The NESTOR Framework: Manage, Access and Exchange Hierarchical Data Structures

Maristella Agosti, Nicola Ferro, and Gianmaria Silvello

Department of Information Engineering, University of Padua, Italy
{agosti, ferro, silvello}@dei.unipd.it

Abstract. In this paper we study the problem of representing, managing and exchanging hierarchically structured data in the context of a *Digital Library (DL)*. We present the *NEsted SeTs for Object hieRarchies (NESTOR)* framework defining two set data models that we call: the "Nested Set Model (NS-M)" and the "Inverse Nested Set Model (INS-M)" based on the organization of nested sets which enable the representation of hierarchical data structures. We present the mapping between the tree data structure to NS-M and to INS-M. Furthermore, we shall show how these set data models can be used in conjunction with *Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH)* adding new functionalities to the protocol without any change to its basic functioning. At the end we shall present how the couple OAI-PMH and the set data models can be used to represent and exchange archival metadata in a distributed environment.

1 Introduction

In *Digital Library Systems (DLSs)* objects are often organized in hierarchies to help in representing, managing or browsing them. For instance, books in a library can be classified by author and then by subject and then by publishing house. Documents in an archive are organized in a hierarchy divided into fonds, sub-fonds, series, sub-series and so on. In the same way the internal structure of an object can be hierarchical; for example the structure of a book organized in chapters, sections and subsections or a web page composed by nested elements such as body, titles, subtitles, paragraphs and subparagraphs. One very important tool extensively adopted to represent digital objects such as metadata, text documents and multimedia contents — the *eXtensible Markup Language (XML)* — has an intrinsically hierarchical structure.

Representing, managing, preserving and sharing efficiently and effectively the hierarchical structures is a key point for the development and the consolidation of DLS technology and services. In this paper we propose the *NEsted SeTs for Object hieRarchies (NESTOR)* framework defining two set data models that we call: the "Nested Set Model (NS-M)" and the "Inverse Nested Set Model (INS-M)" [7]. These models are defined in the context of the ZFC (Zermelo-Fraenkel with the axiom of Choice) axiomatic set theory [5], exploiting the advantages of the use of sets in place of a tree structure. The foundational idea behind these set data models is that an opportune set organization can maintain all the features of a tree data structure with the addition of some new relevant functionalities. We define these functionalities in terms of flexibility of the model, rapid selection and isolation of easily specified subsets of data and extraction of only those data necessary to satisfy specific needs.

Furthermore, these set data models can work in conjunction with the *Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH)* [12] that is the standard *de-facto* for metadata sharing between DLSs in distributed environments. The extension of OAI-PMH with these set data models allows the protocol to manage and exchange complex hierarchical data structure in a flexible way. The extension of OAI-PMH shall permit the exchange of data belonging to a hierarchy with a variable granularity without losing the relationships with the other data in the hierarchy. Furthermore, the OAI-set which is a constituent part of the protocol will be used also to organize the data and not only to enable the selective harvesting. A concrete use case is the archival data that are organized in a hierarchy which preserve the meaningful relationships between the data [6]. When an archival object is shared it has to preserve all the relationships with the preservation context and with the other objects in the archive; since the use of tree data structure in this context turns out to be problematic in terms of accessibility and flexibility, we shall show that the use of the proposed data models in conjunction with OAI-PMH overcomes many of these issues [6].

The paper is organized as follows: Section 2 briefly defines the tree data structure. Section 3 defines the NESTOR framework based on the two proposed set data models and presents the mapping functions between the tree data structure and the set data models. Section 4 describes how OAI-PMH can extend its functionalities by exploiting the NS-M or the INS-M; moreover this section presents a use case in which the set data models and OAI-PMH can be used together to exchange full expressive archival metadata. Section 5 draws some conclusions.

2 The Tree Data Structure

The most common and diffuse way to represent a hierarchy is the tree data structure, which is one of the most important non-linear data structures in computer science [9]. We define a tree as $T(V, E)$ where V is the set of nodes and E the set of edges connecting the nodes. V is composed by n nodes $V = \{v_1, \dots, v_n\}$ and E is composed by $n-1$ edges. If $v_i, v_j \in V$ and if $e_{ij} \in E$ then e_{ij} is the edge connecting v_i to v_j, thus v_i is the parent of v_j. We indicate with $d_V^-(v_i)$ the **inbound degree** of node $v_i \in V$ representing the number of its inbound edges; $d_V^+(v_i)$ is the **outbound degree** of $v_i \in V$ representing the number of its outbound edges. $v_r \in V$ is defined to be the **root** of $T(V, E)$ if and only if $d_V^-(v_r) = 0$; $\forall v_i \in V \setminus \{v_r\}$, $d_V^-(v_i) = 1$. The set of all **external nodes** is $V_{ext} = \{v_i : d_V^+(v_i) = 0\}$ and the set of all the **internal nodes** is $V_{int} = \{v_i : d_V^+(v_i) > 0\}$.

We define with $\Gamma_V^+(v_i)$ the set of **all the descendants** of v_i in V (including v_i ifself); vice versa $\Gamma_V^-(v_i)$ is the set of **all the ancestors** of v_i in V (including

v_i ifself). We shall use the set Γ in the following of this work, so it is worth underlining a couple of recurrent cases. Let $v_r \in V$ be the root of a tree $T(V, E)$ then $\Gamma^-_V(v_r) = \{v_r\}$ and $\Gamma^+_V(v_r) = V$. Furthermore, let v_i an external node of $T(V, E)$, then $\Gamma^+_V(v_i) = \{v_i\}$.

3 The NESTOR Framework

The NESTOR framework is based on two set data models called *Nested Set Model* (NS-M) and *Inverse Nested Set Model* (INS-M) based on an organization of nested sets. The foundational idea behind these set data models is that an opportune set organization can maintain all the features of a tree data structure with the addition of some new relevant functionalities. We define these functionalities in terms of flexibility of the model, rapid selection and isolation of easily specified subsets of data and extraction of only those data necessary to satisfy specific needs.

The most intuitive way to understand how these models work is to relate them to the well-know tree data structure. Thus, we informally present the two data models by means of examples of mapping between them and a sample tree. The first model we present is the **Nested Set Model** (NS-M). The intuitive graphic representation of a tree as an organization of nested sets was used in [9] to show different ways to represent tree data structure and in [3] to explain an alternative way to solve recursive queries over trees in SQL language. An organization of sets in the NS-M is a collection of sets in which any pair of sets is either disjoint or one contains the other. In Figure 1 (b) we can see how a sample tree is mapped into an organization of nested sets based on the NS-M.

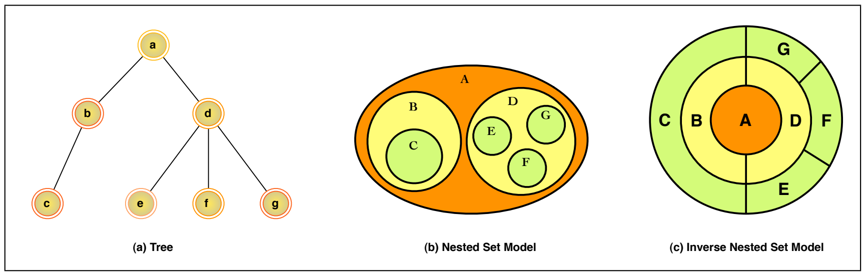


Fig. 1. (a) A tree. (b) Euler-Venn Diagram of a NS-M. (c) Doc-Ball representation of a INS-M.

From Figure 1 (b) we can see that each node of the tree is mapped into a set, where child nodes become *proper subsets* of the set created from the parent node. Every set is subset of at least of one set; the set corresponding to the tree root is the only set without any supersets and every set in the hierarchy is subset of the root set. The external nodes are sets with no subsets. The tree structure is maintained thanks to the nested organization and the relationships between the sets are expressed by the set inclusion order. Even the disjunction between two sets brings information; indeed, the disjunction of two sets means that these belong to two different branches of the same tree.

The second data model is the **Inverse Nested Set Model** (INS-M). We can say that a tree is mapped into the INS-M transforming each node into a set, where each parent node becomes a subset of the sets created from its children. The set created from the tree's root is the only set with no subsets and the root set is a proper subset of all the sets in the hierarchy. The leaves are the sets with no supersets and they are sets containing all the sets created from the nodes composing tree path from a leaf to the root. An important aspect of INS-M is that the intersection of every couple of sets obtained from two nodes is always a set representing a node in the tree. The intersection of all the sets in the INS-M is the set mapped from the root of the tree.

Differently from the NS-M, the representation of the INS-M by means of the Euler-Venn diagrams is not very expressive and can be confusing for the reader [1]. We can represent in a straightforward way the INS-M by means of the "*DocBall representation*" [4]. The DocBall representation is used in [4] to depict the structural components of the documents and can be considered as the representation of a tree structure. We exploit the DocBall ability to show the structure of an object and to represent the "*inclusion order of one or more elements in another one*" [14]. The DocBall is composed of a set of circular sectors arranged in concentric rings as shown in Figure 1 (c). In a DocBall each ring represents a level of the hierarchy with the center (level 0) representing the root. In a ring, the circular sectors represent the nodes in the corresponding level. We use the DocBall to represent the INS-M, thus for us each circular sector corresponds to a set.

In Figure 1 (c) we can see the INS-M mapping of a sample tree by means of the DocBall representation. The root "a" of the tree is mapped into the set "A" represented by the inner ring at level 0 of the DocBall; at level 1 we find the children of the root and so on. With this representation a subset is presented in a ring inner than the set including it. Indeed, we can see that the set A is included by all the other sets. If the intersection of two or more sets is empty then these sets have no common circular sector in the inner rings of the DocBall; in the INS-M this is not possible because the set representing the root (A) is common to all the sets in the INS-M. For instance, we can see that the circular sectors C and E have in common only A, indeed $C \cap E = A$; instead, G and E have in common the sectors D and A, thus $G \cap E = \{D, A\}$.

It is worthwhile for the rest of the work to define some basic concepts of set theory: the family of subsets and the subfamily of subsets, with reference to [5] for their treatment. However, we assume the reader is confident with the basic concepts of ZFC axiomatic set theory, which we cannot extensively treat here for space reasons.

Definition 1 *Let F be a set, I a non-empty set and $\mathcal{C}$ a collection of subsets of F. Then a bijective function $\mathcal{F} : I \longrightarrow \mathcal{C}$ is a **family** of subsets of F. We call I the **index** set and we say that the collection $\mathcal{C}$ is **indexed** by I.*

We use the following notation $\{F_i\}_{i \in I}$ to indicate the family $\mathcal{F}$; the notation $F_i \in \{F_i\}_{i \in I}$ means that $\exists\ i \in I \mid \mathcal{F}(i) = F_i$. We call **subfamily** of $\{F_i\}_{i \in I}$ the **restriction** of $\mathcal{F}$ to $J \subseteq I$ and we denote this with $\{SF_j\}_{j \in J} \subseteq \{F_i\}_{i \in I}$.

Definition 2 *Let $\{F_i\}_{i \in I}$ be a family. We define $\{F_i\}_{i \in I}$ to be a **linearly ordered family** (or **chain**) if $\forall F_j, F_k \in \{F_i\}_{i \in I}$, $F_j \subseteq F_k \vee F_k \subseteq F_j$.*

Furthermore, we can say that a family $\{F_i\}_{i \in I}$ is a linearly ordered family if every two sets in $\{F_i\}_{i \in I}$ are comparable: $\forall F_j, F_k \in \{F_i\}_{i \in I}, F_j \subset F_k \vee F_k \subset F_j$.

Definition 3 *Let $\{F_i\}_{i \in I}$ be a family . We define $\{F_i\}_{i \in I}$ to be a **topped family** if $\exists F_k \in \{F_i\}_{i \in I} \mid \forall F_j \in \{F_i\}_{i \in I}, F_j \subseteq F_k$. If $\nexists F_k \in \{F_i\}_{i \in I} \mid \forall F_j \in \{F_i\}_{i \in I}, F_j \subseteq F_k$ then $\{F_i\}_{i \in I}$ is defined to be a **topless family**.*

Definition 4 *Let F be a set and let $\{F_i\}_{i \in I}$ be a family. Then $\{F_i\}_{i \in I}$ is a **Nested Set** family if:*

$$F \in \{F_i\}_{i \in I}, \tag{3.1}$$

$$\emptyset \notin \{F_i\}_{i \in I}, \tag{3.2}$$

$$\begin{aligned} \forall F_h, F_k \in \{F_i\}_{i \in I}, h \neq k \mid F_h \cap F_k \neq \emptyset \\ \Rightarrow F_h \subset F_k \vee F_k \subset F_h. \end{aligned} \tag{3.3}$$

Thus, we define a Nested Set family (NS-F) as a family where three conditions must hold. The first condition (3.1) states that set A which contains all the sets in the family must belong to the NS-F. The second condition states that the empty-set does not belong to the NS-F and the last condition (3.3) states that the intersection of every couple of distinct sets in the NS-F is not the empty-set only if one set is a proper subset of the other one.

Theorem 1 *Let $T(V, E)$ be a tree and let Φ be a family where $I = V$ and $\forall v_i \in V$, $V_{v_i} = \Gamma_V^+(v_i)$. Then $\{V_{v_i}\}_{v_i \in V}$ is a Nested Set family.*

Proof. Let $v_r \in V$ be the root of the tree then $V_{v_r} = \Gamma_V^+(v_r) = V$ and thus $V \in \{V_{v_i}\}_{v_i}$ (condition 3.1). By definition of descendant set of a node, $\forall v_i \in V$, $|V_{v_i}| = |\Gamma_V^+(v_i)| \geq 1$ and so $\emptyset \notin \{V_{v_i}\}_{v_i \in V}$ (condition 3.2).

Now, we prove condition 3.3. Let $v_h, v_k \in V$, $h \neq k$ such that $V_{v_h} \cap V_{v_k} = \Gamma_V^+(v_h) \cap \Gamma_V^+(v_k) \neq \emptyset$, ab absurdo suppose that $\Gamma_V^+(v_h) \nsubseteq \Gamma_V^+(v_k)\ \wedge\ \Gamma_V^+(v_k) \nsubseteq \Gamma_V^+(v_k)$. This means that the descendants of v_h share at least a node with the descendants of v_k but they do not belong to the same subtree. This means that $\exists\ v_z \in V \mid d_V^-(v_z) = 2$ but then $T(V, E)$ is not a tree.

In the same way we can define the Inverse Nested Set Model (INS-M):

Definition 5 *Let F be a set, let $\{F_i\}_{i \in I}$ be a family and let $\{SF_j\}_{j \in J} \subseteq \{F_i\}_{i \in I}$ be a sub-family. Then $\{F_i\}_{i \in I}$ is an **Inverse Nested Set** family if:*

$$\emptyset \notin \{F_i\}_{i \in I}, \tag{3.4}$$

$$\bigcap_{j \in J} SF_j \in \{F_i\}_{i \in I}. \tag{3.5}$$

$$\begin{aligned}&\exists SF_k \in \{SF_j\}_{j\in J} \mid \forall SF_h \in \{SF_j\}_{j\in J}, SF_h \subseteq SF_k\\ &\Rightarrow \forall SF_h, SF_g \in \{SF_j\}_{j\in J}, SF_h \subseteq SF_g \vee SF_g \subseteq SF_h.\end{aligned} \tag{3.6}$$

Thus, we define an Inverse Nested Set family (INS-F) as a family where three conditions must hold. Condition 3.4 states that the empty-set does not belong to the INS-F. Condition 3.5 states that the intersection of every subfamily of the INS-F belongs to the INS-F itself. Condition 3.6 states that every subfamily of a INS-F can be a topped family only if it is linearly ordered; alternatively, we can say that every subfamily of an INS-F must be a topless family or it must be a chain.

Theorem 2 *Let $T(V,E)$ be a tree and let Ψ be a family where $I = V$ and $\forall v_i \in V$, $V_{v_i} = \Gamma_V^-(v_i)$. Then $\{V_{v_i}\}_{v_i\in V}$ is an Inverse Nested Set family.*

Proof. By definition of the set of the ancestors of a node, $\forall v_i \in V$, $|V_{v_i}| = |\Gamma_V^-(v_i)| \geq 1$ and so $\emptyset \notin \{V_{v_i}\}_{v_i\in V}$ (condition 3.4).

Let $\{B_{v_j}\}_{v_j\in J}$ be a subfamily of $\{V_{v_i}\}_{v_i\in V}$. We prove condition 3.5 by induction on the cardinality of J. $|J| = 1$ is the base case and it means that every subfamily $\{B_{v_j}\}_{v_j\in J} \subseteq \{V_{v_i}\}_{v_i\in V}$ is composed only by one set B_{v_1} whose intersection is the set itself and belongs to the family $\{V_{v_i}\}_{v_i\in V}$ by definition.

For $|J| = n-1$ we assume that $\exists\, v_{n-1} \in V \mid \bigcap_{v_j\in J} B_{v_j} = B_{v_{n-1}} \in \{V_{v_i}\}_{v_i\in V}$; equivalently we can say that $\exists\ v_{n-1} \in V \mid \bigcap_{v_j\in J} \Gamma_V^-(v_j) = \Gamma_V^-(v_{n-1})$, thus, $\Gamma_V^-(v_{n-1})$ is a set of nodes that is composed of common ancestors of the $n-1$ considered nodes.

For $|J| = n$, we have to show that $\exists\ v_t \in V \mid \forall\ v_n \in J,\ B_{v_{n-1}} \cap B_{v_n} = B_{v_t} \in \{V_{v_i}\}_{v_i\in V}$. This is equivalent to show that $\exists\ v_t \in V \mid \forall\ v_n \in J,\ \Gamma_V^-(v_{n-1}) \cap \Gamma_V^-(v_n) = \Gamma_V^-(v_t)$.

Ab absurdo suppose that $\exists\ v_n \in J \mid \forall\ v_t \in V,\ \Gamma_V^-(v_{n-1}) \cap \Gamma_V^-(v_n) \neq \Gamma_V^-(v_t)$. This would mean that v_n has no ancestors in J and, consequently, in V; at the same time, this would mean that v_n is an ancestor of no node in J and, consequently, in V. But this means that V is the set of nodes of a forest and not of a tree.

Now, we have to prove condition 3.6. Let $\{B_{v_j}\}_{v_j\in J}$ be a subfamily of $\{V_{v_i}\}_{v_i\in V}$. Ab absurdo suppose that $\exists B_{v_k} \in \{B_{v_j}\}_{v_j\in J} \mid \forall B_{v_h} \in \{B_{v_j}\}_{v_j\in J}, B_{v_h} \subseteq B_{v_k} \Rightarrow \exists B_{v_h}, B_{v_g} \in \{B_{v_j}\}_{v_j\in J} \mid B_{v_h} \nsubseteq B_{v_g} \wedge B_{v_g} \nsubseteq B_{V_h}$. This means that $\{B_{v_j}\}_{v_j\in J}$ is a topped but not linearly ordered family. This means that we can find $B_{v_g}, B_{v_h}, B_{v_k} \in \{B_{v_j}\}_{v_j\in J} \mid ((B_{v_h} \cap B_{v_k} \neq \emptyset) \wedge (B_{v_h} \cup B_{v_k} \subset B_{v_g}) \wedge (B_{v_h} \nsubseteq B_{v_k}) \wedge (B_{v_k} \nsubseteq B_{v_h})) \Rightarrow \exists v_h, v_k, v_g \in V \mid ((\Gamma_V^-(v_h) \cap \Gamma_V^-(v_k) \neq \emptyset) \wedge (\Gamma_V^-(v_h) \cup \Gamma_V^-(v_k) \subseteq \Gamma_V^-(v_g)) \wedge (\Gamma_V^-(v_h) \nsubseteq \Gamma_V^-(v_k)) \wedge (\Gamma_V^-(v_k) \subseteq \Gamma_V^-(v_h)))$. This means that there are two paths from the root of T to v_g, one throught v_h and a distinct one throught v_k, thus $d_V^-(v_g) = 2$ and so T is not a tree. □

4 How to Exploit the NESTOR Framework in Conjunction with OAI-PMH

The defined set data models can be exploited to improve the data exchange between DL systems in a distributed environment. Our aim is to show how

NESTOR enables OAI-PMH to cope with complex hierarchical structured objects without any losses in its basic features that are: flexibility, adaptability and non-invasivity. In order to explain how NESTOR can be used in conjunction with OAI-PMH it is worthwhile to describe the functioning of this protocol with a magnifying lens over the features particularly well-suited towards the integration of the NESTOR framework.

4.1 The Open Archive Initiative Protocol for Metadata Harvesting

OAI-PMH is based on the distinction between Data Provider and Service Provider which, respectively, offer metadata and harvest metadata to provide services [13]. Data Providers are the components that make metadata available to the Service Providers that harvest metadata. Each Data Provider manages its own metadata and it is independent and autonomous by the outside information systems. Service Provider role is to harvests metadata by the different Data Providers and to performs advanced services on these harvested metadata.

The protocol defines two kinds of harvesting procedures: incremental and selective harvesting. Incremental harvesting permits users to query a Data Provider and ask it to return just the new, changed or deleted records from a certain date or between two dates. Selective harvesting is based on the concept of *OAI-set*, which enables logical data partitioning by defining groups of records. Selective harvesting is the procedure that permits the harvesting only of metadata owned by a specified OAI-set. In OAI-PMH a set is defined by three components: `setSpec` which is mandatory and a unique identifier for the set within the repository, `setName` which is a mandatory short human-readable string naming the set, and `setDesc` which may hold community-specific XML-encoded data about the set.

OAI-set organization may be hierarchical, where hierarchy is expressed in the `setSpec` field by the use of a colon [:] separated list indicating the path from the root of the set hierarchy to the respective node. For example if we define an OAI-set whose `setSpec` is *"A"*, its subset "B" would have *"A:B"* as `setSpec`. In this case "B" is a proper subset of "A": $B \subset A$. When a repository defines a set organization it must *include set membership information in the headers of the records* returned to the harvester requests. Harvesting from a set which has sub-sets will cause the repository to return the records in the specified set and recursively to return the records from all the sub-sets. In our example, if we harvest set A, we also obtain the records in sub-set B [12].

4.2 Set-Theoretical Use of OAI-PMH

In OAI-PMH it is possible to define an OAI-set organization based on the NS-M or INS-M. This means that we can treat the OAI-sets as a Nested Set Family (NS-F) or as an Inverse Nested Set Family (INS-F). The inclusion order between the OAI-sets is given by its identifier which is a `<setspec>` value. This `<setspec>` value is also added in the header of every record belonging to an OAI-set. In the following we describe how it is possible to create a Nested Set family of

OAI-Set and afterward how the same thing can be done with an Inverse Nested Set family.

Let $\mathcal{O}$ be a Nested Set family and let I be the set of the `<setspec>` values where $i \in I = \{s_0 : s_1 : \ldots : s_j\}$ means that $\exists\ O_j \in \{O_i\}_{i \in I} \mid O_j \subset \ldots \subset O_1 \subset O_0$. Every $O_i \in \{O_i\}_{i \in I}$ is an OAI-set uniquely identified by a `<setspec>` value in I. The `<setspec>` values for the $O_k \in \{O_i\}_{i \in I}$ are settled in such a way to maintain the inclusion order between the sets. If an O_k has no superset its `setspec` value is composed only by a single value (`<setspec>`s_k`</setspec>`). Instead if a set O_h has supersets, e.g. O_a and O_b where $O_b \subset O_a$, its `setspec` value must be the combination of the name of its supersets and itself separated by the colon [:] (e.g. `<setspec>`$s_a : s_b : s_h$`</setspec>`). Furthermore, let $R = \{r_0, \ldots, r_n\}$ be a set of records, then each $r_i \in O_j$ must contain the setspec of O_j in its header.

Throughout $\{O_i\}_{i \in I}$ it is possible to represent a hierarchical data structure, such as a tree, in OAI-PMH providing a granularity access to the items in the hierarchy and at the same time enabling the exchange of a single part of the hierarchy with the possibility of reconstructing the whole hierarchy whenever it is necessary.

In the same way we can apply the INS-M to OAI-PMH; let $\mathcal{U}$ be an Inverse Nested Set family and let J be the set of the `<setspec>` values where $j \in J = \{s_0 : s_1 : \ldots : s_k\}$ means that $\exists\ U_k \in \{U_j\}_{j \in J} = U_k \subset \ldots \subset U_1 \subset U_0$. In $\{U_j\}_{j \in J}$ differently that in $\{O_i\}_{i \in I}$ the following case may happen: Let $U_i, U_k, U_w \in \{U_j\}_{j \in J}$ then it is possible that $U_w \subset U_i$ and $U_w \subset U_k$ but either $U_i \not\subseteq U_k$ and $U_k \not\subseteq U_i$. If we consider $\{U_j\}_{j \in J}$ composed only of U_i, U_k and U_w, the identifier of U_i is `<setspec>`s_i`</setspec>` and the identifier of U_k is `<setspec>`s_k`</setspec>`. Instead, the identifier of U_w must be `<setspec>`$s_j : s_w$`</setspec>` and `<setspec>`$s_j : s_w$`</setspec>` at the same time; this means that in $\{U_j\}_{j \in J}$ there are two distinct OAI-sets, one identified by `<setspec>`$s_j : s_w$`</setspec>` and the other identified by `<setspec>`$s_k : s_w$`</setspec>`. This is due to the fact that the intersection between OAI-sets in OAI-PMH is not defined set-theoretically; indeed, the only way to get an intersection of two OAI-sets is enumerating the records. This means that we can know if an OAI-record belongs to two or more sets just by seeing whether there are two or more `<setspec>` entries in the header of the record. In this case the records belonging to U_w will contain two `<setspec>` entries in their header: `<setspec>`$s_j : s_w$`</setspec>` and `<setspec>`$s_k : s_w$`</setspec>`; note that only the `<setspec>` value is duplicated and not the records themselves.

With this view of OAI-PMH we can set a hierarchical structure of items as a well-defined nested set organization that maintains the relationships between the items just as a tree data structure does and moreover we can exploit the flexibility of the sets exchanging a specific subset while maintaining the integrity of the data. Indeed, in the header of the items there is the set membership information which, if necessary, enables the reconstruction of the hierarchy or part of it. Throughout the NS-M and INS-M it is possible to handle hierarchical structures in OAI-PMH simply by exploiting the inner functionalities of the protocol; indeed, no change of OAI-PMH is required to cope with the presented set data models.

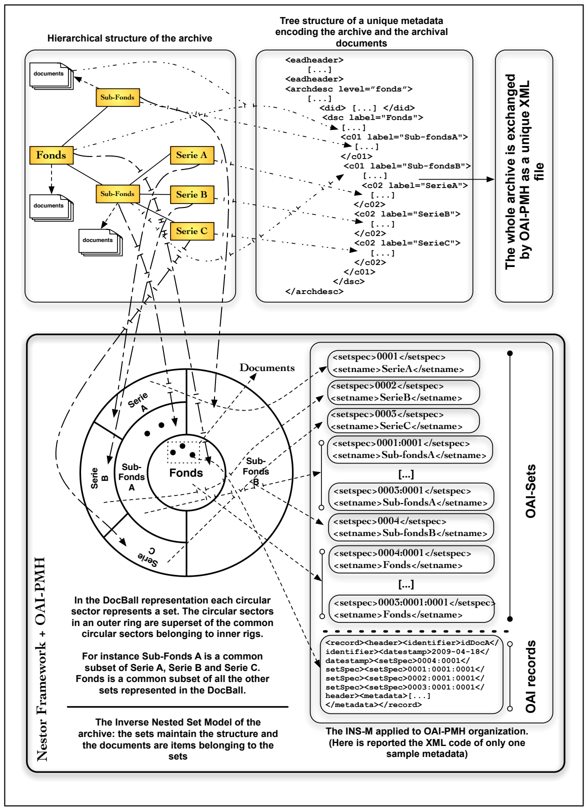
The choice between NS-M and INS-M is based on the application context: NS-M fosters the reconstruction of the lower levels of a hierarchy starting from a node, vice versa INS-M fosters the reconstruction of the upper levels.

4.3 The Set Data Models and OAI-PMH Applied to the Archives

This subsection describes how we can exchange archival metadata in a distributed environment and it is a continuation of the work presented in [6]. A brief introduction regarding the archive peculiarities is worthwhile for a better understanding of the proposed solutions. An archive is a complex cultural organization which is not simply constituted by a series of objects that have been accumulated and filed with the passing of time. Archives have to keep the context in which their documents have been created and the network of relationships among them in order to preserve their informative content and provide understandable and useful information over time. The context and the relationships between the documents are preserved thanks to the strongly hierarchical organization of the documents inside the archive. Indeed, an archive is divided by fonds and then by sub-fonds and then by series and then by sub-series and so on; at every level we can find documents belonging to a particular division of the archive or documents describing the nature of the considered level of the archive (e.g. a fond, a sub-fonds, etc.). The union of all these documents, the relationships and the context information permits the full informational power of the archival documents to be maintained.

In the digital environment an archive and its components are described by the use of metadata; these need to be able to express and maintain such structure and relationships. The standard format of metadata for representing the complex hierarchical structure of the archive is *Encoded Archival Description (EAD)* [10], which reflects the archival structure and holds relations between documents in the archive. On the other hand to maintain all this information an EAD file turns out to be a very large XML file with a deep hierarchical internal structure. Thus, accessing, searching and sharing individual items in the EAD might be difficult without taking into consideration the whole hierarchy. On the other hand, users are often interested in the information described at the item level, which is typically buried very deeply in the hierarchy and might be difficult to reach [11].

In Fig. 2 we can see two approaches to represent the archival organization and documents. The first approach is the EAD-like one in which the whole archive is mapped inside a unique XML file which is potentially very large and deeply hierarchical. All information about fonds, sub-fonds or series as well as the documents belonging to a specific archival division are mapped into several XML elements in the same XML file. With this approach we cannot exchange precise metadata through OAI-PMH, rather we have to exchange the whole archive. At the same time it is not possible to access a specific piece of information without accessing the whole hierarchy.


Hierarchical structure of the archive
documents
Sub-Fonds
Fonds
Serie A
Sub-Fonds
Serie B
documents
Serie C
documents
Tree structure of a unique metadata encoding the archive and the archival documents
The whole archive is exchanged by OAI-PMH as a unique XML file
Documents
Serie A
Serie B
Sub-Fonds A
Fonds
Sub-Fonds B
Serie C
Nestor Framework + OAI-PMH
OAI-Sets
OAI records

```
<eadheader>
    [...]
<eadheader>
<archdesc level="fonds">
    [...]
      <did> [...] </did>
       <dsc label="Fonds">
         [...]
         <c01 label="Sub-fondsA">
           [...]
         </c01>
          <c01 label="Sub-fondsB">
             [...]
            <c02 label="SerieA">
              [...]
           </c02>
           <c02 label="SerieB">
              [...]
           </c02>
           <c02 label="SerieC">
              [...]
           </c02>
         </c01>
       </dsc>
</archdesc>
```

```
<setspec>0001</setspec>
<setname>SerieA</setname>
<setspec>0002</setspec>
<setname>SerieB</setname>
<setspec>0003</setspec>
<setname>SerieC</setname>
<setspec>0001:0001</setspec>
<setname>Sub-fondsA</setname>
[...]
<setspec>0003:0001</setspec>
<setname>Sub-fondsA</setname>
<setspec>0004</setspec>
<setname>Sub-fondsB</setname>
<setspec>0004:0001</setspec>
<setname>Fonds</setname>
[...]
<setspec>0003:0001:0001</setspec>
<setname>Fonds</setname>
```

```
<record><header><identifier>idDocA</
identifier><datestamp>2009-04-18</
datestamp><setSpec>0004:0001</
setSpec><setSpec>0001:0001:0001</
setSpec><setSpec>0002:0001:0001</
setSpec><setSpec>0003:0001:0001</
header><metadata>[...]
</metadata></record>
```

In the DocBall representation each circular sector represents a set. The circular sectors in an outer ring are superset of the common circular sectors belonging to inner rigs.

For instance Sub-Fonds A is a common subset of Serie A, Serie B and Serie C. Fonds is a common subset of all the other sets represented in the DocBall.

The Inverse Nested Set Model of the archive: the sets maintain the structure and the documents are items belonging to the sets

The INS-M applied to OAI-PMH organization. (Here is reported the XML code of only one sample metadata)

Fig. 2. The hierarchical structure of an archive mapped into a metadata with a tree data structure, the alternative mapping in the INS-M and in OAI-PMH.

The second approach based on the INS-M permits us to overcome the presented issues. Indeed, the archival hierarchy is mapped into a family Ψ that for theorem 2 is a INS-F. In Ψ the documents are represented as items belonging to the opportune set. In this way the context information and the relationships between the documents are preserved thanks to the nested set organization and at the same time they are not bound to a rigid structure. Then, Ψ is represented in OAI-PMH throughout the family $\{U_j\}_{j \in J}$ of OAI-sets obtained setting of the `<setspec>` values as described in the previous subsection. For instance, we obtained four sets from the root with four different identifiers: "0004:0001", "0001:0001:0001", "0002:0001:0001" and "0003:0001:0001". In the same way are defined the sets mapped from the children of the root. The sets mapped from the external nodes are identified by "0001", "0002" and "0003". Thus, from the identifier of an OAI-set we can reconstruct the hierarchy through the ancestors to the root. By means of OAI-PMH it is possible to exchange a specific part of the archive while at the same time maintaining the relationships with the other parts of it. The INS-M fosters the reconstruction of the upper levels of a hierarchy that in the archival case often contain contextual information which permit the relationships of the archival documents to be inferred with the other documents in the archive and with the production and preservation environment. If a harvester asks for an OAI-set representing for instance an archival series it recursively obtains all the OAI-subsets and records in the path from the archival series to the principal fond that is the root of the archival tree.

This approach can also be applied with the NS-M mapping the archival hierarchy into a NS-F $\{O_i\}_{i \in I}$ following the procedure illustrated in the previous section. In this case there is a big difference in the harvesting procedure; indeed, the NS-M fosters the reconstruction of the lower levels of a hierarchy; thus, with the couple NS-M and OAI-PMH applied to the archive, if a harvester asks for an OAI-set representing for instance a sub-fond it recursively obtains all the OAI-subsets and items in the subtree rooted in the selected sub-fonds. In the archival context the application of the INS-M is useful when we have to reconstruct the context of an archival document; indeed, often the information required by a user is stored in the external nodes of the archival tree [11]. If we represent the archival tree by means of the INS-F, when a harvester requires an external node of the tree it will receive all the archival information contained in the nodes up to the root of the tree. This means that a Service Provider can offer a potential user the required information stored in the external node and also all the information stored in its ancestors nodes. This information is very useful for inferring the context of an archival metadata which is contained in the required external node; indeed, the ancestor nodes represent and contain the information related to the series, sub-fonds and fonds in which the archival metadata are classified. The application of NS-M is useful when we want to obtain all the sub-fonds, series, etc. in which a fonds is divided. There is not a one-fits-all solution but the choice of the model should be done on the basis of the operations that are performed more frequently on the archive.

5 Conclusions

We have discussed the relevance of the hierarchical structures in computer science with a specific examination of the DLSs. We have presented the tree data structure and highlighted the more relevant aspects to our treatment of hierarchical structures. We have also presented the NESTOR framework defining two set-theoretical data models called Nested Set Model and Inverse Nested Set Model as alternatives of the tree data structure. Furthermore, we have shown how a tree can be mapped in one model or the other. These models maintain the peculiarities of the tree with the flexibility and accessibility of sets. We have shown that without any changes on OAI-PMH, the protocol can exploit the NS-M or the INS-M in order to exchange hierarchical structures in a flexible way. Lastly we have presented a significant application of the presented set data models in conjunction with OAI-PMH represented by the archives. Indeed, we have shown how the hierarchical archive organization can be represented and exchanged in OAI-PMH and thus between different DLSs in a distributed environment.

Acknowledgments

The authors wish to thanks the anonymous reviewers for the useful comments. The work reported has been supported by a grant from the Italian Veneto Region. The study is also partially supported by EuropeanaConnect (Contract ECP-2008-DILI-52800).

References

1. M. Agosti, N. Ferro, and G. Silvello. Access and Exchange of Hierarchically Structured Resources on the Web with the NESTOR Framework. *Web Intelligence and Intelligent Agent Technology, IEEE/WIC/ACM International Conference*, pages 659–662, 2009.
2. K. W. Anderson and D. W. Hall. *Sets, Sequences, and Mappings: The Basic Concepts of Analysis.* John Wiley & Sons, Inc., New York, 1963.
3. J. Celko. *Joe Celko's SQL for Smarties: Advanced SQL Programming.* Morgan Kaufmann, 2000.
4. F. Crestani, J. Vegas, and P. de la Fuente. A Graphical User Interface for the Retrieval of Hierarchically Structured Documents. *Inf. Process. Management*, 40(2):269–289, 2004.
5. B. A. Davey and H. A. Priestley. *Introduction to Lattices and Order - 2nd Ed.* Cambridge University Press, 2002.
6. N. Ferro and G. Silvello. A Methodology for Sharing Archival Descriptive Metadata in a Distributed Environment. In *Proc. 12th European Conf. on Research and Advanced Tech. for Digital Libraries*, pages 268–279. LNCS 5173, Springer, Germany, 2008.
7. N. Ferro and G. Silvello. The NESTOR Framework: How to Handle Hierarchical Data Structures. In *Proc. 13th European Conf. on Research and Advanced Tech. for Digital Libraries*, pages 215–226. LNCS 5714, Springer, Germany, 2009.
8. P. R. Halmos. *Naive Set Theory.* D. Van Nostrand Company, Inc., New York, USA, 1960.
9. D. E. Knuth. *The Art of Computer Programming, third edition*, volume 1. Addison Wesley, 1997.
10. D. V. Pitti. Encoded Archival Description. An Introduction and Overview. *D-Lib Magazine*, 5(11), 1999.
11. S. L. Shreeves, J. S. Kaczmarek, and T. W. Cole. Harvesting Cultural Heritage Metadata Using the OAI Protocol. *Library Hi Tech*, 21(2):159–169, 2003.
12. H. Van de Sompel, C. Lagoze, M. Nelson, and S. Warner. Implementation Guidelines for the Open Archive Initiative Protocol for Metadata Harvesting - Guidelines for Harvester Implementers. Technical report, Open Archive Initiative, p. 6, 2002.
13. H. Van de Sompel, C. Lagoze, M. Nelson, and S. Warner. The Open Archives Initiative Protocol for Metadata Harvesting (2nd ed.). Technical report, Open Archive Initiative, p. 24, 2003.
14. J. Vegas, F. Crestani, and P. de la Fuente. Context Representation for Web Search Results. *Journal of Information Science*, 33(1):77–94, 2007.