International Journal of Computer Integrated Manufacturing, Vol. X, No. X, Month 2007, xxx, xxx

Review Article

Requirements on unique identifiers for managing product lifecycle information - comparison of alternative approaches

K. FRÄMLING*†, M. HARRISON‡, J. BRUSEY‡ and J. PETROW§

† Helsinki University of Technology, P.O.Box 5500, FI-02015 TKK, Finland
‡ Institute for Manufacturing, Cambridge University, Mill Lane, Cambridge CB2 1RX, United Kingdom
§ Trackway Oy, Salomonkatu 17 B, FI-00100 Helsinki, Finland

Correspondence *K. Främling. Email: Kary.Framling@hut.fi

Managing product information for product items during their whole lifetime is challenging, especially during their usage and end-of-life phases. The main difficulty is to maintain a communication link between the product item and its associated information as the product item moves over organizational borders and between different users. As network access will typically not be continuous during the whole product-item lifecycle, it is necessary to embed at least a globally unique product identifier (GUPI) that makes it possible to identify the product item anytime during its lifecycle. A GUPI also has to provide a linking mechanism to product information that may be stored in backend systems of different organizations. GUPIs are thereby a cornerstone for enabling the Internet of Things, where 'intelligent products' can communicate over the Internet. In this paper, we analyze and compare the three main currently known approaches for achieving such functionality, i.e. the EPC Network, DIALOG and WWAI.

Keywords: Internet of Things; Coding schemes; Communication protocols, Decentralized systems

AMS Subject Classification: 68M10; 68M12; 68M14; 94A99

International Journal of Computer Integrated Manufacturing
ISSN 0951-192X print/ISSN 1362-3052 online ª 2005 Taylor & Francis
http://www.tandf.co.uk/journals
DOI: 10.1080/09511920500069507

1 Introduction

The phrase Product Lifecycle Information Management (PLIM) is commonly understood to be a strategic approach that incorporates the management of data associated with products of a particular type, and perhaps the versions and variants of that product type, as well as the business processes that surround this (Stark, 2004; Ameri and Dutta, 2005; CIMdata, 2007). This product definition data is generated when the product is first conceived, and it then continues to evolve with the addition of detailed specifications, user manuals, CAD drawings, manufacturing instructions, service manuals, disposal and recycling instructions and so forth. For such traditional PLIM, the product information generation process seems to end after production. When the product enters actual use, PLIM mainly signifies providing access to the existing information but hardly any new information is generated about the products. This is, perhaps, a reflection of the point of view of the manufacturing industry that tends to see PLIM mainly as a distributed Knowledge Management task of the 'extended enterprise' that created the product (Ameri and Dutta, 2005). With this view of PLIM, there has been only slight interest in how the customer uses each individual product, or in how that product has behaved.

This view has been changing. With the trend towards manufacturers providing 'services' rather than 'products' (Mont, 2002; Auramo and Ala-Risku, 2005), it has become more important for such service providers to understand and track how each product is used and behaves to enable more intelligent maintenance regimes, e.g. predictive maintenance (Lee et al., 2004; Anke and Främling, 2005). Furthermore, as more responsibility for disposal of products at end-of-life (EOL) is placed on the manufacturer, such usage information will enable more efficient reuse and recycling of products and their components. PLIM business processes associated with the in-service

phase are intended to support feedback from use, maintenance, and dismantling procedures and this feedback can trigger improvements to products. Nonetheless, the inability to identify the product instance, as opposed to generic class of product, limits this process. Clearly, the PLIM life cycle needs to be extended to allow the tracking of what happens to individual products, not just during manufacture, but throughout their life.

When moving from the traditional 'product-type' view of PLIM to an item-specific view, the way that product information is stored and accessed changes radically. With product-type PLIM, product information is typically handled on a company or organisational level because they produce most of the product information. In product-item PLIM, a large amount of the product information is produced during the usage phase of the product, outside the organisations that designed or manufactured them. This means that product-item PLIM may deal with much larger volumes of data than product-type PLIM because each individual object is considered to have its own individual life history and the data associated with that history. Furthermore, the usage information might be collected via various organizations that perform repairs or maintenance on the product, some of whom may be unauthorized or unknown to the manufacturer of the product. In some cases, collection of usage information may even rely entirely on built-in sensors within the product to count the number of duty cycles, in-service hours, as well as monitoring temperature in case critical components are overheating or acceleration / shock / vibration, to detect whether the object was dropped.

It is of course necessary to be able to link the 'generic' product-type or class-level information with the individual usage information in an efficient manner. This can be achieved either by embedding the product-type identifier as part of the unique identifier

for the individual object – or alternatively, by providing a cross-reference within the information services to link the individual unique item identifier to the corresponding product-type identifier. The result of either of these approaches is that it is not necessary to duplicate the product-type information for each instance of an object of that product type. Instead, there is a logical linking between data repositories containing unique item-level lifecycle information and data repositories containing the generic product-type information, also known as 'master data'. The logical linking and ability to gather more complete data from multiple distributed data repositories is a fundamental feature of what is sometimes called the *Internet of Things*.

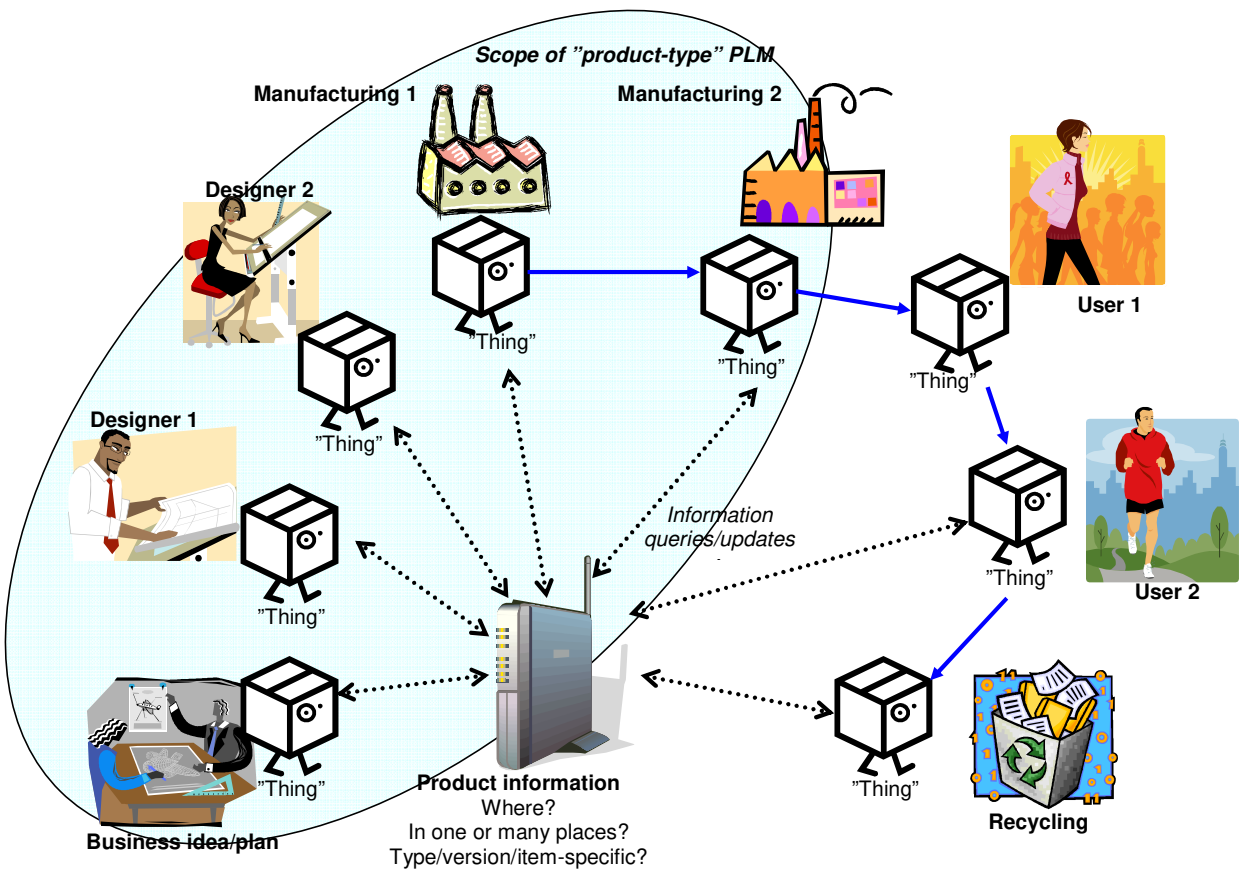

Figure 1. Product-item PLIM. In different phases of the lifecycle, the 'thing' may be an idea, a set of CAD drawings, sub-assemblies that are not yet assembled, the actual 'product item' or disassembled parts of the product item.

Therefore the collection and usage of product information becomes more challenging in product-item PLIM. For example, in some industry sectors (e.g. aircraft parts, heavy machinery), the usage or middle-of-life (MOL) phase may extend up to 30 years, during which time it is important to compile a complete record of maintenance events involving each individual part, for reasons of safety and warranty management, especially as each part may pass among multiple owners and custodians as well as being installed and removed from a number of aircraft or machines during its service lifetime and undergo a number of upgrades and repairs. Also, at the end-of-life (EOL) of the product, the service and maintenance history can be used to make more informed decisions about how to reuse or recycle product components and, for some products, disposal actions must be documented to ensure compliance with environmental regulations.

Product- or item-centric approaches to product information management offer a solution to product-item PLIM (Kärkkäinen, et al., 2003a, 2003b; Bajic and Chaxel, 2002; Chaxel, et al., 1999; Parlikad, et al., 2003). The concepts of *product agent* (Främling et al., 2003, 2006a) or *product avatar* (Hribernik et al., 2006) are the key elements for implementing the necessary product-centric information management. Product-centric information management is also closely related to the Internet of Things, where it would be possible to access information about any tangible 'thing' over the Internet as illustrated by figure 1. The Internet of Things concept is somewhat similar in purpose to a PLM repository, however it is unified (common standardized interfaces between multiple individual repositories), distributed (spread in a decentralised manner over a large number of organisations and computers) and supports decentralized ownership (no single body owns all of the data).

In figure 1 the 'thing' has been illustrated in the same way for the different phases of its lifecycle, which is an obvious simplification. During the design phase, the 'thing' is a collection of ideas, design documents etc. that may even be spread over several organizations. In the manufacturing phase, the 'thing' is a set of parts and subassemblies that may be manufactured by different organizations and that are 'things' already by themselves. The 'thing' that the consumer buys and/or uses is then the tangible result of all the previous phases and of its usage history. The corresponding product information obviously tends to be spread over different organizations, geographical locations and information systems. The product information may also be type-specific for some parts and sub-assemblies while it is item-specific for other parts. In this context, the following challenges arise:

1. It is usually impossible to store all product information with the product item itself, so portions of it need to be stored in 'backend' systems. Indeed, there are many good reasons to store data on the network rather than trying to store all the product information on the product item. These include cheaper, unlimited data storage, better security management of data and being able to retrieve data even when the product item is not physically present.
2. In order to associate product items with the correct product information in backend systems, every product item needs to be uniquely and globally identified among all other product items.
3. Product items usually change their location during their lifecycle, so they tend to have only intermittent network access (typically through Internet). When they have network access, they may need to access, modify or synchronize product information with the backend systems.

A key issue when addressing these challenges is how to create the link between the product-item itself and the associated product information. With many organizations involved, they all tend to use different identifiers or references as keys to accessing information from their information systems. These may include internal part numbers and catalogue numbers, identifiers of drawings and designs, as well as purchase orders, invoice numbers, shipping waybill numbers for tracing all the components that were involved in the assembly of the product. Use of a unique identifier for the object, together with mechanisms to obtain cross-references to other identifiers and references are therefore important for enabling the gathering of complete information. In order to be globally useful in an Internet of Things context, both the issuing organisation and the identifier are needed for globally unique product identification. In this paper, we analyse the pros and cons of the three currently known approaches to create and use *Globally Unique Product Identifiers* (GUPIs) (Främling et al., 2006b) for implementing the Internet of Things:

1. EPC Network approach, which defines standard interfaces for related information systems that associate the product item with serial-level product information in backend systems. The product item is uniquely identified by its EPC (Electronic Product Code), which is a flexible identifier framework that allows a number of existing product identifiers (such as the GTIN) to be embedded.
2. The ID@URI approach, which uses existing product identifiers (item-level or not) and explicitly expresses where product information can be accessed in backend systems.

3. World Wide Article Information (WWAI) approach. WWAI uses existing product-item identifiers and links to product information in backend systems through a peer-to-peer (P2P) based lookup mechanism.

The structure of this paper is as follows. In section 2 we analyse requirements and tradeoffs of product identifiers, in particular from a product-item PLIM point of view. Section 3 attempts to compare how well the three known approaches respond to these requirements in different situations. The final section presents our conclusions.

2 Requirements and tradeoffs of globally unique product identifiers

There are a variety of criteria by which we may qualitatively assess the value of a naming scheme and how appropriate it is for product-item identification. We believe that it is desirable for the naming scheme to be:

- Simple,
- Open,
- Long-lived,
- Standard,
- Extensible,
- Hierarchical,
- Providing some guarantee of uniqueness,
- Distributed,
- Allowing private numbering,
- Providing cost effective registration, and,
- Cost effective per item.

It is desirable for a naming scheme to be simple since complex ones will tend to be costly and difficult to implement. This simplicity will minimise the barriers for entry for software developers and systems integrators, as will the characteristic of being open. An

open naming scheme is desirable since ones that are proprietary or encumbered by restrictive patent licenses are less likely to be widely adopted. Also, open schemes will not restrict users to one particular software package or hardware platform but allow multiple and varied implementations.

GUPIs generated by the naming scheme should be long-lived and must last at least as long as the product that they are associated with and possibly longer. Therefore, the scheme should not encode transitory attributes into the identifier. It might be possible in some cases to access the product, and to update the GUPI, but in general we cannot assume this to be the case.

The naming scheme should be a standard one. A global naming scheme needs common acceptance. It is not enough for the scheme to be well specified and open; it must also be adopted widely. Of course, we must accept that to build consensus and to achieve standardisation requires a workforce and financial support. On the other hand, we should avoid creating new standards if old ones are sufficient. If new ones are required, some support for interchange of data with systems that use legacy standards is desirable.

Much effort will be applied to adopt any particular scheme, but even more effort will be required in the future if it becomes necessary to convert to some new scheme. Therefore it is important that the naming scheme is extensible and allows the set of possible unique names to grow.

A product-naming scheme that is hierarchical might allow the product type to be derived from its name directly, thus potentially simplifying some operations. Furthermore, a hierarchical structure may reduce the amount of duplication in the storage of information that is the same for a particular product type.

It is important for the naming scheme to provide some guarantee of the global uniqueness of the identifier. If the identifier were not unique, some other contextual information would be needed to fully identify the product. Although in some cases, context can be obtained, say through the position of the product, or from the order in which events are seen, it may not always be possible.

Although the easiest way to obtain unique identifiers might seem to be to centralise their naming and name resolution to network addresses, this would also be cumbersome. Rather, the scheme should distribute the resolution of network addresses in such a way that the failure of a node in the network should not disable name resolution nor product information lookup from other nodes. At the same time, it might sometimes be necessary to have private identifiers that are only intended for internal use. Preferably any private identifiers should also be identifiable as such.

The final two requirements have implications for cost effectiveness. First, any registration with a central body will add to the cost of using the scheme. In some cases, this cost may be small if the registration needs to be performed only once for a large range of identifiers. Second, the cost of identifying the item, whether it is via passive or active Radio Frequency Identification (RFID) tag, or simply the addition of a barcode, also increases the cost of using the scheme. In considering this cost, we must also consider how compact the naming scheme is, since tags that require less memory tend to be cheaper.

3 Analysis of relevant GUPI approaches

In this section, we first present three currently existing approaches for GUPIs and how they address the needs of PLIM. In the last sub-section, we attempt to analyse how well these approaches satisfy the requirements set out in section 2.

3.1 EPC Network approach

The Electronic Product Code (EPC) is one approach for creating references between product items and the product agent or backend information services. EPC identifiers are Uniform Resource Names (URNs, Moats, 1997) that uniquely name objects. The EPC URN naming scheme is a ratified published open standard, known as the 'EPC Tag Data Standard', which describes how a number of existing product identifiers may be formatted as a URN for use in the EPC Network. These existing product identifiers include serialized versions of the Global Trade Item Number (GTIN), which is related to the UPC-12 / EAN-13 barcodes already found on many products. When the EPC URN identifier is stored on passive RFID tags, a very compact binary format is used, requiring a minimum of only 64 or 96 bits of tag memory to store a wide range and large number of identifiers. This is achieved by not encoding the URN into binary as 8-bit bytes per character – but instead encoding various identifier fields as binary-encoded integers and replacing the URN prefix with a compact 8-bit 'header' code.

The Object Name Service (ONS) is the lookup mechanism used to obtain one or more URLs where authoritative information can be obtained for a given EPC. ONS is an extended implementation of the Domain Name System (DNS), using NAPTR (Naming Authority Pointer) DNS records. ONS provides a scaleable hierarchical lookup system, re-using existing DNS tools and protocols to perform the lookup. The root-level of ONS has been operational for over two years and resolves the Manager ID (usually points to the manufacturer of the product). The root-level is administered by EPCglobal Inc., and the operation of the root-level servers is currently subcontracted to Verisign Corporation. The second tier of ONS provides for resolution of different product classes within a company. It may be implemented using an in-house DNS name server. Entries

in the root-level ONS lookup system are currently only provided for subscribers of EPCglobal Inc.

ONS records provide not only a set of one or more URLs of information services, but also meta-data to indicate the type of information service provided by each URL in the set. This allows computer programs to automatically select between web pages, EPC information services, web services, XML data files and other services that may be added in the future, without needing to attempt to guess this from the URL pathname.

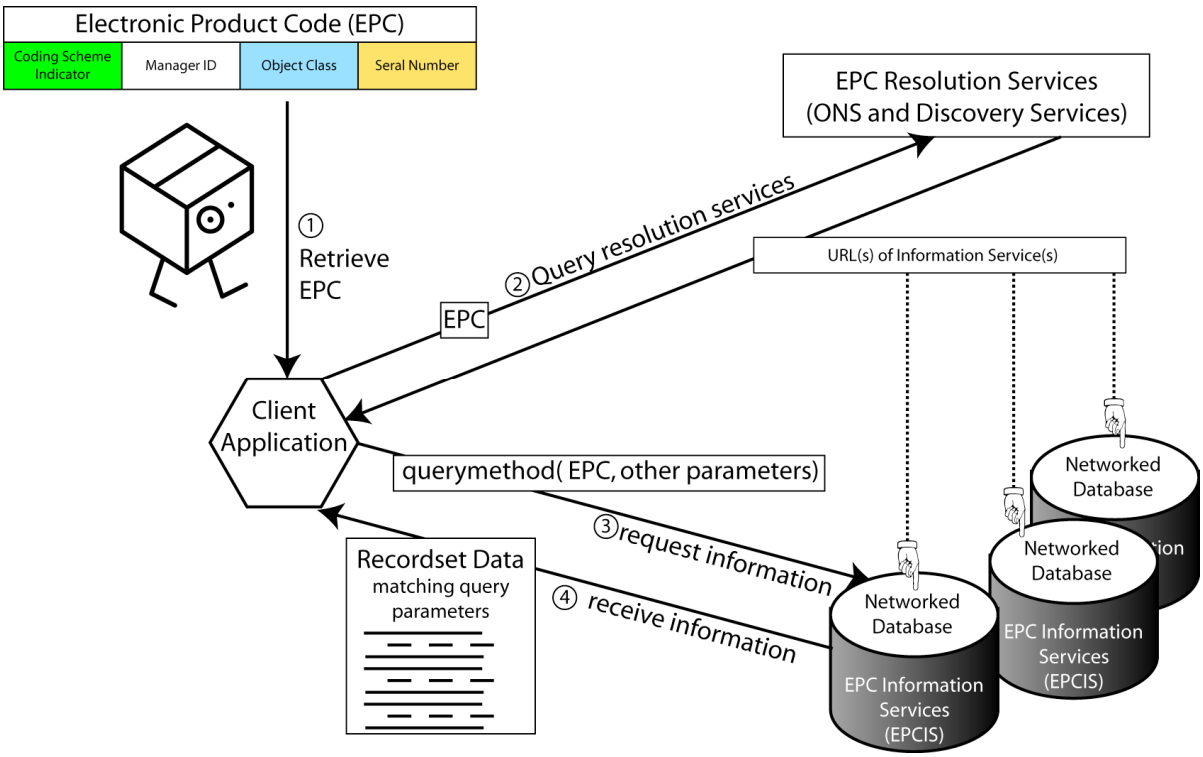

Figure 2. Product information lookup with EPC Network approach.

For Global Trade Item Numbers (GTIN), the Object Name Service provides records only at product class resolution – no serial-level resolution. The EPCglobal Architecture Framework Document (Traub et al., 2005) identifies 'Discovery Services' as a future component of the network that will provide for dynamic serial-level lookup across the entire supply chain, in a way that is both massively scaleable and secure. To date, EPCglobal have not yet chartered a work group to standardize Discovery Services,

although there are several technology vendors currently providing implementations of Discovery Services, albeit with no guarantee of a standard API among their implementations.

In the EPC Network, product item information may be accessed from various networked databases using a standardized interface framework, EPC Information Services (EPCIS) as illustrated in figure 2. The EPCIS specification was ratified in April 2007 and standardizes how client application programs may request current or historical data about EPC-tagged objects, together with higher-level semantic annotations such as the business steps and transactions associated with a particular observation event. EPCIS defines a modular framework for query and capture of such information, together with a standardized reporting format/schema and transport bindings to web services and existing electronic data interchange (EDI) technologies such as EDI INT AS2. This allows companies to be quite specific about which information they request – and also allows companies providing information to control in a very granular way the kinds of information they provide to others. Of particular note is the fact that the data model of EPC Information Services supports the inclusion of additional descriptive meta-data annotations per event, so that each event (i.e. the WHAT, WHEN, WHERE information) can be annotated with a business step (e.g. shipping/receiving, etc.) as well as the 'disposition' (i.e. state) of the object following the event. Furthermore, EPCIS aggregation events allow observations and changes to parent-child relationships to be explicitly recorded and retrieved in a similar way as explained in (Främling et al., 2003; 2006; 2007). In the PLIM context, this means that when components are embedded or removed from a complex product, we could expect to see this as explicit aggregation events within EPCIS and be able to directly query for

just such information in order to find out which parts had been added/removed/replaced from a product.

In addition to Tag Data Standards, ONS and EPC Information Services, the EPC Network intends to develop an end-to-end architecture of layered open standards, ranging from the air interface (reader-tag radio communication) all the way up to interfacing with existing business information systems. This also includes standardization of the Reader Protocol (software interface for reading/writing to tags), Reader Management (network monitoring of readers), Application Level Events (filtering, collection and reporting of observation events). EPC standards are developed through a community participation process involving end-users and technology providers. Following ratification, they are published by EPCglobal and freely available for download. To date, standards have been ratified for Tag Data Standards, Tag Data Translation, Object Name Service, Application Level Events, Reader Protocol, Reader Management and EPC Information Services.

3.2 ID@URI approach and DIALOG information system

In the DIALOG approach (Främling, 2002; Huvio, et al., 2002) an ID@URI notation has been used for creating a GUPI, where the ID part identifies the product item at the URI (Uniform Resource Identifier, Berners-Lee, et al., 1998)). If the URI is a Uniform Resource Locator (URL, Berners-Lee, et al., 1994), it is straightforward to link to a product agent or backend information services so no ONS-type approach is needed. The uniqueness of a URL is guaranteed by the DNS infrastructure. For an ID@URI to be a GUPI, the ID part should be unique for the corresponding URI. At the minimal level the ID@URI reference can be embedded as a barcode or using a passive RFID tag. In that case the URI should preferably remain the same during the product's entire lifecycle because changing it requires physical access to the product item itself. For more

intelligent devices, such as smart cards or car engine control units, this should not usually be an issue because they can update the URI themselves if needed. It is also possible to embed a list of alternative ID@URI references e.g. for ensuring access to backend systems even if some URI would not be operational. Since the URI part uses existing standards and since many possible standards exist for the ID part, this approach does not need any new identifier standards. Examples of usable standards for the ID are the EPC explained in the previous section, well-known industrial standards such as GTIN, SSCC GRAI etc., as well as ordinary serial numbers that are typically designed to be unique by product manufacturers.

Product information can be accessed through a middleware system called DIALOG as illustrated in Fig. 2. The DIALOG system is mainly used for testing and verifying new concepts and models for research purposes. It has also been used in two industrial pilots in a multi-enterprise setting in 2002 and in 2004 for tracking shipments in project deliveries (Kärkkäinen, et al., 2004). The current DIALOG implementation supports three protocols and data formats for message passing. Available protocols are:

SOAP (http://www.w3.org/TR/SOAP/). Programming language-independent protocol. Data is transferred as text using the XML notation.

HTTP-POST/HTML <FORM> (http://www.w3.org/TR/html401/). Programming language-independent protocol. Data is transferred as text using the HTML form format (can also be XML-encoded).

Java RMI (Sun Microsystems, 2002). Mainly used in development and in intra-company installations. RMI is flexible and easy to use, but firewalls and version management tend to be problematic.

The communication protocol to use can be specified in the 'scheme' part of the URI, e.g. *soap://server.comp.com/dialog*, *http://server.comp.com/html_dialog* or

rmi://server.comp.com/dialog, respectively. Only 'http' is documented as an 'official' URI scheme. However, the 'soap' and 'rmi' schemes are also used according to the URI specifications and to the recommendations for new schemes (Masinter et al., 1999). Supporting different communication protocols is technically simple. An average of about twenty lines of code has been needed to implement a new messaging protocol, which represents less than 1% of the total middleware implementation.

In addition to selecting standardised messaging protocols, a major challenge is to standardise the communication interfaces, specifically messages and their contents. For HTTP, the POST method is used, together with the HTML <FORM> notation for representing the data. With SOAP and RMI, only the communication protocol is defined but not the message semantics, i.e. the public interface (methods and parameters) that are used. The DIALOG software is distributed using an open source policy, which means that the message interfaces are publicly available. In practice, an open source solution is not sufficient for creating a standardised communication interface. This is why the DIALOG platform has an extension mechanism that allows other interfaces to be used based on the type of message being sent. This functionality is useful for supporting e.g. EPCIS, WWAI, ebXML (http://www.ebxml.org/) or other messaging interfaces. The Product Lifecycle Support (PLCS) initiative (http://www.oasis-open.org/committees/plcs/) could also provide a good communication interface standard, as well as other standardisation initiatives. Finally, it could be conceivable to use the address of a WSDL (Web Service Definition Language) file as the URI, which would contain the interface specification.

The semantic web (Berners-Lee, et al., 2001) community has also produced several standards for representing and communicating structured information that could be useful for implementing the Internet of Things. In DIALOG, the SOAP and RMI

interfaces are defined in a similar way to HTTP-POST and HTML <FORM>, i.e. with only one method and a generic and extensible object for the data. This approach offers similar extensibility and adaptability as the Internet itself (see e.g. Främling and Holmström, 2006c), where the combination of an object-oriented approach, semantic relations and design patterns are used for managing distributed product information as described in (Främling et al, 2007).

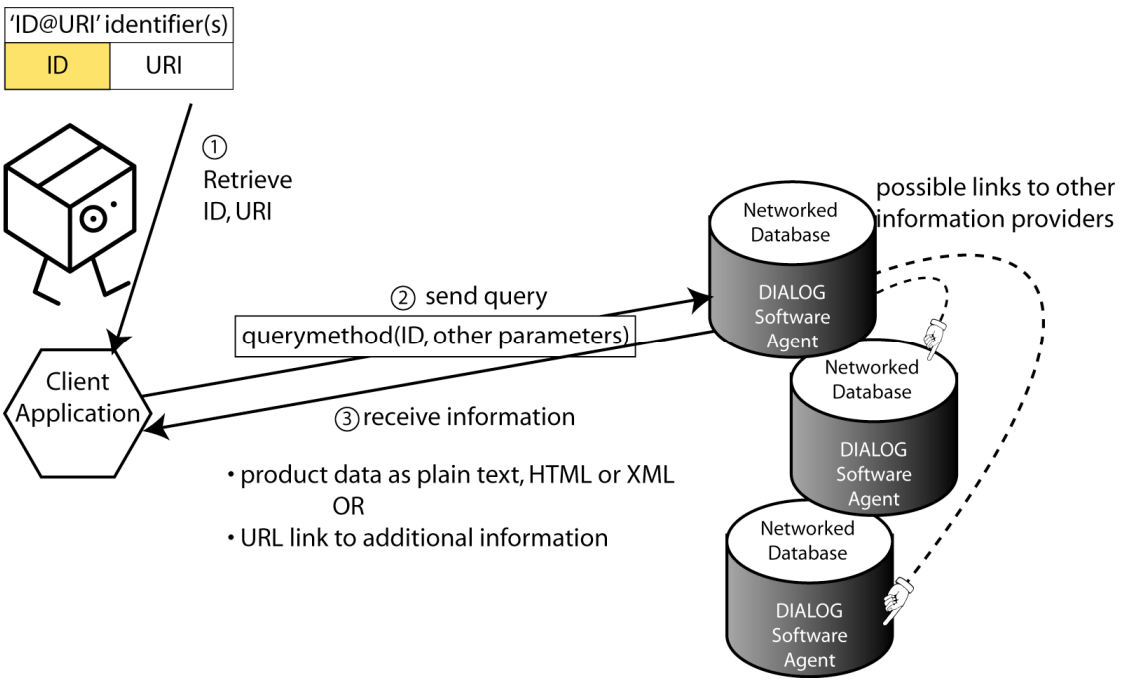

Figure 3. Product information lookup with ID@URI, DIALOG implementation. Possible links to other information providers are stored as semantic relations, e.g. '123@abc.com;is-part-of;321@cba.com' or '456@abc.com;contains;654@cba.com' that are stored e.g. at the moment of assembly. Such explicitly stored relations are an alternative or complementary solution to dynamic discovery mechanisms as in WWAI.

3.3 World Wide Article Information system

A different approach is offered by peer-to-peer (P2P) systems that are mainly known for file sharing of music and movies. However, P2P also has many desirable features for identifying nodes in the network as well as individual items. New nodes and items can be dynamically added at any time and are immediately integrated into the network. The network protocol usually takes care of assigning unique identifiers both for nodes and items automatically. Therefore there is no need for an external authority to manage

codes as in the EPC approach. Other advantages of P2P solutions are that all nodes can maintain complete control of what data is distributed to whom (even though most file sharing applications do not check or restrict who gets access), good fault-tolerance (breakdown of one node affects the whole network very little) and possibilities to do load-balancing by using nodes that are 'close' (in the network communication sense).

The World Wide Article Information (WWAI) protocol (www.wwai.org) developed by Trackway (formerly known as Stockway) is based on P2P principles. Existing company codes as issued by EAN/UCC or other standardisation bodies identify nodes of the network. When a node has joined the network, it can autonomously issue identifiers for individual items (e.g. product items). New nodes are dynamically discovered when appropriate. The WWAI protocol defines messages that enable nodes to exchange any kind of information and link any kinds of objects to each other by named relations. There are two types of networks supported by WWAI. (i) The first one being the global open network of authorized information providers. Joining this network requires certificates issued by a certification authority in order to become an official information provider in the global network. This is motivated by the need to find a compromise between existing coding standards and ensuring the uniqueness of the codes, as well as ensuring data integrity. On the positive side, certificates automatically guarantee the authenticity of the information provider. (ii) WWAI also allows building local private networks by using the pairing key mechanism instead of certificates. Essentially, this means using a known shared key as a token of trust between the partners. The pairing key approach eliminates the requirement to use official certificates, but as a drawback partners using the pairing key are not able to reach nodes outside of their local domain of trust. The scope of the domain is defined by the partners sharing the same pairing key.

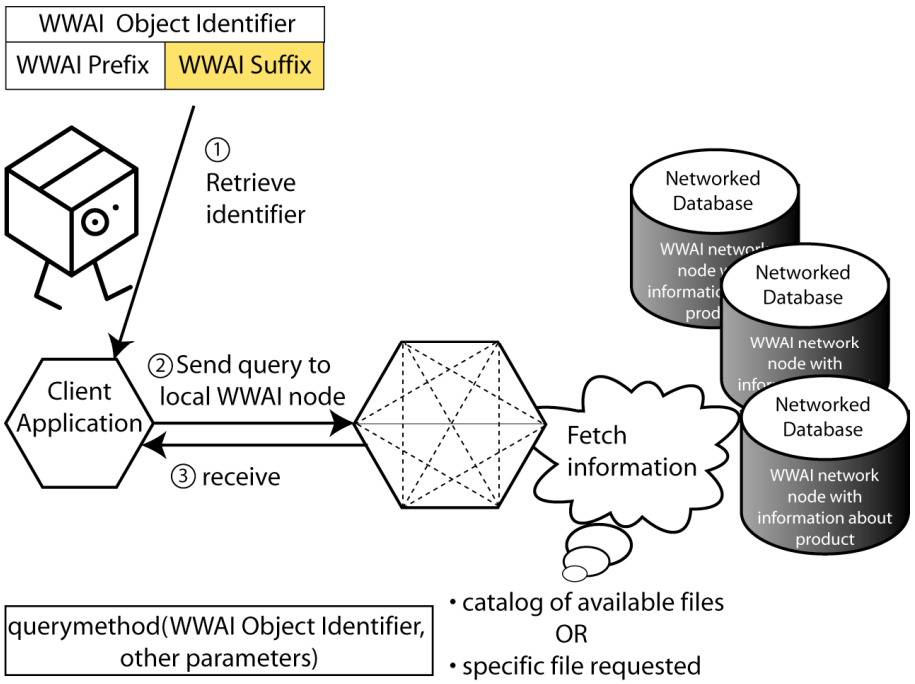

Figure 4. Product information lookup with WWAI approach.

A WWAI node lookup (Fig. 3) is usually only needed when a product identifier for a given company is seen for the first time. After that, network addresses of known nodes are cached so that new node lookups do not need to be performed unless the cached address fails or changes for some reason. In order to get access to other nodes in the network that have information about a specific product item (or some other item, e.g. a document, or a multimedia clip), it is sufficient to have one point of entry to the WWAI network. The network will automatically forward and route the request to all the relevant information providers in the network; the requester will then receive product data from all information providers according to privileges set. The information distribution model is asynchronous by nature which essentially leaves the requester the responsibility to interpret and classify the gathered information. However, synchronization can be achieved by notifying other registered nodes of changes in the data of a product, though this is usually feasible only in specific circumstances and for a specific set of data. Therefore, contrary to the EPC Network and DIALOG, there is no

need for a separate 'discovery' mechanism for accessing all information from all sources/organisations that have information about the product item.

WWAI uses one form of distributed hashing table approach (Naor and Wieder, 2003), even though files themselves are not distributed as in more common file sharing applications. Instead, the pieces of information related to a product and gathered during the life cycle of the product can be distributed in the network. While the product and its core data is created and owned by the manufacturer or brand owner, each additional node owns the information it has created about that product. Information in the WWAI network is found by a lookup mechanism based on the DHT overlay network (Andersen et al., 2001). Each node has its own globally or locally unique ID called a node prefix. These node prefixes form a logical ring topology where each node knows at least its neighbours. The ring with its node prefix space is partitioned (i.e. key-space partitioning) with a variant of consistent hashing (Karger et al., 1997), essentially providing the ability to add or remove nodes without affecting every other node in the network.

3.4 Evaluation of different approaches

Table 1 gives a comparison on how well the different approaches correspond to the requirements set out in section 2. The initial intention of the authors was to provide ratings to each approach, based on how it performs against each of the requirements. In practice, it turned out to be difficult to quantify and to find a common agreement on the ratings. Such ratings also tend to be subjective so a qualitative comparison was performed instead.

Table 1. Comparison of different GUPI approaches for the requirements presented in section 2.

Requirement	**EPC network**	**DIALOG**	**WWAI network**
Simple?	some complexity in converting EPC to network address	network address directly accessible	requires P2P network lookup

Open?	EPCglobal ratified standards are feely available online and build upon existing open standards, e.g. XML, XSD schema, web services	makes use of existing open standards	identifier structure is open, but for the moment supported by only two software products
Long-lived?	supports changing manufacturer's URLs without changing tag	URI on tag will need to change if manufacturer's address changes	WWAI address not tied to a specific URL
Standard?	ratified global standards designed to be agnostic to industry sector. Currently significant adoption by the consumer goods / retail sector	no standard of its own, but makes use of existing standard technologies	no standard of its own, but makes use of existing standard technologies
Extensible?	header provides a mechanism for extending the EPC code The identifier is decoupled from the network addresses	no limit to the number of bits in the ID. URI part could also use future network address resolving methods, e.g. ONS	ID structure allows extensions to support virtually any future coding scheme
Hierarchical?	usually includes manufacturer, and product type identifier parts	yes, if appropriate ID is selected (GTIN, EPC, other)	can include part type and item identification parts, supports variety of coding schemes such as GTIN, EPC, other
Guarantee of global uniqueness?	centrally allocated Manager ID, item-level uniqueness decided and controlled by individual organizations	URI globally unique, item-level uniqueness decided and controlled by individual organizations	Prefix uniqueness ensured by certificate authorization process, item-level uniqueness decided and controlled by individual organizations
Distributed name resolution, product info. lookup?	only one root ONS exists for the moment, may increase in the future	as distributed as DNS, information lookup is vulnerable to node failures	P2P-type name resolution, failure of one node doesn't affect others
Supports 'private' identifiers?	could have private identifiers (using a private / internal ONS) - but no header has yet been designated for 'private' identifiers	can have private identifiers	can have private identifiers
Registration cost?	as defined by EPCglobal cost of membership	DNS registration	Certificate cost from certificate authority
Item identification cost?	compact representation supports cheaper RFID tags	long identifier, needs more expensive RFID tags	identifier length ≥ EPC and ≤ ID@URI

There are many criteria that organizations across the lifecycle need to consider when choosing the appropriate technologies and identifiers for PLIM. One obvious factor is the cost and availability of the tags on individual objects and the infrastructure required for reading those objects and connecting to existing information systems. In the case of

RFID technology, for instance, mass-produced tags can significantly lower implementation costs per object – but may limit which identifiers can actually be encoded within their limited memory capacity. Additional criteria are the ease of linking to additional companies providing information for a specific object, as well as the ease of linking to legacy identifiers that are used as keys for retrieving information from existing databases. In this regard, both the degree of alignment with standard / legacy identifier systems and the 'registration' cost to a company of creating a new identifier are important factors to consider. Further criteria to consider are the extent to which architectures for cross-organizational information sharing are fully developed and implemented, in terms of standardization and availability; standardization is important because it is much more efficient for a large number organizations to exchange information if they all use a small number of agreed common protocols for information sharing rather than having to potentially develop N x (N-1) pair-wise translators; commoditized products certified as conforming to agreed protocols and interfaces, together with the emergence of open source software both lower the cost barriers to trialling and eventual deployment of technologies that enable the automation of PLIM.

The assessment in table 1 has been used for performing a further assessment of the different approaches, where a grouping has been performed according to some representative PLIM contexts. The grouping of PLIM contexts is mainly based on cost, size of manufacturing organisations and the embedded computational capacity of the products:

Cheap products identified with cheap passive RFID: The EPC was developed with this specific requirement in mind and allows using the most compact identifier. 2nd is WWAI, 3rd is ID@URI. A compact identifier can be endoded within a cheap mass-produced RFID tag that stores as

little as 96 bits of ID information. Larger identifiers will require more specialized tags with larger memory, usually resulting in increases of tag costs.

Computationally powerful products with intermittent or slow network access: ID@URI allows for the fastest connection to a backend system and potential URI changes can be handled by device itself. 2nd and 3rd place are shared by WWAI and EPC/ONS because additional resolution services may be needed to enable connection to the backend system.

Computationally powerful products with fast network access: 1st and 2nd are ID@URI (simplest to take into use) and WWAI (most fault-tolerant). 3rd comes EPC Network because it still requires standardization of Discovery Services and widespread adoption of ONS.

Barcode: ID@URI is probably the simplest to use (especially with low-range readers) because existing IDs can be used directly (even though the barcodes may get long depending on the URIs). 2nd and 3rd are WWAI and EPC/ONS.

Big manufacturing companies, strong IT competence: No technical preference but EPC Network may be preferred due to the strong support of commercial actors. However, the fact that a standardized query interface (EPCIS) now exists for selectively requesting events gathered by an organization makes it easier also for smaller solution providers and open source software projects to provide inter-operable solutions. Implementing EPCIS might indeed be at least as beneficial for implementations that use ID@URI or DHT technologies (e.g. WWAI) as it is for the EPC Network.

Small manufacturing companies (SME): ID@URI is both the cheapest and the simplest to take into use. 2nd is WWAI (about as cheap as ID@URI because no central registration is required – only certificate), 3rd is EPC Network due to EPCglobal subscription costs. However, a number of technology solution providers are now offering managed EPC Network services with full technical support for companies with limited IT resources. Furthermore, open source software is now also available for the EPC Network (see http://www.accada.org and http://epcis.mit.edu/).

Adaptability to new identifiers: ID@URI is 1st, because any organization can choose which identifier to use for the ID part, so long as the URI guarantees its global uniqueness (e.g. if the URI is a URL formed from the company's own domain name). 2nd is WWAI, because each organization can append any identifier behind its WWAI prefix. 3rd is EPC, because although many types of identifiers can be embedded within an EPC, the available identifier types are determined by EPCglobal. The flip-side to this is that for all the EPC identifier types in EPCglobal Tag Data Standards (see http://www.epcglobalinc.org/standards), there is a well-defined mapping to standard legacy identifiers already in use.

Adaptability to new software protocols: EPC Network is the most flexible solution because the identifier is always decoupled from the network address, so the resolution services (ONS and Discovery Services) can always support new protocols. ID@URI is 2nd, because the identifier is not always decoupled from the network address, so rewriting of tags may be necessary if the URI breaks. 3rd is WWAI because it uses its own

protocol, even though implementations of other protocols can be provided by WWAI nodes.

Fault-tolerance: WWAI is not subject to DNS vulnerabilities and provides distributed access to information. 2nd is EPC/ONS because several access points to information can be defined in ONS and Discovery Services. 3rd is ID@URI because it assumes one access point to information (even though the information itself can be distributed) and adding more access points requires using more memory (even though the required amount of additional memory is small).

Changing company names, URL of information access points etc.: WWAI handles such functions automatically. 2nd is EPC/ONS because changes can be performed through ordinary DNS functionality. 3rd is ID@URI because it requires physical access to product unless the URI can be remotely updated.

When reading this assessment, it should be remembered that many components of the different approaches can be combined. For instance, a DIALOG software component could perform an ONS lookup when it finds an EPC and no URI. An EPC Network component could also use an ID@URI approach after the first successful ONS lookup has been performed and continue doing so until the URI possibly fails and it has to get a new one. In the same way, a WWAI node could easily implement an ONS lookup that would be performed when needed. Still, even though GUPI compatibility can be achieved in this way, the three approaches are not compatible on the protocol and interface level so the access to product information is still not guaranteed. A deeper analysis of the protocol and interface level has intentionally been left out of the scope of this paper because the EPC Network standards on that level (i.e. EPC Information

Services) were not ratified at the time of writing and because other related standards also exist. Such a deeper analysis is therefore left as a subject of future research.

4 CONCLUSIONS

Based on the comparisons in section 3.4 it is not possible to identify any 'global winner' for PLIM applications. The EPC Network has three key strengths with respect to PLIM: First, it proposes standards that are supported by a world-wide standards body (GS1). Second, the lookup mechanism helps to insulate the data on the tag from needing to be changed. This is particularly important in a domain where the tag must last for the lifetime of the product and may only be accessible intermittently. Third, because it is becoming widespread, it may be the case that products have an EPC tag anyway, and that this tag can also be used for PLIM. Certainly if some other approach was used, it may be necessary to think about how to avoid any confusion with existing EPC systems. Nevertheless, the EPC Network was not initially designed with PLIM as the main priority, and some changes or extensions to the architecture may be required. Also, the registration cost may be too much of a barrier to entry for some users.

WWAI seems to be more technically sophisticated than the other approaches. For instance, it has built-in discovery services and authentication functionality that have been in use since 2002. The main challenge is that it has a small industrial support compared to the EPCglobal Network, so it may have difficulties to impose itself as a standard unless adopted by bigger players. However, WWAI supports EPC coding standards. Currently, there are software products from two software companies using WWAI: Trackway's Trackway product line and Stora Enso's PackAgent brand authentication software.

The DIALOG approach might be the most general-purpose one of the three because it places few restrictions on the format of the data on the tag. It is probably a good

solution for 'high-end' products with computing power and for smaller ad-hoc installations. Nevertheless, some steps may need to be taken to address the longevity of URLs used. It is also important to point out that the DIALOG software is developed mainly for purposes of research without commercial goals or support. However, the ID@URI concept is easily used independently of the DIALOG software. Because the DIALOG source code is published under the GNU Lesser General Public License it is also possible to use it for developing commercial applications.

Acknowledgements

This paper is based on an earlier version published as 'Främling, K., Harrison, M., Brusey, J., Globally unique product identifiers - requirements and solutions to product lifecycle management, in Proceedings of the 12th IFAC Symposium on Information Control Problems in Manufacturing, Saint-Etienne, France, 17-19 May 2006, pp. 811-816. ISBN: 978-0-08-044654-7'.

The authors wish to thank the anonymous reviewers for their insightful comments that have helped us to improve this paper.

References

Ameri, F. and Dutta, D., Product Lifecycle Management: Closing the Knowledge Loops. *Computer-Aided Design & Applications*, 2005, **2**(5), 577—590.

Andersen, D., Balakrishnan, H., Kaashoek, F., Morris, R. , Resilient Overlay Networks, in *Proceedings of 18th ACM Symposium on Operating Systems Principles (SOSP)*, October 2001, Banff, Canada.

Anke, J., Främling, K., Distributed Decision Support in a PLIM scenario, in *Proceedings of Product Data Technology Europe 14th Symposium*, 26-28 September 2005, Amsterdam, Netherlands, pp 129-137.

Auramo, J. and Ala-Risku, T, Challenges for going downstream. *International Journal of Logistics*, December 2005, **8**(4), 333—345.

Bajic, E. and Chaxel, F, Auto-ID mobile information system for vehicle life cycle data management, in *Proceedings of IEEE SMC 2002*, Tunisia, pp. 387-392.

Berners-Lee, T, Masinter, L., McCahill, M., Uniform Resource Locators (URL). December 1994, available online at: http://www.ietf.org/rfc/rfc1738.txt (accessed 23rd January 2007).

Berners-Lee, T., Fielding, R., Irvine, U.C. and Masinter, L., Uniform Resource Identifiers (URI): Generic Syntax. August 1998, available online at: http://www.ietf.org/rfc/rfc2396.txt (accessed 23rd January 2007).

Berners-Lee, T., Hendler, J. and Lassila, O., The Semantic Web. *Scientific American*, May 2001, **284**(5), 34—43.

Chaxel, F., Bajic, E. and Richard, J., Automotive vehicle data management based on holon-product paradigm, in *Proceedings of IEEE SMC 1999*, Tokyo, Japan, pp. 434-439.

CIMdata, Product Lifecycle Management (PLIM) Definition. Available online at: http://www.cimdata.com/PLM/plm.html (accessed 2nd February 2007).

Främling, K, Tracking of material flow by an Internet-based product data management system (in Finnish). *EDISTY magazine*, Tieke, Finland, 2002, No. 1.

Främling, Kary, Holmström, Jan, Ala-Risku, Timo, Kärkkäinen, Mikko, Product agents for handling information about physical objects. Report of Laboratory of Information Processing Science series B, TKO-B 153/03, Helsinki University of Technology, 2003.

Främling, K., Kärkkäinen, M., Ala-Risku, T., Holmström, J., Agent-based Model for Managing Composite Product Information. *Computers in Industry*, 2006, **57**(1), 2006, 72--81.

Främling, K., Harrison, M., Brusey, J., Globally unique product identifiers - requirements and solutions to product lifecycle management, in *Proceedings of 12th IFAC Symposium on Information Control Problems in Manufacturing (INCOM)*, 17-19 May 2006, Saint-Etienne, France, pp. 811-816.

Främling, K. and Holmström, J., How to create evolving information models by a layered information architecture, in *Proceedings of The Modern Information Technology in the Innovation Processes of the Industrial Enterprises (MITIP'2006)*, Budapest, Hungary, pp.173-178.

Främling, K., Ala-Risku, T., Kärkkäinen, M., Holmström, J., Design Patterns for Managing Product Lifecycle Information. In press for *Communications of the ACM*, 2007.

Huvio, E., Grönvall, J. and Främling, K., Tracking and tracing parcels using a distributed computing approach, in *Proceedings of the 14th Annual Conference for Nordic Researchers in Logistics (NOFOMA'2002)*, Trondheim, Norway, pp. 29-43.

Hribernik, K.A., Rabe, L., Thoben, K-D., Schumacher, J. The product avatar as a product-instance-centric information management concept, *International Journal of Product Lifecycle Management*, 2006, 1(4), 367—379.

Karger, D., Lehman, E., Leighton, T., Panigrahy, R., Levine, M., Lewin, D., Consistent hashing and random trees: distributed caching protocols for relieving hot spots on the World Wide Web, in *Proceedings of the 29th annual*

ACM symposium on Theory of computing, El Paso, Texas, United States, pp. 654-663.

Kärkkäinen, M., Främling, K. and Ala-Risku, T., Integrating material and information flows using a distributed peer-to-peer information system. In *Collaborative Systems for Production Management*, edited by H.S. Jagdev, J.C. Wortmann and H.J. Pels, pp. 305-319, 2003 (Kluwer Academic Publishers: Boston, USA).

Kärkkäinen, M., Ala-Risku, T. and Främling, K., The product centric approach: a solution to supply network information management problems? *Computers in Industry*, 2003, **52**(2), 147—159.

Kärkkäinen, M., Ala-Risku, T. and Främling, K., Efficient Tracking for Short-Term Multi-Company Networks. *International Journal of Physical Distribution and Logistics Management*, 2004, **34**(7), 545–564.

Lee, J., Qiu, H., Ni, J., Djurdjanovic, D., Infotronics Technologies and Predictive Tools for Next-Generation Maintenance Systems, in *Proceedings of the 11th IFAC Symposium on Information Control Problems in Manufacturing*, April 5-7th, 2004, Salvador, Brasil.

Masinter, L., Alvestrand, H., Zigmond, D., Petke, R., Guidelines for new URL Schemes. November 1999, available online at: http://www.ietf.org/rfc/rfc2718.txt (accessed 23rd January 2007).

Moats, R., URN Syntax. May 1997, available online at: http://www.ietf.org/rfc/rfc2141.txt (accessed 2nd February 2007).

Mont, O., Clarifying the Concept of Product-Service System. *Journal of Cleaner Production,* 2002, **10**(3), 237–245

Naor, M., Wieder, U., Novel Architectures for P2P Applications: the Continuous-Discrete Approach, in *Proceedings of ACM Symposium on Parallel Algorithms and Architectures (SPAA'03)*, June 7-9, 2003, San Diego, California, USA.

Parlikad, A.K., McFarlane, D.C., Fleisch, E. and Gross, S., The Role of Product Identity in End-of-Life Decision Making. Auto-ID Centre White Paper, 2003, available online at: http://www.autoidlabs.org/uploads/media/CAM-AUTOID-WH017.pdf (accessed 5th January 2007).

Stark, J, Product Lifecycle Management: 21st century Paradigm for Product Realisation, 2004 (Springer).

Sun Microsystems, RMI Specification. 2002, available online at: http://java.sun.com/j2se/1.3/docs/guide/rmi/ (accessed 6th January 2007).

Traub, K., Allgair, G., Barthel, H., Burstein, L., Garret, J., Hogan, B., Rodrigues, B., Sarma, S., Schmidt, J., Schramek, C., Stewart, R., Suen, K.K., EPCglobal Architecture Framework. July 2005, available online at: http://www.epcglobalinc.org/standards/Final-epcglobal-arch-20050701.pdf (accessed 5th January 2007).