

Using ORM in an Ontology Based Approach for a Common Mapping Across Heterogeneous Applications

Baba Piprani

SICOM, Canada
babap@attglobal.net

Abstract. Enterprise application integration efforts face several impediments like: lack of common data semantics, conflicting or ambiguous meanings, homonymic usage, including non-existent documentation etc., resulting in massive duplications of master data—master data being defined as one common and true version of the truth in terms of client, customer, item, or mainstay entities for an organization. This use case demonstrates how ORM facilitated the definition of ontological bridges to establish and enable common mappings across disparate applications, which subsequently were realized in SQL based implementations that operate in a heterogeneous RDBMS environment. Using ISO 11179 ontological based entry points, the concept of ORM derived common mapping keys bridged across multiple applications enabled access to applications that contain data specialized in their subject area— enabling on-demand assembly and dynamic integration across stove-piped systems.

1 Application Integration Issues

Data integration from application systems continues to face traditional issues of conflicting and ambiguous meanings for the same item of data. Lack of common data semantics is seen to be pervasive across applications, resulting in massive duplication of master data. It is appropriate to quote the ‘Helsinki Principle’ from ISO TR9007 [2]:

“Note: These utterances are to be interpreted (recursively) as international English utterances: Any meaningful exchange of utterances depends upon the prior existence of an agreed set of semantic and syntactic rules. The recipients of the utterances must use only these rules to interpret the received utterances, if it is to mean the same as that which was meant by the utterer.”

In other words, for an application to be able to effectively communicate with another application, there must be a set of agreed upon semantic and syntactic rules of the utterances that are being exchanged. This fundamental concept is often overlooked and is often distorted by the adoption of current computer technology programming requirements for cross referencing data via multiple mappings at the lexical (value type) or syntactic level.

The following examples in the air transportation sector demonstrate actual cases where the applications are unable to share data, or data searches that require complex query processing:

- Entity naming: Say a particular helicopter services company is called “Canadian Helicopters” in one database application, and is called “Canadian Helicopters Limited” in another database application. Are these the same entity with perhaps mis-spelt names or incomplete names? Not given enough supporting data on that organization, we don’t know for sure.
- Entity identification: The Canadian airline known to the traveler public as “First Air” has an ICAO (International Civil Aviation Organization---an agency of the United Nations, that codifies the principles and techniques of international air navigation) identification code as ‘FAB’ . This same airline is given an IATA (International Air Transport Association---a trade association representing and serving the airline industry world-wide) code of ‘7F’. That means if one has to communicate across an ICAO code based application to an IATA code based application, a mapping has to be done between the 2 codes to be able to retrieve data about the same entity---not all airlines have both codes. It is almost impossible to establish a trace since several organizations A query to follow through on a series of occurrences concerning a particular aircraft that has changed ownership may not even materialize
- Multiple Codes for airport locations: A similar situation exists where each country may internally assign their own airport location codes for local airports that may or may not be in the purview of the international arena. There is no definitive single answer for a location
- Rounding Errors for Latitude/Longitude: resulted in multiple airports at the same location.

2 Use Case – Transport Canada Requirement and Background

A critical air traffic situation arose in Canada on September 11, 2001 where hundreds of planes bound to the US over the Atlantic were re-routed to Canada. There was an urgent flurry of activity to determine the most suitable match between airport runway properties; emergency service facilities; airside airport services; passenger airport facilities etc. and, the incoming aircraft with respect to the aircraft type; aircraft size; number of passengers in each aircraft; fuel conditions in each aircraft etc. This presented a Herculean task for Transport Canada (TC), the regulatory agency responsible for the transportation sector in Canada, to access such data across multiple applications on demand! Needless to say, not all applications were in a condition to be cross-navigated considering the urgent timing of the requirement.

What this situation demonstrated was that data needed to be accessed on demand, across different perspectives and dimensions, and dynamically assembled as required within the chosen contexts. The Transport Canada Aviation Object Dictionary (AOD) was born. In brief, the objectives of the AOD were:

- Enable logical (virtual) integration between participating systems;
- Enable harmonization of identification schemes and industry standards for airports and air carriers;

- Allow unrelated applications to share a common pool of concept-based connected data via concept-based bridges.

A consultant study recommended a solution that would create cross reference mappings for selected syntactic data items like airport locations, airline, aircraft types etc. at the lexical object (value type) level across the initial set of 5 involved applications. It was quickly determined that this simple cross referencing requirement at the syntactic level could result in an exponential increase in the mappings across systems e.g. 5 systems → 20 mappings, resulting in 100s of mappings across systems. Conclusion: another solution had to be found i.e. apply knowledge based concepts for the Aviation Object Dictionary.

3 Use Case – TC Ontology Based Solution Approach

We explored an ontology based approach for navigating across concepts and across subject domains i.e. to define semantic links so as to establish relationships between different sets of data based on common ontological terms.

There are many definitions of ontology. For the purposes of this paper, the preferred definition is: *“An ontology is a model of some portion of the world and is described by defining a set of representational terms. In an ontology, definitions associate the names of entities in a universe of discourse (e.g., classes, relations, functions, or other objects) with human-readable text describing what the names mean, and formal axioms that constrain the interpretation and well-formed use of these terms.”* [3] In other words, ontology refers to ‘what exists’, ‘what is’, ‘something that is’, i.e. “an explicit specification of a conceptualization” [4].

When it comes to modeling the Universe of Discourse, the modeling constructs of the popular Object Oriented models and Entity Relationship or attribute based models only provide definitions and associations among the names of the entities via objects, attributes, relations, functions etc., i.e. they generally do not support the required axioms to constrain the interpretation of the necessary ontological constructs [9]. NIAM, on the other hand, supports many of the ontological aspects mentioned above [8]. The ISO TR9007 Conceptual Schema report [2] took an example scenario and applied Entity Relationship, NIAM, and Interpreted Predicate Logic approaches to solve this scenario so as to demonstrate some measurability in terms of semantic and expressiveness match of the modeling approaches. The solutions clearly demonstrated the ability of NIAM in supporting a much larger number of axioms and constraints in its modelling paradigm than does the ER approach.

A clear distinction not much emphasized is the separation of the ‘thing’ from the ‘name of the thing’. Object-oriented models and Entity Relationship or attribute based models lack this important distinction. The ‘native way of thinking’ in these approaches is to essentially pick up an ‘identifier’ or a ‘name of a thing’ and use this to map across other identifiers. NIAM and ORM on the other hand, have made this a clear demarcation, even to the extent that there are axioms, relationships and constraints between how the ‘thing’ can be ‘named’. It is this important highlight that is the main theme of this paper and how this very practical real life situation is being solved, not to mention that this lack of separation between the ‘thing’ and the ‘name

of the thing' in itself contributes to limiting the expressive constructs in the declaration of integrity constraints in the respective modelling paradigm.

Even in Ontological topology there do not appear to be clear rules to distinguish between the 'thing' and 'reference of the thing'. Take for example the domain of analgesic agents. Navigating down the hierarchy of analgesic agents in Ontology we come to Non-Narcotic Analgesic, then Analgesic and Antipyretic, the next level being Acetaminophen. Herein is an interesting and distinguishing trait demonstrating the differences in the modelling of this concept domain. The Ontological topology continues to identify 'Tylenol', 'Anacin-3', 'Temptra', and 'Datril' as the next level, so if we want to analyze patients taking analgesic agents, any one of these would then be counted. But these names of the drugs 'Tylenol', 'Anacin-3', 'Temptra', and 'Datril' are treated as different and separate concepts, including their parent concept, Acetaminophen.

In NIAM and ORM, unless there is a specific constraint or a total relationship associated with these 4 'subtypes', they would be treated as referencing objects or 'lexical objects' for the same 'thing' i.e. Analgesic and Antipyretic agent, with even Tylenol becoming yet one other referencing object. In other words, in NIAM and ORM we see a clear separation of applying the referencibility of the concept of Analgesic and Antipyretic' agent to that of Acetaminophen, Tylenol, Anacin-3, Temptra and Datril, whereas this distinction is not formalized in other modelling paradigms. The reader is referred to related work in Ontology and ORM to amplify some of these aspects, see [1]. In NIAM, an ontological object is a non-lexical object type or in ORM terms, an entity type.

Applying the ontological aspects to the situation on hand, the task for AOD was to develop a centralized registry of data elements for standardized objects with respect to naming, identification and meaning for objects in Civil Aviation: Carrier (Airline); Aircraft; Airport; Contacts; Operational; Occurrence; and Regulatory. The AOD would then be used to provide "services" to applications needing access to data from participating systems based on the ontological object bridges that were defined.

We came up with a semantic model for AOD drawn up in ORM. We describe the ontology by defining and associating a set of Non Lexical Object Types (entity types) and their relationships for the object domains as above. However, in this paper only a small part of one ontological domain is shown. Then, using ORM, we separated the representational aspects from the Non Lexical Object Types (entity types) to form Lexical Object Types (or value types) for naming and identification.

There are 2 parts within the definitional aspects of the Non Lexical Object Type Model (or Entity Type model). First, there is the hierarchy of concepts in the ontology model, which would map directly to the ORM subtype hierarchy (and thus elegantly to the SQL sub table model). See Fig. 1 for an ontology model topology for location.

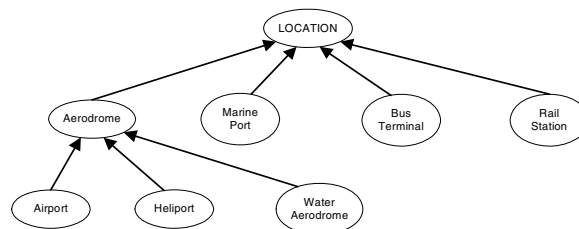


Fig. 1. Location Object Hierarchy

Secondly, we need to add suitable axioms to express other relationships or associations between concepts, and to constrain their intended interpretation. This second step is transformed into an ORM information grammar to take into account the necessary attributes required to define a unique location and properties of interest in the selected scope, as to establish correctness.

Since each application had its own reason for following a particular identification scheme, it was up to AOD to provide a stepping stone over these different identification schemes, i.e. AOD would provide a UNION operation and tag each object occurrence with a unique identifier, and turn around and provide mappings to whatever identifiers existed.

Fig. 2 shows how the semantics for ‘location’ as represented across various applications with their local identification scheme. In effect, there is no total 1:1 unique reference bridge between the non-lexical object (ontology object location) and the multiple or non-total references shown.

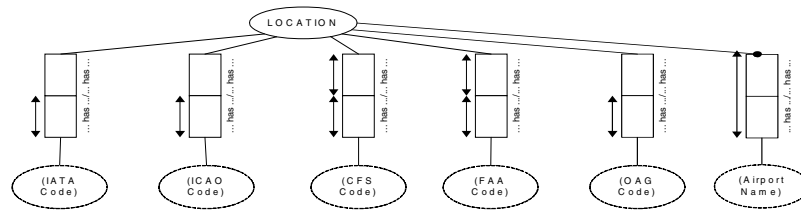


Fig. 2. Multiple or non-total references for an object

The solution for AOD was to establish a 1:1 total reference. It was a simple ORM bridge model that would provide references from the non-total lexical objects (entity types) to other applicable lexical objects (value types) through the ‘total AOD 1:1 reference’ for the non lexical objects. See Fig. 3.

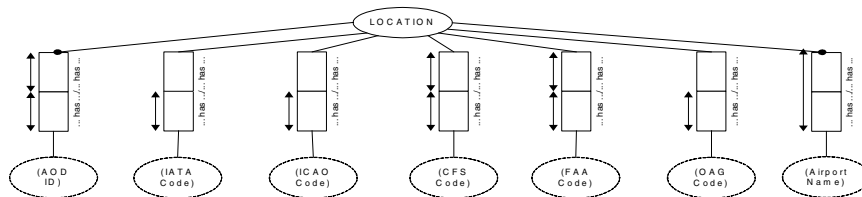


Fig. 3. Adding a total 1:1 reference via AOD

4 Use Case – The Reality of Implementation

After a brief period of prototyping including paper prototypes in ORM populations, transcribing to an attribute based data model in ERWin (in-house standard toolset), and creating schema instances on SQL Server, a successful prototype was conducted using actual and realistic data. This section provides details on the factors encountered in the implementation.

4.1 Establishing the ‘One-Version-of-the-Truth’ AOD Master Data Set

In reality, this “simplistic notion” of a total 1:1 reference for the non-lexical objects (entity types) to be available presents a formidable challenge. A master data set that would represent the one-version-of-the-truth had to be created first in order for other non-total references to be mapped to the total 1:1 reference to be provided via AOD.

A set of probes were designed to search, filter, sort and match the incoming source data that would map or transfer the incoming data to the heavily constrained master data schema (AOD database).

The master data set was contained in a SQL Server environment using the schema integrity constraint syntax as per ISO SQL99 [6]. Automated metadata parameters would determine update or insert actions for incoming source data and appropriately route the data to the master data set via a data quality firewall [10] that would validate all incoming occurrence values. Typical reject rates were in the order of 10-20% with integrity violations, or missing data in the order of up to 40% along with overlaps from multiple sources being accommodated in a temporal database—the biggest culprits being improperly sequenced chronological data where period overlaps were not in synchronization, along with duplications based on syntactical corrections or white spaces/company suffixes (e.g. co., Ltd., Limited, inc., inc, incorporated etc.).

An ORM schema for implementing the involved concepts was drawn up and shown in Fig. 4 showing the location concept as it is referred to in a participating logical implementation. This information will eventually be used in the Common Mapping Key (CMK) database used to cross reference to AOD.

Necessary attributes for Location (General Aviation Airport), the master data set contains the latitude/longitude, names of airports, address, contact, main city and cities served, along with their non-total alternate identifiers ICAO, OAG, IATA, CFS and FAA etc. In addition, probes could be used to verify airport location validity by number of terminals including runway types, runway areas, airport areas etc.

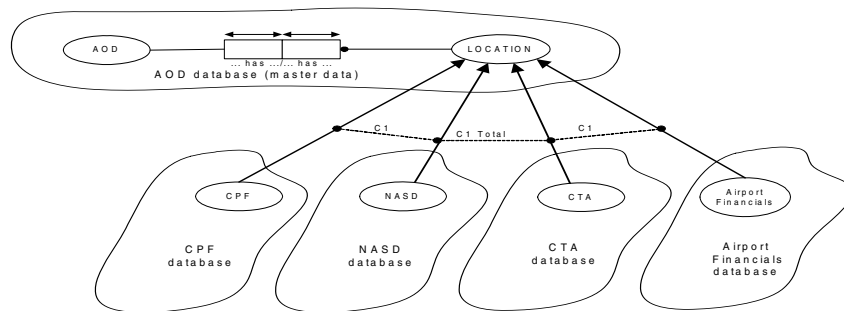


Fig. 4. An ORM schema across alternate concept references logically treated as a subtype configuration is aligned with physical databases based on alternate concept references

4.2 Establishing the CMK Ontology Based Bridges

Having established a valid master data set to be able to provide the total 1:1 reference to the ontological objects via AOD database, the task now was to establish bridges

from the involved concepts to available identifiers in the participating application systems. Towards this, a metadata collection approach was initiated that would establish a common mapping key (CMK) to bridge the concept along with its identification attributes, to the data element name in the participating application. Fig. 7 demonstrates the sentence types involved for CMK.

There are 2 kinds of bridges—one maps the concept at the metadata level to the qualified column of the data element in its local host environment, and secondly, a value-based mapping that maps a particular local column value to the AOD key value. The metadata level CMK mapping would enable one to connect from the concept to the column name(s) in the participating application via a bridge, as shown in Fig.5.

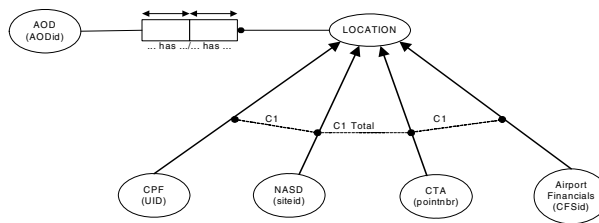


Fig. 5. CMK mapping at the metadata level for columns in participating databases

The value-based mapping would be a 2-step approach to enable one to connect a particular value of a column occurrence for an identifier in the participating application to the definitive AOD-identifier, and then use this AOD-identifier to map to the identifier value in the selected system of choice, see Fig. 6.

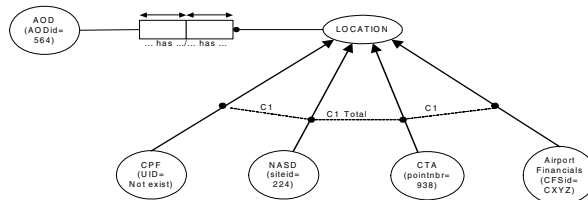


Fig. 6. CMK mapping at the value-based level for columns in participating databases

5 ISO 11179 and ISO 19763 Alignment

The ISO 11179:1993 Metadata Registries standard [5] deals with registering and administering associated properties of concepts. Part 3 of the standard defines the Metamodel that defines the registry, object classes, data element concepts and data elements. The ISO 11179 standard was used to help derive a mapping of the AOD definitions at Transport Canada. The ISO 19763 Framework for Metamodel

Table 1. ISO 11179 based mappings used to derive the Common Mapping Key constructs

Object Class	Data Element Concept	Data Elements
Carrier	Name	Legal name
Carrier	Name	Operating name
Carrier	Name	Trade name
Carrier	Name	Preferred name
Carrier	Name	Carrying on Business name
Carrier	Identity	IATA Code (Carrier)
Carrier	Identity	OAG Code (Carrier)
Carrier	Identity	ICAO Code (Carrier)
Carrier	Identity	AOD Party id
Location	Identity	IATA Code (Location)
Location	Identity	OAG Code (Location)
Location	Identity	ICAO Code (Location)
Location	Identity	CFS Code (Location)
Location	Identity	FAA Code (Location)
Location	Identity	AOD Location id

Interoperability standard [7] was used to help establish the architecture and metadata model. Table 1 shows the mappings used for deriving CMK values. For the scope of this paper, object class instantiations contain Carrier and Location.

6 Defining the CMK Database

The CMK database bridges the Object Classes, associated Data Element Concepts, Data Elements with the Application, and the qualified column metadata of the application's database. This basic construct is used for mapping the AOD identifier with the corresponding Data Element Concept as CMK sentence types in Fig. 7.

<p>Sentence Type 1: <i>Application CPF contains object class CARRIER with data element concept IDENTITY having data element ICAO-CODE;</i></p> <p>Sentence Type 2: <i>Schema CPFDB has Table CARRIERUID with column UID;</i></p> <p>Now bringing the parts together, Sentence Type 3: <i>[Sentence Type 2] is contained in [Sentence type 1]</i> i.e. <i>[Application CPF contains object class CARRIER with data element concept IDENTITY having data element ICAO-CODE]</i> <i>is contained in [Schema CPFDB has Table CARRIERUID with column UID];</i></p> <p>Promulgating these sentences to the 1:1 total reference to AOD: <i>[Application AOD contains object class CARRIER with data element concept IDENTITY having data element PARTY_ID] is contained in [Schema AODDB has Table AOD_PARTY with column AOD_ID];</i></p>
--

Fig. 7. CMK Sentence Types

7 Positioning the AOD and CMK Databases in the Data Quality Firewall Architecture

Since data quality is of utmost importance to the viability of the AOD master data set and CMK databases, the advanced generation data warehouse architecture embodying the data quality firewall as defined by [10] was used in the configuration shown as paralleled databases, one for the master data (AOD Collector) and the other for CMK (column mappings and value mappings) as shown in Fig 8.

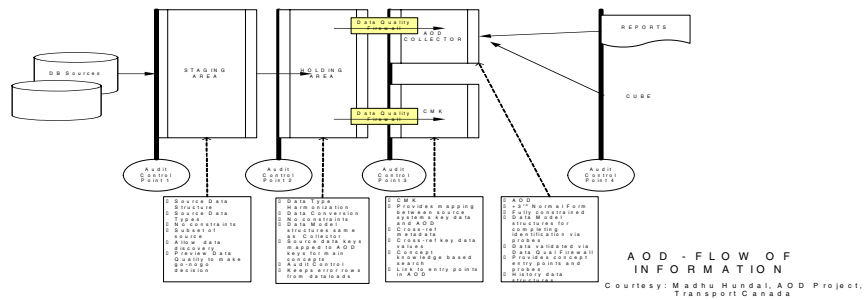


Fig. 8. Positioning AOD and CMK in the Data Quality Firewall Architecture

8 Use Case – User Acceptance

The AOD had initial success with data from 3 participating systems. The delivery architecture is being designed as a Service Oriented Architecture (SOA). The user’s vision [11] in Fig. 9 is that a user functional area has certain requirements for data in the form of a dashboard collected from various sources for analysis. AOD provides an on-demand retrieval mechanism from heterogeneous databases via automated generation of the associated SQL query, subsequently stored in a query library.

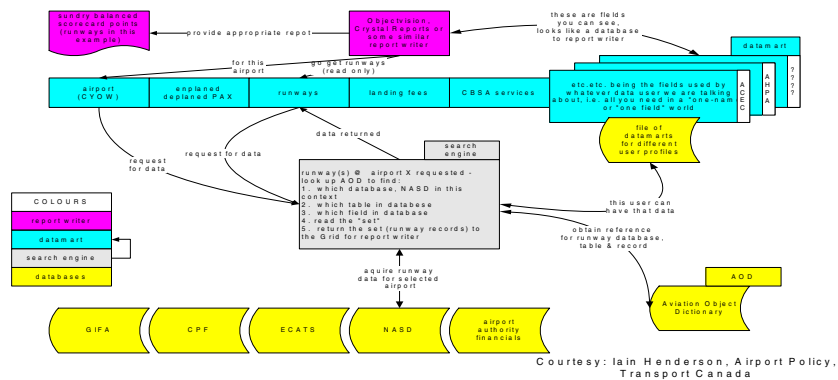


Fig. 9. AOD from a user's perspective

9 Concluding Remarks

Using ORM with an Ontology based bridge approach helped solve a serious and complex problem. Using ISO 11179 constructs as defined via ORM, an attribute based data model (ER) was defined which then was mapped to a SQL Server database. Interface stored procedures were established to allow dynamic assembly of user required data elements from various participating heterogeneous systems (SQL Server, Oracle), enabling virtual integration across stove-piped systems, giving the vision of a virtual data warehouse without the extraction and loading of data. The data quality firewall architecture was used to validate and guarantee integrity in the 2 supporting databases for AOD master data, and Common Mapping Key (CMK) which contained the ontological bridge data.

I would like to acknowledge and thank the AOD team, staff and users, Michel Villeneuve, Madhu Hundal, Vasko Mioviski, Iain Henderson, Dave Dawson, Tom Nash, Dave McCutcheon, Peter Wesley, Tracey Boicey, Rob Henkel and the AOD Working Group and Steering Committee for their support and vision in making AOD happen.

References

1. Bollen, P.: Extending the ORM conceptual schema design procedure with the capturing of the domain ontology. In: EMMSAD 2007. Proceedings of the Eleventh International Workshop on Exploring Modeling Methods in Systems Analysis and Design (2007)
2. van Greutheysen, Joost, ed.: Technical Report on Concepts and Terminology for the Conceptual Schema and the Information Base. ISO Technical Report ISO IEC TR9007:1987. International Standards Organization, Geneva (1987)
3. Gruber, T.: A Translation Approach to Portable Ontology Specifications. Knowledge Acquisition 5(2), 199–220 (1993)
4. Gruber, T.: Toward Principles for the Design of Ontologies Used for Knowledge Sharing. Technical Report KSL 93-04. Knowledge Systems Laboratory, Stanford University (1993)
5. International Standard ISO IEC 11179:1993. Metadata Registries. International Standards Organization, Geneva (1993)
6. International Standard ISO IEC 9075:1999. Database Language SQL. International Standards Organization, Geneva (1999)
7. International Standard ISO IEC 19763:2007. Metamodel Framework for Interoperability (Part 1: Reference Model, Part 3: Metamodel for Ontology Registration, and FCD Part 2: Core Model). International Standards Organization, Geneva (2007)
8. Nijssen, G.M., Halpin, T.A.: Conceptual Schema and Relational Database Design. Prentice Hall, Victoria, Australia (1989)
9. Mahalingam, K., Huhns, M.N.: Ontology tools for semantic reconciliation in distributed heterogeneous information environments. Intelligent Automation and Soft Computing 6(3), 185–192 (2000)
10. Piprani, B.: Using ORM-based Models as a Foundation for a Data Quality Firewall in an Advanced Generation Data Warehouse. In: Meersman, R., Tari, Z., Herrero, P. (eds.) OTM 2006 Workshops. LNCS, vol. 4278, Springer, Heidelberg (2006)
11. Piprani, B.: Ontology Based Approach to a Common Mapping across Heterogeneous Systems. Presentation at Metadata Open Forum, New York (2007), <http://metadataropenforum.org/index.php?id=2,0,0,1,0,0>