Meta-data alignment in open Tracking & Tracing systems

Fred van Blommestein[1], Dávid Karnok[2], Zsolt Kemény[2]
[1]University of Groningen, [2]MTA SZTAKI Budapest

Table of contents

Summary

In Tracking and Tracing systems, attributes of objects (such as location, time, status and temperature) are recorded as these objects move through a supply chain. In closed, dedicated systems, the attributes to record and store are determined at design time. However, in open Tracking and Tracing systems, the attributes are not known beforehand, as the type of objects and the set of stakeholders may evolve over time.

Many supply chains require open Tracking and Tracing systems. The participants in the supply chain are individual companies, spread over many countries. Their trading relations change constantly. Usually they participate in multiple supply chains. E.g., a company producing chemicals may serve the chemical industry, the food industry and the textile industry at the same time. Transport companies carry goods for multiple industry sectors. Yet, they play a role in the traceability of all goods they produce or carry.

Open tracking and Tracing systems are not dedicated for a certain type of product or object nor for a specific industry sector. They simply record the location, time and other attributes of the identified objects, and store that information in the data store of the object owner, based on the identification (e.g. RFID) tag. What attributes are to be stored is determined by stakeholders, such as (end) users of the object. In some cases (e.g. food) legislation prescribes what to record. An open Tracking and Tracing system therefore needs to be able to dynamically handle the set of attributes to be recorded and stored.

In this chapter, a method is presented that enables components of Tracking and Tracing systems to negotiate at run time what attributes may be stored for a particular object type. Components may include scanning equipment, data stores and query clients. Attributes may be of any data type, including time, location, status, temperature and ownership. Apart from simple attributes, associations between objects may be recorded and stored, e.g. when an object is packed in another object, loaded in a truck or container or assembled to be a new object.

The method makes use of findings in ontology engineering and of type theory. New types are based on existing types, with some restrictions.
Both the range of values of a type and its meta-attributes (such as cardinality) may be restricted to define a new type. Programmatically, concepts of co- and contra variance are used to make the method implementable.

The method was developed in two European funded research projects: TraSer and ADVANCE. In TraSer, a truly open and extensible Tracking and Tracing system was developed (TraSer project consortium, 2006; Monostori et al., 2009). In ADVANCE, a distributed management information system for logistics operations was designed and implemented, that makes use of Tracking and Tracing information (ADVANCE project consortium, 2010; Kemény et al., 2011a).

1 Open Tracking & Tracing

In the present business landscape, companies should not be considered to be independent entities, but parts of supply chains that are interwoven to multi-echelon networks. Material flow transparency, specifically the visibility to inventories and deliveries in the whole supply network, is considered an imperative requirement for successful supply-chain management, and has been associated with significant efficiency and quality improvements (Ala-Risku and Kärkkäinen, 2004; Ballard, 1996; Clarke, 1998; Främling et al., 2004; Kärkkäinen et al., 2004).

Apart from logistics, transparency of the origin of goods and their manufacturing conditions is increasingly required by consumers and regulators. Food safety and food composition are hot topics. Consumers wish to be informed under what conditions products were produced and packed, and use criteria such as sustainability, animal well-being, fair trade and worker's conditions to guide their purchasing (Hiscox and Smyth, 2011).

Manufacturers, on the other hand, need to know where their products are ultimately sold and consumed. They are increasingly held responsible for maintenance, spare parts supply and reverse logistics (Kosk, 2014). In case of defects or quality issues, consumers may need to be warned and products may need to be recalled. Without a system that keeps track of product destinations, too many products must be called back and too many consumers are alarmed unnecessary.

Tracking products and electronic product representations across enterprise boundaries currently requires substantial manual work, or extensive system-to-system integration work. From an application point of view, tracking functionality is tightly coupled to the systems and practices of the individual supply chain participants, resulting in network level operational processes being rare and expensive. Few companies, regardless of their desire for supply-chain efficiency, have implemented supply-chain transparency solutions (Kaplan, 1998; Gunasekaran and Ngai, 2004). Even fewer have developed solutions for transparent product customisation, delivery or the networks involved in maintenance and repair.

An example in the food industry may illustrate the problem area. Initially, a tracking and tracing system is used to track pallets through the supply chain, from distribution centres to outlets. Each pallet is identified, and is by means of scanning linked to a purchase order, a picking event, a loading event and a receiving event. Supply chain partners upload the scan data to a (centralised or distributed) database that is accessible to all of them. A few years later, the temperature of the pallets is to be controlled by the same tracking and tracing system. The data to be processed after each event is extended with the temperature at the time of scanning. Yet later, the system is to track the products that are stacked on the pallets, their source and their best before date. So an event is added: the stack event. Then at the time of receiving in the outlets, the outlet inventory system is to be updated after scanning the pallet. Again an extension of the data to be processed, both by uploading and by querying systems, is needed. Note that the roll-out of the extra functions may be stretched over a lengthy period, one outlet at the time. During roll-out, systems may need to support multiple versions.

Most present tracking and tracing (T&T) systems are closed. Their use is limited to the supply chain partners of one product brand or function (e.g., transport). Companies that serve multiple brands need to install multiple systems. Systems, dedicated for some function (e.g., transport, anti-theft or product quality) are seldom interconnected.

EPC Global (2014) is a relative open system. Every company may join if they adhere to the conditions and pay a fee. The types of events that may be recorded in the system and the types of data that is stored are extensible. However, extensions should be approved by the EPC Global standardisation committees. Private extensions are allowed, but no mechanism exists to inform supply chain partners of such

extensions. In any case, implementation of extensions requires reprogramming of all systems in the affected supply chain.

Note that not only products need to be tracked, but also the equipment that is used to produce and transport the products and the people that are responsible for that (Ilie-Zudor et al., 2011). Events may cause very complex transactions in T&T systems.

As supply chains are interwoven, solutions that are tailored for a specific product or brand are not viable. The same type of product may be sold to factories as components for other products, to wholesalers and directly to ultimate consumers via a web shop. Transport companies may carry goods of various industry sectors.

Trading relationships are not cast in concrete, they are volatile. Smaller companies, such as retailers, may not be expected to modify their IT system to accommodate the tracking and tracing solution of each new supplier.

The only feasible system that may support the above mentioned challenges is an open tracking and tracing system. In the next section requirements of such a system are listed. In the remainder of this chapter, the main requirement: the ability for the system to extend the information model of product types with new attributes and associations is elaborated.

2 Requirements for open T&T

From the observations in section 1 we can formulate the following requirements for an open Tracking & Tracing system:

1. Any party must be able to use it
   An Open T&T system should be open. The system components and the services that maintain them should allow interconnection with components of any other user. Of course security should be guaranteed.

2. It must be possible to track any product type
   It cannot be foreseen what T&T requirements will develop in the future and how supply chains will interconnect. So any product type or better, any object type should be traceable with the system.

3. The types of information, stored for a product type must be extensible
   As it cannot be foreseen what products will be tracked, it certainly cannot be foreseen what information of the products or objects should be traced. So the object information model must be extensible 'on the fly', while the system is being used.

4. Owners must be able to disclose information to selected receivers
   Supply chain information may be sensitive and is sometimes even strategic for companies. They must be able to shield this information from, e.g., competitors.

5. Scalability: the system must be distributed.
   The value of an open T&T system increases with its size and scope. It may need to track billions of objects. On that scale it is not feasible to organise it around a central component. So the system should be truly distributed.

3 Open T&T architecture

An open T&T system consists of the following components:

- Objects that carry a unique identification number
- Reading devices that can read Object Identifications
- Upload Clients that allow users to register Object related observations with or without reading devices
- Servers that store Object related observations
- Query Clients that may query or subscribe to Object related information

Objects may be tangible (a box) or intangible (a purchase order). Objects are of a certain type. Objects have properties. The Object type prescribes the kind of properties (Property Types) an object may have. Properties are directed, tagged associations with other objects or with data types. For example Objects that are of Object Type "Box" may have dimensions and weight as attributes and may have a "stacked on" association with an object that is of Object Type "Pallet".

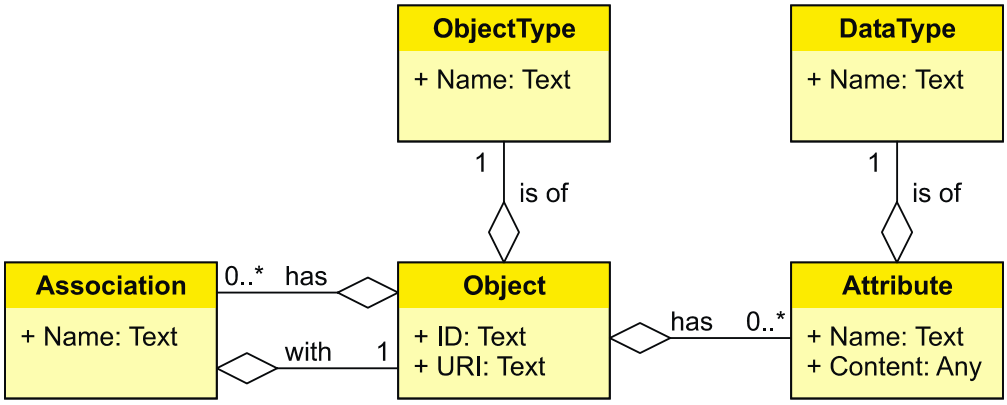

Figure 1. Data model

Objects have a world-wide unique identification number as a special property. The identification of an Item consists of a URI and a number, unique within the scope of that URI (Främling, 2002). Identifications may be affixed to the object in the form of a machine readable label: a bar code, an RFID tag or otherwise. For many (most) objects however, the ID-URI combination is virtual. It only exists in information systems. Only the objects that have to be tracked have an automatic readable tag or label. Objects, such as locations, companies, persons, production equipment and orders, that need to be associated with the objects-to-be-tracked are known to the information system that uploads tracking information to the open T&T system.

The function of the URI is twofold. Through the Internet Domain Name Service it guarantees uniqueness of the identification number, provided that the owner of the URI does not assign duplicate numbers. It also defines the Internet address where information on the Object can be obtained and where information may be uploaded.

Note that many existing T&T systems do not provide a URI as part of the Object identification. These systems (e.g., GS1 EPC Global, 2014) use other methods to guarantee uniqueness of the identification numbers. Such systems still can be part of an open T&T system if they provide the Internet addresses where information may be uploaded and obtained in some other way. The URI is then implicitly known, if the Object identification can be identified as being part of the specific existing system.

In open T&T systems, all objects carry an ID (and a URI), also locations, companies and production lines that are needed as part of the tracking information on products. The owner of the information on these non-product objects may be different from the product owner. The information may be retrievable from different URI's than the URI that is part of the product identification.

For the open T&T system to function, the (explicit or implicit) URI must be resolvable and must support defined protocols to store and retrieve Object information. Specification of these technical (Web Service) protocols is beyond the scope of this chapter. Here we focus on the content of the messages to be exchanged between nodes and servers.

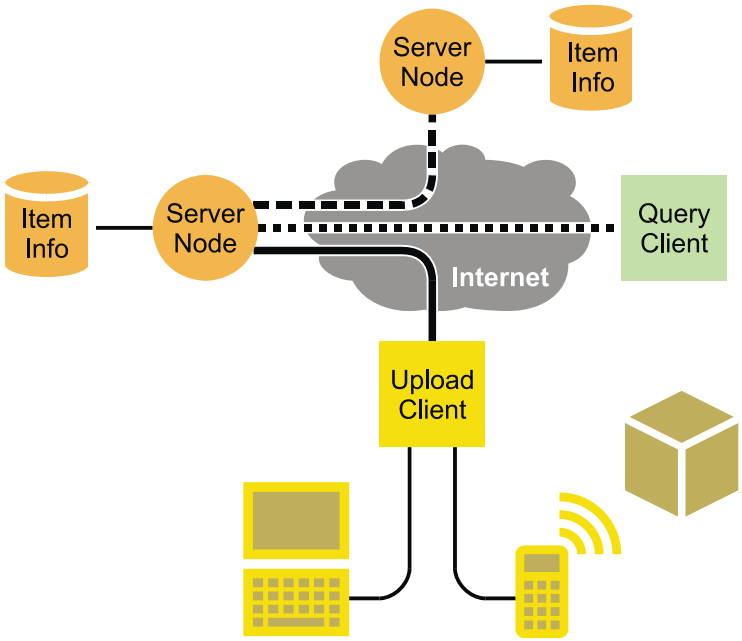

Figure 2. Open T&T configuration

Reading devices read the identification of labels or tags of Items. They may also read or add other information and transmit that to the Client that is to upload the recorded information as an observation. For instance, the RFID chip that is affixed to the object may in addition to the ID and URI contain temperature information. The reading device or the upload client may add a timestamp and location information, plus the information that the object was unpacked from a box (with another ID and URI). The observation that is recorded and will be uploaded to the Server node then contains {ID, URI, temperature, timestamp, location, {unpacked from: ID, URI}}.

The upload client agrees with the server node which information to upload for an object type. The upload client needs to know of which type an object is. The upload client therefore first queries the server client. If it is the first time the upload client handles an object of this type, it then queries the server node what information the server node expects for the specific event. The upload client then uploads the information by means of an observation message (see figure 3).

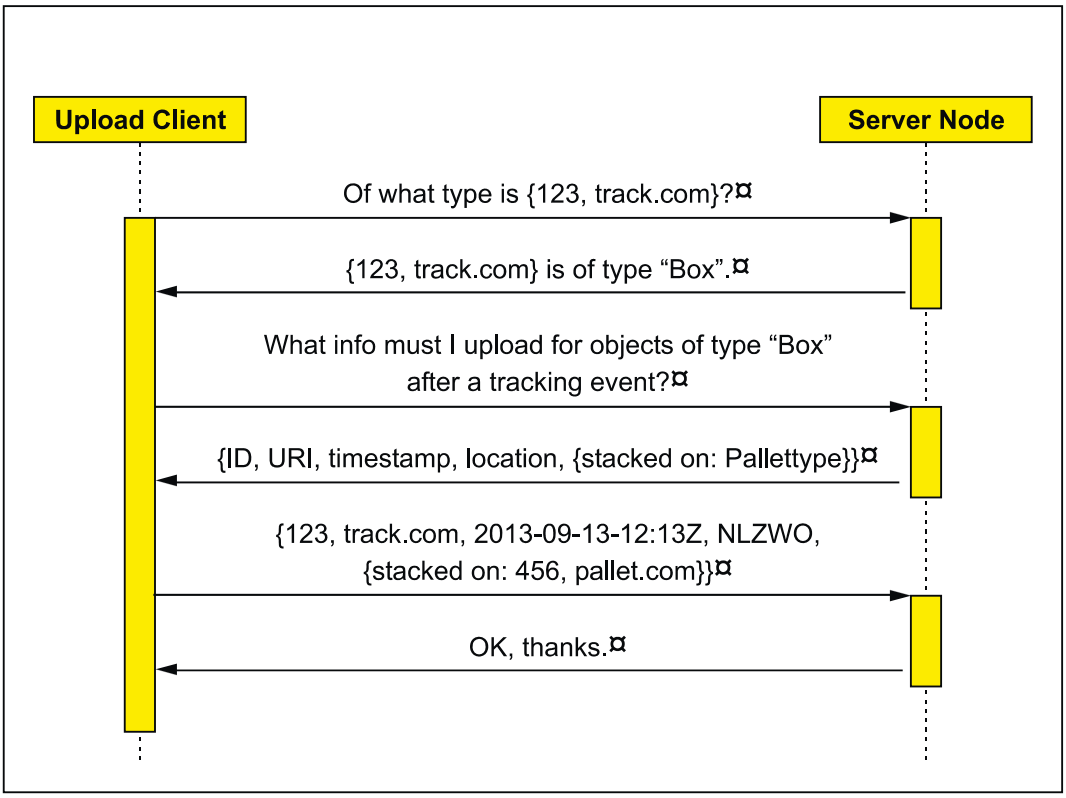

Figure 3. Conversation between upload clients and server nodes

The server node then stores the uploaded information. That information may be retrieved by Query clients, or it may actively be distributed to Clients that have subscribed to the information. Events may affect other Items as well. Based on rules to be stored on the Server, events may be created and uploaded to the applicable servers. In the above example, for instance, the observation may be forwarded to the server node that services information on the pallet ("pallet.com").

Query Clients may retrieve Item related information from Servers by querying them or by subscribing on periodic or event triggered reports. Server nodes should always provide information that has an ID@URI as key to query clients that are entitled to receive the information. Free queries and selection of items by other properties can optionally be serviced.

4 Meta-data alignment

In requirement 3 in section 2 is stated that the object information model must be extensible 'on the fly', while the system is being used. This is not the case in present T&T systems. In the EPC Global (2014) system, for example, data sets to be processed are extensible, but when they are extended, message schemas and application interfaces change. So systems must be reprogrammed or at least reconfigured. This is undesirable in many situations. E.g., in the example in section 1, some outlets may be less advanced than other outlets. An update of the T&T system should not force outlets to change simultaneously.

In open T&T systems, extensions should not be managed centrally (and certainly not by standardisation committees), but bilaterally, peer to peer.

For extension and alignment of meta-data in an open T&T system, we therefore use the method, described in Blommestein (2014). The method is based on ontology engineering (Sowa, 2000). In ontology engineering, both individual objects (instances) and types or classes are treated in a similar way and may be mixed in reasoning. They are both organised in triples: Object (type) - Property (type) - Object (type). This way of organising knowledge is derived from the structure of natural languages, in which a subject and an object (both objects or object types) are glued together by a verb and possible adverbs (together constituting the property or property type).

The meta-data in the open T&T system consists of:

- Data types of object properties
- Object property types
- Object types
- Event types

Both query and upload clients may interrogate server nodes which property-, object- and event types the server node supports. If some type is not supported, but required, the server node may be requested to add the type to its meta data scheme.

As meta-data is to be manipulated, it is stored as normal data. E.g. names of object and event types are ordinary properties (see figure 1).

Open T&T clients and server nodes may, apart from observations, queries and responses, exchange the following messages to align meta data:

- *Definitions* of new object types and of new data types. Definition of new object types is elaborated in section 5, definition of new data types is described in section 6.
- *Extensions* of object types with new properties (attributes and associations). Extensions are described in section 6.
- *Observation types*, the structure of the information to be uploaded, depending on object type and event type. Definition of new observation types is described in section 7.

5 Definition of new object types

New object types may be defined by genus and difference. An object type inherits its definition from a more general object type (the genus) and adds a set of constraints to the properties of the general object type to differentiate itself from other specializations or subtypes of the more general object type. Objects that are instances of the newly defined object type are also instances of the more general object type. The ‘differences’ are defined by constraining the properties. Properties can only be constrained if they have been defined. In order to define a specialisation therefore the properties that are to be constrained must have been defined on the level of the more general object type.

New object types are defined based on existing object types, so the basic semantics of object types and properties are already known. Those semantics are refined by further constraining the properties of the object type members, in other words, the 'membership' criteria are made stricter.

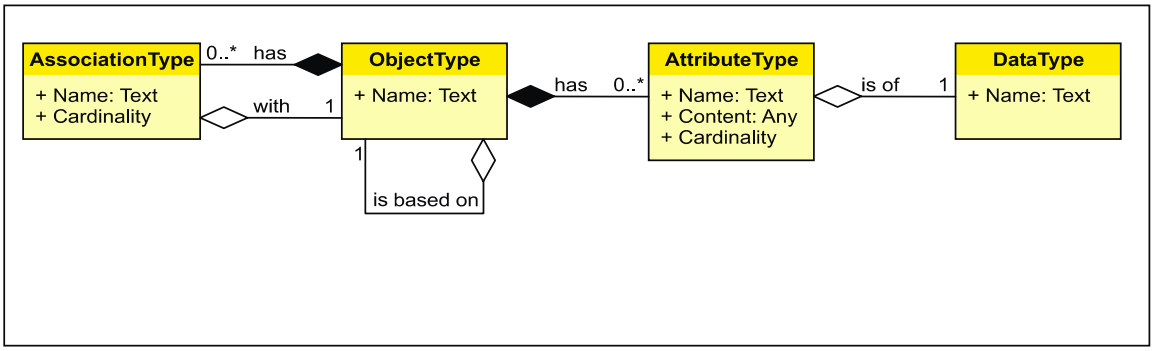

Figure 4. Object Type

As an illustration, we take the example of a pallet. A pallet is defined as "Transport Equipment, consisting of a portable platform on which goods can be moved, stacked, and stored, with the aid of a forklift". Inspection of this definition reveals that the genus of a pallet is "Transport Equipment". The difference consists of three elements:

- Shape (a portable platform)
- Function (on which goods can be moved, stacked, and stored)
- Usage (with the aid of a forklift).

This means that generic Transport Equipment must also have a shape, function and usage. The elements are narrowed to define a pallet as transport equipment. E.g., where generic transport equipment may also be shaped as a box (e.g. a container), a pallet is shaped as a platform.

The Definition pattern is illustrated by means of the following XML snippet:

```
<Definition>
        <ObjectType Name="Pallet">
                <BasedOnObjectType Name="Transport Equipment"/>
                <Attribute Name="Shape" Datatype="ShapeCode" Value="Platform"/>
                <Association Name="Move" ObjectType="Goods"/>
                <Association Name="Stack" ObjectType="Goods"/>
                <Association Name="Store" ObjectType="Goods"/>
                <Association Name="Use" ObjectType="Forklift">
        </ObjectType>
</Definition>
```

The definition by means of genus and difference, structured in properties, is not strictly needed for Server nodes to be able to store the observations uploaded by client nodes. The data model in figure 1 allows storing observations on objects of arbitrary object types and with arbitrary properties. The information is however ultimately to be used by business information systems, such as ERP systems. Such systems usually are less tolerant with regard to the data model. The definition of object types as specialisation of existing object types allows such a system to store the information on a pallet as information on transport equipment.

The definition does not specify all properties, only the properties that are needed to recognise a pallet among all transport equipment. The additional properties that may be used in observations are specified by means of Extensions. The following XML snippet illustrates an Extension:

```
<Extension>
```

```
        <ObjectType Name="Pallet">
                <BasedOnObjectType Name="Transport Equipment"/>
                <Attribute Name="Length" DataType="LengthMeasure"/>
                <Attribute Name="Width" DataType="LengthMeasure"/>
                <Attribute Name="MaxLoad" DataType="WeightMeasure"/>
                <Attribute Name="Location" DataType="UNLoCode"/>
                <Association name="Load" ObjectType="Vehicle"/>
                <Association name="Unload" ObjectType="Vehicle"/>
                <Association name="Place" ObjectType="Warehouse_Location"/>
        </ObjectType>
</Extension>
```

Definitions and extensions, accepted or amended by a server node need to be propagated to other server nodes, to prevent differently defined object types with the same name.

The properties, defined in Definitions and Extensions are available for Observations that are uploaded by Upload clients.

6 Definition of attributes

An object type can be referred to by a name and is represented by a set of properties. Properties are directed, tagged associations with other objects or with data types. Objects are identified by a unique identifier. Data type instances, however, are identified by their values.

For example, a numeric data type has a number as the value of an instance. A number represents a point on the mathematical numeric scale. A textual data type has as instance values strings of characters of some alphabet. Usually such text has some meaning in a natural language.

In this section the structure and representation of data types and data type values are inspected.

The scales on which data type values are projected may be abstract mathematical scales (such as the numeric scale) but may also be physical scales, such as the scale of geographical locations or the time scale. Data type scales need not to be one-dimensional and the different dimensions of a scale may have very different semantics. For example a ‘measurement’ may consist of a (one-dimensional) value and a measure unit that is defined on the ‘measure unit scale’ (e.g. the SI system).

It is possible to define subsets on scales. This is done by means of ‘facets’. Facets may limit the length of the scale, the precision or may define specific value patterns.

Data typing includes the mapping of semantic units or ontological constructs to sign systems. To enable the processing of signs by computers, signs and sign constructs are encoded in binary systems. In fact binary systems are a special kind of sign systems, using bits as signs. Other sign systems include printed text, icons, sounds, etc.

Names and values might be directly represented in bit patterns. However, different computer languages, operating systems and storage technologies use different bit representations for the same functional content. In order to be technology-independent and to be able to specify T&T communication between computers that use different languages and operating systems, the data type system to use for T&T should be layered. In the higher layers of the stack, semantic structures can be specified, while in the lower layers, mappings can be realized to character and bit representation. For open T&T we propose to use the XML language for this representation.

The Core Component Technical Specification (CCTS) (UN/ECE, 2003) offers a language to describe data models and messages in a technology- and syntax-neutral way. CCTS 3.0 makes a distinction between Core Data Types (CDTs) and Business Data Types (BDTs). BDTs specialise CDTs: the domain value of a CDT is restricted for a BDT. CDTs (and therefore also BDTs) have a not-too-complex internal structure, but they are not scalars. In the sequel of this section the CCTS data type system is taken as the basis for open T&T communication.

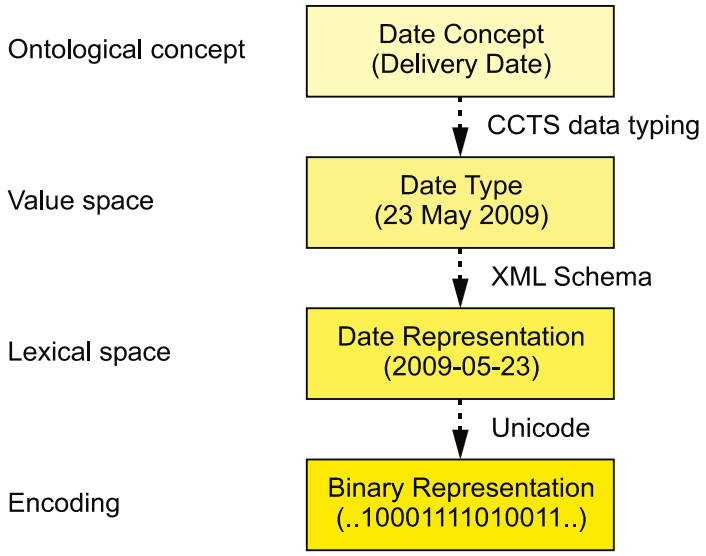

Figure 5. Encoding of information

A more generic data type system is described in the XML Schema specification (W3C, 2004). The XML schema specification makes a distinction between the value space of a data type and the lexical space of a data type. The value space denotes the semantic scale of the represented information, such as Date Time, Numeric, or Text. The lexical space defines the way in which values in that dimension are represented in Unicode (for XML messaging).

CCTS data types are not one-dimensional scalars. CCTS defines data types consisting of a Content Components and one or more Supplementary Components that further specify the semantics of the content. E.g. an Amount has a number as content and a currency code as Supplementary Component to specify the currency. This mechanism resembles closely the “Semantic Values” as proposed by Sciore (1994) and Lee (2000).

Data types are defined bottom-up. At the bottom a number of (pre-standardized) scales are defined, such as the set of numbers, the set of texts and the set of date-time combinations. By constraining these scales by means of facets and by combining them, the specific data types are being defined.

Facets are constraints that are specific to the scale. Some scales are ordered (such as the numeric and the date-time scale); others are not (such as the text scale). Only for ordered scales can minimum and maximum values be specified. Precision may be specified for numerics in a straightforward way, for dates and times precision is fairly complex.

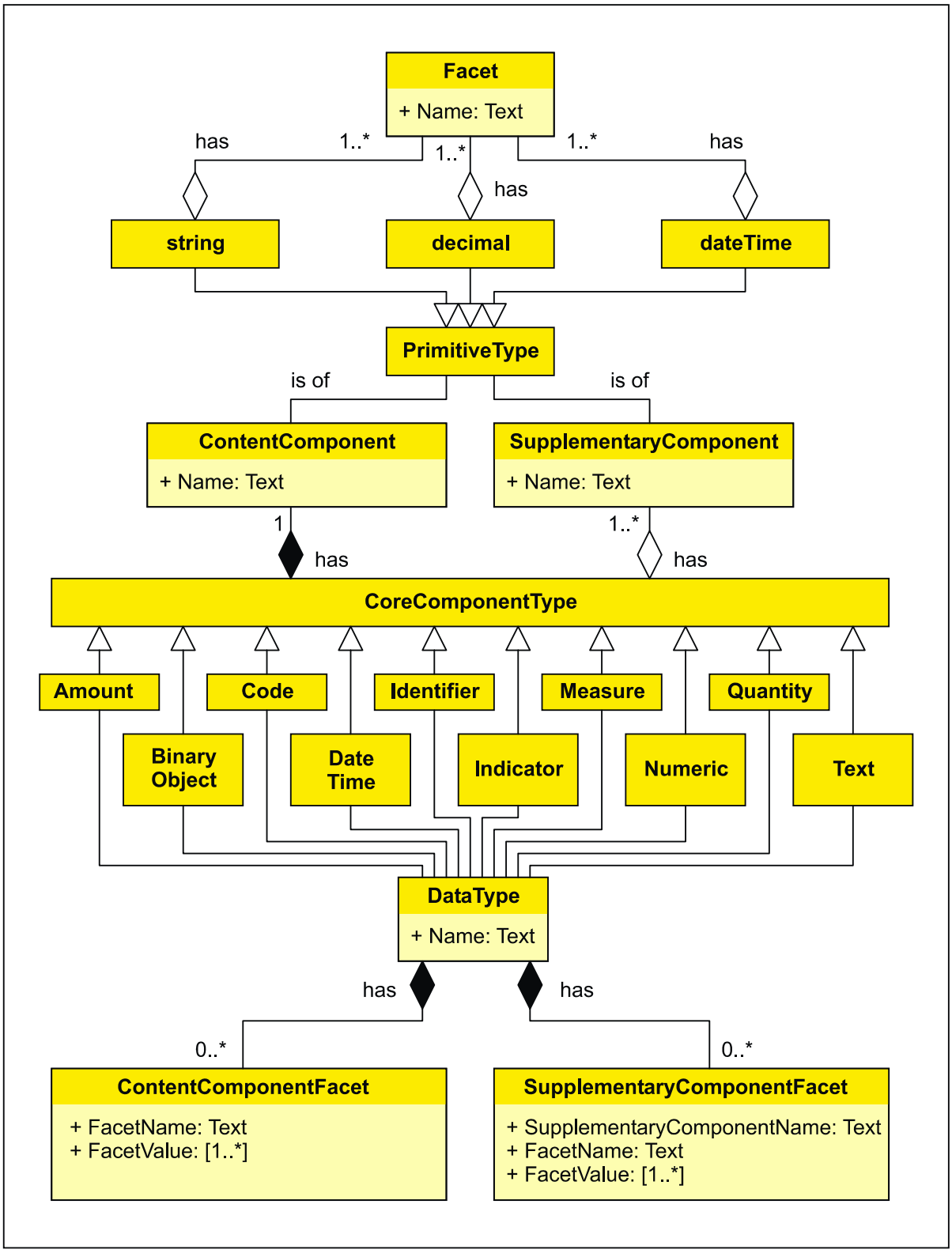

Figure 6. Data-type meta-model

The basic scales that are needed for the CCTS v.2.01 data types are the textual scale, numeric scale and the date-time scale. These scales map (not coincidental) neatly to the data type system of XML Schema. From this system, only dateTime, string and decimal are used for open T&T systems.

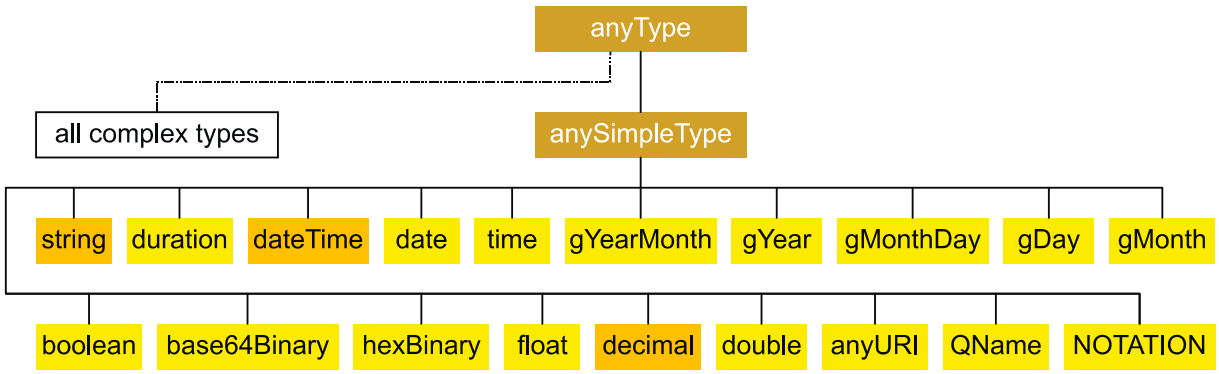

Figure 7. XML data type system

The basic scales are called “Primitive types” in CCTS.

Primitive Type	Format Restrictions or Facets	Definition
String	Expression	Defines the set of characters that can be used at a particular position in a string.
	Length	Defines the required length of the string.
	Minimum Length	Defines the minimum length of the string.
	Maximum Length	Defines the maximum length of the string.
	Enumeration	Defines the exhaustive list of allowed values.
Decimal	Total Digits	Defines the maximum number of digits to be used.
	Fractional Digits	Defines the maximum number of fractional digits to be used.
	Minimum Inclusive	Defines the lower limit of the range of allowed values. The lower limit is also an allowed value.
	Maximum Inclusive	Defines the upper limit of the range of allowed values. The upper limit is also an allowed value.
	Minimum Exclusive	Defines the lower limit of the range of allowed values. The lower limit is no allowed value.
	Maximum Exclusive	Defines the upper limit of the range of allowed values. The upper limit is no allowed value.
Date-Time	Minimum Inclusive	Defines the lower limit of the range of allowed dates. The lower limit is also an allowed date.
	Maximum Inclusive	Defines the upper limit of the range of allowed dates. The upper limit is also an allowed date.
	Minimum Exclusive	Defines the lower limit of the range of allowed dates. The lower limit is no allowed date.
	Maximum Exclusive	Defines the upper limit of the range of allowed dates. The upper limit is no allowed date.

Figure 8.Primitive types (from UN/ECE, 2003)

Based on the three Primitive Types ten Core Data Types are defined in CCTS.

CCT Dictionary Entry Name	Definition	Content and Supplementary Components
Amount. Type	A number of monetary units specified in a currency where the unit of currency is explicit or implied.	Amount. Content (Decimal) Amount Currency. Identifier (String) Amount Currency. Code List Version. Identifier (String)
Binary Object. Type	A set of finite-length sequences of binary octets.	Binary Object. Content (String) Binary Object. Format. Text (String) Binary Object. Mime. Code (String) Binary Object. Encoding. Code (String) Binary Object. Character Set. Code (String) Binary Object. Uniform Resource. Identifier (String) Binary Object. Filename. Text (String)

CCT Dictionary Entry Name	Definition	Content and Supplementary Components
Code. Type	A character string (letters, figures or symbols) that for brevity and/or language independence may be used to represent or replace a definitive value or text of an *Attribute* together with relevant supplementary information.	Code. Content (String) Code List. Identifier (String) Code List. Agency. Identifier (String) Code List. Agency Name. Text (String) Code List. Name. Text (String) Code List. Version. Identifier (String) Code. Name. Text (String) Language. Identifier (String) Code List. Uniform Resource. Identifier (String) Code List Scheme. Uniform Resource. Identifier (String)
Date Time. Type	A particular point in the progression of time together with relevant supplementary information.	Date Time. Content (Date-Time) Date Time. Format. Text (String)
Identifier. Type	A character string to identify and distinguish uniquely, one instance of an object in an identification scheme from all other objects in the same scheme together with relevant supplementary information.	Identifier. Content (String) Identification Scheme. Identifier (String) Identification Scheme. Name. Text (String) Identification Scheme Agency. Identifier (String) Identification Scheme. Agency Name. Text (String) Identification Scheme. Version. Identifier (String) Identification Scheme Data. Uniform Resource. Identifier (String) Identification Scheme. Uniform Resource. Identifier (String)
Indicator. Type	A list of two mutually exclusive Boolean values that express the only possible states of a *Property*.	Indicator. Content (String) Indicator. Format. Text (String)
Measure. Type	A numeric value determined by measuring an object along with the specified unit of measure.	Measure. Content (Decimal) Measure Unit. Code (String) Measure Unit. Code List Version. Identifier (String)
Numeric. Type	Numeric information that is assigned or is determined by calculation, counting, or sequencing. It does not require a unit of quantity or unit of measure.	Numeric. Content (Decimal) Numeric. Format. Text (String)
Quantity. Type	A counted number of non-monetary units possibly including fractions.	Quantity. Content (Decimal) Quantity. Unit. Code (String) Quantity Unit. Code List. Identifier (String) Quantity Unit. Code List Agency. Identifier (String) Quantity Unit. Code List Agency Name. Text (String)
Text. Type	A character string (i.e. a finite set of characters) generally in the form of words of a language.	Text. Content (String) Language. Identifier (String) Language. Locale. Identifier (String)

Figure 9.Core data types (from UN/ECE, 2003)

In fact many more scales (Core Data Types or even Primitive Data Types) could be defined, such as a colour scale, a location scale, a taste scale, etc. For e.g. locations, many different representations exist (postal, geographical, official, etc.). Each of them has a different structure (number-pairs, codes, text blocks). It seems therefore more feasible for an open T&T system to build forward on the three mentioned Primitive Types, by defining Core Data Types and specialise those in Business Data Types.

Transmitting XML data between nodes poses an important challenge. Similarly typed message-parts might not represent the same data on both sides of an XML exchange. If XML is chosen as a basis for a type system with subtyping, methods are required that can tell the relation between two schemas. To match XML-parts, similarity measures have been proposed, such as Jeong et al. (2008) which use supervised learning. Another example is the Cupid generic schema matching tool (Madhavan et al., 2001), which employs linguistics-based matching, element- and structure-based matching and key and reference constraints. Unfortunately, common tools and methods such as Cupid or the XML reduction algorithm mentioned above do not provide the required information. By always basing new types on previously agreed types, the challenge is avoided.

The “Based On” relationship is grounded in type theory, as elaborated in Milner (1978). The challenge is to construct a type system, for information systems to be tolerant with regard to new types, for instance when information flows are extended with new event types or specialised attributes.

How to process specialised or generalised types is defined by a (meta)property named “Variance”. In modern type theory; the following three variance types are defined (Karnok and Kemény, 2012)

- Covariance: where you expect a type T, you may pass in a U, where U extends T
  - Example: you want to print a comma separated list of objects, you can pass in a list of strings safely, because strings extend object and are allowed to be treated as simple objects (‘String’ is based on ‘Object’).
  - Function input types are considered covariant
- Contra variance: where you expect a type T, you may pass in an U where T extends U
  - Example: I return an apple to my client but he treats it only as fruit (‘Apple is based on ‘Fruit’).
  - Function return types are considered contra variant
- No variance: you expect a T and you get a T, not any subtype or super type
  - Example. I give you an apple, you take a bite and give it back to me as Apple
  - Functions doing side-effects on its parameter (modifies it in place)

In terms of an open T&T type system the variance cases can be described via the “Based On” or “Specialise” operator. The operator is not XML specific; it is independent from syntactical representation of the types. As newly defined associations, attributes and data types are always "Based on" existing types, they are covariant to the existing types. This means that a system that supports the existing type also will support the new type.

The covariancy is defined on the lexical representation of data types, not on the semantics. Semantically information may be lost. If one receives the height of a pallet load, but stores it as a Pallet Measurement, syntactically this information will fit. Semantically, though, we are not sure any more which pallet measurement has been stored: it could be the pallet length or width instead of the height. In these cases a type code or qualifier is needed to retain the semantics.

A definition of a new data type is illustrated by the XML snippet:

```
<Definition>
        <DataType="LengthMeasure">
                <BasedOn DataType="Measure">
                <SupplementaryComponentFacet>
                        <MeasureUnit>MTR</MeasureUnit>
                </SupplementaryComponentFacet>
        </DataType>
        <DataType="ShapeCode">
                <BasedOn type="Code">
                <ContentComponentFacet>
                        <Enumeration>Box</Enumeration>
                        <Enumeration>Platform</Enumeration>
                        <Enumeration>Pipe</Enumeration>
                </ContentComponentFacet>
        </DataType>
</Definition>
```

7 Definition of new messages

The information that is uploaded by Upload clients to Server Nodes is assembled in observation messages.

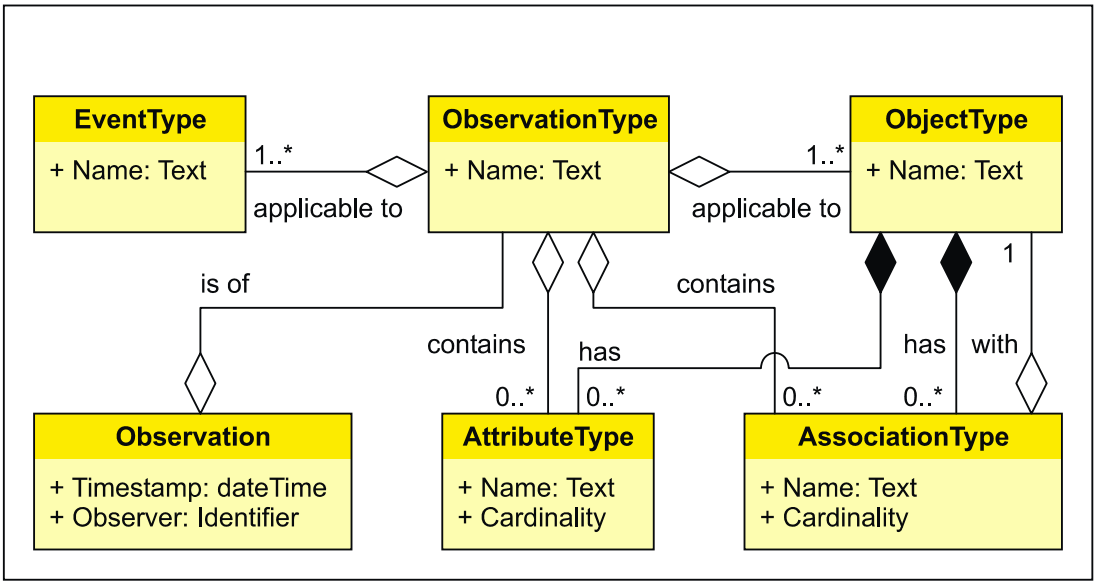

Figure 10. Observation Type

The combination of an Event Type and an Object Type determines the Observation Type. The structure of an Observation Type is communicated by the Server Node to the Upload Client. That structure follows the data model of figure 1. In fact the information to be uploaded is a sub set of the information model of the Object Type. Not all Attributes and Associations in the information model need to be present in the Observation Type, but all Attributes and Associations in the Observation Type must exist in the information model.

An Observation Type definition is illustrated by the following XML snippet:

```
<Definition>
        <ObservationType Name="UnloadPallet">
                <Object ObjectType="Pallet"/>
                <Event EventType="Unload"/>
                <Attribute Name="Location" Datatype="UNLoCode"/>
                <Association Name="Use" ObjectType="Forklift">
        </ObservationType>
</Definition>
```

Definitions of Observation Types are propagated by Server Nodes to the other Server Nodes.

An actual Observation as an instance of this Observation Type may look like:

```
<Observation Timestamp="2013-09-14-15:38:00Z" Observer="8734567">
        <ObservationType Name="UnloadPallet"/>
        <Event EventType="Unload"/>
        <Object ObjectType="Pallet" ID="123" URI="pallet.com">
                <Attribute Name="Location" Value="NLZWO"/>
                <Association Name="Use" ObjectType="Forklift" ID="567" URI="truck.com"/>
        </Object>
</Observation>
```

8 Implementation considerations

As stated previously, the technical protocols to be used for uploading and querying in an open T&T system are outside the scope of this chapter. A simple web service protocol, as specified in WS-i (2006) suffices. The XML messages to be used have a very simple structure, as was illustrated in the various XML snippets. A few XML schemas may be defined and referenced to in WSDL files that support dialogues such as illustrated in figure 3.

The ID-URI combination offers a simple but powerful mechanism for identification. It can be integrated in transport labels, such as described by ISO 15394 (ISO, 2009) and ISO 22742 (ISO, 2010). The identification may also be programmed into RFID transponders or be printed as part of consumer labels with product information, optionally together with usage instructions and freshness information (Kemény.et al., 2011b). For consumers, applications (or 'apps') may be developed, that immediately present relevant information upon scanning the identification with a mobile phone or tablet.

As the information upload protocol is not very complex, the upload function may be performed by an 'app' in a mobile phone as well. That would allow any party in a supply chain to scan product identifications and upload information, without previous agreements or arrangements.

Prerequisite for an open T&T system is that the Server Nodes are interconnected, as (at least) definitions need to be propagated. Upload Clients need to connect to the Server Nodes that administer the URI's of the Objects the Upload Client handles. Query clients in fact need to connect to only one Server Node. As Server Nodes are interconnected, Queries and Query-parts may be propagated to those Server Nodes that hold the information requested.

Although the T&T system is open, it does not mean that all data contained in the system is available to everyone. Tracking and Tracing information is often sensitive. Information owners need to control who is entitled to which information. Server Nodes are under the control of the Object information owner (the one that assigned the URI to the Object). Therefore the information owner can control who has access to the information (what queries will be responded to). The authentication mechanism and the authorisation scheme to use is outside the scope of this chapter.

When the ownership of an Object is transferred, the new owner should take over the ability to control information access. As the URI may be affixed to the Object in an unalterable way (programmed in an RFID chip, printed as a barcode or otherwise), this poses a challenge. One of the solutions is to use Server Node services that are trusted by all participants in the supply chain. Another solution may be some technical redirect mechanism. To specify such mechanism also is outside the scope of this chapter.

9 Discussion and future work

In this chapter an open Tracking and Tracing system is described that allows to track and trace any object through any supply chain. Because the data model of the objects is extensible the system is truly ubiquitous. It can be used for tracking logistical units through a transport chain, for determining the source of products and ingredients, for controlling maintenance and reverse logistics etc. The system can handle food products, fast moving consumer goods, high tech equipment, apparel, in fact any type of objects that can be identified.

The open T&T system can work with any identification scheme, for instance the "License plate" scheme that is standardized by ISO (ISO, 2006). This scheme is compatible with the Serial Shipping Container Code that is used by the Fast Moving Consumer Goods sector, the Logistics sector and the Automotive sector. It

is also compatible with the EPC Global system. The only extra element is the inclusion of a URI where messages can be sent to and where information on the objects can be retrieved.

The description in this chapter is not complete. The architecture and protocols are only sketched. Data structures and message structures are only illustrated, not exactly specified. In implementation projects the details must be elaborated. That may (and should) lead to standardisation of the protocols.

References

ADVANCE project consortium (2010): ADVANCE official project website. URL: http://advance-logistics.eu/ Last accessed: January 7, 2014.

Ala-Risku, T. and Kärkkäinen, M. (2004): A Solution for the Material Delivery Problems in Construction Projects; Thirteenth International Working Seminar on Production Economics, February 16-20, 2004, Igls/Innsbruck, Austria.

Ballard, R., L. (1996): Methods of inventory monitoring and measurement; Logistics Information Management, Vol. 9, No, 3, pp. 11-28.

Blommestein, F.B.E. van (2014): Structured Communication for Dynamic Business, PhD thesis, University of Groningen, ISBN 978-90-367-6396-7, http://www.flowcanto.com/thesis.pdf

Clarke, M. (1998): Virtual logistics; International Journal of Physical Distribution & Logistics Management, Vol. 28, No. 7, pp. 486-507.

EPC Global (2014), http://www.epcglobalinc.org/index.html, retrieved 2013-09-14.

Främling, K. (2002): Tracking of material flow by an Internet-based product data management system (in Finnish: Tavaravirran seuranta osana Internet-pohjaista tuotetiedon hallintaa). Tieke EDISTY magazine, No. 1, Publication of Tieke (Finnish Information Society Development Centre), Finland.

Främling, K., Kärkkäinen, M., Ala-Risku, T., and Holmström, J., (2004) ”Managing Product Information in Supplier Networks by Object-oriented Programming Concepts”, International IMS Forum, May 17-19, 2004, Villa Erba - Cernobbio - Lake Como, Italy.

Gunasekaran, A., and Ngai, E.W.T., (2004): Information systems in supply-chain integration and management. European Journal of Operational Research, Vol. 159, No. 2, pp. 269-295.

Hiscox, M.J. and Smyth, N.F.B. (2011): Is There Consumer Demand for Improved Labor Standards? Evidence from Field Experiments in Social Product Labeling, Social Science Research Network, April 2011

Ilie-Zudor, E.; Kemény, Zs.; van Blommestein, F.; Monostori, L.; van der Meulen, A. (2011): A survey of applications and requirements of unique identification systems and RFID techniques; Computers in Industry 01/2011

ISO (International Organization for Standardization) (2006): ISO/IEC 15459-1:2006 Information technology -- Unique identifiers -- Part 1: Unique identifiers for transport units

ISO (International Organization for Standardization) (2009): ISO 15394:2009, Packaging -- Bar code and two-dimensional symbols for shipping, transport and receiving labels

ISO (International Organization for Standardization) (2010): ISO 22742:2010, Packaging -- Linear bar code and two-dimensional symbols for product packaging

Jeong, B.; Lee, D.; Cho, H.; Lee, J. (2008): A novel method for measuring semantic similarity for XML schema matching. Expert Syst. Appl., 34(3), pp. 1651–1658, DOI: 10.1016/j.eswa.2007.01.025, URL: http://dx.doi.org/10.1016/j.eswa.2007.01.025.

Kaplan, R. S., (1998): Innovation Action research: Creating New Management Theory and Practice. Journal of Management Accounting Research, vol 10, pp.89 –118.

Kärkkäinen, M., Ala-Risku, T., and Främling, K., (2004), “Efficient tracking in short-term multi-company networks”, International Journal of Physical Distribution & Logistics Management, Vol. 34, No. 7., pp. 545 – 564.

Karnok, D, Kemény, Zs. (2012): Definition and handling of data types in a dataflow-oriented modelling and processing environment, MITIP 2012.

Kemény, Zs.; Ilie-Zudor, E.; Fülöp, J.; Ekárt, A.; Buckingham, C.; Welch, P.G. (2011a): Multiple-participant hub-and-spoke logistics networks: challenges, solutions and limits. In: Proc. of the 13th International Conference on Modern Information Technology in the Innovation Processes of Industrial Enterprises MITIP2011, Trondheim, Norway, June 22–24 2011, pp. 20–29.

Kemény, Zs.; Szathmári, M.; Kemény, L.; Bozóki, S.; Ilie-Zudor, E. (2011b): Quality indication and supply management of perishable products with optical labels for low-end demands. Proc. of the 13th Int. Conf. on Modern Information Technology in the Innovation Processes of Industrial Enterprises MITIP 2011, Trondheim, Norway, 2011, pp. 202–211

Kosk, N. (2014): the Reverse Logistics Cycle, Supply & Demand-Chain Executive, http://www.sdcexec.com/, retrieved 2013-09-14.

Lee (2000): Context-dependent Semantic Values for E-negotiation, Proceedings of WECWIS, 2000

Madhavan, J.; Bernstein, P.A.; Rahm, E. (2001): Generic Schema Matching with Cupid. In: Proceedings of the 27th International Conference on Very Large Data Bases, VLDB '01, pp. 49–58, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, ISBN 1-55860-804-4,
URL: http://dl.acm.org/citation.cfm?id=645927.672191.

Milner, R. (1978): A theory of type polymorphism in programming. Journal of Computer and System Sciences, 17, pp. 348–375

Monostori, L.; Kemény, Zs.; Ilie-Zudor, E.; Szathmári, M.; Karnok, D. (2009): Increased transparency within and beyond organizational borders by novel identifier-based services for enterprises of different size, CIRP Annals – Manufacturing Technology, Vol. 58, No. 1, 2009, pp. 417–420. (DOI: 10.1016/j.cirp.2009.03.086)

Sciore (1994): Using Semantic Values to Facilitate Interoperability Among Heterogeneous Information Systems, ACM Transactions on Database Systems, Vol. 19, No 2, June 1994.

Sowa, J. (2000): Knowledge representation. Brooks Cole Publishing Co., Pacific Grove, CA.

TraSer project consortium (2006): TraSer official project website. URL: http://www.traser-project.eu/ Last accessed: January 7, 2014.

UN/ECE (2003): Core Component Technical Specification,
http://www.unece.org/fileadmin/DAM/cefact/codesfortrade/CCTS/CCTS_V2-01_Final.pdf, retrieved 2013-09-14.

W3C (2004): XML Schema Part 2: Datatypes Second Edition, 2004, http://www.w3.org/TR/xmlschema-2/, retrieved 2013-09-14.

WS-i (2006): Webservice Interoperability Organisation Basic Profile Version 1.1 Final Material 2006-04-10, http://www.ws-i.org/profiles/basicprofile-1.1.html, retrieved 2013-09-14.