liability as well as maximising utility and even detecting counterfeit products (Ranasinghe et al., 2007).

This product-instance oriented definition of PLM is significantly different from definitions used in other contexts. PLM as used by CAD/CAM and even ERP system vendors usually signifies the management of product design documents and the different variations and versions of those documents, while the “usage” phase of product instances tends to be omitted. In the commercial context, a product’s lifecycle is the time span from the market launch of a product type, model or family until it is no longer being produced and sold. These different uses of the concepts product lifecycle and PLM tend to be a source of confusion; in this paper the product lifecycle is the lifecycle of a product instance from when it is conceived and produced until it is destroyed or recycled, while PLM signifies the processes and systems related to managing that lifecycle.

PLM strategies are particularly important for products or parts that have a high monetary value, long service life, as well as objects or parts that are frequently inspected and repaired or...
maintained (especially where safety depends on their integrity and fitness for use) and for objects which pass between multiple organisations throughout their service life such as leased fleets of mining equipment or service and maintenance in the aerospace industry.

While the concept of PLM is seen as a strategic approach to through-life management of products as well as the associated business processes, more recent environmentally friendly thinking has led to legislation such as WEEE (Waste of Electrical and Electronic Equipment) Directive 2002/96/EC, RoHS (Restriction of Hazardous Substances) Directive 2002/95/EC (http://ec.europa.eu/environment/waste/weee_index.htm) and ELV (End of Life Vehicle) Directive 2000/53/EC established by the European Parliament. The intention of these directives is to place the burden of responsibility on manufacturers to minimize the environmental impact of their products throughout a product’s lifecycle with the intentions of reducing landfill and increasing recycling and has delivered the added stimulus to generate adequate momentum towards adopting PLM concepts.

The primary constituents of a PLM strategy are capture of product data, processing and transformation of that data into knowledge and finally being able to share the accumulated body of product knowledge with various custodians of the product as well as all the parties that interact with that product throughout its life. Such a strategy requires a holistic approach that encompasses the life of a product in its entirety to develop product lifecycle information management (PLIM) systems (Fränling et al., 2007a). In Kirtisis et al. (2003), the authors discussed the key research challenges in developing PLIM systems. They identified the need to develop a comprehensive platform capable of closing the information gaps that exist through the life of a product from manufacture to its eventual end as landfill or recyclable waste as the key component that must precede the implementation of any effective and comprehensive PLM strategy. Since then we have seen a growing research interest to address the information gap through attempts to gather, process and share object related information across disparate and globally distributed systems spread across various geographical boundaries. Growth in this research area gathered particular momentum with the maturity and the sharp reduction in the cost curve of technologies such as radio frequency identification (RFID). Further impetus is provided by the continued miniaturisation as well as the development of low power sensor technologies based on micro-electro-mechanical systems (MEMS) technology. The strength of interest in this field is also demonstrated by research projects such as PROMISE (PROMISE, 2009a), BRIDGE (BRIDGE, 2009a) and commercial activities (BOEING, 2007, CATERPILLAR, 2009) aimed at using automatic identification and data capture technologies for collecting product related data and developing systems to manage and use data associated with products throughout their lifecycle. Further examples of the use of embedded technologies for managing product lifecycle data are reported in Cao et al. (2007), Jun et al. (2007).

The key element of many of these activities is the use of the Internet to link individual product instances to their digital representations distributed across various parties associated with the product throughout its lifecycle in what is called an object centric approach to PLM (Kärkkäinen et al., 2003; Bajic and Chaxel, 2002; Chaxel et al., 1999; Cao et al., 2007; Stuit and Meyer, 2009). Here, the problem of locating and assimilating information dispersed across a growing number of actors participating in the life of a product is solved by providing each object with a unique identifier (Ashton 2000) so that this can be used to retrieve the necessary information via the Internet.

In this paper we will consider three alternative approaches employing the object centric paradigm to managing product lifecycle information at the product instance level and analyse their ability as well as their suitability to meet the technical requirements for supporting various PLM strategies. While PLM strategies are well documented in literature it is only with recent advances in automatic identification technologies such as RFID and the miniaturisation of sensors that such strategies can finally be implemented. The process of mapping requirements to implement PLM strategies to the information systems that will ultimately support those strategies based on technologies such as RFID and wireless sensors has not been undertaken before to the best of our knowledge. Furthermore our work seeks to progress research aiming to address the divide between achieving the benefits of PLM strategies and the infrastructure required to achieve those strategies.

Our approach is based on studying three practical PLM application scenarios to identify key technical requirements for a PLIM system. The application cases considered were developed through the collaborative efforts of end-users, technology providers as well as researchers to demonstrate the business value of PLM strategies in the European project PROMISE. The application scenarios form a foundation for capturing the aspirations of all stakeholders in the lifecycle of a product.

The article is organised into four specific sections: introduction to the product lifecycle and an overview of the alternative approaches to PLIM; case studies and the development of information system requirements for PLIM; comparison of the alternative approaches and in the final section we present our conclusions.

2. Background and current approaches

2.1. Background

We can characterise the life of a product into three distinct phases: beginning, middle, and end, where the beginning of life (BOL) refers to the time of activities that take place before an asset begins its useable life, while middle of life (MOL) refers to the time of activities such as service and maintenance while the asset is in use, and finally end of life (EOL) refers to the time of activities involving recycling and disposal of the asset after it has finished its use phase of the lifecycle as shown in Fig. 1.

In Fig. 1 the EOL and BOL phases are connected in a loop as products are remanufactured, reused or recycled. However, they may be disposed and become part of landfill but eventually, and even from a philosophical point of view, they will begin another life as raw material for a new product.

It is useful to look at the product lifecycle in three separate but related stages since user requirements for business models are different in each product lifecycle stage. Fig. 2 illustrates the different phases that a product undergoes as well as the actors involved from the time it first emerges as a collection of ideas or concepts to design, manufacture, movement through the supply chain and then to users and finally they are repossessed by recycling companies, dismantlers or simply discarded as landfill.
2.1.1. Beginning of life
A product first begins its life at the design stage in the beginning of life (BOL) phase where the product is a collection of ideas or user requirements that are then transformed into design documents. In the manufacturing phase, the product may be in the form of raw material or a set of parts and subassemblies in their own BOL phase, possibly produced by different manufacturers. For instance the components in a FIAT engine consists of components designed and manufactured by various other SMEs (small and medium enterprises).

More formally, the BOL stage involves the generation of the product concept, and the manufacture of its physical model. Lifecycle management in the BOL phase are related to product design and production systems. The main PLM strategies are based on achieving the following:

- Producers should be able to make product design improvements using data related to the object obtained from other stages of a product's lifecycle.
- Ability to use product lifecycle information to generate new products.
- Ability to adapt production systems to improve product design, performance and efficiency based on product performance information obtained during other stages of a product’s lifecycle.
- Provide improved visibility of products in the supply chain with advanced tracking and tracing information, for example, to improve warehouse management.

2.1.2. Middle of life
As products leave retailers or dealerships they enter into their middle of life stage. In the MOL phase a product may undergo various business, maintenance, repair and work processes by different end-users of the product. PLM strategies in MOL are centred on enhancement of maintenance/service using additional or improved quality of information to achieve the following:

- Develop predictive maintenance (Lee et al., 2004; Anke and Främling, 2005; Binstrup et al., 2008) capabilities to reduce often unnecessary unscheduled maintenance and to remedy possible failures before they occur. As a result, reduce disruptions to business operations.
- Support maintenance/service operation by providing instant access to maintenance histories of components, or outlining correct maintenance procedures specific to components or providing the ability to better utilise limited maintenance resources such as people or tools.

2.1.3. End of life
Then, as a tangible result of all the previous phases of its usage history, the product reaches its end of life (EOL). Activities in this phase are concerned with the processing of products after they are discarded or their usable life has ended. It might involve recycling, improvements and re-sale, or disposal. There is an increasing number of motivating drivers for EOL management such as environmental regulations, cost recovery, and “green image” marketing (Fleischmann et al., 1997).

The management of products at the EOL stage mainly require implementing decision support processes to determine the outcome of the items to be retired from service, such as the following:

- The assessment of a product for its ability to be recycled, reused and re-manufactured.
- Support EOL decisions by suggesting a whether a part should be, re-used, recycled, remanufactured or disposed.

2.2. Existing approaches

Listed below are three currently known industrial and academic information management systems architectures based on an object centric paradigm for managing product information at the individual product instance level:

(1) EPC network architecture with its standard interfaces for collecting and accessing product related data
(2) The approach taken by the DIALOG information system using its ID@URI approach and further developed within the PROMISE project.
(3) World Wide Article Information (WWAI) approach using a peer-to-peer (P2P) lookup method to access and store data in backend systems.

The following sections will provide an overview of these architectures. Then in Section 4 we will analyse each individual approach and assess the degree to which they are capable of meeting the various PLIM system requirements.

2.2.1. EPCglobal network architecture (EPC network)

EPCglobal (EPCglobal, 2009a) is an organization focusing on industry-driven standards for the Electronic Product Code (EPC) Network to support RFID and other Auto-ID Labs technologies in fast moving, information rich and distributed trading networks to achieve interoperability and to allow hardware and software vendors to be able to compete in a fair and open market in the supply of technology and equipment to establish the EPC network as described in the EPCglobal Architecture Framework document. The architecture is a framework consisting of a collection of standards for hardware, software and data exchange, together with core services (provided by EPCglobal). This framework layers the architecture and splits the functionality into isolated modules. All ratified EPCglobal standards are freely available in electronic format (http://www.epcglobalinc.org/standards/).

A simplified view of the EPC network is shown in Fig. 3 where the arrows indicate the flow of data from tags to the network support systems and the flow of commands and data back to the readers and tags (EPCglobal, 2009b). While the architecture is explained and illustrated in greater detail in EPCglobal (2009b), Ranasinghe et al. (2007). We provide a brief overview here. In Fig. 3, readers collect data from tagged objects. The RFID tagged objects communicate a unique ID (called an EPC—Electronic Product Code) code to a reader and thus identify themselves as a unique entity. The data originating from the network of readers is passed to backend systems that control and collect data while...
providing service layer functionalities. For example the filtering middleware provides filtered 'event data' (what, when, where) about a collection of tag identities (EPCs), while removing duplicate EPCs. At higher levels in the EPC network stack, networked databases and information services provide access to this event data but annotated with additional contextual meta-data (for instance disposition = 'shipped'). The EPC Information Services (EPCIS) interface allows client applications to request the higher-level data, complete with annotations about contextual data.

The EPC Architecture Framework is defined in terms of standardised interfaces that are intended to guarantee interoperability between solutions from different technology providers, while remaining agnostic about implementation details such as the operating system or type and configuration of the underlying databases as well as the technologies used in the implementation. The primary layers of the architecture stack and the layering mechanisms are illustrated in Fig. 3.

2.2.2. DIALOG system (Helsinki University of Technology)

The DIALOG system (DIALOG, 2009) is an open-source solution for tracking objects and accessing data about the objects (Kärkkäinen et al., 2003) using the DIALOG middleware system as illustrated in Fig. 4. It was originally conceived for tracking international shipments related to investment projects, which signifies that information has to be exchanged between a great number of organisations of different sizes. The involved organisations tend to be geographically dispersed and loosely-coupled in the sense that the supply network typically changes for every new project. However, the “location property” of an object rapidly became only one property among others, so the DIALOG system has later been used also for transmitting other kinds of information, such as sensor measurements.

The DIALOG system is a very worthy and, reportedly, affordable entry solution to networking products or objects, with the advantage that any organization can create globally unique identifiers in a decentralised manner at almost zero marginal cost, because the unique identifier used by DIALOG consists of two components, a unique ID string and a URI (Uniform Resource Identifier, Berners-Lee et al., 1994a, 1994b) where the software ‘agent’ of the physical object resides. The URI could be based on a domain name that is already owned by the organization—or is anyway cheap to register.

Since the beginning of the PROMISE project in 2004, DIALOG development became tightly coupled with PROMISE. The PROMISE project has allowed extensions to be made to the DIALOG architecture to provide a PROMISE messaging interface (PMI), where the Web Service based PMI is the main information exchange mechanism implemented by information resources. The implementation of PMI made at Helsinki University of Technology, based on the DIALOG system, is capable of supporting data models appropriate for capturing PLM data.
2.2.3. WWAI network

The WWAI network (whose technology is owned by Trackway Oy of Finland) is reported to be the world’s first peer-to-peer RFID middleware solution (WWAI, 2007).

The key component to the WWAI network is an information object with a unique identity. There is delegation of uniqueness in the sense that the WWAI identity code consists of a prefix that identifies a particular organization responsible for that object and the remainder, with which each organization guarantees the uniqueness of that particular object within their own prefix code. This is logically similar to the delegation approach of the Handle system (Kahn and Wilensky, 2006) except that, syntactically, the WWAI network does not use a discernible delimiter, whereas Handles use the slash character to separate the Naming Authority code from the Unique Local Name.

The WWAI code can be used to access additional data about the object. These can be considered as file ‘attachments’ attached to the object, since it is necessary to specify both the WWAI code and the filename in order to retrieve the documents. It is also possible to request a catalogue of available files for each WWAI object. Fig. 5 illustrates a product information lookup using the WWAI approach.

In the Trackway WWAI network, messages are sent as XML data packets using a proprietary schema over TCP/IP connections. Headers within the packets indicate the sender and sender’s credentials, consisting of the originating node (address(es) of the sender’s server) and the certificate details.

3. Case studies and requirements

3.1. Case studies

While it is possible to directly examine various approaches to PLIM, a more useful approach is to develop a comparison that examines how various approaches can satisfy specific needs relevant to product lifecycle management. We have examined a number of key PLM application cases developed by the PROMISE project consortium to formulate a set of requirements.

The PROMISE project (PROMISE, 2009a) was funded under the Sixth Framework Program of the European Union. The project brought together a consortium of 23 partners from Europe, which included end users such as FIAT, Bombardier Transportation, Caterpillar, Indesit and Intracom Telecom. The vision of the...
project was to develop and demonstrate PLM concepts and strategies, discussed in the previous sections, using new enabling technologies such as RFID and wireless sensors to allow all participants in a product’s lifecycle to manage, control and utilise product information as required from any geographical location. A significant outcome of the project is the development of a PLIM system and a set of ten application demonstrators designed by end users based on their PLM requirements. We have focused on three significant demonstrator applications from the pool of ten developed by the PROMISE consortium for our case studies. The following sections outline and summarise the three case studies and provide a categorized listing of requirements that a PLIM system must be able to satisfy to achieve the objectives of the application scenarios.

3.1.1. Case study 1: FIAT

The domain of the application scenario is the end of life (EOL) phase of the product lifecycle. It specifically deals with the return of end of life vehicles (ELVs) to dismantlers for reprocessing. The demonstrated scenario illustrates how recording information relevant to individual components of a FIAT Punto automobile from production phase through to its usage and maintenance phase in MOL can be used to make decisions regarding whether to re-use a component, recycle a component or dispose of it based on the residual life of the component computed by considering its life history. Understanding the history of products increases the probability of recycling in response to legislative directives which demand reduction in waste and increased reuse of materials, at the same time reducing cost and increasing profitability.

This scenario is increasingly becoming a significant driver for change due to the EU ELV Directive 2000/53/EC discussed in Section 1. Table 1 summarises the various requirements expected from a PLIM system to support the objectives of the demonstrator application.

3.1.2. Case study 2: CATERPILLAR

The demonstrator scenario in this case focuses on the value of PLM to re-manufacturing of components and parts. This is a significant source of revenue especially when the remanufactured product is of potentially large commercial value.

This demonstrator study focused on the re-manufacture and re-use of CATERPILLAR engines from excavators. As such, it is required to automatically record all the information possible regarding the part itself (built date, fabrication plant, type of machine equipped with the engine, etc.), condition and the maintenance history of individual engines and their primary components and to make that information available at a point in time when the machine is retired from service. This data can then be used for decision making at end of life of the excavators to both increase the lifetime of engine components as well as to make decisions regarding whether any primary parts of the engine itself are suitable for remanufacture. Since components may have multiple life cycles it is also important to track parts throughout multiple life cycles of engine components and through the re-manufacturing processes. Table 2 summarises the main requirements for this application.

3.1.3. Case study 3: INDESIT

The third demonstrator application scenario aims to optimise the lifecycle of a refrigerator using a “an intelligent control system”. The purpose of the control unit was to obtain statistical data from the appliance installed at a customer location so that this information can be analysed to predict service needs prior to actual failures (predictive maintenance PLM strategy). Consequently, the middle of life usage information can be used to determine, in advance, the spare parts needed for a maintenance activity and improve the scheduling of resources such as personnel and tools while moving to minimize inventory stock levels. As a result predictive maintenance will increase the availability of machines, reduce unexpected failures and disruptions to operations as well as unnecessary service operations performed when these are based on fixed time periods rather than on the actual state and usage of the machine. Consequently, a predictive maintenance strategy enables the scheduling of maintenance considering a component’s past and present performance statistics and parameters linked to the usage of the appliance. The data collected in MOL can then be used by the product designers at INDESIT to improve the design of parts in the future. A PLIM system needs to provide access to machine usage data from MOL such as power consumption, warnings (e.g. sensor failures) and field knowledge on usage (such as failures to close the door, length of time doors have been opened). Table 3 outlines the main requirements to enable the demonstrator scenario’s PLM strategies. More significantly this demonstrator is a demonstration of the “The Servitization of Products” concept

Table 1
Summary of requirements obtained from the demonstrator in case study 1.

<table>
<thead>
<tr>
<th>Life cycle stage</th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOL</td>
<td>Identify individual components and the storage of related quasi-static data at an individual component level</td>
</tr>
<tr>
<td></td>
<td>Access previous lifecycle information on recycled parts to enable their suitable use in other FIAT automobiles</td>
</tr>
<tr>
<td></td>
<td>Make component design improvements as well as new components using readily available MOL and EOL data on deployed instances of similar components</td>
</tr>
<tr>
<td>MOL</td>
<td>Capture and storage of component usage information (such as hours of operation, engine temperature profile) and significant component related events (such as peak pressure or excessive temperatures)</td>
</tr>
<tr>
<td></td>
<td>Capture and storage of maintenance events such as history of maintenance activity, list of replaced parts and corresponding data for individual engines</td>
</tr>
<tr>
<td>EOL</td>
<td>Access component relevant information captured from BOL and MOL</td>
</tr>
<tr>
<td></td>
<td>Decision support functions to support human decision making at EOL (recycle/re-use/ remanufacture)</td>
</tr>
</tbody>
</table>

Table 2
Summary of requirements obtained from the demonstrator in case study 2.

<table>
<thead>
<tr>
<th>Life cycle stage</th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOL</td>
<td>Uniquely identify individual machine structures and the storage of related quasi-static data (e.g. build date, fabrication plant and type of machine equipped with the engine)</td>
</tr>
<tr>
<td></td>
<td>Access previous lifecycle information (for example service and maintenance histories) on re-used structures</td>
</tr>
<tr>
<td>MOL</td>
<td>Similarly to the first case study, capture maintenance events and significant event occurrences (e.g. exceeding temperature thresholds) during engine life defined as hours of operation of the machine</td>
</tr>
<tr>
<td></td>
<td>Capture and storage of maintenance events such as the history of maintenance activity, list of replaced parts and engine performance data</td>
</tr>
<tr>
<td>EOL</td>
<td>Access all specific component relevant information captured from BOL and MOL (such as service and maintenance histories)</td>
</tr>
<tr>
<td></td>
<td>Decision support functions to support human decision making at EOL (recycle/re-use/remanufacture)</td>
</tr>
<tr>
<td></td>
<td>Track parts for increased visibility in order to support logistics operations along the remanufacturing processes</td>
</tr>
</tbody>
</table>
The applications, requires an information management system to provide the capability to track individual lifecycle histories of each unique object and provide timely access to that information in a manner that is seamless to the physical state or the geographical location of the object. Given such a context, a PLM information system using a product centric paradigm needs to address the following key issues:

1. **Information resources** (Where to store data?).
2. **Product-item link** (How do we create a link between products and its data sources?).
3. **Timely information** (How do we ensure that we meet real-time data requirements?).
4. **Synchronisation** (How do we ensure data synchronicity in an environment of multiple data sources for an individual object?).
5. **Reconfigurability** (Is the information system robust enough against changes to underlying systems and heterogeneity in data capture devices?).
6. **Application support** (Can the system provide support to high level PLM applications?).

### 3.2. Requirements

There are clearly a set of intersecting requirements among the various demonstrators. The case studies together represent a movement of a product through the various phases in its lifecycle (BOL, MOL and EOL) and the distributed collectors and users of the product's lifecycle data. The case studies also highlight information flow requirements between various stages of the life cycle as shown in **Fig. 6**. Each case study has shown a requirement for access to information from other stages of the lifecycle for various processes.

The applications show that the product information first begins at the design stage in the BOL phase where the product is a collection of ideas, design documents or requirements. This stage of the process can be spread across different organisations and across geographical boundaries as is the case with both FIAT and CATERPILLAR. PLM strategies in the BOL phase rely on accurate and timely information from both the MOL and EOL phases of similar products or components, while the service and maintenance activity strategies (for example predictive maintenance) rely on information from both MOL and even EOL where predictive maintenance requires access to historical data from previous lifecycles for statistical analysis (Cao et al., 2007). Finally processes in the EOL phase require a holistic view of the product’s entire lifecycle to make decisions regarding its life after use.

Providing support to various PLM strategies, as highlighted by the applications, requires an information management system to provide the capability to track individual lifecycle histories of each unique object and provide timely access to that information in a manner that is seamless to the physical state or the geographical location of the object. Given such a context, a PLM information system using a product centric paradigm needs to address the following key issues:

1. **Information resources** (Where to store data?).
2. **Product-item link** (How do we create a link between products and its data sources?).
3. **Timely information** (How do we ensure that we meet real-time data requirements?).
4. **Synchronisation** (How do we ensure data synchronicity in an environment of multiple data sources for an individual object?).
5. **Reconfigurability** (Is the information system robust enough against changes to underlying systems and heterogeneity in data capture devices?).
6. **Application support** (Can the system provide support to high level PLM applications?).

#### 3.2.1. Information resources

It is generally not possible nor particularly advantageous to store all the information at various phases of a product with the product item itself. This essentially requires that the information be stored in networked ‘backend’ systems or information resources.

The significant advantage with networked information systems is the ability for one or more parties to access the information from anywhere, soon after it is generated. There are numerous other reasons for storing data on backend systems such as cheaper and unlimited data storage capacity, considerably better data security and access to data in the absence of the product.

Collection and storage of through life information requires a collection of information resources to be available on a network. Here, there is a need for standards about how the information is accessed or expressed, in order to be able to make sense of the information in an efficient manner, rather than having to decipher the meaning and data structure each time from the raw data for each individual data source.

The available information related to an object may fall into various categories, such as:

- Information about geometry, composition, design and assembly, such as computer-aided design (CAD) files.
- Information which is intended to be human-readable, such as photos and web-pages containing descriptions, instructions such as disassembly procedures.
- Data collected about the object during its life-cycle, such as historical records of its location, temperature or usage information (hours of operation) as well as references to business transactions in which it was involved, such as warranty information and repair records.
- Information services related to the object, such as an interactive instruction manual, diagnostic tools or software updates for firmware or device drivers.
- Each of the previous custodians of the object (including its creator or originator) may provide a variety of these types of information resources, most likely stored at a centralised location such as their own respective data warehouse.

#### 3.2.2. Product-item link

Given the distributed nature of data collection and storage there must be a mechanism for associating products with their relevant lifecycle data in networked information systems as well as in/on products themselves.
3.2.2.1. Unique identifier. The case studies have shown a need for considering not just engines of excavators but a particular engine of a particular excavator. The need for uniquely identifying objects can be easily addressed by attaching unique identifiers to objects. The use of a unique identifier (UID) with a global scope for each object under consideration then enables the collection, location and retrieval of information related to an individual product anywhere in the world. Requirements on such unique identifiers for PLM have been considered in Främling et al. (2007a).

The unique identifier can be used to discover and access information associated with the UID from distributed information resources, similarly to the manner in which web addresses or uniform resource locators (URLs) are used to access information from the Internet. The UID forms the link between the product and its associated information collected and possibly distributed at various organisations and locations. It is this concept that has underpinned the recent developments in the product centric paradigm for lifecycle information management.

As a simple illustration, consider a primary one-to-one mapping for each object, linking the object with information provided by the creator or originator of the object, for example the manufacturer at BOL. This primary link therefore forms a pointer to authoritative information about the object.

More generally, there is a need for a one-to-many mapping or sequence of pointers for a UID to multiple suppliers or custodians of information associated with an object, since the object may have passed through various parties or stages in its life cycle before reaching its current custodian or status. As illustrated in Fig. 4, during the passage of time different parties collect and record information about an object. In addition, due to the nature of various lifecycle phases and the complexity of products whose composition will consist of other sub products, information related to products is spread across various organisations, locations and possibly collected by different systems. A routing or a lookup mechanism is also needed to find and, where necessary, update lifecycle information resources that may be distributed across organisations and locations.

3.2.2.2. Routing and lookup mechanism. While the UID allows the creation of a link between a specific product and its associated information located on one or many information resources as shown in Fig. 7, a routing or lookup mechanism is required to bridge the gap between the UID and the information resources linked to that object.

It is important to distinguish the mechanism as ‘routing’ only when referring to the transmission of data that has a well-defined recipient or destination. If the functionality of the method used is based on retrieving the destination or source of the data, the mechanism is a ‘lookup’. For instance, DNS (domain name server) and search engines help with lookup while TCP/IP (transmission control protocol/internet Protocol), SMTP (simple mail transfer protocol) and UDP (user datagram protocol) handle routing.

Automated use of a lookup mechanism is required to provide not only a set of pointers—but also to indicate the context of each (that is the type of information available by way of each pointer) in order that the most appropriate link is followed, depending on the kind of information service required. Hence information systems must not only support a UID but must also provide a routing or lookup mechanism.

3.2.3. Timely information

The vision of PLM is that changes in the physical world are reflected by timely changes in the world of information—and ultimately, that the converse can also be true, namely that PLM improves automation and the vision of computer systems to the extent that movements of objects and physical actions upon them can be effected merely by changes of information or flows of data. This is reflected in applications requiring predictive maintenance and the use of MOL data to predict design modifications. Therefore an expectation of a PLM system is that the system should be responsive, with the ability to provide timely information.

3.2.4. Synchronisation

During the movement of products through their lifecycle they are likely to have only intermittent network access (this was indeed the case with the case study 2 in Section 3.1.2 where CATERPILLAR technicians perform in-field servicing of machines typically at remote destinations, usually with little or no network access to the Internet). While the case studies have highlighted the importance of data availability and data collection, what has been concealed is the implied requirement for maintaining causality and validity of the data during distributed data collection and storage. There must be mechanisms where,
especially during MOL, products can update data collected (such as maintenance events or product usage records) or modify data about objects (such as an update of a BOM) with backend systems while maintaining the validity of information and correct sequence of events duplicated in several places and collected at various lifecycle phases or locations. Therefore there is a need for synchronising remote event data captured and stored locally on the object with networked information resources since these remote events may not otherwise be accessible to requests for MOL data. This is primarily as a result of the intermittent access of products to network resources (typically through the Internet) during the various phases of their lifecycle. The task of managing data collected by objects in the field and linking them to networked resources must be carefully orchestrated to ensure that conflicts of validity can be resolved seamlessly and to ensure that the correct sequencing of data and event updates is guaranteed (for example maintenance events at different locations) when there are multiple actors collecting lifecycle data at multiple locations, especially in the MOL phase (Suzuki and Harrison, 2007).

3.2.5. Ease of reconfigurability

In a PLIM system it should be possible to transparently reconfigure a number of implementation aspects with minimal disturbance to applications and parties involved at various lifecycle stages of the product who access the system. Examples of changes that might be required are changes to automatic data capture technologies (RFID tag or wireless sensor vendors, protocols, operating frequency of wireless sensors), changes to network topology of devices, as well as changes to business partners (for example a preferred maintenance company) and processes. Clearly ease of reconfigurability is another essential requirement for an information system to be robust against changes to underlying systems and its ability to support a heterogeneous set of data capture technologies.

3.2.6. Application support

The information architecture should allow the easy implementation of an application layer. For instance, it should provide decision support tools and track-and-trace algorithms to support PLM decisions, processes and operations.

4. Comparing ability and suitability

An information systems architecture for PLM needs to be able to address Product Lifecycle Information Management (PLIM) requirements from the design and manufacturing (BOL) phase, through the usage phase (MOL) until the end-of-life (EOL) phase; and back again in the sense of feeding back information to earlier phases in the lifecycle. We have identified a set of key technical requirements for such a PLIM system in Section 3 and these requirements are summarised in Table 4.

4.1. Ability to meet technical requirements

We can compare the alternative approaches taken by the three different architectures to address the key technical requirements outlined in Table 4 to analyse their ability to support PLM strategies.

4.1.1. EPCglobal network architecture approach

4.1.1.1. Globally unique identifier—The Electronic Product Code (EPC). The Electronic Product Code is designed to be a scalable licence-plate identification number that enables linking between an individual product and its associated information resources or backend information services. EPC product identifiers can be formatted as URNs (Universal Resource Names, Moats, 1997) for use in the EPC network as described in the 'EPC Tag Data Standard' (a ratified open standard, EPCglobal, 2007b).

The Electronic Product Code achieves uniqueness by delegating the responsibility for blocks of its number space (EPC Manager numbers) to particular companies, while guaranteeing uniqueness globally by central management of the allocation of EPC Manager numbers, to ensure that only one company would be assigned any given EPC Manager number. Furthermore, in an effort towards coherence, the existing family of GS1 coding schemes (e.g. serialised GTIN—Global Trade Identification Number, SSCC—Serial Shipping Container Code, or GRAI—Global Returnable Asset Identifier) can be expressed within the EPC format.

4.1.1.2. The lookup mechanism—The Object Name Service (ONS) and Discovery Services. The Object Name Service (ONS) provides a lookup service that decouples the EPC identity from the address(es) of associated information resources. The ONS provides a lookup service to obtain a list of URLs where authoritative information, usually the information held by the manufacturer, related to an object’s EPC can be obtained. Not only do ONS records provide a set of URLs, they also provide meta-data to specify the type of information service provided by each URL, for example a web service, product information web page or an EPC Information Service (EPCIS).

ONS is merely an implementation of the Domain Name Service (DNS). All ONS records for EPCs are stored within DNS records for subdomains of a domain such as onsepc.com. ONS records use DNS type 35 (NAPTR—Naming Authority Pointer) records for returning results to ONS queries. ONS supports a scalable hierarchical lookup service using existing DNS technology and protocols. However, ONS does not currently provide for client authentication or access controls. It is also considered unsuitable for storage of links to multiple lifecycle information providers per individual object, for reasons of lack of security and potential overloading of the underlying DNS infrastructure, when dealing with billions or trillions of highly dynamic records per year. The root-level of the ONS is currently administered by GS1 EPCglobal Inc. and the operation of the servers is currently subcontracted to Verisign Corporation.

The ONS lookup services do not provide serial level lookup for individual objects. For serial level resolution, work is currently under way to develop standards for Discovery Services (DS) to provide serial-level lookup services across organisations (multiple pointers to object related information and services). DS will

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Globally unique identifier</td>
<td>To provide the link between product instance and information</td>
</tr>
<tr>
<td>Routing or lookup mechanisms</td>
<td>To provide the ability to find or search for product related data</td>
</tr>
<tr>
<td>Information resources</td>
<td>For the persistent storage of life cycle data</td>
</tr>
<tr>
<td>Timely information</td>
<td>To ensure that the information system can support PLM strategies based on real-time information</td>
</tr>
<tr>
<td>Synchronisation</td>
<td>To ensure data synchronicity in an environment of multiple data sources and storage locations for an individual object</td>
</tr>
<tr>
<td>Reconfigurability</td>
<td>To provide robustness and interoperability between technologies, vendors and devices</td>
</tr>
<tr>
<td>Application support</td>
<td>To ease the burden of creating high level PLM applications</td>
</tr>
</tbody>
</table>
provide a dynamic lookup service such that DS records can be updated in real time, with immediate effect. DS enable a client to discover multiple sources of serial-level (individual product) object related information that is distributed across many information systems from multiple providers (multiple organisations who have collected information about the object at some time in its lifecycle).

4.1.1.3. Information resources—EPC Information Services (EPCIS). In the EPC network architecture, information resources at various organisations and locations gather product-related information at various stages of its lifecycle. This product related information can then be accessed through the standardised interface provided by the EPCIS interface as illustrated in Fig. 3.

A client program can use the lookup mechanisms to find the locations of lifecycle data and access current and historical data related to a product using the EPCIS interface. Client applications can obtain serial-level product information as well as obtaining higher-level semantic annotations such as business transactions or processes associated with event observations related to objects.

4.1.1.4. Timely information—events, filtering and ‘push’ mechanisms. ALE and EPCIS standards are primarily concerned with standardizing access to selected ‘event data’ which is relevant or actionable to business applications such as warehouse management systems. Both ALE and EPCIS support a ‘publish and subscribe’ mechanism, whereby a client application may ‘subscribe’ to a particular stream of filtered data or a particular EPCIS query which has been already defined, with the assurance that any newly received data which matches the criteria of the filter or query will be automatically returned or sent on to a specified recipient destination. This is also known as a ‘push’ mechanism, as opposed to polling mechanisms where a client application would have to periodically check whether new data is available. The filtering middleware or information services ‘publish’ information about new events to ‘subscribers’ as soon as the data is available.

4.1.1.5. Reconfigurability—a layered architecture. The whole EPC network has been designed with a layered architecture, to provide maximum flexibility of reconfiguration to users, while minimizing disruption when changes are made. The EPC Architecture Framework is defined in terms of standardized interfaces that are intended to guarantee interoperability between solutions from different technology providers, while remaining agnostic about implementation details such as the operating system or type and configuration of the underlying databases as well as the technologies used in the implementation. The primary layers of the architecture stack and the layering mechanisms are illustrated in Fig. 5 and described in Table 5.

Both Application Level Events (ALE) and EPC Information Services (EPCIS) support a web services interface for cross-platform operations between software written in different programming languages. EPCIS also supports other transport bindings such as EDI (Electronic Data Interchange) technologies such as EDI INT AS2, which is already widely deployed in the consumer goods/retail sector. Both standards define XML schema (XSD) to standardize the format of the query or filtering and also the returned payload data.

4.1.1.6. Data analysis. Discovery Services may provide not only tracking capability to locate the current custodian—but also traceability, to locate all previous custodians of a given object and perhaps additional functions related to product recalls or product status verification. Development of an API with a built in library of tracking algorithms to complement the EPC network stack was recently carried out in the BRIDGE project Work Package 3, also funded under the 6th Framework Program Information Society Technologies (BRIDGE, 2009b).

4.1.2. DIALOG system approach

4.1.2.1. Globally unique Identifier—ID@URI. Its unique identifier consists of two components, a unique ID string and a URI (Uniform Resource Identifier, Berners-Lee et al., 1994a, 1994b) where the software ’agent’ of the physical object resides. This approach of ID@URI parallels the format of e-mail addresses, where the software ’agent’ of the physical object resides. This approach of ID@URI parallels the format of e-mail addresses, where the local part or mailbox name before '@' is required to be unique within a particular URI specified after the '@' symbol. When RFID tags are used, unique ID is provided by the unique Tag ID (which is often hard-coded by the RFID Tag Manufacturer in Read-Only Memory)—and by recording the URI into the tag’s ID memory. However, as long as the ID is unique inside the URI space, an existing serial number, an EPC code, a GTIN code or any other proprietary or standard identifier can be used. For product data, this might be a URI within a domain owned by the manufacturer. For tracking shipments, this might be the URI of a company that requires, manages and securely distributes the tracking data. In some applications, it is preferable to include the ID directly in the URI or as an HTTP parameter rather than representing it explicitly in ID@URI form.

In the DIALOG system, the central authority guaranteeing uniqueness of the URI parts is effectively IANA (Internet Assigned Numbers Authority), since the company hosting the software agent at a particular URI must have registered the corresponding domain—and must continue to keep it registered so long as objects encoded with those URIs remain in circulation.

4.1.2.2. The Routing mechanism. Even though the identifier is of the form ID@URI, the URI tends to be a URI in practice. In the case of a URL, routing is initially provided by DNS and Fig. 4 illustrates the process of a user query for an object’s ID@URI. Depending on the hostname within the URI, web servers, web services, etc. may perform some level of routing based on the pathname information within the URL.

If the URI is a URN the DIALOG system does not use nor provide any specific routing mechanism. At the time of writing,
there are very few general purpose mechanisms for resolving URNs into URLs other than the Handle System (http://www.handle.net/) or DOI (Digital Object Identifier) system (http://www.doi.org/) and domains specializing in routing or permanent addressing, such as pur org. Conceivably, a DIALOG identifier might use one of these for the URI and thereby have the best of both worlds, i.e. a stable, long-lived but also actionable identifier.

**4.1.2.3. The Information Resources—software agents.** The software agents provide interfaces for the following.

- Receiving information updates about products (e.g. location updates about shipments, sensor values and alerts from products).
- Linking the DIALOG system identifier, ID@URI, to internal company references such as transaction IDs, shipping waybills or serialized product IDs such as barcodes/SKUs augmented with serial numbers.
- Retrieving and displaying item-related information.

The DIALOG system uses DNS to resolve the URI to obtain the IP address of the software agent which can provide services for the product. The particular URI scheme used by DIALOG can also contain other information such as a particular agent service, a protocol specification, a desired port number and directory paths (Huvio et al., 2002). Once a connection is established, a bi-directional information exchange can occur directly with the remote software agent representing the object, which is responsible for storing the data.

If other ‘information providers’ for an object need to be integrated than the one indicated by the URI, it has been proposed in Främling et al. (2007b) to explicitly store retrievable ID@URI links to those information providers. The model for doing so comes from the software engineering concept called ‘Design Patterns’ and notably the ‘Composite’ and ‘Observer’ patterns. However, storing such explicit links may be fragile in practice and a more robust and dynamic mechanism for discovering information providers would be desirable.

In the DIALOG system, the presumption is that all parties who handle the tagged object would need to contact the object’s agent (indicated by the URI) and either provide it with the data it collected (e.g. observations) or provide it with a reference to the data. If all information about the object is not directly available at the URI, the object’s agent may link to other parties as additional information providers. However, if the link is altered or broken, it raises the issue of how to access the lost information behind the broken link.

**4.1.3. WWAI network approach**

**4.1.3.1. Globally unique identifier—The WWAI identity code.** The key component to the WWAI network is an information object with a unique identity. There is delegation of uniqueness in the sense that the WWAI identity code consists of a prefix which identifies a particular organization responsible for that object and the remainder, with which each organization guarantees the uniqueness of that particular object within their own prefix code. This is logically similar to the delegation approach of Handle system (Kahn and Wilensky, 2006) except that, syntactically, the WWAI network does not use a discernible delimiter, whereas Handles use the slash character to separate the Naming Authority code from the Unique Local Name.

**4.1.3.2. The Information Resources—nodes storing files and catalogues of files.** The Trackway WWAI network has no centralized storage; all information about the WWAI objects is stored by information providers (nodes on the WWAI network) who control which other parties may access their data.

The WWAI code can be used to access additional data about the object. These can be considered as file ‘attachments’ attached to the object, since it is necessary to specify both the WWAI code and the filename in order to retrieve the documents. It is also possible to request a catalogue of available files for each WWAI object.

Similarly to the DIALOG approach, the WWAI system already claims to incorporate other ‘information providers’ for an object, that is other than the one indicated by the WWAI identity code, by explicitly storing retrievable links to those information providers (Främling et al., 2003, 2006, 2007b).

**4.1.3.3. The Routing Mechanism—the WWAI joining protocol.** Unlike the EPC network, there is no centralized hierarchical Object Name Service—instead, objects join the network and the WWAI joining protocol adds the objects as new nodes on a dynamic virtual map (i.e. overlay network) in which each node typically knows paths to six or more neighbouring nodes. The route to the source of relevant information for the object is then obtained by navigating through the peer-to-peer overlay network, rather than the hierarchical navigation which DNS and ONS use. Both the joining mechanism and the search mechanism use the same algorithm for navigating the peer-to-peer network. The dynamic joining mechanism means that an information provider may change IP address or URL and that the new address will be automatically detected upon rejoining the WWAI network, without needing to update any ONS records. Fig. 5 illustrates a product information lookup using the WWAI approach.

**4.1.3.4. Reconfigurability.** Like the EPC network and unlike the DIALOG system, the object’s unique identifier is decoupled from the URL hosting the information service or software agent, which should make the WWAI network more resilient to changes in the information provider’s address.

**4.1.3.5. Timely information—subscription to events.** The Trackway WWAI network provides mechanisms for other parties to subscribe to an object and receive events and updated information about a particular object of interest, as well as allowing the owner of the object to control whether each of their WWAI objects is public or private—and to impose restrictions on which partners are allowed to receive updates.

Conversely, other players may register with the owner of a WWAI object as an information provider. For example, a distribution company or retailer could register with the manufacturer as additional information providers for a particular object. The owner (e.g. manufacturer) would be free to accept or decline—but would maintain additional links to other information providers whom they accepted. More significantly, the WWAI model of information control is radically different from current information sharing practices in many industry sectors, where manufacturers typically receive very little fine-grained information or feedback on tracking from downstream parties on the supply chain. By contrast, for the EPC network, once Discovery Services are standardised and implemented, provision of links across the supply chain to other information providers would be determined by the policy of the operators of Discovery Services, rather than primarily by the manufacturer.

In the Trackway WWAI network, messages are sent as XML data packets using a proprietary schema over TCP/IP connections. Headers within the packets indicate the sender and sender’s credentials, consisting of the originating node (address(es) of the sender’s server) and the certificate details.
4.1.4. Summary

Table 6 describes the varying degrees of capabilities of the architectures to meet the needs of product instance level management throughout a product’s lifecycle. Table 6 reveals that all three architectures indeed satisfy a majority of the key requirements to varying degrees but none of the approaches currently provides a comprehensive set of capabilities required to realise PLM strategies.

In particular, none of the approaches currently provide effective and comprehensive mechanisms for data synchronisation. It is a significant issue when dealing with PLM data where data may be collected by several actors interacting with an object throughout its life cycle (Suzuki and Harrison, 2007). It is also a significant challenge. For example, it may be necessary to have potentially two-way data synchronisation from a physical object to the network (via intermediary such as hand-held device or PDA with intermittent network connectivity) and vice versa, from the network to the physical object, while also supporting and checking digitally signed event updates.

In addition, there is a lack of tools and support for building an application layer capable of providing the services needed to implement the PLM strategies and concepts we have discussed in this paper.

While we have examined a set of primary technical requirements, two important considerations that were not highlighted through the analyses centre around support for sensor data management, and an implied requirement for bi-directional information flows within the approaches needs to be articulated here.

For the vertical information flow, data might flow only upwards from edge devices such as RFID tags towards repositories or also downward, e.g. writing data or control information back to tags or other embedded devices. The flow of data from repositories may also be critical in some service operations in the middle of life, where parameters such as thresholds or the behaviour of automatic data capture devices attached to objects need to be altered from the application level.

However, currently there is support for only an upward flow of information from the edge of the network with both the EPCglobal Network and the WWAI approach. In addition, EPCglobal Network currently has no standardised support for sensor data propagation upwards to data repositories. Lack of support for managing sensor data is a significant drawback for the use of EPCglobal network for PLM applications since the management of sensor data is critical for monitoring the condition of products in all three case studies. Also the EPCglobal and WWAI approaches are mainly focused on using RFID as the enabling technology. This is not sufficient for most PLM applications that require support for a mix of RFID, wireless sensors and other embedded systems.

However each approach has its own set of distinctive advantages as well as hurdles hindering the adoption as a PLM system.

4.2. Suitability

The following sections will discuss the merits of different approaches taken and assess the suitability of the architecture to support various PLIM system requirements.

4.2.1. EPCglobal approach

A key advantage of the EPC network is the definition of granular and standard interfaces through an industry driven effort to enable testing and certification of interoperability as well as the wide scale industry adoption of EPCglobal standards.

Despite the availability of standards and industry support for the EPC network approach, to date, there is no standard specification for Discovery Services and the scalability of DS approach has proven to be a difficult challenge. As a result of the commercial sensitivity of serial-level data about volumes and flows of goods, Discovery Services are like likely to have much more stringent security requirements than the Object Name Service requires. DS will be required to handle and apply individually customised security policies to multiple queries for the same unique object id. Another significant drawback of the approach has until recently been the requirement by companies wishing to use the network to obtain a paid membership from EPCglobal. In particular, the membership provides a company with a guarantee of the uniqueness of each EPC and grants them access to core EPCglobal services such as the ONS.

4.2.2. DIALOG system approach

Like the EPC network, the DIALOG system allows each company to maintain ownership of the data they create or collect

\begin{table}
\centering
\begin{tabular}{|c|c|c|}
\hline
 & EPCglobal network & DIALOG & WWAI \\
\hline
Globally unique identifier & Yes. The electronic product (EPC) & Yes. ID@URI approach & Yes. WWAI identity code \\
Routing mechanism & Lookup services provided by the ONS and DS & No specific routing mechanism but relies on the URI obtained from the object identifier & No specific routing mechanism but relies on registration of nodes on joining the network \\
Information resources & Decentralised data storage and standardised access to networked resources & An agent representation of the object stores object information which is accessed through the agent hosted at a URL. Real-time response support is limited to “pushed” events & Decentralised data storage; all information related to objects is stored on the nodes of the WWAI network Real-time responses are possible \\
Timely information & Real-time responses are supported & There are no mechanisms for data synchronisation & There are no mechanisms for data synchronisation \\
Synchronisation & Limited support through the use of two time stamps to mark event generation time and event receipt time & There are no mechanisms for data synchronisation & \\
Reconfigurability & Extremely flexible because of the layered service-oriented architecture and standardised interfaces & Limited flexibility due to URIs embedded into products themselves & More flexible than DIALOG as an object’s UID is decoupled from the URL \\
Data analysis & Supports only track-and-trace functionality* & No formal data analysis layer but application-specific agents handle storage and analysis of data (e.g. track-and-trace) & No formal data analysis layer \\
\hline
\end{tabular}
\caption{Comparing the ability of various architectures satisfy PLIM requirements.}
\end{table}

* The BRIDGE project has developed enhanced track and trace tools and decision-support tools.
and to exchange the data packets in a uniform way between peers as soon as the address of the information service of software agent is known. However, for the DIALOG system, there is currently no definition of the internal content or syntax of the data that is exchanged nor a fully-developed fine-grained query mechanism for accessing relational data about the object. This means that two communicating parties have to agree on the information exchange standard to use (as defined, e.g. by EPCIS) or use their own proprietary exchange format specification in order to interpret the information in the same way at both ends. The PMI developed through the PROMISE project is expected to address this issue through the provision of a data exchange standard. Through DIALOG’s “protocol plug-in” mechanism, supporting other protocols such as EPCIS might also be feasible without significant changes to the core DIALOG system.

Unlike the EPC network, no separate ‘manager number’ is required for the URI used. Since many companies in the developed world have already registered a domain name, they may then freely create any URIs and subdomains they choose, so the marginal cost of adopting the DIALOG network is minimal. However, in practice, the URI used in the DIALOG system is usually a URL rather than a URN. As such, the system is quite fragile; if the URL or more specifically the local path of the software agent is changed without taking care to forward from the original URL, objects whose tags have been written with a URL which is no longer valid will fail to resolve on the DIALOG network. Besides fragility, a further disadvantage of directly encoding the URI in the tag memory of an RFID tag is that a much larger number of bits may be required than for a more compact identifier such as the EPC identifier is not tied to network addresses.

Further, queries are limited to those that specify the timely response to queries as well as the quality of the response. Furthermore, queries are limited to those that specify unique object identifiers as opposed to classes or types of products.

Perhaps one of the more important distinctions is that the EPCglobal approach provides a layered architecture stack with well defined standard interfaces and data models. While DIALOG is an open-source approach providing access to non-standardised interfaces and message formats, the implementation provides support for other interfaces by adding simple protocol adapters. In contrast WWAI provides no such standard interfaces or data models. Use of standards is critical when dealing with data exchange between multiple organisations and heterogeneous systems from various providers that need to be interoperable for systems to function. Standards also play a role in enhancing competition in the technology and systems provider market by allowing multiple vendors to compete for system components and technologies.

In addition, the DIALOG architecture requires relatively more expensive RFID tags (re-writable) compared to write-once RFID technologies that can be used with the EPCglobal approach. In particular the EPCglobal architecture is supported by strong industry-driven standards, however, DIALOG does not provide standard interfaces such as the EPCIS for exchanging object related data, although an EPCIS interface could be integrated in addition to the current interface implementations (seven interface implementations, including standardised, open-source based and proprietary ones).

Furthermore, there is a significant shift in the implementation of access control away from the WWAI approach where manufacturers of objects exercise dominant control over collecting information from other parties and sharing of that information with client applications. In contrast, the EPC network (through the Discovery Services mechanism) allows highly granular access control policies to be specified by parties collecting information in order to determine access rights by other entities to product related data.

Finally, the architectures appear to differ in their current suitability for different phases of a product’s lifecycle. Table 8 summarises the suitability of the different architectures for satisfying PLIM requirements in the various lifecycle phases. The three approaches provide varying degrees of support to satisfy the information needs of each lifecycle phase identified in the case studies in Section 3.

It is evident that the initial focus of the EPCglobal and WWAI approaches was the supply chain phase and therefore these approaches provide strong support in the BOL phase, particularly since, the EPCglobal approach was initially motivated by the use of RFID technology1 to implement the product centric computing paradigm. The EPCglobal approach provides specific support to track and trace application functionalities. Although a product’s lifecycle does include the supply chain phase, in the BOL stage, it is only a limited part of the entire lifecycle. However, at the time of writing this article GS1 EPCglobal is in the process of engaging with a number of industry sectors and looking beyond mere supply chain usage.2

Compared to DIALOG, both EPCglobal and WWAI approaches revealed to be very limited in their current ability to be used in PLIM systems, particularly for meeting the needs of managing product information in the MOL phase and subsequently in the EOL phase. Furthermore, the DIALOG approach has the

---

1. However, EPCglobal standards are not limited exclusively to use with RFID systems; indeed EPCglobal standards such as ALE, ONS, EPCIS and Discovery Services can be used with other Auto-ID technologies including linear barcodes and 2-dimensional matrix code.
2. Its Consumer Electronics Industry Action Group is currently gathering use cases and requirements considering the product lifecycle from manufacturing through to End of Life (see http://www.epcglobalinc.org/what/action_group/groups/).
have found that the architectures we have considered only partially address the needs of PLM strategies, although, none of them yet provide a comprehensive platform. Consequently, the comparisons and the analyses in this paper have not provided a clear ‘front runner’ approach to meeting the requirements for a PLM system. Nevertheless, the DIALOG approach currently provides the most potential for supporting the PLM needs of MOL and EOL lifecycle phases, while EPCglobal and WWAI are currently better adapted to BOL and supply chain management applications. Still, all three architectures have extension mechanisms that can allow them to evolve and be adapted for new uses. It is also conceivable that elements from one architecture could be re-used in the others.

There is a simple reason behind the failure of the architectures to adequately support the needs in the MOL phase (see Table 8) although they support key technical requirements. Meeting the product information needs in the usage phase of products (MOL) is challenging, in particular for consumer products. Before the usage phase, all information about the product was managed and stored in organisational information systems. When the product

### 5. Conclusions

This paper presents the results of an architectural study for distributed RFID and sensor data management systems, which aims to integrate event data from various nodes distributed across the lifecycle of a product to respond to the information needs of PLM strategies at BOL, MOL and EOL. We have surveyed related architectural approaches to meet the information needs of PLM by closing the information loop between various stages of a product’s lifecycle. We identified and discussed key technical requirements for implementing PLM objectives. We then compared the ability and suitability of different architecture alternatives to meet the key technical requirements we have identified.

From a technical, capability and a suitability perspective we have found that the architectures we have considered only partially address the needs of PLM strategies, although, none of them yet provide a comprehensive platform. Consequently, the comparisons and the analyses in this paper have not provided a clear ‘front runner’ approach to meeting the requirements for a PLM system. Nevertheless, the DIALOG approach currently provides the most potential for supporting the PLM needs of MOL and EOL lifecycle phases, while EPCglobal and WWAI are currently better adapted to BOL and supply chain management applications. Still, all three architectures have extension mechanisms that can allow them to evolve and be adapted for new uses. It is also conceivable that elements from one architecture could be re-used in the others.

There is a simple reason behind the failure of the architectures to adequately support the needs in the MOL phase (see Table 8) although they support key technical requirements. Meeting the product information needs in the usage phase of products (MOL) is challenging, in particular for consumer products. Before the usage phase, all information about the product was managed and stored in organisational information systems. When the product
instance is taken into use, it becomes a unique individual that produces and possibly collects information about its state, environment etc. through its sensors and embedded information systems. Even for products that have little or no embedded processing power, information about their use can be collected by systems of different users or organisations, such as service companies or computing devices in people's homes.

When information about a product instance becomes distributed over embedded devices and various information systems, interoperable messaging of instance-level information becomes necessary in order to collect or access all the information when and where it is needed. Consequently, interoperable messaging, robust synchronization and unambiguous interpretation of instance-level information is a key requirement and enabler for PLIM, especially in the MOL phase. It is particularly important to be able to correctly determine the state or configuration of an individual product instance, given a number of events that may originate from different sources and organization and correspond to sensor readings, records about physical modifications, etc.

The authors have been involved in the implementation of a great number of real-life instance-level information systems for many application domains, ranging from multi-organisational shipment tracking to the collection of sensor and event data from both industrial and consumer products. In all cases, the lack of a universal messaging protocol has posed a significant challenge for developing PLM strategies. Unfortunately closing the information gap in the MOL phase usually leads to implementing several domain-specific protocols for different needs, such as EPCIS for supply-chain tracking with RFID or oBIX (Open Building Information Xchange – see http://www.obix.org/) for building automation systems. In many cases, no suitable messaging protocol can even be found, which leads to the implementation of many proprietary protocols.

Consequently, an architecture pursued by PROMISE project demonstrator developers was based on addressing this previous challenge by extending DIALOG, perhaps the architectural approach currently most suitable to PLIM because of its ability to support MOL PLM strategies. A key development made to DIALOG was the design and implementation of the PROMISE Messaging Interface (PMI) at Helsinki University of Technology. This interface has enabled the architecture to support data models appropriate for capturing PLM data (not limited to unique identity information as in the EPCglobal approach but other condition information such as that provided by embedded sensors) not only in the BOL phase but in MOL and EOL stages as well (Främling and Nyman, 2009).

The PROMISE Messaging Interface (PMI) addresses the above challenges and requirements, and Fig. 8 outlines the PMI connectivity model. The PMI model is similar to that of the Internet itself. Where the Internet uses the HTTP protocol for transmitting HTML-coded information mainly intended for human users, PMI is used for transmitting XML-coded information mainly intended for automatic processing by information systems (see Fig. 9). This approach is similar to that employed by the EPCIS standard of the EPCglobal architecture but PMI has sought to provide data models to provide specific support for PLM data, such as sensor data. In contrast, EPCIS currently provides specific data models suitable for supply chain management operations, although these data models do provide the capability to support extensions.

PROMISE had to deal with a great number of application domains. PMI fulfils the requirements for a generic messaging protocol, which includes extension mechanisms for exploiting existing standards for product information semantics such as STEP (Standard for the Exchange of Product Model Data) (ISO, 1994) and PLCS (Produce Life Cycle Support — see http://www.plcs-resources.org/). PMI is currently proposed as a standard through the QLM Consortium of The Open Group (http://www.opengroup.org/qlm/).

Furthermore, the extensions to DIALOG also sought to address some deficiencies in meeting two key technical requirements, timely responses and synchronisation, as highlighted in Tables 6 and 7 (Främling and Nyman, 2009). These extensions allow DIALOG to meet the previous requirements in a manner similar to the EPCglobal approach (see Table 6).

Fig. 8. Illustration of PMI connectivity model (PROMISE, 2009a, 2009b).
Instance-level subscription mechanism introduced through PMI implementation provides an architecture capable of providing specific sensor readings, other information changes and events both through a call-back and a pull-based mechanism.

A degree of support has been added for data synchronisation mechanisms through two provisions within the PMI implementation. Firstly, unsent information updates can be stored for later transmission as is the case, for instance, for mobile products. This is based on DIALOG’s message persistence mechanism. Secondly, all messages handled by DIALOG (for instance events) have two timestamps, event generation time and event reception time at the current node. Timestamps and caching of events allow unordered events to be correctly sequenced at the node processing these events. These modifications allow DIALOG agents to correctly interpret state information based on a sequence of events from different sources. However, aspects related to the automatic and correct interpretations of a sequence of events, Unordered Event Stream (UnES) processing (Demers et al., 2007), is a significant challenge (Suzuki and Harrison, 2007), especially as some properties (e.g. modification level of an aircraft part) may be simply overwritten/updated, whereas others depend on a particular sequence of changes or an accumulation of values (e.g. total time in operation, total number of operational cycles).

However, using DIALOG with the ID@URI product instance identifiers, still does not address the issues of re-routing when URLs change and relatively long identifiers of the DIALOG system outlined in Section 6. Consequently, the questions of what product instance identifiers to use and how to lookup or discover information sources about those instances is still largely an open issue.

Finally, we propose the following key developments in order for the architectures to be adopted as PLIM systems.

**EPCglobal approach and WWAI approach:**

- Instance-level subscription mechanism introduced through PMI implementation provides an architecture capable of providing specific sensor readings, other information changes and events both through a call-back and a pull-based mechanism.
- A degree of support has been added for data synchronisation mechanisms through two provisions within the PMI implementation. Firstly, unsent information updates can be stored for later transmission as is the case, for instance, for mobile products. This is based on DIALOG’s message persistence mechanism. Secondly, all messages handled by DIALOG (for instance events) have two timestamps, event generation time and event reception time at the current node. Timestamps and caching of events allow unordered events to be correctly sequenced at the node processing these events. These modifications allow DIALOG agents to correctly interpret state information based on a sequence of events from different sources. However, aspects related to the automatic and correct interpretations of a sequence of events, Unordered Event Stream (UnES) processing (Demers et al., 2007), is a significant challenge (Suzuki and Harrison, 2007), especially as some properties (e.g. modification level of an aircraft part) may be simply overwritten/updated, whereas others depend on a particular sequence of changes or an accumulation of values (e.g. total time in operation, total number of operational cycles).

**DIALOG approach (after the integration of PMI):**

- Development of industry driven standards, in particular the PMI interface following on from the work carried out within the PROMISE project to provide, in particular, a standard for data exchange.
- Addressing the fragility of the UID which is generally a URL rather than an URN through the use of, for example, the handle mechanism and using URNs instead of URLs.
- The DIALOG system uses the existing DNS infrastructure more directly than the EPC network, which first requires a trivial manipulation of the EPC identifier into a hostname format before performing the DNS lookup. In contrast the Object Name Service supports multiple service types for a given EPC class, with standardised meta-data in the NAPTR records to distinguish between them. Therefore, in the DIALOG system, the URI would need to provide a machine-readable list of pointers and corresponding meta-data to allow a DIALOG software agent at a single URI to provide multiple service types, since the DIALOG system initially only resolves any tagged object to a single URI.

All three approaches would need to develop:

- Tools for supporting PLM application requirements such as the development of API’s for implementing, for instance, decision support systems to assist human operators to decide the fate of a product at its end of life (discard, recycle or re-manufacture).4
- Further support of data synchronisation mechanisms for achieving correct processing and interpretation of events arriving out of sequential order (unordered event stream processing) in the EPCglobal and DIALOG approaches. In the case of WWAI, developing mechanisms for data synchronisation since there are no methods supported in the present architecture.

---

3 At the time of compiling this article, a technical work group for development of Discovery Services technical standards has been chartered within the Software Action Group of GS1 EPCglobal.

4 It is likely that this requirement will remain outside of the scope of GS1 EPCglobal standards work for the foreseeable future since the standardisation work carried out by the GS1 EPCglobal community focuses primarily on developing data models, interfaces and protocols for the architecture/EPC Network infrastructure and does not usually prescribe the detailed functionality or logic of application software that makes use of the EPC Network.
Acknowledgements

This work was partly funded by the European Commission within the Sixth Framework Program (2002-2006) IP No. IST-FP6-033546. The authors would also like to thank the PROMISE project consortium as the success of the project as well as the results are an achievement of the entire group.

The authors would also like to thank all the members of the PROMISE consortium since the work in extending the DIALOG architecture and the development of the PROMISE demonstrators would not have been possible without their support and intellectual wealth.

References


intellectual wealth.

would not have been possible without their support and PIM consortium since the work in extending the DIALOG consortium as the success of the project as well as the results are

Acknowledgements

Berners-Lee T, Masinter L, McCahill M. Uniform Resource Locators (URL) / EPCglobal web site. Available online at: / EPCglobal web site. Available online at:

EPCglobal web site.


