ΜΕΡΟΣ IV:

XPath
XSLT
XSL FO

Πίνακας Περιεχομένων

ΚΕΦΑΛΑΙΟ 9:
XML Path Language (XPath)

9.1. Εισαγωγή

Η *XML Path Language* (*XPath*) είναι μια γλώσσα μέσω της οποίας μπορούμε να αναφερθούμε σε τμήματα ενός XML τεκμηρίου. Η ονομασία της XPath οφείλεται στην υιοθέτηση από μέρους της γλώσσας μιας *σύνταξης μονοπατιού* (path notation), παρόμοιας με αυτήν που συναντάμε στα URLs, η οποία επιτρέπει την πλοήγηση στην ιεραρχική δομή ενός XML τεκμηρίου. Η XPath βασίζεται στην ιδέα ότι ένα XML τεκμήριο μπορεί να παρασταθεί σαν ένα δέντρο.
Εκτός από την αναφορά σε τμήματα ενός XML τεκμηρίου, η XPath μπορεί να χρησιμοποιηθεί για *ταίριασμα* (matching), για τον έλεγχο δηλαδή αν ένας κόμβος ταιριάζει με μια συγκεκριμένη *μορφή* (pattern). Η XPath έχει σχεδιαστεί κυρίως για να χρησιμοποιηθεί ως συστατικό τμήμα άλλων προτύπων όπως είναι η XSLT και η XPointer. Τα πρότυπα αυτά θα συζητηθούν σε επόμενα κεφάλαια.

9.2. Το Μοντέλο Δεδομένων της XPath

Η XPath αντιμετωπίζει ένα XML τεκμήριο σαν ένα *δέντρο* (tree). Η αντιμετώπιση αυτή είναι ιδεατή και δεν επιβάλει κάποια συγκεκριμένη υλοποίηση που να αποτυπώνει αυτήν τη δομή. Ένα δέντρο περιέχει *κόμβους* (nodes). Στο δεντρικό μοντέλο της XPath προβλέπονται επτά τύποι κόμβων, οι οποίοι αντιστοιχούν στα δομικά στοιχεία της XML και είναι οι ακόλουθοι:

1. *κόμβοι ρίζας* (root nodes)
2. *κόμβοι στοιχείων* (element nodes)
3. *κόμβοι κειμένου* (text nodes)
4. *κόμβοι γνωρισμάτων* (attribute nodes)
5. *κόμβοι χωρών ονομάτων* (namespace nodes)
6. *κόμβοι οδηγιών επεξεργασίας* (processing instruction nodes)
7. *κόμβοι σχολίων* (comment nodes)

Μερικοί κόμβοι (όπως οι κόμβοι στοιχείων και οι κόμβοι γνωρισμάτων) έχουν ένα *επεκταμένο όνομα* (expanded-name), το οποίο αποτελείται από ένα τοπικό τμήμα και ένα URI χώρου ονομάτων.

Για κάθε κόμβο, οποιουδήποτε τύπου, υπάρχει και μια αντίστοιχη *τιμή συμβολοσειράς* (string value).
Το δέντρο έχει μια *ρίζα* (root) που είναι η ρίζα του τεκμηρίου. Για κάθε στοιχείο του τεκμηρίου υπάρχει και ένας κόμβος στοιχείου στο δέντρο. Η ρίζα καθώς και κάθε στοιχείο έχει μια διατεταγμένη λίστα από *κόμβους παιδιά* (child nodes). Κάθε κόμβος, εκτός από τη ρίζα, έχει ακριβώς ένα κόμβο *γονιό* (parent). Οι *απόγονοι* (descendants) ενός κόμβου είναι τα παιδιά του, τα παιδιά των παιδιών του κ.λ.π. Τα παιδιά ενός κόμβου στοιχείου είναι οι κόμβοι στοιχείων, οι κόμβοι σχολίων, οι κόμβοι οδηγιών επεξεργασίας και οι κόμβοι κειμένου που αποτελούν το περιεχόμενο του στοιχείου. Κάθε κόμβος στοιχείου συσχετίζεται με ένα σύνολο κόμβων γνωρισμάτων (που είναι τα γνωρίσματα του συγκεκριμένου στοιχείου). Το στοιχείο θεωρείται γονιός καθενός από τα γνωρίσματα αυτά, όμως τα γνωρίσματα δε θεωρούνται παιδιά του στοιχείου αυτού.
Ένας κόμβος στοιχείου είναι δυνατό να διαθέτει ένα μοναδικό *αναγνωριστικό* (identifier). Το αναγνωριστικό αυτό είναι η τιμή του γνωρίσματος του στοιχείου, ο τύπος του οποίου έχει δηλωθεί στο DTD ως `ID`. Μερικά γνωρίσματα όπως είναι το `xml:lang` και το `xml:space` έχουν την ιδιότητα να εφαρμόζονται και σε όλους τους απογόνους του στοιχείου στο οποίο εμφανίζονται εκτός εάν επανακαθοριστεί η τιμή τους από νεώτερο στιγμιότυπο του ίδιου γνωρίσματος.
Τα δεδομένα χαρακτήρων ομαδοποιούνται σε κόμβους κειμένου κατά τέτοιον τρόπον ώστε σε κάθε κόμβο κειμένου να ομαδοποιείται το μέγιστο δυνατό πλήθος χαρακτήρων. Ένας κόμβος κειμένου δεν έχει ποτέ έναν άλλο κόμβο κειμένου που να είναι αδελφός του και να είναι ο αμέσως προηγούμενος ή ο αμέσως επόμενος κόμβος.
Με κάθε στοιχείο συνδέεται ένα σύνολο από κόμβους χώρων ονομάτων έτσι ώστε να έχουμε έναν τέτοιο κόμβο για κάθε χώρο ονομάτων στην εμβέλεια του οποίου βρίσκεται το στοιχείο.
Για κάθε οδηγία επεξεργασίας (εκτός από αυτές που βρίσκονται στο DTD) υπάρχει και ένας κόμβος οδηγίας επεξεργασίας.
Για κάθε σχόλιο (εκτός από αυτά που βρίσκονται στο DTD) υπάρχει και ένας κόμβος σχολίων.
Η φυσική διάταξη των στοιχείων, των γνωρισμάτων και των υπολοίπων συστατικών της XML μέσα σε ένα XML τεκμήριο, η σειρά δηλαδή με την οποία εμφανίζονται αυτά στο τεκμήριο, ονομάζεται *διάταξη τεκμηρίου* (document order). Είναι φανερό ότι στη διάταξη αυτή ο κόμβος της ρίζας εμφανίζεται πρώτος, ενώ κάθε στοιχείο εμφανίζεται πριν από τα παιδιά του. Τα στοιχεία διατάσσονται με βάση τη σειρά εμφάνισης της ετικέτας αρχής τους. Η ανάστροφη διάταξη από τη διάταξη τεκμηρίου ονομάζεται *ανάστροφη διάταξη τεκμηρίου* (reverse document order).
Πρέπει να τονιστεί ότι τα XML τεκμήρια στα οποία μπορεί να επενεργεί η XPath πρέπει να υπακούουν στις προδιαγραφές που θέτει η σύσταση της W3C για τους Χώρους Ονομάτων XML (βλέπε Κεφάλαιο 5).
Στη συνέχεια του κεφαλαίου θα ασχοληθούμε μόνο με τις τέσσερις πρώτες κατηγορίες κόμβων, οι οποίες είναι και οι σημαντικότερες.

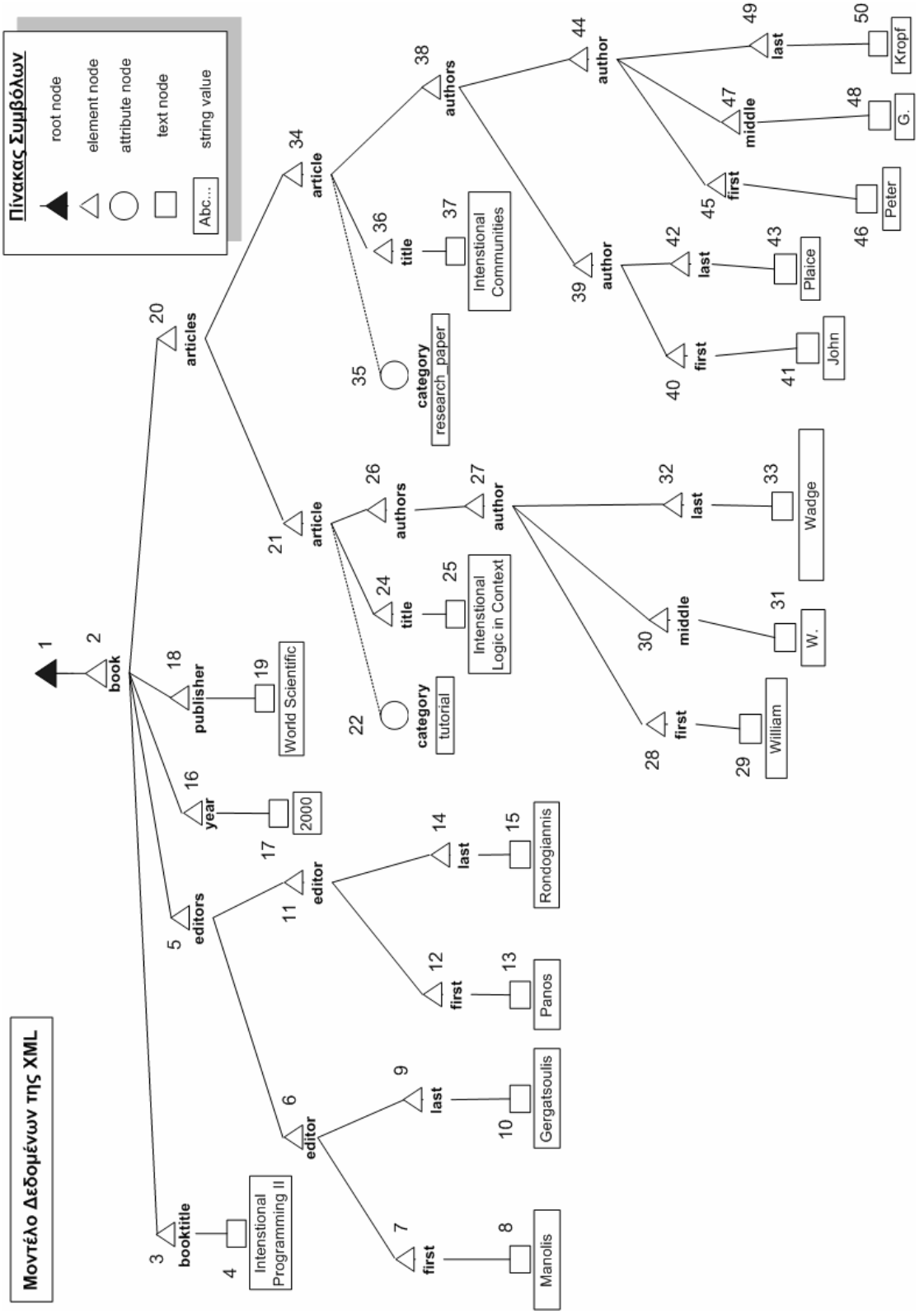
Παράδειγμα 9.1. Στο παράδειγμα αυτό θα εξετάσουμε το μοντέλο δεδομένων των XML τεκμηρίων με τη βοήθεια του τεκμηρίου που φαίνεται στον Πίνακα 9.1. Το τεκμήριο αυτό περιγράφει ένα τμήμα του περιεχομένου ενός βιβλίου (το οποίο περιλαμβάνει μια συλλογή άρθρων).

```
<book>
   <booktitle>Intensional Programming II</booktitle>
   <editors>
      <editor>
         <first>Manolis</first>
         <last>Gergatsoulis</last>
      </editor>
      <editor>
         <first>Panos</first>
         <last>Rondogiannis</last>
      </editor>
   </editors>
   <year>2000</year>
   <publisher>World Scientific</publisher>
   <articles>
      <article category = "tutorial" >
         <title>Intensional Logic in Context</title>
         <authors>
            <author>
                <first>William</first>
                <middle>W.</middle>
                <last>Wadge</last>
            </author>
         </authors>
      </article>
      <article category = "research_paper" >
         <title>Intensional Communities</title>
         <authors>
            <author>
                <first>John</first>
                <last>Plaice</last>
            </author>
            <author>
                <first>Peter</first>
                <middle>G.</middle>
                <last>Kropf</last>
            </author>
         </authors>
      </article>
   </articles>
</book>
```

Πίνακας 9.1.: Παράδειγμα XML τεκμηρίου.

Στο σχήμα που ακολουθεί (Σχήμα 9.1), βλέπουμε τη δεντρική αναπαράσταση του XML τεκμηρίου του Πίνακα 9.1. Στο δέντρο φαίνονται τέσσαρις διαφορετικοί τύποι κόμβων: κόμβος ρίζας, κόμβοι στοιχείων, κόμβοι κειμένου, και κόμβοι γνωρισμάτων. Για την αναπαράσταση κάθε τύπου κόμβου έχει υιοθετηθεί και ένα αντίστοιχο ειδικό σύμβολο όπως φαίνεται

στον ένθετο πίνακα συμβόλων του σχήματος. Οι κόμβοι του δέντρου έχουν αριθμηθεί για να διευκολυνθεί η αναφορά στους κόμβους αυτούς στα παραδείγματα που ακολουθούν.



Σχήμα 9.1. Το δέντρο που αντιστοιχεί στο XML τεκμήριο του Πίνακα 9.1.

9.3. Εκφράσεις και Περιβάλλοντα

Η βασική συντακτική δομή της XPath είναι η *έκφραση* (expression). Το αποτέλεσμα που προκύπτει από τον υπολογισμό μιας έκφρασης είναι ένα αντικείμενο το οποίο ανήκει σε μια από τις ακόλουθες κατηγορίες:

- *Σύνολο κόμβων* (node-set) (μια μη διατεταγμένη συλλογή κόμβων χωρίς επαναλήψεις).
- *Τιμή τύπου Boolean* (true ή false).
- *Αριθμός* (πραγματικός αριθμός κινητής υποδιαστολής - floating-point number).
- *Συμβολοσειρά* (string) (μια ακολουθία χαρακτήρων UCS).

Μια έκφραση υπολογίζεται ως προς ένα *περιβάλλον* (context). Ένα περιβάλλον περιλαμβάνει:

- Έναν *κόμβο περιβάλλοντος* (context node).
- Ένα ζευγάρι θετικών ακέραιων αριθμών (μη μηδενικών) οι οποίοι αποτελούν τη *θέση περιβάλλοντος* (context position) και το *μέγεθος περιβάλλοντος* (context size). Ο αριθμός που αντιπροσωπεύει τη θέση περιβάλλοντος είναι πάντοτε μικρότερος ή ίσος με αυτόν που αντιπροσωπεύει το μέγεθος περιβάλλοντος.
- Ένα σύνολο από *αναθέσεις τιμών* για τις μεταβλητές.
- Μια *βιβλιοθήκη συναρτήσεων* (function library).
- Το σύνολο των *δηλώσεων χώρων ονομάτων* στην εμβέλεια των οποίων βρίσκεται η έκφραση.

9.4. Μονοπάτια τοποθεσίας

Τα *μονοπάτια τοποθεσίας* (location paths) αποτελούν ιδιαίτερα χρήσιμο είδος εκφράσεων. Ένα μονοπάτι τοποθεσίας υπολογίζεται ως προς έναν κόμβο περιβάλλοντος και επιλέγει ένα σύνολο κόμβων. Τα μονοπάτια τοποθεσίας μπορούν με τη σειρά τους να περιέχουν αναδρομικά άλλες εκφράσεις οι οποίες χρησιμοποιούνται για το φιλτράρισμα των κόμβων που προκύπτουν.

Παράδειγμα 9.2. Στο παράδειγμα αυτό παρουσιάζεται ένα δείγμα μονοπατιών τοποθεσίας καθώς και το αποτέλεσμα που προκύπτει από τον υπολογισμό τους ως προς το δέντρο του Σχήματος 9.1.:

1. child::editor : επιλέγει τα παιδιά του κόμβου περιβάλλοντος τα οποία είναι στοιχεία με ετικέτα editor. Η τιμή του μονοπατιού αυτού, αν ο κόμβος περιβάλλοντος είναι ο 5 (το στοιχείο editors) είναι το σύνολο {6,11}. Αν όμως ο κόμβος περιβάλλοντος είναι ο 2 (το στοιχείο book),

τότε το αποτέλεσμα είναι το κενό σύνολο { }, (αφού το στοιχείο book δεν έχει παιδιά τα οποία είναι στοιχεία με ετικέτα editor.

2. descendant::last : επιλέγει τα στοιχεία που είναι απόγονοι του κόμβου περιβάλλοντος και έχουν ετικέτα last. Η τιμή του μονοπατιού αυτού, αν ο κόμβος περιβάλλοντος είναι ο 5 (το στοιχείο editors), είναι το σύνολο {9,14}. Περιλαμβάνει δηλαδή τα επώνυμα των editors του βιβλίου. Αντίθετα, αν ο κόμβος περιβάλλοντος είναι ο 2, τότε η τιμή που προκύπτει από τον υπολογισμό του μονοπατιού αυτού είναι το σύνολο {9,14,32,42,49}. Στην περίπτωση αυτή έχουμε τα επώνυμα τόσο των editors όσο και των συγγραφέων των άρθρων.
3. child::articles/descendant::last : επιλέγει τους κόμβους στοιχεία με ετικέτα last οι οποίοι είναι απόγονοι ενός κόμβου με ετικέτα articles ο οποίος είναι παιδί του κόμβου περιβάλλοντος. Η τιμή του μονοπατιού αυτού, αν ο κόμβος περιβάλλοντος είναι ο 2 (το στοιχείο book), είναι το σύνολο {32,42,49}. Περιλαμβάνει δηλαδή τα επώνυμα των συγγραφέων των άρθρων του βιβλίου.
4. child::article[position()=1] : επιλέγει το πρώτο από αριστερά κόμβο στοιχείο με ετικέτα article ο οποίος είναι παιδί του κόμβου περιβάλλοντος. Ο υπολογισμός του μονοπατιού αν ο κόμβος περιβάλλοντος είναι ο 20 (ο κόμβος με ετικέτα articles) μας δίνει ως αποτέλεσμα τον κόμβο 21 (το πρώτο κατά σειρά εμφάνισης άρθρο).
5. /descendant::first : επιλέγει όλους τους κόμβους στοιχείων με ετικέτα first οι οποίοι είναι απόγονοι της ρίζας του δέντρου. Το αποτέλεσμα του υπολογισμού του μονοπατιού είναι το σύνολο των κόμβων {7,12,28,40,45}.
6. /descendant::article[attribute::category="research_paper"]: επιλέγει όλους τους κόμβους στοιχείων με ετικέτα article οι οποίοι είναι απόγονοι της ρίζας και έχουν ένα γνώρισμα με όνομα category και τιμή "research_paper". Το αποτέλεσμα του υπολογισμού του μονοπατιού είναι το σύνολο {34}.
7. /descendant::author/child::*[position()=last()] : επιλέγει το τελευταίο στοιχείο που είναι παιδί κάθε στοιχείου author. Το αποτέλεσμα του υπολογισμού του μονοπατιού είναι το σύνολο {32,42,49}.
8. attribute::* : επιλέγει όλους τους κόμβους γνωρισμάτων του κόμβου περιβάλλοντος. Το αποτέλεσμα του υπολογισμού του μονοπατιού αν ο κόμβος περιβάλλοντος είναι ο 21 είναι το σύνολο {22}.

❑

Γενικά στην XPath υπάρχουν δύο είδη μονοπατιών τοποθεσίας, τα *σχετικά μονοπάτια τοποθεσίας* (relative location paths) και τα *απόλυτα μονοπάτια τοποθεσίας* (absolute location paths).

9.4.1. Σχετικά Μονοπάτια Τοποθεσίας

Ένα σχετικό μονοπάτι αποτελείται από ένα ή περισσότερα *βήματα τοποθεσίας* (location steps) τα οποία χωρίζονται μεταξύ τους με το σύμβολο /. Τα μονοπάτια των περιπτώσεων 1, 2, 3, 4 και 8 στο Παράδειγμα 9.2 είναι σχετικά μονοπάτια. Τα βήματα τοποθεσίας στα σχετικά μονοπάτια υπολογίζονται από αριστερά προς τα δεξιά. Το πιο αριστερά τοποθετημένο βήμα μονοπατιού υπολογίζεται ως προς τον κόμβο περιβάλλοντος και δίνει σαν αποτέλεσμα ένα σύνολο κόμβων. Στη συνέχεια υπολογίζεται το επόμενο βήμα μονοπατιού. Το βήμα αυτό θεωρεί σαν κόμβο περιβάλλοντος καθέναν από τους κόμβους που προέκυψαν από το προηγούμενο βήμα. Για κάθε έναν από αυτούς παράγει ως αποτέλεσμα ένα σύνολο κόμβων. Το αποτέλεσμα του υπολογισμού του βήματος είναι η ένωση των επιμέρους συνόλων (των αποτελεσμάτων) που προκύπτουν κατ' αυτόν τον τρόπο. Η διαδικασία αυτή επαναλαμβάνεται μέχρι να υπολογιστούν όλα τα βήματα του μονοπατιού. Το αποτέλεσμα από τον υπολογισμό του τελευταίου βήματος, είναι και το αποτέλεσμα που προκύπτει από τον υπολογισμό του (συνολικού) μονοπατιού.

Παράδειγμα 9.3. Ο υπολογισμός του μονοπατιού:

child::article/descendant::last

θεωρώντας ως κόμβο περιβάλλοντος τον κόμβο 20 (που αντιστοιχεί στο στοιχείο articles) γίνεται ως εξής. Αρχικά υπολογίζεται το βήμα child::article ως προς τον κόμβο περιβάλλοντος 20. Αυτό δίνει σαν αποτέλεσμα το σύνολο {21,34}. Στη συνέχεια υπολογίζεται το βήμα descendant::last για καθένα από τους κόμβους περιβάλλοντος 21 και 34. Προκύπτουν έτσι τα σύνολα {32} και {42,49} αντίστοιχα. Το αποτέλεσμα από τον υπολογισμό του μονοπατιού child::article/descendant::last είναι επομένως το σύνολο {32,42,49} που προκύπτει ως ένωση των συνόλων {32} και {42,49}.

❑

9.4.2. Απόλυτα Μονοπάτια Τοποθεσίας

Ένα απόλυτο μονοπάτι τοποθεσίας περιλαμβάνει το σύμβολο / ακολουθούμενο προαιρετικά από ένα σχετικό μονοπάτι τοποθεσίας. Το / επιλέγει τη ρίζα του τεκμηρίου. Αν το / ακολουθείται από σχετικό μονοπάτι τότε επιλέγεται το σύνολο των κόμβων που προκύπτει από την εφαρμογή του σχετικού μονοπατιού θεωρώντας τη ρίζα σαν κόμβο περιβάλλοντος. Τα μονοπάτια των περιπτώσεων 5, 6 και 7 στο παράδειγμα 9.2 είναι απόλυτα μονοπάτια.

9.5. Βήματα Τοποθεσίας

Στη γενική του μορφή ένα βήμα τοποθεσίας χωρίζεται σε τρία μέρη τα οποία κατά σειρά εμφάνισης είναι:

- Ο *άξονας* (axis), ο οποίος χρησιμοποιείται για να καθορίσει τους κόμβους που επιλέγονται από το βήμα τοποθεσίας με κριτήριο τη σχετική θέση τους ως προς τον κόμβο περιβάλλοντος στη δεντρική δομή του τεκμηρίου. Τα child, descendant, και attribute στο Παράδειγμα 9.2 είναι άξονες.
- Ο *έλεγχος κόμβου* (node test), ο οποίος καθορίζει ποιοι από τους κόμβους που ικανοποιούν το δομικό κριτήριο που επιβάλλει ο άξονας επιλέγονται τελικά. Στο Παράδειγμα 9.2 τα ονόματα των στοιχείων ή το σύμβολο * που ακολουθούν το :: είναι έλεγχοι κόμβου.
- Μηδέν ή περισσότερα *κατηγορήματα* (predicates). Τα κατηγορήματα χρησιμοποιούν εκφράσεις για να διυλίσουν παραπέρα το σύνολο των κόμβων που προκύπτουν από την εφαρμογή του άξονα και του ελέγχου κόμβου. Το [position()=1], το [position()=last()], και το [attribute::category='research_paper'] στο Παράδειγμα 9.2 είναι κατηγορήματα.

Συντακτικά, η ονομασία του άξονα χωρίζεται από τον έλεγχο κόμβου με το σύμβολο :: (double colon). Μετά τον έλεγχο κόμβου ακολουθούν τα κατηγορήματα καθένα από τα οποία περικλείεται σε [...].

Παράδειγμα 9.4. Στο βήμα μονοπατιού child::*[self::first or self::last] το child είναι ο άξονας, το * είναι ο έλεγχος κόμβου, ενώ το [self::first or self::last] είναι κατηγόρημα. Η εφαρμογή αυτού του βήματος στο δέντρο του Σχήματος 9.1, θεωρώντας σαν κόμβο περιβάλλοντος τον κόμβο 27 (με ετικέτα author), θα δώσει σαν αποτέλεσμα το σύνολο {28,32}, δηλαδή τα στοιχεία first και last που είναι παιδιά του κόμβου 27. Επομένως, η σημασία του βήματος αυτού μπορεί να εκφραστεί από την πρόταση: «δώσε μου όλα τα παιδιά του κόμβου περιβάλλοντος τα οποία έχουν για ετικέτες first ή last».
❑

Το σύνολο των κόμβων που αποτελεί το αποτέλεσμα της εφαρμογής ενός βήματος τοποθεσίας μπορεί να υπολογιστεί συλλέγοντας αρχικά τους κόμβους που προκύπτουν από την εφαρμογή του άξονα μαζί με τον έλεγχο κόμβου και στη συνέχεια φιλτράροντας αυτό το σύνολο κόμβων μέσω της εφαρμογής κάθε ενός από τα κατηγορήματα διαδοχικά.

9.6. Άξονες

Η XPath διαθέτει 13 διαφορετικούς άξονες καθένας από τους οποίους ορίζει ένα σύνολο κόμβων σε σχέση με τον κόμβο περιβάλλοντος. Στον πίνακα που ακολουθεί φαίνονται οι άξονες που διαθέτει η XPath μαζί με την ερμηνεία τους:

Άξονας	Εξήγηση
child	Αναφέρεται στα παιδιά του κόμβου περιβάλλοντος.
descendant	Αναφέρεται στα στοιχεία που είναι απόγονοι του κόμβου περιβάλλοντος (τα παιδιά, τα παιδιά των παιδιών κ.λ.π.).
parent	Αναφέρεται στο γονιό του κόμβου περιβάλλοντος (αν υπάρχει).
ancestor	Αναφέρεται στους προγόνους του κόμβου περιβάλλοντος (ο γονιός, ο γονιός του γονιού κ.λ.π.), μέχρι και τη ρίζα.
following-sibling	Περιλαμβάνει όλα τα στοιχεία που είναι αδέλφια του κόμβου περιβάλλοντος τα οποία έπονται από αυτόν.
preceding-sibling	Περιλαμβάνει όλα τα στοιχεία που είναι αδέλφια του κόμβου περιβάλλοντος τα οποία προηγούνται από αυτόν.
following	Περιλαμβάνει τους κόμβους του τεκμηρίου οι οποίοι απαντώνται στο τεκμήριο μετά από το τέλος του κόμβου περιβάλλοντος, με εξαίρεση τους κόμβους γνωρισμάτων και τους κόμβους χώρων ονομάτων.
preceding	Περιλαμβάνει τους κόμβους του τεκμηρίου οι οποίοι τερματίζονται (εμφανίζεται η ετικέτα τέλους τους) στο τεκμήριο πριν από την έναρξη του κόμβου περιβάλλοντος, με εξαίρεση τους κόμβους γνωρισμάτων και τους κόμβους χώρων ονομάτων.
attribute	Περιλαμβάνει τα γνωρίσματα του κόμβου περιβάλλοντος.
namespace	Περιλαμβάνει τους κόμβους χώρων ονομάτων που αντιστοιχούν στον κόμβο περιβάλλοντος. Είναι το κενό σύνολο αν ο κόμβος περιβάλλοντος δεν είναι κόμβος στοιχείου.
self	Περιλαμβάνει μόνο τον κόμβο περιβάλλοντος.
descendant-or-self	Περιλαμβάνει τον κόμβο περιβάλλοντος και τους απογόνους του.
ancestor-or-self	Περιλαμβάνει τον κόμβο περιβάλλοντος και τους προγόνους του (επομένως περιέχει πάντα τη ρίζα).

Πίνακας 9.2: Οι άξονες της XPath.

9.7. Έλεγχοι κόμβων

Κάθε άξονας έχει ένα *πρωτεύοντα τύπο κόμβων* (principal node type). Για όλους τους άξονες ο πρωτεύον τύπος είναι ο "element" εκτός από τους άξονες attribute και namespace για τους οποίους ο πρωτεύον τύπος είναι τα "attribute" και "namespace" αντίστοιχα.

Αν ο έλεγχος κόμβου είναι όνομα τότε ένας κόμβος ικανοποιεί τον έλεγχο όταν ο τύπος του κόμβου ταυτίζεται με τον πρωτεύοντα τύπο κόμβων (του άξονα) και το όνομα του κόμβου ταυτίζεται με το όνομα που αποτελεί τον έλεγχο κόμβου (σε περίπτωση που έχουμε χρήση χώρων ονομάτων τότε και τα δύο ονόματα θεωρείται ότι βρίσκονται στην επεκταμένη μορφή τους, δηλαδή έχουν προστεθεί τα κατάλληλα προθέματα σε αυτά).
Η χρήση του συμβόλου * ως ελέγχου κόμβου υποδηλώνει ότι ο έλεγχος ικανοποιείται για κάθε κόμβο που ανήκει στον πρωτεύοντα τύπο κόμβου. Ο έλεγχος κόμβου text() ικανοποιείται για κάθε κόμβο κειμένου. Ο έλεγχος comment() αληθεύει για κάθε κόμβο σχολίων. Ο έλεγχος processing-instruction() αληθεύει για κάθε κόμβο οδηγιών επεξεργασίας. Στην περίπτωση του processing-instruction() μπορούμε να έχουμε σαν όρισμα της συνάρτησης και μια σταθερά που αντιστοιχεί στο όνομα της οδηγίας επεξεργασίας. Τέλος, ο έλεγχος node() ικανοποιείται για κάθε κόμβο οποιουδήποτε τύπου.

Παράδειγμα 9.5. Παραδείγματα ελέγχου κόμβων:
1. Το βήμα child::editor επιστρέφει ως τιμή το σύνολο που περιλαμβάνει όλους τους κόμβους στοιχείων που είναι παιδιά του κόμβου περιβάλλοντος και οι οποίοι έχουν για ετικέτα το editor.
2. Το βήμα ancestor::* επιστρέφει ως τιμή το σύνολο που περιλαμβάνει όλους τους κόμβους στοιχείων που είναι πρόγονοι του κόμβου περιβάλλοντος.
3. Το βήμα attribute::* επιστρέφει ως τιμή το σύνολο των γνωρισμάτων του κόμβου περιβάλλοντος.
4. Το βήμα child::text() επιστρέφει όλους τους κόμβους που είναι παιδιά του κόμβου περιβάλλοντος και είναι κόμβοι κειμένου.

❑

9.8. Κατηγορήματα

Ένα κατηγόρημα φιλτράρει ένα σύνολο κόμβων ως προς έναν άξονα και παράγει ένα νέο σύνολο κόμβων (που είναι υποσύνολο του αρχικού). Για κάθε κόμβο του αρχικού συνόλου κόμβων, υπολογίζεται η *έκφραση κατηγορήματος* (PredicateExpr). Αν η τιμή της έκφρασης κατηγορήματος για έναν κόμβο είναι η τιμή true τότε ο κόμβος συμπεριλαμβάνεται στο σύνολο κόμβων που προκύπτει ως αποτέλεσμα, διαφορετικά ο κόμβος αποκλείεται από το αποτέλεσμα.
Προκειμένου να παρουσιάσουμε τον τρόπο υπολογισμού της έκφρασης κατηγορήματος θα πρέπει να εισάγουμε τις έννοιες του *ευθύ άξονα* (forward axis) και *ανάστροφου άξονα* (reverse axis).
Ένας άξονας θα λέμε ότι είναι ευθύς αν περιλαμβάνει πάντα τον κόμβο περιβάλλοντος ή/και κόμβους οι οποίοι βρίσκονται στο τεκμήριο μετά τον κόμβο περιβάλλοντος (ως προς τη διάταξη τεκμηρίου). Αντίθετα, ένας άξονας

θα λέμε ότι είναι ανάστροφος αν περιλαμβάνει πάντα τον κόμβο περιβάλλοντος ή/και κόμβους οι οποίοι βρίσκονται στο τεκμήριο πριν τον κόμβο περιβάλλοντος. Με βάση αυτόν τον ορισμό, είναι φανερό ότι οι άξονες ancestor, ancestor-or-self, preceding, και preceding-sibling είναι ανάστροφοι άξονες ενώ όλοι οι υπόλοιποι είναι ευθείς άξονες. Ο άξονας self είναι ταυτόχρονα και ευθύς και ανάστροφος άξονας.
Η *εγγύτητα* (proximity position) του μέλους ενός συνόλου κόμβων αναφορικά με έναν άξονα ορίζεται ως η θέση του κόμβου (ξεκινώντας την αρίθμηση από το 1) στο σύνολο κόμβων αν αυτό διαταχθεί ως προς τη σειρά εμφάνισης των κόμβων στο τεκμήριο εφόσον ο άξονας είναι ευθύς ή ως προς την ανάστροφη σειρά εμφάνισης αν ο άξονας είναι ανάστροφος.
Για τον υπολογισμό της έκφρασης κατηγορήματος ως προς έναν κόμβο, ο κόμβος αυτός θεωρείται κόμβος περιβάλλοντος, ο αριθμός των κόμβων στο σύνολο κόμβων εκλαμβάνεται ως το μέγεθος του περιβάλλοντος και η εγγύτητα του κόμβου ως προς τον άξονα θεωρείται ως η θέση του περιβάλλοντος. Κατά τον υπολογισμό της έκφρασης κατηγορήματος το αποτέλεσμα μετατρέπεται σε τιμή τύπου boolean. Κατά τον υπολογισμό, αν το αποτέλεσμα είναι ακέραιος αριθμός τότε αυτός μετατρέπεται σε true αν είναι ίσος με τη θέση περιβάλλοντος, διαφορετικά μετατρέπεται σε false. Αν το αποτέλεσμα δεν είναι αριθμός τότε αυτό μετατρέπεται σε boolean καλώντας τη συνάρτηση boolean( ).

9.9. Συντετμημένη σύνταξη

Η σύνταξη των μονοπατιών τοποθεσίας που παρουσιάσαμε στις προηγούμενες ενότητες μπορεί να απλοποιηθεί αν εκμεταλλευτούμε διάφορες συντμήσεις που παρέχονται από την XPath. Η πιο σημαντική σύντμηση αφορά την παράλειψη του child:: από ένα βήμα τοποθεσίας, θεωρώντας τον άξονα child ως τον *προκαθορισμένο* (default) άξονα. Για παράδειγμα, το μονοπάτι τοποθεσίας editor/last είναι συντομογραφία του child::editor/child::last. Στον Πίνακα 9.3 παρουσιάζονται οι περισσότερο χρήσιμες συντμήσεις που παρέχει η XPath.

Συντόμευση	***Πλήρης γραφή***
.	self::node()
..	parent::node()
name	child::*name*
@*name*	attribute::*name*
//	/descendant-or-self::node()/
[*N*]	[position()=*N*]
Πίνακας 9.3: Συντμήσεις μονοπατιών της XPath.	

Παράδειγμα 9.6. Το μονοπάτι:

/child::book/child::articles/child::article[attribute::category="tutorial"]
μπορεί να γραφτεί σε συντετμημένη μορφή ως:

/book/articles/article[@category="tutorial"]

και αν υπολογιστεί στο τεκμήριο του Σχήματος 9.1, επιλέγει το σύνολο κόμβων με ετικέτα article οι οποίοι διαθέτουν γνώρισμα category με τιμή tutorial. Επομένως, το αποτέλεσμα του υπολογισμού του μονοπατιού αυτού είναι το {21}.

❑

Παράδειγμα 9.7. Το μονοπάτι:

/child::book/descendant-or-self::node()/child::first

μπορεί να γραφεί σε συντετμημένη μορφή ως:

/book//first

Αν υπολογιστεί το μονοπάτι αυτό με βάση το τεκμήριο του Σχήματος 9.1, θα επιλεγούν όλα τα στοιχεία που είναι απόγονοι του κόμβου book και έχουν ετικέτα first. Έτσι, το αποτέλεσμα του υπολογισμού του μονοπατιού αυτού είναι το σύνολο {7,12,28,40,45}.

❑

Παράδειγμα 9.8. Το μονοπάτι:

parent::node()/child::first

μπορεί να γραφεί σε συντετμημένη μορφή ως:

../first

και επομένως θα επιλέξει τα στοιχεία-παιδιά με ετικέτα first του πατέρα του κόμβου περιβάλλοντος. Έτσι, αν ο κόμβος περιβάλλοντος είναι ο κόμβος 32 του Σχήματος 9.1, (ο οποίος είναι ο κόμβος με ετικέτα last που αντιστοιχεί στο επώνυμο του συγγραφέα του πρώτου από τα άρθρα), τότε ο υπολογισμός του μονοπατιού θα μας δώσει σαν αποτέλεσμα το σύνολο {28} (που περιλαμβάνει μόνο τον κόμβο 28 ο οποίος αντιστοιχεί στο μικρό όνομα του ίδιου συγγραφέα).

❑

Παράδειγμα 9.9. Το μονοπάτι:

child::author[child::last="Kropf"]

μπορεί να γραφεί σε συντετμημένη μορφή ως:

author[last="Kropf"]

και επομένως θα επιλέξει τα στοιχεία-παιδιά του κόμβου περιβάλλοντος τα οποία έχουν ετικέτα author και έχουν ένα ή περισσότερα παιδιά με ετικέτα last και τιμή "Kropf". Έτσι, αν ο κόμβος περιβάλλοντος είναι ο κόμβος 38 του Σχήματος 9.1, τότε ο υπολογισμός του μονοπατιού θα μας δώσει ως αποτέλεσμα το {44} (που περιλαμβάνει τον κόμβο 44 ο οποίος αντιστοιχεί στο συγγραφέα με επώνυμο "Kropf").

❑

9.10. Εκφράσεις και Τελεστές

Η XPath παρέχει ένα σύνολο από τελεστές (αριθμητικούς, τελεστές σύγκρισης, λογικούς τελεστές) για το συνδυασμό εκφράσεων και τη δημιουργία σύνθετων εκφράσεων από απλούστερες.

9.10.1. Λογικοί Τελεστές και Τελεστές Σύγκρισης

Για το συνδυασμό boolean εκφράσεων χρησιμοποιούνται οι λογικοί τελεστές `or` και `and`. Για τον υπολογισμό μιας παράστασης που περιλαμβάνει τον τελεστή `or` υπολογίζεται κάθε μια από τις δύο εκφράσεις που συνδέονται μέσω του τελεστή και η τιμές που θα προκύψουν μετατρέπονται σε boolean. Η (συνολική) έκφραση έχει την τιμή true αν μια τουλάχιστον από τις επιμέρους εκφράσεις πάρει την τιμή true. Διαφορετικά η έκφραση υπολογίζεται σε false. Ανάλογα, στην περίπτωση υπολογισμού έκφρασης που περιλαμβάνει τον τελεστή `and`, η συνολική έκφραση υπολογίζεται σε true μόνο στην περίπτωση που και οι δύο επιμέρους εκφράσεις υπολογίζονται σε true. Διαφορετικά η έκφραση υπολογίζεται σε false.
Οι τελεστές σύγκρισης `=`, `!=`, `<`, `>`, `<=`, και `>=` χρησιμοποιούνται για τη σύγκριση (των τιμών) εκφράσεων και επιστρέφουν τιμή τύπου boolean. Η σημασία των συμβόλων αυτών είναι αυτή που φαίνεται στις αντίστοιχες παρενθέσεις: `=` (ίσο), `!=` (διάφορο), `<` (μικρότερο), `>` (μεγαλύτερο), `<=` (μικρότερο ή ίσο), και `>=` (μεγαλύτερο ή ίσο).
Πρέπει να τονιστεί ότι όταν μια έκφραση η οποία περιλαμβάνει ένα από τα σύμβολα `<` ή `>` πρόκειται να χρησιμοποιηθεί σε ένα XML τεκμήριο, τότε τα σύμβολα `<` και `>` πρέπει να μετατραπούν σε `<` και `>` αντίστοιχα.

9.10.2. Αριθμητικοί Τελεστές

Οι παρακάτω τελεστές χρησιμοποιούνται για αριθμητικές πράξεις:
Ο τελεστής `+` για πρόσθεση.
Ο τελεστής `-` για αφαίρεση.
Ο τελεστής `*` για τον πολλαπλασιασμό.
Ο τελεστής `div` για διαίρεση (πραγματικών αριθμών).
Ο τελεστής `mod` για τον υπολογισμό του υπολοίπου μιας διαίρεσης.

9.11. Η Βιβλιοθήκη Συναρτήσεων της XPath

Η XPath υποστηρίζει πλήθος *συναρτήσεων* (functions) οι οποίες μπορούν να χρησιμοποιηθούν σε εκφράσεις μονοπατιών και οι οποίες πρέπει να υποστηρίζονται από τις διάφορες υλοποιήσεις της γλώσσας. Στην ενότητα αυτή περιγράφουμε ένα μικρό δείγμα τέτοιων συναρτήσεων. Ο αναγνώστης

που ενδιαφέρεται για περισσότερες λεπτομέρειες μπορεί να ανατρέξει στο τεχνικό εγχειρίδιο της γλώσσας [1] για μια πλήρη περιγραφή της βιβλιοθήκης συναρτήσεων της XPath.

9.11.1. Συναρτήσεις χειρισμού κόμβων

Μερικές χαρακτηριστικές συναρτήσεις της κατηγορίας αυτής είναι οι ακόλουθες:

- Η συνάρτηση `last( )`
  Επιστρέφει έναν ακέραιο αριθμό που αντιπροσωπεύει το μέγεθος του περιβάλλοντος.
- Η συνάρτηση `position( )`
  Επιστρέφει έναν ακέραιο αριθμό που αντιπροσωπεύει τη θέση περιβάλλοντος.
- Η συνάρτηση `count(`*node-set*`)`
  Επιστρέφει το πλήθος των κόμβων του συνόλου *node-set.*
- Η συνάρτηση `id(`*object*`)`
  Επιλέγει ένα στοιχείο με βάση το μοναδικό `ID` του. (Η συνάρτηση γενικεύεται και για την περίπτωση που το *object* είναι σύνολο κόμβων.)
- Η συνάρτηση[1] `local-name(`*node-set*`?)`
  Επιστρέφει το τοπικό τμήμα του ονόματος του κόμβου του *node-set* που είναι πρώτος σε εμφάνιση στη διάταξη τεκμηρίου. Σε περίπτωση που το όρισμα *node-set* έχει παραλειφθεί τότε η συνάρτηση εφαρμόζεται στον κόμβο περιβάλλοντος.
- Η συνάρτηση `namespace-uri(`*node-set*`?)`
  Επιστρέφει το URI του επεκταμένου ονόματος του κόμβου του *node-set* που είναι πρώτος σε εμφάνιση στη διάταξη κειμένου. Σε περίπτωση που το όρισμα *node-set* έχει παραλειφθεί τότε η συνάρτηση εφαρμόζεται στον κόμβο περιβάλλοντος.
- Η συνάρτηση `name(`*node-set*`?)`
  Επιστρέφει μια συμβολοσειρά που αντιστοιχεί στο επεκταμένο όνομα του κόμβου του *node-set* που είναι πρώτος σε εμφάνιση στη διάταξη κειμένου. Σε περίπτωση που το όρισμα *node-set* έχει παραλειφθεί τότε η συνάρτηση εφαρμόζεται στον κόμβο περιβάλλοντος.

Παράδειγμα 9.10. Το μονοπάτι τοποθεσίας:

```
child::article[position()=2]/descendant::author[position()=1]
```

επιστρέφει τον πρώτο συγγραφέα του δεύτερου άρθρου που είναι παιδί του κόμβου περιβάλλοντος.

❑

[1] Το σύμβολο ? υποδηλώνει ότι η παρουσία του ορίσματος στη συνάρτηση δεν είναι υποχρεωτική.

Παράδειγμα 9.11. Το μονοπάτι τοποθεσίας:

```
id("tutorial")
```

επιλέγει το στοιχείο με το μοναδικό γνώρισμα τύπου ID το οποίο έχει την τιμή "tutorial". Το μονοπάτι τοποθεσίας:

```
id("tutorial")/title
```

επιλέγει το στοιχείο title που είναι παιδί του στοιχείου με το γνώρισμα τύπου ID το οποίο έχει τη μοναδική (σε όλο το τεκμήριο) τιμή "tutorial".

❑

9.11.2. Συναρτήσεις χειρισμού συμβολοσειρών

Η βιβλιοθήκη συναρτήσεων της XPath περιλαμβάνει επίσης μια ομάδα συναρτήσεων πάνω σε συμβολοσειρές, μεταξύ των οποίων είναι και οι ακόλουθες:

- Η starts-with(*string1,string2*) η οποία επιστρέφει true αν η συμβολοσειρά *string2* είναι αρχικό τμήμα της συμβολοσειράς *string1.*
- Η concat(*string1,string2,…,stringk*) η οποία επιστρέφει τη συνένωση των συμβολοσειρών που αποτελούν τα ορίσματα της.
- Η string-length(*string*) η οποία επιστρέφει τον αριθμό των χαρακτήρων στο *string.* Αν το όρισμα *string* απουσιάζει τότε η συνάρτηση εφαρμόζεται στον κόμβο περιβάλλοντος (αφού μετατραπεί σε συμβολοσειρά).
- Η normalize-space(*string*) η οποία επιστρέφει το όρισμα *string* αφού απαλείψει τα κενά που είναι στην αρχή και στο τέλος της συμβολοσειράς καθώς και τις πολλαπλές εμφανίσεις των ενδιάμεσων κενών.
- Η substring-before(*string1,string2*) η οποία επιστρέφει την υποσυμβολοσειρά της *string1* η οποία προηγείται της πρώτης εμφάνισης της *string2* μέσα στη *string1.*

Παράδειγμα 9.11.
Η substring-before("tutorial","ori") επιστρέφει τη συμβολοσειρά "tut", ενώ η string-length("tutorial") επιστρέφει την τιμή 8 που αντιπροσωπεύει το μήκος της συμβολοσειράς "tutorial".

❑

9.11.3. Άλλες ομάδες συναρτήσεων

Η βιβλιοθήκη συναρτήσεων της XPath περιλαμβάνει επίσης και άλλες ομάδες συναρτήσεων όπως είναι οι *Boolean συναρτήσεις* οι οποίες επιστρέφουν τιμές τύπου boolean, οι *αριθμητικές συναρτήσεις* (number functions), οι οποίες επιστρέφουν για τιμές αριθμούς, π.χ. η συνάρτηση sum(*node-set*) η οποία επιστρέφει έναν αριθμό που αντιστοιχεί στο άθροισμα των κόμβων (αφού

μετατρέψει τις τιμές που είναι συμβολοσειρές σε αριθμούς), κ.α. Ο αναγνώστης που ενδιαφέρεται για περισσότερες πληροφορίες παροτρύνεται να ανατρέξει στο [1].

9.12. Η XPath σαν Γλώσσα Αιτημάτων

Αξίζει να τονιστεί, όπως θα έχει ήδη διαπιστώσει ο αναγνώστης, ότι η XPath συμπεριφέρεται ως μια γλώσσα αιτημάτων, δηλαδή ως γλώσσα με τη βοήθεια της οποίας μπορούμε να διατυπώσουμε αιτήματα (ερωτήματα) σε μια «βάση δεδομένων» η οποία στην πραγματικότητα είναι το XML τεκμήριο. Ο υπολογισμός του μονοπατιού μας επιστρέφει τις απαντήσεις στο αίτημα που αντιστοιχεί στην έκφραση που υπολογίζεται. Αυτή η παρατήρηση είναι πράγματι σωστή, όμως δυστυχώς η XPath από μόνη της δε μας επιτρέπει να διατυπώσουμε πολλές από τις χρήσιμες ερωτήσεις που θα επιθυμούσαμε να μπορούν να απαντηθούν με βάση την πληροφορία που περιλαμβάνει ένα XML τεκμήριο. Όμως η XPath αποτελεί συστατικό στοιχείο ισχυρότερων γλωσσών αιτημάτων καθώς και γλωσσών μετασχηματισμού XML τεκμηρίων όπως είναι η XSLT [3] και η XQuery [5] οι οποίες προσφέρουν την εκφραστική ικανότητα που δεν μπορεί να παράσχει από μόνη της η XPath.

ΚΕΦΑΛΑΙΟ 10:
XSL Transformation (XSLT)

10.1. Εισαγωγή

Η γλώσσα ***E**xtensible **S**tylesheet **L**anguage* (XSL) [4] περιλαμβάνει μια (υπο)*γλώσσα μετασχηματισμού* (transformation language) και μια (υπο)*γλώσσα μορφοποίησης* (formatting language). Οι δύο αυτές γλώσσες μπορούν να χρησιμοποιηθούν και ανεξάρτητα η μια από την άλλη. Η γλώσσα μετασχηματισμού, η XSLT (XSL Transformations) [3], παρέχει δυνατότητες μέσω των οποίων μπορούμε να περιγράψουμε το μετασχηματισμό ενός XML τεκμηρίου σε ένα άλλο XML τεκμήριο. Το XML τεκμήριο που προκύπτει ως αποτέλεσμα του μετασχηματισμού είναι δυνατό να χρησιμοποιεί τις ετικέτες του αρχικού ή ακόμη και να υπακούει στο DTD του αρχικού, είναι όμως επίσης δυνατό να περιλαμβάνει και νέα στοιχεία (με ετικέτες που δεν εμφανίζονται στο αρχικό τεκμήριο ή και ετικέτες που έχουν οριστεί από την XSL και προορίζονται για την περιγραφή του τρόπου εμφάνισης του τεκμηρίου με τη βοήθεια ενός *φυλλομετρητή* (browser)) ή ακόμη να πάρουμε ως αποτέλεσμα τεκμήρια HTML ή τεκμήρια που περιλαμβάνουν ελεύθερο κείμενο.
Στο κεφάλαιο αυτό θα μελετήσουμε τη γλώσσα XSLT.

10.2. Η βασική ιδέα της XSLT

Με τη γλώσσα XSLT μπορούν να διατυπωθούν περιγραφές του μετασχηματισμού ενός XML τεκμηρίου οι οποίες ονομάζονται *φύλλα στυλ* (XSLT stylesheet). Ένας *επεξεργαστής XSLT* (XSLT processor) αναλαμβάνει να επεξεργαστεί ένα XML τεκμήριο εφαρμόζοντας πάνω του το φύλλο στυλ και να παράγει ένα νέο τεκμήριο.
Ένα φύλλο στυλ εκφράζεται και αυτό ως ένα XML τεκμήριο το οποίο μπορεί να περιλαμβάνει στοιχεία-εντολές της XSLT καθώς επίσης και στοιχεία που δεν ορίζονται στην XSLT. Τα στοιχεία-εντολές της XSLT, καθοδηγούν τον επεξεργαστή στην παραγωγή του αποτελέσματος. Το αποτέλεσμα του μετασχηματισμού μπορεί να είναι:

1. Ένα νέο XML τεκμήριο.
2. Ένα τεκμήριο διατυπωμένο σε HTML.
3. (Με λίγη παραπάνω προσπάθεια) Ελεύθερο κείμενο.

Το μοντέλο του τεκμηρίου πάνω στο οποίο βασίζεται η φιλοσοφία του XSLT είναι αυτό του διατεταγμένου XML δέντρου που γνωρίσαμε στο Κεφάλαιο 9.

Η XSLT λειτουργεί μετατρέποντας ένα XML δέντρο σε ένα άλλο XML δέντρο. Αυτό επιτυγχάνεται λόγω του ότι η XSLT διαθέτει τελεστές υπό τη μορφή XML στοιχείων οι οποίοι έχουν ειδική σημασία για τον επεξεργαστή XSLT, και μπορούν να χρησιμοποιηθούν για την επιλογή, την αναδιάταξη και αποστολή στην έξοδο κόμβων ενός XML δέντρου (πάνω στο οποίο εφαρμόζεται το φύλλο στυλ) ενδεχομένως μαζί με νέα στοιχεία και άλλα δομικά στοιχεία της XML τα οποία προστίθενται στο τεκμήριο εξόδου από την εφαρμογή του φύλλου στυλ.

Ένα φύλλο στυλ περιγράφει κανόνες για τη μετατροπή του αρχικού XML δέντρου στο τελικό δέντρο. Ο μετασχηματισμός που περιγράφεται στο φύλλο στυλ επιτυγχάνεται μέσω της συσχέτισης *προτύπων* ή *υποδειγμάτων* (patterns) με *οδηγούς* (templates) οι οποίοι συγκεκριμενοποιούνται για να προκύψει το αποτέλεσμα της εφαρμογής του κανόνα. Ο παραπάνω μηχανισμός εκφράζεται ως ένα σύνολο από *κανόνες-οδηγούς* (template rules). Κάθε κανόνας-οδηγός περιλαμβάνει ένα πρότυπο μέσω του οποίου προσδιορίζονται οι κόμβοι στους οποίους εφαρμόζεται ο κανόνας και έναν οδηγό μέσω του οποίου προδιαγράφεται το αποτέλεσμα της εφαρμογής του κανόνα. Ο οδηγός μπορεί με τη σειρά του να περιλαμβάνει οδηγίες με τη μορφή στοιχείων της XSLT οι οποίες περιγράφουν τον τρόπο δημιουργίας τμημάτων του τεκμηρίου του αποτελέσματος. Κάθε φορά που ένας οδηγός συγκεκριμενοποιείται, αυτό γίνεται ως προς έναν *τρέχοντα κόμβο* (current node) και μια *τρέχουσα λίστα κόμβων* (current node list).

Η εφαρμογή ενός φύλλου στυλ γίνεται ως εξής. Ο επεξεργαστής XSLT διατρέχει το δέντρο του XML τεκμηρίου και επεξεργάζεται έναν-έναν τους κόμβους του. Κατά την επεξεργασία ενός κόμβου, ο κόμβος ελέγχεται εάν ταιριάζει με το πρότυπο κάθε κανόνα-οδηγού στο φύλλο στυλ. Στην περίπτωση που ο κόμβος ταιριάζει με το πρότυπο ενός κανόνα τότε εφαρμόζεται ο κανόνας και παράγεται ως αποτέλεσμα ο οδηγός του κανόνα αφού συμπληρωθεί κατάλληλα (με τρόπο που θα δούμε στη συνέχεια). Σε περίπτωση που ένας κόμβος ταιριάζει με τα πρότυπα περισσοτέρων του ενός κανόνων τότε επιλέγεται ένας από τους κανόνες αυτούς για να εφαρμοστεί. Η διαδικασία επιλογής του κανόνα αυτού (η οποία αναφέρεται και ως *επίλυση σύγκρουσης*) θα εξεταστεί σε επόμενη παράγραφο.

10.3. Η δομή ενός φύλλου στυλ

Η γλώσσα XSLT χρησιμοποιεί την XML ως γλώσσα για τη διατύπωση των φύλλων στυλ. Η ρίζα ενός φύλλου στυλ είναι ένα στοιχείο η ετικέτα του οποίου είναι η `stylesheet` και ανήκει στο χώρο ονομάτων με URI το `http://www.w3.org/1999/XSL/Transform`. Στο κεφάλαιο αυτό θα χρησιμοποιήσουμε το `xsl` ως πρόθεμα για τα στοιχεία που ανήκουν στο χώρο ονομάτων της XSLT. Η γενική μορφή ενός στοιχείου `stylesheet` είναι η ακόλουθη:

```
<xsl:stylesheet
        id = id
        extension-element-prefixes = tokens
        exclude-result-prefixes = tokens
        version = number>
        … περιεχόμενο του στοιχείου ...
</xsl:stylesheet>
```

Από τα γνωρίσματα του στοιχείου **xsl:stylesheet** υποχρεωτικά είναι τα γνωρίσματα **version** και **xmlns:xsl**. Το γνώρισμα **version** έχει την τιμή **1.0** ενώ το **xmlns:xsl** έχει την τιμή **http://www.w3.org/1999/XSL/Transform**. Με να υπόλοιπα (προαιρετικά γνωρίσματα) του στοιχείου **xsl:stylesheet** δε θα ασχοληθούμε στα πλαίσια του παρόντος κεφαλαίου.
Το περιεχόμενο του στοιχείου **xsl:stylesheet** μπορεί να είναι ένα ή περισσότερα από τα παρακάτω στοιχεία:

- xsl:import
- xsl:include
- xsl:strip-space
- xsl:preserve-space
- xsl:output
- xsl:key
- xsl:decimal-format
- xsl:namespace-alias
- xsl:attribute-set
- xsl:variable
- xsl:param
- xsl:template

Τα στοιχεία που είναι παιδιά του **xsl:stylesheet** θα λέμε ότι είναι *στοιχεία ανωτάτου επιπέδου* (top-level elements). Η σειρά των στοιχείων αυτών σε ένα φύλλο στυλ δεν παίζει κάποιο ιδιαίτερο ρόλο εκτός από τα στοιχεία με ετικέτα **xsl:import**.
Ως συνώνυμο του στοιχείου **xsl:stylesheet** είναι δυνατό να χρησιμοποιούμε το στοιχείο **xsl:transform** το οποίο έχει την ίδια ακριβώς δομή με το στοιχείο **xsl:stylesheet**.

Παράδειγμα 10.1. Το ακόλουθο XML τεκμήριο είναι ένα φύλλο στυλ:

```
<?xml version="1.0"?>
<xsl:transform version="1.0"
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
            ... περιεχόμενο του στοιχείου ...
</xsl:transform>
```

❑

10.4. Τρόποι εφαρμογής ενός φύλλου στυλ

Ένα φύλλο στυλ εφαρμόζεται σε ένα XML τεκμήριο με έναν από τους ακόλουθους τρόπους:

1. Το XML τεκμήριο μαζί με το φύλλο στυλ στέλνονται σε έναν (πελάτη) φυλλομετρητή ο οποίος μετασχηματίζει το XML τεκμήριο ακολουθώντας τις οδηγίες που περιλαμβάνονται στο φύλλο στυλ και στη συνέχεια εμφανίζει το αποτέλεσμα στο χρήστη.
2. Ο εξυπηρετητής εφαρμόζει το φύλλο στυλ στο XML τεκμήριο και το μετασχηματίζει σε κάποια άλλη μορφή (συνήθως HTML). Στη συνέχεια το αποτέλεσμα αποστέλλεται στον πελάτη (φυλλομετρητή).
3. Ένα τρίτο πρόγραμμα μετασχηματίζει το αρχικό XML τεκμήριο σε κάποια άλλη μορφή (συνήθως HTML) πριν το τεκμήριο τοποθετηθεί στον εξυπηρετητή. Τόσο ο εξυπηρετητής όσο και ο πελάτης χειρίζονται πλέον το μετασχηματισμένο τεκμήριο.

Κάθε μια από τις παραπάνω προσεγγίσεις χρησιμοποιεί διαφορετικό λογισμικό. Όλες οι προσεγγίσεις όμως χρησιμοποιούν τα ίδια XML τεκμήρια και XSLT φύλλα στυλ.

10.4.1. Απευθείας προβολή των XML τεκμηρίων σε ένα φυλλομετρητή

Όπως αναφέρθηκε προηγουμένως το XML τεκμήριο μαζί με το φύλλο στυλ είναι δυνατό να αποσταλεί σε έναν (πελάτη) φυλλομετρητή ο οποίος μετασχηματίζει το XML τεκμήριο ακολουθώντας τις οδηγίες που περιλαμβάνονται στο φύλλο στυλ και στη συνέχεια προβάλλει το αποτέλεσμα στο χρήστη. Στην περίπτωση αυτή περιορίζεται ο φόρτος εργασίας για τον εξυπηρετητή ενώ αυξάνει για τον πελάτη. Πρέπει να σημειωθεί ότι το φύλλο στυλ οφείλει στην περίπτωση αυτή να μετατρέψει το XML τεκμήριο σε μορφή που να γίνεται κατανοητή από τον πελάτη (π.χ. HTML) αν αυτός δεν κατανοεί απευθείας τις οδηγίες μορφοποίησης της XSL.
Για να επισυναφθεί ένα φύλλο στυλ σε ένα XML τεκμήριο θα πρέπει να συμπεριληφθεί στο XML τεκμήριο αμέσως μετά τη δήλωση XML, μια οδηγία επεξεργασίας `xml-stylesheet`. Σε αυτήν τη δήλωση η τιμή του γνωρίσματος `type` είναι η `text/xml` ενώ η τιμή του γνωρίσματος `href` είναι ένα URL το οποίο δείχνει το φύλλο στυλ[2].

Παράδειγμα 10.2. Στο παρακάτω XML τεκμήριο έχει επισυναφθεί το φύλλο στυλ που βρίσκεται στο αρχείο `http://www.ionio.gr/style1.xsl`.

```
<?xml version="1.0"?>
```

[2] Πάντως καλό είναι να εξετάσει κανείς τις τυχόν ιδιαιτερότητες του φυλλομετρητή στον οποίο θέλει να απευθυνθεί μιας και υπάρχουν ακόμη αρκετά προβλήματα συμβατότητας.

```
<?xml-stylesheet type="text/xml" href="http://www.ionio.gr/style1.xsl"?>
```

... ακολουθούν τα στοιχεία του XML τεκμηρίου ...

❑

10.5. Κανόνες οδηγοί

Οι *κανόνες-οδηγοί* (template rules) ορίζονται με τη βοήθεια του στοιχείου xsl:template και αποτελούν το σημαντικότερο κομμάτι της XSLT. Οι κανόνες-οδηγοί συσχετίζουν το αποτέλεσμα το οποίο παράγουν με το (τμήμα του) τεκμήριο το οποίο επεξεργάζονται. Κάθε κανόνας οδηγός περιλαμβάνει ένα *πρότυπο ή υπόδειγμα* (pattern) και έναν *οδηγό* (template). Ο ρόλος του προτύπου είναι να προδιαγράψει το σύνολο των κόμβων πάνω στο οποίο εφαρμόζεται ο κανόνας-οδηγός, μέσω ενός συνόλου περιορισμών τους οποίους πρέπει να πληρούν οι κόμβοι οι οποίοι ταιριάζουν με το πρότυπο. Ο ρόλος του οδηγού είναι να προδιαγράψει το αποτέλεσμα (τη μορφή και τις οδηγίες διαμόρφωσης του) το οποίο θα παραχθεί εφόσον εφαρμοστεί ο συγκεκριμένος κανόνας-οδηγός.

10.5.1. Τα πρότυπα

Για τη διατύπωση των προτύπων χρησιμοποιείται ένα υποσύνολο των εκφράσεων της γλώσσας XPath. Πιο συγκεκριμένα, ένα πρότυπο είναι ένα σύνολο από *πρότυπα μονοπατιών τοποθεσίας* (location path patterns) τα οποία χωρίζονται μεταξύ τους με το σύμβολο |. Ένα πρότυπο μονοπατιού τοποθεσίας είναι ένα μονοπάτι τοποθεσίας στα βήματα τοποθεσίας του οποίου οι μόνοι άξονες οι οποίοι επιτρέπεται να περιλαμβάνονται είναι οι άξονες child και attribute. Αντίθετα, στα κατηγορήματα που περιλαμβάνονται στα πρότυπα μονοπατιών τοποθεσίας μπορούν τα οποία να περιλαμβάνουν όλες τις εκφράσεις που επιτρέπονται στα κατηγορήματα των μονοπατιών τοποθεσίας της XPath. Στα πρότυπα μονοπατιών τοποθεσίας είναι επίσης επιτρεπτό να περιλαμβάνονται οι τελεστές / και // καθώς επίσης και να ξεκινούν με κλήση των συναρτήσεων id ή key. Ο υπολογισμός ενός προτύπου καταλήγει σε ένα σύνολο κόμβων. Ένας κόμβος θα λέμε ότι *ταιριάζει* (matches) με ένα πρότυπο αν υπάρχει πιθανό περιβάλλον τέτοιο ώστε ο κόμβος να περιλαμβάνεται στο σύνολο των κόμβων που προκύπτει από τον υπολογισμό της έκφρασης του προτύπου ως προς το περιβάλλον αυτό. Το σύμβολο | ανάμεσα στα πρότυπα μονοπατιών τοποθεσίας ερμηνεύεται ως **λογικό ή** και επομένως ένας κόμβος ταιριάζει με ένα πρότυπο αν ταιριάζει με κάποιο από τα πρότυπα μονοπατιών τοποθεσίας τα οποία περιλαμβάνει το πρότυπο. Τα βήματα τοποθεσίας σε ένα πρότυπο μονοπατιού τοποθεσίας υπολογίζονται από δεξιά προς τα αριστερά. Έτσι, ένας κόμβος ταιριάζει με ένα πρότυπο αν ταιριάζει με το δεξιότερο βήμα τοποθεσίας του και ταυτόχρονα υπάρχει κατάλληλα επιλεγμένος κόμβος (ο οποίος είναι ο πατέρας του κόμβου αν πριν από το εν λόγω βήμα τοποθεσίας βρίσκεται το

σύμβολο / ή κάποιος πρόγονος του κόμβου αν πριν από το βήμα τοποθεσίας βρίσκεται το σύμβολο //) ο οποίος ταιριάζει με το πρότυπο που προκύπτει μετά την αφαίρεση του δεξιότερου βήματος τοποθεσίας του αρχικού προτύπου.

Παράδειγμα 10.3. Το πρότυπο:

```
articles//authors/author[1]
```

ταιριάζει με έναν κόμβο στοιχείου, αν ο κόμβος αυτός έχει ετικέτα author, είναι το πρώτο παιδί του πατέρα του ο οποίος είναι ο κόμβος στοιχείου με ετικέτα authors, και αυτός ο κόμβος authors έχει κάποιον πρόγονο με ετικέτα articles. Έτσι, στο δέντρο του Σχήματος 9.1, οι κόμβοι οι οποίοι ταιριάζουν με το πιο πάνω πρότυπο είναι οι κόμβοι 27 και 39.

❑

10.5.2. Οι οδηγοί

Οι οδηγοί αποτελούν προδιαγραφές του αποτελέσματος το οποίο προκύπτει από την εφαρμογή ενός κανόνα-οδηγού. Ένας οδηγός είναι δυνατό να περιέχει κείμενο ή/και στοιχεία της XML τα οποία θα εμφανιστούν στο τεκμήριο που θα προκύψει ως αποτέλεσμα. Είναι επίσης δυνατόν να περιέχει και εντολές της XSLT (με τη μορφή στοιχείων XML) οι οποίες καθορίζουν ποια τμήματα του τεκμηρίου εισόδου θα αντιγραφούν ή θα συμβάλλουν με κάποιο τρόπο στη διαμόρφωση του αποτελέσματος. Οι εντολές της XSLT αναγνωρίζονται σε σχέση με τα υπόλοιπα XML στοιχεία από το πρόθεμα του χώρου ονομάτων της XSLT με το οποίο συνοδεύονται. Ένας οδηγός, αφού συγκεκριμενοποιηθεί, δηλαδή αφού αντικατασταθούν κατάλληλα εκείνα τα τμήματα του τα οποία αποτελούν εντολές, με τα αποτελέσματα που προκύπτουν από την εκτέλεση των εντολών αυτών, αποστέλλεται στη έξοδο και αποτελεί τμήμα του αποτελέσματος που παράγεται από την εφαρμογή του μετασχηματισμού που περιγράφεται στο φύλλο στυλ.

10.5.3. Η μορφή των κανόνων-οδηγών

Η γενική μορφή ενός κανόνα οδηγού είναι η ακόλουθη:

```
<xsl:template
        match = pattern
        name = qname
        priority = number
        mode = qname>
        … περιεχόμενο του στοιχείου ...
</xsl:template>
```

Όλα τα γνωρίσματα του στοιχείου **xsl:template** είναι προαιρετικά εκτός από το γνώρισμα **match** το οποίο εμφανίζεται υποχρεωτικά (εκτός εάν υπάρχει το γνώρισμα **name**). Η τιμή του γνωρίσματος **match** είναι ένα πρότυπο το οποίο καθορίζει το σύνολο των κόμβων στους οποίους εφαρμόζεται ο κανόνας. Το περιεχόμενο του στοιχείου **xsl:template** αποτελεί τον οδηγό από τη συμπλήρωση του οποίου θα προκύψει το αποτέλεσμα.

Παράδειγμα 10.4. Ο κανόνας-οδηγός που ακολουθεί:

```
<xsl:template match="/">
        <html>
                <head>Δοκιμή</head>
                <body>Ο πρώτος μου μετασχηματισμός σε XSLT</body>
        </html>
</xsl:template>
```

εφαρμόζεται στον κόμβο της ρίζας του δέντρου εισόδου. Ο επεξεργαστής της XSLT διαβάζει το τεκμήριο εισόδου και το πρώτο στοιχείο που συναντά είναι η ρίζα και επομένως οι περιορισμοί του προτύπου ικανοποιούνται. Ως αποτέλεσμα της εφαρμογής του κανόνα ο επεξεργαστής της XSLT στέλνει στην έξοδο τον οδηγό (ο οποίος δεν περιλαμβάνει εντολές της XSLT αλλά αποτελεί ένα καλά δομημένο HTML τεκμήριο):

```
<html>
        <head>Δοκιμή</head>
        <body>Ο πρώτος μου μετασχηματισμός σε XSLT</body>
</html>
```

❑

10.6. Η εφαρμογή των κανόνων οδηγών

Όπως αναφέρθηκε στην Ενότητα 10.2, ο επεξεργαστής XSLT διατρέχει το δέντρο του XML τεκμηρίου και επεξεργάζεται έναν-έναν τους κόμβους του δέντρου ελέγχοντας αν ο κόμβος ταιριάζει με τα πρότυπα των κανόνων-οδηγών στο φύλλο στυλ. Στην περίπτωση που ένας κόμβος ταιριάζει με το πρότυπο ενός κανόνα τότε παράγεται ως αποτέλεσμα ο οδηγός του κανόνα αφού συμπληρωθεί κατάλληλα. Ένας οδηγός συχνά περιλαμβάνει άλλες εντολές με τη μορφή στοιχείων της XSLT, οι οποίες επιτρέπουν την επεξεργασία όλου του XML δέντρου ή τμημάτων του δέντρου αυτού. Ένα στοιχείο της XSLT που συναντάμε συχνά στους οδηγούς είναι το στοιχείο **xsl:apply-templates**. Η γενική μορφή του στοιχείου **xsl:apply-templates** είναι η ακόλουθη:

```
<xsl:apply-templates
        select = node-set-expression
        mode = qname>
        … περιεχόμενο του στοιχείου ...
</xsl:apply-templates>
```

Όλα τα γνωρίσματα του στοιχείου **xsl:apply-templates** είναι προαιρετικά. Όταν ένα στοιχείο **xsl:apply-templates** περιλαμβάνεται στον οδηγό ενός στοιχείου **xsl:template** τότε αυτό οδηγεί στην επεξεργασία των κόμβων-παιδιών του κόμβου στον οποίο εφαρμόζεται το **xsl:template**. Κάθε κόμβος-παιδί αντιμετωπίζεται σαν πλήρες XML δέντρο. Πετυχαίνουμε έτσι την αναδρομική εφαρμογή των κανόνων οδηγών στους κόμβους σε όλο το μήκος του XML τεκμηρίου. Αξίζει να σημειωθεί ότι είναι δυνατό να εμφανίζονται περισσότερα του ενός στοιχεία **xsl:apply-templates** στα πλαίσια ενός οδηγού.

Παράδειγμα 10.5. Το τμήμα XSLT τεκμηρίου που ακολουθεί:

```
<xsl:template match="/book/articles">
   <result>
      <xsl:apply-templates/>
   </result>
</xsl:template>
```

δημιουργεί (εφόσον εφαρμοστεί στο XML τεκμήριο του Σχήματος 9.1) ένα στοιχείο με ετικέτα **result** για το στοιχείο **articles** του τεκμηρίου και στη συνέχεια επεξεργάζεται τους κόμβους παιδιά του στοιχείου **articles** (χρησιμοποιώντας τους κανόνες-οδηγούς που υπάρχουν στο φύλλο στυλ οι οποίοι μπορούν να εφαρμοστούν στους κόμβους αυτούς) και τοποθετεί το αποτέλεσμα που προκύπτει από αυτήν την επεξεργασία ως περιεχόμενο του στοιχείου **result**.

❑

10.6.1. Το γνώρισμα "select" του "xsl:apply-templates"

Όπως είδαμε προηγουμένως στις περιπτώσεις που το γνώρισμα **select** έχει παραλειφθεί, το στοιχείο **xsl:apply-templates** επεξεργάζεται αναδρομικά όλα τα παιδιά του τρέχοντος κόμβου. Αντίθετα, στην περίπτωση που το στοιχείο **xsl:apply-templates** διαθέτει το γνώρισμα **select** τότε μέσω τις τιμής του γνωρίσματος αυτού, που είναι μια έκφραση η οποία υπολογίζεται σε ένα σύνολο κόμβων, επιλέγονται οι κόμβοι στους οποίους θα εφαρμοστούν στη συνέχεια οι κανόνες-οδηγοί. Το γνώρισμα **select** χρησιμοποιεί πρότυπα όπως και το γνώρισμα **match** του στοιχείου **xsl:template**.

Παράδειγμα 10.6. Το τμήμα XSLT τεκμηρίου που ακολουθεί:

```
<xsl:template match="/book/articles">
   <result>
     <xsl:apply-templates select="//author"/>
   </result>
</xsl:template>
```

επεξεργάζεται όλους τους απογόνους του στοιχείου articles με ετικέτα author (εφαρμόζοντας κατάλληλους κανόνες οδηγούς οι οποίοι δε φαίνονται στο παράδειγμα).

❑

Παράδειγμα 10.7. Το τμήμα XSLT τεκμηρίου που ακολουθεί:

```
<xsl:template match="/book/editors">
   <first_editor>
     <xsl:apply-templates select="editor[position()=1]"/>
   </first_editor>
   <second_editor>
     <xsl:apply-templates select="editor[position()=2]"/>
   </second_editor>
</xsl:template>

<xsl:template match="editor">
        <xsl:value-of select="."/>
</xsl:template>
```

αν εφαρμοστεί στο XML τεκμήριο του Πίνακα 9.1 θα δώσει ως αποτέλεσμα το ακόλουθο τμήμα XML τεκμηρίου:

```
<first_editor>Manolis Gergatsoulis</first_editor>
<second_editor>Panos Rondogiannis</second_editor>
```

❑

Αξίζει να αναφερθεί ότι μέσω του γνωρίσματος select είναι δυνατό να προκαλέσουμε την επεξεργασία κόμβων οι οποίοι δεν είναι κατ' ανάγκη απόγονοι του τρέχοντος κόμβου. Συνήθως όμως το στοιχείο xsl:apply-templates χρησιμοποιείται για την επεξεργασία στοιχείων τα οποία είναι απόγονοι του τρέχοντος κόμβου. Αυτό εξασφαλίζει ότι δε θα οδηγηθούμε σε υπολογισμούς που δεν τερματίζονται ποτέ.

10.7. Επίλυση Σύγκρουσης κανόνων-οδηγών

Κατά την επεξεργασία ενός XML δέντρου αναζητούνται οι κανόνες-οδηγοί το πρότυπο των οποίων ταιριάζει με τον κόμβο που βρίσκεται κάθε φορά υπό επεξεργασία. Είναι όμως πιθανόν να ταιριάζουν με το συγκεκριμένο κόμβο περισσότεροι του ενός κανόνες-οδηγοί από αυτούς που περιλαμβάνονται στο φύλλο στυλ. Στην περίπτωση αυτή θα πρέπει να επιλεγεί ένας μόνο από τους κανόνες αυτούς για να εφαρμοστεί στο συγκεκριμένο κόμβο. Κάθε φορά που επιλέγεται να εφαρμοστεί ένας κανόνας οδηγός αυτός ο κανόνας γίνεται ο *τρέχον κανόνας οδηγός* (current template rule). Η διαδικασία επιλογής του τρέχοντος κανόνα οδηγού ονομάζεται *επίλυση σύγκρουσης* (conflict resolution). Η επιλογή του κατάλληλου κανόνα γίνεται κατά τρόπον ώστε να επιλέγεται κάθε φορά ο κανόνας με τη μεγαλύτερη προτεραιότητα.
Η προτεραιότητα ενός κανόνα-οδηγού καθορίζεται μέσω του γνωρίσματος priority. Το γνώρισμα αυτό παίρνει για τιμή ένα (θετικό ή αρνητικό) πραγματικό (real) αριθμό. Δεδομένου ότι το γνώρισμα priority δεν είναι υποχρεωτικό και συχνά το γνώρισμα αυτό παραλείπεται από τους κανόνες οδηγούς, η XSLT έχει καθιερώσει κάποιους κανόνες για τον υπολογισμό της *προκαθορισμένης προτεραιότητας* (default priority) κάθε κανόνα.
Λεπτομέρειες σχετικά με τον καθορισμό της προτεραιότητας των κανόνων-οδηγών παρέχονται στην ενότητα 5.5. του εγχειριδίου των προδιαγραφών της XSLT [3].

10.8. Το γνώρισμα "mode"

Τα στοιχεία xsl:template και xsl:apply-templates είναι δυνατό να διαθέτουν το γνώρισμα mode, το οποίο είναι προαιρετικό και για τα δύο αυτά στοιχεία. Ένα στοιχείο xsl:apply-templates το οποίο διαθέτει ένα γνώρισμα mode εφαρμόζεται μόνο σε στοιχεία xsl:template τα οποία επίσης διαθέτουν το γνώρισμα mode με την ίδια τιμή, ενώ αντιθέτως ένα στοιχείο xsl:apply-templates το οποίο δε διαθέτει το γνώρισμα mode εφαρμόζεται μόνο σε στοιχεία xsl:template τα οποία επίσης δε διαθέτουν αυτό το γνώρισμα.
Αξίζει να σημειωθεί ότι μέσω της χρήσης του γνωρίσματος mode μπορούμε να επιτύχουμε την επεξεργασία ενός στοιχείου πολλές φορές έτσι ώστε κάθε φορά να παράγεται διαφορετικό αποτέλεσμα (το αποτέλεσμα να μορφοποιείται κατά διαφορετικό τρόπο).

Παράδειγμα 10.8. Στο τμήμα φύλλου στυλ που ακολουθεί φαίνεται η χρήση του γνωρίσματος mode για την εφαρμογή δύο διαφορετικών μορφοποιήσεων στο αποτέλεσμα:

```
<xsl:template match="articles">
  <HTML>
```

```
    <HEAD><TITLE>Articles included in the book</TITLE></HEAD>
    <BODY>
        <H2>Titles of the articles</H2>
        <UL>
              <xsl:apply-templates select="article" mode="simple"/>
        </UL>
        <H2>Detailed descriptions of the articles</H2>
        <UL>
              <xsl:apply-templates select="article" mode="full"/>
        </UL>
    </BODY>
  </HTML>
</xsl:template>

<xsl:template match="article" mode="simple">
  <LI><A>
      <xsl:value-of select="title"/>
  </A></LI>
</xsl:template>


<xsl:template match="article" mode="full">
  <LI><A>
    <xsl:for-each select="authors/author">
      <xsl:value-of select="first"/>
      <xsl:text> </xsl:text>
      <xsl:value-of select="last"/>,
      </xsl:for-each>
      &amp;quot;<xsl:value-of select="title"/>&amp;quot;.
      In M. Gergatsoulis, P. Rondogiannis (editors),
      <I>Intensional Programming II,</I> World Scientific, 2000.
  </A></LI>
</xsl:template>
```

Η εφαρμογή του πιο πάνω φύλλου στυλ στο XML τεκμήριο του Πίνακα 9.1. θα δημιουργήσει ως αποτέλεσμα ένα HTML τεκμήριο η προβολή του οποίου σε ένα φυλλομετρητή θα έχει το ακόλουθο αποτέλεσμα:

Titles of the articles

- Intensional Logic in Context
- Intensional Communities

Detailed descriptions of the articles

- William Wadge, "Intensional Logic in Context". In M. Gergatsoulis, P. Rondogiannis (editors), *Intensional Programming II,* World Scientific, 2000.
- John Plaice, Peter Kropf, "Intensional Communities". In M. Gergatsoulis, P. Rondogiannis (editors), *Intensional Programming II,* World Scientific, 2000.

❑

10.9. Ενσωματωμένοι κανόνες-οδηγοί

Η XSLT διαθέτει ένα σύνολο από *ενσωματωμένους κανόνες-οδηγούς* (built-in template rules) οι οποίοι, αν και δεν έχουν οριστεί από το χρήστη θεωρείται ότι περιέχονται σε κάθε φύλλο στυλ. Στην πραγματικότητα συμπεριφέρονται ως εάν να έχουν εισαχθεί στην αρχή του φύλλου στυλ με μια δήλωση xsl:import. Οι ενσωματωμένοι κανόνες-οδηγοί της XSLT είναι γενικοί για να εφαρμόζονται στις διάφορες κατηγορίες δομικών στοιχείων κάθε XML τεκμηρίου.

10.9.1. Ενσωματωμένος κανόνας για κόμβους στοιχείων

Ο ενσωματωμένος κανόνας-οδηγός που ακολουθεί εφαρμόζεται στη ρίζα και σε κάθε στοιχείο του XML δέντρου:

```
<xsl:template match="*|/">
    <xsl:apply-templates/>
</xsl:template>
```

και στοχεύει στο να εξασφαλίσει ότι κάθε στοιχείο του XML δέντρου θα τύχει τελικά επεξεργασίας ακόμη και αν δεν έχουμε συμπεριλάβει κάποιο σχετικό κανόνα-οδηγό στο φύλλο στυλ.

10.9.2. Ενσωματωμένος κανόνας για κόμβους γνωρισμάτων και κόμβους κειμένου

Ο ενσωματωμένος κανόνας-οδηγός που ακολουθεί εφαρμόζεται σε κόμβους γνωρισμάτων και κόμβους κειμένου:

```
<xsl:template match="text( )|@*">
    <xsl:value-of select="."/>
</xsl:template>
```

Το αποτέλεσμα της εφαρμογής του κανόνα είναι η αποστολή στην έξοδο της τιμής του κόμβου κειμένου ή της τιμής του γνωρίσματος του κόμβου στον οποίο εφαρμόζεται.

10.9.3. Ενσωματωμένος κανόνας για κόμβους σχολίων και οδηγιών επεξεργασίας

Ο ενσωματωμένος κανόνας-οδηγός που ακολουθεί εφαρμόζεται σε κόμβους σχολίων και σε κόμβους οδηγιών επεξεργασίας:

```
<xsl:template match="processing-instruction( )|comment( )"/>
```

Αυτός ο κανόνας δεν κάνει τίποτα (απλά αγνοεί τους κόμβους στους οποίους εφαρμόζεται και δεν τους εμφανίζει στην έξοδο).

10.9.4. Ενσωματωμένοι κανόνες για κάθε τιμή του γνωρίσματος "mode"

Για κάθε πιθανή τιμή του γνωρίσματος mode θεωρείται ότι υπάρχει και ένας ενσωματωμένος κανόνας ο οποίος, σε περίπτωση έλλειψης σχετικών κανόνων-οδηγών, εξασφαλίζει την αναδρομική συνέχιση της επεξεργασίας των κόμβων (στοιχείων) με την ίδια τιμή του γνωρίσματος mode. Έτσι, για μια τιμή "m" του γνωρίσματος mode θεωρείται ότι υπάρχει ένας ενσωματωμένος κανόνας της μορφής:

```
<xsl:template match="*|/" mode="m">
    <xsl:apply-templates mode="m" />
</xsl:template>
```

10.10. Στοιχεία της XSLT για τη δημιουργία του αποτελέσματος

Η XSLT διαθέτει μια σειρά από στοιχεία τα οποία συμβάλουν στην παραγωγή του αποτελέσματος συμπληρώνοντας κατάλληλα τον οδηγό.

10.10.1. Το στοιχείο "xsl:value-of" και ο υπολογισμός τιμών των κόμβων

Το στοιχείο xsl:value-of της XSLT υπολογίζει την τιμή κάποιου κόμβου και στέλνει το αποτέλεσμα στην έξοδο (στο τεκμήριο που αποτελεί την έξοδο του μετασχηματισμού). Το αποτέλεσμα της εντολής xsl:value-of είναι τύπου text. Ο κόμβος η τιμή του οποίου υπολογίζεται αναφέρεται πάντα ως προς τον τρέχοντα κόμβο. Ο κόμβος αυτός προσδιορίζεται μέσω του γνωρίσματος

select του στοιχείου xsl:value-of. Στην περίπτωση που υπάρχουν πολλοί πιθανοί κόμβοι οι οποίοι μπορούν να επιλεγούν για να υπολογιστεί η τιμή τους μέσω ενός στοιχείου xsl:value-of, τότε επιλέγεται ο πρώτος από αυτούς.

Παράδειγμα 10.9. Στο παρακάτω φύλλο στυλ φαίνεται η χρήση του στοιχείου xsl:value-of:

```
<xsl:template match="*">
  <name>
    <xsl:value-of select="articles//author/first"/>
    <xsl:text> </xsl:text>
    <xsl:value-of select="articles//author/last"/>
  </name>
</xsl:template>
```

Το αποτέλεσμα από την εφαρμογή του κανόνα αυτού στο δέντρο του Σχήματος 9.1 είναι το στοιχείο:

```
<name>William Wadge</name>
```

❑

Παρατηρήστε στο Παράδειγμα 10.9 ότι έχει υπολογιστεί μόνο η τιμή του κόμβου που σχετίζεται με τον πρώτο συγγραφέα άρθρου που εμφανίζεται στο XML τεκμήριο.
Γενικά υπάρχουν δύο τρόποι για την αντιμετώπιση του προβλήματος αυτού (της εμφάνισης των τιμών όλων των υποψήφιων κόμβων και όχι μόνο του πρώτου). Ο πρώτος τρόπος είναι να αναφερθούμε μέσα από το γνώρισμα select ακριβώς στους κόμβους των οποίων η τιμή επιθυμούμε να υπολογιστεί. Ο δεύτερος τρόπος είναι να χρησιμοποιήσουμε το στοιχείο xsl:for-each το οποίο, όπως θα δούμε σε επόμενη παράγραφο επεξεργάζεται με τη σειρά καθένα από τα στοιχεία τα οποία επιλέγονται από το γνώρισμα select το οποίο διαθέτει.

10.10.2. Στοιχεία της XSLT για τη δημιουργία κόμβων

Μια σειρά στοιχείων της XSLT παρέχουν δυνατότητες δημιουργίας κόμβων διαφόρων τύπων στο τεκμήριο που προκύπτει από την εφαρμογή του φύλλου στυλ. Στην κατηγορία αυτή ανήκουν τα στοιχεία.

xsl:element
xsl:attribute
xsl:attribute-set
xsl:processing-instruction
xsl:comment
xsl:text

Στη συνέχεια θα περιγράψουμε τον τρόπο που εφαρμόζονται τα στοιχεία αυτά.

10.10.2.1 Δημιουργία στοιχείων με το "xsl:element"

Όπως είδαμε στα παραδείγματα που προηγήθηκαν είναι δυνατό να δημιουργήσουμε στοιχεία στο αποτέλεσμα της επεξεργασίας απλά τοποθετώντας τις ετικέτες αρχής και τέλους τους στον οδηγό και φροντίζοντας ώστε να διαμορφωθεί κατάλληλα το περιεχόμενο τους. Πολλές φορές όμως το όνομα του στοιχείου που επιθυμούμε να δημιουργηθεί εξαρτάται από το XML τεκμήριο πάνω στο οποίο θα εφαρμοστεί το φύλλο στυλ και επομένως δεν είναι γνωστό κατά τη στιγμή της συγγραφής του φύλλου στυλ. Στην περίπτωση αυτή μπορούμε να χρησιμοποιήσουμε το στοιχείο **xsl:element** της XSLT για να προδιαγράψουμε το στοιχείο που θέλουμε να δημιουργηθεί στο τεκμήριο-αποτέλεσμα. Το όνομα του στοιχείου που θα δημιουργηθεί καθορίζεται μέσω του (υποχρεωτικού) γνωρίσματος **name** και του (προαιρετικού) γνωρίσματος **namespace** τα οποία διαθέτει το στοιχείο **xsl:element**. Η τιμή του γνωρίσματος **name** εκλαμβάνεται ως οδηγός για τον υπολογισμό του ονόματος του στοιχείου που εισάγεται. Τα γνωρίσματα και το περιεχόμενο του στοιχείου που θα δημιουργηθεί καθορίζεται από το περιεχόμενο του στοιχείου **xsl:element**.

Παράδειγμα 10.10. Το τμήμα φύλλου στυλ που ακολουθεί αντικαθιστά το στοιχείο **article** με το στοιχείο **tutorial** ή με το στοιχείο **research_paper**, ανάλογα με την τιμή του γνωρίσματος **category** του στοιχείου **article**.

```
<xsl:template match="article">
        <xsl:element name="{@category}" namespace="">
                <xsl:value-of select="title"/>
        </xsl:element>
</xsl:template>
```

Το αναμενόμενο αποτέλεσμα από την εφαρμογή του τμήματος αυτού του φύλλου στυλ στο XML τεκμήριο του Πίνακα 9.1 είναι το παρακάτω τμήμα του XML τεκμηρίου-αποτελέσματος:

```
<tutorial>Intensional Logic in Context</tutorial>
<research_paper>Intensional Communities</research_paper>
```

❑

10.10.2.2 Δημιουργία γνωρισμάτων με το "xsl:attribute" και το "xsl:attribute-set"

Η XSTL διαθέτει τα στοιχεία xsl:attribute και xsl:attribute-set για την εισαγωγή γνωρισμάτων στα στοιχεία του τεκμηρίου του αποτελέσματος. Όπως και στην περίπτωση του στοιχείου xsl:element, τα στοιχεία αυτά είναι χρήσιμα όταν τα ονόματα ή/και οι τιμές των γνωρισμάτων που πρόκειται να εισαχθούν εξαρτώνται από το τεκμήριο εισόδου και κατά συνέπεια δεν είναι γνωστά κατά τη στιγμή της συγγραφής του φύλλου στυλ, αφού σε διαφορετική περίπτωση τα γνωρίσματα και οι αντίστοιχες τιμές μπορούν να εισαχθούν κατευθείαν στα στοιχεία του οδηγού χωρίς να είναι αναγκαία η χρήση των στοιχείων xsl:attribute και xsl:attribute-set.
Η γενική μορφή του στοιχείου xsl:attribute είναι η ακόλουθη:

```
<xsl:attribute
        name = { qname }
        namespace = { uri-reference }>
        …περιεχόμενο ...
</xsl:attribute>
```

Το πλήρες όνομα του προς εισαγωγή γνωρίσματος προδιαγράφεται μέσω των γνωρισμάτων name και namespace, ενώ η τιμή του γνωρίσματος είναι το περιεχόμενο του στοιχείου xsl:attribute.
Ένα στοιχείο xsl:attribute τοποθετείται ως παιδί είτε ενός στοιχείου xsl:element είτε ενός στοιχείου το οποίο έχει εισαχθεί απευθείας στον οδηγό του αποτελέσματος. Και στις δύο περιπτώσεις το παραγόμενο γνώρισμα (όνομα και τιμή) επισυνάπτεται ως γνώρισμα στο στοιχείο που δημιουργείται από τον πατέρα του στοιχείου xsl:attribute.

Παράδειγμα 10.11. Το τμήμα φύλλου στυλ που ακολουθεί αποτελεί επέκταση του φύλλου στυλ του Παραδείγματος 10.10. Εδώ στα στοιχεία που παράγονται ως αποτέλεσμα της εφαρμογής του φύλλου στυλ επισυνάπτονται και δύο γνωρίσματα. Το γνώρισμα first στο οποίο δίνεται ως τιμή το όνομα του πρώτου συγγραφέα του άρθρου και το γνώρισμα last στο οποίο δίνεται ως τιμή το επώνυμο του πρώτου συγγραφέα του άρθρου:

```
<xsl:template match="article">
    <xsl:element name="{@category}" namespace="">
        <xsl:attribute name="first">
            <xsl:value-of select="authors/author[1]/first"/>
        </xsl:attribute>
        <xsl:attribute name="last">
            <xsl:value-of select="authors/author[1]/last"/>
        </xsl:attribute>
        <xsl:value-of select="title"/>
    </xsl:element>
```

```
</xsl:template>
```

Το αποτέλεσμα από την εφαρμογή του παραπάνω τμήματος φύλλου στυλ στο XML τεκμήριο του Πίνακα 9.1 είναι το ακόλουθο τμήμα του XML τεκμηρίου-αποτελέσματος:

```
<tutorial first="William" last="Wadge">
        Intensional Logic in Context</tutorial>
<research_paper first="John" last="Plaice">
        Intensional Communities</research_paper>
```

❑

Το στοιχείο xsl:attribute-set είναι ένα στοιχείο ανωτάτου επιπέδου, το οποίο επιτρέπει να ορίσουμε ένα σύνολο γνωρισμάτων και να δώσουμε ένα όνομα στο σύνολο αυτό. Το όνομα του συνόλου γνωρισμάτων μπορεί να χρησιμοποιηθεί για την εισαγωγή των γνωρισμάτων που το απαρτίζουν σε ένα ή περισσότερα στοιχεία.
Η γενική μορφή του στοιχείου xsl:attribute-set είναι η ακόλουθη:

```
<xsl:attribute-set
        name = qname
        use-attribute-sets = qnames>
        ... περιεχόμενο του στοιχείου...
</xsl:attribute-set>
```

Το όνομα που δίνεται στο σύνολο γνωρισμάτων καθορίζεται από το γνώρισμα name ενώ μέσω του γνωρίσματος use-attribute-sets είναι δυνατό να χρησιμοποιηθούν άλλα σύνολα γνωρισμάτων. Το περιεχόμενο του στοιχείου xsl:attribute-set αποτελείται από ένα ή περισσότερα στοιχεία xsl:attribute καθένα από τα οποία περιγράφει ένα γνώρισμα του συνόλου γνωρισμάτων.

10.10.2.3 Δημιουργία σχολίων και οδηγιών επεξεργασίας

Για τη τοποθέτηση σχολίων στο τεκμήριο-αποτέλεσμα χρησιμοποιείται το στοιχείο xsl:comment.

Παράδειγμα 10.12. Το παρακάτω τμήμα φύλλου στυλ:

```
<xsl:comment>Αυτό είναι ένα σχόλιο!</xsl:comment>
```

εισάγει το σχόλιο:

```
<!-- Αυτό είναι ένα σχόλιο! -->
```

στο τεκμήριο του αποτελέσματος.

❑

Είναι φανερό ότι δεν είναι δυνατό να τοποθετήσουμε σχόλια στο αποτέλεσμα απλά παραθέτοντας τα στον οδηγό ενός κανόνα-οδηγού, αφού αυτά θα εκληφθούν ως σχόλια του φύλλου στυλ και όχι ως τμήμα του αποτελέσματος.

Για τη δημιουργία οδηγιών επεξεργασίας στο τεκμήριο του αποτελέσματος χρησιμοποιούμε το στοιχείο **xsl:processing-instruction**. Το στοιχείο αυτό διαθέτει το γνώρισμα **name** μέσω του οποίου καθορίζεται το όνομα της οδηγίας επεξεργασίας.

10.10.2.4 Εισαγωγή κειμένου με την "xsl:text"

Η XSLT παρέχει τη δυνατότητα να χρησιμοποιήσουμε το στοιχείο **xsl:text** για να εισάγουμε το περιεχόμενο του ως κείμενο στο τεκμήριο εξόδου. Η χρησιμότητα του στοιχείου αυτού είναι γενικά περιορισμένη αφού μπορούμε να εισάγουμε κείμενο στο τεκμήριο του αποτελέσματος απλά τοποθετώντας το στην κατάλληλη θέση του οδηγού. Παρόλα αυτά, το στοιχείο **xsl:text** είναι χρήσιμο σε κάποιες περιπτώσεις όπως όταν θέλουμε να διατηρήσουμε τους κενούς χαρακτήρες ή όταν θέλουμε να συμπεριλάβουμε στο τεκμήριο του αποτελέσματος χαρακτήρες όπως ο **<** ή ο **&**.

10.10.3. Αντιγραφή τμημάτων του XML τεκμηρίου με το στοιχείο "xsl:copy"

Το στοιχείο **xsl:copy** αντιγράφει τον τρέχοντα κόμβο (χωρίς τα γνωρίσματα και τα παιδιά του) στο τεκμήριο εξόδου. Το περιεχόμενο του στοιχείου **xsl:copy** είναι οδηγός μέσω του οποίου μπορούμε να χειριστούμε τα γνωρίσματα και τα παιδία του τρέχοντος κόμβου.

Παράδειγμα 10.13. Με το τμήμα του φύλλου στυλ που ακολουθεί επιτυγχάνεται η αντιγραφή όλου του τεκμηρίου εισόδου στην έξοδο (έχοντας όμως παραλείψει τα γνωρίσματα των στοιχείων):

```
<xsl:template match="*">
        <xsl:copy>
                <xsl:apply-templates/>
        </xsl:copy>
</xsl:template>
<xsl:template match="text()">
        <xsl:value-of select="."/>
```

```
</xsl:template>
```

Αν όμως τροποποιήσουμε το φύλλο στυλ ως ακολούθως τότε πετυχαίνουμε την αντιγραφή του τεκμηρίου εισόδου μαζί με τα γνωρίσματα των στοιχείων:

```
<xsl:template match="*">
        <xsl:copy>
                <xsl:for-each select="@*">
                        <xsl:copy/>
                </xsl:for-each>
                <xsl:apply-templates/>
        </xsl:copy>
</xsl:template>
<xsl:template match="text()">
        <xsl:value-of select="."/>
</xsl:template>
```

❑

Όπως είδαμε πιο πριν, με το στοιχείο **xsl:copy** μπορούμε να αντιγράψουμε τον τρέχοντα κόμβο. Η XSLT παρέχει επίσης το στοιχείο **xsl:copy-of** με τη βοήθεια του οποίου μπορούμε να αντιγράψουμε άλλους κόμβους (πιθανά περισσότερους από έναν). Οι προς αντιγραφή κόμβοι επιλέγονται μέσω του γνωρίσματος **select** το οποίο διαθέτει το στοιχείο **xsl:copy-of**.

10.10.4. Στοιχεία της XSLT για τον έλεγχο της επεξεργασίας

Μια σειρά στοιχείων της XSLT παρέχουν δυνατότητες ελέγχου της διαδικασίας εφαρμογής των κανόνων-οδηγών. Στην κατηγορία αυτή μπορούμε να εντάξουμε τα ακόλουθα στοιχεία:

xsl:for-each
xsl:if
xsl:choose
xsl:when
xsl:otherwise

Ο τρόπος χρήσης των στοιχείων αυτών περιγράφεται στη συνέχεια.

10.10.4.1 Επανάληψη με το στοιχείο "xsl:for-each"

Το στοιχείο **xsl:for-each** χρησιμοποιείται για την επεξεργασία πολλών κόμβων (με παρόμοιο τρόπο). Η γενική μορφή του στοιχείου αυτού είναι η ακόλουθη:

```
<xsl:for-each
        select=έκφραση_συνόλου_κόμβων>
    ... περιεχόμενο (οδηγός) ...
</xsl:for-each>
```

Το γνώρισμα select, το οποίο είναι υποχρεωτικό, καθορίζει το σύνολο των κόμβων για τους οποίους θα υπολογιστεί ο οδηγός.

Παράδειγμα 10.14. Το τμήμα φύλλου στυλ που ακολουθεί επεξεργάζεται καθένα από τα στοιχεία με ετικέτα article και για κάθε τέτοιο στοιχείο στέλνει στο τεκμήριο του αποτελέσματος το στοιχείο title πού είναι παιδί του article.

```
<xsl:template match="articles">
    <xsl:for-each select="article">
        <title>
            <xsl:value-of select="title"/>
        </title>
    </xsl:for-each>
</xsl:template>
```

Το αναμενόμενο αποτέλεσμα από την εφαρμογή του τμήματος αυτού του φύλλου στυλ στο XML τεκμήριο του Πίνακα 9.1 είναι το παρακάτω τμήμα του XML τεκμηρίου-αποτελέσματος:

```
<title>Intensional Logic in Context</title>
<title>Intensional Communities</title>
```

❑

10.10.4.2 Επεξεργασία υπό συνθήκη

Η XSLT παρέχει τα στοιχεία xsl:if και xsl:choose τα οποία μας επιτρέπουν να καθορίσουμε το αποτέλεσμα με βάση τα χαρακτηριστικά του τεκμηρίου εισόδου. Συγκεκριμένα, το στοιχείο xsl:if διαθέτει ένα γνώρισμα test, η τιμή του οποίου είναι έκφραση. Αν η έκφραση υπολογιστεί σε true (είναι αληθής), τότε ο οδηγός (που αποτελεί το περιεχόμενο του στοιχείου xsl:if) αποστέλλεται στην έξοδο αφού συμπληρωθεί κατάλληλα. Σε διαφορετική περίπτωση δε γίνεται τίποτα (δεν αποστέλλεται τίποτα στην έξοδο).

Παράδειγμα 10.15. Το τμήμα φύλλου στυλ που ακολουθεί επεξεργάζεται καθένα από τα στοιχεία με ετικέτα article και επιστρέφει τον τίτλο των άρθρων που ανήκουν στην κατηγορία (έχουν τιμή του γνωρίσματος category) "research_paper":

```
<xsl:template match="articles">
```

```
            <xsl:for-each select="article">
                <xsl:if test="@category='research_paper'">
                        <title>
                            <xsl:value-of select="title"/>
                        </title>
                </xsl:if>
            </xsl:for-each>
        </xsl:template>
```

Το αναμενόμενο αποτέλεσμα από την εφαρμογή του τμήματος αυτού του φύλλου στυλ στο XML τεκμήριο του Πίνακα 9.1 είναι το παρακάτω τμήμα του XML τεκμηρίου-αποτελέσματος:

```
<title>Intensional Communities</title>
```

❑

Η XSLT δε διαθέτει στοιχεία **xsl:else** και **xsl:else-if** (όπως ενδεχομένως θα περίμενε κάποιος που είναι εξοικειωμένος με τις γλώσσες προγραμματισμού). Η λειτουργίες που θα μπορούσαν να εξασφαλίσουν τέτοια στοιχεία εξασφαλίζονται στην XSLT με τα στοιχεία **xsl:choose**, **xsl:when** και **xsl:otherwise**.

Παράδειγμα 10.16. Στο ακόλουθο φύλλο στυλ χρησιμοποιούμε τα στοιχεία **xsl:choose**, **xsl:when** και **xsl:otherwise**:

```
<xsl:template match="*">
  <xsl:choose>
    <xsl:when
       test="(child::first[1] and child::last[position()=last()])">
        <name>
            <xsl:value-of select="first"/>
            <xsl:text> </xsl:text>
            <xsl:value-of select="last"/>
        </name>
    </xsl:when>
    <xsl:otherwise>
       <xsl:copy>
          <xsl:for-each select="@*">
             <xsl:copy/>
          </xsl:for-each>
          <xsl:apply-templates/>
       </xsl:copy>
    </xsl:otherwise>
```

```
    </xsl:choose>
</xsl:template>
<xsl:template match="text()">
    <xsl:value-of select="."/>
</xsl:template>
```

Το αποτέλεσμα της εφαρμογής αυτό του φύλλου στυλ στο XML τεκμήριο του Πίνακα 9.1 είναι το τεκμήριο:

```
<?xml version="1.0" encoding="UTF-16"?>
<book>
   <booktitle>Intensional Programming II</booktitle>
   <editors>
      <name>Manolis Gergatsoulis</name>
      <name>Panos Rondogiannis</name>
   </editors>
   <year>2000</year>
   <publisher>World Scientific</publisher>
   <articles>
      <article category="tutorial">
         <title>Intensional Logic in Context</title>
         <authors>
            <name>William Wadge</name>
         </authors>
      </article>
      <article category="research_paper">
         <title>Intensional Communities</title>
         <authors>
            <name>John Plaice</name>
            <name>Peter Kropf</name>
         </authors>
      </article>
   </articles>
</book>
```

Παρατηρήστε ότι τα στοιχεία της μορφής:

```
<author>
  <first>X</first>
  <last>Z</last>
</author>
```

και τα στοιχεία της μορφής:

```
<author>
  <first>X</first>
  <middle>Y</middle>
  <last>Z</last>
</author>
```

όπως και τα στοιχεία της μορφής:

```
<editor>
   <first>X</first>
   <last>Z</last>
</editor>
```

του αρχικού τεκμηρίου έχουν αντικατασταθεί με στοιχεία της μορφής:

```
<name>X Z</name>
```

ενώ όλα τα υπόλοιπα στοιχεία του αρχικού τεκμηρίου έχουν απλά αντιγραφεί.

❑

10.10.5. Παραγωγή ταξινομημένων αποτελεσμάτων

Πολλές φορές είναι χρήσιμο να διατάξουμε τα αποτελέσματα με διαφορετική σειρά από αυτήν που προκύπτει με βάση το αρχικό XML τεκμήριο. Η XSLT παρέχει αυτήν τη δυνατότητα μέσω του στοιχείου **xsl:sort**. Το στοιχείο **xsl:sort** τοποθετείται ως παιδί ενός στοιχείου **xsl:apply-templates** ή ενός στοιχείου **xsl:for-each**. Ένα στοιχείο **xsl:apply-templates** ή **xsl:for-each** είναι δυνατό να διαθέτει περισσότερα του ενός στοιχεία-παιδιά **xsl:sort**. Στην περίπτωση αυτήν το πρώτο από αυτά (κατά σειρά εμφάνισης) καθορίζει το *πρωτεύον κλειδί της ταξινόμησης* (primary sort key), το δεύτερο καθορίζει το *δευτερεύον κλειδί* της ταξινόμησης (secondary sort key), κ.ο.κ. Η βασική ιδέα για τη χρήση του στοιχείου **xsl:sort** είναι ότι όταν ένα στοιχείο **xsl:apply-templates** ή ένα στοιχείο **xsl:for-each** διαθέτει ένα ή περισσότερα στοιχεία παιδιά **xsl:sort**, τότε η επεξεργασία των κόμβων δε γίνεται με βάση τη διάταξη τεκμηρίου, δηλαδή τη σειρά εμφάνισης των στοιχείων στο XML τεκμήριο, αλλά με βάση τη σειρά που προσδιορίζεται από τα στοιχεία **xsl:sort**.

Η γενική μορφή του στοιχείου **xsl:sort** είναι η ακόλουθη:

```
<xsl:sort
        select = string-expression
        lang = { nmtoken }
        data-type = { "text" | "number" | qname-but-not-ncname }
        order = { "ascending" | "descending" }
        case-order = { "upper-first" | "lower-first" } />
```

Το γνώρισμα **select** προσδιορίζει το κλειδί το οποίο θα χρησιμοποιηθεί για την ταξινόμηση.
Το γνώρισμα **order** προσδιορίζει αν η ταξινόμηση θα γίνει κατά *αύξουσα* (τιμή "ascending") ή κατά *φθίνουσα* (τιμή "descending") διάταξη ως προς το κλειδί.
Το γνώρισμα **data-type** καθορίζει τον τύπο των κλειδιών με βάση τα οποία θα γίνει η ταξινόμηση. Επιτρεπτές τιμές του γνωρίσματος **data-type** είναι η τιμή **text** η οποία υποδεικνύει λεξικογραφική διάταξη (με βάση και τα χαρακτηριστικά της γλώσσας που χρησιμοποιείται όπως αυτή προσδιορίζεται από το προαιρετικό γνώρισμα[3] **lang**), ή η τιμή **number** η οποία επιβάλει τη μετατροπή των κλειδιών ταξινόμησης σε αριθμούς και στη συνέχεια την ταξινόμηση με βάση τις αριθμητικές τιμές των κλειδιών.
Σε περίπτωση απουσίας του γνωρίσματος **data-type**, το οποίο είναι προαιρετικό, θεωρείται ότι αυτό έχει την τιμή **text**.
Το γνώρισμα **case-order** καθορίζει (σε περίπτωση που το γνώρισμα **data-type** έχει την τιμή **text**) αν τα κεφαλαία θα προηγούνται των μικρών χαρακτήρων (τιμή του γνωρίσματος "upper-first") ή τα μικρά θα προηγούνται των κεφαλαίων (τιμή του γνωρίσματος "lower-first").

Παράδειγμα 10.17. Θεωρείστε το παρακάτω XML τεκμήριο το οποίο κωδικοποιεί τα στοιχεία των φοιτητών του τμήματος.

```
<students>
  <student>
      <scode>2388</scode><first>Νίκος</first><last>Νικολάου</last>
  </student>
  <student>
      <scode>2218</scode><first>Αντώνης</first><last>Αντωνίου</last>
  </student>
  <student>
      <scode>1234</scode><first>Πέτρος</first><last>Πέτρου</last>
  </student>
  <student>
```

[3] Το σύνολο των πιθανών τιμών του γνωρίσματος **lang** είναι το ίδιο με το σύνολο των πιθανών τιμών του γνωρίσματος **xml:lang** της XML.

```
        <scode>2212</scode><first>Νίκος</first><last>Πέτρου</last>
   </student>
</students>
```

Το τμήμα φύλλου στυλ που ακολουθεί:

```
<xsl:template match="students">
        <φοιτητές>
                <xsl:apply-templates select="student">
                        <xsl:sort select ="scode" data-type = "number"/>
                </xsl:apply-templates>
        </φοιτητές>
</xsl:template>

<xsl:template match="student">
        <φοιτητής>
                <xsl:value-of select="scode"/>
                <xsl:text> </xsl:text>
                <xsl:value-of select="last"/>
                <xsl:text> </xsl:text>
                <xsl:value-of select="first"/>
        </φοιτητής>
</xsl:template>
```

Μετασχηματίζει το αρχικό XML τεκμήριο στο ακόλουθο XML τεκμήριο στο οποίο τα στοιχεία των φοιτητών εμφανίζονται ταξινομημένα κατά αύξουσα σειρά ως προς τον κωδικό φοιτητή.

```
<φοιτητές>
        <φοιτητής>1234 Πέτρου Πέτρος</φοιτητής>
        <φοιτητής>2212 Πέτρου Νίκος</φοιτητής>
        <φοιτητής>2218 Αντωνίου Αντώνης</φοιτητής>
        <φοιτητής>2388 Νικολάου Νίκος</φοιτητής>
</φοιτητές>
```

Αν όμως στο πιο πάνω φύλλο στυλ αντικαταστήσουμε το στοιχείο:

```
<xsl:sort select ="scode" data-type = "number"/>
```

με το ζευγάρι στοιχείων:

```
<xsl:sort select ="last" data-type = "text"/>
```

```
<xsl:sort select ="first" data-type = "text"/>
```

τότε το αποτέλεσμα που θα προκύψει είναι το ακόλουθο:

```
<φοιτητές>
        <φοιτητής>2218 Αντωνίου Αντώνης</φοιτητής>
        <φοιτητής>2388 Νικολάου Νίκος</φοιτητής>
        <φοιτητής>2212 Πέτρου Νίκος</φοιτητής>
        <φοιτητής>1234 Πέτρου Πέτρος</φοιτητής>
</φοιτητές>
```

Παρατηρήστε ότι στο τεκμήριο αυτό οι φοιτητές εμφανίζονται κατά αλφαβητική σειρά.

❑

10.11. Επίκληση κανόνων-οδηγών μέσω ονομάτων

Όπως είδαμε στην Παράγραφο 10.5.3 το στοιχείο **xsl:template** διαθέτει το προαιρετικό γνώρισμα **name**. Το γνώρισμα αυτό χρησιμοποιείται για να δώσουμε (εφόσον το επιθυμούμε) ονόματα στους κανόνες οδηγούς. Στην περίπτωση που ένας κανόνας οδηγός διαθέτει όνομα, αυτό μπορεί να χρησιμοποιηθεί για να καλέσουμε τον κανόνα αυτόν. Η κλήση του κανόνα γίνεται με το στοιχείο **xsl:call-template** της XSLT το οποίο έχει την ακόλουθη μορφή:

```
<xsl:call-template
        name = qname>
        … περιεχόμενο του στοιχείου ...
</xsl:call-template>
```

Η εκτέλεση μιας τέτοιας δήλωσης οδηγεί στη κλήση το κανόνα οδηγού που φέρει το όνομα *qname*. (το γνώρισμα **name** στο στοιχείο **xsl:call-template** είναι υποχρεωτικό). Η κλήση του στοιχείου **xsl:call-template** δεν αλλάζει τον τρέχοντα κόμβο ή λίστα κόμβων.

10.12. Μεταβλητές και παράμετροι

Η XSLT παρέχει τη δυνατότητα χρήσης *μεταβλητών* (variables) και *παραμέτρων* (parameters) κατά τρόπον που θυμίζει γλώσσες προγραμματισμού. Μια μεταβλητή έχει ένα όνομα και μπορεί να πάρει κάποια τιμή η οποία μπορεί να είναι οποιουδήποτε τύπου από αυτούς που μπορούν να αποδοθούν ως τιμή μιας έκφρασης. Τα στοιχεία της XSLT τα οποία μπορούν να

χρησιμοποιηθούν για να δώσουμε τιμή σε μια μεταβλητή είναι τα xsl:variable και xsl:param. Το όνομα της μεταβλητής ορίζεται μέσω του γνωρίσματος name το οποίο είναι υποχρεωτικό για τα στοιχεία xsl:variable και xsl:param. Μια μεταβλητή μπορεί επίσης να πάρει ως τιμή ένα *τμήμα του δέντρου του αποτελέσματος* (result tree fragment).
Μια μεταβλητή μπορεί να πάρει τιμή μέσω του γνωρίσματος select του στοιχείου xsl:variable ή xsl:param στο οποίο ορίζεται. Το γνώρισμα select είναι προαιρετικό.
Στην περίπτωση που το στοιχείο διαθέτει το γνώρισμα select τότε η τιμή του γνωρίσματος αυτού είναι μια έκφραση η οποία υπολογίζεται και η τιμή που προκύπτει δίνεται ως τιμή της μεταβλητής.

Παράδειγμα 10.18. Με την εκτέλεση της ακόλουθης εντολής:

```
<xsl:variable name="X" select="/book/articles/article[1]/title"/>
```

ορίζεται η μεταβλητή με όνομα X η οποία παίρνει τιμή μέσω της παράστασης που δίνεται ως τιμή στο γνώρισμα select. Η τιμή της μεταβλητής είναι ο τίτλος του πρώτου άρθρου του βιβλίου. Με βάση το παράδειγμα του Πίνακα 9.1 η τιμή αυτή είναι η:

```
Intensional Logic in Context
```

❑

Σε περίπτωση που το γνώρισμα select απουσιάζει, και το στοιχείο στο οποίο ορίζεται η μεταβλητή (xsl:variable ή xsl:param) δεν είναι κενό, τότε το περιεχόμενο του στοιχείου αυτού εκλαμβάνεται ως οδηγός ο οποίος αφού συγκεκριμενοποιηθεί δίνεται ως τιμή στη μεταβλητή.

Παράδειγμα 10.19.Στο στοιχείο που ακολουθεί:

```
<xsl:variable name="var1">
        This is the value of the variable var1
</xsl:variable>
```

δηλώνεται μια μεταβλητή με όνομα var1 και τιμή τη συμβολοσειρά This is the value of the variable var1.

❑

Στην περίπτωση που το στοιχείο είναι κενό και δε διαθέτει γνώρισμα με όνομα select, τότε η μεταβλητή παίρνει ως τιμή την κενή συμβολοσειρά. Για να προσπελάσουμε την τιμή μιας μεταβλητής παραθέτουμε το σύμβολο του δολαρίου $ πριν από το όνομα της μεταβλητής. Για παράδειγμα μπορούμε να χρησιμοποιήσουμε το στοιχείο xsl:value-of της XSLT για να τοποθετήσουμε

την τιμή μιας μεταβλητής στο σημείο του αποτελέσματος στο οποίο επιθυμούμε.

Παράδειγμα 10.20. (Συνέχεια από το Παράδειγμα 10.19) Στο στοιχείο που ακολουθεί:

```
<xsl:value-of select="$var1"/>
```

στέλνεται στην έξοδο η τιμή της μεταβλητής var1 δηλαδή η συμβολοσειρά This is the value of the variable var1.

❑

Αξίζει να σημειωθεί ότι το περιεχόμενο του στοιχείου xsl:variable (το οποίο προσδιορίζει την τιμή μιας μεταβλητής) μπορεί να περιλαμβάνει άλλα στοιχεία της XSLT τα οποία υπολογίζονται έτσι ώστε το αποτέλεσμα του υπολογισμού να δοθεί σαν τιμή στη μεταβλητή.
Τα στοιχεία xsl:variable ή xsl:param επιτρέπεται να είναι στοιχεία ανωτάτου επιπέδου. Δίνεται έτσι η δυνατότητα ορισμού μεταβλητών οι οποίες είναι *καθολικές* (global). Οι καθολικές μεταβλητές είναι ορατές από οποιοδήποτε σημείο του φύλλου στυλ.
Με τη χρήση παραμέτρων και μεταβλητών σχετίζεται επίσης και το στοιχείο xsl:with-param με τη βοήθεια του οποίου μπορούμε να περάσουμε παραμέτρους σε κανόνες-οδηγούς. Περισσότερες λεπτομέρειες για τη λειτουργία αυτή στο τεχνικό εγχειρίδιο της γλώσσας [3].

10.13. Συνδυασμός διαφορετικών φύλλων στυλ

Η XSLT παρέχει τη δυνατότητα να συνδυάσουμε διαφορετικά φύλλα στυλ. Αυτό γίνεται με τη χρήση των στοιχείων xsl:include και xsl:import τα οποία είναι στοιχεία ανωτάτου επιπέδου (μπορούν επομένως να συμπεριληφθούν μόνο ως υποστοιχεία των στοιχείων xsl:stylesheet και xsl:transform).

10.13.1. Το στοιχείο xsl:include

Ένα φύλλο στυλ μπορεί να συμπεριλάβει ένα άλλο φύλλο στυλ με τη βοήθεια του στοιχείου xsl:include. Το στοιχείο xsl:include συντάσσεται ως ακολούθως:

```
<xsl:include
    href = αναφορά-URI />
```

Το τεκμήριο το οποίο βρίσκεται στην τοποθεσία που προσδιορίζει η *αναφορά-URI* ανακτάται και αναλύεται συντακτικά σαν ένα XML τεκμήριο και όλα τα

παιδιά του στοιχείου xsl:stylesheet του τεκμηρίου αυτού αντικαθιστούν το στοιχείο xsl:include στο αρχικό φύλλο στυλ.

10.13.2. Το στοιχείο xsl:import

Ένα φύλλο στυλ μπορεί να εισάγει ένα άλλο φύλλο στυλ με τη βοήθεια του στοιχείου xsl:import. Το στοιχείο xsl:import συντάσσεται ως ακολούθως:

```
<xsl:import
    href = αναφορά-URI />
```

Τα στοιχεία xsl:import τοποθετούνται πριν από τα υπόλοιπα υποστοιχεία του xsl:stylesheet. Η λειτουργία του μηχανισμού εισαγωγής με τη χρήση του στοιχείου xsl:import είναι παρόμοια με την εισαγωγή με τη βοήθεια του στοιχείου xsl:include. Η διαφορά είναι ότι στην περίπτωση του xsl:import οι κανόνες-οδηγοί και τα υπόλοιπα στοιχεία του εισαγόμενου φύλλου στυλ λαμβάνουν χαμηλότερη προτεραιότητα έναντι των στοιχείων του φύλλου στυλ που τα εισάγει. Για περισσότερες λεπτομέρειες για τον τρόπο καθορισμού των προτεραιοτήτων στις περιπτώσεις πολλαπλών εντολών xsl:import τα οποία εισάγουν φύλλα στυλ τα οποία με τη σειρά τους περιλαμβάνουν στοιχεία xsl:import ο αναγνώστης μπορεί να αναφερθεί στο τεχνικό εγχειρίδιο [3].

10.14. Συμπεράσματα

Η XSLT είναι μια εξαιρετικά χρήσιμη γλώσσα μέσω της οποίας μπορούμε να περιγράψουμε μετασχηματισμούς που μετατρέπουν ένα XML τεκμήριο σε ένα άλλο XML τεκμήριο. Στο κεφάλαιο αυτό έγινε προσπάθεια να παρουσιαστούν τα βασικά συντακτικά στοιχεία της XSLT και να γίνει κατανοητή η χρήση τους μέσω παραδειγμάτων. Ο αναγνώστης που ενδιαφέρεται για περισσότερες λεπτομέρειες ενθαρρύνεται να ανατρέξει στο τεχνικό εγχειρίδιο της γλώσσας [3].

ΚΕΦΑΛΑΙΟ 11:
XSL Formatting Objects (XSL FO)

11.1. Εισαγωγή

Η XSL Formatting Objects (XSL-FO) είναι η δεύτερη υπογλώσσα της XSL. Δια μέσου της XSL-FO, η οποία είναι με τη σειρά της μια XML γλώσσα, περιγράφουμε την εμφάνιση που επιθυμούμε να έχουν οι σελίδες όταν παρουσιαστούν στον αναγνώστη. Για το σκοπό αυτό η XSL-FO παρέχει ένα κατάλληλο λεξιλόγιο XML στοιχείων με ειδική σημασία που σχετίζεται με τον τρόπο παρουσίασης της πληροφορίας.

Βιβλιογραφία

1. *"XML Path Language (XPath) Version 1.0"*, W3C Recommendation, 16 November 1999. Διαθέσιμο από τη διεύθυνση: http://www.w3.org/TR/1999/REC-xpath-19991116
2. *"XML Path Language (XPath) 2.0"*, W3C Working Draft, 12 November 2003. Διαθέσιμο από τη διεύθυνση: http://www.w3.org/TR/2003/WD-xpath20-20031112
3. *"XSL Transformations (XSLT) Version 1.0"*,W3C Recommendation 16 November 1999. Διαθέσιμο από τη διεύθυνση: http://www.w3.org/TR/1999/REC-xslt-19991116
4. "*Extensible Stylesheet Language (XSL), Version 1.0*" W3C Recommendation 15 October 2001. Διαθέσιμο από τη διεύθυνση: http://www.w3.org/TR/2001/REC-xsl-20011015/
5. *"XQuery 1.0: An XML Query Language"*, W3C Working Draft 12 November 2003. Διαθέσιμο από τη διεύθυνση: http://www.w3.org/TR/2003/WD-xquery-20031112/

Ευρετήριο Όρων