Storage of information on manufactured products using ”communicating materials”

Sylvain Kubler[1] and **William Derigent**[1] and **Éric Rondeau**[1] and **André Thomas**[1]

Abstract. The amount of data output into our environment is increasing each day, and the development of new technologies constantly redefines how we interact with this information. It is therefore necessary to control the different ways information is diffused. As a result, a data dissemination methodology in the framework of the Supply Chain Management is introduced. A specific stage of the methodology is detailed in this paper which aims at storing directly on the product, relevant data to the subsequent users. To do so, a new type of product is presented referred to as ”communicating material”.

1 Introduction

New challenges and opportunities arise with concepts such as Internet of Things (IoT), Ubiquitous/Pervasive Computing [15] or still Artificial Intelligence [12]. Through these concepts, objects of the real world are linked with the virtual world. Thus, connections are not just people to people or people to computers, but people to things and most strikingly, things to things [14, 13]. Ley [8] quotes the example of clothes able to carry their own information, and thus enabling the washing machine to automatically adapt its washing program. Such applications rely on ever more complex information systems combined with ever increasing data volumes, which are stored in a large number of information vectors. These vectors may be fixed (computers) or mobile (wireless devices, RFID).

Any product, during its life, passes through numerous companies and undergoes various operations (manufacturing, transportation, recycling...). Technical, semantic and organizational interoperability between these companies is not always ensured, thus, conducing to information loss. If one considers the product as an information vector (on which information could be stored), it would contribute to improve interoperability all along its life cycle. Meyer et al. [9] provides a complete survey on *intelligent products*, i.e. products carrying their own information and intelligence. Främling et al. [2] argue that it is a formidable challenge to link the product related information to the products themselves, making the information of all the product components easily achievable. However, most of the time, products are only given an identifier (e.g. via a RFID tag) which provides a network pointer to a linked database and decision making software agent [10]. Moreover, this kind of product is still limited on some points: risk of tag damage, small memory capacity, problem of data transfer (e.g. when the product is cut), etc.

As a result, we propose a new concept referred to as *communicating material*, which considers the material as intrinsically communicating. First, recent works carried out on this concept are discussed in section 2. This corpus of works mainly focus on the development of a data dissemination process to identify what the relevant information to users is and where it should be stored: on databases or on the product themselves? One important step of this framework deals with the storage/retrieval of data on/from the communicating materials, which is the subject of the paper. An appropriate architecture of communication is developed in section 3 which aims at splitting data over the communicating material and, at determining where this information is (or will be) located. An applicative scenario is finally presented in section 4.

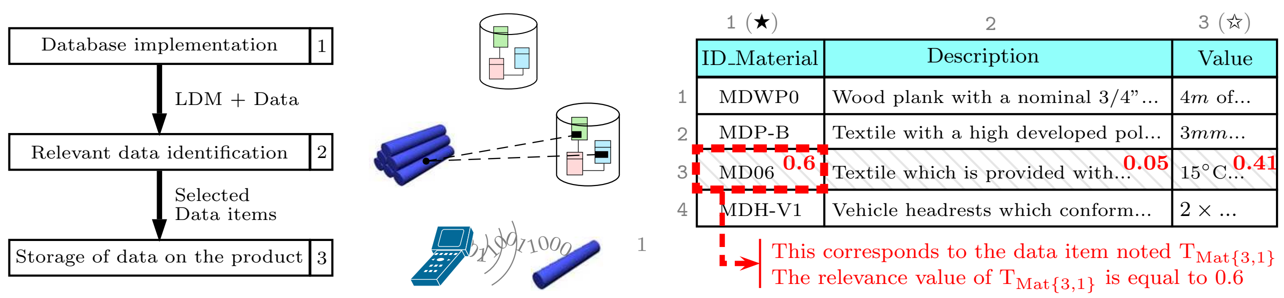
2 Data dissemination process

As said previously, the product passes through numerous companies during its life cycle. Each actor/operation requires product related information (e.g., for decision-making, production orders) which is not always available (inaccessible information, e.g. owned by another supplier) and is not always up-to-date [4] (unavailable information, e.g not shared by the supplier). Accordingly, solutions and platforms have emerged to link the product related information to the products themselves such as EPCglobal, ID@URI or WWAI [9]. However, information is generally deported on the network through these solutions because products are memory-constrained and the question of what information is relevant to users and where information needs to be stored is not answered. For this reason, we have proposed in recent publications [6, 7] a data dissemination process consisting of 3 steps as shown in Figure 1(a):

➭ Process step 1 consists in implementing the database system architecture, which can be either centralized or distributed. Many works in the literature help the designer to choose the more suitable system according to the application constraints [11].

➭ Process step 2 aims at selecting *context-sensitive information*. When users want to write information on the product at a given time (e.g. before the product leaves the company), it is necessary to identify information that is relevant (e.g. for subsequent users). This identification is achieved thanks to the process step 2. Our approach, detailed in [6], consists in assessing what is the relevance of storing a given data

[1] Centre de Recherche en Automatique de Nancy, Université de Lorraine, CNRS UMR 7039
Campus Sciences, BP 70239, F-54506 Vandœuvre-lès-Nancy Cedex, France, email: Sylvain.Kubler@cran.uhp-nancy.fr


Database implementation 1
LDM + Data
Relevant data identification 2
Selected Data items
Storage of data on the product 3
0100 1000
1
This corresponds to the data item noted $T_{Mat\{3,1\}}$
The relevance value of $T_{Mat\{3,1\}}$ is equal to 0.6

	1 (★)	2	3 (☆)
	ID_Material	Description	Value
1	MDWP0	Wood plank with a nominal 3/4"...	$4m$ of...
2	MDP-B	Textile with a high developed pol...	$3mm$...
3	MD06 **0.6**	Textile which is provided with... **0.05**	15°C... **0.41**
4	MDH-V1	Vehicle headrests which conform...	2 × ...

(a) Data dissemination process (b) Relational table *Material* & Data item relevance

Figure 1. Overview of the data dissemination process and the data assessment (data from a relational table)

item on the product according to the user concerns, the environment details, etc. One data item corresponds to a cell of a relational table (i.e. the intersection between a column, named "attribute", and a row, named "tuple") as emphasized in Figure 1(b). For instance, the data item located at row 3-column 1 in table `Material`, noted $T_{Mat\{3,1\}}$, has the value `MD06`. Only data items which have an interest of being stored on the product are assessed (such a selection is also proposed in [6]). For instance, only the tuples 3 in `Material` is assessed (see dashed background in Figure 1(b)) and the relevance of each data item from this tuple is computed (e.g. the relevance of $T_{Mat\{3,1\}} = 0.6$). To compute the relevance value, the approach uses the notion of priorities which are numerical values either supplied or generated through observation and experimentation and are assigned through a multi criteria evaluation [6]. At the end, all data items from all relational tables of the database are assessed and then, classified in order of relevancy. The higher the relevance value, the higher the probability that these data items will be stored on the product. The storage/retrieval of data items on/from the communicating material is done via the process step 3, as depicted in Figure 1(a).

➭ Two things are needed in process step 3: (i) first, the product must be instrumented in order to carry such information. As introduced previously, [5] propose a new concept referred to as *communicating material*, which considers the material as intrinsically and wholly communicating thanks to a huge amount of RFID μtags scattered in the material. Different textile prototypes were designed[2]through an industrial process with different types of RFID tags (μtags from Hitachi that can only store an identifier or Omron tags that can store up to 64 bytes). An example of communicating textile is shown in Figure 2. (ii) Secondly, it is necessary to design an architecture of communication to be able to store data fragments (i.e. data items) on a communicating material.

3 Data fragment storage/retrieval using communicating materials

Process step 3 of the data dissemination process deals with the storage of data items (identified across the process step 2) on the product. In this section, an appropriate architecture combined with a protocol of communication is developed in order to split data items over the material and then, to rebuild them. Subsequently, we develop a method to determine where data are located on the communicating material.

[2] Designed in collaboration with the CETELOR laboratory.

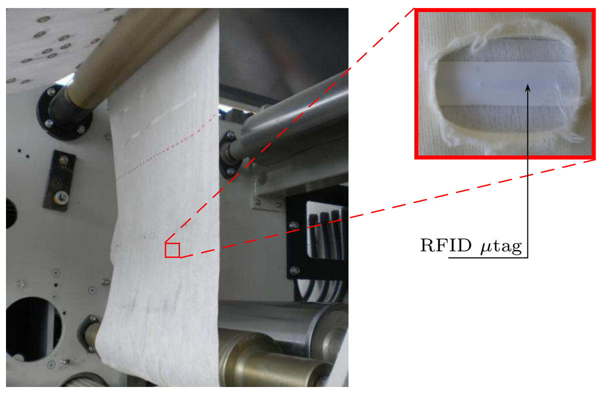
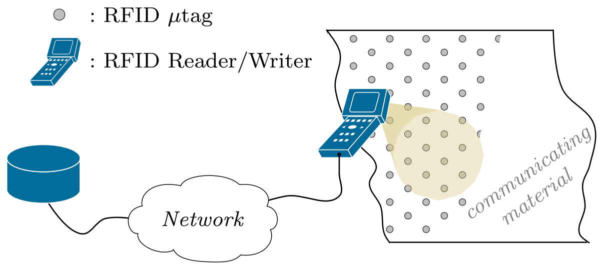


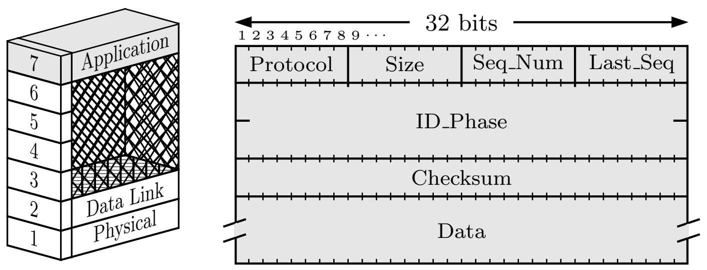
Figure 2. Prototype communicating textile designed in [5]

3.1 Data storage on communicating materials

First, it is necessary to have a communicating material, as the communicating textile designed in our previous work [5]. Let us remind ourselves that a huge quantity of RFID tags are spread over/in the material. Since the RFID tags are memory-constrained, the idea is to split the set of data items over several tags. To do so, a specific architecture must be implemented and a specific application protocol is developed (named *protocol of splitting* in our paper). Figure 3(a) depicts the global architecture with a RFID reader, a communicating material and a database (containing data which are assessed and which may be store on the material). The *protocol of splitting* respects the RFID standard ISO/IEC 18000-1 [3], which relies on the 1st, 2nd and 7th OSI layer as shown in Figure 3(b). Layer 1 corresponds to the physical part of the RFID (i.e antenna, analog part). Layer 2 deals with the communication protocol and especially the collision mechanisms. Layer 7 deals with the application data (this is the memory portion in which data can be added or modified by users). A RFID tag may store more or less information according to the technology and, therefore, one data item may require more memory space than that available in a unique tag. That is why we propose a protocol of splitting. This application protocol is obviously defined at layer 7 of the OSI model (cf. Figure 3(b): gray background). The application data consists of 7 fields, 6 are reserved to the header (used to rebuild the



(a) Global communication architecture



(b) Frame structure used in the *protocol of splitting*

Figure 3. Protocol of communication developed to read/write data fragments from/on communicating materials

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 which refers to the frame structure defined in Figure 3(b) (specific to our application),
2. **Size** (8 *bits*)**:** Integer from 0 to 255 which indicates the size of data included into the field `Data` (7^{th} field),
3. **Seq_Num** (8 *bits*)**:** Integer from 0 to 255. It provides the sequence number of the current frame (1 frame/tag). The sequence number is used to determine the order of the different frames that have been written over the RFID tags (needed to rebuild the set of data items),
4. **Last_Num** (8 *bits*)**:** Integer from 0 to 255. It provides the sequence number of the last frame which contains data related to the same writing phase. The notion "same writing phase" is important because one data item may be written at two different times in its life cycle and then, data inconsistency/conflict may occur,
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 (it does not detect errors in the header),
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`. For instance, when the data item $T_{\text{Mat}\{3,3\}}$ is written (see Figure 1(b)), the index `Material.Value.MD06` is added. Then, the data item content is added (i.e. `15°C...`).

Regarding the application layer, 128 bits (32 × 4) out of n useful are used in each tag for the application header. The application logic rebuilds each data item thanks to the `Seq_Num` and the `Last_Num`, which indicate in what order data items have been split. In section 4, the tag memory is itemized after having been written. Let us note that according to the set of data items carried by the product, queries may be answered or unanswered. Nevertheless, some methods can be deployed to know in advance (i.e. before data items will be written on the product) if queries may be answered or not (e.g. by transforming queries to corresponding bitmaps [1]).

3.2 Data location on communicating materials

In this section, we develop an approach to know where information will be written on the material (in case of writing) or where information is located (in case of reading). In certain production processes like textile manufacturing, materials are transformed with highly automatised machines, at high manufacturing speed. However, some defects may occur on specific material zones (e.g., holes, stains). Because machines are not aware of these defaults, end products could not be sold. In our vision, materials are consider hard disks, able to store information about themselves. If a defect is detected, this information is then stored on the material and can be reused ad libitum when needed. The type of data manipulated is area-related : consider a grease spot on the textile. Information related to the grease spot must be stored as close as possible to the real grease spot. As a result, information must be located. Our method is designed to help locating information on the material, thus enabling machines not to be blindfolded and to adapt their behavior. To do so, a more complex architecture is required than previously. Indeed, in the previous section, the data location over the material was not taking into account and therefore, data was written/read anywhere ('on the fly') as illustrated in Figure 3(a). However, if the user desires to know where a specific information is located over the material, he needs to use a specific method which is detailed hereafter.

First, let $\mathcal{C}$ be the set of RFID tags present in the material. Let $\mathcal{R}$ be the set of RFID readers which are aligned as depicted in Figure 4 (e.g. on a ramp), with $\mathcal{R} = \{\mathrm{r}_1, \mathrm{r}_2, \mathrm{r}_3\}$. Each RFID tag and RFID reader has a reference number respectively noted $\texttt{ID}_c$ and $\texttt{ID}_r$ with $c \in \mathcal{C}$ and $r \in \mathcal{R}$. A reader r generates an event $e_{r,c}$ if the presence of the tag c is detected. This event is made up of $\{\texttt{ID}_r, \texttt{ID}_c, \mathrm{t}_{r,c}\}$, with $\mathrm{t}_{r,c}$ the acquisition time of the event. All the events form the set $\mathcal{E}$.

Let us focus now on the algorithm which calculates the theoretical positions of the tags in the material (i.e. $\in \mathcal{C}$). First, it is necessary to model the *reading zone* (i.e. zone in which a RFID reader and a RFID tag can communicate) which depends on the RFID technology. This *reading zone* must be modeled as a cylinder[3] as depicted in Figure 4 (the three RFID readers have the same cylindrical shape). Once the *reading zone* is modeled, the algorithm for computing the location $2D$ of tags takes as input the set of events $\mathcal{E}$. Since the *reading zone* is modeled as a cylinder, it is possible to calculate

[3] A methodology to model the *reading zone* of a given RFID technology is presented in [5].

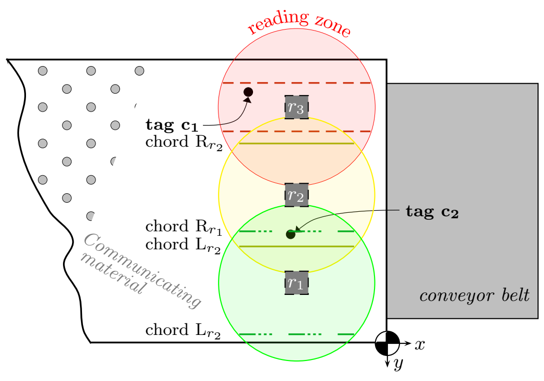


Figure 4. Architecture needed to locate information

the tag position based on the chord of the circle. The RFID reader transmits events as long as one tag is in its *reading zone* and as many events as acquisition cycles (i.e. if a tag c stays during n acquisition cycles, n events are generated). By this way, it is possible to calculate the chord based on the difference between the date of the last and the first event. Thus, we obtain two possible positions on y-axis via equation 1 for each RFID reader : r_1 and r_2 detect together c_2 and compute respectively the right chord (noted chord R_{r_1} and chord R_{r_2} in Figure 4) and the left chords (noted chord L_{r_1} and chord L_{r_2}) so as to locate c_2 ; r_3 only detects c_1 and proceeds to the same computation. Let us note that in computing the average between the two chords, the measured error is reduced by two. The tag coordinate in x axis is calculated via equation 2.

$$y_{c_t} = y_r \pm \sqrt{rad^2 - \left[\left(\frac{t_{r,c_L} - t_{r,c_F}}{2}\right) \times v\right]^2} \quad (1)$$

$$x_{c_t} = \left(\frac{t_{r,c_F} + t_{r,c_L}}{2} - t_s\right) \times v \quad (2)$$

x_{c_t}, y_{c_t}	theoretical coordinates x-y of the tag c ($c \in \mathcal{C}$)
x_r, y_r	coordinates in x and y-axis of the reader ($r \in \mathcal{R}$)
t_{r,c_F}	first date to which the reader r detected the tag c
t_{r,c_L}	last date to which the reader r detected the tag c
v	conveyor speed (or material speed)
t_s	textile detection date (thanks to a presence sensor)
rad	radius of the *reading zone*

The error made on the chord computation depends on both the acquisition time cycle and the conveyor speed. In fact, the higher the acquisition time cycle, the lesser the error. Obviously, the error depends also on the modeled *reading zone* (modeled as a perfect cylindrical shape). Figure 4 illustrates that the computed chords do not exactly overlap with the RFID tag c_1 and c_2. By using the chord approach, two possible chords are determined, which means that two possibilities are returned. If one tag is detected by only one RFID reader, it implies two possible coordinates y_{c_t} (impossible to know if the real tag is located on the left or right chord). This is illustrated in Figure 4 with the tag c_1 (only detected by $r3$). This problem does not occur when a tag is detected by several readers, as illustrated with c_2 (detected by r_1 and r_2). Indeed, the tag might most likely be located between ch_{r1_r} and ch_{r2_l}, as shown in Figure 4. The higher the number of readers detect one tag, the higher the precision.

4 Applicative scenario

In this scenario, we use the communicating textile reel designed in [5] (cf. Figure 2). This textile reel is preparing to leave the company X to a company Y as depicted in Figure 5 (it will be used for designing vehicle headrests). Accordingly, the supplier X decides to implement the data dissemination process for storing on the textile reel, the product related information which are useful for the supplier Y. This is done at `Activity X` as shown in Figure 5: implementation of step 1, 2 and 3 for identifying and storing the relevant data from the database, on the communicating textile reel. Let us note that the supplier X does not mind where information will be stored on the textile (i.e. it is unnecessary to use the architecture from Figure 4). Then, the textile reel will continue its transformation and will arrive at `Activity Y` as depicted in Figure 5. In this activity, the machine retrieve information carried by that one and wants to locate information on the textile (i.e. it is necessary to use the architecture from Figure 4). Both activities must implement the *protocol of splitting* detailed in section 3.1. Moreover, `Activity Y` must implement the architecture and algorithm detailed in section 3.2 for locating data on the textile.

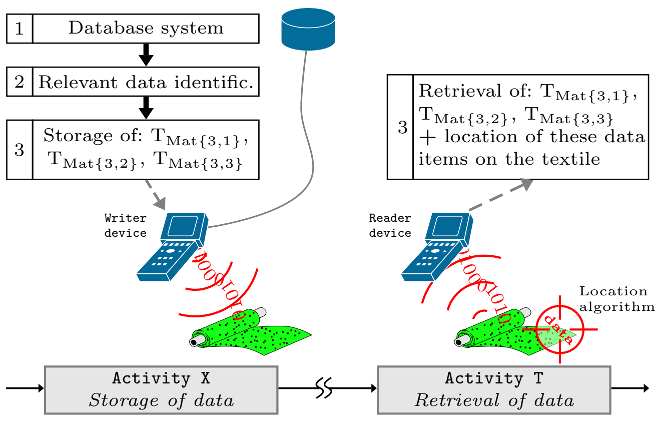
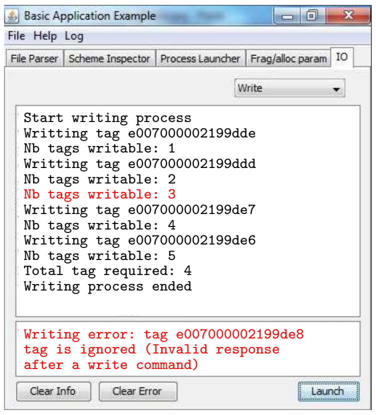

1 Database system
2 Relevant data identific.
3 Storage of: T_{Mat{3,1}}, T_{Mat{3,2}}, T_{Mat{3,3}}
3 Retrieval of: T_{Mat{3,1}}, T_{Mat{3,2}}, T_{Mat{3,3}} + location of these data items on the textile
Writer device
Reader device
Location algorithm
data
Activity X
Storage of data
Activity T
Retrieval of data

Figure 5. Description of the scenario applicative activities

4.1 Data storage on the communicating textile prototype: Activity X

The communicating textile arrives at `Activity X`, where the machine starts the relevant data identification (i.e. step 2 of the data dissemination process). This step returns the set of data items that need to be stored on the textile. Let us assume that among these data items, the ones composing the tuple 3 of table `Material` (see Figure 1(b)) are included. The machine then starts to write $T_{Mat\{3,1\}}$, $T_{Mat\{3,2\}}$ and $T_{Mat\{3,3\}}$ over the communicating textile thanks to the *protocol of splitting* described in section 3.1, as depicted in Figure 5.

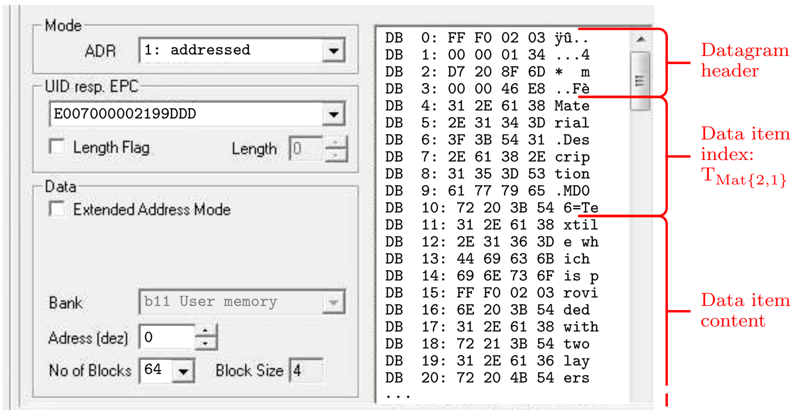
As said before, the supplier in `Activity X` does not mind where information is written over the textile. As a result, we use the architecture given in Figure 3(a) (i.e. data is written 'on the fly'). Figure 6(a) shows the log events generated when writing the three data items over the textile. It can be observed that 4 tags are required for splitting the three data items. However, 5 tags have been written because one



```
Start writing process
Writting tag e007000002199dde
Nb tags writable: 1
Writting tag e007000002199ddd
Nb tags writable: 2
Nb tags writable: 3
Writting tag e007000002199de7
Nb tags writable: 4
Writting tag e007000002199de6
Nb tags writable: 5
Total tag required: 4
Writing process ended
```

```
Writing error: tag e007000002199de8
tag is ignored (Invalid response
after a write command)
```

(a) Log events of the writing process



ADR: 1: addressed
UID resp. EPC: E007000002199DDD
Length: 0
Bank: b11 User memory
Adress (dez): 0
No of Blocks: 64
Block Size: 4

```
DB  0: FF F0 02 03 ÿû..
DB  1: 00 00 01 34 ...4
DB  2: D7 20 8F 6D *  m
DB  3: 00 00 46 E8 ..Fè
DB  4: 31 2E 61 38 Mate
DB  5: 2E 31 34 3D rial
DB  6: 3F 3B 54 31 .Des
DB  7: 2E 61 38 2E crip
DB  8: 31 35 3D 53 tion
DB  9: 61 77 79 65 .MD0
DB  10: 72 20 3B 54 6=Te
DB  11: 31 2E 61 38 xtil
DB  12: 2E 31 36 3D e wh
DB  13: 44 69 63 6B ich
DB  14: 69 6E 73 6F is p
DB  15: FF F0 02 03 rovi
DB  16: 6E 20 3B 54 ded
DB  17: 31 2E 61 38 with
DB  18: 72 21 3B 54 two
DB  19: 31 2E 61 36 lay
DB  20: 72 20 4B 54 ers
...
```

(b) List of data items ordered from the highest P_d to the smallest

Figure 6. Results of the data item relevance

write operation failed (tag e007000002199de8). This highlights that all data items are stored on the material even if writing errors occur[4]. Now, let us focus on the datagram content of a given RFID tag. Figure 6(b) details the datagram contained in the tag e007000002199ddd after having written the three data items. It is important to note that a RFID tag memory (whatever the RFID technology) consists of several Data Blocks (denoted DB in Figure 6(b)). The RFID tags disseminated in our textile are the *Omron's V720-D52P03* and their memory is divided in 64 blocks of size 4 bytes[5]. Accordingly, the datagram header occupies the four first bytes of each tag memory as highlighted in Figure 6(b) (DB0 to DB3). Then, the remaining DB are exclusively reserved to the content of data items. However, let us remind that one index is added at beginning of each data item in order to locate it in the database (cf. section 3.1). Figure 6(b) shows the index related to $T_{Mat\{3,2\}}$, which is Material.Description.MD06. Then, the content of this data item (string of value: Textile which is provided with two layers of protective coatings) is added (cf. Figure 6(b)).

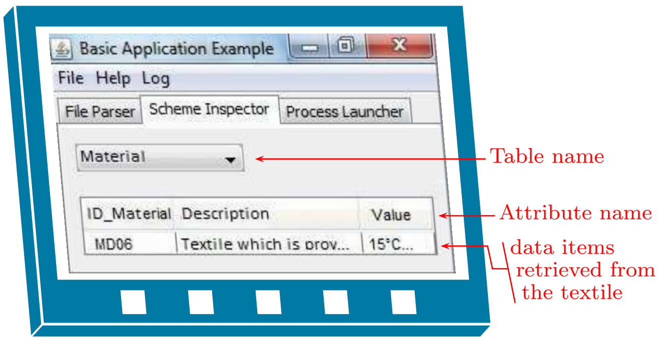
Then, the textile reel arrives at Activity Y in which the machine requires data carried by that one and must identify where it is located over the textile.

4.2 Data retrieval and data location: Activity Y

Since information must be located over the textile, the architecture described in Figure 4 is implemented. Then, the communicating textile passes under the ramp of RFID readers and is read. The three data items are retrieved and their content is displayed via an application software (JAVA programming) as shown in Figure 7. Then, services can be programmed: for instance, queries may be directly performed via the JAVA software based on the set of data items retrieved from the communicating textile (unanswered queries could happen). Many services may therefore be imagined as the example of Ley [8] with the washing machine (cf. section 1).

[4] This depends on the technology but most RFID technologies implement error correction mechanisms.

[5] Let us remind that one ASCII character occupies 1 byte.



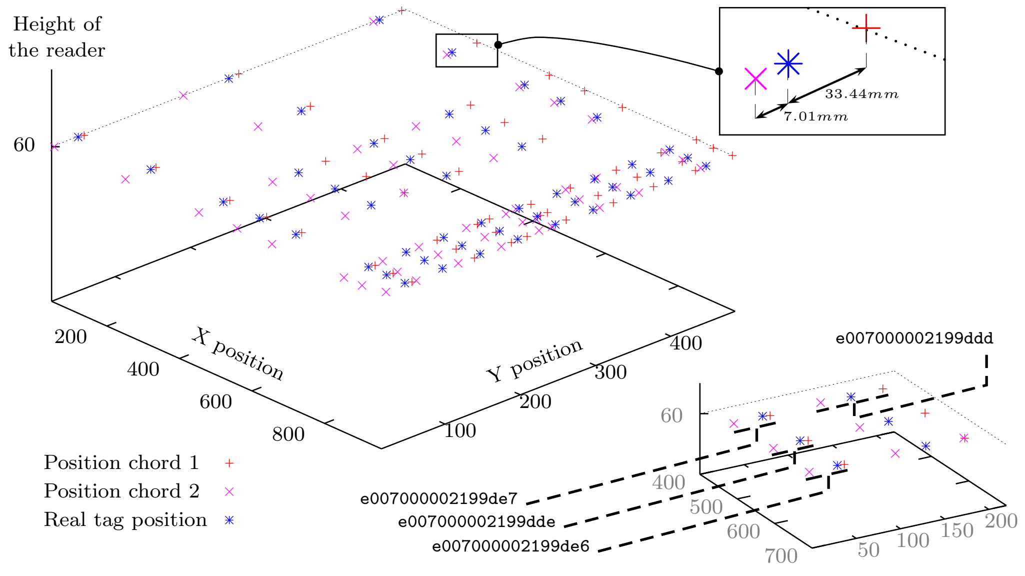
ID_Material	Description	Value
MD06	Textile which is prov...	15°C...

Figure 7. Display of tuples related to Material

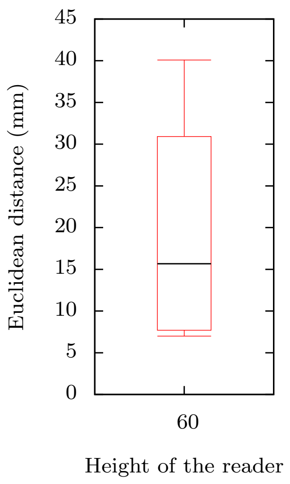
Until this point, data items are retrieved from the communicating textile. To locate them over the textile, we implement the algorithm described in section 3.2. To assess the algorithm precision, the theoretical results will be confronted to the real tag positions. The virtual map given in Figure 8(a) details the real tag position as well as those returned by the algorithm (two possible solutions for one tag). These results have been obtained when the textile is read with a distance of $60mm$ from the ramp of readers[6]. If we look at the detailed case in Figure 8(a), the euclidean distance between the real tag position and those computed by the algorithm is equal to $7.01mm$ for the nearest position (chord 2) and $33.44mm$ for the furthest (chord 1). The focus in Figure 8(a) emphasized where the tags e007000002199de6, e007000002199de7, e007000002199ddd and e007000002199dde (which have been used for storing the 3 data items) are located on the textile.

For each tag, the minimal error is measured between the real tag position and the nearest position among the two computed by the algorithm (euclidean distance). The set of all the computed errors obtained is synthesized in a box-and-whisker diagram in Figure 8 (the reader is positioned $60mm$ above the textile). We can see that the minimal error (i.e. the minimal euclidean distance) is $\simeq 7mm$. 25% of the errors are inferior to $8mm$ (cf. the 1^{st} quartile) and 25% are superior to $31mm$ (cf. the 3^{rd} quartile). In average, the error is about $15mm$ with a reader positioned $60mm$ above the textile, with the conveyor speed equals to $4m/s$ and with an acquisition time

[6] The maximum distance depends on the modeled *reading zone*.



(a) Comparison between real and calculated tag positions



(b) Errors synthesized in form of whisker-box plots

Figure 8. Assessment of the precision of the location algorithm

cycle fixed to $5ms$ (errors depending on these 3 parameters).

5 Conclusion

New challenges and opportunities arise with concepts such as Internet of Things, Ubiquitous Computing and Artificial Intelligence. Nowadays, products are more and more fitted with electronic devices (e.g. sensors, RFID tags) which give them abilities such as data storage, decision making, monitoring. Some authors argued the usage of intelligent products in the framework of the supply chain management. Indeed, these products are able to control their own life, evolution and could serve as an interoperability hub between the supplier members. However, most of the time, products are only given an identifier (e.g. via a RFID tag) which provides a network pointer to a linked database and decision making software agent. As a result, a new kind of material is discussed in this paper referred to as "communicating material" which allows to embed significant proportion of data directly on the manufactured product. In order to answer the question of what information is relevant to store on the product during its lifecycle, a data dissemination process (consisting of three steps) is presented and makes reference to previous works. One important step deals with the storage and retrieval of data on/from the communicating material. In this paper, an appropriate architecture of communication is developed which aims at splitting information over the "communicating material" and at determining where this information is located.

REFERENCES

[1] D. Chan and J.F. Roddick, 'Context-sensitive mobile database summarisation', in *26th Australasian computer science conference*, volume 16, pp. 139–149, (2003).

[2] K. Främling, T. Ala-Risku, M. Kärkkäinen, and J. Holmström, 'Agent-based model for managing composite product information', *Computers in Industry*, **57**(1), 72–81, (2006).

[3] ISO/IEC. Information technology aidc techniques – rfid for item man- agement – air interface, part 1 – generic parameters for air interface communication. ISO/IEC 18000-1, 2004.

[4] M. Kärkkäinen, J. Holmström, K. Främling, and K. Artto, 'Intelligent products–a step towards a more effective project delivery chain', *Computers in Industry*, **50**(2), 141–151, (2003).

[5] S. Kubler, W. Derigent, A. Thomas, and E. Rondeau, 'Prototyping of a communicating textile', in *International Conference on Industrial Engineering and Systems Management*, pp. 1333–1342, (2011).

[6] S. Kubler, W. Derigent, A. Thomas, and E. Rondeau, 'Information dissemination process for context-aware products', in *14th Symposium of Information Control Problems in Manufacturing*, (2012).

[7] S. Kubler, W. Derigent, A. Thomas, and E. Rondeau, 'Key factors for information dissemination on communicating products and fixed databases', in *Service Orientation in Holonic and Multi-Agent Manufacturing Control*, ed., Springer, volume 402, (2012).

[8] D. Ley, 'Ubiquitous computing', *Ubiquitous Computing, emerging technologie*, **2**, 64–79, (2007).

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

[10] G. Morel and B. Grabot, 'Editorial of special issue', *Engineering Applications of Artificial Intelligence*, **16**(4), 271–275, (2003).

[11] MT Ozsu and P. Valduriez, 'Principles of Distributed Database Systems', *Prentice Hall*, (1991).

[12] S. Russell and P. Norvig, 'Artificial intelligence: A modern approach', *Prentice-Hall, Egnlewood Cliffs*, (1995).

[13] M. Schneider and A. Kroner, 'The smart pizza packing: An application of object memories', in *Intelligent Environments, 2008 IET 4th International Conference on*, pp. 1–8, (2008).

[14] H. Sundmaeker, P. Guillemin, P. Friess, and S. Woelfflé, 'Vision and challenges for realising the Internet of Things', *Cluster of European Research Projects on the Internet of Things, European Commision*, (2010).

[15] M. Weiser, 'The computer for the 21st century', *Scientific American*, **265**(3), 94–104, (1991).