ENTITY IDENTITY INFORMATION MANAGEMENT (EIIM)

(Research-in-Progress)

Yinle Zhou
University of Arkansas at Little Rock
yxzhou@ualr.edu

John R. Talburt
University of Arkansas at Little Rock
jrtalburt@ualr.edu

Abstract: This paper introduces and defines the concept of entity identity information management (EIIM). EIIM is a component of entity identity management (EIM) that utilizes data structures, data integration, and entity resolution (ER) methods and algorithms in order to maintain entity identity integrity, a goal that EIIM shares with master data management (MDM). The paper also explores some of the design choices to be considered in implementing EIIM systems in order to balance system efficiency and effectiveness. The paper also describes five types of asserted resolution that complement inferred resolution in supporting the identity information life cycle and maintaining persistent entity identifiers. Finally the paper describes how EIIM has been implemented in the open source ER system OYSTER.

Keywords: Entity resolution, entity identity management, entity identity information management, entity identity structures, OYSTER open source entity resolution system

INTRODUCTION

EIIM is a component of entity identity management (EIM) that utilizes data structures, data integration, and entity resolution (ER) methods and algorithms in order to maintain entity identity integrity. Entity identity integrity is one of the basic tenets of data quality that applies to the representation of a domain of real-world entities in an information system [1]. Entity identity integrity requires that

- Each entity in the domain has one and only one representation in the system
- Distinct entities have distinct representations in the system

The same concept has also been described as "proper representation" [2]. Figure 1 illustrates how EIIM is positioned at the intersection of entity identity management (EIM), entity resolution (ER), and master data management (MDM).

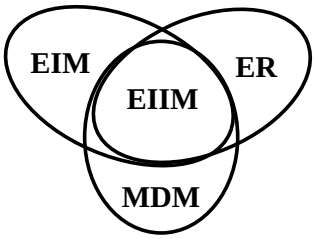

Figure 1: EIIM at the Intersection of EIM, ER, and MDM

Entity Identity Management (EIM)

Although EIIM is a component of EIM, the primary focus of EIM is on cyber-security practices and procedures within organizations where the entities are both system users who need system access and potential bad actors who should be denied access [3]. For example, the problem of issuing information system users with credentials and unique identifiers during an initial registration phase, or the issues

around authenticating users and controlling their access to services and resources based on their identifiers and credentials during the service operation phase [4]. Another concern is the "revocation" of identity credentials, which is the process of rescinding access permissions as part of an identity management process [5]. In general EIM tends to focus on risk management in areas such as healthcare, financial services, government, national security, energy, and consumer services [6].

Entity Resolution (ER)

ER is the process of determining whether two records in an information system referring to real-world objects are referring to the same object or to different objects [7]. ER is a key task in data integration where different data sources provide information for a common set of entities. ER in customer relationship management (CRM) is often referred to as customer data integration (CDI) [8]. ER also plays an important role in price comparison search for online shopping [9] and data mining for counter terrorism [10].

The term entity in ER describes the real-world object, such as a person, product, place, or event that has an individual identity. The identity of an entity as represented in an information system comprises a set of attribute values for that entity along with a set of distinct rules that allow that entity to be distinguished from all other entities of the same class in a given context [11]. Although ER is an essential process for EIIM, not every ER system uses EIIM. Most ER systems are cross-sectional in operation because they are primarily designed to resolve identities across a large batch of entity references at a single point in time. On the other hand, EIIM has a longitudinal focus and is an essential component of any ER system that attempts to maintain persistent entity identifiers, i.e. entity identifiers that do not change from process to process. In order to accomplish this EIIM systems must provide some data structure, called an entity identity structure (EIS), for storing the identity information captured or updated during the ER process.

Master Data Management (MDM)

Master data in an organization are the data items that reference the entities that represent the organization's non-fungible assets, such as, customers, employees, products, and equipment. MDM comprises the policies, procedures, and infrastructure needed to accurately capture, integrate, and manage master data [12]. MDM and EIIM share the same goal of maintaining entity identity integrity with EIIM basically representing the technical aspects of MDM that also encompasses the issues of policy and governance.

PROBLEM

EIIM comprises a number of components and processes that interact in subtle, but important ways that are often not well-understood even by those people who use them. Moreover, EIIM has not been the subject of systematic study despite the critical role that plays in EIM, ER, and MDM. There are many different ways in which EIIM can be implemented but often the system architect does not have a complete grasp of the principles at work or the relative merits and consequences of various design options.

THE COMPONENTS OF EIIM

A high-level view of EIIM components and processes is shown in Figure 2.

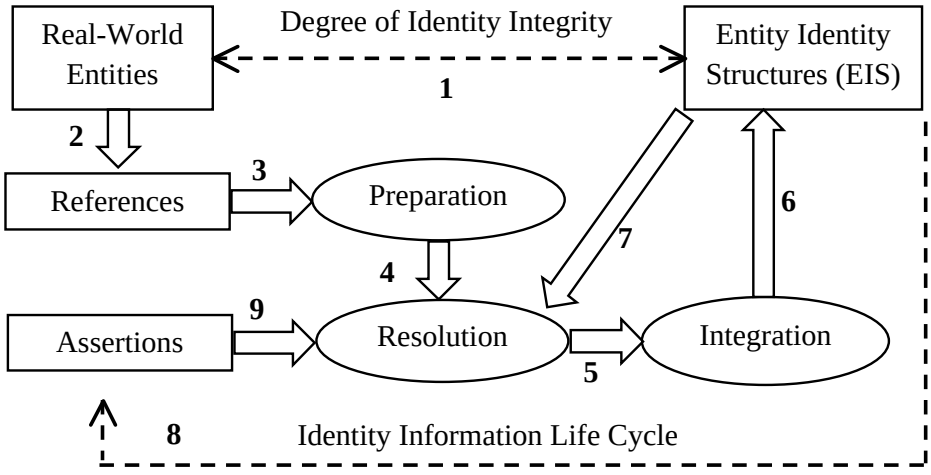

Figure 2: EIIM Components and Interactions

The labeled items in Figure 2 are described as follows

1. The fundamental goal of EIIM is to effect entity identity integrity which is essentially to maintain a one-to-one correspondence between the entity identity structures in the information system and the real-world entities in the domain of interest.
2. References are records in the information system that refer to the real-world entities. References may be generated internally or come from outside the information system through automated or manual interfaces. They may also occur in both structured and un-structured formats. Structured references might be rows in a database that describe entities or simply transactions that refer to entities. Unstructured references are often found in business documents such as contracts or email messages.
3. In most cases, entity references undergo a number of preparation steps to improve the quality of the data. These steps may include entity attribute extraction (in the case of unstructured information), reformatting, standardization, correction, and enhancement. The effectiveness of these data quality processes can have a profound impact on the success of the overall process. Inconsistent representation of attribute values between references due to different coding schemes and data quality is one of the greatest enemies of ER because it creates uncertainty in the matching process. The classification problem in data mining also relies on the matching of attribute values, and it has been shown [13] that inconsistency has an even greater effect on the outcome than the accuracy, timeliness, or completeness of the data.
4. After references have been prepared, the next step is to resolve the references against other references and also against existing EIS. For any given input reference the resolution process must decide if that reference is equivalent to any other input reference or a previously created EIS. This is most often done by using a set of pair-wise matching rules, but can also be done by recognizing that a reference has a particular relationship with other references, a process called "relationship resolution". There are three critical considerations in the resolution process
    a. The selection of the algorithm by which successively selects references and EIS to compare. For efficiency most algorithms try to minimize the number of comparisons that must be made. However as will be shown later, both the way in which EIS is constructed and the way in which the matching rules are mapped to the EIS can limit the degree to which these comparisons can be minimized.
    b. The rules or relationships that will determine equivalence between two references, between a reference and an EIS, or between two EIS.

   c. The mapping of the rules or relationships to the EIS. The most commonly used mappings are attribute-based and record-based mappings. These will be discussed in the next section.
5. In the case that two references or EIS are resolved as equivalent, the transitive closure of equivalence requires that they be integrated into a single EIS. The simplest form of integration is when the EIS is simply a container for maintaining the set (cluster) of equivalent references. This is called a record-based EIS or union structure [14]. Because many references in a record-based EIS will have the same values for an attribute, another approach is to use an attribute-based EIS that simply maintains a list of unique values for each attribute.
6. The integration process will create a new EIS from a single input reference when the ER process is unable to resolve the input reference to any existing EIS. When an input reference resolves to an EIS, the EIS is updated to include the new information from the reference. In the case that it resolves to more than one EIS, the reference and EIS are all integrated into a single EIS.
7. As described above, the EIS are also inputs to the resolution process as well as outputs from the integration process.
8. Perhaps the most important aspect of EIIM is that it is a cyclical process rather than a one-time process. Just as with any type of information, entity identity information has a life cycle as new identities are created, updated, combined, and eventually discarded. EIIM systems have a continual influx of new reference information, and the resolution and integration of these references will impact the state of entity identity integrity of the system. Generally the previously described Steps 3, 4, 5, and 6 are automated steps that infer the equivalence of references and EIS based on the equivalence rules or patterns. Invariably some of these inferences will be incorrect. Combining two references or EIS that are not equivalent is a false positive error, and failing to combine two references or EIS that are equivalent is a false negative error. Because references will almost always have some level of data quality problems and because of the probabilistic nature of the resolution decisions, there will always be some level of error in the resolution process. This is compounded by the fact that the process is repeated. Even if an automated resolution process were to have no more than a 1% false positive error rate, repeating the process twenty times could lead to an overall error rate of more than 18%. Consequently, successful EIIM must balance inferred resolution with external knowledge that is introduced into the system through asserted resolution.
9. Asserted resolution is when resolution decisions are made based on knowledge from external sources rather than inferences based on values and relationships within the system. Assertions override the equivalence rules in the resolution process to directly create, update, or combine EIS directly. Asserted resolution allows the EIIM process to be adjusted for drift caused by errors inherent in the inferred resolution process. For most systems not every false positive or false negative issue can be solved by changing an existing equivalence rule or adding a new equivalence rule. Even small rule changes designed to solve one particular error can often create many other unintended errors. Five types of assertions that assist in the EIIM process are described later in the paper.

As the preceding list shows, EIIM comprises several components and processes. However, the remainder of this paper will focus primarily on factors that affect resolution and integration, why assertion is needed to support EIIM in practical applications, and a discussion of current and future research directions.

RESOLUTION AND INTEGRATION IN EIIM

The resolution and integration components as described in Steps 4 and 5 are core processes and represent traditional ER component of EIIM. It is important to understand some of the factors that influence these processes and their ultimate impact on identity integrity. Two of these factors are the way in which

resolution rules (match rules) are mapped to the EIS and how these mappings can influence the choice of the resolution algorithm necessary to systematically compare and integrate EIS.

Record-Based and Attribute-Based Rule Mappings

As noted in the previous section, a critical component of the resolution process is the mapping of the equivalence rules to the identity attribute values in the EIS. The two most common rule mappings are attribute-based and record-based mappings. The terms "attribute-based" and "record-based" can also be used as qualifiers to describe the organization of the EIS as well as the mapping of the rules to the EIS. Consider the case of record-based mapping. With respect to equivalence rules, record-based denotes the mapping constraint that the attribute values used in a particular rule must all come from the same input reference. In the case that one input reference is being compared to another input reference, record-based mapping is the only option. However when a reference is compared to an EIS or one EIS is compared to another EIS, other mappings are possible and the choice of mapping must be carefully considered. The implementation of record-based rule mapping also implies a record-based EIS organization, i.e. a structure that preserves the relationship among attribute values that came from the same input reference. The simplest form of a record-based EIS organization is one in which the EIS simply serves as a container for the collection of equivalent input references often referred to as a "cluster".

An alternative is attribute-based rule mapping. Attribute-based mapping allows any value for an identity attribute to be used in a matching rule independently of the source of the values used for other attributes in the rule. For example, if a rule compared both first and last names, attribute-based mapping would allow the first and last name values being compared to come from different input references. From a data structure perspective, an attribute-based EIS is only required to aggregate all of the values for an attribute from equivalent input references. An attribute-based EIS does not have to retain the original reference source relationship among attribute values. In its simplest form an attribute-based EIS only maintains a list of distinct values for each attributes. This can result in significant savings in storage when there are many equivalent references with identical values for certain attributes. However, this comes at the cost of losing data provenance with respect to which reference or references provided a given value.

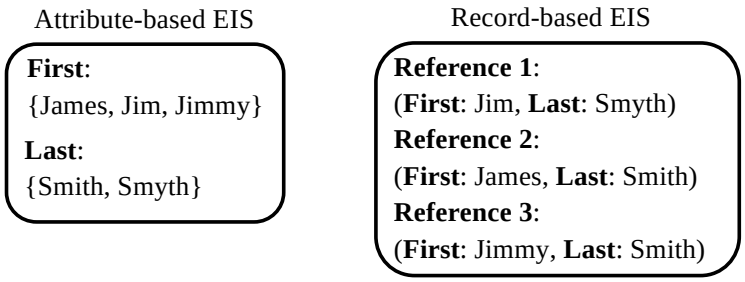

Figure 3: Attribute-based and Record-based Identity EIS Organization

It is important to note here that an attribute-based rule mapping does not require an attribute-based EIS. It is easy to see how a record-based EIS can support attribute-based equivalence rules. Because a record-based EIS retains a list of equivalent references it necessarily retains all of the values. The disadvantage to using a record-based EIS to support attribute-based mapping is that it requires either additional computation or additional storage. Creating the list of unique attribute values by traversing the list of references each time the EIS is used in a rule requires additional computational steps. At the same time maintaining the list of unique values each time a reference is added to the EIS uses less computation, but requires additional storage above and beyond maintaining the list of equivalent references. The obvious advantage of using record-based EIS is that it can support both record-based and attribute-based mappings.

Figure 3 shows an example both an attribute-based and record-based EIS. Note that both contain the same five attribute values, three different values for First and two different values for Last. Without capturing more information in the attribute-based EIS shown in Figure 3, it is not possible to tell how many equivalent references were integrated to create it. Therefore an attribute-based EIS organization cannot support record-based rule mapping. The record-based structure shown in Figure 3 has data redundancy with two duplicate values of "Smith" and the metadata tags of "First" and "Last" are repeated in each reference.

One-Pass and R-Swoosh Resolution Algorithms

Another design choice in EIIM system is the algorithm for systematically comparing references and EIS in the resolution process. The two algorithms discussed here are One-Pass record linking and R-Swoosh [14].

The One-Pass Record Linking algorithm is very straightforward in that each input reference is selected in order and compared to every other input reference one, and only one, time. However to work correctly, One-Pass requires both record-based rule mapping and record-based EIS organization. The One-Pass algorithm operates using two lists, the first is the list of input references and the second is a list of EIS. In all of the following algorithm descriptions, each input reference can be thought of as a simple EIS so that essentially all of the comparisons are EIS-to-EIS comparisons. If at the beginning of the algorithm there are existing EIS being maintained by the EIIM system, these existing EIS are used to initially populate the EIS list, otherwise the EIS list is empty at the start of the algorithm.

In the One-Pass algorithm, the first reference in the input list is compared to every structure in the EIS list. Because One-Pass uses record-based rule mapping and record-based EIS organization, then comparing an input reference to each EIS in the EIS list is in effect comparing that input reference to all of the other input references previously processed. When an input record resolves to one of the references in an EIS, the EIS is flagged. After the input reference has been compared to every EIS in the EIS list, and at least one EIS has been flagged, then the input reference is integrated into the first flagged EIS. In the case that more than one EIS was flagged, the next flagged EIS is integrated into the EIS and so on until the input reference and all of the flagged EIS have been integrated into a single EIS.

On the other hand, if after all of the comparisons no EIS was flagged, then the input reference is used to create a new EIS that is appended to the end of the EIS list. In either case, the input reference is removed from the input list, and the next input reference in the input list is processed. This continues until the input list is empty and the process ends.

When the rule mapping and EIS are both record-based, each EIS at the end of the One-Pass algorithm will represent the transitive closure of the references with respect to record-to-record matching as expressed in the equivalence rules. In other words, there will exist an ordering of the input references comprising each EIS in which each pair of consecutive references match each other by one the equivalence rules. This is because in the course of the One-Pass algorithm, when an input reference matches a reference in two different EIS, all of the references in both EIS are deemed equivalent and are integrated into a single EIS thereby creating their transitive closure. An input reference that matches references in two different EIS and causes them to be integrated is sometimes called a "glue record."

If the One-Pass is executed on N input references, and there are no pre-existing EIS (representing previously processed references) in the EIS List, then there are no comparisons possible for the first reference, it is simply made into an EIS and moved into the EIS list. This leaves N-1 references in the input list. The next reference in the input list has one comparison with the reference that was moved to the EIS list. The third reference in the input list is compared to the two references that are in the EIS list regardless of whether these two references were found to be equivalent and integrated into the same EIS or whether they formed two distinct EIS. The fourth input will then be compared to the three previous and so on. Without considering blocking, indexing, or other methods for reducing comparisons, it is easy to see that the number of comparison required by the One-Pass algorithm will be

$$\frac{N\cdot(N-1)}{2}$$

A more general model for ER is the Stanford Entity Resolution Framework [14]. In the SERF Model, generic ER is defined in terms of two functions that operate on a set of input references R. The match function (M) determines when two references are equivalent. The merge function (μ) operates on two matching references and combines them into a new object that has the matching properties of both references. The merge function essentially plays the part of the integration process in EIIM except that the SERF model does not explicitly define the data structure for an EIS, only its behavior.

Both the match and merge functions can operate on the newly created merge objects as well as the original set of references. The strength of the model is that is can be used to formally define the entity resolution of a set of references R (designated as ER(R)) and also the properties that the functions M and μ must have in order for ER(R) to exist and to be finite and unique, a characteristic called "consistent ER". An important consequence of the SERF definition of ER with respect to EIIM is that it does not require either the equivalence rule mapping or EIS organization to be record-based. Equivalence rule mappings and EIS that are not record-based can meet the conditions of consistent ER. A primary example being the attribute-based rule mapping and attribute-based EIS organization described in the previous section. In the case that the equivalence rule mappings that are not record-based, then the EIS in the final ER(R) result will not necessarily represent the transitive closure the input references with respect to the matching as was true in the One-Pass algorithm. In addition, the application of the One-Pass algorithm to R using non-record-based rule mappings will not always lead to a unique ER(R). Fortunately the SERF Model also describes a resolution algorithm called R-Swoosh that always results in a unique ER(R) when match and merge function have the consistent ER properties.

As with One-Pass, the R-Swoosh algorithm begins with two lists, an input list and an output list. The input list is pre-populated with the EIS created by converting each input reference into a simple EIS. Because the SERF model does not address EIIM or EIS directly, the original description of the R-Swoosh always assumes that the output list is empty at the start. However as described in the One-Pass algorithm, the R-Swoosh algorithm will still work correctly if at the beginning of the algorithm, the output list is pre-populated with any pre-existing (managed) EIS.

Each iteration of R-Swoosh starts by selecting the first EIS in the input list and comparing it in order to each EIS in the output list. But unlike the One-Pass, as soon as the EIS from the input list is found to be equivalent to an EIS in the output list, the two EIS are integrated and removed from their respective lists, and the integrated (merged) EIS is appended to the end of the input list. After this, the next EIS in the input list is selected and the next iteration of processing begins.

In the case that the first EIS in the input list does not match any EIS in the output list, the input EIS is simply appended to the end of the output list. The algorithm ends when the input list is empty.

The main difference between One-Pass and R-Swoosh is that when an EIS from the input list is integrated with an EIS in the output list, the integrated EIS is moved to the input list. The reason for this is that under a non-record-based rule mapping, the newly integrated EIS may be equivalent to a previously processed EIS. Therefore the newly created EIS cannot be left in the output list, but must be returned to the input list so that it can be re-processed. With record-based equivalence rules this will not happen and the integrated record-based EIS can be left in the output list.

The number of comparisons necessary to complete R-Swoosh will depend upon how often and at what point in the list an integration event occurs. For N input references and J equivalence classes in ER(R), the maximum number of comparisons required for R-Swoosh is given by

$$(N-1)^2-\frac{(J-1)(J-2)}{2}$$

In the case that no two input references are equivalent (meaning that no integrations event will occur), then R-Swoosh would proceed the same as the One-Pass algorithm. In this situation there would be no integrated structures to append to the end of the input list and R-Swoosh would finish in one pass of

the input file. Since every reference would form its own EIS, then there would be N equivalence classes, i.e. when J = N the number of comparisons for R-Swoosh and One-Pass are the same.

It is also easy to see that for the same set of input references and the same equivalence rules, the number of equivalence classes given by an attribute-based mapping will be less than the number of equivalence classes given by a record-based mapping. The reason is simple. An attribute-based mapping will always include every record-based combination, but may also include other cross-record combinations that can be used by the equivalence rules. Consequently attribute-based mappings will find all of the equivalences that could arise from a record-based mapping, but may find additional equivalences based on the cross-record combinations.

Scenarios

Some very simple examples based on students in a school system can be used to construct several EIIM scenarios that illustrate the concepts that have been discussed. All of the examples will use the same set of three input references as given in Table 1. Each reference in Table 1 has four attributes, a unique reference identifier (RefNbr), the student's first name (First), the student's last name (Last), and the identifier assigned to a student by one the schools in the system that he or she attended (SID)

Table 1: Input References

RefNbr	First	Last	SID
1	John	Doe	G45
2	John	Doel	H37
3	Jon	Doe	H37

All of the examples will also use the same set of identity equivalence rules. Table 2 defines two equivalence rules to be used in the resolution process. The first rule states that the ER process will consider two references (or EIS) to be equivalent if they have exactly the same first and last name. The second rule states that they are considered equivalent if they have the same school identifier regardless of the name values. Also there is an implied OR relationship between the two rules meaning that two structures need only satisfy one or the other rule and not necessarily both in order to be considered equivalent.

Table 2: Equivalent Rules

Rule	First	Last	SID
1	Exact	Exact	*
2	*	*	Exact

Scenario 1: One-Pass, Record-based Mapping and EIS

In the One-Pass algorithm, the first reference is moved from the input list into the EIS List. Next the second reference is compared to the first reference. In this case neither rule is satisfied so the second reference is appended as a new EIS to the end of the EIS List. Finally the third reference is compared to the two references in the EIS List. When comparing the third reference to the first, neither rule is satisfied, but when comparing the third reference to the second, Rule 2 is satisfied so the second and third references are integrated and the algorithm is finished. The resulting EIS are as follows

EIS_1 = {{(First, John), (Last, Doe), (SID, G45)}}

EIS_2 = {{(First, John), (Last, Doel), (SID, H37)}, {(First, Jon), (Last, Doe), (SID, H37)}}

Scenario 2: R-Swoosh, Attribute-based Mapping and EIS

In the R-Swoosh algorithm, the three input references are all converted to simple EIS and placed in the input list. The first input EIS is moved the output List. Next the second input EIS is compared to the first EIS in the output list. In this case neither rule is satisfied so the second input EIS is appended to the end

of the output list. Next the third input EIS is compared to the EIS in the output list. When comparing the third input EIS to the first EIS in the output list, neither rule is satisfied. However, when it is compared to the second EIS in the output list, Rule 2 is satisfied.

Until this point the R-Swoosh and One-Pass have operated in the same way. However at the point the third input EIS resolves to the second EIS in the output list, the integrated EIS is moved back to the input list. Therefore, unlike the One-Pass algorithm, the R-Swoosh algorithm is not finished. The EIS formed by the integration of the second and third input EIS and put back into the input list must now be compared to the one remaining EIS in the output list. When these are compared Rule 1 is satisfied because under the attribute-based mapping the value "John" is in the list of values for attribute First in the integrated EIS and the value "Doe" is in the list of values for attribute Last. Consequently the end result is the following single integrated EIS

EIS_1 = {{(First, John), (First, Jon)}, {(Last, Doe), (Last, Doel)},
{(SID, G45), (SID, H37)}}

Scenario 3: One-Pass, Attribute-based Mapping and EIS

For the order of input references as given in Table 2, the One-Pass algorithm with attribute-based mapping and EIS will run to completion and will give the same result as in Scenario 1, two EIS given by

EIS_1 = {{(First, John), (Last, Doe), (SID, G45)}}
EIS_2 = {{(First, John), (Last, Doel), (SID, H37)}, {(First, Jon), (Last, Doe), (SID, H37)}}

However, if the order of processing is reversed so that references in Table 2 are processed in the order 3, 2, 1, then the One-Pass algorithm with attribute-based mapping and EIS will give the same result as R-Swoosh in Scenario 2 which is also the correct result of

EIS_1 = {{(First, John), (First, Jon)}, {(Last, Doe), (Last, Doel)},
{(SID, H37), (SID, H37)}}

So the problem is not that One-Pass cannot use an attribute mapping, the problem is that it will give different results depending upon the order of processing.

Scenario 4: R-Swoosh, Record-based Mapping and EIS

If Scenario 2 is changed so that R-Swoosh uses a record-based mapping and EIS instead of an attribute-based mapping and EIS, then the One-Pass and R-Swoosh will arrive at the same result as shown in Scenario 1. In this case then the R-Swoosh algorithm will proceed exactly as described in Scenario 2 except that when the EIS created by integrating the second and third input reference is compared to the third reference, neither rule will be satisfied under the record-based mapping. Consequently the two structures will not be integrated and the result is the same as in Scenario 1 namely

EIS_1 = {{(First, John), (Last, Doe), (SID, G45)}}
EIS_2 = {{(First, John), (Last, Doel), (SID, H37)}, {(First, Jon), (Last, Doe), (SID, H37)}}

Scenario 5: R-Swoosh, Attribute-based Mapping, and Record-based EIS

As noted earlier, record-based EIS can support attribute-based rule mapping. If Scenario 2 above is changed so that the EIS are record-based, but the mapping remains attribute-based, then the end result will still be a single equivalence class, the only difference will be in its structure. In this case single integrated EIS would appear as

EIS_1 = {{(First, John), (Last, Doe), (SID, G45)},
{(First, John), (Last, Doel), (SID, H37)},
{(First, Jon), (Last, Doe), (SID, H37)}}

By creating all of the combinations of the two algorithms (One-Pass and R-Swoosh), two equivalence rule mappings (record-based and attribute-based), and two EIS organizations (also record-based and attribute-based), eight possible scenarios are possible. However some of these combinations are not valid and should be avoided in the design of an EIS. Table 3 gives a summary of these combinations and their outcomes.

Table 3: Summary of EIIM Scenarios

Scenario	Algorithm	Rule Mapping	EIS Organization	Valid	ER(R)
1	One-Pass	Record-based	Record-based	Yes	N Classes
2	R-Swoosh	Attribute-based	Attribute-based	Yes	M Classes where M ≤ N
3	One-Pass	Attribute-based	Attribute-based	No	Varies by Order of Input
4	R-Swoosh	Record-based	Record-based	Yes	N Classes
5	R-Swoosh	Attribute-based	Record-based	Yes	M Classes where M ≤ N
6	One-Pass	Record-based	Attribute-based	No	*
7	One-Pass	Attribute-based	Record-based	No	Varies by Order of Input
8	R-Swoosh	Record-based	Attribute-based	No	*

ASSERTED RESOLUTION TO SUPPORT EIIM

Asserted resolution is a fourth method of resolution in an ER system that can be an important tool in support of EIIM. Shown as Process 9 in Figure 2, asserted resolution brings external knowledge from trusted information sources uses it to force the equivalence (or non-equivalence) of entity references and identity structures even though none of the equivalence rules may have been satisfied. Assertion is primarily a manual process that allows users to fine-tune the ER process without tampering with the equivalence rules. The trusted information used for assertion is often self-reported, directly observed, or obtained from public records or reliable commercial data providers, such as magazine subscription services.

Five specific forms of assertion have proposed to support EIIM processes [15]. They are reference-to-reference assertion, reference-to-structure assertion, structure integration assertion, structure split assertion, and negative structure-to-structure assertion.

- Reference-to-reference assertion. Two or more input references are asserted as equivalent and form a new EIS. Reference-to-reference assertion can be used to create an initial set of EIS when the true equivalences among the references is already known. For example, the references have been exported from another system in which entity identity integrity has been maintained. In other cases it may be used address a single exception where two reference are known to be equivalent, but would not be found equivalent by the system's equivalence rules. Assertion forces equivalence between the references without changing the equivalence rules.
- Reference-to-structure assertion. Basically the same as reference-to-reference but is used when there is already an existing EIS built by a previous EIIM process. Reference-to-structure forces one or more input references to be equivalent to (and integrated into) a specific EIS.
- Structure integration assertion. Two or more existing EIS are asserted as equivalent and integrated into a new EIS. As with reference-to-reference assertion, structure integration assertion can be used to address a small number of false negative exceptions where two structures are known to be equivalent references, but would not be found equivalent by the system's equivalence rules. Structure integration assertion would force the integration of the EIS without making changes to the rules.
- Structure split assertion. This type of assertion addresses the false positive resolution issues where references have been over consolidated, i.e. non-equivalent references are known to have been integrated into the same EIS. The structure split assertion allows the user to disassociate the references (split) into two or more EIS. Structure split assertion is most easily effected in EIS that have record-based organization that allows the user to identify and trace back the individual input references that comprise the EIS to their original sources. EIS with attribute-based organization in which record integrity has been lost are more difficult to untangle.
- Negative structure-to-structure assertion. This is used to prevent equivalence rules from integrating two EIS that are known to be non-equivalent. The two structures reference each other

and act as an override on the equivalence rules. Negative structure-to structure assertion is often used in conjunction with structure split assertion to prevent split structures from being re-integrated in later processes.

The purpose of these assertions is to provide ways to supplement the knowledge inferred from the content of the input sources with external knowledge asserted by other trusted sources of information. Asserted resolution plays an important role in the long-term management of persistent entity identification by allowing users to selectively override equivalence rules.

EIIM IN OYSTER

OYSTER (Open sYSTem Entity Resolution) is an ER system developed by the ERIQ Research Center at the University of Arkansas at Little Rock (ualr.edu/eriq). OYSTER provides access to a variety of entity resolution algorithms that enables users to uncover duplicate and redundant entity references. In addition, OYSTER also provides support for EIIM and persistent entity identifiers.

The configuration for each OYSTER run including attribute descriptions, equivalence rules, records layouts, and file locations is determined by user-created XML scripts that are interpreted at run-time [16]. At this writing, the latest version is OYSTER v3.1, and the source code and documentation are available from ERIQ website (ualr.edu/eriq/downloads).

OYSTER can be configured to implement four basic ER architectures: merge-purge, identity resolution, identity capture, and identity update. Merge-purge, also known as record linking, is perhaps the most common form of ER where a large number of entity references are systematically compared to each other and partitioned into clusters (subsets) of equivalent records. In OYSTER this partitioning of the input records is represented as a link index, a CSV output file with three attributes: RecID, OysterID, and Rule, as shown in Figure 4.

```
RecID, OysterID, Rule
R1, QUC0BJS2IA3KPGDN, [1]
R2, QUC0BJS2IA3KPGDN, [1]
R3, WZ3TB04QZJOGXU5G, null
```

Figure 4: Example OYSTER Link Index

In the link index the RefID is an identifier for each input reference formed by combining the source name (given the XML Source Descriptor) with the record sequence number given in the source file. OysterID is the identifier created and assigned by OYSTER during resolution process. Equivalent references are assigned the same identifier which is unique string of 16 characters. In Figure 4, there are two different values of OysterID which means that OYSTER partitioned the three input records into two cluster, {R1, R2} and {R3}. The Rule attribute shows which rule is used to create the link, e.g. R1 and R2 are clustered based on Rule 1. No rule was satisfied by R3 causing it to create a separate cluster.

Identity resolution is the form of ER in which the input references are resolved against a predefined set of managed identities represented as identity structures. Each identity structure in an identity resolution system has a fixed identifier that can be used to link references that are equivalent to the identity, thus creating a persistent identifier. When OYSTER runs in identity resolution mode, it will produce a link index similar to that produced in merge-purge mode. The difference is that OysterID value in the link index is the identifier of the managed identities to which the input reference is equivalent. In the case that the input reference does not resolve to any of the managed identities, the OysterID value for that input reference is given as a string of 16 “X” characters to indicate that no resolution was found.

Identity capture is another form of ER in which the system builds (learns) a set of identities from the references it processes. Identity capture is essentially a “smart” version of merge-purge or record linking in which the knowledge gained in resolving the references is retained in the form of EIS.

Identity capture and identity resolution work hand-in-hand in the EIIM process. Identity capture mode is used to create or update identity structures from trusted reference sources. Identity resolution mode would be used when the objective is to identify the entities represent in a set of input records (transactions), but not to capture their identity information. For example, identity capture might be used to build student identities from school enrollment records, while identity resolution might be used to process examination records to identify the student taking each exam by appending his or her persistent entity identifier to the examination record.

Identity update is a hybrid of identity capture and identity resolution. It starts with existing identities, but can create new identities and update existing identities with information from the references being processed. Identity capture, identity resolution, and identity update configurations serialize their identity structures at the end of the resolution process as XML documents.

Regardless of the type of ER configuration, it is clear that the final resolution result will depend upon equivalence rules and the EIIM scenario context. OYSTER 3.1 supports EIIM Scenarios 1 and 5 shown in Table 3. Both Scenarios 1 and 5 implement record-based EIS organization. Scenario 1 is the One-Pass algorithm and record-based mapping. The keyword used in the OYSTER Run Script to setup Scenario 1 is EREngine= “FSCluster”. The Run Script keyword for Scenario 5 to is EREngine=“RSwooshStandard” or EREngine= “RSwooshEnhanced” [17].

Identity Capture with OYSTER

Identity Capture mode in OYSTER is set by given input references and equivalence rules, specified the destination paths for output identities and link index. The input references used in this example are from ListOne.txt, a synthetic test file that can be found on the ERIQ website (ualr.edu/eriq/downloads). ListOne is a text file of 284 references, each with five, comma-delimited attributes. A segment of records in ListOne is shown in Figure 5. The attribute values for Field 1, 2, 3, and 4 are single letters from “A” to “G”.

```
RefID, Field1, Field2, Field3, Field4
...
175,F,F,B,D
176,D,D,F,B
177,E,D,A,D
...
```

Figure 5: Segment of Input References from ListOne.txt

ListOne is used to demonstrate how dramatic the difference produced by attribute-based and record-base rule mappings can be. In this experiment, two OYSTER runs are performed, Run1 using attribute-based rule mapping and Run2 using record-based rule mapping. Both Runs 1 and 2 use ListOne as the input reference source, and both use the same equivalence rules. The equivalence rules are that two references are equivalent if any three out of the four attributes are the same. Run1 is set to use ER engine “RSwooshStandard” and Run 2 is set to use ER engine “FSCluster”. The results are that Run 1 generates 1 cluster while Run 2 generates 15 clusters from the same set of 284 input references.

Figures 6 and 7 show segments of the outputs from Run 1 where attribute-based rule mapping and the R-Swoosh algorithm have been used. A segment of the record-based identity structure that was created by is shown in Figure 6. The link index produced from Run 1 is shown in Figure 7. Note that in this case, all the input references comprise a single identity structure labeled with the identifier “I7A8C95ZZYSWYWM7”.

Notice that the XML document shown in Figure 6, has a <Metadata> section in addition to the <Identities> block. The main component of <Metadata> is <Attributes> where each attribute is assigned an upper-case letter for storage compression. For example “Field 2” is represented by letter “B”. Each

element <Identity> sub-block in the <Identities> block describes one OYSTER EIS. In Figure 6 there is only one identity structure with identifier " I7A8C95ZZYSWYWM7". There are 284 references in this structure which are written in <Reference> sub-block. The record values are stored as attribute-value pairs where the attribute tag is separated from the attribute value by the caret character (^). Value pairs are separated by the pipe character (|). For example the <Reference> with the value " A^L1.175|B^F|C^B|D^D|E^F " corresponds to reference 175 of ListOne in Figure 5. This tagged format follows CoDoSA XML framework [18].

Figure 8 shows the link index from Run 2 that implement record-based rule mapping and One-Pass algorithm. OYSTER creates 15 clusters out of 284 input references in this run. Each cluster is assigned with a unique OysterID. For example, in Figure 8, reference 175 and 176 are in one cluster with identifier “I7A8C95ZZYSWYWM7” and reference 177 is in another cluster with identifier “JKDAOZ8BX4ZJ986V”. The EIS from Run 2 is saved as same format as Figure 6 with 15 clusters in total.

```
<root>
<Metadata>
…
        <Attributes>
                <Attribute Name="@RefID" Tag="A"/>
                <Attribute Name="Field2" Tag="B"/>
                <Attribute Name="Field3" Tag="C"/>
                <Attribute Name="Field4" Tag="D"/>
                <Attribute Name="Field1" Tag="E"/>
        </Attributes>
<Metadata>
<Identities>
        <Identity Identifier=" I7A8C95ZZYSWYWM7"  CDate="2011-06-21">
                <References>
                        …
                        <Reference Value="A^L1.175|B^F|C^B|D^D|E^F"/>
                        <Reference Value="A^L1.176|B^D|C^F|D^B|E^D"/>
                        <Reference Value="A^L1.177|B^D|C^A|D^D|E^E"/>
                        …
        </References>
        </Identity>
</Identities>
</root>
```

Figure 6: A Segment of the Single EIS from ListOne using Attribute-based Rule Mapping

RefID	OysterID,	Rule
…		
L1.175	I7A8C95ZZYSWYWM7	[2, 4, 1]
L1.176	I7A8C95ZZYSWYWM7	[1, 3, 4]
L1.177	I7A8C95ZZYSWYWM7	[2, 1]
...		

Figure 7: A Segment of the Link Index for ListOne using Attribute-based Rule Mapping

RefID	OysterID	Rule
…		
L1.175	I7A8C95ZZYSWYWM7	[1]
L1.176	I7A8C95ZZYSWYWM7	[1]
L1.177	JKDAOZ8BX4ZJ986V	null
…		

Figure 8: Link Index of Run 2 Produced by OYSTER 3.1 (Type="Record-based rule mapping")

Reference-to-Reference Assertion in OYSTER

OYSTER version 3.1 supports reference-to-reference assertion logic and support for the other types of assertion described in the previous section are in currently in development for release in later versions of the system.

Consider the following situation. There is a master record (MR) which includes student records from School 1 (S1) and School 2 (S2) as shown in Figure 9. With asserted information from calling to schools, OYSTER can build the identity knowledgebase from the asserted input references.

RefID, FirstName, LastName, DOB, SCode, Assert
S1.1, Edgar,Jones, 20001104, G34, 1
S2.1, Eddie, Jones, 20001104, H15, 1
S1.2, Mary, Smith, 19990921, G34, 2
S2.2, John, Doe, 20010806, H15, 3

Figure 9: References with Assertion Attributes

In Figure 9, there are six attributes: Reference ID (RefID), First Name, Last Name, Date of Birth (DOB), School Code and Assert. The RefID value "S1.1" indicates Reference 1 from School 1 and "S2.1" indicates Reference 1 from School 2. The attribute Assert is used to label equivalent references with the same Assert value, e.g. reference S1.1 and S2.1 are tagged by Assert value "1". OYSTER uses the keyword "@RelAssert" in the source descriptor to specify the asserted input.

OYSTER Run 3 performs the asserted resolution using the input references shown in Figure 9. The resulting Link Index from Run 3 is shown as Figure 10. Based on the asserted input, reference S1.1 and S2.1 from MR are assigned the same OysterID value of "GTZ519RT7V0B8GRG" whereas S1.2 is assigned a different OysterID. The corresponding EIS is built in a similar format as shown in Figure 6.

RefID	OysterID	Rule
MR.S1.1	GTZ519RT7V0B8GRG	[assert]
MR.S2.1	GTZ519RT7V0B8GRG	[assert]
MR.S1.2	KXLUVF7B93U0N5TW	null
MR.S2.2	SU4LQWH0EOVWX5WS	null

Figure 10: Run 3 Link Index Showing OYSTER Reference-to-Reference Assertion

CONCLUSION

The ability to create and maintain persistent entity identity structures and identifiers over time is a critical process for master data management (MDM) and in the rapidly growing application of entity-based information exchange systems. However, the example scenarios of Table 3 make it clear that a combination of factors must be considered in the design of ER systems that attempt to support EIIM. It is also clear that more research and experimentation needs to be done to fully understand and recommend best practices for EIIM. Some of the research questions that need to be explored include:

- A complete formulation of the conditions that must exist for attributes and equivalence rules in order to accurately predict when attribute-based and record-based rule mappings will produce the same or different resolution results.

- For which EIIM applications does attribute-based resolution provide better results than record-based resolution and vice versa?
- The question of whether there are other rule-mappings and EIS organization schemes other than attribute-based and rule-based that could be useful in a particular EIIM application context.
- Experimental results and case studies on the effectiveness of asserted resolution in support of effective EIIM. Are the five proposed form of assertion sufficient? Are other forms needed?

ACKNOWLEDGEMENTS

This research is partially supported by grants from the Arkansas Department of Education and the US Air Force Research Laboratory at Wright Patterson Air Force Base in Dayton, Ohio. Special thanks to Dr. James Zaiss at the Center for Identity at University of Texas at Austin for his review and suggestions.

REFERENCES

[1] Maydanchik, A.. *Data Quality Assessment.* Bradley Beach : Technics Publications, 2007.

[2] Huang, K., Lee, Y. W., and Wang, R. Y.. *Quality Information and Knowledge Management.* Prentice Hall, 1999.

[3] The Center for Identity, UT Austin. Shifting Organizational changes in Identity Management Practices. [Online] 2011. http://identity.utexas.edu/media/stdocs/White_Paper-Organizations_and_IdM.pdf.

[4] Josang, A., and Pope, S.. User Centric Identity Management . *Proceedings: AusCERT Conference*. 2005.

[5] Slone, S.. *Identity Mangement.* The Open Group Identity Management Work Area. 2004.

[6] Center for Identity Risk Model Research. The Center for Identity Risk Assessment Model (CIRAM). [Online] 2011. http://identity.utexas.edu/research/model.

[7] Talburt, J. R.. *Entity Resolution and Information Quality.* Burlington : Morgan Kaufmann, 2011.

[8] Dyché, J., and Levy, E.. *Customer data integration: Reaching a single version of the truth.* New York : Wiley, 2006.

[9] Bilenko, M., Basu, S., and Sahami, M.. Adaptive product normalization: Using online learning for record linkage in comparison shopping. *Proceedings: ICDM-2005.* pp. 58-65.

[10] Hsiung, P., Moore, A., Neill, D. and Schneider, J.. Alias Detection in Link Data Sets. *Proceedings: International Conference on Intelligence Analysis.* 2004.

[11] Lim, E. P., Srivastava, J., Prabhakar, S., and Richardson J.. Entity identification in database integration. *Proceedings: Ninth International Conference on Data Engineering.* 1993. pp. 294-301.

[12] Loshin, D.. *Master Data Management.* Morgan Kaufmann, 2008.

[13] Blake, R., and Mangiameli, P.. The Effects and Interactions of Data Quality and Problem Complexity on Data Mining. *Proceedings: International Conference on Information Quality (ICIQ*). 2008. pp. 160-175.

[14] Benjelloun, O., Garcia-Molina, H., Menestrina, D., Su, Q., Whang, S.E., and Widom, J.. Swoosh: a generic approach to entity resolution. *The International Journal on Very Large Data Bases, Vol. 18*, 2009. pp. 255-276.

[15] Zhou, Y., and Talburt, J.. The Role of Asserted Resolution in Entity Identity Management. *Proceedings: The 2011 International Conference on Information and Knowledge Engineering (IKE'11).* 2011. pp. 291-296.

[16] Zhou, Y., Talburt, J., Su, Y., and Yin, L.. OYSTER: A Tool for Entity Resolution in Health Information Exchange. *Proceedings: The 5th International Conference on the Cooperation and Promotion of Information Resources in Science and Technology (COINFO"10).* 2010. pp. 356-362.

[17] Nelson, E., and Talburt, J.. Entity Resolution for Longitudinal Studies in Education using OYSTER. *Proceedings: The 2011 International Conference on Information and Knowledge Engineering (IKE'11).* 2011. pp. 286-290

[18] Talburt, J., and Nelson, E.. CoDoSA: A light-weight, XML framework for integrating unstructured textual information. *Proceedings: 15th Americas Conference on Information Systems.* 2009. pp. 489.