DI1.6: Evaluation of PROMISE Standards

Damith C. Ranasinghe, Cambridge
Mark Harrison, Cambridge
Kary Framling, HUT

DELIVERABLE NO	DI1.6: Evaluation of PROMISE standards
DISSEMINATION LEVEL	**PUBLIC**
DATE	02. December 2006
WORK PACKAGE NO	WP I1: Standardisation
VERSION NO.	0.1
ELECTRONIC FILE CODE	PROMISE DI1.6 ArchitectureComparisionVer0.7.doc
CONTRACT NO	507100 PROMISE A Project of the 6th Framework Programme Information Society Technologies (IST)
ABSTRACT	This report is the second of a three-part series of deliverables. The main objective is to compare prevalent information management architectures to the evolving PROMISE architecture to evaluate and to aid the refinement of PROMISE architecture specifications. The report also aims to provide useful feedback to improve the development of the PROMISE architecture.

STATUS OF DELIVERABLE		
ACTION	BY	DATE (dd.mm.yyyy)
SUBMITTED **(author(s))**	**Damith C. Ranasinghe**	**06.05.2007**
VU **(WP Leader)**	**Ajith Parlikad**	**11.12.2007**
APPROVED **(QIM)**	**Kiritsis Dimitris**	**18.12.2007**

Revision History

Date (dd.mm.yyyy)	Version	Author	Comments
09.02.2007	0.1	Damith C. Ranasinghe	Draft
10.25.2007	0.2	Mark Harrison	Review draft
11.12.2007	0.3	Damith C. Ranasinghe	Update version 0.2
12.12.2007	0.3	Damith C. Ranasinghe	Submitted for quality review
12.12.2007	0.4	Mark Harrison	Updated Table 11 and Table 13
18.12.2007	0.5	Kary Främling	Review version 0.3
18.12.2007	0.6	Mark Harrison	Review version 0.4
20.12.2007	0.6	Damith C. Ranasinghe	Incorporate changes
21.12.2007	0.7	Damith C. Ranasinghe	Version for submission

Author(s)' contact information

Name	Organisation	E-mail	Tel	Fax
Damith C. Ranasinghe	Cambridge University	aknp2@cam.ac.uk	+44 1223 764620	+44 1223 765597
Mark Harrison	Cambridge University	mgh12@cam.ac.uk		
Kary Framling	HUT	Kary.Framling@hut.fi	+358 50 5980451	

Table of Contents

1 INTRODUCTION 6
1.1 PURPOSE OF THIS DOCUMENT 6
1.2 ORGANISATION 6
2 SCOPE OF PLM 7
2.1 OBJECTIVES OF THE A1 DEMONSTRATOR 7
2.2 OBJECTIVES OF THE A2 DEMONSTRATOR (EOL) 7
2.3 OBJECTIVES OF THE A3 DEMONSTRATOR 7
2.4 OBJECTIVES OF THE A5 DEMONSTRATOR 8
2.5 OBJECTIVES OF THE A8 DEMONSTRATOR 8
2.6 OBJECTIVES OF THE A10 DEMONSTRATOR 8
2.7 OBJECTIVES OF THE A11 DEMONSTRATOR 9
3 REQUIREMENTS OVERVIEW 9
3.1 USER REQUIREMENTS 9
3.1.1 Beginning of Life (BOL) 9
3.1.2 Middle of Life (MOL) 9
3.1.3 End of Life (EOL) 10
3.2 FUNCTIONAL REQUIREMENTS 10
4 INFORMATION ARCHITECTURE REQUIREMENTS FOR PLM 13
4.1 UNIQUE IDENTIFIER 14
4.2 INFORMATION RESOURCES 15
4.3 ROUTING OR LOOKUP MECHANISM 16
4.4 TIMELY INFORMATION 16
4.5 SYNCHRONISATION 16
4.6 EASE OF RECONFIGURABILITY 16
5 ANALYSIS OF RELEVANT ARCHITECTURE APPROACHES 17
5.1 PROMISE ARCHITECTURE 17
5.1.1 The identifier 17
5.1.2 The Routing mechanism 18
5.1.3 The Information Resources 18
5.1.4 Timely Information 18
5.1.5 Reconfigurability 18
5.1.6 Data Analysis 18
5.1.7 Summary 18
5.2 EPCGLOBAL NETWORK ARCHITECTURE (EPC NETWORK) 18
5.2.1 The identifier – The Electronic Product Code (EPC) 22
5.2.2 The lookup mechanism – The Object Name Service (ONS) and Discovery Services 22
5.2.3 Information Resources – EPC Information Services (EPCIS) 23
5.2.4 Timely information – events, filtering and 'push' mechanisms 23
5.2.5 Reconfigurability – a layered architecture 23
5.2.6 Data analysis 24
5.2.7 Summary 25
5.3 DIALOG SYSTEM (HELSINKI UNIVERSITY OF TECHNOLOGY) 25
5.3.1 The identifier – ID@URI 25
5.3.2 The Routing mechanism 26
5.3.3 The Information Resources – software agents 26
5.3.4 Summary 27
5.4 WWAI NETWORK 28
5.4.1 The Identifier – The WWAI identity code 28
5.4.2 The Information Resources – nodes storing files and catalogues of files 28
5.4.3 The Routing Mechanism – the WWAI joining protocol 29
5.4.4 Reconfigurability 29
5.4.5 Timely information – subscription to events 29
5.5 SUMMARY 29
6 INFORMATION ARCHITECTURE COMPARISON 32
7 ACTION PLAN FOR EVALUATION AND REFINEMENT OF PROMISE STANDARDS 34
7.1 STANDARD FOR PRODUCT LIFECYCLE DATA REPRESENTATION 34

7.2 STANDARD FOR PRODUCT LIFECYCLE DATA EXCHANGE 35
8 CONCLUSION 35

List of figures

FIGURE 1 ROUTING MECHANISMS MAY LINK THE UNIQUE ID TO PROVIDERS OF AUTHORITATIVE INFORMATION, AS WELL AS TO MULTIPLE PROVIDERS OF LIFECYCLE INFORMATION. 15
FIGURE 2 PROMISE ARCHITECTURE OVERVIEW. 17
FIGURE 3 AN OVERVIEW OF THE EPC NETWORK. THE PART THAT IS LOCAL TO A SINGLE ORGANIZATION IS SHOWN WITHIN THE SHADED REGION WITH THE DASHED BORDER [15]. 19
FIGURE 4. AN OVERVIEW OF A WIDE AREA EPC NETWORK [15]. 20
FIGURE 5 EPC NETWORK ARCHITECTURE SHOWING STANDARDIZED INTERFACES [15]. 21
FIGURE 6 DIALOG ARCHITECTURE OVERVIEW. 25
FIGURE 7 WWAI ARCHITECTUE OVERVIEW. 28

Abbreviations

BOL: Beginning of Life

MOL: Middle of Life

EOL: End of Life

PEID: Product Embedded Information Device

PMI: PROMISE Messaging Interface *NOTE: prior to M36 referred to as the* ***PROMISE Middleware Interface***

PDKM: Product Data Knowledge Management

PLCS: Product Life Cycle Support

UML: Unified Modelling Language

XML: eXtensible Markup Language

PDM: Product Data Management

DBMS: Data Base Management System

PLM: Product Lifecycle Management *Note: All references to PLM refers specifically to* ***closed-loop PLM***

DSS: Decision Support System

OMG: Object Management Group

MDA: Model Driven Architecture

ERP: Enterprise Resource Planning

WWAI: World Wide Article Information

1 Introduction

1.1 Purpose of this document

This report provides a comparison of different Information Management Architectures and evaluates them against the requirements for product lifecycle management.

The report draws desired requirements from PROMISE demonstrator design documents for the three scenarios of BOL (Beginning of Life), MOL (Middle of Life) and EOL (End of Life) as well as for EOL decisions including parts tracking capability for EOL process improvement.

The activities performed in this report are:

- Examine each of the demonstrator design requirements (demonstrator design documents to be found in A*x*.3 documents) to draw a set of requirements for PLM (Product Lifecycle Management). The requirements have been drawn from A1, A2, A3, A5, A8, A10 and A11 demonstrators, since this set of demonstrators adequately covers the different yet related set of requirements from various application owners.
- Analyse relevant information architecture for their ability to meet the requirement of a PLM Information Architecture.
- Examine key areas of the various architectures and evaluate them against simple criteria developed for assessing information systems to provider a wider perspective on PROMISE in relation to designs and approaches taken by other information architectures.
- As a conclusion, highlight directions for improving the PROMISE architecture.

This document should be considered as a “Living Document” that will be modified and updated as the *PROMISE Architecture Series Volume 1* and *Volume 3* become available during the development process of the PROMISE architecture. This process will ensure that this deliverable remains relevant after the Architecture Series is finalized and that the deliverable does not use information that has since been in the process of modification.

1.2 Organisation

This document has four specific sections: capture of requirements; development of information architecture requirements for PLM; an overview and an analysis of PROMISE, EPC Network and DIALOG and WWAI information architectures; architecture comparison utilising a criteria developed by the authors.

While it is possible to develop a general information architecture comparison, a more useful approach is to develop an architecture comparison taking into account the specific needs relevant to product lifecycle management. We have therefore focused on the key application cases to extract requirements in the first section of this document. These requirements have then been extended to develop a comprehensive architecture comparison framework to compare the PROMISE architecture with three other relevant and important architectures; EPC Network architecture framework, DIALOG and WWAI information architecture.

2 Scope of PLM

Prior to discussing information architecture it is important to understand the requirements that must be fulfilled by such Product Lifecycle Management Information Architecture (PLMIA). The scope of these requirements can be extracted from the descriptions of the demonstrators designed by the application owners.

2.1 Objectives of the A1 demonstrator

The domain of the Application Scenario for A1 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 so that they can be reprocessed. This scenario is increasingly becoming a significant driver for change due to the EU directive on ELVs – EU-Directive 2000/53/EC- preventing waste from ELVs, promoting collection, re-use and recycling of vehicle components and placing a burden on manufacturers to minimize the environmental impact at all stages of life cycle of vehicles. The objectives of this demonstrator are summarized in Table 1.

Table 1: Summary of main objectives of the A1 demonstrator.

Objective
Identify components that are reusable when deregistering the vehicle
Recording relevant component information from BOL (production phase) and during MOL (component usage and maintenance information)
Decide whether to re-use a component or not, using all the information from BOL and information collected during MOL.
Transfer onto the component “some” relevant information about its post–deregistering life

2.2 Objectives of the A2 demonstrator (EOL)

Demonstrator objectives are to collect data with part tracking and information from the engine regarding use conditions, to be used for decision making at end of life, with the end goal to increase the whole lifetime of engine components. Table 2 summarizes the main objectives.

Table 2 Summary of main objectives of the A2 demonstrator.

Objective
Part tracking capability throughout multiple life cycles of engine components for supply chain process improvements.
Decision-making at EOL of an engine and engine components using historical data of engine use and service conditions captured during MOL and accessible at EOL.
Making BOL data (built date, fabrication plant, type of machine equipped with the engine, etc.) available at EOL.

2.3 Objectives of the A3 demonstrator

The objectives of the A3 demonstrator application is to showcase the tracking and tracing of products identified for recycling in combination with indoor and outdoor navigation systems. Table 3 provides a summary of objectives.

Table 3 Summary of main objectives of the A3 demonstrator.

Objective
To increase 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.
To track and trace materials and manage the availability, security, accuracy and integrity of all relevant product data at every stage of the recycling process.
To use all available information during the EOL phase of the chosen product (e.g. car bumpers) to optimise and automate human decision making at input to the recycling process.

2.4 Objectives of the A5 demonstrator

Demonstrator objectives are to optimize product lifecycle of structures and to enable predictive maintenance. Table 4 provides an overview of the objectives.

Table 4 Summary of main objectives of the A3 demonstrator.

Objective
To be able to schedule maintenance on structures considering their fatigue status, obtained through embedded sensors, and parameters linked to application severity, i.e. machine configuration (list of attachments equipping the machine) and historical payload (for instance, described by total payload from cylinders as well as number of times the lifter and ripper have been manoeuvred up and down).
To improving the design of structural parts thanks to fatigue damage monitoring of main structures and field knowledge on machine use and application severity.

2.5 Objectives of the A8 demonstrator

The main purpose of the A8 demonstrator is to realize the benefits of a predictive maintenance strategy as outlined in Table 5.

Table 5 Summary to A8 demonstrator objectives

Objective
To enable predictive maintenance by remote monitoring and periodical reporting of the health of a household appliance (refrigerator).

2.6 Objectives of the A10 demonstrator

The A10 demonstrator aims to use life cycle data collected and analysed within Bombardier Transportation (BT), specifically for the TRAXX™ Platform, for improving product design. More specifically, the A10 demonstrator aims at generating knowledge to use for the improvement of the design of next generation of BT locomotives through an appropriate analysis of field data mainly from maintenance records of BT locomotives. The main objectives are given in Table 6.

Table 6 Summary of A10 demonstrator objectives.

Objective

™ Bombardier TRAXX is a Trademark of Bombardier Inc or its subsidiaries

Objective
To develop the DfX (Design for X) decision strategy using a DSS (Decision Support System) and improved DfX knowledge management using life cycle data collected about products during their MOL.
To generate knowledge regarding RAM/LCC (reliability, availability, maintainability/life cycle cost) and safety.

2.7 Objectives of the A11 demonstrator

The A11 demonstrator focuses on the BOL (design and production) of a product in an application to an adaptive production process aimed at closing the information loops between the experience in the product's MOL and EOL phases and the decisions needed to adapt the production system in the BOL phase. The objective of this demonstrator is summarised in Table 7.

Table 7 Summary of A11 demonstrator objectives.

Objective
To develop a DSS that uses data collected during BOL (production phase), MOL (field data) and EOL to predict product modification requests.

3 Requirements overview

The primary requirement of an information architecture for a new PLM system is to make information available over the whole product lifecycle for various user requirements, such as streamlining product lifecycle operation, through remote and seamless connection to information repositories. The following sections will first consider user requirements as well as functional requirements that can be derived from demonstrator scenarios to construct a necessary set of requirements for a PLM information architecture (PLMIA).

3.1 User requirements

The user requirements for the business model can be divided into three categories according to the product lifecycle phases in PLM: Beginning Of Life (BOL), Middle Of Life (MOL), and End Of Life (EOL).

3.1.1 Beginning of Life (BOL)

The user requirements of BOL are related to product design and production system. The main requirements are as follows:

- Product design improvement using MOL and EOL data
- New product design generation based on product lifecycle information
- The ability to adapt production systems based on product performance in MOL
- Improvement of warehouse management with advanced tracking and tracing information
- Development of efficient logistics operation in production with advanced tracking and tracing information

3.1.2 Middle of Life (MOL)

The user requirements of MOL are about enhancement of maintenance/service using additional

information:

- Development of the predictive maintenance method
- Supporting of maintenance/service operation

3.1.3 End of Life (EOL)

The main keyword of EOL requirements is the decision making process at the EOL of a product. At the EOL phase of a product, the following is required.

- Track and trace ability for EOL product logistics
- Optimization of EOL process from reverse logistics to disposal
- Assessment of EOL product such as ability to be recycled, reused and re-manufactured
- Supporting EOL decisions (for instance reuse, recycle, remanufacturing, or disposal)

3.2 Functional requirements

Ingredients for an information architecture capable of achieving the objectives of the demonstrators for PLM can best be obtained from the descriptions given in Section 2 of this document. The functional requirements extracted from the demonstrators are summarised in Table 8 below.

Table 8 An overview of requirements.

Demonstrator Description	Stage of the life cycle	Functional requirements
A1 - Application Scenario for A1 is the EOL phase of the product lifecycle. It specifically deals with the return of ELVs to dismantlers so that they can be reprocessed.	BOL	Globally unique identifier for objects and the storage of object related quasi-static data.
		Access to previous lifecycle information on recycled parts.
		Product design improvement using MOL and EOL data.
		New product design generation based on MOL and EOL data.
	MOL	Object usage information capture: • record significant component related events such as peak pressure or excessive temperatures, • external temperature, • temperature in the engine area
		Capture maintenance events: • History of maintenance activity, • List of replaced parts and corresponding data, • Related information about aging statistics (and possible reset after substitution)
	EOL	Access to component relevant information from BOL and MOL.
		Data analysis functions – DSS • Automate decision making at EOL (recycle/re-use/remanufacture)
A2 - Part tracking and data collection from the parts regarding use conditions, to be used for decision making	BOL	Globally unique identifier for objects and the storage of object related quasi-static data.
		Access to previous lifecycle information on recycled parts
	MOL	Capture maintenance events: • similarly to A1 MOL event capture

Demonstrator Description	Stage of the life cycle	Functional requirements
at end of life, with the end goal to increase the whole lifetime of engine components.		• event occurrences recorded during engine life defined at hours of operation
		Object usage information capture: • similarly to A1 MOL component usage information capture. • Hours of operation
	EOL	Access to component relevant information from BOL and MOL.
		Data analysis functions – DSS • Automate decision making at EOL (recycle/re-use/remanufacture)
		Track and trace: • part tracking capability for increased visibility in logistics along the remanufacturing process
A3 - Showcase the tracking and tracing of products identified for recycling	BOL	Record new BOL data for objects.
	EOL (recycling)	Access to component relevant information from BOL.
		Track and trace: • object track and trace capability during the recycling phase
		Data analysis functions - DSS • Automate decision making at EOL (logistical decisions – storage, production, clearing) • DSS is a complex entity based on several logistical systems and processes.
A5 - Optimize the lifecycle of products and enable predictive maintenance	BOL	Globally unique identifier for objects (structures) and the storage of object related quasi-static data.
		Product design improvement using MOL (field data) and EOL data.
		New product design generation based on MOL and EOL data.
	MOL	Capture maintenance events: • Similarly to A1 MOL, more specifically record repair or part replacement of failed structures. • Record failure modes. • event occurrences recorded during engine life defined at hours of operation
	EOL	Data analysis functions – DSS • Decide whether to re-use old structures.
		Access to component relevant information from BOL and MOL.
A8 - Predictive maintenance by remote monitoring and periodical reporting of a products health.	MOL	Object usage information capture: • Record fault and failure events • Record status of appliance/objects (power usage, operations temperature, component usage times)
	MOL/EOL	Data analysis functions – DSS (predictive maintenance) • Ability to assimilate distributed field data from different and distributed information resources during MOL.
A10 - Use data collected during various lifecycle	BOL	Globally unique identifier for objects and the storage of object related quasi-static data.
		Product design improvement using MOL and EOL data.

Demonstrator Description	Stage of the life cycle	Functional requirements
phases to improving product design.		New product design generation based on MOL and EOL data.
	MOL	Capture maintenance events: • similarly to A1 MOL event capture
		Object usage information capture. • Record fault and failure events
	MOL/ EOL	Data analysis functions – DSS • Ability to assimilate distributed field data from different and distributed information resources during MOL. • A complex DSS system is required to provide input to a other systems.
A11 – Adaptive production process based on BOL, MOL, and EOL data.	MOL	Capture maintenance events
		Object usages information capture • Record fault and failure events
	BOL	Data analysis functions – DSS
		Access to component relevant information from BOL and MOL.

There are clearly a set of intersecting requirements among the various demonstrators and the demonstrators together represents movement of the product through the various phases in its lifecycle (BOL, MOL and EOL) and the distributed collectors and users of the products lifecycle data.

Thus the information architecture must, in its most general configuration, be able to support through-life information management. The functional requirements from all the demonstrators are summarised in Table 9.

Table 9 Summary of functional requirements

Stage of the life cycle	Functional requirements
BOL	Globally unique identifier for objects and the storage of object related quasi-static data.
	Access to previous lifecycle information on recycled parts.
	Product design improvement using MOL and EOL data.
	New product design generation based on MOL and EOL data.
MOL	Object usage information capture.
	Capture maintenance events.
EOL	Track and trace.
	Access to component relevant information from BOL and MOL.
	Data analysis functions – DSS

Thus far we have considered user requirements and the functional requirements of a PLM infrastructure. The following section will consider the capabilities that a PLMIA needs to provide to satisfy the functional requirements identified in the demonstrator cases.

4 Information architecture requirements for PLM

Clearly, managing product lifecycle information is a strategic business approach that encompasses the management of data associated with a product, of a particular manufacturer, of a particular type and of a particular version as well as the business processes that surround the product [1, 2, 3].

The product information first begins its inception at the design stage at BOL where the product is a collection of ideas, design documents or requirements. This stage of the process can be spread across different organisation and across geographical boundaries. 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.

When the product is sold to a consumer it enters its MOL phase. The product may undergo various business, maintenance, repair and work processes during its MOL phase. Then, as a tangible result of all the previous phases of its usage history, the product reaches its EOL.

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 various systems.

Given such a context, the user requirements and the functional requirements summarised in Table 9, a PLMIA will need to address the following challenges.

1. 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 'backend' systems or information resources. The significant advantage is the ability 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 the access to data in the absence of the product.
2. A routing or a lookup mechanism is needed to find and where necessary update lifecycle information resources that may be distributed across organisations and locations.
3. Given the distributed nature of data collection and storage there must be a mechanism for associating products with their relevant lifecycle data on backend systems as well as products themselves.
4. The system should be able to operate and respond in real time, for instance, providing input to decisions at a maintenance event during MOL or monitor critical events for condition or predictive maintenance.
5. During the movement of products through their lifecycle they are likely to have only intermittent network access. While functional requirements have only highlighted the importance of data availability and data collection, what has been concealed is the implied requirement of 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 phases or locations.

6. A robustness so that the system should have mechanisms to insulate application programs from changes to actual configuration details, in terms of changes to both hardware and software.
7. Finally the information architecture should allow the easy implementation of data analysis functions such as decision support and track-and-trace to support PLM decisions and operations.

The challenges above need to be addressed by a PLMIA. Addressing the key issues require a PLMIA to allow seven essential capabilities summarised in Table 10 below. Each of these capabilities is discussed in more detail in the following sections.

Table 10 Addressing PLM information management requirements

	Requirement
1	Globally unique identifier
2	Routing or lookup mechanisms
3	Information resources
4	Timely information
5	Synchronisation
6	Reconfigurability
7	Data analysis

4.1 Unique identifier

The key issue in the preceding discussion is the problem of creating a link between a product and the related product information. The use of a unique identifier (UID) with a global scope for each object under consideration enables the collection, location and retrieval of information related to an individual product.

The unique identifier can be used to discover and access information associated to the UID from distributed information resources, similarly to the manner in which web addresses or Uniform Resource Locators (URLs) [8] 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.

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. During the passage of time various parties may have collected and recorded some information about the object. This is shown in Figure 1. For applications requiring track-and-trace functionality as well as analysis applications based on retrospective data requiring information from the MOL and BOL phases of an object, it is helpful if these multiple pointers are persistent, rather than purely transient.

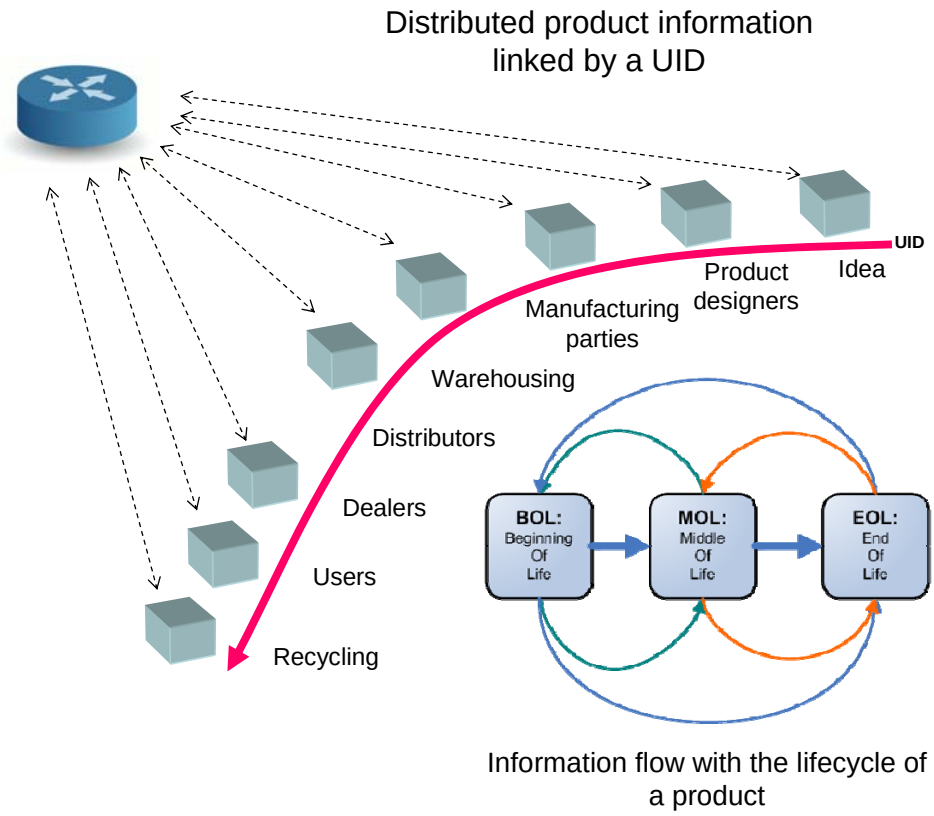

Figure 1 Routing mechanisms may link the unique ID to providers of authoritative information, as well as to multiple providers of lifecycle information.

4.2 Information resources

Collection and storage of through life information requires a collection of information resources being 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 different source of data.

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 [9, 10], 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.
- 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.

4.3 Routing or 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, what has not been considered thus far is a routing or lookup mechanism 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 of the data, the mechanism is a 'lookup'. For instance, DNS and search engines help with lookup while TCP/IP, SMTP and UDP 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.

4.4 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 information in a timely manner.

4.5 Synchronisation

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 its 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.

4.6 Ease of reconfigurability

In a PLMIA 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 hardware such as readers, tags or PEIDs (vendor, reader/tag type and protocol, operating frequency, number of readers/antennae per cluster monitoring a particular location), changes to underlying databases (vendor, database type, structure), changes to network IP addresses (for readers, sensors, PEIDs, databases, computers which filter or process the data), as well as changes to business partners and processes, for example preferred maintenance company.

The following sections will analyse the various architecture approaches with regards to the requirements outlined in Table 10.

5 Analysis of Relevant Architecture Approaches

Listed below is a set of currently known approaches of four information architectures to managing product information.

1. PROMISE architecture.
2. EPC network architecture with its standard interfaces for collecting and accessing product related data
3. The approach taken by the DIALOG information system using its ID@URI approach.
4. 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 a brief overview of these architectures prior to analysing the different approaches they have taken using the PLMIA requirements we have considered in Section 4.

5.1 PROMISE architecture

The PROMISE architecture has evolved and matured since its early inception in the PROMISE project. The current view of the PROMISE architecture, presented in [13], is illustrated in Figure 2. *<<To be completed after Architecture Series Volume 1 is completed>>*

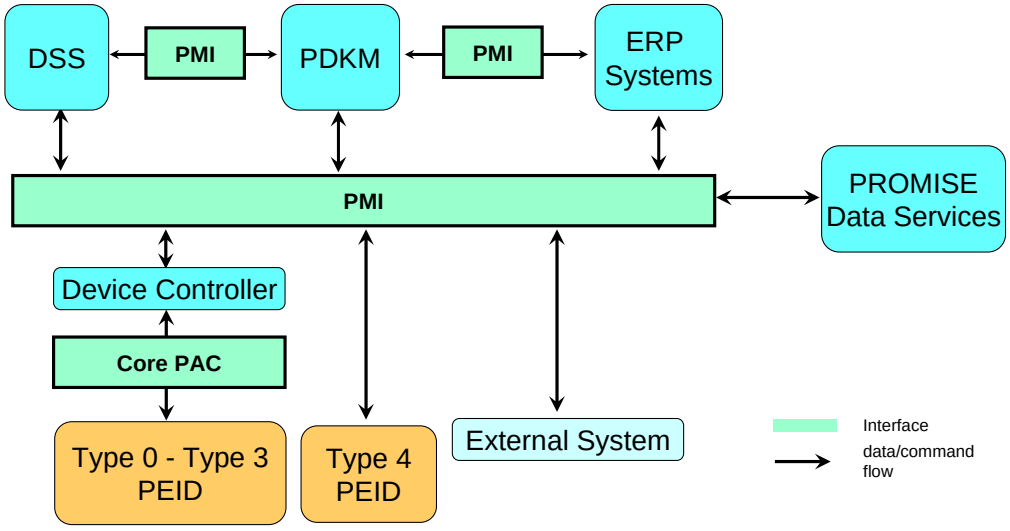

Figure 2 PROMISE architecture overview.

5.1.1 The identifier

PROMISE architecture is capable of using any identifiers as long as it is possible to retrieve routing information from it. Thus it may be and EPC where routing information is obtained by way of the ONS or it may be an ID@URI or a WWAI identifier. *<<To be completed after Architecture Series Volume 1 is completed>>*

5.1.2 The Routing mechanism

The routing mechanism will depend on the identifier used by the PEID device. As such, PROMISE does not provide a routing mechanism but is expected to utilise the lookup or routing mechanism provided by the identifier. *<<To be completed after Architecture Series Volume 1 is completed>>*

5.1.3 The Information Resources

As a central component of the PROMISE approach, the Product Data Knowledge Management (PDKM) system integrates and manages product-related data from all lifecycle phases to support comprehensive data analysis and to enhance operational businesses with obtained insights.

The PDKM system aims to integrate and manage data from all lifecycle phases of products. Lifecycle data from different phases are collected and used to support data analysis tools discussed in Section 5.1.6. The approach that PROMISE has taken involves the storage of information and events to a centralized location maintained by the originator of the final product delivered to the market (manufacturer).

5.1.4 Timely Information

<<To be completed after Architecture Series Volume 1 is completed>>

5.1.5 Reconfigurability

The PROMISE architecture provides a standard format for accessing and storing PLM data Using the PMI as the data exchange standard and the PDKM object model as the data representation standard. *<<To be completed after Architecture Series Volume 1 is completed>>*

5.1.6 Data Analysis

PROMISE has developed a number of decision support systems (DSS) to complement the PDKM. The functionality provided by the DSS tools developed are an important feature of the PLMIA. The DSS tools are developed to meet various demonstrator requirements and the functionality provided by the data analysis tools provide predictive maintenance, diagnosis and EOL decisions.

Currently the DSS is coupled with the PDKM and support is provided to decision making at BOL, MOL and EOL phases of a product's lifecycle.

5.1.7 Summary

<<To be completed after Architecture Series Volume 1 is completed>>

5.2 EPCglobal network architecture (EPC Network)

The EPC Network can be described as an intelligent ubiquitous infrastructure that automatically and seamlessly links physical objects to the global Internet. The networking of physical objects is achieved by integrating an RFID tag, into each object. The system interfaces with objects seamlessly by communicating with these tags using interrogators at suitably placed locations, for example: RFID portals; handheld readers; and potentially, eventually for some tags, continuously

throughout the environment by a network of readers. Readers collect data from tagged objects. The RFID tagged objects communicate a unique ID (or 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.

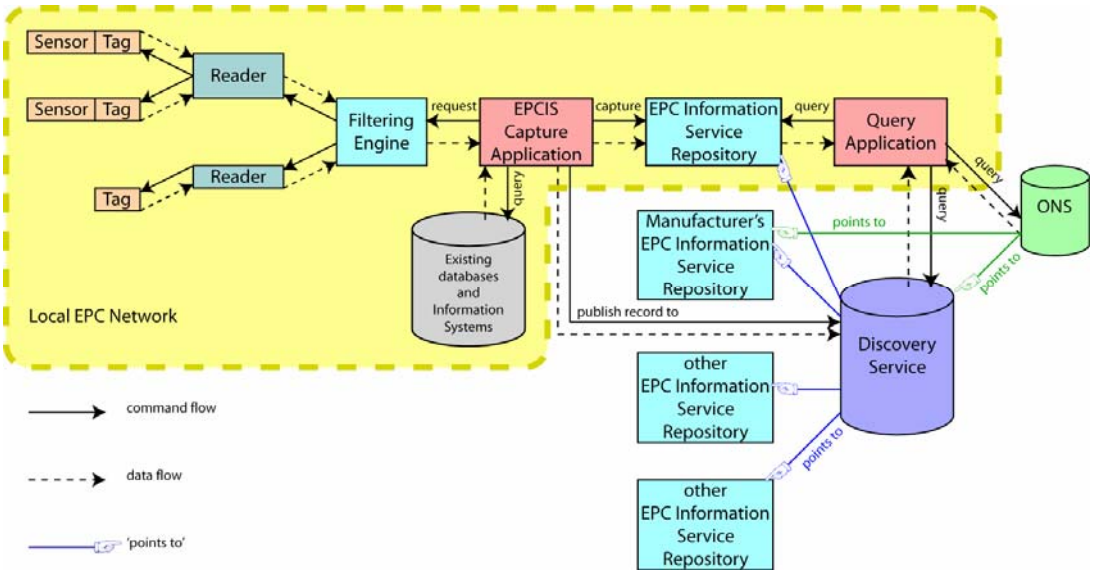

Figure 3 An overview of the EPC Network. The part that is local to a single organization is shown within the shaded region with the dashed border [15].

An illustration of the components constituting the EPC Network is shown in Figure 3 where the arrows indicate the flow of data from tags to the network support system and the flow of commands and data back to the readers and tags.

The EPC Network architecture is significantly different from more traditional computer networks; the flow of data and information is from many nodes (RFID tags) at the edge of the network towards a number of servers that collect and process this data. In RFID networks, readers detect certain events or readers query RFID tags to obtain event data and forward the resulting information to backend applications or servers. The application systems then respond to these events and application processes orchestrate corresponding actions such as ordering additional products, sending theft alerts, raising alarms regarding harmful chemicals or replacing unsafe components before failure.

The EPC Network architecture can be separated into six primary modules, some physical, some logical: (1) RFID tags (also known as 'labels' or 'inlays'), (2) RFID tag readers (also known as interrogators), (3) EPC, (4) the filtering middleware that supports the Application Level Events (ALE) interface, (5) Object Name Service (ONS), and (6) EPC Information Service (EPCIS).

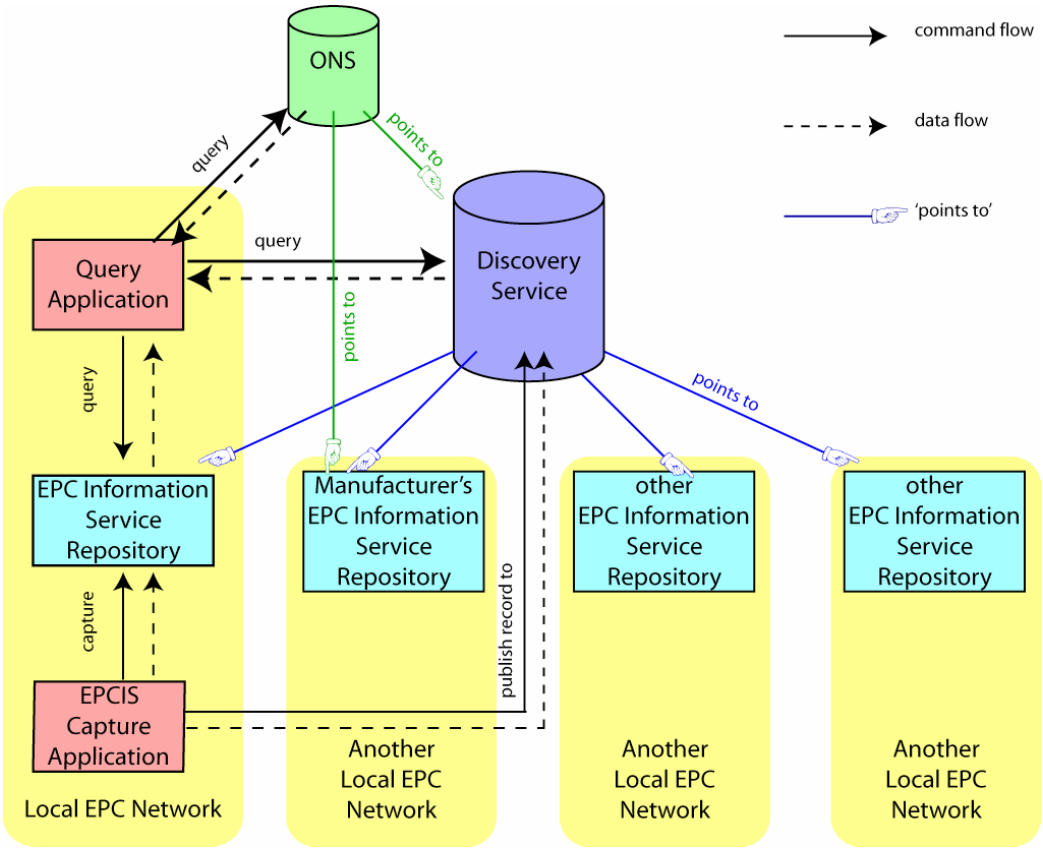

Figure 4. An overview of a wide area EPC Network [15].

The EPC Network shown in Figure 3 is a local area EPC Network akin to a LAN. This model captures the architecture of the system at a local site, company or organization, or a private network. Although data collected within an organization from a local EPC network may be centralized towards a local EPCIS of that company, the architecture is designed to support decentralized distributed information management across the supply chain or product lifecycle.

The EPC Network architecture achieves the latter goal by linking local EPC Networks together through the already well established backbone of the Internet. The resulting wide area EPC Network achieves a global flow of information and data, while at the same time extending the reach and the usefulness of the network. The wide area network architecture enables the exchange of information between organizations at the level of individually identifiable unique objects and supports the extraction of, for instance, 'lifecycle' information for an object which may be fragmented across multiple organizations rather than being held in one single centralized database. Figure 4 illustrates such an architecture where a global public ONS system together with Discovery Services may be used to link to multiple local area EPC Networks.

Thus as a product moved through different phases during its lifecycle or different organisations during its life, the data is gathered, stored and made available by that organisation who collected the data. The discovery services and the ONS systems are together used to locate data associated with a product which may be distributed across different locations and organisations.

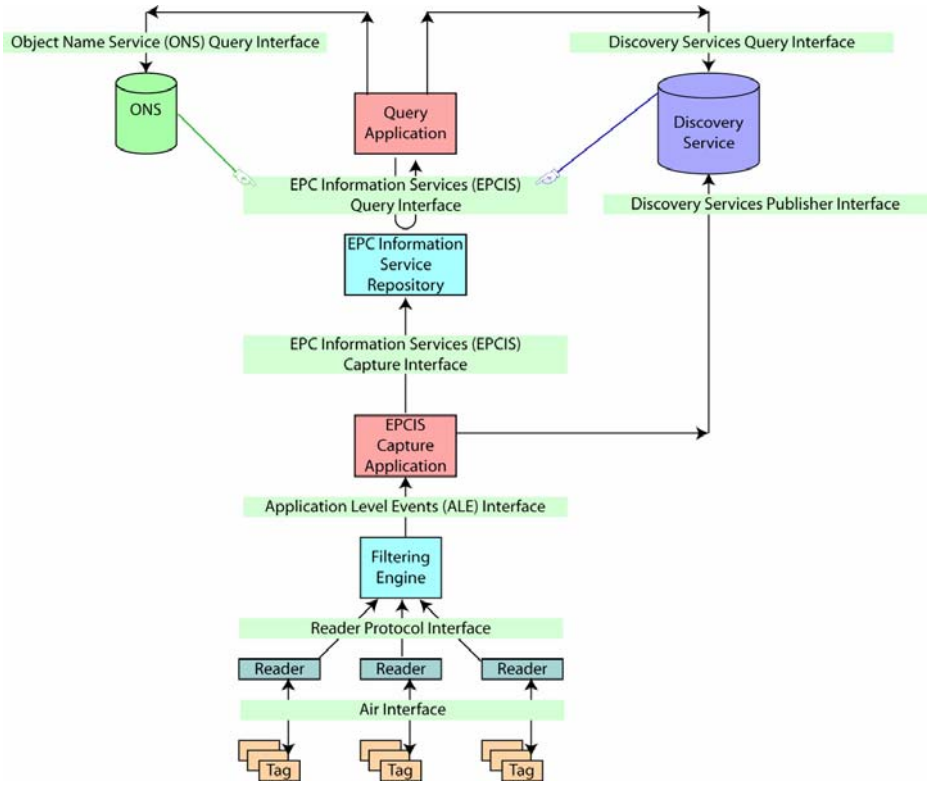

Figure 5 EPC Network architecture showing standardized interfaces [15].

There are ongoing collaborative efforts to standardize the interfaces linking the modules outlined in Figure 3 by various actions groups of EPCglobal, and the Auto-ID Labs, 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 EPC Networks. Figure 5 identifies the interface layers being standardized by EPCglobal [4]. At the time of compiling this document all of the interfaces identified in Figure 5 have been developed and ratified as standards by EPCglobal, with the exception of standardized interfaces for Discovery Services, for which the process of gathering of user requirements is underway. All ratified EPCglobal standards are freely available in electronic format [5].

The air interface and the reader protocol interface describe standardised interfaces for reading and writing to tags and receiving low-level event data and providing this in a standard format. The filtering layer provides filtered 'event data' about a collection of tag identities (EPCs) read by a particular logical reader (which often corresponds to a physical location) within a particular event cycle, while removing duplicate EPCs. The Application Level Events (ALE) interface standard allows client applications to request filtered data from readers. 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') as well as associations with business transaction IDs.

The EPC Information Services (EPCIS) standard allows client applications to request the higher-level data, complete with annotations about business context, relationships to business transaction IDs, as well as aggregation relationships between an object and its parent container. Provision is also made for access to serial-level attribute data.

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 filter and also the returned payload data.

5.2.1 The identifier – The Electronic Product Code (EPC)

The Electronic Product Code was 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.

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.

The EPC concept has developed to include a fast filter value, to allow efficient selection between tags on different packaging levels (for example pallet or case or item level). Furthermore, in an effort towards coherence, the existing family of GS1 coding schemes (GTIN, SSCC, GLN, GRAI, GIAI) can now be expressed within the EPC format. Although there is some additional complexity in the mapping between these coding schemes and the binary EPC format used for storage of the number on tags, the hierarchical, delegated structure of the EPC continues – and covers not only unique identifiers for products – but also unique identifiers for shipments, locations, returnable assets and high-value or long-lived assets such as medical equipment.

The EPC naming scheme is a ratified published open standard known as the ‘EPC Tag Data Standard’. EPC product identifiers can be formatted as URNs (Universal Resource Names) for use in the EPC network as described in the ‘EPC Tag Data Standard’.

5.2.2 The lookup mechanism – The Object Name Service (ONS) and Discovery Services

The Object Name Service (ONS) provides a lookup service which decouples the EPC identity from the address(es) of associated information resources. The ONS provides a lookup service used 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 webpage 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 the domain `onsepc.com`. ONS records use DNS type 35 (NAPTR) records for returning results to ONS queries. ONS supports a scalable hierarchical lookup services utilising 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 administered by 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 with to develop standards for Discovery Services (DS) to provide serial-level lookup services across supply chains (multiple pointers to object related information and services). DS will 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 organizations who have collected information about the object at some time in its lifecycle). Furthermore, because of the commercial sensitivity of serial-level data about volumes and flows of goods, Discovery Services are likely to have much more stringent security requirements than the Object Name Service requires.

5.2.3 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 as illustrated in Figure 5.

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 obtain higher-level semantic annotations such as business transactions or processes associated with event observation associated with objects.

5.2.4 Timely information – events, filtering and 'push' mechanisms

ALE and EPCIS standards are concerned primarily 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.

5.2.5 Reconfigurability – a layered architecture

The whole EPC Network has been redesigned with a layered architecture, to provide maximum flexibility of reconfiguration to users, while minimizing disruption when changes are made. The EPC Network 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. The various layers of the architecture stack and the layering mechanisms are illustrated in Table 11.

Table 11 Reconfiguabilty through a layered architecture.

Level of Architecture	Layering mechanism / Insulation afforded
EPC Identifier	Information systems will use a URN format which is specific to each particular coding scheme (e.g. SGTIN, SSCC) even though multiple binary representations may exist for each coding scheme, having different numbers of bits (64 bits, 96 bits, etc.)
Reader Protocol	Provides a standard software interface for low-level reading/writing of tags, regardless of actual air interface protocol, frequency (HF, UHF, etc.) or make/model of reader
Application Level Events (ALE)	Event Cycle Specifications (which define the filtering and reporting criteria) refer to Logical Readers rather than IDs of Physical Readers. Logical Readers may represent one or more physical readers, allowing physical readers to be exchanged without reconfiguring applications which rely on filtered data.
	Events reported by the filtering middleware consist of 'significant events', with duplicate reads of EPCs removed. Business applications are insulated from needing to deal with interfacing to different types and makes of RFID reader
	The filtering middleware is agnostic to the technology used to detect observation events, whether RFID, barcode, human input – or other
EPC Information Services (EPCIS)	Provides a standard software interface for capture and query of higher-level events. The EPCIS interface is designed to be agnostic to operating system, programming language or details of the type and configuration of the underlying databases.
	The EPCIS capture interface is agnostic to the technology used to detect observation events, whether RFID, barcode, human input – or other
Object Name Service	Provides flexibility for the responsible party to change the binding or association between the Manager ID of an EPC and the address of the corresponding information resources or services.
	Uses meta-data in the NAPTR DNS records to provide support for different types of information resources to be listed – and automatically detected.
	Uses DNS records to provide for primary, secondary information services, as well as to express load-balancing preferences etc.
Discovery Services	Provides for links to multiple sources of lifecycle information about a given object, even when the individual supply chain path is not known ahead of time.

5.2.6 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 is currently being carried out in the BRIDGE project Work Package 3, also funded under the 6th Framework Programme Information Society Technologies [14].

5.2.7 Summary

In the EPC Network, the Electronic Product Code (EPC) forms the globally unique identifier, the Object Name Service and Discovery Services provide the lookup mechanisms, EPC Information Services provide access to information from networked databases. Real-time response is achieved by filtering of RFID data to propagate only significant events to applications and information systems and by the support for publish and subscribe triggers in requests for data from both ALE-compliant Filtering layer and from EPC Information Services.

EPC Information architecture currently provides no mechanisms for data synchronisation. However this is an important issue dealing with PLM data where data may be collected by several owners of an object throughout its life cycle [12].

5.3 DIALOG System (Helsinki University of Technology)

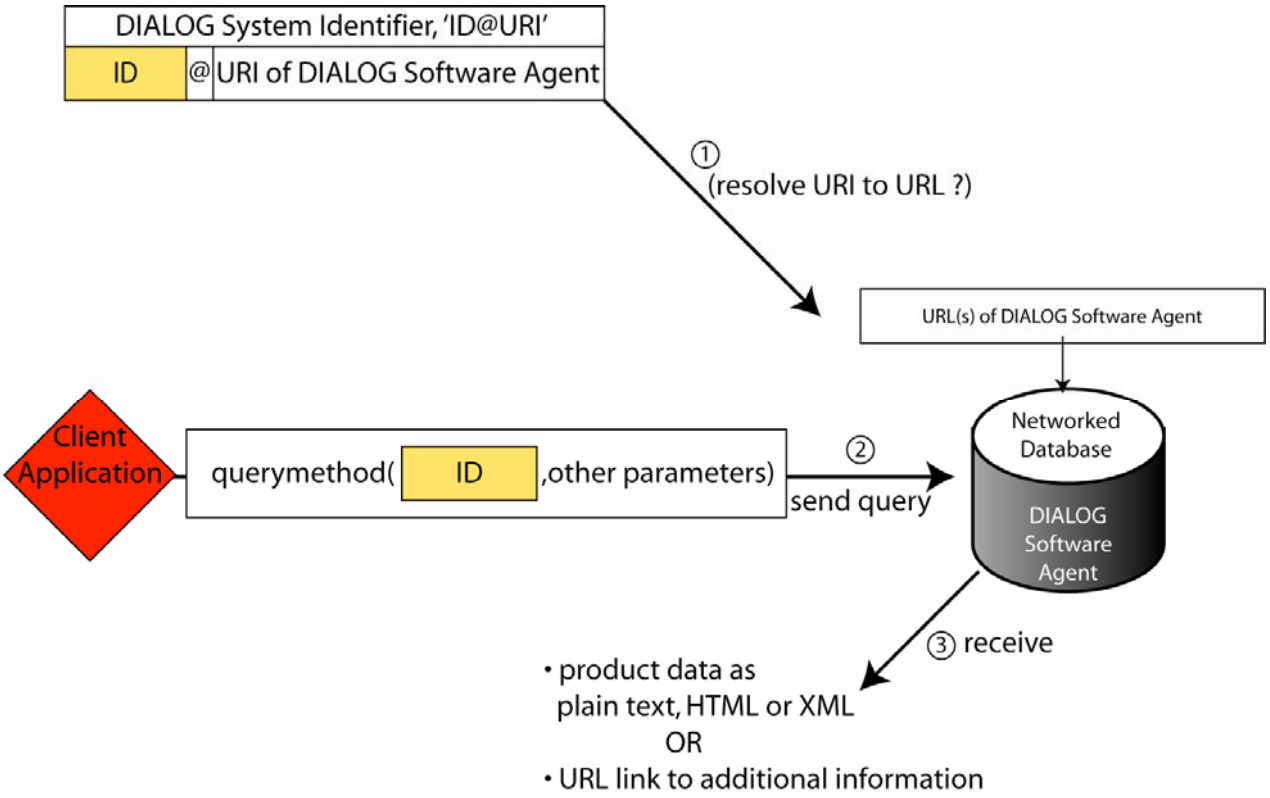

Figure 6 DIALOG Architecture overview.

The DIALOG System [6] is an open-source solution for tracking objects and accessing data about the objects [7].

5.3.1 The identifier – ID@URI

Its unique identifier consists of two components, a unique ID string and a URI 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 the '@' symbol is required to be unique within a particular mailserver, subdomain or domain specified after the '@' symbol. The authors

of the DIALOG system envisage that originators of physical objects might use for the unique ID part the unique Tag ID (which is often hard-coded by the RFID Tag Manufacturer in Read-Only Memory) – and record into the tag's ID memory the URI. For product data, this might be a URL within a domain owned by the manufacturer. For tracking shipments, this might be the URL of a company that requires, manages and securely distributes the tracking data.

5.3.2 The Routing mechanism

Since the identifier is of the form ID@URI, where a URI may be either a URN or a URL, it is not clear whether the DIALOG system uses or provides a specific routing mechanism. In the case of a URL, some routing may be provided by DNS, depending on the hostname within the URL, whereas services such as web servers, web services etc. may perform some level of routing based on the pathname information within the URL.

At the time of writing, there are very few general purpose mechanisms for resolving URNs into URLs other than the Handle/DOI system and domains specializing in routing or permanent addressing, such as purl.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.

5.3.3 The Information Resources – software agents

The software agents provide interfaces for:

- receiving tracking location updates
- 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 [7]. 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.

Like the EPC Network, the DIALOG system allows each company to maintain ownership of the data they create or collect and to exchange the data packets in a standardized 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. Currently, only mechanisms for file exchange (similar to e-mail attachments) are discussed.

In the DIALOG system, the central authority guaranteeing uniqueness of the URI parts is effectively IANA, 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. Unlike the EPC Network and Global Data Synchronization Network, no separate 'manager number' is required. Since many companies in

the developed world have already registered a domain name – and may then freely create any URIs and subdomains they choose, 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, since 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 URL in the tag memory of an RFID tag is that a much larger number of bits will be required than for a purely numeric identifier such as the EPC. If the binary-encoded URI cannot fit within the capacity of high-volume low-cost RFID tags, such as those which are currently being produced for EPC use [currently 96 – 256 bits], the higher cost of tags may hinder adoption of the DIALOG network, even though the 'subscription' cost is negligible and the need for a routing mechanism apparently obviated.

The ID part of ID@URI could be aligned with existing identifiers, subject to suitable serialization, so that for example a serialized GTIN code could be used if desired.

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. Whereas the Object Name Service supports multiple service types for a given EPC class, with standardized meta-data in the NAPTR records to distinguish between them, in the DIALOG System, the URI would need to provide a machine-readable list of pointers and corresponding meta-data if it were desired that a DIALOG software agent at a single URI would provide multiple service types, since the DIALOG System initially only resolves any tagged object to a single URI.

In the publications to date about the DIALOG Network, we cannot find any discussion of other supply chain partners registering as additional 'information providers' for the tagged object. The WWAI system already claims to incorporate this feature [16, 17, 18] – and the EPC Network will soon support this via real-time updateable Discovery Services for serial-level tracking. In the DIALOG System, the presumption appears to be 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 the authoritative agent with a pointer so that the agent may link to other parties as additional information providers. However, the automatic notification of location updates to the object's originator's software agent can be switched off it desired, although it does raise the issue of how other parties access information from providers who have switched off updating the object's originator.

5.3.4 Summary

The DIALOG System is a very worthy, apparently affordable entry solution to networking objects, with the advantage that there is no centralized numbering authority with a vested interest in managing or restricting who is allowed to participate and charging an additional fee for the privilege. What remains to be developed are a better-defined mechanism for linking to additional providers of information about a tagged object and standardized interfaces for fine-grained queries (as opposed to file retrieval). For example, EPC Information Services are designed to provide for rich relational information to be expressed and queried, such as associations with business transactions, aggregations and dis-aggregations of multiple objects, as well as very specific queries constrained by time range, location as well as by object identity or product class.

5.4 WWAI Network

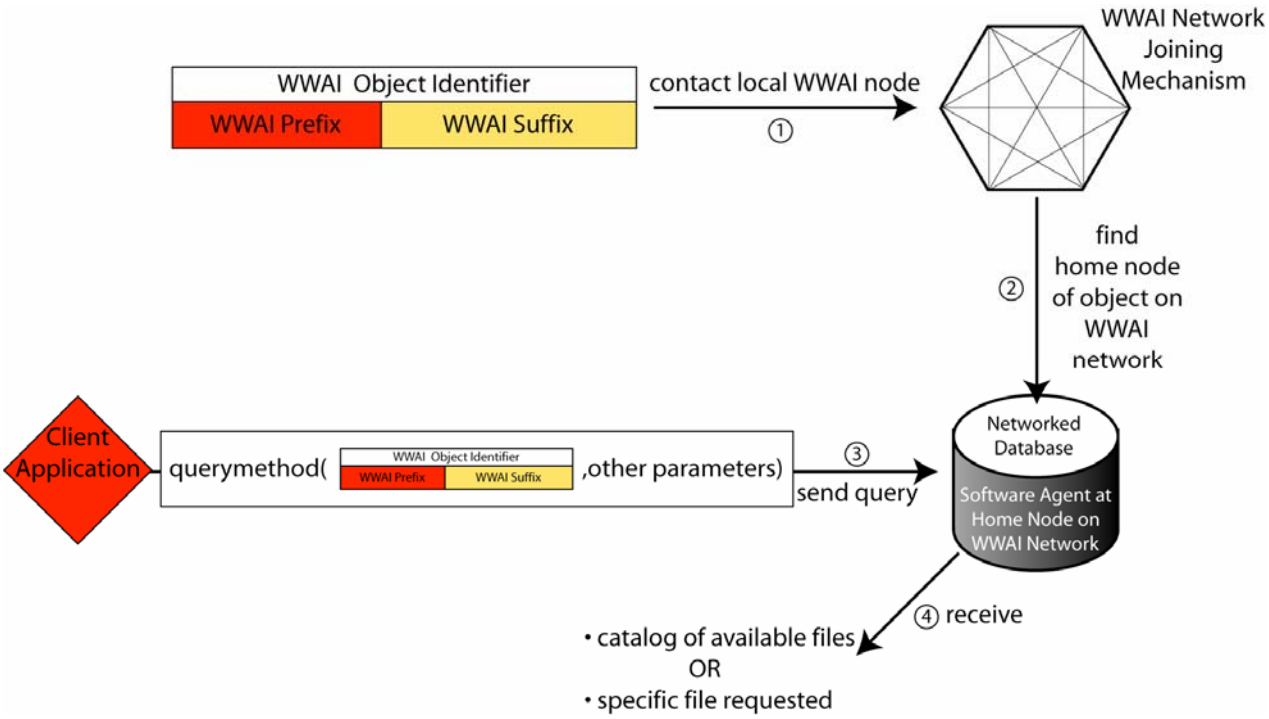

Figure 7 WWAI Architectue overview.

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 [11].

5.4.1 The 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 Handles 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.

5.4.2 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.

5.4.3 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.

5.4.4 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 and EPC network more resilient to changes in the information provider's address.

5.4.5 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. Note that 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 standardized 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.

5.5 Summary

The following table summarizes the discussions above on the four architectures to compare the different approaches used to address the needs of meeting PLM functional requirements and the user requirements.

Table 12 Summary of how various architectures satisfy PLM requirements

	PROMISE	EPCglobal Network	DIALOG	WWAI
Globally unique identifier	*<<To be completed after Architecture Series Volume 1 is completed>>*	The Electronic Product Code (EPC). Allocated centrally by EPCGlobal. Cost of guaranteeing uniqueness of company identifier requires subscription to EPCglobal. (However no subscription is required for use of USDOD-64 and USDOD-96 EPC identifiers)	ID@URI approach. Relies on each company owning a unique domain name. Cost of domain name registration is nominal.	WWAI identity code. WWAI prefix acts as company identifier to guarantee uniqueness. Currently only Trackway is the only authority able to issue WWAI prefixes.
Routing mechanism	PROMISE middleware routes information to a centralised networked resource. *<<It is not clear if PROMISE middleware supports multiple recipients of data and events collected>>* *<<To be completed after Architecture Series Volume 1 is completed>>*	Object Name Service (ONS) provides for lookup of originator's information but ONS currently only holds records per object class (product type) – not per individual object. Discovery Services for references per unique object, to distributed information resources (EPCIS repositories) maintained by multiple parties in possession of the object during its lifecycle.	No specific routing mechanism but relies on the URI obtained from the object identifier. A URL in the identifier would locate the originator's information. A URN in the identifier is more flexible but general URN to URL mappings are not yet widely deployed on the internet.	WWAI network joining mechanism discovers current address of software agent or information service of object's originator (manufacturer). Other parties in the lifecycle of the object may register with the originator as additional information providers. WWAI supports dynamic changes to IP addresses or URLs as they are detected by way of the joining mechanism
Information Resources	Centralised data storage - PDKM. Providing a standardised method of accessing data from the PDKM using the PMI interface and an associated data model is being defined. *<<Is access (read/write) to PDKM by third parties possible?>>*	Decentralised data storage and standardised access to networked resources through the EPCIS interface. Authorised third parties may access data repositories through an EPCIS query interface. The granularity of data can range from specific object level lifecycle data to product level data.	An agent representation of the object stores object information which is accessed through the agent hosted at a URL. However no data model for the data exchanged and all partners are expected to update the data source nominated by the object by way of its ID@URI identifier. Other parties may hold data locally but then	No centralised storage; all information related to objects is stored on the nodes of the WWAI network. The nodes notify and request to be listed as providers of information. The originator may accept or decline. The nodes control access to data.

	PROMISE	EPCglobal Network	DIALOG	WWAI
			they are expected to provide pointers to the object's originator for accessing that data.	
Timely information	Depends on effective synchronization with PDKM (as virtual counterpart of data held in PEID) and availability of access to third parties. *<<To be completed after Architecture Series Volume 1 is completed>>*	Real-time response is achieved by filtering of RFID data to propagate only significant events to applications and information systems and by the support for publish and subscribe triggers in requests for data from both ALE-compliant Filtering layer and from EPC Information Services.	Supported through "pushed" events such as the notification of location updates on objects to software agents.	Provides mechanism for other parties to subscribe to events and updated information about a particular object. This mechanism can be used to obtain a real-time response.
Synchronisation	There are no protocols or methods for data synchronisation	There are no protocols or methods for data synchronisation	There are no protocols or methods for data synchronisation	There are no protocols or methods for data synchronisation.
Re-configurability	*<<To be completed after Architecture Series Volume 1 is completed>>*	Layered service-oriented architecture to insulate applications from changes to reconfiguration of hardware. Granular interfaces to enable testing and certification of interoperability. Forwards and backwards compatibility built into standard interfaces by design (e.g. extension points in XML)	The ability to use any ID and the standardised interfaces between agents presents this architecture with a great degree of re-configurability. However the network is not resilient to changes in the URL of the information provider.	An object's UID is decoupled from the URL, hence more resilient (than DIALOG only) to changes in the information provider's address.
Data analysis	Provides methods for data analysis, specifically for decision support and track and trace.	No formal data analysis layer however the architecture does supports track-and-trace functionality. The BRIDGE project is developing enhanced track & trace tools and decision-support tools.	No formal data analysis layer, however such functionality can be integrated into the agents or built on top of the agent model.	No formal data analysis layer.

6 Information architecture comparison

The section above has identified a set of necessary requirements for a PLMIA based on PLM functional requirements and user requirements. The analysis presented in section 5 has illustrated the approach taken by the architectures to achieve the elements necessary for a PLMIA. However, the analysis has not given a comparison of the architectures and has not examined specific details of the architecture designs. The criteria outlined in Table 13 are formulated to clearly differentiate between various designs to highlight their advantages and shortcomings in respect to other architectures being compared. The criteria in Table 13 also take into account the functional needs of the demonstrators outlined in Table 9.

Table 13 Criteria for comparing architecture designs.

Criteria	Description
Vertical flow of information	upwards-only or also involving writing data back to tags etc.
Support for distributed storage/ownership of data	centralized storage / directed routing of data vs. decentralized storage supported by data gathering mechanisms and lookup services and query interfaces.
Access to historical data	Only able to process current events or able to access historical data about objects.
Query/filter processing over a time window	For instance, checking how long a sensor value exceeded a threshold, how long an object remained in a particular location.
Ability to support multiple simultaneous clients with multiple needs	Mechanisms for organizations to 'subscribe' to lifecycle data streams that are of interest to them
Horizontal flow of information	Ability to push and/or pull data, level of detail available
Defines standard interfaces or defines functional components	It is much easier to test for conformance with a standard interface, since the functional implementation does not need to be revealed – the technology simply has to accept valid and correctly formatted input and provide valid and correctly formatted output, in accordance with what the interface defines. Defining standard interfaces also provides technology providers with the option to 'span' across multiple intermediate interfaces (some of which they may choose to support according to the standards) – and also provides end-users with a much stronger assurance of interoperability between solutions from different vendors

The following table provides a comparison of the architectures discussed in this document using the criteria outlined in Table 13.

Table 14 Comparions of the information architectures based on the evaluation criteria outlined in Table 13.

	PROMISE	EPCglobal Network	DIALOG	WWAI
Vertical flow of information	Supports both upwards and downwards flow of information (reading	Currently supports only upward flow of information from the edge of the network.	Does not have a notion of "upward" or "downward" because all nodes are treated	Supports only up-ward flow of information from the edge of the

	PROMISE	EPCglobal Network	DIALOG	WWAI
	and writing to devices). Unlike the EPCglobal Network has support for collection and propagation of sensor data.	Currently has no standardized support for sensor data propagation toward data repositories	equally. Differences between nodes mainly depend on the connectivity, computation capacity and role in the application.	network.
Distributed storage and ownership of data	Centralised data storage and ownership.	Distributed data storage and ownership.	Distributed data storage and ownership.	Distributed data storage and ownership.
Access to historical data	Provides access to historical data collected and assimilated in the PDKM along with built-in analysis tools for decision support unlike the approach in the EPCglobal Network.	Provides access to an object's historical data by providing lookup services pointing to various distributed locations at which an object's data resides. A data analysis layer is being implemented on top of this architecture. Provides good de-coupling between the data analysis layer and the distributed information systems as well as access to various data analysis algorithms that can be selected by the user.	Provides access to an object's historical data by providing lookup information about various distributed locations at which an object's data resides.	Provides access to an object's historical data by providing lookup information about various distributed locations at which an object's data resides.
Query/filter processing of events over a specified time window	No such provisions *<<To be completed after Architecture Series Volume 1 is completed>>*	Middleware can be instructed to perform complex query functions over a time window.	No such provisions	No such provisions
Ability to support multiple simultaneous clients with multiple needs	*<<To be completed after Architecture Series Volume 1 is completed>>*	EPCIS supports standing queries from multiple authorised organisations	Multiple parties may access data from DIALOG nodes.	Multiple parties may access data from nodes on the WWAI network.
Horizontal flow of information	*<<To be completed after Architecture Series Volume 1 is completed>>*	Clients can use ONS and Discovery Services to locate multiple information providers and can then make one-off queries or standing queries to each	Each stakeholder maintains ownership of the data they create or collect. Then data can be exchanged across clients in a standardized way as	The P2P approach of the WWAI architecture allows nodes on the network to access information across different

	PROMISE	EPCglobal Network	DIALOG	WWAI
		information provider for more detailed information, subject to the access controls specified by each information provider, in relation to that particular client or client role, which in turn depends on the business relationship between client and each information provider.	soon as the address of the information service of software agent is known. However, 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. Currently, only mechanisms for file exchange (similar to e-mail attachments) are discussed.	information providers. However each node retains an access policy for determining which other node can access its data.
Defines standard interfaces or defines functional components	Provides the PROMISE Messaging Interface as a data exchange standard and the semantic object model as a data representation standard for PLM.	Layered architecture stack with well defined standard interfaces.	Open-source approach provides access to non-standardised interfaces and message formats. However, the implementation provides support for other interfaces by adding simple protocol adapters.	

7 Action plan for evaluation and refinement of PROMISE standards

PROMISE architecture specifications have to be evaluated and refined to formulate proposals for standardization bodies. A number of tasks have been planned from M36 – M42 to complete the standardisation process for two standards; PMI (as a data exchange standard) and the PDKM Semantic Object Model (as a data representation standard for PLM). The activities are outlined in the following sections.

7.1 Standard for product lifecycle data representation

- **Development of an open source implementation of the PDKM**.

  This will form a reference implementation which will verify the PDKM semantic object model. Documentation on the open source implementation path to the open source PDKM will be the primary outcome of this activity. This activity will facilitate wider industrial community adoption of the standard without getting locked into proprietary software and technologies. This work will be carried out in the next six months and will be presented in DI1.6 due M42.

- **Refinement of the PDKM semantic object model.**

PROMISE PDKM semantic object model needs to be refined based on a final analysis of the eventual new requirements that have arisen in the last year of project activity from the ten PROMISE demonstrators (during which the model was not substantially modified). The refinements will be presented in DI1.6 due M42 in the form of a definitive UML 2.0 class diagram with the associated descriptions.

- **Development of an XML schema representation of the PDKM Semantic Object Model**

  This task will involve the development an XML schema of the UML 2.0 representation of the semantic object model to complement and offer an unambiguous data model for standardisation proposals.

- **Development of a definitive version of the PDKM Semantic Object Model representing for standardisation proposals**.

  This task involves the amalgamation of the definitive version of the UML 2.0 class diagram as well as the XML schema representing the PROMISE PDKM semantic object model to document the PDKM semantic object model specification. This deliverable is to be presented in DI1.6 due M42.

7.2 Standard for product lifecycle data exchange

- **Development of the PMI specification as the data exchange standard**

  The PROMISE product lifecycle data exchange standard will be based on the PMI. This task will deliver the PMI specification necessary for preparing standardisation proposals. The final version of the PMI will appear in Architecture series volume 3 and it will be based on PMI version 3.0 which will be ready by the end of January 2008. This is seen as a critical step in the standardisation process.

8 Conclusion

This report is the second of a three-part series of deliverables aimed to document the activities on evaluation and refinement of PROMISE standards and architecture specifications. In this report we have illustrated the approaches of three prevalent and relevant information architectures and compared them with the PROMISE information architecture. We have also evaluated their ability to meet the requirements for PLM and these are summarised in Table 12 and Table 14. In doing so, we have illustrated the differences between the PROMISE architecture and other architectures.

The comparison also showed areas for improving the PROMISE architecture. In order to achieve inter-organisational information sharing as well as a platform and a system independent architecture it will be necessary to ensure that the PROMISE information architecture has clearly defined layers as in an n-tier layered service oriented architecture. To achieve interoperability it is necessary to ensure that various architectural components that may be supplied by various vendors in a competitive marketplace can be tested for compliance with standard interfaces. Testing for compliance is an important issue to be considered since a certification of compliance to PROMISE standards will need guarantee that systems adhere to PROMISE standards and are interoperable.

The current definition of the PROMISE architecture can be improved by providing more granularity to support multi-vendor competition for developing various components of the information system and to help achieve compliance testing after the standardisation process is completed. Clear definitions of interfaces and the articulation of sections that are mandatory, optional and conditional will greatly improve the compliance testing process.

Issues of data synchronisation have not been considered in the formulation of the architecture and this may prove to be an issue with certain PLM requirements, as is the case with the aerospace industry [12].

Finally, we described the action plan for further evaluation and refinement of PROMISE information system specifications over the next 6 month period. The next deliverable due M42 will report on evaluation and refinement activities carried out in the six month period.

Bibliography

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

[2] Ameri, F. and Dutta, D., Product lifecycle management: closing the knowledge loops. Comput.-Aided Des. Applic., 2005, 2, 577– 590.

[3] CIMdata, Product Lifecycle Management (PLIM) Definition. Available online at: http://www.cimdata.com/PLM/plm.html (accessed 10 November 2007).

[4] EPCglobal web site. Available from: http://www.epcglobalinc.org/home/ (accessed 12 December 2007)

[5] EPCglobal standards. Available from: http://www.epcglobalinc.org/standards/ (accessed 12 Dec 2007)

[6] Dialog System - Distributed Information Architectures for Collaborative Logistics. Available from: http://dialog.hut.fi, (date last accessed: 14/12/2007).

[7] Kärkkäinen M, Ala-Risku T, Främling K. “The product centric approach: a solution to supply network information management problems?”, Comput Ind, Vol. 52(2), pp. 147-159, 2003.

[8] Berners-Lee T, Masinter L, McCahill M. Uniform Resource Locators (URL) - RFC 1738 [http://www.ietf.org/rfc/rfc1738]. RFC: IETF - Internet Engineering Task Force, 1994.

[9] Parlikad AK, McFarlane DC. Investigating the role of product information in End-of-Life decision making. 11th IFAC Symposium on Information Control Problems in Manufacturing. Brazil, 2004.

[10] Harrison M, Parlikad AK, McFarlane DC, et al. “Information management in the product lifecycle - The role of networked RFID”. In: Schoop R,Colombo A,Bernhardt R, et al., eds. 2nd IEEE International Conference on Industrial Informatics (INDIN'04). Berlin: IEEE, 2004; 507-512.

[11] World Wide Article Information (WWAI) web site. Available from: http://www.wwai.org/index.html (accessed 12 Dec 2007).

[12] Suzuki, S., Harrison, M., "Data synchronization specification", Aerospace-ID program report, Auto-ID Labs, University of Cambridge. Available from: http://aero-id.org/mediawiki/.

[13] PROMISE, "Architecture Series Volume 1".

[14] BRIDGE, Work Package 3. Available from: http://www.bridge-project.eu/index.php/workpackage3/en/.

[15] Ranasinghe, D.C., Harrison, M., Cole, P.H., "EPC Network Architecture", In "Networked RFID Systems and Anti-counterfeiting: Raising Barriers to Product Authentication" eds. Cole, P.H., Ranasinghe, D.C., Springer-Verlag, 2007.

[16] Främling, K, Holmstrom, J., Ala-Risku, T., Kärkkäinen, M., "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. 20 p.

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

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

