Noname manuscript No.
(will be inserted by the editor)

Embedding data on "communicating materials" from context-sensitive information analysis

Sylvain Kubler · William Derigent · André Thomas · Éric Rondeau

Received: date / Accepted: date

Abstract Nowadays, intelligent products carrying information or having decision-making abilities are becoming widespread. The idea of using intelligent products to ensure an information continuum all along the product life cycle (PLC) is more and more shared today. However, it is not that easy to identify the information that must be linked to the product. As a result, this paper proposes an information dissemination process for selecting information sensitive to the context of use of the product. This information is then stored on the products themselves using a new type of augmented material, referred to as "communicating material".

Keywords Intelligent product · Product life cycle management · Data dissemination · Internet of Things · Radio-frequency identification

1 Introduction

New challenges and opportunities arise with concepts such as the Internet of Things (IoT) and ubiquitous/pervasive computing [17]. Through these concepts, real-world objects are linked with the virtual world, thereby enabling connectivity anywhere, anytime, for anything and anyone. The IoT based on RFID (Radio Frequency IDentification) usage is also a substantial topic, dealing with access to information disseminated via any kind of physical object and the development of new smart services and applications [19].

It is not uncommon today to use intelligent products for ensuring an information continuum all along the PLC. Indeed, a product moves through numerous companies and technical, semantic and organizational interoperability

S. Kubler · W. Derigent · A. Thomas · É. Rondeau
Centre de Recherche en Automatique de Nancy (CRAN),
Campus Sciences, B.P. 70239, 54506 Vandœuvre-ls-Nancy
Tel.: +33 383 684 425
E-mail: s.kubler@univ-lorraine.fr

is not always ensured, thereby contributing to information loss. Considering the product as an information vector (to which information can be linked) should contribute to improved the transfer of information among actors of the PLC. Over the last decade, those involved with intelligent-manufacturing systems have demonstrated that systems that integrate intelligent products can be more efficient, flexible and adaptable [3,13,18,2]. Främling et al. [6] argue that it is a formidable challenge to link product-related information to the products themselves (i.e., making the information easily accessible). However, most of the time, products only provide a network pointer to a linked database (via a RFID tag) and a decision making software agent [14]. It should be noted that these products have limitations in some respects: risk of tag damage, small memory capacity, etc.

After many years of considering a communicating product as a physical product associated with an informational product (realized via Auto-ID technologies such as RFID), a new paradigm has been proposed by Thomas [16] which changes drastically the way to consider the material. This concept aims to give the ability for the material to be intrinsically and wholly communicating, as discussed in [8,9,10]. The futuristic idea behind this concept is to imagine that the material is by its very nature, communicating; what ever the technical solution is. Although current technologies do not allow building an "intrinsically communicating material", the designing of new solutions providing new services to users may be addressed. For instance, a product made of "communicating material" could have special abilities like data storage, copy/redundancy/backup information solutions. Accordingly, some open questions remain to be addressed, these include for example a specification for the information to be gathered, stored and distributed over the PLC and the linkage of new hardware and software systems with current systems [12, 20]. To address some of these matters, this paper develops an information dissemination process which consists of two main steps :

1. selecting relevant information from the external database that should be stored/replicated on the product, at a given moment of the PLC,
2. storing information on the product and, subsequently, retrieving it [9,10].

This paper mainly focuses on the first process step which is the subject of section 2. A brief summary of previous works carried out on the second process step is nevertheless provided in section 3. Finally, the data dissemination process is applied in a case study in section 4.

2 Process step 1: relevant data identification

During its life cycle, a product is accessed by many actors and undergoes numerous operations. It is therefore essential to identify context-sensitive information. Accordingly, an identification method that uses the logical data model (LDM) is developed.

Fig. 1 gives insight into part of such an LDM. A given LDM entity corresponds, once implemented, to the relational table shown as `MaterialDefini-`

`tion` in Fig. 1, where the attributes listed for each entity correspond to the table columns and each table row is referred to as a tuple/instance of the relation. The identification method has two stages:

A. Identification of all product-related information throughout all tables: it requires searching all tuples related somehow to the network pointer carried by the communicating product (i.e., the identifier, denoted d_p in our study). The identification is achieved via an algorithm detailed in section 2.1. Fig. 1 gives an example in which tuple 3 of `MaterialDefinition` (hatched background) is a product-related tuple that has been retrieved via the algorithm.
B. Assessment of the information relevance[1]: it indicates if data must be stored on the product according to user concerns, environmental details, etc. To achieve this, the quantitative model developed by Chan and Roddick [4] is implemented and adapted. Their approach is interesting in the sense that they try to match the context with data, to identify context-sensitive information. All tables are split in data items[2] and a relevance value is computed for each of them.

Fig. 1 gives an example of three data items (belonging to the product-related tuple 3 retrieved in Stage A), for which the relevance values have been calculated and displayed. The higher the relevance value, the higher is the probability that this data item will be stored on the product. The approach developed by Chan and Roddick is detailed in section 2.2.

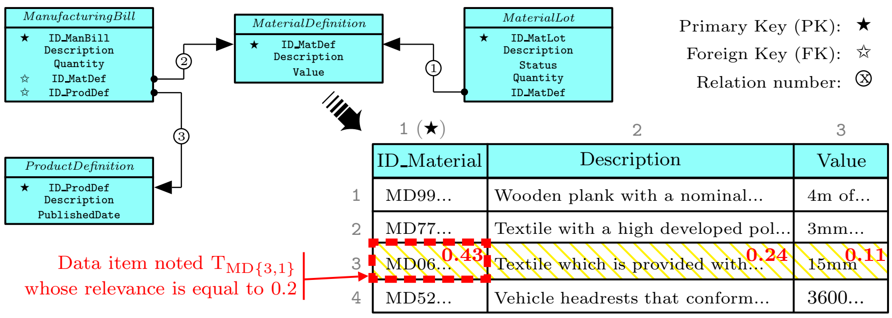


Fig. 1 View of an LDM and a relational table

2.1 Protocol for identifying product-related information

In this section, the goal is to identify all tuples from all tables that are somehow related to the product instance d_p. Concretely, it is necessary to define the first query to retrieve d_p and then, to use the results of that query to build the next

[1] Relevance is only computed for the set of product-related tuples retrieved in stage A.
[2] One data item corresponds to a table cell.

queries dynamically to explore neighbouring tables, which themselves give rise to new queries and so on (until the entire LDM is explored).

One possibility consists in using database-automation programming languages such as Java DataBase Connectivity. Based on this approach, the set of tables $\mathcal{T}$ are extracted in a matrix format at appropriate times (i.e., when users want to store information on the product) and then, are explored via the `RetrievalData` algorithm given in Algorithm 1. This algorithm relies on two subfunctions, namely `ExplorePK` and `ExploreFK`. All variables used in these algorithms are listed in Table 1.

Table 1 Variable definition used in Algorithms 1, 2 et 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 an attribute 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 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 l^{th} element of I)

`RetrievalData` retrieves all tuples which are somehow related to d_p by moving between the tables included in $\mathcal{T}$. To achieve this, it is necessary to identify the relations between tables based on the primary keys (PK) and foreign keys (FK)[3]. This functionality is provided by the subfunction `ExplorePK` (see Algorithm 2) which identifies the FKs in the table currently being explored, and moves to the corresponding tables (i.e., where the FKs are PKs) to identify new product-related tuples. Note that `ExplorePK` may not be sufficient to explore all tables because some relations may not be reached. Indeed, when a table has no further FK, it is necessary to continue in reverse. This means searching relations where the PKs from tables already explored are FKs in tables yet to be explored. This is achieved by a second subfunction `ExploreFK`, given in Algorithm 3. The algorithm steps are detailed in the case study in section 4, which should make it easier to understand.

[3] An FK will be the PK in another table and vice versa.

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

output: $\mathcal{R}$

begin
- $\mathcal{I} \longleftarrow d_p$; // Initialization of $\mathcal{I}$ with the data item referenced by the product
- $\mathcal{E} \longleftarrow \emptyset$; // The list of tables already explored is set to empty
- **while** $\mathcal{I} \neq \emptyset$ **do** // Table exploration is carried on as long as $\mathcal{I} \neq \emptyset$
  - **while** $\mathcal{I} \neq \emptyset$ **do** // Table exploration is carried on as long as $\mathcal{I} \neq \emptyset$
    - $[\mathcal{I}'; \mathcal{R}] \longleftarrow$ `ExplorePK`($\mathcal{T}$, $\mathcal{R}$, $\mathcal{I}_1$); // Data items concerned by the product are set to *true* in $\mathcal{R}$ and data items not explored yet are identified
    - $\mathcal{I} \longleftarrow \mathcal{I} \cup \mathcal{I}'$; // data items not explored yet are added to $\mathcal{I}$
    - $\mathcal{E} \longleftarrow \mathcal{E} \cup \mathcal{I}_1$; // We memorize that the table including $\mathcal{I}_1$ was explored
    - $\mathcal{I} \longleftarrow \mathcal{I} - \mathcal{I}_1$; // We remove the data item used for the exploration: $\mathcal{I}_1$
  - $\mathcal{I} \longleftarrow$ `ExploreFK`($\mathcal{E}$,$\mathcal{T}$) // New relations are searched in not explored tables

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

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

begin
- $\mathcal{I}' \longleftarrow \emptyset$ **forall the** $j_1 \in \{1, n_t\}$ **do**
  - $\mathcal{R}_t\{i, j_1\} \longleftarrow$ *true* **if** $\mathcal{K}_t\{j_1\} = 2$ **then**
    - **forall the** $t' \in \mathcal{T}, j_2 \in \{1, n_{t'}\}|$`value`$(\mathrm{T}_t\{j_1\})$=`value`$(\mathrm{T}_{t'}\{j_2\})$ & $\mathcal{K}_{t'}\{j_2\} = 1$ **do**
      - **forall the** $i_2 \in \{1, m_{t'}\}|$`cont`$(\mathrm{T}_t\{i, j_1\})$=`cont`$(\mathrm{T}_{t'}\{i_2, j_2\})$ **do**
        - $\mathcal{I}' \longleftarrow \mathcal{I}' \cup \mathrm{T}_t\{i_2, j_2\}$

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

output: $\mathcal{I}$

begin
- $\mathcal{I} \longleftarrow \emptyset$ $\mathcal{N} \longleftarrow \emptyset$ **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**
  - $\mathcal{N} \longleftarrow \mathcal{N} \cup \mathrm{T}_{t'}$
- **forall the** $\mathrm{T}_t \in \mathcal{N}, j_1 \in \{1, n_t\}|\mathcal{K}_t\{j_1\} = 2$ **do**
  - **forall the** $\mathrm{T}_{t'} \in \mathcal{E}, j_2 \in \{1, n_{t'}\}|$`value`$(\mathrm{T}_t\{j_1\})$=`value`$(\mathrm{T}_{t'}\{j_2\})$, $\mathcal{K}_{t'}\{j_2\} = 1$ **do**
    - **forall the** $i_1 \in \{1, n_{t'}\}|\mathcal{R}_{t'}\{i_1, j_2\} = true$ **do**
      - **forall the** $i_2 \in \{1, m_t\}|$`cont`$(\mathrm{T}_t\{i_2, j_1\})$ = `cont`$(\mathrm{T}_{t'}\{i_1, j_2\})$ **do**
        - $\mathcal{I} \longleftarrow \mathcal{I} \cup \mathrm{T}_t\{i_2, j_1\}$

2.2 Data selection method

In literature, the vast majority of approaches do not take into account the context of use of the database (or the product) to disseminate data as argued by Chan and Roddick [4]. Indeed, their attention is mainly focused on query frequencies to partition and allocate data in database systems [1]. Accordingly, Chan and Roddick developed an approach which uses the notion of priorities to select context-sensitive information, with these priorities being computed at the level of the data item [4]. The priorities are numerical values either supplied or generated via observation and experimentation. They are assigned using a multifaceted evaluation of eight criteria. In our approach, only the most appropriate criteria with regard to our context (i.e., PLC) are implemented, namely three of the eight criteria. For each criterion, the calculation of a relative priority ρ_x (where x represents a criterion) and an assigned priority $\phi_x(l)$ is performed (where l represents a data item). The first (relative) priority indicates the importance of criterion x with respect to the other criteria. The second (assigned) priority corresponds to the priority (between 0 and 1) computed for a data item l with respect to criterion x. Both priorities are combined in a unique formula (equation 1) that provides the relevance value P_l for the data item l, where len_l is its length (e.g., in bytes). Fig. 1 gives the relevance value P_l for each product related data item, as exemplified with $\mathrm{T}_{\mathrm{MD}}\{3,1\}$ which has a relevance of 0.43. Data items can therefore be classified in order of relevance (i.e., according to P_l).

$$P_l = \frac{\sum^x \rho_x . \phi_x(l)}{ln\,(len_l + 1)} \tag{1}$$

The three criteria used in our approach are as follows:

1. *Enumeration* (C_e): users enumerate data that they consider useful (i.e., data that they recommend be attached to the product),
2. *Contextual* (C_c): this can be used to include data guided by standards or expert recommendations. Indeed, knowledge of the context of use can be useful in inferring the data that may be needed by users. Let us note that a multitude of information systems exist over the PLC (e.g. ERP, PDM, MES) which are not concerned by the same data (i.e. the same LDM entities). The idea is to identify specific "entity groups" through the LDM according to, for instance, the information systems and, therefore, to evaluate their importance over the PLC. Evaluations are performed by experts who focus on the entire PLC,
3. *Model-based* (C_m): this is based on the relationships implied by the LDM. This criterion favors data close to the product table[4]. Indeed, according to Chan and Roddick, the shorter the distance between tables, the higher will be the data correlation.

The following subsections detail when and how adjustments and computations of priorities ρ_x and $\phi_x(l)$ are carried out throughout the PLC.

2.2.1 Adjustment of the relative priority 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, with q the number of

[4] The product table is defined as that which includes d_p.

criteria. The importance of criterion i over criterion j is noted s_{ij}. This evaluation is based on the 1 to 9-point scale designed by Saaty [15]: $\{1, 3, 5, 7, 9\}$. $s_{ij} = 1$ means that critetria i and j are equal in importance and $s_{ij} = 9$ means that criterion i is strongly favored over criterion j. The relative importance of a criterion x is obtained by computing the eigenvalues of the matrix (eigenvalue noted $\lambda(\mathrm{C}_x)$) as in equation 3 and are synthesized by the vector Λ_ρ in equation 4.

$$D_\rho = \begin{array}{c} \\ 1 \\ \vdots \\ q \end{array} \begin{array}{c} \begin{array}{ccc} 1 & \dots & q \end{array} \\ \begin{bmatrix} s_{11} & \dots & s_{1q} \\ \vdots & \ddots & \vdots \\ s_{q1} & \dots & s_{qq} \end{bmatrix} \end{array} \quad s_{ji} = \begin{cases} 1 & i = j \\ s_{ij}^{-1} & i \neq j \end{cases} \tag{2}$$

$$\lambda(\mathrm{C}_x) = \frac{\sum_{k=1}^{q} s_{xk}}{\sum_{k=1}^{q} \sum_{l=1}^{q} s_{kl}} \qquad \forall\ x = [1, 2, .., q] \tag{3}$$

$$\Lambda_\rho = \begin{bmatrix} \lambda(\mathrm{C}_1) \cdots \lambda(\mathrm{C}_q) \end{bmatrix} = \begin{bmatrix} \rho_1 \cdots \rho_q \end{bmatrix} \tag{4}$$

2.2.2 Adjustment of the assigned priority $\phi_x(l)$

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

i. C_e: as mentioned above, this criterion gives a certain freedom to users to select information they deem relevant to store 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 detailed 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 $s(\mathrm{T}_t(v))$.

$$s(\mathrm{T}_t(v)) = \begin{cases} 1 & \text{enumerated} \\ 0 & \text{not enumerated} \end{cases} \qquad \text{with } v \text{ an attribute from table } t \tag{5}$$

ii. C_c: this criterion aims at moderating and balancing C_e. Indeed, users in C_e enumerates attributes they deem useful but they may have limited knowledge regarding the entire PLC. As introduced previously, the idea is to identify specific “entity groups” through the LDM according to, for instance, the information systems and, therefore, to evaluate their importance over the PLC. In our study, decision makers perform pairwise comparisons between entity groups as in equation 6, with w the number of entity groups. The importance of group i over group j is noted s_{ij} (based on the Saaty’s scale). The relative importance of an entity group i is obtained by computing its eigenvalue, noted $\lambda(\mathrm{G}_i)$, as in equation 7. All relative entity group importances are synthesized by the vector Λ_g in equation 8.

If a data item l is contained in a table included in G_i, its priority with respect to C_c, noted $\phi_c(l)$, is therefore equal to $\lambda(\mathrm{G}_i)$.

$$D_g = \begin{array}{c|c} & \begin{array}{ccc} \mathrm{G}_1 & \cdots & \mathrm{G}_w \end{array} \\ \begin{array}{c} \mathrm{G}_1 \\ \vdots \\ \mathrm{G}_w \end{array} & \begin{bmatrix} s_{11} & \cdots & s_{1z} \\ \vdots & \ddots & \vdots \\ s_{w1} & \cdots & s_{ww} \end{bmatrix} \end{array} \quad s_{ij} = \begin{cases} 1 & i = j \\ s_{ji}^{-1} & i \neq j \end{cases} \tag{6}$$

$$\lambda(\mathrm{G}_i) = \frac{\sum_{k=1}^{w} s_{ik}}{\sum_{k=1}^{w} \sum_{l=1}^{w} s_{kl}} \qquad \forall\ i = [1, 2, .., w] \tag{7}$$

$$\Lambda_g = \left[\lambda(\mathrm{G}_1) \cdots \lambda(\mathrm{G}_w)\right] \tag{8}$$

iii. C_m: the model-based criterion favors data close to the product table. This criterion is based on the relationships implied by the LDM. First, it is necessary to compute all distances between the product table A and any other table B. The distance corresponds to the shortest path reaching B from A (i.e. the minimal number of relations that separate them). For example, let `MaterialLot` be the product table A and `ManufacturingBill` be the table B in Fig. 1. The distance between both tables is 2 (relations ①-②). The product table would be the focus of our interest, with ϕ_m decreasing as the modeled distance increases. Chan and Roddick propose equation 9, with $k \in [1; \infty]$ a constant and $a \in \mathbb{N}$ the distance. The coefficient k must be adjusted. Fig. 2 highlights the fact that, for small values of k, the more remote information will be favored, and conversely (*cf.* $k = 1.01$ 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. It can be noted that it is not so simple for an expert to choose the most suitable value of k. To avoid such a problem, self-learning systems could be further implemented in order to learn about the used data model, the application features and thus, to automatically adjust k.

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

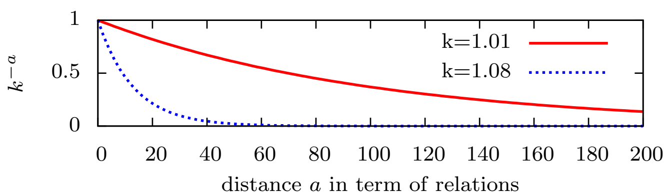


Fig. 2 Adjustment of the coefficient k according to the LDM structure

3 Process step 2: Storage/Retrieval of data on/from the product

Process step 2 deals with the storage of data items (identified in process step 1) on/from the product. Appropriate materials and architectures have been developed in our previous works. In [9], a communicating textile is designed in which a huge quantity of RFID μtags is spread ($\approx$ 1500 tags/m^2). Fig. 3 shows the communicating textile being processed. In [10], a communication protocol for splitting data items among several RFID tags is developed. Fig. 4 resumes the architectures/tools required to store data items on the product/textile (Fig. 4(a)) and then, to retrieve it (Fig. 4(b)):

1. Storage of information : information is written "on the fly" on the material as depicted in Fig. 4(a). The architecture consists of :
   - a RFID reader,
   - a software, named "Java/Matlab" (developed in Java® and Matlab® programming languages), for extracting information from the database and for computing the data item relevances. This software implements the approach proposed in this paper,
   - a software, named "Java Split" (developed in Java programming language), for splitting data items over the material. This software uses a specific protocol header developed in [10],
2. Retrieval of information : information carried by the material is read "on the fly" as depicted in Fig. 4. The architecture consists of :
   - a RFID reader,
   - the "Java Split" application to reconstruct the set of data items/tables,
   - optionally, an access to the database when information retrieved from the product must be updated in the database or when additional information is required (e.g. to answer a query).

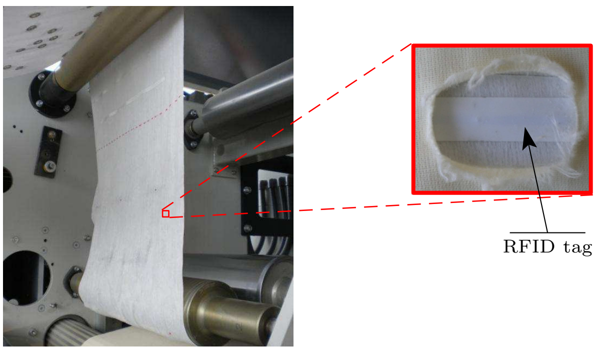
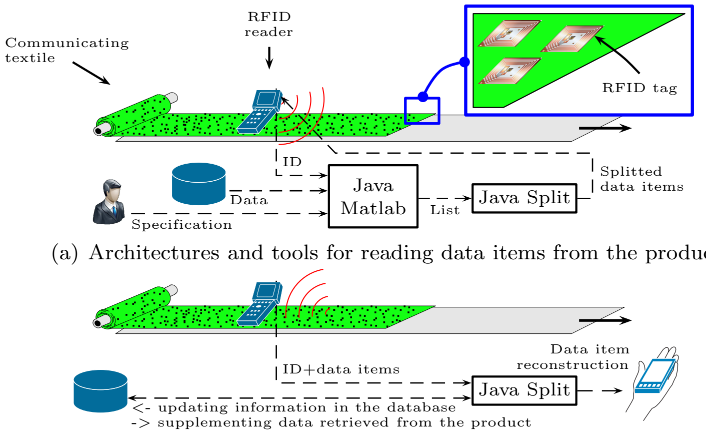


Fig. 3 Design of a communicating textile prototype: [9]

The platform detailed in this section is generic and can be used with any kind of communicating material (wood, cement, . . .). The only condition is to use a communicating material that integrates RFID tags R/W (Readable & Writable), whatever the memory space available in tags.

4 Case study

In our case study, only the part of PLC related to communicating textile reels is considered and is considerably simplified, as shown in Fig. 5. The goal of



(a) Architectures and tools for reading data items from the product

(b) Architectures and tools for writing data items on the product

Fig. 4 Architectures and tools of the data dissemination process

the case study is to focus on moments of the PLC where users want to store or retrieve information on/from the textile. Let us assume that communicating textiles have already been designed in phase 1 of the PLC (see Fig. 5). In our application, the communicating textile prototype designed in [9] is used. Let us consider that a supply chain member wants to store information on the textile at the `Writing point A` in phase 1 of the PLC. Accordingly, the architecture for writing information on this one is implemented as illustrated in Fig. 5. Then, a second supply chain member in phase 2 of the PLC, who works in cutting the textile in several pieces, wants to retrieve information carried by the textile reels (before cutting it). Accordingly, the architecture for reading information on the textile is implemented. In this study, the database implements a part of B2MML (Business To Manufacturing Markup Language) standard[5] [7] (19 entities exactly).

Section 4.1 details, on the one hand, the specifications/adjustments made by users concerned by the `Writing point A` and, on the other hand, the stages of computation needed to obtain the data item relevance values and ranking. Finally, section 4.2 briefly focuses on the retrieval of information from the textile in the `cutting operation`.

4.1 Writing point A: phase 1 of the PLC

This section details the specifications made by the different experts with regard to the different criteria. These multiple sources of expertise will be described in section 4.1.1 in the following order:

i. *Enumeration expertise*: users enumerate information they deem important to store on the product,

[5] B2MML is meant to be a common data format to link business enterprise applications with manufacturing enterprise applications.

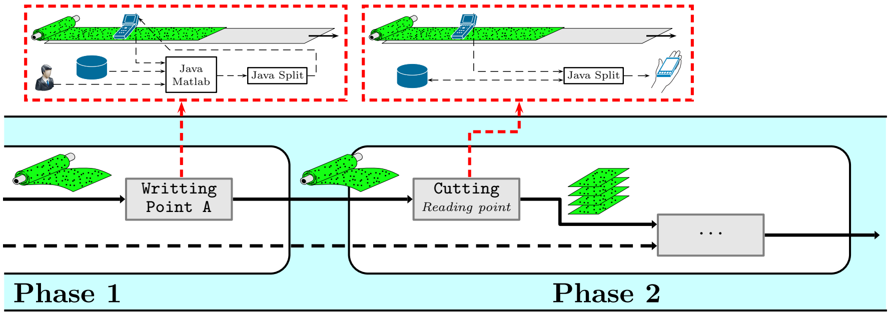


Fig. 5 Sequence of manufacturing activities related to a communicating textile application

ii. *Contextual expertise*: experts define the entity groups and adjust their importance over the PLC,
iii. *Model-based expertise*: experts adjust the coefficient k,
iv. *Criterion expertise*: experts define the criteria importance.

Then, section 4.1.2 details respectively the execution of process step 1 and 2 when the product arrives to the `writing point A`.

4.1.1 Expert specifications

i. The users concerned by the `writing point A` enumerate the information (i.e., the table attributes) listed in Table 2. These relate to the *Product data* group (G_1) and the *Production data* group (G_3).

Table 2 Table attributes enumerated in step 7

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

ii. The group of experts has convened a meeting before the textile reel begins its processing to adjust the contextual weights. They define four entity groups using the LDM, which are detailed in Table 3. The *Equipment* and *Personal information* groups report on equipment and people that are somehow related to the product (e.g., equipment used in its manufacture). The *Material* and *Production data* groups report on bills of materials(e.g., raw materials and component parts) and operations (e.g., production rules and production scheduling), respectively. Fig. 1 shows four of the 19 entities that are included in G_1 and G_3. The experts then carry out pairwise comparisons for the four groups with respect to each phase of the textile-transformation process. They define the pairwise comparison given in equation 10 regarding the first phase of the PLC (see Fig. 5). Note that the experts strongly favor G_1 over G_2

($s_{12} = 9$) and slightly favor G_1 over G_3 ($s_{13} = 3$). The normalized eigenvalue with regard to G_1, noted $\lambda(\mathrm{G}_1)$, is computed in equation 11. Eigenvalues related to the four entity groups are finally synthesized by Λ_g in equation 12 and it can be noted that information related to G_1 is much more important than information from the other groups: $\lambda(\mathrm{G}_2) < \lambda(\mathrm{G}_4) < \lambda(\mathrm{G}_3) < \lambda(\mathrm{G}_1)$.

$$D_g = \begin{array}{c} \\ \mathrm{G}_1 \\ \mathrm{G}_2 \\ \mathrm{G}_3 \\ \mathrm{G}_4 \end{array} \begin{array}{c} \begin{array}{cccc} \mathrm{G}_1 & \mathrm{G}_2 & \mathrm{G}_3 & \mathrm{G}_4 \end{array} \\ \begin{bmatrix} 1 & 9 & 3 & 7 \\ \frac{1}{9} & 1 & \frac{1}{7} & \frac{1}{5} \\ \frac{1}{3} & 7 & 1 & 7 \\ \frac{1}{7} & 5 & \frac{1}{7} & 1 \end{bmatrix} \end{array} \tag{10}$$

$$\lambda(\mathrm{G}_1) = \frac{1+9+3+7}{1+9+3+7+\frac{1}{9}+1+\ldots+\frac{1}{7}+1} = 0.55 \tag{11}$$

$$\Lambda_c = \begin{array}{c} \begin{array}{cccc} \lambda(\mathrm{G}_1) & \lambda(\mathrm{G}_2) & \lambda(\mathrm{G}_3) & \lambda(\mathrm{G}_4) \end{array} \\ \begin{bmatrix} 0.55 & 0.04 & 0.31 & 0.10 \end{bmatrix} \end{array} \tag{12}$$

Table 3 Definition of four entity groups

Material (G_1)	Personal (G_2)	Production (G_3)	Equipment (G_4)
`MaterialLot`	`Person`	`ProductionOrder`	`Equipment`
`MaterialDefinition`	`PersonClass`	`ProductSegment`	`EquipmentClass`
`MaterialClass`	`ActualPersonSeg.`	`ProductDefinition`	`EquipmentSegmentSpecif.`
`ManufacturingBill`	`PersonSegmentSpecif.`	`SegmentRequirement`	`ActualEquipmentSegment`
`ActualMaterialLot`		`SegmentResponse`	
`MaterialSegmentSpecif.`			

iii. The experts then proceed to the adjustment of coefficient k. Because the model is not very large (in our case study, the maximum distance through the LDM is 15), the experts agree on a value of 1.08 for k (see Fig. 2).

iv. Finally, experts carry out pairwise comparisons for all criteria as in equation 13. The normalized eigenvector Λ_ρ is then computed and is provided in 14. Note that the experts highly favor C_e over both C_c and C_m ($\rho_m < \rho_c < \rho_e$). This means that the experts prefer to store information they deem relevant instead of information contextually important.

$$D_\rho = \begin{array}{c} \\ \mathrm{C}_e \\ \mathrm{C}_c \\ \mathrm{C}_m \end{array} \begin{array}{c} \begin{array}{ccc} \mathrm{C}_e & \mathrm{C}_c & \mathrm{C}_m \end{array} \\ \begin{bmatrix} 1 & 5 & 7 \\ \frac{1}{5} & 1 & 5 \\ \frac{1}{7} & \frac{1}{5} & 1 \end{bmatrix} \end{array} \tag{13}$$

$$\Lambda_\rho = \begin{array}{c} \begin{array}{ccc} \rho_e & \rho_c & \rho_m \end{array} \\ \begin{bmatrix} 0.72 & 0.22 & 0.06 \end{bmatrix} \end{array} \tag{14}$$

All specifications needed for the `writing point A` are now defined and the process steps 1 and 2 are respectively launched when the communicating textile arrives at this point.

4.1.2 Execution of process step 1 *and* 2

Assume now that a textile arrives at the `writing point A` (*cf.* Fig. 5).

First, it is necessary to start the protocol for identifying product-related information from all tables. To achieve this, the set of tables $\mathcal{T}$ that compose the LDM are explored using the function `RetrievalData` (see Algorithm 1). Now, the algorithm steps will be outlined by focusing on three of the relational tables $\in \mathcal{T}$, namely {`MaterialLot`, `MaterialDefinition` and `ManufacturingBill`} illustrated in Fig. 6. In the following explanation, these three tables are denoted {`ML`, `MD`, `MB`}, respectively. To run the algorithm, the RFID reader obtains the textile identifier d_p, which refers to the data item $\mathrm{T}_{\mathrm{ML}}\{2, 1\}$ = `LPB61` (i.e., the product-lot instance), as shown in Fig. 6 (*cf.* "☞"). An exploration of the LDM based on `ExplorePK` then begins with d_p = `LPB61`. First, the tuple containing d_p is retrieved (i.e. row 2 of `ML` which is a "product-related tuple"). Second, the function discovers if there are FKs with respect to this tuple. This is true for `MD06` (i.e., $\mathrm{T}_{\mathrm{ML}}\{2, 5\}$) which is a PK in `MD` as highlighted in Fig. 6 (blue/solid arrow). The function can therefore continue and row 2 of `MD` is identified as a product-related tuple. The function then checks if there are FKs in this tuple. This is not true and, consequently, the algorithm cannot explore the LDM further. To sidestep such a break, then exploration based on`ExploreFK`, which searches for other relations where PKs from tables already explored are FKs in tables yet to be explored. In our LDM, `MB` is a neighbour of `MD` (see Fig. 1) and has yet to be explored. As a result, `ExploreFK` checks if there are PKs from `MD` which are FKs in `MB`. This is true for `MD06` as highlighted in Fig. 6 (red/dashed arrow). Therefore, new product-related tuples are identified (tuples 1 and 3 of `MB`). These new tuples serve as inputs for exploring the rest of the LDM by reusing `ExplorePK`.

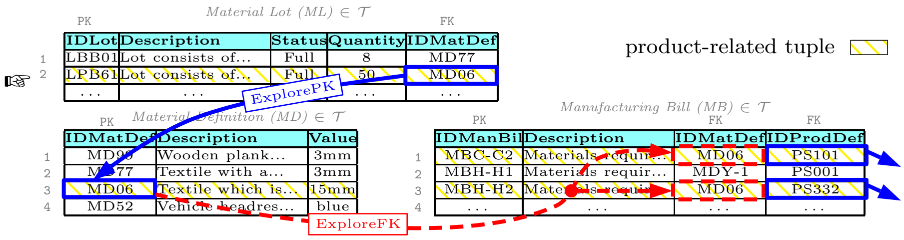


Fig. 6 Identification of "product-related tuples" thanks to `RetrievalData`

Second, it is necessary to compute the relevance of the retrieved product-related information. Fig. 7 details the example of `MaterialDefinition`. All data items that have no relation with d_p are set to 0. Regarding the Enumeration criterion, attributes `IDMatDef` and `Description` are set to 1 because they are enumerated in Table 2. Regarding the Contextual criterion, all attributes are set to 0.55 because `MaterialDefinition` is included in G_1, which has an importance of 0.55 (*cf.* equation 12). For the last criterion, Model-based, the weight is set to k^{-1} because the distance between `MaterialDefinition` and `MaterialLot`[6] is 1 (see Fig. 1). The relevance computation is then performed for all data items as exemplified with $\mathrm{T}_{\mathrm{MD}\{3,1\}}$ with $k = 1.08$, ρ_e, ρ_c, ρ_m respectively equal to 0.72, 0.22, 0.06 (see equation 14) and $len(\texttt{MD06001}) = 7$ (one ASCII character uses 1 byte). Finally, the relevance $P_{\mathrm{T}_{\mathrm{MD}}\{3,1\}}$ becomes 0.43.

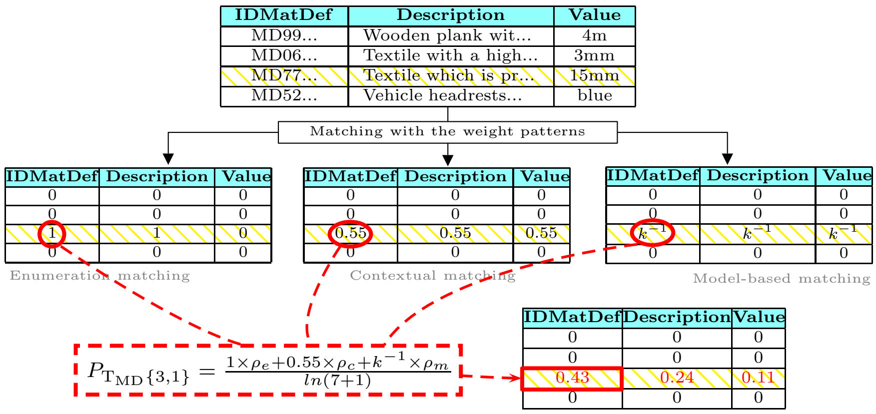


Fig. 7 Relevance computation of "product-related data items"

In this part, the results about relevances of all data items are discussed. The term "list" is used to refer to all data items having a prioritization $P_l > 0$. Fig. 8(a) gives the ordered list from the highest P_l to the lowest. The most relevant data item (with $P_l = 0.7066$) is the one located in the `ProductSegment` table, for the attribute `Duration`, related to the tuple whose PK is `PSTBOB`. Because of the large number of data items included in the list (524 exactly), results are presented in the form of diagrams.

First, consider the whisker diagram in Fig. 8(b). For each table $\mathrm{T}_t \in \mathcal{T}$, the figure shows the minimum, the 1^{st} and 3^{rd} quartile, the mean and the maximum P_l value(s) related to the set of data items included in the list and belonging to T_t. Note that the highest relevant data items come from `ProductSegment`, but also highly relevant are `MaterialDefinition`, `ProductionOrder` and `MaterialLot`. This is because some attributes of these tables are enumerated (see Table 2) and the experts place high importance on the opinion of the users ($\rho_m < \rho_c < \rho_e$). Moreover, note that these four tables are included in the entity groups G_1 and G_3 and that the experts recommended highly that information from both these groups should be selected ($\lambda(\mathrm{G}_2) < \lambda(\mathrm{G}_4) < \lambda(\mathrm{G}_3) < \lambda(\mathrm{G}_1)$).

Consider now the pie chart in Fig. 8(c). The percentages are relative to a given data group G_i, indicating the proportion of the data items that are in the first part of the list (i.e., that should be stored on the communicating textile) and that belong to G_i. At the `writing point A`, no more than 159 data

[6] `MaterialLot` is the reference table because it contains d_p.

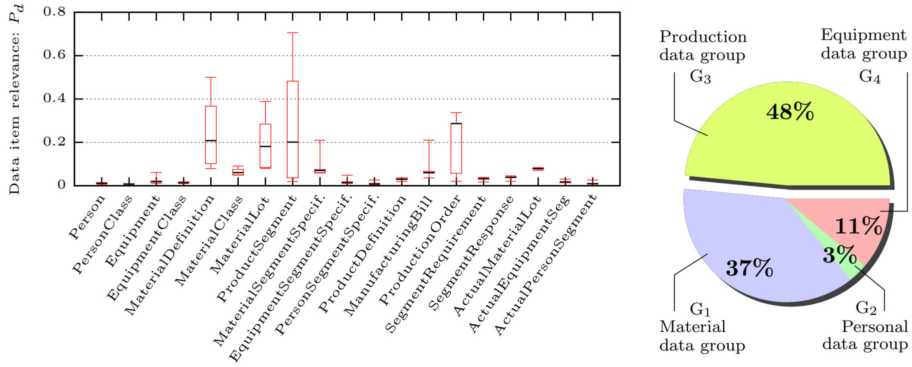
Rang	Table Name	Attribute Name	PK	Cell Value	P_l
1^{st}	ProductSegment	Duration	PSTBOB	0.5	0.7066
2^{st}	ProductSegment	Unit duration	PSTBOB	Hours	0.6102
3^{th}	MaterialDefinition	IDMatDef	MD06	15C	0.4311
...	...	...	...	...	...
159^{th}	SegmentResponse	ActualStartTime	PSR0001	10:15:00	0.0567
⋮	⋮	⋮	⋮	⋮	⋮
523^{th}	Equipment	Description	ET3A01	Line 3A,...	0.0051
524^{th}	PersonClass	Description	PCPLW01	Production...	0.0051

High probability that data items are stored on the product

Storage limit on the material

Lesser probability of being stored on the product

(a) List of data items ordered from the highest P_l to the lowest



(b) Whisker diagram (c) Pie chart

Fig. 8 Results of the data item relevance

items can be stored on the communicating textile because there is insufficient memory space, as highlighted in Fig. 8(a). Fig. 8(c) indicates that 48% of the 159 data items (i.e., ≈ 72) are data items from the *production data* group and should be stored on the communicating textile. The pie chart clearly accords with the previous explanation, because most of the data items stored on the textile come from G_1 and G_3, accounting for 37% and 48%, respectively. This is largely due to the choices made for the enumeration and contextual criteria. In contrast, the *personal data* group (G_2) is not relevant with 2% of data items embedded on the product, because there is no enumeration and no high recommendation from the contextual criterion. At this stage, data items to be stored on the communicating product are selected starting from the top of the list until the storage limit of the communicating material is reached. However, further work should be undertaken to optimize this list and to ensure that the maximum number of queries are answerable.

The process step 2 is then performed and the 159 data items are stored/ spread over the communicating textile reel thanks to the software "Java Split" (see Fig. 4(a)). For information purposes, the software spent $\approx 40s$ to perform this operation, but let us remind ourselves that the primarily goal of the research presented in this paper is devoted to investigate the new concept of "communicating material" and to propose the first tools to communicate with it. Further work will lead to more effective tools.

4.2 Reading phase: cutting operation in Phase 2 of the PLC

The textile reel arrives at the `cutting operation` in Stage 2 (see Fig. 5), where the machine/user wishes to retrieve information carried by that one. As described above, the communication architecture defined in Fig. 4(a) is implemented on the production line. The textile is then read and the data-retrieving operation is achieved. The software "Java Split" is once again used, but this time, for rebuilding and displaying the 159 data items on a mobile device as shown in Fig. 9 with the table `MaterialDefinition` (the software spent $\approx 30s$ this time to perform the operation). It can be seen that the three data items composing `MaterialDefinition` were included in the first 159^{th} ranks of the *list*. However, in some cases, it is impossible to rebuild the entire tuple because of some data items are located in the second part of the list (i.e., $> 159^{th}$) and are not stored on the communicating textile.

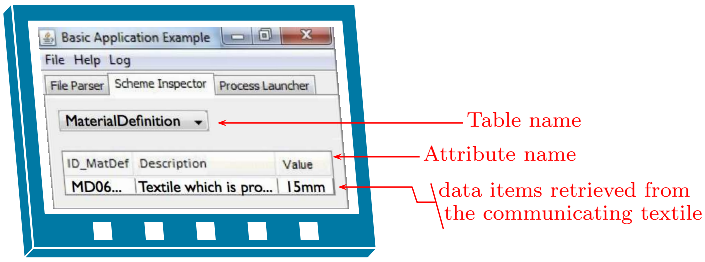


Fig. 9 JAVA software to display the product-related information

5 Conclusion

It is not uncommon today to use intelligent/communicating products to create an information continuum over the product life cycle (PLC) (e.g., for traceability purposes). Indeed, considering the product as an information vector (within which information can be stored) would contribute to improve interoperability throughout its PLC. However, it is not that easy to identify, at any given stage in the PLC, the information that should be stored on the product. To address this issue, a data-dissemination process is developed in this paper for selecting context-sensitive information from a database and storing it on the product. A case study gives insights into appropriate storage/retrieval of data on/from a new kind of material described as a "communicating material". The results of this scenario shows that the selected data (i.e. data that is stored on the product) largely meets the expectations formulated by users at a given stage of the PLC. Further work should investigate new approaches to deal with particularities of the context of use of the product. For instance, ontologies could be used as a complement or a substitute of our approach [5, 11] and should be examined.

As regards the "communicating material" concept, several advantages may be emphasized compared to products currently being used (e.g. a product with a unique RFID tag). For example, it is both possible to perform data redundancy on the product and to ensure data sustainability over the PLC. Indeed, a same data can be copied to several parts over the material that is useful in cases where it is relevant that little or no product-related data is lost. Further, new substances could be considered to design "communicating materials" like wood, cement or still paint.

References

1. Abuelyaman, E.S. (2008). An optimized scheme for vertical partitioning of a distributed database. *International journal of computer science & network security*, 8(1), 310–316
2. Babiceanu, R.F., Chen, F.F. (2006). Development and applications of holonic manufacturing systems: a survey. *Journal of Intelligent Manufacturing*, 17(1), 111–131.
3. Brewer, A., Sloan, N., Landers, T. (1999). Intelligent tracking in manufacturing. *Journal of Intelligent Manufacturing*, 10(3), 245–250.
4. Chan, D., Roddick, J. (2003). Context-sensitive mobile database summarisation. In: *26th Australasian computer science conference*, 139–149.
5. Chandra, C., Kamrani, A.K. (2003). Knowledge management for consumer-focused product design. *Journal of Intelligent Manufacturing*, 14(6), 557–580.
6. Främling, K., Ala-Risku, T., Kärkkäinen, M., Holmström, J. (2006). Agent-based model for managing composite product information. *Computers in Industry*, 57(1), 72–81
7. ISO: Enterprise - control systems integration (2003). *IEC/ISO 62264-1.*
8. Kubler, S., Derigent, W., Thomas, A., Rondeau, É (2010). Problem definition methodology for the "Communicating Material" paradigm. In: *10th IFAC Workshop on Intelligent Manufacturing Systems.*
9. Kubler, S., Derigent, W., Thomas, A., Rondeau, É (2011). Prototyping of a communicating textile. In: *International Conference on Industrial Engineering and Systems Management*, 1333–1342.
10. Kubler, S., Derigent, W., Thomas, A., Rondeau, É (2012). Storage of information on manufactured products using "communicating materials". In: *1st Worshop on AI Problems and Approaches for Intelligent Environments*, 9–14.
11. Goncalves, R., Sarraipa, J., Agostinho, C., Panetto, H. (2011). Knowledge framework for intelligent manufacturing systems. *Journal of Intelligent Manufacturing*, 22(5), 725-735.
12. McFarlane, D., Giannikas, V., Wong, A.C., Harrison, M. (2012). Intelligent products in the supply chain - 10 years on. In: *14th Symposium of Information Control Problems in Manufacturing.*
13. Mehrabi, M.G., Ulsoy, A.G., Koren, Y. (2000). Reconfigurable manufacturing systems: key to future manufacturing. *Journal of Intelligent Manufacturing*, 11(4), 403–419.
14. Meyer, G., Främling, K., Holmström, J. (2009). Intelligent products: A survey. *Computers in Industry*, 60(3), 137–148.
15. Saaty, T. (1980). *The Analytic Hierarchy Process.* (Ed.) New York: McGraw-Hill
16. Thomas, A. (2009). RFID et nouvelles technologies de communication; enjeux économiques incontournables et problèmes d'éthique. In: *6ème Conférence Internationale Conception et Production Integrées.*
17. Weiser, M. (1993). Hot topics-ubiquitous computing. *Computer*, 26(10), 71–72.
18. Wong, C.Y., Mcfarlane, D., Zaharudin, A.A., Agarwal, V. (2002). The Intelligent Product Driven Supply Chain. In: *International Conference on Systems, Man and Cybernetics*, 4–6.
19. Yan, L., Zhang, Y., Yang, L., Ning, H. (2008). *The Internet of Things: From RFID to the Next-Generation Pervasive Networked Systems.* (Ed.) Auerbach Publications.
20. Zbib, N., Pach, C., Sallez, Y., Trentesaux, D. (2012). Heterarchical production control in manufacturing systems using the potential fields concept. *Journal of Intelligent Manufacturing*, 23(5), 1649-1670.