Information dissemination framework for context-aware products

Sylvain Kubler[b,*], William Derigent[a], Éric Rondeau[a], André Thomas[a], Kary Främling[b]

[a] *Université de Lorraine, CRAN, UMR 7039, Vandœuvre lès Nancy, F-54506, France*
CNRS, CRAN, UMR 7039, Vandœuvre lès Nancy, F-54506, France
[b] *Aalto University, School of Science, Espoo, Finland.*
P.O. Box 15400, FI-00076 Aalto, Finland

Abstract

In recent years, some scholars claimed the usage of intelligent products to make systems more efficient throughout the Product Life Cycle (PLC). Integrating intelligence and information into products themselves is possible with, among others, auto-ID technologies (barcode, RFID,...). In this paper, a new kind of intelligent product is introduced, referred to as "communicating material" paradigm. Through this paradigm, a product is *i)* capable of embedding information on all or parts of the material that it is made of, *ii)* capable of undergoing physical transformations without losing its communication ability and the data that is stored on it. This new material is used in our study to convey information between the different actors of the PLC, thus improving data interoperability, availability and sustainability. Although "communicating materials" provide new abilities compared to conventional products, they still have low memory capacities compared to product databases that become larger and larger. An information dissemination framework is developed in this paper to select the appropriate information to be stored on the product, at different stages of the PLC. This appropriateness is based on a degree of data relevance, which is computed by taking into account the context of use of the product (actor's expectations, environment,...). This framework also provides the tools to split information on all or parts of the material. A case study is presented, which aims at embedding context-sensitive information on "communicating textiles".

Keywords: Product Life Cycle, Data Dissemination, Intelligent Product, Context-aware product, Intelligent Manufacturing System

1. Introduction

New challenges and opportunities arise with concepts such as Internet of Things (IoT) and Ubiquitous/Pervasive Computing [43]. Through these concepts, objects of the real world are linked with the virtual world, thus enabling connectivity anywhere, anytime and for anything. It refers to a world where physical objects and beings, as well as virtual data and environments, all interact with each other in the same space and time [9, 40]. In short, connections are not just people to people or people to computers, but people to things and most strikingly, things to things. Many applications in various sectors exist: medical [5], automobile [10], military application [34], home automation [6, 10], manufacturing [44, 25, 42]. Such applications and environments rely on ever more complex information systems combined with ever increasing data volumes, which are stored in a large number of "Things" (databases, smartphones, RFID tags, sensors,...) [11, 22, 30].

Although widely explored in the Computer Science field, IoT applications in the framework of Material Flow Management (MFM) are still limited [1, 19, 27]. However, such a concept may turn out to be a good strategy [8, 38]. During its PLC, a product moves through numerous companies to various core business sectors, where information is quite often scattered within organizations; it is a matter of adopted materials, applications used to manage technical data (e.g. product data management systems - PDM), applications that manage business and product information (e.g. enterprise resource planning - ERP) or still applications that manage customer information (e.g. customer relationship management - CRM). The major challenges are maintaining the information up-to-date [17, 7] and ensuring interoperability among the different information systems [33]. If one considers the product as an "information vector" (i.e. to which information could be linked), it should contribute to improve data interoperability, availability and sustainability. These contributions have been widely demonstrated over the last decade in the community of Intelligent Manufacturing Systems (IMS) [44, 29, 27]. For instance, research conducted within the PROMISE EU project[1] (2004-2008) showed that systems

*Corresponding author
Email addresses: sylvain.kubler@aalto.fi (Sylvain Kubler), william.derigent@univ-lorraine.fr (William Derigent), eric.rondeau@univ-lorraine.fr (Éric Rondeau), andre.thomas@univ-lorraine.fr (André Thomas), kary.framling@aalto.fi (Kary Främling)

[1] http://promise-innovation.com

using intelligent products are able to gather, process and exchange information throughout their whole life, thereby improving their visibility in the PLC [20, 15, 19] and making systems more efficient and flexible[2]. Accessing to the exact product-related information at any moment and in any location is essential (e.g. a car or an airplane may need to be repaired in any part of the world). In that regard, Meyer et al. [29] claim it is a formidable challenge to link the product-related information to the products themselves. However, in most applications, information is accessed remotely since products only provide a network pointer (e.g. via a RFID tag) to a linked database and a decision making software agent [9]. It may be noted that this kind of products, referred to as "conventional" products in this paper, are somewhat limited on some points:

- *discrete reading*: a specific zone of the product needs to be read (problem of product positioning),
- *risk of tag damage*: if the unique RFID tag is damaged, any product-related information, whether local or remote, is therefore lost. [41] emphasizes the significant proportion of defective tags and false reads which has been as high as 20-50% in some pilot projects,
- *problem of information transfer*: when a product is transformed (e.g. cut), the resulting parts are blank of information,
- *small memory capacity*: RFID tags currently available are memory-constrained (about several Kilo or Megabytes) compared to product databases (several Giga or Terabytes) [16, 36, 2],
- *aggregation level of intelligence*: most of intelligent products do not act as intelligent containers, as explained by Främling et al. [7]. The authors define a product as "intelligent container" when it is able to manage information, notifications and/or decisions about the components that it is made of.

For numerous years and after considering intelligent products as a physical product associated with an informational one (e.g. via auto-ID technologies), a new paradigm was proposed in recent works referred to as "communicating material" [23], which changes drastically the way to see the material. It aims at giving two main abilities to the material:

1. *the ability of being intrinsically and wholly communicating*: even if the product undergoes physical transformation, the resulting pieces shall still be able to communicate. The strong and futuristic idea is to imagine a material communicating at the cellular/molecular level. Let us consider the analogy of a piece of steel (note that steel, at the molecular level, is made of iron and carbon atoms). This piece of steel would be intrinsically communicating if carbon atoms, for example, would be able to communicate,
2. *the ability of managing its data itself*: pushing the paradigm to its extreme, the material should be able to manage its own data according to the events occurring in its environment. For instance, the material could decide itself to propagate/replicate specific data onto different material parts knowing that a physical transformation is scheduled, thus avoiding data loss. Another example would be the mutation of the data when adverse events occurs.

This vision is far from being possible today, especially due to the technological limitations, but some current research seems to be promising. For instance, [24] studied a manner of giving the ability to the material to be intrinsically and wholly communicating, spreading a huge amount of RFID μ-tags into the material. A prototype of communicating textile has been designed with up to 1500 μtags/m^2, making possible the dissemination of item-related information on all or parts of the textile. Although current technologies do not allow building a communicating material up to the molecular level, the designing of new strategies, as those mentioned regarding the second ability, may be addressed. Indeed, the communicating material paradigm opens up many new questions regarding how information should be managed during the PLC [26]; what information is appropriate to be stored on the product [16] (i.e. appropriate to be conveyed between the actors of the PLC)?

Abundant research in the framework of database systems deals with the problem of data distribution [32]. However, solutions developed in this field too often neglect the context of use of the data to assess its appropriateness to be stored on the mobile database/device [4, 31, 18, 3, 35]. This problem is of importance in the framework of PLC since the product environment dramatically changes over its life; the product moves through numerous companies with various core business sectors and many information systems [39]. The appropriate information therefore depends upon a variety of factors (user concerns, product environment,...). Accordingly, this paper develops an information dissemination framework to determine what information is appropriate to be stored on the product itself and that, at each stage of the PLC. This framework is first discussed in section 2. Sections 3 and 4 provide more details about the steps that compose this framework, namely to compute the data appropriateness and then to store/split it on all or parts of the communicating material. Finally, this framework is applied in the context of a textile fabric in section 5, where communicating textiles are used to convey context-sensitive information between the PLC actors.

[2]Flexible means that a product may have abilities to assess/manage itself and to influence the system in which it operates.

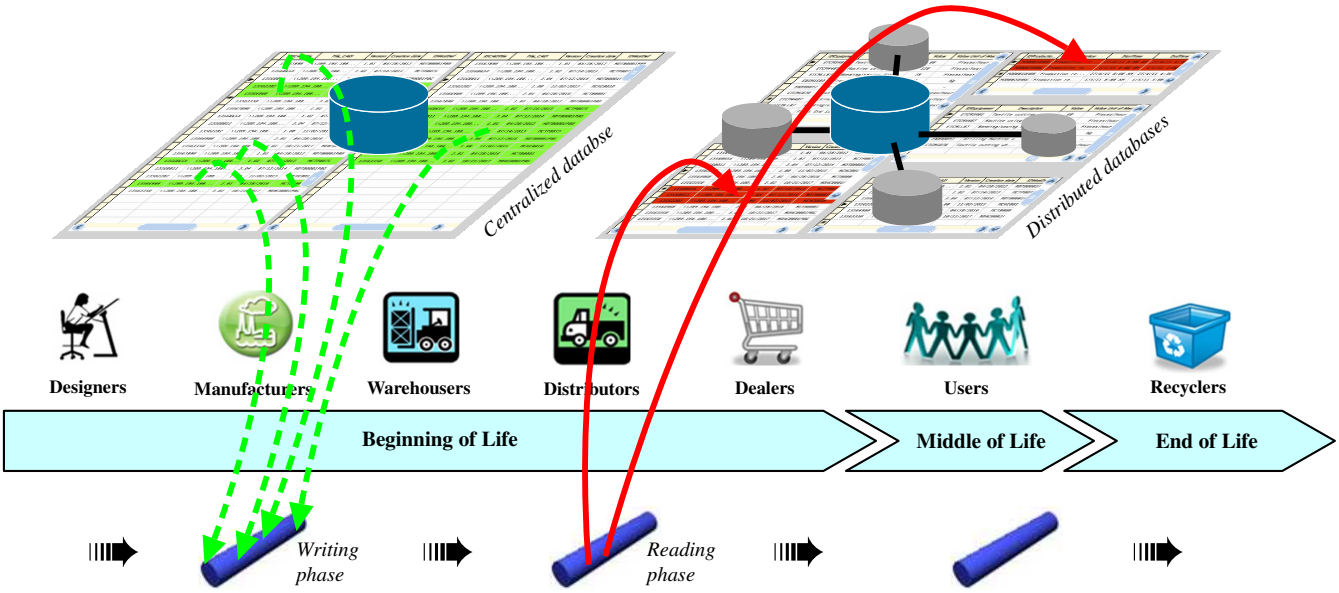


Figure 1: Global view on the information dissemination framework: *writing* and *reading phases*

2. Information dissemination framework

2.1. Context definition

In general, the PLC consists of three main phases as depicted in Figure 2: Beginning of Life (BoL), including design and production, Middle of Life (MoL), including use and maintenance, and End of Life (EoL), including recycling and disposal [20]. In each phase, the actors require specific product-related information (e.g. for traceability purposes, production orders or maintenance orders). As mentioned, products made of communicating material are used to convey appropriate information between the different actors and information systems of the PLC. Two situations may occur:

- *a writing phase*: at one moment of the PLC, product-related information deemed appropriate should be stored on the product (see Figure 2),
- *a reading phase*: later in the PLC, the product-related information carried by the product is retrieved (see Figure 2) and then used/processed (e.g. displayed to users on a smartphone or, re-injected in the user's database).

These two phases form the information dissemination framework, which is described in greater detail in section 2.2.

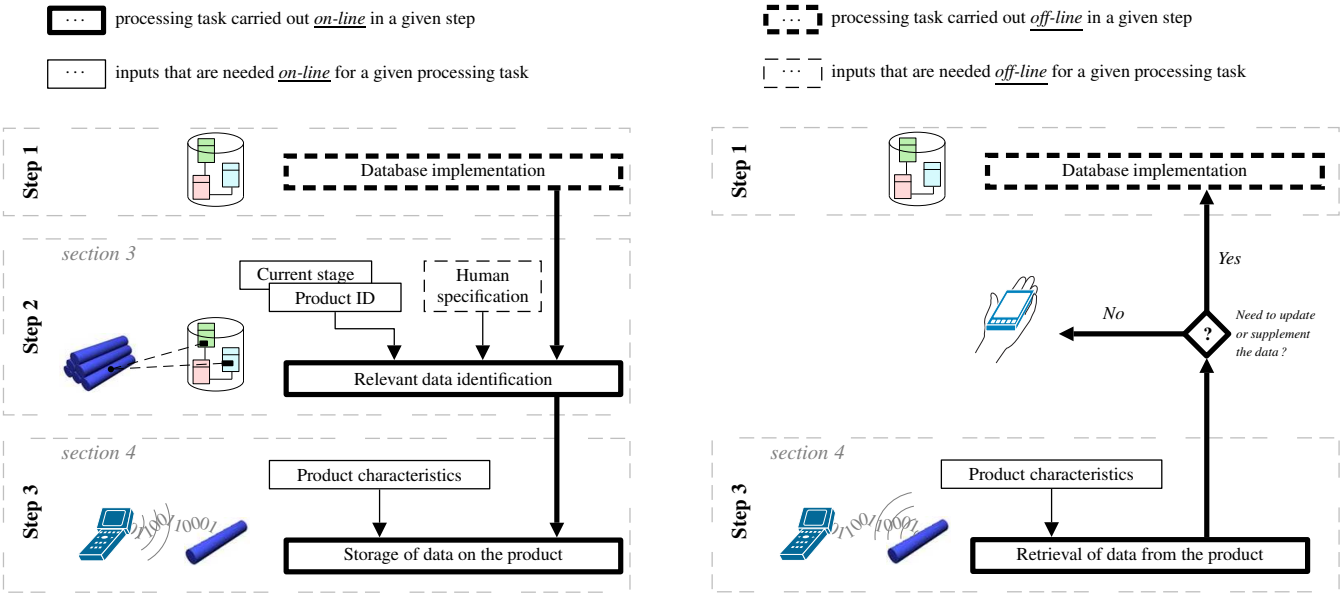
2.2. Framework overview

As mentioned, information systems (PDM, ERP, MES, *etc.*) are often scattered within organizations throughout the PLC. Many standards such as IEC 62264, B2MML, ISA-88 have emerged to achieve integration and interoperability of these systems [21, 33]. These standards are, in fact, Logical Relationship Models (LDM) which give rise, once implemented, to relational databases. As explained previously, the *writing phase* identifies the appropriate data from these databases, which is then stored on the product. The *writing phase* consists of 3 steps as detailed in Figure 2(a), while the *reading phase* only requires one of these steps (step 3), or two at the most[3], as depicted in Figure 2(b). The purpose of each process step, the inputs needed for each one and the type of execution (*on-line* or *off-line*) are described hereafter. A step performed *on-line* or an input provided *on-line* means that the set of operations/computations are carried out when the product is physically present. A step performed *off-line* or an input provided *off-line* means that the set of operations/computations are carried out when the product is out-off process.

Process step 1 consists in implementing the database system architecture. Many works through the literature help the designer to choose the most suitable system according to his expectations and the application constraints [32]. Process step 1 is logically performed *off-line* (i.e. before the system initialization).

Process step 2 consists in selecting the appropriate data from the database. More concretely, a quantitative approach is used to define a degree of data relevance, which indicates how useful the data is for the subsequent actors and activities, and consequently how useful it is to be stored on the product. This quantitative approach is based on a multi-criteria decision making study and it requires as inputs:

[3] One step is required to only access the data carried by the product. Two steps are required when data needs to be updated in the database or supplemented by other data from the database (e.g. to get answerable queries).



(a) Steps composing a *writing phase*

(b) Steps composing a *reading phase*

Figure 2: Global view on the information dissemination framework: *writing* and *reading phases*

- the access to all data of the database at a given moment of the PLC (i.e. at the *writing phase*),
- human specifications, defined *off-line* by experts,
- the current PLC stage in which the product is handled (provided *on-line*),
- the product ID (provided *on-line*).

Process step 3 deals with the storage of the data (selected across step 2) on the product and the manner to retrieve it[4]. Step 3 is carried out *on-line* and it requires as inputs:

- the set of data identified through step 2,
- some product characteristics (e.g. the type of RFID technology).

This paper mainly focuses on process steps 2 and 3, based on an existing database. These two steps are respectively described in sections 3 and 4.

3. Step 2: Relevant data identification

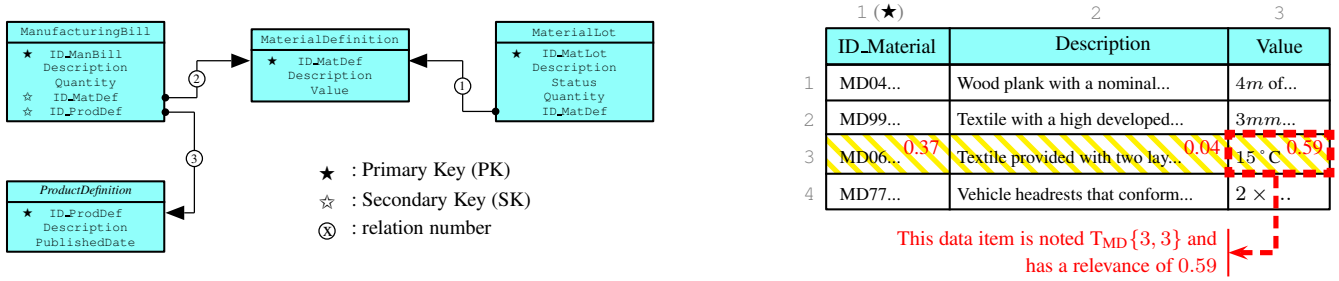
The quantitative approach developed in this paper uses the LDM. Figure 3(a) gives insight into a part of such a LDM. A given entity of LDM corresponds, once implemented, to a relational table as shown in Figure 3(b) with `MaterialDefinition`. The attributes listed for each entity correspond to the table columns, each table row is referred to as a tuple/instance of the relation and one table cell is called a data item. Our approach consists of two stages:

A. identification of all product-related information through all tables. In other words, it consists in searching all tuples which are somehow in relation with the product ID. The product ID (denoted d_p in this paper) is provided by the product when reaching a *writing phase* and it refers to a tuple of a given table. Figure 3(b) gives the example where only tuple 3 of `MaterialDefinition` is identified as a product-related tuple (represented by a hatched background),

B. assessment of the degree of relevance of the product-related information. This degree indicates whether an information must be stored on the product. The higher the degree value, the higher is the necessity to store this information on the product. In our study, the quantitative model developed by Chan and Roddick [4] is implemented to compute this degree, which is computed for each product-related information identified in stage A. Their approach is interesting because the authors try to match the context with information in order to select context-sensitive information. This degree is actually computed at the level of the data item. Figure 3(b) shows the relevance degree of each data item of tuple 3. In this example, the data item located at row 3, column 3 in `MaterialDefinition`, noted[5]

[4]In our study, products are made of communicating material but the quantitative approach defined in process step 2 may be used regardless the type of intelligent product.

[5]`MD` is used as abbreviation of `MaterialDefinition` in the paper.



(a) Example of a Logical Data Model (LDM)

(b) Table `MaterialDefinition` & Data item relevance

Figure 3: View of a Logical Data Model (LDM) and a relational table

$T_{MD}\{3,3\}$, is the most relevant with a degree of 0.59. It therefore should be stored in priority on the product.

The set of tools developed in stages A and B are respectively introduced in sections 3.1 and 3.2.

3.1. Protocol for identifying product-related information

In this section, the goal is to identify all tuples from all tables which are somehow related to the product ID (d_p). On the one hand, possibilities of using classical database queries for such an identification are explored and then, a new approach is introduced.

With traditional queries, it is necessary to define the first query to retrieve d_p and then, to use results of that query to build dynamically the next queries so as to explore neighboring tables in the LDM, which themselves give rise to new queries and so on (until exploring the entire LDM). In Structured Query Language (SQL), building queries referring to results of other queries can be done thanks to two mechanisms which are "sub queries" or "recursive queries". Sub queries [12] are SQL statements embedded within another and allow executing series of related queries in a single SQL code. However, this mechanism is not adapted to our case because the higher the number of tables, the higher the number of sub queries. Moreover, any change in the LDM (e.g. new table, attribute) would make it impossible to build a generic algorithm based on sub queries. The second mechanism uses recursive queries, as defined in the SQL:1999 standard [28]. As an example, SQL recursive queries are formed using a Common Table Expression (CTE), which is a temporary table with the significant advantage of being able to reference itself, thereby creating a recursive table. A recursive table is one in which an initial CTE is repeatedly executed to return subsets of data until the complete result set is obtained. Returning hierarchical data is a common use of recursive queries, for example *i)* displaying employees in an organizational chart or, *ii)* displaying data in a bill of materials scenario in which a product is made of one or many components which are, in turn, made of subcomponents or are parents of other components and so on. Our data identification process is somehow recursive because it repeats the same queries until no additional product-related tuples are found. However, these queries are not repeated on the same table, which prevents the use of such a mechanism. The last possibility is to use programming using Database Automation Programming Languages like JDBC (Java DataBase Connectivity). This last solution makes it possible to develop algorithms able to use each query output result in order to explore dynamically the entire LDM. Based on this approach, we propose to extract when the time comes (i.e. at each *writing phase*), the set of tables $\mathcal{T}$ from the database in a matrix format and then, to explore them thanks to an algorithm named `RetrievalData`. This algorithm is given in Algorithm 1 and relies on two sub-functions: `ExplorePK`, `ExploreFK`. All variables used in these algorithms are listed in Table 1.

`RetrievalData` retrieves all tuples which are somehow related to d_p by moving from table to table included in $\mathcal{T}$. To do so, it is necessary to identify the relations between tables based on the primary keys (PKs) and foreign keys (FKs)[6]. This functionality is provided by a sub-function named `ExplorePK` (see Algorithm 2). This sub-function identifies the FKs from the table which is currently explored, and moves to the corresponding tables (i.e. where FKs are PKs) to identify new product-related tuples. Let us note that `ExplorePK` may not be sufficient to explore all tables because some relations may not be reached. Indeed, when a table has no FK, it is necessary to proceed in reverse. This means searching relations where PKs from tables already explored are FKs in tables not explored yet. This is achieved by the second sub-function, named `ExploreFK`, which is given in Algorithm 3. In order to make the understanding easier, the algorithm steps are detailed on a specific case in section 5.

3.2. Assessment method

Chan's approach [4] uses the notion of priorities to select context-sensitive information. The priorities are numerical values either supplied or generated through observation and experimentation and are assigned through a multifaceted evaluation of different criteria (8 in total).

[6] One FK is PK in another table and vice versa.

Table 1: Variable definition used in Algorithms 1, 2 and 3

Variables	Description
$\mathcal{T} = [\mathrm{T}_1..\mathrm{T}_t..\mathrm{T}_z]$	set of tables in the database where T_t represents the table t and z indicates the number of tables extracted from the LDM
$\mathrm{T}_t\{i,j\}$	represents the data item located at row i, column j from T_t, with $i \in \{1, m_t\}$ and $j \in \{1, n_t\}$, where m_t is the number of tuples and n_t is the number of attributes in T_t
$\texttt{cont}(\mathrm{T}_t\{i,j\})$	indicates the content of $\mathrm{T}_t\{i,j\}$. $\texttt{cont}(\mathrm{T}_t\{i_1,j_1\}) = \texttt{cont}(\mathrm{T}_{t'}\{i_2,j_2\})$ means that contents of both data items are identical
$\texttt{value}(\mathrm{T}_t\{j\})$	indicates the name of column j from T_t. $\texttt{value}(\mathrm{T}_t\{j_1\}) = \texttt{value}(\mathrm{T}_{t'}\{j_2\})$ means that names of attributes j_1 and j_2 from T_t and $\mathrm{T}_{t'}$ respectively are identical (e.g. when a primary key - PK, is a foreign key - FK)
$\mathcal{K} = [\mathrm{K}_1..\mathrm{K}_t..\mathrm{K}_z]$	set of keys related to the LDM, where K_t is a vector giving the key type of each attribute of T_t. An element of K_t is noted $\mathrm{K}_t\{j\} \in \{0,1,2\}$, where 0 means that the attribute j of T_t is neither a PK nor a FK, 1 means that it is a PK and 2 means that it is an FK
$d_p = \mathrm{T}_p\{i_p, j_p\}$	starting point of the algorithm such that $K_p\{j_p\} = 1$. d_p is provided by the machine/device used when writing/reading the product
$\mathcal{R} = [\mathrm{R}_1..\mathrm{R}_t..\mathrm{R}_z]$	set of matrices, where R_t represents an *image* of the T_t, indicating if data items are product-related information. A data item in R_t is noted $\mathrm{R}_t\{i,j\} = true$ if it is a product-related information and $\mathrm{R}_t\{i,j\} = false$ otherwise
D	Matrix giving the distances between tables, where $D_{a,b}$ is the minimal distance in term of relations between tables T_a and T_b
$\mathcal{E}$	Set of tables already explored in the LDM
$\mathcal{N}$	Set of tables, yet to be explored, which are linked to tables already explored
$\mathcal{I}$	Set of data-items that remain to be explored (I_l is the lth element of I). $\mathcal{I}'$ is a subset of $\mathcal{I}$

Algorithm 1: `RetrievalData`($\mathcal{T}$, d_p)

output: $\mathcal{R}$

```
begin
    I ⟵ d_p;                                   // Initialization of I with the data item referenced by the product
    E ⟵ ∅;                                     // The list of tables already explored is set to empty
    while I ≠ ∅ do                             // Table exploration is carried on as long as I ≠ ∅, after the explorations based on PK and FK
        while I ≠ ∅ do                         // Table exploration is carried on as long as I ≠ ∅, after the exploration based on PK
            [I'; R] ⟵ ExplorePK(T, R, I_1);    // Data items in relation with the product (with the 1^st element of I here) are
            selected and their images are set to true in R. Those not explored yet are identified and added to I'
            I ⟵ I ∪ I';                        // Data items not explored yet are added to the ones already present in I
            E ⟵ E ∪ I_1;                       // We memorize that the table including I_1 was explored
            I ⟵ I − I_1;                       // The data item used for the exploration (i.e. I_1) is removed
        I ⟵ ExploreFK(E,T)                     // New relations (i.e. new data items) are searched in tables not explored yet
```

In our approach, we implement 3 of the 8 criteria that we consider as the most appropriate in the framework of PLC applications. For each criterion, the calculation of a relative priority ρ_x (with x a criterion) and an assigned priority $\phi_x(l)$ ($l = \{1, \ldots, n\}$, n being the number of product-related data items identified in stage A) are performed. The first priority (relative) indicates how important the criterion x is over the others. The second one (assigned) corresponds to the priority value (between 0 and 1) computed for each data item l with respect to criterion x. Both priorities are then combined as in equation 1, which provides the relevance degree of the data item l, noted $R(l)$. The size of a data item l is noted size(l) (expressed in bytes). This formula shows that the higher the priorities ρ_x and $\phi_x(l)$, the higher is the data item relevance. Finally, data items are classified in order of relevance (according to $R(l)$).

$$R(l) = \frac{\sum^x \rho_x.\phi_x(l)}{ln\,(\text{size}(l))} \quad \text{size}(l) > 1, \quad x = \{e, c, m\} \qquad (1)$$

The three criteria considered in our study are:

1. *Enumeration* - C_e: this criterion allows the user to enumerate information he considers as important to be stored on the product,

Algorithm 2: `ExplorePK`($\mathcal{T}$, $\mathcal{R}$, $\mathrm{T}_t\{i,j\}$)

output: $\mathcal{I}'$, $\mathcal{R}$

begin
- $\mathcal{I}' \longleftarrow \emptyset$; // The list which contains the data items remaining to be explored is set to empty
- **forall the** $j_1 \in \{1, n_t\}$ **do** // For all attributes j_1 from T_t (i.e. from the table that contains $\mathrm{T}_t\{i,j\}$) do
  - $\mathcal{R}_t\{i, j_1\} \longleftarrow true$; // Set to $true$ each product-related data item that composes the tuple i
  - **if** $\mathcal{K}_t\{j_1\} = 2$ **then** // If the attribute j_1 from T_t is a FK, then do
    - **forall the** $t' \in \mathcal{T}, j_2 \in \{1, n_{t'}\} | \mathtt{value}(T_t\{j_1\}) = \mathtt{value}(T_{t'}\{j_2\})$ & $\mathcal{K}_{t'}\{j_2\} = 1$ **do** // All tables $\mathrm{T}_{t'} \in \mathcal{T}$ are explored to find in which table and for which attribute $j_2 \in \mathrm{T}_{t'}$, the FK j_1 in $\mathcal{T}_t$ is PK
      - **forall the** $i_2 \in \{1, m_{t'}\} | \mathtt{cont}(T_t\{i, j_1\}) = \mathtt{cont}(T_{t'}\{i_2, j_2\})$ **do** // If j_2 is identified, all tuples from table $\mathrm{T}_{t'}$ are explored in order to find whether the content of any data item from column j_2 (i.e. column PK from $\mathrm{T}_{t'}$) is equal to the one from the initial table (i.e. data item noted $\mathrm{T}_t\{i, j_1\}$)
        - $\mathcal{I}' \longleftarrow \mathcal{I}' \cup \mathrm{T}_t\{i_2, j_2\}$; // If the condition is validated, it means that $\mathrm{T}_t\{i_2, j_2\}$ corresponds to a data item that must be further explored. As a result, it is added to $\mathcal{I}'$

Algorithm 3: `ExploreFK`($\mathcal{E}$, $\mathcal{T}$, $\mathcal{R}$)

output: $\mathcal{I}$

begin
- $\mathcal{I} \longleftarrow \emptyset$; // The list which contains the data items remaining to be explored is set to empty
- $\mathcal{N} \longleftarrow \emptyset$; // The list of tables not explored yet which are in direct relation with tables already explored is set to empty
- **forall the** $\mathrm{T}_t \in \mathcal{E}, \mathrm{T}_{t'} \in \mathcal{T} | D_{\mathrm{T}_t, \mathrm{T}_{t'}} = 1$ & $\mathrm{T}_{t'} \notin \mathcal{E}$ **do** // We search for tables $\mathrm{T}_{t'}$ which have not been explored (i.e. $\mathrm{T}_{t'} \notin \mathcal{E}$) and are distant of one relation from the tables T_t already explored (i.e. $\mathrm{T}_t \in \mathcal{E}$). This is noted $D_{T_t, T_{t'}} = 1$
  - $\mathcal{N} \longleftarrow \mathcal{N} \cup \mathrm{T}_{t'}$; // If the condition is validated, $\mathrm{T}_{t'}$ is then added to $\mathcal{N}$
- **forall the** $\mathrm{T}_t \in \mathcal{N}, j_1 \in \{1, n_t\} | \mathcal{K}_t\{j_1\} = 2$ **do** // For all attributes j_1 from tables T_t that compose $\mathcal{N}$, we search attributes j_1 that are FK (i.e. $\mathcal{K}_t\{j_1\} = 2$)
  - **forall the** $\mathrm{T}_{t'} \in \mathcal{E}, j_2 \in \{1, n_{t'}\} | \mathtt{value}(\mathrm{T}_t\{j_1\}) = \mathtt{value}(\mathrm{T}_{t'}\{j_2\})$, $\mathcal{K}_{t'}\{j_2\} = 1$ **do** // If j_1 and T_t are identified, we search in which tables already explored $\mathrm{T}_{t'} \in \mathcal{E}$ and for which attributes $j_2 \in \mathrm{T}_{t'}$, the FK $j_1 \in \mathrm{T}_t$ is PK
    - **forall the** $i_1 \in \{1, n_{t'}\} | \mathcal{R}_{t'}\{i_1, j_2\} = true$ **do** // If $\mathrm{T}_{t'}$ and j_2 are identified, all tuples from table $\mathrm{T}_{t'}$ are explored in order to find whether the content of the data item located at row i_1, column j_2 from $\mathrm{T}_{t'}$ is a product-related data item (i.e. whether $\mathcal{R}_{t'}\{i_1, j_2\} = true$)
      - **forall the** $i_2 \in \{1, m_t\} | \mathtt{cont}(\mathrm{T}_t\{i_2, j_1\}) = \mathtt{cont}(\mathrm{T}_{t'}\{i_1, j_2\})$ **do** // If $\mathcal{R}_{t'}\{i_1, j_2\}$ is a product-related data item, we test if the content of $\mathrm{T}_{t'}\{i_1, j_2\}$ (i.e. the PK $\mathrm{T}_{t'}$) is present in column j_1 from table T_t
        - $\mathcal{I} \longleftarrow \mathcal{I} \cup \mathrm{T}_t\{i_2, j_1\}$; // If the condition is validated, $\mathrm{T}_t\{i_2, j_1\}$ is added to $\mathcal{I}$

2. *Contextual* - C_c: experts may not be aware of all the data needed by the downstream actors of the PLC and could omit important classes of information. Indeed, the different information systems over the PLC (CAD, PDM, CRM,...) are not concerned by the same data (i.e. the same entities from the LDM) and the relevance of the entities should therefore change according to the product location in the PLC. This criterion involves a group of experts, issued from a consortium of networked enterprises or from standards organizations, which aims at assessing the importance of different groups of information in the LDM according to the PLC phases,

3. *Model-based* - C_m: this criterion favors the storage of data close to the product's table[7]. Indeed, as explained by Chan and Roddick, the shorter the distance between tables, the higher the data correlation.

The following sections detail how to adjust and to compute the relative priorities $\rho_x \ \forall \ x = \{e, c, m\}$ and the assigned priorities $\phi_x(l) \ \forall \ l = \{1, \ldots, n\}$.

3.2.1. Adjustment of the relative priorities ρ_x

Regarding the priority ρ_x, it is necessary to specify the importance of each criterion at a given stage of the PLC. In our study, the decision maker performs pairwise comparisons between criteria as in equation 2, the importance

[7] The product's table is defined as that which includes d_p.

of criterion x over criterion x' being noted $s_{xx'}$. This evaluation is based on the 1 to 9-point scale designed by Saaty [37]: $\{1, 3, 5, 7, 9\}$. $s_{xx'} = 1$ means that criteria x and x' are of equal importance and $s_{xx'} = 9$ means that criterion x is strongly favored over criterion x'. The relative importance of a criterion x, noted $\lambda(\mathrm{C}_x)$, is computed as in equation 3. All relative criteria importances are then synthesized by the vector Λ_ρ in equation 4.

$$\begin{array}{c} \\ \mathrm{C}_e \\ \mathrm{C}_c \\ \mathrm{C}_m \end{array} \begin{array}{c} \begin{array}{ccc} \mathrm{C}_e & \mathrm{C}_c & \mathrm{C}_m \end{array} \\ \begin{bmatrix} s_{ee} & s_{ec} & s_{em} \\ s_{ce} & s_{cc} & s_{cm} \\ s_{me} & s_{mc} & s_{mm} \end{bmatrix} \end{array} s_{xx'} = \begin{cases} 1 & x = x' \\ s_{x'x}^{-1} & x \neq x' \end{cases} \tag{2}$$

$$\lambda(\mathrm{C}_x) = \frac{\sum_{k=\{e,c,m\}} s_{xk}}{\sum_{k=\{e,c,m\}} \sum_{l=\{e,c,m\}} s_{kl}} \tag{3}$$

$$\Lambda_\rho = \begin{array}{c} \begin{array}{ccc} \rho_e & \rho_c & \rho_m \end{array} \\ \begin{bmatrix} \lambda(\mathrm{C}_e) & \lambda(\mathrm{C}_c) & \lambda(\mathrm{C}_m) \end{bmatrix} \end{array} \tag{4}$$

3.2.2. Adjustment of the assigned priorities $\phi_x(l)$

This section details the priority computations $\phi_x(l)$ with regard to each criterion: C_e, C_c and C_m.

i. C_e: this criterion gives a certain freedom to users to select information they deem relevant to be stored on the product. Concretely, users select the class attributes they consider useful. Let $\mathrm{T}_t(v)$ be an attribute of table t. The score of this attribute, noted $s(\mathrm{T}_t(v))$ is equal to 1 if the user enumerates $\mathrm{T}_t(v)$, 0 otherwise, as formalized in equation 5. If a data item l belongs to the attribute $\mathrm{T}_t(v)$, its score with respect to C_e, noted $\phi_e(l)$ is therefore equal to $s(\mathrm{T}_t(v))$.

$$s\left(\mathrm{T}_t\left(v\right)\right) = \begin{cases} 1 & \text{enumerated} \\ 0 & \text{not enumerated} \end{cases} \tag{5}$$

ii. C_c: As mentioned, it is necessary to identify important data with regard to the global PLC. This evaluation cannot be focused on each data item, as in C_e, but rather on a group of tables. Accordingly, the main idea is to cluster all entities of the LDM needed by one information system in an "entity group". Figure 4 gives an example of a LDM in which four entity groups have been defined by the experts. Experts then performed evaluations for each group depending on their utility for the current and downstream PLC stages. In our study, experts perform pairwise comparisons between entity groups as in equation 6, with z the number of entity groups. The importance of group i over group j is noted s_{ij} and is based on the Saaty's scale. The relative importance of a group i, noted $\lambda(\mathrm{G}_i)$, is computed as in equation 7. All relative entity group importances are then synthesized by the vector Λ_c in equation 8.

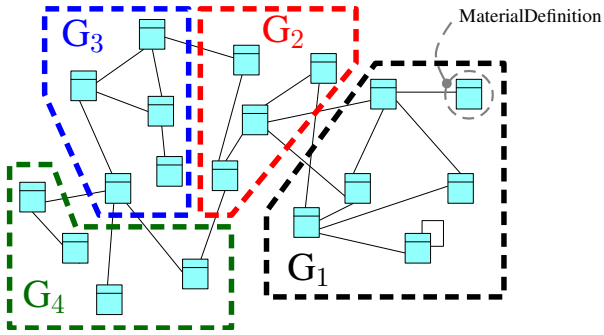


Figure 4: Entity groups in the LDM

$$\begin{array}{c} \\ \mathrm{G}_1 \\ \vdots \\ \mathrm{G}_z \end{array} \begin{array}{c} \begin{array}{ccc} \mathrm{G}_1 & \cdots & \mathrm{G}_z \end{array} \\ \begin{bmatrix} s_{11} & \cdots & s_{1z} \\ \vdots & \ddots & \vdots \\ s_{z1} & \cdots & s_{zz} \end{bmatrix} \end{array} s_{ij} = \begin{cases} 1 & i = j \\ s_{ji}^{-1} & i \neq j \end{cases} \tag{6}$$

$$\lambda(\mathrm{G}_i) = \frac{\sum\limits_{k=\{1,..,z\}} s_{ik}}{\sum\limits_{k=\{1,..,z\}} \sum\limits_{l=\{1,..,z\}} s_{kl}} \tag{7}$$

$$\Lambda_c = \begin{bmatrix} \lambda(\mathrm{G}_1) & \ldots & \lambda(\mathrm{G}_z) \end{bmatrix} \tag{8}$$

iii. C_m: this criterion is based on the relationships implied through the LDM since the goal is to favor data close to the product's table. First, it is necessary to compute distances between the product's table A and any other table B. The distance corresponds to the shortest path to reach B from A (i.e. the number of relations which separate them). For instance, let `MaterialLot` be the product's table A and `ManufacturingBill` be the table B in Figure 3(a). The distance between both tables is equal to 2 (relations ①-②). Since the product's table is at the centre of our concerns, $\phi_m(l)$ would then decrease as the modeled distance increases. Chan and Roddick define the equation 9, with $k \in [1; \infty[$ a constant adjusted by the expert, and $a \in \mathbb{N}$ the distance. Figure 5 shows that distant information is more favored for small values of k than for high values, and reciprocally (see $k = 1.01$ and 1.08). It is therefore necessary to study the entire LDM to fix values for k. Indeed, maximum distances within the LDM of 10 or 200 will certainly lead to different values for k.

$$\phi_m(l) = k^{-a} \tag{9}$$

3.3. Synthesis

In resume, when the machine/operator needs to store data on the product, all product-related data items from the database are identified via Algorithm 1 and then assessed in term of relevance via equation 1. Data items are

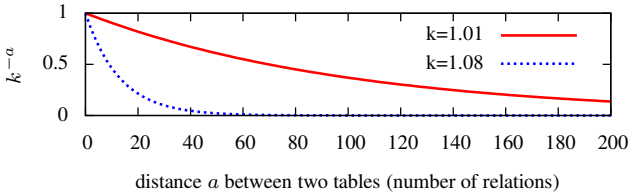


Figure 5: Adjustment of coefficient k

then ranked in order of relevance and information with the highest relevance is stored on the product made of communicating material (according to the available memory space). The next section details how information is split on all or parts of the material.

4. Step 3: Data storage/retrieval using "communicating materials"

Process step 2 of the information dissemination process deals with the storage of data items on the communicating material. A RFID tag may store more or less information according to the available memory on the tag and thus, one data item may require more memory space than that available in one tag. The idea is therefore to split the set of data items among several tags that compose the communicating material. To do so, a specific application protocol, named *splitting protocol*, is developed in our study. This protocol defines a *header* that is added in each RFID tag to know in which order the data items are split. This header respects the RFID standard ISO/IEC 18000-1 [13] and is detailed in Appendix A. It requires in each RFID tag 16 bytes out of the n writable. In this section, we show how the data item $\mathrm{T_{MD}}\{3,3\}$ (*cf.* Figure 3(b)) is split via the *splitting protocol* and then stored among several tags that compose a communicating textile (see Figure 6).

The RFID tags that compose the communicating textile are Read/Write tags with a memory of 34 bytes. Regarding $\mathrm{T_{MD}}\{3,3\}$, the character string that must be split is composed of both the *index* (to know in which table, attribute and tuple is located $\mathrm{T_{MD}}\{3,3\}$) and the content of the data item (i.e. the value "15°C"). This character string therefore consists of 41 characters as emphasized in Figure 6, knowing that 1 ASCII character is encoded on 1 byte. These 41 bytes are added to the 16 bytes of the header required by each RFID tag. Accordingly, three tags are needed to split $\mathrm{T_{MD}}\{3,3\}$ as calculated in equation 10.

$$\left\lceil \frac{\overbrace{41}^{\text{index+content (bytes)}}}{\underbrace{34}_{\text{tag memory (bytes)}} - \underbrace{16}_{\text{header (bytes)}}} \right\rceil = 3 \qquad (10)$$

Figure 6 shows how $\mathrm{T_{MD}}\{3,3\}$ is split among three RFID tags a, b and c. These tags have[8] :

- the same *Protocol* value (see tag c for the name of each header field), $(255)_{10} = (\mathrm{FF})_{16}$, because the same header structure is used, i.e. the one defined for the *splitting protocol* which is identified by the value $(255)_{10}$,
- the same *Size* value, $(34)_{10} = (12)_{16}$, because tags a, b and c are of the same technology: memory of 30 bytes (see tag a: $32 \times 8.5 = 272$ bits $= 34$ bytes),
- a *Seq_Num* value which provides the splitting sequence. The first part of the character string is stored in tag a, *Seq_Num* $= (1)_{10} = (01)_{16}$, the second one in tag b, *Seq_Num* $= (2)_{10} = (02)_{16}$, and the last one in tag c, *Seq_Num* $= (3)_{10} = (03)_{16}$,
- a *Prev_Num* value which provides the sequence number of the previous RFID tag. For instance, *Prev_Num* in tag b is equal to $(01)_{16}$ (i.e. the *Seq_Num* of tag a). *Seq_Num* and *Prev_Num* of tag a are identical, i.e. $(01)_{16}$, because tag a is the first written tag,
- the same *ID_Phase* value because data have been written at the same time. The value $(1344940753)_{10} = (00000000502a2ad1)_{16}$ represents the *timestamp* which translates into the date : 08/14/2012 at 13:05:52,
- the checksum value (checksum calculations are not presented in this example),
- a part of the character string related to $\mathrm{T_{MD}}\{3,3\}$ (*index* + *content*). Each character composing this string is encoded in ASCII as shown in Figure 6 (e.g. letter `M`=4D, letter `a`=61,...). It can be noted that the index requires the full memory of tags a, b and a piece of tag c. The content of $\mathrm{T_{MD}}\{3,3\}$ (i.e. 15 °C) is then stored at the end of tag c.

According to the set of data items stored on the product, queries may be answered or unanswered. Nonetheless, some methods can be deployed to know in advance (i.e. before data items are split over the product) whether queries may be answered or not (e.g. by transforming queries to corresponding bitmaps [4]).

5. Case study

In this section, the focus is on a textile fabric that produces raw products conditioned in "textile reels", which are used for various purposes: vehicle upholstered chair seat manufacturing, cloth making, *etc.* The textile pass through several coordinated activities and is processed/transformed at many steps throughout its manufacturing life cycle. Many supply chain members have noted points about the textile manufacturing process management that need to be improved, like:

- *tracking issues*: many cases of counterfeiting of clothes, sheets (made from textile reels) have occurred in recent years. Moreover, many defective tags and false reads have been noted,

[8] For a generic description of each header field, please refer to Appendix A.

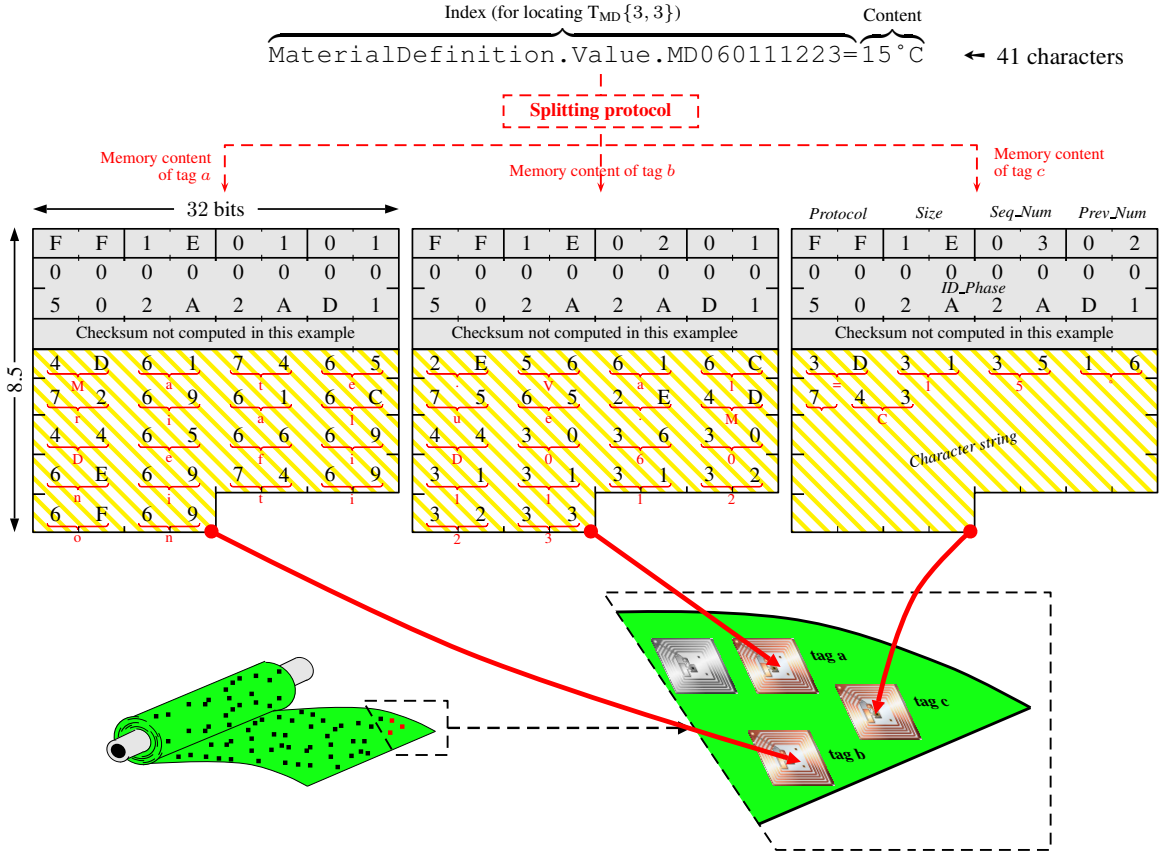


Figure 6: Illustration of how $T_{MD}\{3,3\}$ is split over a "communicating textile" via the *splitting protocol*

- *transformation/processing issues*: product-related information is often lost when textile is transformed (e.g. trimming operation),
- *interoperability issues*: various supply chain members think it would be judicious to link textile-related information to the textile itself for improving the product visibility and data interoperability. Indeed, some actors cannot access specific information owned by supply chain pairs,
- *aggregation level of intelligence*: over its life, the textile is dissociated (e.g. cutting operation), assembled (e.g. sewing, pasting operation) with other materials or products. If communicating textiles are used, then, they could act as *intelligent containers* (*cf.* section 1).

As a consequence, the supply chain members think it would be judicious and beneficial to reconsider the current tracking, managing and interoperability system related to textile reels. They propose replacing classical textiles with communicating textiles. Thus, product-related information could be linked to the products themselves, whatever the nature of the product made of communicating textile (e.g. textile pieces, clothes, sofa, upholstered chair seat). Jointly with the use of communicating textile, the information dissemination framework is implemented in order to select context-sensitive information at different instants of the PLC (i.e. information that should be stored on products). It is therefore necessary to identify where *writing phases* must be implemented in the PLC and then, where information must be retrieved.

In this case study, the part of PLC of the communicating textile reels is strongly simplified as shown in Figure 7, where communicating textiles have already been designed in stage 1. In our application, the communicating textile prototype designed in [24] is used. Figure 8 shows such a textile currently being manufactured.

The communicating textile coming from stage 1 is cut in stage 2 (*cf.* *cutting operation* in Figure 7) before being pasted on wood planks (*cf.* *pasting operation*), which have been painted beforehand (*cf.* *painting operation*). The *pasting operation* gives rise to upholstered chair seats which are intended for subsequent PLC stages. We assume that two *writing phases* are defined: at the end of stage 1 and at the end of stage 2 (e.g. before leaving a supply chain member). Moreover, the actor related to the *cutting operation* desires to read information that has been stored on the communicating textile. Initially, wood planks are not made of communicating material and thus, cannot get the most out of this concept. However, when communicating textile pieces are pasted on wood planks, the resulting products (i.e. upholstered chair seats) are able to embed

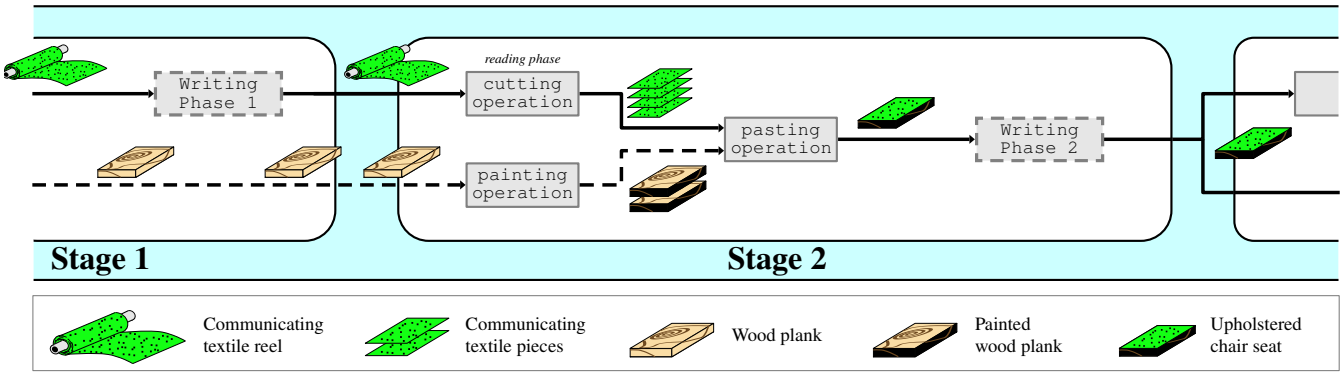


Figure 7: Sequence of activities related to a textile application using "communicating textile" reels

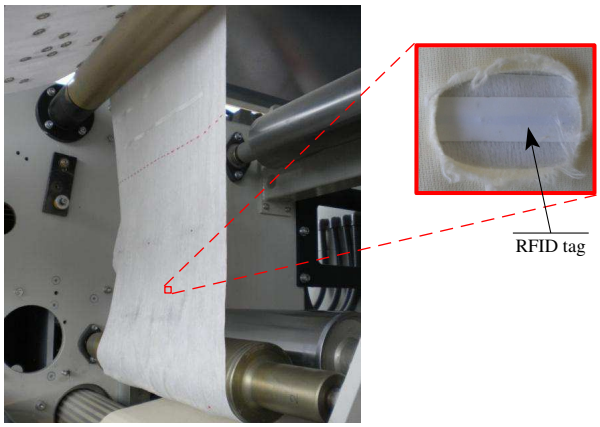


Figure 8: Prototype of "communicating textile" designed in [24]

wood plank-related information using the part made of communicating textile. Obviously, this is only possible if a *writing phase* is subsequently defined, which is the case in stage 2. In resume, considering this simplified part of PLC, the information dissemination framework developed in this paper must be implemented in three activities, namely:

- at two *writing phases*: one defined at the end of stage 1 and one at the end of stage 2,
- at the *cutting operation*: communicating textiles must be read in this activity. Consequently, the *reading phase* must be implemented.

In this study, the database is already implemented. It relies on a part of B2MML and consists of 19 entities. Accordingly, only process step 2 and 3 of these *writing* and *reading* phases are presented in sections 5.1 to 5.3.

5.1. Writing phase: Stage 1

Specifications made by the different experts (needed in process step 2) are detailed in the following order:

i. C_e: the expert(s) enumerate information they deem important to store on textile reels,

ii. C_c: the experts define the entity groups and adjust their importance over the PLC,

iii. C_m: the expert(s) adjust the coefficient k,

iv. Criteria: the expert(s) define the importance of each criterion with regard to stage 1.

❊ ❊ ❊

i. The experts concerned by the *writing phase 1* enumerate the table attributes listed in Table 2.

Table 2: Table attributes enumerated in step 7

Table Name	Enumerated attributes
`MaterialLot`	{`IDLot`, `Description`}
`MaterialDefinition`	{`IDMatDef`, `Value`}
`ProductSegment`	{`IDProdSeg`, `Duration`, `UnitDurat.`}
`ProductionOrder`	{`IDProdOrder`, `StartTime`, `EndTime`}

ii. The group of experts in charge of the contextual weight adjustments define four entity groups through the LDM, which are given in Table 3. The *Equipment* and *Personal data groups* report information about equipments and persons which/who are somehow in relation with the product (e.g. equipments used for manufacturing it). The *Material* and *Production data groups* relate respectively information about product composition (e.g. raw materials, component parts) and operations (e.g. production rules, production scheduling). Figure 3(a) depicts 4 of the 19 entities which are included in G_1 and G_3. The experts then carry out pairwise comparisons between groups as in equation 11. It can be noted at this stage (i.e. stage 1 of the PLC) that the experts strongly favor G_1 over G_2 ($s_{12} = 9$). The relative importance of each group is then computed as detailed for $\lambda(G_1)$ in equation 12. All relative group importances are synthesized by the vector Λ_c in equation 13. It can be observed that experts favor information from G_1 over the other groups ($\lambda(G_2) < \lambda(G_4) < \lambda(G_3) < \lambda(G_1)$).

Table 3: Set of entities composing the LDM: grouped according to the type of information

Product data group (G_1)	Personal data group (G_2)	Production data group (G_3)	Equipment data group (G_4)
`MaterialLot`	`Person`	`ProductionOrder`	`Equipment`
`MaterialDefinition`	`PersonClass`	`ProductSegment`	`EquipmentClass`
`MaterialClass`	`ActualPersonSegment`	`ProductDefinition`	`EquipmentSegmentSpecification`
`ManufacturingBill`	`PersonSegmentSpecification`	`SegmentRequirement`	`ActualEquipmentSegment`
`ActualMaterialLot`		`SegmentResponse`	
`MaterialSegmentSpecification`			

$$\begin{array}{c} \\ G_1 \\ G_2 \\ G_3 \\ G_4 \end{array} \begin{array}{c} \begin{array}{cccc} G_1 & G_2 & G_3 & G_4 \end{array} \\ \begin{bmatrix} 1 & 9 & 3 & 7 \\ \frac{1}{9} & 1 & \frac{1}{7} & \frac{1}{5} \\ \frac{1}{3} & 7 & 1 & 7 \\ \frac{1}{7} & 5 & \frac{1}{7} & 1 \end{bmatrix} \end{array} \tag{11}$$

$$\lambda(G_1) = \frac{1+9+3+7}{1+9+3+7+\frac{1}{9}+1+\ldots+\frac{1}{7}+1} = 0.55 \tag{12}$$

$$\Lambda_c = \begin{array}{c} \begin{array}{cccc} \lambda(G_1) & \lambda(G_2) & \lambda(G_3) & \lambda(G_4) \end{array} \\ \begin{bmatrix} 0.55 & 0.04 & 0.31 & 0.10 \end{bmatrix} \end{array} \tag{13}$$

iii. Regarding the coefficient k, experts agree to fix k at 1.08 (*cf.* Figure 5) since the model is not so large (the maximum distance through the LDM is equal to 15).

iv. The experts in charge of the definition of the criteria importance in *writing phase 1* then carry out pairwise comparisons between criteria as in equation 14. The normalized vector Λ_ρ is then computed (see equation 15). The experts highly favor C_e over respectively C_c and C_m ($\lambda(C_m) < \lambda(C_c) < \lambda(C_e)$). This means that the experts prefer to store information they deem relevant instead of following the recommendation made by experts in the contextual criterion.

$$\begin{array}{c} \\ C_e \\ C_c \\ C_m \end{array} \begin{array}{c} \begin{array}{ccc} C_e & C_c & C_m \end{array} \\ \begin{bmatrix} 1 & 5 & 7 \\ \frac{1}{5} & 1 & 5 \\ \frac{1}{7} & \frac{1}{5} & 1 \end{bmatrix} \end{array} \tag{14}$$

$$\Lambda_\rho = \begin{array}{c} \begin{array}{ccc} \lambda(C_e) & \lambda(C_c) & \lambda(C_m) \end{array} \\ \begin{bmatrix} 0.72 & 0.22 & 0.06 \end{bmatrix} \end{array} \tag{15}$$

All specifications required in *writing phase 1* have now been defined. Then, process step 2 and 3 are launched when the communicating textile reaches this *writing phase*.

5.1.1. Step 2: Data relevance

Assume now that a textile reel arrives at *writing phase 1*. First, it is necessary to start the protocol for identifying all product-related information in the database. For this purpose, the set of tables $\mathcal{T}$ that compose the LDM are explored using `RetrievalData` (*cf.* Algorithm 1). The algorithm steps are presented by focusing on three tables $\in \mathcal{T}$: {`MaterialLot`, `MaterialDefinition`, `ManufacturingBill`} which are detailed in Figure 9(a). In the following explanation, these three tables are respectively denoted {`ML`, `MD`, `MB`}. In order to run the algorithm, the RFID reader gets the textile ID (d_p) which refers to the data item $T_{ML}\{2,1\}$ = `LPB61` (i.e. the product lot instance), as highlighted in Figure 9(a). Figure 9(b) details the different steps of `RetrievalData` through a flow chart where the syntax $\#i$ makes reference to the row i in Algorithm 1.

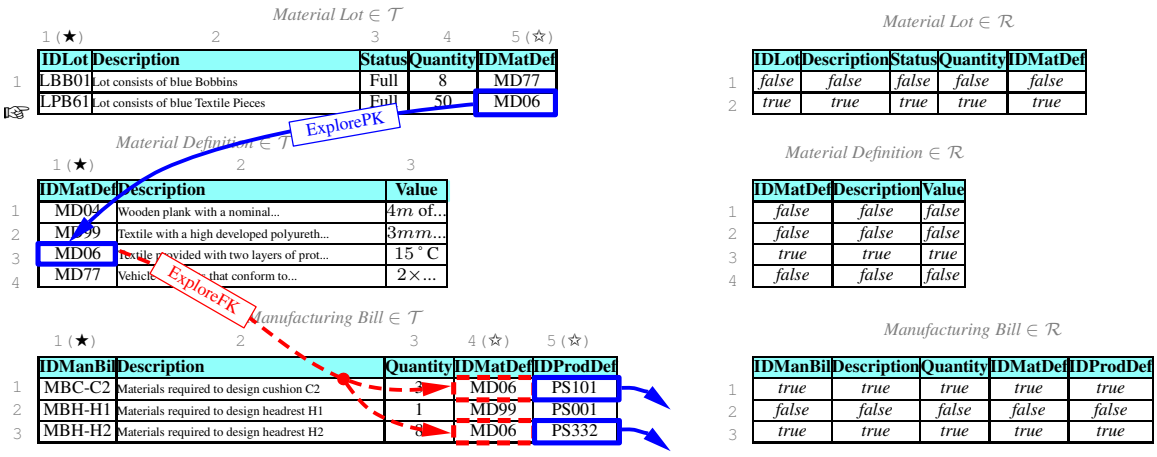
First, $T_{ML}\{2,1\}$ is added to $\mathcal{I}$ (#2) and $\mathcal{E}$ is set to empty (#3). Since $\mathcal{I} \notin \emptyset$ (#4,#5), `RetrievalData` calls up the sub-function `ExplorePK`($\mathcal{T}, \mathcal{R}, T_{ML}\{2,1\}$) (#2)[9]. This sub-function, on the one hand, sets to *true* in $\mathcal{R}$[10] the tuple containing $T_{ML}\{2,1\}$ (i.e. row 2 of `ML` as shown in Figure 9(a)) and, on the other hand, checks whether FKs related to this tuple exist. This is true with `MD06` (i.e. $T_{ML}\{2,5\}$) which is a PK in `MD` as shown in Figure 9(a) (blue/solid arrow from `ML`). $T_{MD}\{3,1\}$ is therefore added to $\mathcal{I}$ (#7) and the exploration continues (#8, #9, #5[11]). The data item $T_{MD}\{3,1\}$ is thus used to explore `MD` via `ExplorePK`, as indicated with `ExplorePK`($\mathcal{T}, \mathcal{R}, T_{MD}\{3,1\}$) (#6) in Figure 9(b). This sub-function, on the one hand, sets to *true* in $\mathcal{R}$ the tuple containing $T_{MD}\{3,1\}$ (i.e. row 3 of `MD`) and, on the other hand, checks whether FKs related to this tuple exist. No FK exists in this table. $\mathcal{I}' = \emptyset$ and $\mathcal{I}$ then becomes empty (#7,#8,#9,#5[12]). The exploration of the LDM can no longer continue. To avoid such a break, the exploration based on FK is called up, `ExploreFK`($\mathcal{E}$, $\mathcal{T}$, $\mathcal{R}$) (#10), which searches other potential relations where PKs from tables already explored (i.e. PKs from tables $\in \mathcal{E}$) are FKs in tables not explored yet (i.e. in tables $\notin \mathcal{E}$). In our example, `MB` is a neighbor of `MD` (the attribute 1 of `MD` is a FK in `MB` - see attribute 4 of `MB` in Figure 9(a)) and has not been explored yet. Accordingly, `ExploreFK` finds out if there are PKs from `MD` which are FKs in `MB`. This is true with `MD06` as highlighted in Figure 9(a) (red/dashed arrow). New product-related data items are identified ($T_{MB}\{1,4\}$ and $T_{MB}\{3,4\}$) and serve as inputs for exploring the rest of the LDM, using

[9] The steps of the sub-functions `ExplorePK` and `ExploreFK` are only presented in sentence form hereafter.

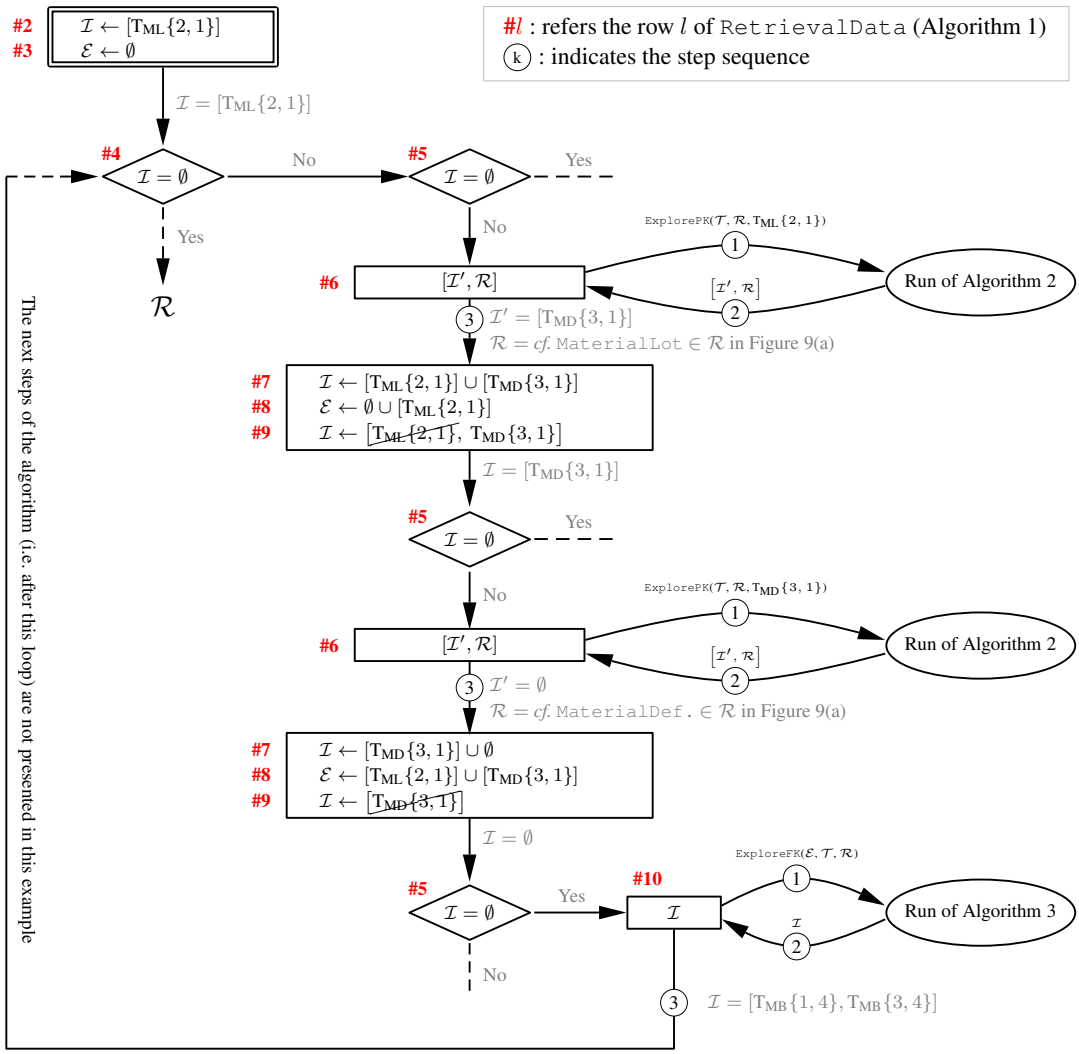
[10] $\mathcal{R}$ shows tuples which are "product-related tuples" by setting them to *true* (see Table 1).

[11] #5 corresponds to the 2nd run in the while-loop #5 of `RetrievalData`.

[12] #5 corresponds to the 3rd un in the while-loop #5 of `RetrievalData`.



(a) Steps of the exploration steps and setting of "product-related tuples" to *true*



(b) Flow chart of `RetrievalData` (Algorithm 1)

Figure 9: Identification of 'product-related tuples" in tables $\in \mathcal{T}$ via the function `RetrievalData`

again `ExplorePK` (*cf.* blue/solid arrow from `MB`). This new algorithm-loop is not detailed here but Figure 9(a) reveals the tuples set to *true* in $\mathcal{R}$, namely tuples 1, 3 from `MB`.

Secondly, it is necessary to compute the relevance of all retrieved product-related data items. Figure 10 details the calculation regarding `MaterialDefinition`. Non-product-related data items (i.e. the ones set to *false* in $\mathcal{R}$) are set to 0 (*cf.* Figure 10). Regarding C_e, attributes `IDMatDef` and `Value` are set to 1 because they were enumerated by users in opposition to `Description` (*cf.* Table 2). Regarding C_c, all product-related data items are set to 0.55 because `MaterialDefinition` is included in G_1, whose relative importance is equal to 0.55 (*cf.* equation 13). Regarding C_m, priorities are equal to k^{-1} because the distance between `MaterialDefinition` and `MaterialLot`[13] is equal to 1 (*cf.* Figure 3(a)). The relevance is then computed based on equation 1 for all data items. Figure 10 details the calculation of $R(T_{MD}\{3,1\})$ with $k = 1.08$, ρ_e, ρ_c and ρ_m respectively equal to 0.72, 0.22 and 0.06 (*cf.* equation 15) and size(`MD060111223`) = 11 (11 ASCII characters = 11 bytes). The relevance of $T_{MD}\{3,1\}$ is thus equal to 0.37.

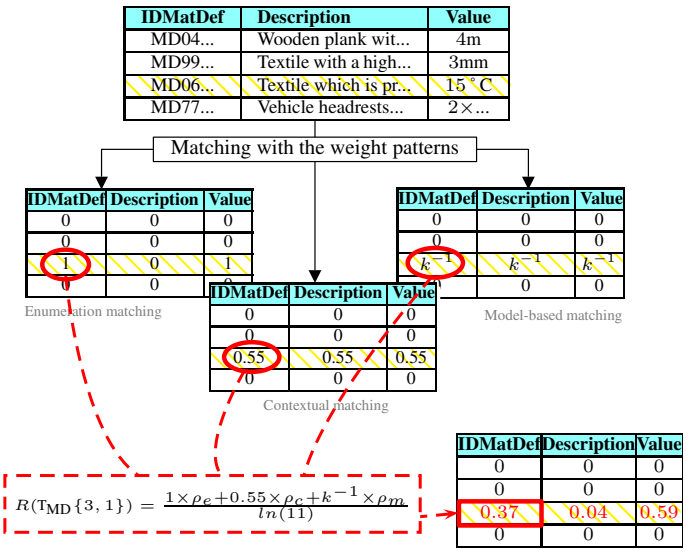


Figure 10: Relevance computation of "product-related data items"

Now, results about the relevance of all data items from the database are commented. The term "list" is used in the following explanations to refer to all product-related data items ($l = \{1,\ldots,n\}$). Figure 11(a) provides the resulting list ordered from the most relevant $R(l)$ to the lowest. The most relevant data item with $R(l) = 0.7066$ is located in `ProductSegment`, in the attribute named `Duration`, in the tuple whose the PK is `PSTB0B`. Due to the large amount of data items included in the list (524 exactly), results are presented in the form of diagrams (pie chart and whisker diagram). First, let us look at the whisker diagram in Figure 11(b). For each table $t \in \mathcal{T}$ is given the min, the 1st and 3rd quartile, the median and the max relevance of the set of data items belonging to t. It can be observed that the most relevant data items come from `ProductSegment` (which includes the two first data items of the list), but also from `MaterialDefinition`, `MaterialLot` and `ProductionOrder`. This is due to the fact that attributes from these four tables have been enumerated by users (*cf.* Table 2) and because C_e is the most important criterion at this stage of the PLC (*cf.* equation 15). Moreover, one can note that these four tables are included in G_1 and G_3 and that the experts, concerned by C_c, have highly recommended to select information from both groups (*cf.* equation 13).

At this instant, product-related data items are stored on the product from the most relevant towards the least relevant, until no more memory is available. In our case, no more than the first 159 data items can be stored on the communicating textile as highlighted in Figure 11(a) (representing $\simeq$ 70Mbytes when the entire list represents $\simeq$ 450Mbytes). The pie chart in Figure 11(c) shows the percentage of data items among the 159 that come from each entity group. For instance, 48% (i.e. $\simeq$ 76 data items) are included in tables of G_3 and 37% in tables of G_1. This is largely due to the choices made in the enumeration and contextual criteria as explained previously. In this respect, these results meet the expert specifications.

The list, the whisker diagram and the pie chart are always displayed to the user and are used as decision-support tools.

5.1.2. Step 3: Data storage

Step 3 of the information dissemination framework is then used to store/split the first 159 data items on the communicating textile, as exemplified in section 4. Then, the textile continues its PLC. The next section takes focus on how data is retrieved and handled by the actor concerned by the *cutting operation* (*cf.* Figure 7).

5.2. Cutting operation: Stage 2

The textile then arrives at the *cutting operation* in stage 2. Process step 3 is implemented on the production line to enable the user/machine to retrieve and rebuild the information. All data items are therefore displayed on a mobile device thanks to a specific software[14]. Figure 12(a) shows the unique tuple from `MaterialDefinition` which can be re-built. In some cases, it is impossible to rebuild the entire tuple as shown in Figure 12(b), where two data items are missing in the second tuple. This is because both data items got a ranking $> 159^{th}$ and, consequently, have not been stored on the textile.

[13] `MaterialLot` is the product's table since d_p refers to that one.

[14] The software is developed in JAVA® and enables, among other, to perform queries without requiring access to the database.

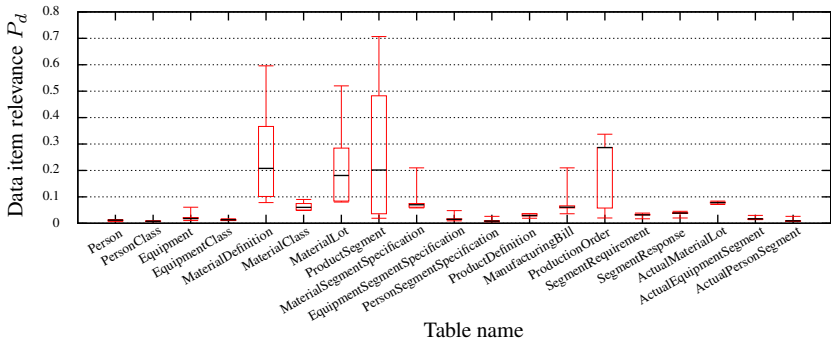
Rang	Table Name	Attribute Name	Primary Key Value	Cell Value	$R(l)$
1st	ProductSegment	Duration	PSTB0B	0.5	0.7066
2st	ProductSegment	Duration of Unit	PSTB0B	Hours	0.6102
3th	MaterialDefinition	IDMatDef	MD0601...	15°C	0.5960
4th	Lot	IDMaterialLot	LPB61	LPB61	0.3915
...	...	...	...	...	...
159th	SegmentResponse	ActualStartTime	PSR0001	10:15:00	0.044
⋮	⋮	⋮	⋮	⋮	⋮
523th	Equipment	Description	ET3A01	Line 3A, Qualific...	0.0051
524th	PersonClass	Description	PCPLW01	Production Line...	0.0051

Set of data items that must be stored on the product

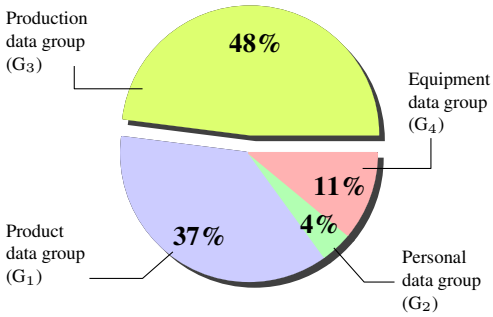
storage limit on the material

data items which cannot be stored on the product

(a) List of data items ordered from the most relevant $R(l)$ to the lowest

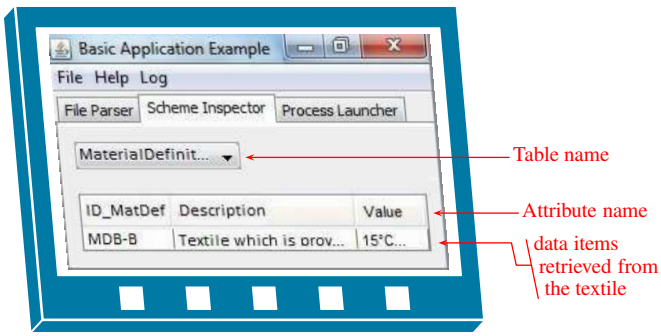


(b) Result of data items' relevance in a whisker diagram: Stage 1

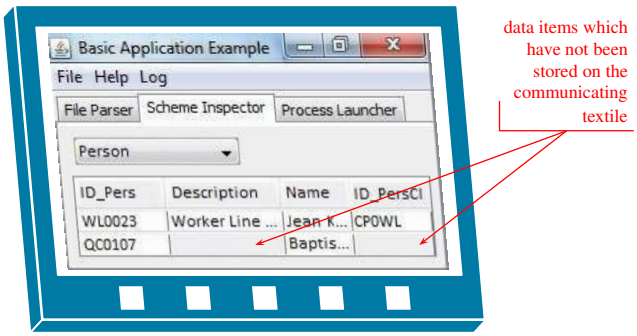


(c) Result in form of a pie chart: Stage 1

Figure 11: Results of the data item's relevance



(a) Display of tuples related to `MaterialDefinition`



(b) Display of tuples related to `Person`

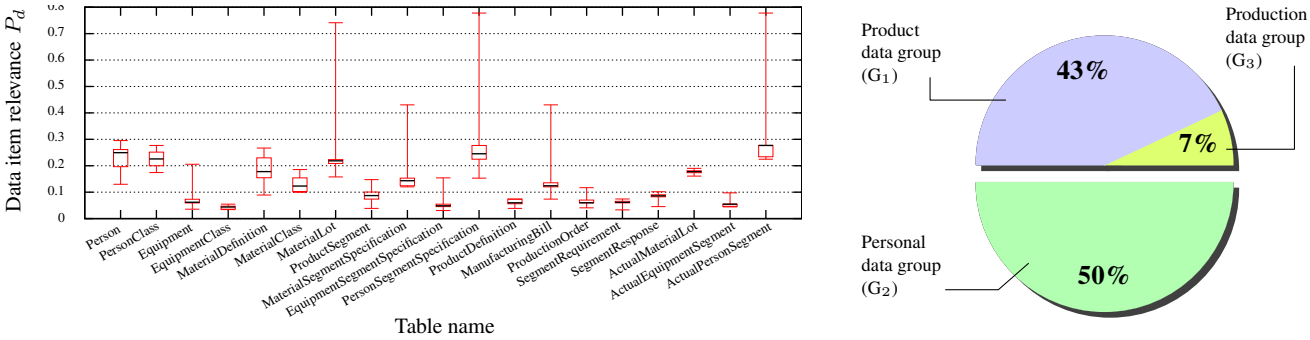
Figure 12: JAVA software developed to display the product-related information retrieved from a "communicating material"

5.3. *Writing phase: Stage 2*

When products arrive at the *writing phase 2*, the textile reels have been cut in textile pieces and then, have been pasted on wood planks to design upholstered chair seats (see Figure 7). In *writing phase 2*, the experts define the same criteria importances as in *writing phase 1* (*cf.* equation 15). Regarding C_e, the experts do not enumerate any attributes, preferring to follow the recommendations from experts in C_c. In C_c, the relative importances of the four entity groups with regard to stage 2 of the PLC are given in equation 16. This time round, G_2 and G_1 are respectively the most important entity groups. Finally, experts fix k at 1.08 in C_m.

$$\Lambda_c = \begin{array}{c} \begin{array}{cccc} \lambda_c(G_1) & \lambda_c(G_2) & \lambda_c(G_3) & \lambda_c(G_4) \end{array} \\ \left[\begin{array}{cccc} 0.24 & 0.54 & 0.14 & 0.08 \end{array} \right] \end{array} \quad (16)$$

Process step 2 is run when upholstered chair seats reach the *writing phase 2* and, as previously, product-related information is first identified and then assessed in term of relevance. The whisker diagram in Figure 13(a) shows that the spreads between the relevance values are not so significant compared to the *writing phase 1*. Indeed, $R(l)$ values in Figure 13(a) vary from 0 to 0.14, while values in Figure 11(b) vary from 0 to 0.5. This is due to the fact that no data is enumerated in *writing phase 2*, combined to the fact that experts strongly favored C_e over C_c and C_m (*cf.* equation 15). In Figure 13(a), it can be observed that tables included in G_2 have the highest relevance scores (*cf.* the median values of `PersonSegmentSpecification`, `ActualPersonSegment`, `Person`, `PersonClass`) because experts highly recommend to select information from this entity group ($\lambda_c(G_2) = 0.54$). It can also be noted that



(a) Result of data items' relevance in a whisker diagram: Stage 2

(b) Result in form of a pie chart: Stage 2

Figure 13: Results of the data item's relevance

`MaterialLot` is quite relevant, which can be explained by the fact that d_p is located in this table and, as a consequence, the model-based criterion favors data items from this table ($k^{-0} = 1$).

This time, it is only possible to store 60 data items on the upholstered chair seats. The pie chart in Figure 13(b) shows that almost all data items among the 60 come from G_2 and G_1 with respectively 43% and 50%, and no data items from G_3 are selected.

6. Discussion

Today, companies recognize that future profits will not come from the manufacture of products in developed countries. Companies in countries where costs are 10% or 20% of those in Europe will be able to carry out manufacturing activities at a much lower cost [39]. However, production is only one phase of the PLC and there are other areas where companies can add values. They can, for instance, develop ideas for new environment-friendly products (MOL), provide customized and advanced products and improve the customer experience (MOL) [19]. In the PROMISE EU project, it was found that many stakeholders in the product supply and value chain (from designers to recyclers) desire to enable seamless information flow, tracing and updating of information about the product or process, even after its delivery to the customer. Accordingly, this consortium created the concept of *closed-loop PLM* (closed-loop Product Lifecycle Management) [20, 15][15], which is an extension of traditional PLM systems, making all the product-related data visible from BoL to MoL to EoL using feedback information between the different PLC phases. Although the "communicating material" paradigm is in its incipient stage, this paradigm could bring the concept of *closed-loop PLM* a step further. Indeed, information could be gathered on all or parts of the material that the product is made of, and that on a lifelong basis. As mentioned, such a product could thus have new abilities compared to conventional products like the copy/redundancy/backup of information on specific parts of the material.

Although "communicating materials" provide new abilities compared to conventional products, they still have low memory capacities compared to product databases that become larger and larger, but it seems to us that technology changes will continue to accelerate and to open up very important challenges. Accordingly, this paper (which is one of the first research work on this paradigm) proposes an information dissemination framework to select context-sensitive information from the database, and also provides the tools necessary to split it on all or parts of the material. Context-sensitive information is selected thanks to a degree of relevance, which takes into account the context of use of the product (actor's expectations, application features). A case study shows how this framework can be implemented in the context of a textile fabric using communicating textiles. The results show that the selected information, i.e. the information stored on the communicating textiles, largely meets the expectations formulated by the actors of the textile lifecycle.

7. Conclusion

Concepts such as Internet of Things, Ubiquitous Computing redefine how we interact with product information. It is not uncommon to use intelligent products to ensure an information continuum over the product life cycle - PLC (e.g. for traceability purposes). Linking the product-related information to the products themselves is a formidable challenge, thus making the information easily accessible. Over the last decades, several solutions have been designed to enable such a linkage. However, information is often deported through the network (stored in databases) and is accessed remotely. In this paper, a new kind of intelligent product referred to as "communicating material" is introduced, which provides the opportunity to embed data on all or parts of the material that the it is made of. In our context, such products are used to convey information between the different actors of the PLC,

[15] Closed-loop PLM has recently been renamed CL_2M (Closed Loop Lifecycle Management: http://www.cl2m.com

thus improving data interoperability, availability and sustainability. Indeed, the same data can be copied to several parts over the material that is useful, for instance, in cases where it is relevant that little or no product-related data is lost.

One important part of the data selection process is the difficulty met by experts to fix the different values (e.g. in the comparison matrices, the k coefficient,...) because our approach does not basically allow to express vagueness or uncertainty. In addition, the aggregation of different experts' point of view is not supported. Accordingly, the fuzzy set theory could be further considered to improve our methodology. In further work, new challenges should be addressed regarding the communicating material paradigm, namely:

- *the design of new kinds of communicating material*: new substances could be considered to design communicating material like wood, cement or still paint. For information purposes, a new form of "communicating wood" is currently studied in [14], where Nuclear Quadrupole Resonance is used for mass marking, thus enabling new identification code strategies,
- *the development of strategies for data diffusion*: specific methods should be further developed to specify whether an information should be stored or replicated on specific part(s) of the material. Pushing the communicating material paradigm at its extreme, it could be imagined that new strategies of data mutation into the material could be designed (e.g. information could mutate when adverse events occurs for security or privacy purposes),
- *the development of strategies for online diagnosis:* in further research, μdevices able to gather and process data directly on the material (e.g. μsensors) could be used/disseminated in that one, as is done in our paper with RFID μtags. Online diagnosis strategies could therefore be developed to detect or anticipate defects in the material (e.g. cracks).

References

[1] Atzori, L., Lera, A., & Morabito, G. (2010). The internet of things: A survey. *Computer Networks*, *54*(15), 2787–2805.

[2] Bai, Y. B., Zhang, K., & Wu, H. R. (2012). RFID-based Ubiquitous Positioning Techniques – the Past, Present and Future. *National Committee for Space Science & National Space Society of Australia, Melbourne, Australia.*

[3] Baltrunas, L., Ludwig, B., Peer, S., & Ricci, F. (2012). Context relevance assessment and exploitation in mobile recommender systems. *Personal and Ubiquitous Computing*, *16*(5), 507–526.

[4] Chan, D., & Roddick, J. F. (2003). Context-sensitive mobile database summarisation. In *Proceedings of the 26th Australasian computer science conference, Adelaide, Australia* (Vol. 16, pp. 139–149).

[5] Cherenack, K., Zysset, C., Kinkeldei, T., Münzenrieder, N., & Tröster, G. (2010). Context relevance assessment and exploitation in mobile recommender systems. *Advanced Materials*, *22*(45), 5178–5182.

[6] Cook, D. J., Augusto, J. C., & Jakkula, V. R. (2009). Ambient intelligence: Technologies, applications, and opportunities. *Pervasive and Mobile Computing*, *5*(4), 277–298.

[7] Främling, K., Ala-Risku, T., Kärkkäinen, M., & Holmström, J. (2007). Design patterns for managing product life cycle information. *Communications of the ACM*, *50*(6), 75–79.

[8] 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*, *20*(7), 715–726.

[9] Främling, K., Holmström, J., Ala-Risku, T., & Kärkkäinen, M. (2003). Product agents for handling information about physical objects. *Report of Laboratory of information processing science series B, TKO-B*, *153*(3).

[10] Främling, K., Holmström, J., Loukkola, J., Nyman, J., & Kaustell, A. (2013). Sustainable PLM through Intelligent Products. *Engineering Applications of Artificial Intelligence*, *26*(2), 789–799.

[11] Garcìa, A., Seok Chang, Y., & Valverde, R. (2006). Impact of new identification and tracking technologies on a distribution center. *Computers & Industrial Engineering*, *51*(3), 542–552.

[12] Garcia-Molina, H., Ullman, J. D., & Widom, J. (2008). *Database systems: the complete book.* Prentice Hall Press.

[13] ISO/IEC 18000-1 (2004). *Information technology AIDC techniques – RFID for item management – air interface, part 1 – generic parameters for air interface communication.*

[14] Jover, J., Thomas, A., Leban, J., & Canet, D. (2010). Pertinence of new communicating material paradigm: A first step towards wood mass marking. *Proceedings of the New achievements in materials and environmental sciences, Nancy, France.*

[15] Jun, H.B., Kiritsis, D., & Xirouchakis, P. (2007). Research issues on closed-loop PLM. *Computers in industry*, *58*(8), 855–868.

[16] Jun, H.B., & Suh, H.-W. (2009). Decision on the memory size of embedded information systems in an ubiquitous maintenance environment. *Computers & Industrial Engineering*, *56*(1), 444–451.

[17] Kärkkäinen, M. (2003). Increasing efficiency in the supply chain for short shelf life goods using RFID tagging. *International Journal of Retail & Distribution Management*, *31*(10), 529–536.

[18] Khosroanjom, D., Ahmadzade, M., Niknafs, A., & Mavi, R. K. (2011). Using fuzzy AHP for evaluating the dimensions of data quality. *International Journal of Business Information Systems*, *8*(3), 269–285.

[19] Kiritsis, D. (2011). Closed-loop PLM for intelligent products in the era of the internet of things. *Computer-Aided Design*, *43*(5), 479–501.

[20] Kiritsis, D., Bufardi, A., & Xirouchakis, P. (2003). Research issues on product lifecycle management and information tracking using smart embedded systems. *Advanced Engineering Informatics*, *17*(3), 189–202.

[21] Koronios, A., Nastasie, D., Chanana, V., & Haider, A. (2007). Integration Through Standards - An overview of international standards for engineering asset management. In *Proceedings of the 2nd World Congress on Engineering Asset Management and the 4th International Conference on Condition Monitoring, Harrogate, United Kingdom.*

[22] Kubicki, S., Lepreux, S., & Kolski, C. (2012). RFID-driven situation awareness on TangiSense, a table interacting with tangible objects. *Personal and Ubiquitous Computing*, *16*(8), 1079-1094.

[23] Kubler, S., Derigent, W., Thomas, A., & Rondeau, É. (2010). Problem definition methodology for the "communicating material" paradigm. In *Proceedings of the 10th IFAC Workshop on Intelligent Manufacturing Systems, Lisbon, Portugal.*

[24] Kubler, S., Derigent, W., Thomas, A., & Rondeau, É. (2011). Prototyping of a communicating textile. In *Proceedings of the International Conference on Industrial Engineering and Systems Management, Metz, France* (pp. 1333–1342).

[25] Lin, L. C. (2009). An integrated framework for the development

of radio frequency identification technology in the logistics and supply chain management. *Computers & Industrial Engineering*, *57*(3), 832–842.

[26] McFarlane, D., Giannikas, V., Wong, C. Y., & Harrison, M. (2012). Intelligent Products in the Supply Chain – 10 Years On. In *Proceedings of the 14th Symposium of Information Control Problems in Manufacturing, Bucharest, Romania*.

[27] McFarlane, D., Giannikas, V., Wong C. Y., & Harrison, M. (2013). Product intelligence in industrial control: Theory and practice. *Annual Reviews in Control*, http://dx.doi.org/10.1016/j.arcontrol.2013.03.003

[28] Melton, J., & Simon, A. (2001). *SQL: 1999-Understanding Relational Language Components*, Morgan Kaufmann.

[29] Meyer, G. G., Främling, K., & Holmström, J. (2009). Intelligent products: A survey. *Computers in Industry*, *60*(3), 137–148.

[30] Mitrokotsa, A., Sheng, Q. Z., & Maamar, Z. (2012). User-driven RFID applications and challenges. *Personal and Ubiquitous Computing*, *16*(3), 223–224.

[31] Ono, C., Takishima, Y., Motomura, Y., & Asoh, H. (2009). Context-aware preference model based on a study of difference between real and supposed situation data. In *Proceedings of the 17th International Conference on User Modeling, Adaptation, and Personalization, Trento, Italy* (pp. 102–113).

[32] Özsu, M. T., & Valduriez, P. (1991). Distributed database systems: where are we now?. *Computer*, *24*(8), 68–78.

[33] Panetto, H., Dassisti, M., & Tursi, A. (2012). ONTO-PDM: Product-driven ONTOlogy for Product Data Management interoperability within manufacturing process environment. *Advanced Engineering Informatics*, *26*(2), 334–348.

[34] Park, S., Mackenzie, K., & Jayaraman, S. (2002). The wearable motherboard: a framework for personalized mobile information processing (PMIP). *39th Proceedings of Design Automation Conference, New Orleans, USA* (pp. 170–174).

[35] Ropponen, A., Linnavuo, M., & Sepponen, R. (2013). A novel concept of a wearable information appliance using context-based human-computer interaction. *Personal and Ubiquitous Computing*, *17*(1), 159–167.

[36] Roussos, G., & Kostakos, V. (2009). RFID in pervasive computing: State-of-the-art and outlook. *Pervasive and Mobile Computing*, *5*(1), 110–131.

[37] Saaty, T. L. (1980). *The Analytic Hierarchy Process*, New York: McGraw-Hill.

[38] Sánchez López, T., Ranasinghe, D. C., Harrison, M., & McFarlane, D. (2012). Adding sense to the Internet of Things. *Personal and Ubiquitous Computing*, *16*(3), 291–308.

[39] Stark, J. (2011). *Product lifecycle management: 21st century paradigm for product realisation*, Springer-Verlag GmbH.

[40] Sundmaeker, H, Guilemin, P., Friess, P., & Woelffle, S. (2010). Vision and challenges for realizing the Internet of Things. *Cluster of European Research Projects on the Internet of Things, Brussels, Belgium*.

[41] Tajima, M. (2007). Strategic value of RFID in supply chain management. *Journal of purchasing and supply management*, *13*(4), 261–273.

[42] Wang, L.-C., & Lin, S.-K. (2009). A multi-agent based agile manufacturing planning and control system. *Computers & Industrial Engineering*, *57*(2), 620–640.

[43] Weiser, M. (1993). Hot topics-ubiquitous computing. *Computer*, *26*(10), 71–72.

[44] Wong, C. Y., McFarlane, D., Ahmad Zaharudin, A., & Agarwal, V. (2002). The Intelligent Product Driven Supply Chain. *Proceedings of the International Conference on Systems, Man and Cybernetics, Hammamet, Tunisia*.

Appendix A. Splitting Protocol

The *splitting protocol* is defined at layer 7 of the OSI model as illustrated by the datagram in Figure A.14 (gray background). The application data consists of 7 fields, 6 are reserved to the header (used to rebuild the data item) and the last one contains the data item value:

1. **Protocol (8 bits):** Integer from 0 to 255 which enables to know which fields compose the packet. The value 255 is defined in our application, referring to the frame structure given in Figure A.14,

2. **Size (8 bits):** Integer from 0 to 255 indicating the size of data included in the 7th field "Data",

3. **Seq_Num (8 bits):** Integer from 0 to 255 providing the sequence number of the current frame (1 frame/ tag). The sequence number is used to know in what order the datagrams have been split among the RFID tags (needed to rebuild the set of data items),

4. **Last_Num (8 bits):** Integer from 0 to 255 providing the sequence number of the last frame which contains data related to the same "writing phase". The notion "writing phase" indicates when data has been written on the product. This field will allow programing new services to deal with (i) redundant information (a same information will be differentiated thanks to the writing phase) (ii) data inconsistency (it is possible to identify outdated information) (iii) traceability purposes (to maintain the data history),

5. **ID_Phase (64 bits):** Integer from 0 to 2^{64} which is the identifier of the writing phase (the date of writing is currently used). Several frames may have the same "ID_Phase" but a couple "ID_Phase/Seq_Num" is unique,

6. **Checksum (32 bits):** Integer from 0 to 2^{32}. Used for data error-checking.

7. **Data ($n-128$ bits):** The content of the data item is added in this field, which is a string. This string may or may not be stored integrally in a unique RFID tag according to the technology (where n is the number of writable data bits in one RFID tag). Let us note that an index is added for locating each data item in the database (i.e. the table name, the attribute name and the instance concerned). In our method, the index is coded as follows: `Tablename.Attributename.PrimaryKeyValue`.

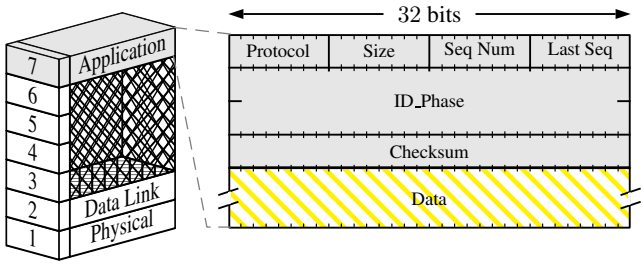


Figure A.14: Header definition for the *Splitting protocol*