From Tracking with RFID to Intelligent Products

Kary Främling
Helsinki University of Technology
BIT Research Centre
PO Box 5500, FI-02015 TKK, Finland
Kary.Framling@hut.fi

Jan Nyman
Posintra Oy
Electrical Building Services Centre
Kipinätie 1, FI-06150 Porvoo, Finland
jan.nyman@posintra.fi

Abstract

The Internet of Things notion is sometimes being employed to signify the use of Radio Frequency Identification (RFID) for tracking and tracing of products in supply chains. However, RFID is only one Auto-ID technology among others. The world is already full of products with embedded identification, processing power, sensors etc. that can do much more than ordinary RFID tags. When such products will have the means to communicate between themselves and with other systems we will eventually reach a limit where they could be called Intelligent Products.

Intelligent Products are capable of collecting information and reacting on it proactively, e.g. estimating needs for maintenance or repair. With increased computing power and communication capabilities, products may also become pro-active.

1. Introduction

The Internet of Things has been proposed as an extension to the Internet, where it would be possible to access information about any tangible "thing" over the Internet. The Internet of Things concept is explicitly mentioned e.g. in [3][14] and [13], but the name Internet of Things seems to have been used in different contexts already before these papers. Unfortunately, the Internet of Things concept is often interpreted in a very RFID- and Supply Chain Management (SCM) centric way as in [3] and as promoted by the EPCglobal organization (http://www.epcglobalinc.org/). This limited view of the Internet of Things concept is rather focused on product identification technologies, tracking of product locations and stock levels than on everyday objects. Because of the focus on one Auto-ID technology of many (RFID) and one specific application area (SCM), the information architecture and the interface standards created e.g. by EPCglobal tend to have limitations that make the name Internet of Things inappropriate for them. Such limitations are for instance:

- Hierarchical and uni-directional: data flows "upwards" only from RFID readers towards backend systems (as opposed to the device-to-device communication that is typical in Ubicomp systems).
- The identifier space is "closed" by the support for Electronic Product Codes (EPC) only, which are centrally managed by GS1. Other alternatives to the EPC are presented e.g. in [10].
- The focus on RFID tags and their price means that devices with embedded computing power are hardly considered in the architecture.

In order to make the Internet of Things become a similar ecosystem as Internet and the WWW, it would probably be useful to follow their design principles and learn from how they evolved into the current system. Within the domain of information modeling, a two-layered model has been proposed as a bridge between two views – that of the real world and the design of an information system [21]. A major challenge in information modeling is how to keep the models flexible enough to allow for future evolution and innovation of the information systems. Parsons and Wand [21] argue that most existing information modeling methods fail to be flexible enough because they are based on the assumption of inherent classification, i.e. that "things" can be referred to only as instances of classes. As a solution, they propose using a two-layered architecture to information modeling. The first layer represents instances with their properties, while the second layer consists of class definitions based on sets of properties defining classes of interest. An important property of the first layer is that all instances are unique and are universally identified. For the second layer there is no such restriction; there may be any number of class definitions that allow instances to be classified according to users, views, purposes etc.

In the ideal case, an information system should be flexible enough to provide a kind of ecosystem that needs no central control to expand, evolve and create initially unexpected "life forms". At Helsinki University of Technology, the DIALOG research team [6] has been developing information architectures for the Internet of Things. In this paper we will attempt to explain how the DIALOG system conforms to the two-layered architecture in such a way that it could allow for a similar ecosystem to evolve as the current Internet.

The structure of the paper is as follows. After this introduction, section 2 describes the principles of the two-layered architecture, how to create the link between physical products and information, as well as how to manage that information using Design Patterns. Section 3 attempts to give a signification to the Intelligent Product concept based on earlier work. Section 4 shows how some Intelligent Products have been implemented and what is their significance in the Internet of Things, followed by conclusions.

2. ID@URI: concept and applications

One of the main challenges with the Internet of Things is how to access the information that is not stored locally in the thing itself but is available over the Internet (Figure 1). In 2001, an ID@URI notation (Figure 2) and the associated DIALOG information system (http://dialog.hut.fi) were developed at Helsinki University of Technology. DIALOG made it possible to query and update product information about tangible things over the Internet. DIALOG implements the first layer of a two-layered architecture and was used in two multi-organizational pilot installations for shipment and product tracking and tracing.

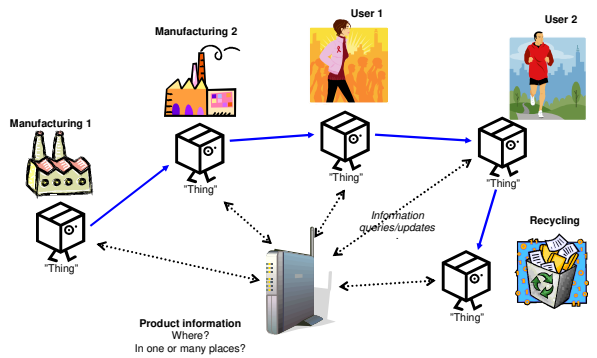

Figure 1. Internet of Things. The Thing is the unique instance with its properties, while the different users of that Thing have different views and interfaces to it.

Figure 2. ID@URI represented as 1-D barcode.

Alternatives to ID@URI and DIALOG are the EPCglobal initiative and the World-Wide Article Information (WWAI) protocol. What is common to all these approaches is that they use "globally unique product identifiers" [10], which is a much broader concept than e.g. record identifiers in databases or object identifiers in object-oriented programs, which only need to be unique within their respective information system or running environment.

2.1. Tracking and tracing

The need for tracking and tracing items along the supply chain has been long since recognized and logistic companies have therefore set out to offer tracking and data gathering services to solve the problem. The academic communities along with standardization organizations are also actively taking part in efforts to create global identification methods for items. The standards developed mainly concern identification of items and as such, do not directly define any connection to product tracking systems.

Companies such as Savi Technologies (www.savi.com) that have their business focus set on developing supply chain management systems on a global scale, usually build their systems around one server which functions as a central storage vault for all of the tracking data.

This centralized approach to item tracking can result in rather proprietary solutions often meaning that the companies only can track shipments that are managed by themselves. On the other hand, involving a third party in the data gathering process could also be considered a risk, not all companies are keen on allowing another company to handle their data. Larger scale companies that have built a tracking system of their own usually have a similar approach, with the same problems [5][2]. For instance they may not be able to share their information with their partners without restructuring the data to suite the individual formats and needs of each. In general the companies co-operating in the logistics chain must agree on certain pre-meditated means by which they exchange information. This makes the system inflexible for changes and expansion.

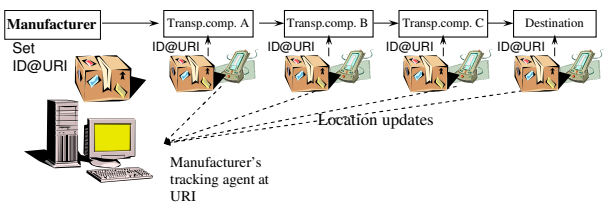

Figure 3. ID@URI based tracking, reproduced from (Främling et al., 2003).

The DIALOG research project was defined based on experience gained from earlier e-commerce projects where computer programs based on the peer-to-peer paradigm had been developed mainly for exchanging sales forecasts between different organizations. The initial application area of DIALOG was to develop a

forwarder independent tracking-and-tracing system for worldwide project deliveries (Figure 3). In order to create a globally unique product identifier that would at the same time indicate where information updates about shipments should be sent, the "ID@URI" was used, where URI is a computer address (e.g. 'www.some_company.com') and ID is a serial number or any other unique number at the URI indicated. A system using this notation was installed in 2002 for forwarder-independent tracking of project deliveries [20]. In this pilot, the ID was the unique serial number of the RFID tags used, while the URI part was written into the RFID tag's memory. Then location updates were sent to the projecting company whenever a shipment was observed at a tracking point, as illustrated by Figure 3. The same principle was used in another pilot performed in 2003 but with both ID and URI written as barcodes [20]. An extensive comparison between ID@URI and other alternatives can be found in [10].

The DIALOG information system was the initial model for the TraSer EU FP6 project (http://www.traser-project.eu/), where most research and development related to tracking and tracing has been continued since 2005.

2.2. Accessing product information

Since the beginning of the DIALOG project, it became clear that shipments are just a more transient variant of products or physical objects in general, while the location of an object is simply a property among others of the object. Therefore any property of any object could be updated or retrieved (if permitted by the security settings) using the same architecture (Figure 4).

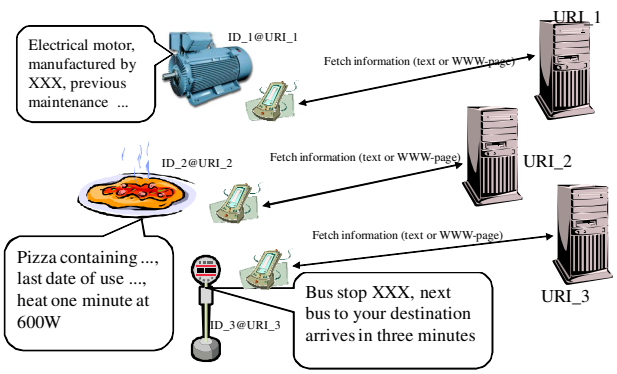

Figure 4. Accessing and updating information via ID@URI.

The product agent concept [7][8] was introduced as the virtual counterpart of the physical object that would enable the creation of Intelligent Products [17]. In the SCM domain, these intelligent products were the cornerstone behind the product-centric information management concept described in [18]. Identifying the parallel between object-oriented programming and product agents made it possible to apply Design Patterns [12] also to managing data about product items even when it is spread over organizational borders [9].

3. Intelligent Products

As pointed out in [11], the notion of intelligent product is still rather undecided and we don't currently have a single established model that holds up to all questions. A first question to answer is what we consider to be a product. Is it simply an item of commercial value – whether consumable or reusable (asset)? Or is it any "thing" that may have information and some kind of intelligence associated with it, even though the thing does not have any commercial value as such? Can things such as humans, pets, design documents, multimedia files etc. also be intelligent products? From an information architecture point of view, all such things can in principle be handled in similar ways – as long as the information architecture does not impose restrictions on which things can be intelligent products and which ones can't.

Manufacturers today are under increased pressure to add value to their product offerings in order to counter the relentless cost pressure applied via low cost production in emerging countries. Two common approaches have been to either seek to add greater value to products by customizing them to better meet customer needs (e.g. automobiles, furniture) or to enhance the way products are used by providing them with associated information offerings (e.g. mobile phones, laser printers). In each case, a specialization of the product occurs – whether it be in terms of different functionalities, different delivery paths, different usage modes – making it increasingly important to maintain information unique to each item in some way. Product individualization also occurs as soon as the product is sold and being used. Even two initially identical product items will have different owners or be used in different conditions. The product usage phase may even begin before it is sold because many products nowadays go through an individual initialization procedure where "normal" operation profiles are recorded and stored. The product's control system can also be fine-tuned so that individual differences in sensor outputs and actuator operations are taken into account, as for engine control in modern cars that continues adapting the control system to changes in the engine during its whole lifetime.

Apart from the societal trends mentioned above, it is clear that developments in ICT have made the notion of gathering, storing and associating information with a particular item increasingly feasible. Information systems developments such as wireless information gathering (RFID, WSNs, RTLS), embedded processing, distributed data management, distributed information search and retrieval mechanisms are nowadays available. Artificial intelligence methods such as reinforcement

learning and software agents are increasingly providing mechanisms for enabling the decision making of inanimate objects. Finally, the ubiquity of the internet provides a readily available mechanism for enabling the connection between an object and any information retained about it – be it in one or many locations.

In the commercial area, the advent of technologies such as low cost RFID over the last 5 years has provided a stimulus for automated product identification and tracking in the commercial sector, and has led to a range of models for associating product information with physical goods – whether the information is physically attached to the product (e.g. on a so called Product Embedded Information Device - PEID) [15] or held elsewhere on a network. Other work has focused on the challenge of enabling products to make their own decisions - or more realistically to provide support for decisions associated with the product during transportation or usage. This work has begun to explore the practical issues in associating forms of intelligence with physical products rather than developing a unifying vision for an intelligent product.

Other concepts related to intelligent products are e.g. holons [4], product avatars [14] and product agents [7]. A detailed study of intelligent products compared to those concepts can be found e.g. in [11].

4. Implementation of the Internet of Things

As a partner of the PROMISE project [22] that started in 2004, it was a challenge to see to what extent the DIALOG architecture would also be suitable for Product Lifecycle Management (PLM) in a context where data needed to be collected from many kinds of product items during their whole lifetime. This could even imply the collection of data when a product item was used by consumers as in the refrigerator and car applications of PROMISE. Since the initial system architecture ideas described e.g. in [1], the PROMISE system architecture has gradually evolved into the one illustrated in Figure 5. This architecture uses a peer-to-peer information exchange model, where any device that implements the Web Service-based PROMISE Messaging Interface (PMI) can communicate with any other device that supports PMI, no matter the size of the device. If the Product Embedded Information Device (PEID) does not have enough computational power or communication capabilities for implementing PMI, then they connect either through a device-specific Device Controller or using the UPnP-based CorePAC interface defined in PROMISE. Otherwise it is called a PEID:4 according to a classification based on computation and communication capabilities that is documented in PROMISE deliverable "DR5.4: Generic PEID roadmap for each group".

The PMI is a key interface which enables a web-services based approach, permitting any PMI-enabled user to exchange data with another. Depending on the complexity of any specific application, this can be achieved on a simple peer-to-peer basis if the two users are known to each other, or on a more complex wide-area basis.

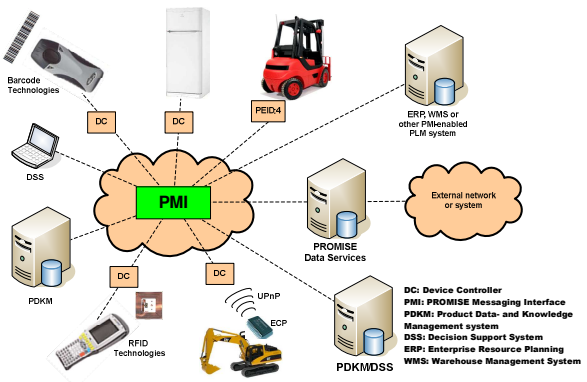

Figure 5. Illustration of PROMISE architecture and connectivity (PROMISE, 2008).

The PROMISE connectivity 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, PROMISE uses PMI for transmitting XML-coded information mainly intended for automatic processing by information systems (Figure 6). It is important to understand these relationships because PROMISE in effect proposes an extension to the Internet itself.

```
<?xml version="1.0" encoding="UTF-8"?>
<pmiEnvelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="../pmiSchema.xsd"
type="readData" version="0.8">
<readRequest ttl="1000" interval="-1"
wsCallBack="http://dialog.hut.fi/pmi/services/pmi" requestTargetType="device">
        <targetDevices>
                <targetDevice>
                        <ids>
                                <id>Refrigerator_001</id>
                        </ids>
                        <infoItems>
                                <infoItem>
                                        <id>freezerDoorWarning</id>
                                        <id>powerConsumption</id>
                                </infoItem>
                        </infoItems>
                </targetDevice>
        </targetDevices>
</readRequest>
</pmiEnvelope>
```

Figure 6. Example PMI subscription message.

The next section shows how DIALOG has been used as an implementation platform for PMI and the PROMISE architecture in general. As the original DIALOG architecture and the PROMISE architecture are nearly identical, this implementation task mainly consisted in adding a new networking component that uses PMI instead of using one of the existing DIALOG networking methods. Because DIALOG was already from the beginning designed to support different protocols and interfaces, adding a new one for PMI was rather straightforward.

Today, many consumer products have an embedded data processing system, which controls various functions of the product. A good example is the computer system embedded in modern cars, which monitors the various subsystems, provides the user with reminders about scheduled maintenance visits, and notifies the owner of possible error conditions by the "Malfunction Indicator Light" that is usually labelled "Check Engine" on the car dashboard. Such product-embedded information devices are also starting to appear in ordinary household appliances. In this section we look at how such an appliance is integrated with DIALOG, and how an installation in a real building or home would be configured. We also take a look at how vehicle diagnostics could be transmitted using PMI.

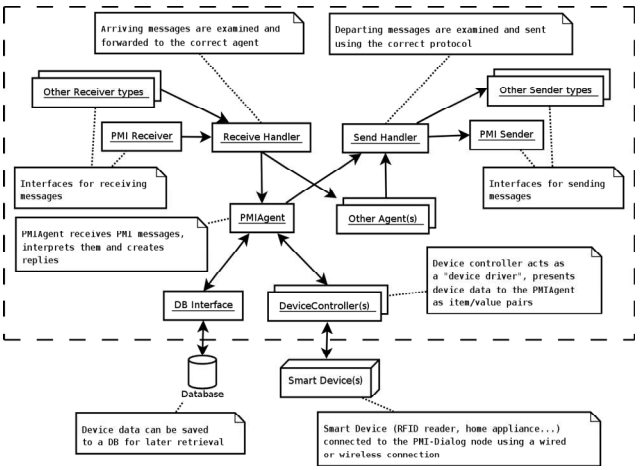

Figure 7. Internal architecture of a DIALOG node.

DIALOG is a "generic" software in the sense that it provides protocol- and interface-neutral message passing mechanisms with message persistence functionality, security mechanisms etc. that are abstracted away from the "business logic" itself, implemented by "agents". Figure 7 illustrates the internal architecture of a DIALOG node. We see that the components involved in sending and receiving messages, and agents, which consume and produce messages, are separated as their own classes with a common interface, i.e. the receive and send handlers. This signifies that different protocols and messaging interfaces can be easily supported. Already before PROMISE, DIALOG supported a Java remote method invocation (RMI) interface, a Web Service interface using the Simple Object Access Protocol (SOAP) and an interface using HTTP POST messages. For implementing the PMI, it was sufficient to add new SOAP-based PMI receiver and sender classes. A simple and configurable mapping mechanism that is internal to the DIALOG node defines what messages should go to which agent(s) and what sender should be used for which messages.

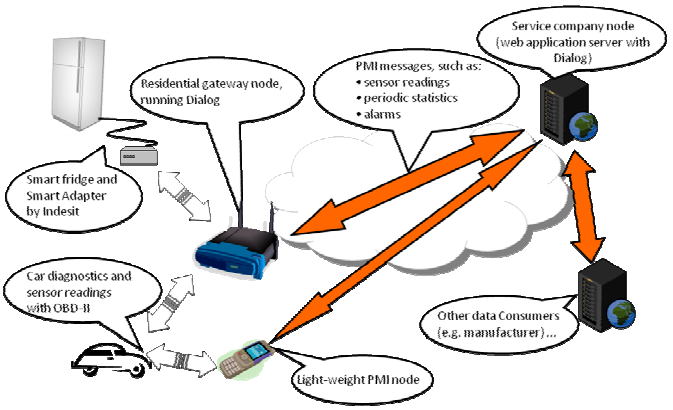

Figure 8. Residential gateway (or alternatively, a mobile phone) acts as a message interface that enables simple product-embedded information devices to participate in PMI communications over the Internet.

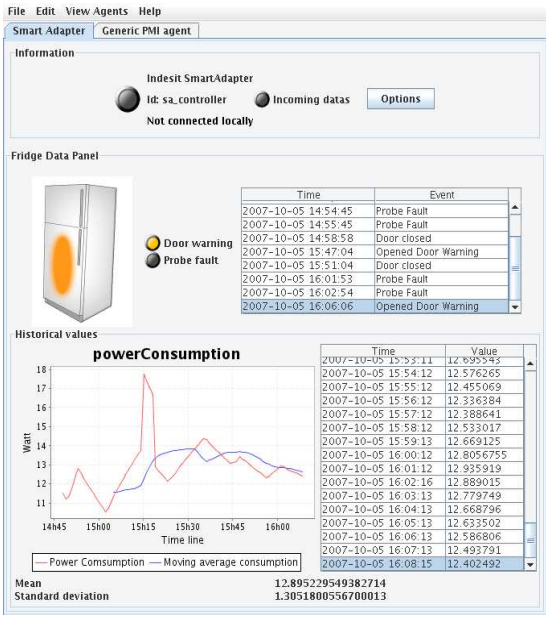

Figure 9. User interface showing collected real-time power consumption and events.

DIALOG agents are free to process the messages as they like, and may send messages at will. DIALOG was adapted to support PMI-specific functionality by the addition of a PMI-specific agent, which mainly means implementing Device Controller (DC) functionality. Two demonstrator implementations were developed together with industrial partners of the PROMISE project: an intelligent refrigerator (with a European white goods manufacturer) and connectivity to on-board computers of vehicles (with a European car and truck manufacturer) (Figure 8). As one of the PROMISE

demonstrators, support for an intelligent refrigerator control system was added to DIALOG by adding a corresponding DC that enables statistical data to be obtained from the refrigerator and sent to a remote location using PMI (Figure 9).

The statistical data obtained from the appliances installed at the customer's premises can be used to detect service needs in advance, before a failure occurs (condition-based maintenance), thus offering improved service for the customer. It could also be possible to determine in advance which spare parts are needed for the job and improve the scheduling of service personnel. This is an example of how a refrigerator product-selling activity could gradually change into a service-selling activity, i.e. selling "refrigeration services" rather than selling the physical refrigerator itself.

Figure 10. Preliminary user interface on mobile phone for accessing information from car ECU.

A specific Device Controller is needed for interfacing with the refrigerator due to the proprietary protocol used by the refrigerator. Other possibilities could have been to implement the PMI on the refrigerator itself or to implement the more light-weight UPnP-based CorePAC protocol on it. However, both options were excluded because they would have increased the cost of the system too much to remain commercially defendable. Therefore, at least in the near future, such household appliance's PEIDs are likely to implement a simpler protocol for communicating data to a PMI node that functions as the endpoint for PMI communications. The PMI node could take the form of a residential gateway, similar to the broadband routers on the market today, but equipped with PMI- and DC-implementing software.

The PEID is usually involved in controlling functional aspects of the product, such as engine control in a car, or climate control of a refrigerator. Another important functionality that PEIDs provide is diagnostics. In a state-of-the-art car, the diagnostics notifications are only for the driver to notice and act upon. The effectiveness of on-board diagnostics could be enhanced by enabling the notifications to pass to the service company directly, for instance via an ordinary mobile phone with suitable software. We have implemented such a system on a Nokia Series 60 mobile phone by a Java MIDP program capable of acting as a node that sends PMI messages over the mobile network with data downloaded from the car's Engine Control Unit (ECU) (Figure 10). The connection between the mobile phone and the car's ECU was implemented using a commercially available OBD-II protocol converter connected to the mobile phone via Bluetooth. The setup enables diagnostics notifications from the car to be sent to a remote node in real time. The remote monitoring node can in turn place a request for the PMI client in the mobile phone to send specific sensor values periodically to the remote monitoring node, to aid in further problem determination, scheduling a time for service, ordering needed spare parts or taking some other proactive actions. The collected information could also potentially be transmitted to the car manufacturer. If the car manufacturer could collect such real-use information from a sufficient amount of cars, it could lead to improved maintenance scheduling, product design and manufacturing procedures. This is true also for most other manufacturing companies (such as the refrigerator manufacturer), i.e. product design, manufacturing and possibly also recycling might be improved if a sufficient amount of in-use information could be collected.

In addition to the previously described implementations, DIALOG with PMI has also been used for tracking and tracing of assets in hospitals. In that case, DIALOG was used as middleware for creating a connection between RFID tags, sensors, button panels etc. and an existing surveillance system, which could also transmit information further to other systems using PMI. Implementation work is also ongoing e.g. for integrating water taps made by the company Oras (www.oras.fi) as well as other "Smart Home" appliances.

5. Conclusions

As shown by earlier industrial pilots in shipment tracking and tracing and other real-life implementations performed in the PROMISE project, the proposed information architecture can fulfill the needs of Intelligent Products in the Internet of Things. PMI and other PROMISE technologies are currently being standardized with the Open Group. Even though we are not currently aware of other architectures with the same

scope, partially similar but more domain-specific approaches exist such as oBIX (Open Building Information Xchange) for "intelligent buildings" or the EPC Network for SCM. Time will show what the final architecture will be called and what standards will become predominant but at least the building blocks now exist for implementing real-life Internet of Things applications. Such applications would open new market opportunities for manufacturing companies, service providers and many other commercial actors so the economical impact on society will be significant.

Acknowledgements

The work reported here has mainly been funded by Tekes (Finnish Funding Agency for Technology and Innovation) through the DIALOG and EloCore research projects and by the EU 6th Framework Program through the PROMISE and TraSer projects. The industrial partners of the DIALOG project have also contributed financially, while the industrial partners of DIALOG, PROMISE and TraSer projects have provided useful insight into real-world applications, as well as the possibility to perform pilot installations with access to their equipment, infrastructure and professional experience.

The PROMISE architecture and the PMI are a joint result of PROMISE partners, notably Trackway (especially Björn Forss and Jouni Petrow), Indyon (David Potter), SAP (several participants), InMediasP (notably Michael Marquard) BIBA (notably Robertino Solanas) and the authors of this paper. The authors hope they have not forgotten any key contributors but if we have done so, we present our excuses in advance.

In addition to the authors, especially Vincent Michel from Ecole Nationale Supérieure des Mines de Saint-Etienne, France, has done a major programming effort on the PMI implementation in DIALOG. The authors would like to thank him and other contributors to DIALOG for their effort.

References

[1] Anke, J., Främling, K. (2005), "Distributed Decision Support in a PLM scenario", In: Proceedings of Product Data Technology Europe 14th Symposium, 26-28 September 2005, Amsterdam, Netherlands, pp 129-137.

[2] Booker, E., (1999), "Service Maps To Needs – DHL Lets Customers Dictate Which Services The Company Delivers", Internetweek, July 12 1999, p. 17.

[3] Brock, D. L. (2001), The electronic product code (EPC)-a naming scheme, Technical Report MIT-AUTOIDWH-002, MIT Auto-Id Center, Massachusetts Institute of Technology, Available from http://www.autoidlabs.org/uploads/media/MIT-AUTOID-WH-002.pdf, accessed May 20, 2009.

[4] van Brussel, H., Bongaerts, L., Wyns, J., Valckenaers, P., van Ginderachter, T. (1999), "A Conceptual Framework for Holonic Manufacturing: Identification of Manufacturing Holons", Journal of Manufacturing Systems, Vol. 18, No. 1, pp. 35-52.

[5] Coia, A. (2001), "Express services unite Europe", Logistics Management and Distribution Report, Vol. 40, No. 9.

[6] DIALOG (2001), Distributed information architectures for collaborative logistics, Available from http://dialog.hut.fi/, accessed May 12, 2009.

[7] Främling, K., Holmström, J., Ala-Risku, T., Kärkkäinen, M. (2003), Product agents for handling information about physical objects, Technical Report of Laboratory of Information Processing Science series B, TKO-B 153/03, Helsinki University of Technology, 2003.

[8] Främling, K., Kärkkäinen, M., Ala-Risku, T., Holmström, J. (2006), "Agent-based Model for Managing Composite Product Information", Computers in Industry, Vol. 57 No. 1, pp. 72-81.

[9] Främling, K., Ala-Risku, T., Kärkkäinen, M., Holmström, J. (2007) "Design Patterns for Managing Product Life Cycle Information", Communications of the ACM, Vol. 50 No. 6, pp. 75-79.

[10] Främling, K., Harrison, M., Brusey, J., Petrow, J. (2007), "Requirements on unique identifiers for managing product lifecycle information - comparison of alternative approaches", International Journal of Computer Integrated Manufacturing , Vol. 20 Issue 7, pp. 715-726.

[11] Främling, Kary, McFarlane, Duncan. Editorial for Special Issue on Intelligent Products. Computers in Industry , Volume 60, Issue 3 April 2009. pp. 135-136.

[12] Gamma, E., Helm, R., Johnson, R., Vlissides, J. (1995), Design Patterns: elements of reusable object-oriented software, Addison-Wesley Publishing Company, Reading, Massachusetts.

[13] Gershenfeld, N., Krikorian, R., Cohen, D. (2004), "The Internet of Things", Scientific American, Vol. 291 No. 4, pp.76–81.

[14] Hribernik, K.A., Rabe, L., Thoben, K-D., Schumacher J. (2006), "The product avatar as a product-instance-centric information management concept", International Journal of Product Lifecycle Management, Vol. 1, No.4, pp. 367-379.

[15] Huvio, E., Grönvall, J., Främling, K (2002), "Tracking and tracing parcels using a distributed computing Approach", in: Solem, O. (Ed.), Proceedings of the 14th Annual Conference for Nordic Researchers in Logistics (NOFOMA'2002), Trondheim, Norway, 12-14 June 2002, pp. 29-43.

[16] Kiritsis, D., Bufardi, A., Xirouchakis, P.C (2003), "Research issues on product lifecycle management and information tracking using smart embedded systems", Advanced Engineering Informatics, Vol. 17 No. 3-4, pp. 189-202.

[17] Kärkkäinen, M., Holmström, J., Främling, K., Artto, K. (2003), "Intelligent products - a step towards a more

effective project delivery chain", Computers in Industry, Vol. 50 No. 2, pp. 141-151.

[18] Kärkkäinen, M., Ala-Risku, T., Främling, K. (2003b), "The product centric approach: a solution to supply network information management problems?", Computers in Industry, Vol. 52 No. 2, pp. 147-159.

[19] Kärkkäinen, M., Ala-Risku, T., Främling, K. (2004), "Efficient Tracking for Short-Term Multi-Company Networks", Int. J. of Physical Distribution and Logistics Management, Vol. 34 No. 7, pp. 545-564.

[20] Kärkkäinen, M., Ala-Risku, T., Främling, K., Collin, J. (2005), "Establishing inventory transparency to temporary storage locations", in: Proceedings of Advances in Production Management Systems (APMS), 18-21 September 2005, Washington, USA.

[21] J. Parsons, Y. Wand, Y., "Emancipating Instances from the Tyranny of Classes in Information Modeling", *ACM Transactions on Database Systems*, Vol. 25/2, pp. 228-268, 2000.

[22] PROMISE (2004), Product Lifecycle Management and Information Tracking using Smart Embedded Systems, Available from http://www.promise-plm.com/ and http://www.promise.no/, accessed March 20, 2008.

[23] PROMISE (2008), PROMISE Architecture Series Volume 1: Architecture Overview, accessed May 12th 2009: http://cl2m.com/system/files/private/PROMISE AS Volume 3 Architecture Reference PMI.pdf (may require registration to cl2m web site).