

Patterns for Unique Product Identification

Diethelm Bienhaus

Institute of Nanostructure Technologies and Analytics

Technological Electronics

University of Kassel

bienhaus@uni-kassel.de

Introduction

The increasing competition in global markets forces companies to optimise their manufacturing and logistic processes. Tied supply chains, just in time production are means to reduce costs while shortening the needed “time to market”. Customers are attracted not only by the product prize, but properties like quality and safety. In case of higher prized goods the ability to offer individualised products is an competitive advantage. Manufacturers are facing the challenge of cost reduction by mass production with a lot size of one at the same time.

The manufacturing of complex products involves often several companies. Thus product information has to be managed during a product’s whole lifecycle: from design, manufacturing, maintenance to disposal or recycling.

In most applications it is not possible, not desirable or not necessary to store all the product information along with the product itself. Typical reasons are:

- The amount of data is too large for the available storage or representation capacity.
- Not all of the product information should be accessible openly or has to be hidden due to legal restrictions.
- Only a very small portion of the product data is necessary.

To enable process automation machine readable identification labels are attached to products. The contained identifiers allow to look up the associated product information.

Separation of identifiers and associated information raises several challenges, most importantly:

- Product items have to be uniquely and system wide or globally identified among all other product items.
- An infrastructure for lookup mechanisms as well as data storage and management has to be established and maintained.
- Organisational and legal issues of information ownership and responsibility during a product’s lifecycle have to be addressed.
- The appropriate level of uniqueness is changing from a product type focused to a item level identification.

Patterns of this collection expose solutions addressing these tasks. The patterns of this collection and the topic which they address are listed here:

- **AUTOMATIC IDENTIFICATION**
The pattern setting the overall problem domain.
- **DE-CENTRALIZED ID-GENERATION**
A pattern for the problem of decentralized automatic generation of unique identifiers using computers.
- **WRAPPED FORMATS**
Introducing new product identification mechanisms while keeping compatibility to existing standards.
- **COMPOSITE IDENTIFIERS**
Generation of unique identifiers for composite products.

Overview

Fig. 1 illustrates the relationships of the patterns in this collection.

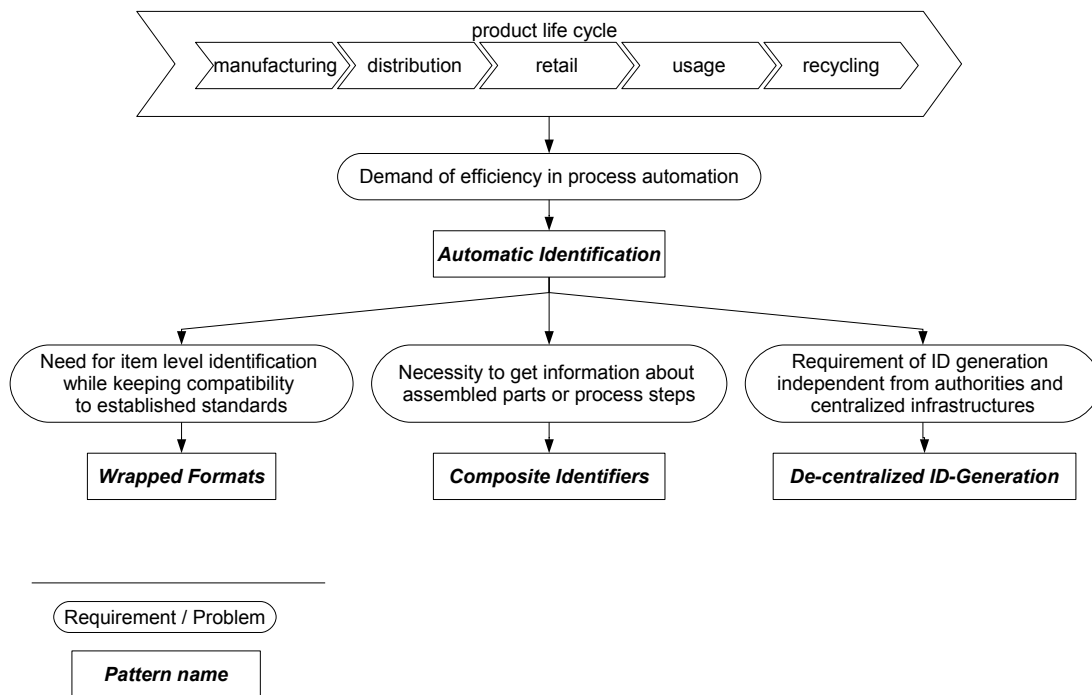


Figure 1: Pattern Relationship Graph

1. Pattern

AUTOMATIC IDENTIFICATION

Alias Names: MACHINE-BASED IDENTIFICATION, MACHINE-READABLE IDENTIFIERS

Context

Automation of processes in manufacturing and logistics. Tracking products during their life-cycle and keeping a link to the associated product information.

Forces

- Identification of products is essential in automated processes. Human beings have the ability to distinct objects due to their recognizable physical attributes. But engagement of human workers is costly.
- Repetitive tasks in a working process tend to result in failures if done by human workers.
- Human workers can identify single product items by recognition of physical properties and comparison based on experience. Failure rates of the correct identification increases if concentration decreases or inexperienced workers perform that task.
- Textual product information on labels attached to the product allow humans the correct identification of single items and acquiring of product data. To give machines access to textual represented data an optical character recognition has to be done first. OCR reading procedures have a certain failure rate influenced by aspects like the printing quality or environmental light conditions.

Problem

Efficient process automation needs product identification without human involvement.

Solution

Attach machine recognizable product data markings on the product itself or the packaging of the product. Choose a machine oriented data representation and communication.

Rationale

Machine readable identifiers and additional data enable cost efficient automation of processes while reading failures can be minimised and boring tasks for humans can be avoided.

While textual characters evolved over a long period of time and humans are trained in school learn to read it this representation is not optimised for machines. To minimise reading failures the signal to noise ratio in the reading process should be as good as possible. Digital codes based on two states provide a good distinction between two states. In case of optical technologies symbols consist of base elements like bars or squares in a colour as a representation of the one state with a high contrast to the colour representing the other state.

Electronically stored data is transferred serially as a bit stream modulated for a wireless or wire-based transmission.

Known Uses

Optically recognizable identification marks

Optical recognizable codes are based on symbologies, that can be read by means of scanners or camera systems. Due to the assembling of the codes a distinction between one- and two-dimensional symbologies can be made.

One-dimensional symbologies consists of sequences of parallel bars and spaces. A defined number of lines represent a character. Widely used is the **European Article Number (EAN)** standard, defined by the standards organisation GS1 (see [GS1]). The EAN-13 barcode consists of 13 numerals and is worldwide used to mark retail goods. Figure 2 shows on the left side an example of a EAN 13 encoding of the numerals “4123456543214” where the first two or three digits represent a country code followed by the manufacturer’s code. The next five digits represent the article code and the last digit is a check number.



Figure 2: EAN 13 encoded barcode (left), DataMatrix code (right)

In case of two-dimensional symbologies patterns of dots are spread in a defined manner on an area. Finder patterns allow reading devices decode the data which can be text or raw any alpha-numeric raw data. Depending on the symbol size up to 2 kilobytes of data can be encoded.

Error correction codes allow correct reading even if the code is partially damaged. Figure 2 illustrates on the right side the encoding of the string “Some Pattern Language on the Challenge of Unique Product Identification” in the DataMatrix format which is popular to mark small items and to print postage stamps or parcel labels.

Radio Frequency based identification Technologies

Radio Frequency Identification (RFID) uses an radio frequency based communication with data storing chips – so called transponders or tags. Transponders contain

- the chip to store data like an identification number or alpha numeric text to describe the product,
- an antenna and controller to exchange data based on a specific communication protocol,
- and in case of active tags an power source.

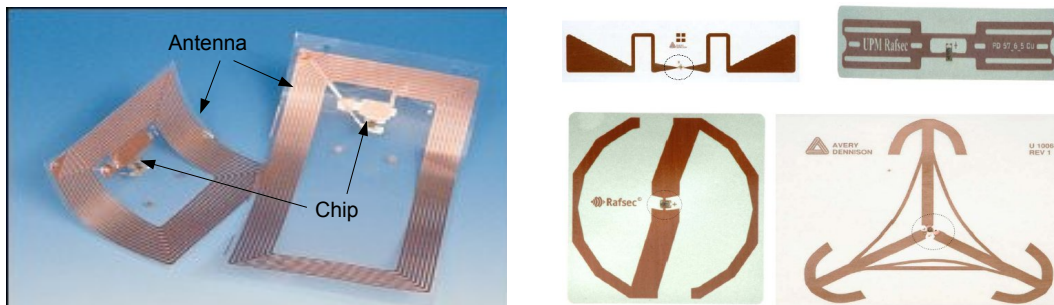


Figure 3: VHF Transponders (Image sources: TI) (left), UHF Transponders (Image sources: Avery, Rafsec) (right)

Depending on the application field, the used frequency and needed power supply a large variation of configurations are produced. Figure 3 shows some examples.

Consequences

As AUTOMATIC IDENTIFICATION enables highly efficient automation of manufacturing and distribution processes there still is the need for a human readable format of the product identifier and additional data. This demand occurs when machine based reading can not be applied e.g. if the necessary devices have broken down or are not in duty because they are to expensive.

Therefore providing access to product data for both machines and humans is good practise:

Bar code labels do not only code a identifier in the machine readable way as a sequence of black and white bars. Additionally the numbers are written below the machine readable code as illustrated in figure 2 on the left side.

In case of RFID transponders labels are available that contains a transponder and can be printed on.

Figure 4 shows an example with two machine reading alternatives – barcode and RFID – and additionally clear text for human workers.

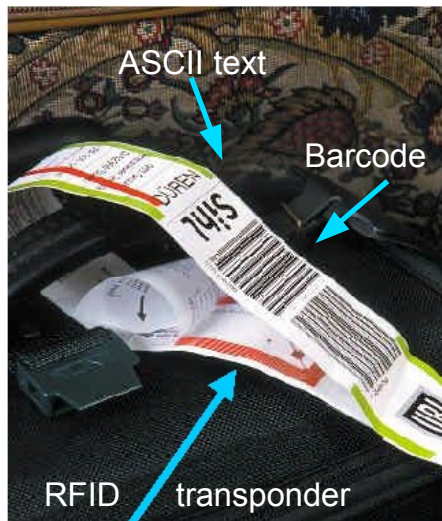


Figure 4: Suitcase Label

Current research of Fraunhofer Institute IZM aims on development of a RFID transponder combined with a bistable display on which clear text can be shown corresponding to the data stored on the RFID chip.

Related Patterns

The need for unique identifiers occurs in object oriented programming languages and database systems as well. Several patterns address such problems.

OBJECT IDENTIFIER [BW98] deals with the problem of preserving an object's identity in a relational database. The proposed solution is, to assign an object identifier to each persistent object, typically a long integer which is guaranteed to be unique for a class of objects. (see [Ris00])

IDENTIFICATION SCHEME [Fow97] introduces the solution to define schemes so each identifier refers to only one unit. Then each identifier refers to only one object while different parties can refer to the object differently. In AutoID applications the problem of dealing with several interested parties that need to identify a single physical object is solved by printing the different ID codes on a single identification plate or using several label that hold one single code.

2. Pattern

DE-CENTRALIZED ID-GENERATION

Alias Names: LOCALLY GENERATED IDENTIFIERS, DECENTRALLY GENERATED IDENTIFIERS

Context

Decentralized automatic generation of unique identifiers using computers.

Forces

- Centralized management of identifiers facilitates generation, assignment and maintenance of unique identifiers. But centralized systems are more error-prone than distributed systems.
- The service of providing and maintaining centralized numbering schemas demands technical and personal resources. Often such services are not free of charge. High fees may exclude users from using unique identifiers who not able or willing to spend money on getting central administered Ids.
- Distributed generation of identifiers bear the risk ambiguousness.
- If centralised creation mechanisms are temporarily not accessible unique identifiers can not be obtained unless they have been delivered and stored in advance.
- Generation and assigning identifiers in advance run the risk of mismatching the real demand of identifiers.

Problem

How can unique identifiers be generated de-centralised that are unique or at least gain uniqueness at maximum probability.

Solution

Use locally available properties for decentral generation of unique identifiers (UID). UID creation mechanisms typically rely on contextual or descriptive properties: current time or name strings. If dependence on such properties has to be avoided mechanisms can use random or pseudo-random number generation.

Rationale

Physical properties like current time or location can be obtained without access to a central allocation infrastructure.

Strings representing an items name and augmented by further data like a description or the item's history are unique with a high probability. Hashing algorithms can generate ID numbers based on such strings.

Without any additional information pseudo-random or truly random number generation is another way to generate numbers that are unique with a certain probability which tends to be less than using in-situ available properties or strings.

Known Uses

[Int05] standardizes the generation, and optional registration, of *Universally Unique Identifiers* (UUIDs). UUIDs consist of 128 bits representing an octet string of 16 octets (128 bits).

The mechanisms specified in [Int05] enable users to generate UUIDs without any registration procedure. The generation mechanisms shall guarantee uniqueness across space and time. While the origin of UUIDs goes back to the Apollo Network Computing System it became later a part of the Open Software Foundation's (OSF) Distributed Computing Environment (DCE), and is now integrated in in Microsoft Windows platforms.

[Int05] defines three algorithms to generate unique UUIDs using mechanisms based on different inherent properties:

- Time-based mechanism: UUIDs can be generated within a single computer system where a 60-bit time-stamp with a granularity of 100 nanoseconds, based on Coordinated Universal Time (UTC) is used as a Time value. To avoid Ids with the same time-stamp generated on different systems, additionally the 48-bit Media Access Control (MAC) address is integrated as a "Node value".
- Name-based mechanism: Cryptographic hashing is used to produce a 128-bit UUID value from a text string which has to be globally unambiguous.
- Random-number based mechanism: A pseudo-random or truly random number generation algorithm is used to produce the bits of the UUID (except of a version value).

Consequences

In case of de-centralised generation of unique identifiers based solely on local attributes the uniqueness can not be guaranteed – sophisticated mechanisms and integration of many specific information can minimise the probability of duplicate IDs.

Uniqueness can be guaranteed by integration of authority based unique identifiers like MAC addresses into the local ID generation mechanisms. Alternative strategies to ensure uniqueness are:

- de-central generation of identifiers together with a central registration mechanism that verifies the IDs uniqueness
- on demand centrally generated Ids

In both cases the advantages of complete decoupling are lost.

Related Pattern

The pattern `UUID FOR EJB` [Mar02] addresses the problem of generation of universally unique primary keys without requiring a database or a singleton.

The solution is based on the combination of enough system information to make the UID unique across space and time. System information in that pattern comprises a system timestamp in milli seconds, IP address, the identity hash code which is internally assigned by a Java virtual machine for distinct objects and a random number generated by the `java.security.SecureRandom` class.

3. Pattern

WRAPPED FORMATS

Context

Identifiers have to be generated compatible to a certain standard while existing formats are wide spread and common.

Forces

- Identification data conform to wide spread standards can be used easily: writing and reading devices as well as processing infrastructures are well established. But standards like EAN 13 cannot cover an identification on the level of individual items.
- Item level identifiers caused the establishing of new standards due to need of larger amounts of numbers. But item level identification is not always necessary. Not every participant of the manufacturing process or in the supply chain has the demand for getting item level IDs or the necessary infrastructure for reading and processing such data.

Problem

How to enable item level product identification without losing compatibility to well established and widely used standards.

Solution

Compose identifiers conform to item level identification standards by wrapping existing formats for type level identifiers.

Encapsulate the ID code representing type level related data and add new properties for the individual product. Delimiters allow to separate the different parts of the contained data fields.

Rationale

Individual identification of physical objects raises the need for a numbering scheme that cannot be covered by standards like EAN 13.

By encapsulation of existing standards item level IDs can be generated while keeping already established numbering mechanisms. If an item level identification is not necessary only the part of the identifier has to be read and processed which refers to the type level.

Known Uses

Uniform Unique Identifiers

Uniform Unique Identifiers (UUID) are represented as a sequence of hexadecimal digits. In order to build a Uniform Resource Name (URN) as specified in IETF RFC 2141 the string “urn:uuid:” has to be the first part of the URN string followed by the UUID.

An example of the string representation of a UUID as a URN looks like:

```
urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6
```

To represent this example conform to UUID integer value format the UUID has to be converted in that format and the prefix “OID” has to be added:

```
urn:oid:2.25.329800735698586629295641978511506172918
```

Identifiers compliant to standards like MAC-48 and EUI-48TM values can be encapsulated in larger identifiers that work as containers. To do so algorithms are specified defining the integration of the smaller ID code while avoiding duplicated or conflicting values (details are provided in [LMS05]).

Electronic Product Code

The Electronic Product Code (EPC) developed by the MIT AutoID Center is an companies with globally unique identifiers. The coding scheme extends the EAN coding which is used in barcodes by adding a unique serial number to each product code in addition to the already defined version number, manufacturer and product type found in the EAN specification ([EPC04]).

4. Pattern

COMPOSITE IDENTIFIERS

Context

Assigning single unique identifiers to products which are assembled of several parts or to products which have changed significantly.

Forces

- A single and unique identifier is needed to lookup and assign product data to a product item. But in case of assembled products several individual parts form together a new part that has to be assigned a new unique identifier. Nevertheless the identity of the building parts may still be important - even after the assemblage.
- Steps during manufacturing, assembling and distribution processes constitute a product's history. A type level identifier cannot resolve individuals. If items get their own identification number individuals can be resolved but changes have still to be maintained separately. The accuracy of data that belongs to an individual item is difficult to guarantee during a product's life cycle since mostly different manufacturers and distributors are involved.

Problem

When composite products are identified using newly generated identifiers information about contained parts or performed process steps cannot be resolved from that ID. How to generate identifiers for composite products such that information about contained parts or performed process steps can be gained from the new identifier.

Solution

Compose identifiers by aggregating identification data of the contained parts. To document a product history data can be assembled which is related to the process steps the product has passed through.

Rationale

Important production steps of a single item or during the assemblage can be associated with specific data. To assign the product data of a completed product to its history the identification data of the product should be a composition of the identifiers of the aggregated parts or production steps.

Consequences

Aggregated Attributes contain information about a product's assemblage and process history in the identifier itself. Hence the identifier has to be able to store or represent the accumulated data. RFID transponder can store about several kilobyte of digital values. Standardised two dimensional symbologies like the DataMatrix code can represent up to 3 kilobyte alpha numerical digits.

Nevertheless, a compact coding is necessary due to storage capacities and reading speed. If one of them is not sufficient or the disclosure of a product's details are not intended or allowed then that data has to be stored apart.

Related Pattern

COMPOSITE [GHJV96] deals with the problem of the composition of objects into tree structures in order to represent part-whole hierarchies. The solution enables clients to treat individual objects and hierarchical compositions of objects uniformly.

In [FARKH07] the adoption of (software) object references for the handling of product item information (hence "hardware" i.e. physical objects) is discussed. It is proposed to apply the COMPOSITE and VISTOR patterns to represent information about subassemblies on a database.

The identifiers then work as a key to get access to the full product information which is stored on the database or to get a link to the product description which is accesible on the Internet. The latter case is introduced as a VIRTUAL COUNTERPART in [Bie05].

References

- [Bie05] Diethelm Bienhaus. A Pattern Language for the Network of Things. In *Proceedings of 10th European Conference on Pattern Languages of Programs (EuroPlop 2005)*, Irsee, Germany, 2005.
- [BW98] K. Brown and B. Whitenack. Crossing Chasms: A Pattern Language for Object-RDBMS Integration. In J. M. Vlissides, J. O. Coplien, and N. L. Kerth, editors, *Pattern Languages of Program Design*, Reading, MA, 1998. Addison-Wesley.
- [EPC04] EPC Global Inc. EPC Tag Data Standard Version 1.1 rev 1.26. Technical report, EPC Global Inc., 2004. Online available: http://www.epcglobalinc.org/standards_technology/specifications.html.
- [FARKH07] Kary Främling, Timo Ala-Risku, Mikko Kärkkäinen, and Jan Holmström. Design patterns for managing product life cycle information. *Commun. ACM*, 50(6):75–79, 2007.
- [Fow97] Martin Fowler. *Analysis Patterns*. Addison-Wesley, 1997.
- [GHJV96] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison Wesley, Reading, MA, 1996.
- [GS1] GS1. EAN Specification. Available on the internet: www.gs1.org.
- [Int05] Internet Engineering Task Force. (IETF), Int. Standard X.667: "Information technology - Open Systems Interconnection - Procedures for the operation of OSI Registration Authorities: Generation and registration of Universally Unique Identifiers (UUIDs) and their use as ASN.1 Object Identifier components". Technical report, ITU-T Rec. X.667 and ISO/IEC 9834-8:2005, 2005.
- [LMS05] Paul J. Leach, Michael Mealling, and Richard Salz. A Universally Unique Identifier (UUID) URN Namespace, Internet proposed standard RFC 4122. Technical report, Internet Engineering Task Force, July 2005.
- [Mar02] Floyd Marinescu. *EJB Design Patterns: Advanced Patterns, Processes, and Idioms with Poster*. John Wiley & Sons, Inc., New York, NY, USA, 2002.
- [Ris00] Linda Rising. *The Pattern Almanac 2000*. Addison-Wesley Longman, Amsterdam, 2000.