2.1 Εισαγωγή κεφαλαίου

2.2 Persistent Data στην Πληροφορική

Η «διατήρηση δεδομένων»(data preservation) στην Πληροφορική είναι η διαδικασία διατήρησης αρχείων και δεδομένων σε ένα στάδιο που τα καθιστά προσβάσιμα καθ' όλη τη διάρκεια του χρόνου. Τα δεδομένα αυτά θα πρέπει να αποθηκεύονται σε μορφές αρχείων που θα τα καθιστούν πιο χρήσιμα στο μέλλον, θα πρέπει να φυλάσσονται σε πολλές τοποθεσίες και τέλος να διατηρούνται σε ένα προστατευμένο «περιβάλλον» προκειμένου να διατηρηθούν(ref).

Συγκεκριμένα, τα μόνιμα δεδομένα(persistent data) αναφέρονται σε δεδομένα που διατηρούνται επ' αόριστόν σε μη πτητικές συσκευές αποθήκευσης(non-volatile storage), όπως μαγνητικές ταινίες, σκληρούς δίσκους, Solid State Drives αλλά και οπτικούς δίσκους. Τα δεδομένα αυτά διατηρούνται ακόμη και μετά την απενεργοποίηση της συσκευής, δηλαδή χωρίς ρεύμα, από όπου προέρχεται και η ονομασία τους Non-Volatile Memory. Φυσικά, χρησιμοποιούνται πολλές τακτικές για να εξασφαλιστεί η προσβασιμότητα και η μακροπρόθεσμη διατήρησή τους. Χρησιμοποιούνται συστήματα αρχείων για την οργάνωση των δεδομένων(file systems), εφαρμόζονται τεχνικές δημιουργίας αντιγράφων ασφαλείας, όπως διαφορικά(differential), αυξητικά(incremental) ή πλήρη(complete) αντίγραφα ασφαλείας, τα οποία αποθηκεύονται εντός, εκτός ή στο cloud, και γίνεται αντιγραφή των δεδομένων σε διάφορες συσκευές ή τοποθεσίες για λόγους ανθεκτικότητας. Επιπλέον, η αρχειοθέτηση δεδομένων, η οποία συχνά κάνει χρήση εξειδικευμένων συστημάτων αποθήκευσης, διατηρεί ανενεργές αλλά σημαντικές πληροφορίες για λόγους συμμόρφωσης ή ιστορικούς σκοπούς. Τα δεδομένα προστατεύονται από την ανεπιθύμητη πρόσβαση μέσω κρυπτογράφησης και η ορθότητα των δεδομένων διασφαλίζεται με δοκιμές ακεραιότητας που μέσω hashing ή συνόλων ελέγχου(checksums). (ref)

2.3 Data Serialization

Στην υποενότητα αυτή γίνεται μία σύντομη ανασκόπηση στους ορισμούς του Serialization και Deserialization όπως αυτοί χρησιμοποιούνται στην Πληροφορική (εν. 2.3.1) και εξηγούνται οι βασικοί τύποι serialization όπως είναι οι XML, JSON, YAML και binary (εν. 2.3.2). Στη συνέχεια, αναφέρονται τα βασικά χαρακτηριστικά επιλογής ενός serializer (εν. 2.3.3) και γίνεται μία σύντομη επεξήγηση της διαδικασίας του serialization και συγκεκριμένα της εξάλειψης των object reference στη μνήμη του υπολογιστή κατά τη διαδικασία αυτή (εν. 2.3.4). Τέλος, γίνεται μία επισκόπηση των επιλεγμένων serializer προς ανάλυση (εν. 2.3.5) όπως αυτή εμφανίζεται στο κεφάλαιο της Μεθοδολογίας (κεφ. 3).

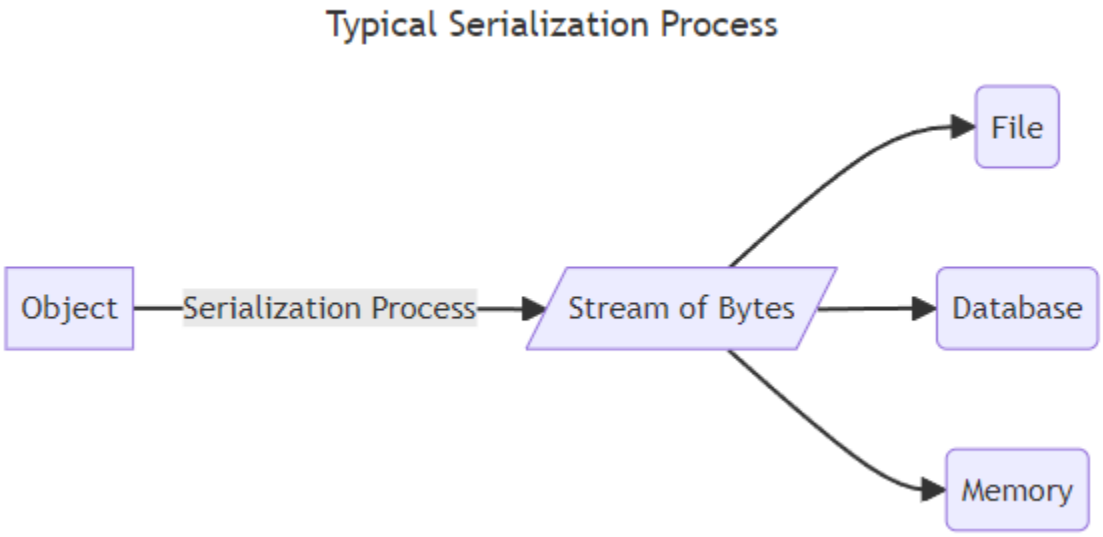
2.3.1 Ορισμός του Serialization και Deserialization

Στον κόσμο της Πληροφορικής, η διαδικασία της μετατροπής δεδομένων σε μία μορφή κατανοητή από έναν υπολογιστή ονομάζεται Serialization. Αναλυτικότερα, όπως διακρίνεται στην Εικόνα 1, η σειριοποίηση δεδομένων(data serialization) αναφέρεται στην πράξη του μετασχηματισμού περίπλοκων δομών δεδομένων ή καταστάσεων αντικειμένων (ref) από τη κατάσταση τους στη μνήμη σε έναν πίνακα από bytes ή σε μια μορφή που μπορεί εύκολα να καταγραφεί σε ένα αρχείο, να μεταφερθεί και να

ξαναδημιουργηθεί αργότερα μέσω της αντίστροφης διαδικασίας, ονόματη Deserialization. Συγκεκριμένα, το Deserialization είναι η διαδικασία μετάφρασης του αποθηκευμένου πίνακα ή αποθηκευμένης σειράς από bytes στην μορφή που ήταν πριν το serialization, κάτι που καθιστά τη διαδικασία του deserialization άμεσα συνδεδεμένη με τη διαδικασία του serialization. (ref)

Επιπλέον, είναι σημαντικό να αναφερθεί πως ο Serializer είναι κάτι διαφορετικό από το Serialization Format που εξάγει μετά τη διαδικασία του serialization. Το serialization format που εξάγει κάθε serializer βασίζεται σε συγκεκριμένη σύνταξη αναπαράστασης των serialized δεδομένων είτε αυτό είναι ένα JSON string, ένα YAML ή XML structure είτε είναι η binary αναπαράσταση αυτών. (ref)

Τέλος, κάθε serializer και με τη σειρά της η μορφή serialized δεδομένων που εξάγει έχει πλεονεκτήματα και εφαρμογές, όπου αυτά μπορούν να κυμαίνονται από το ποσοστό δυνατότητας ανθρώπινης ανάγνωσης μέχρι και το τελικό μέγεθος του παραχθέντος αρχείου για λόγους αποθήκευσης και μεταφοράς.(ref)



Εικόνα 1: Serialization process

2.3.2 Serialization formats

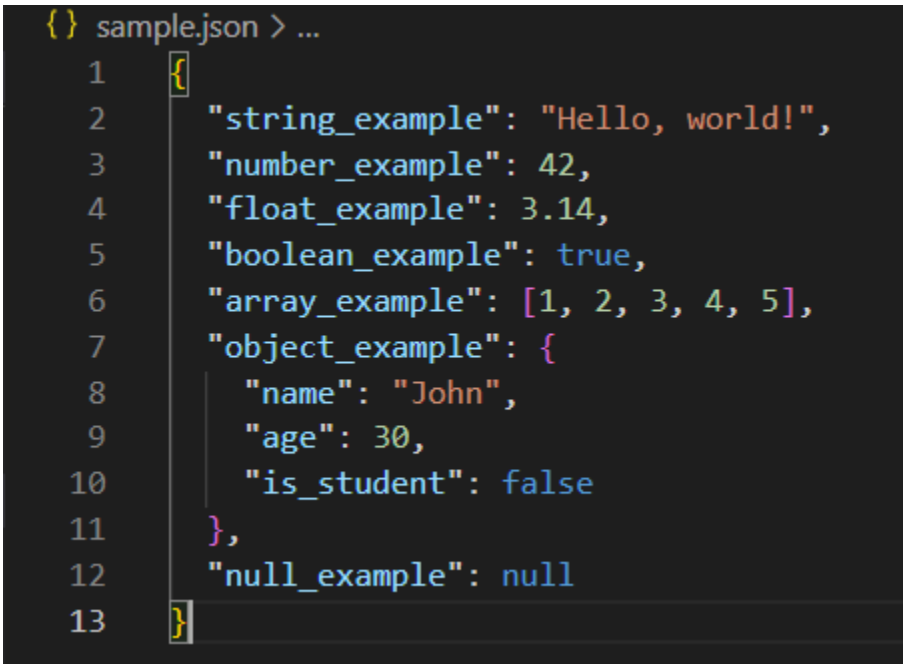
Η ακριβής σύνταξη και η δομή που χρησιμοποιείται για την αναπαράσταση των serialized δεδομένων καθορίζονται από τη μορφή του serialization. Οι μορφότυποι serialization εμφανίζουν ένα εύρος χαρακτηριστικών, όπως αναγνωσιμότητα, αποδοτικότητα και πολυπλοκότητα. Τα JSON, YAML, XML και Message Pack (binary) είναι μερικά παραδείγματα από αυτά τα μορφότυπα. Κάθε μορφή serialization περιγράφει κανόνες για την αναπαράσταση τύπων δεδομένων, την οργάνωση δομών δεδομένων και τον χειρισμό ειδικών περιπτώσεων όπως nested objects ή πίνακες, παρέχοντας έτσι μία σχεδίαση (schema) για το serialization δεδομένων σε διαφορετικά συστήματα και πλατφόρμες. (ref)

Τέλος, τα formats JSON, YAML, XML και binary διαθέτουν το καθένα μοναδικές ικανότητες προσαρμοσμένες σε συγκεκριμένες περιπτώσεις χρήσης, καλύπτοντας ποικίλες απαιτήσεις

αναπαράστασης (data presentation), μετάδοσης (transmission) και αποθήκευσης δεδομένων (data preservation).

2.3.2a JSON Serialization

Ιστορικά, η μορφή JSON ή αλλιώς JavaScript Object Notation, είναι βασισμένη στο ανοικτό πρότυπο ECMA-262 3rd Edition 12/1999 της JavaScript (ref ECMA-262) από όπου και δημιουργήθηκε το «ECMA-404 The JSON Data Interchange Standard». Σκοπός της δημιουργίας του ήταν η μετατροπή των αναπαραστάσεων δεδομένων σε μορφή απλού κειμένου που διαβάζονται από τον άνθρωπο σε αντικείμενα ECMAScript. Οι συμβολισμοί που χρησιμοποιεί είναι παρόμοιοι με εκείνους των κοινών γλωσσών προγραμματισμού όπως η C, η C++ και η Java και είναι εντελώς ανεξάρτητη από τις άλλες γλώσσες προγραμματισμού, έτσι αποτελεί μια κατάλληλη επιλογή για τη μετάδοση δεδομένων μεταξύ συστημάτων λόγω της αναγνωσιμότητας και της απλότητάς γραφής του (ref ECMA-404). Η ομαλή ενσωμάτωση καθίσταται δυνατή χάρη στον ελαφρύ πηγαίο σχεδιασμό του και την εγγενή υποστήριξη για την πλειονότητα των γλωσσών προγραμματισμού όπως προαναφέρθηκε. Το JSON λειτουργεί καλά σε καταστάσεις όπως τα διαδικτυακά API και τα αρχεία ρυθμίσεων, όπου είναι απαραίτητη η εύκολη ανάγνωση από τον άνθρωπο και η οργάνωση των δεδομένων. Όπως φαίνεται στην Εικόνα 2, τα δεδομένα αποθηκεύονται σαν ζεύγη κλειδιών-τιμών (key-value pairs) και οι διαθέσιμοι τύποι δεδομένων προς αποθήκευση κυμαίνονται στα strings, στους integer/floating-point αριθμούς, στις Boolean τιμές, στις λίστες τακτοποιημένων τιμών (ordered lists), στις συλλογές μη τακτοποιημένων ζευγών κλειδιών-τιμών (unordered key-value collections) και τέλος, την αναπαράσταση της απουσίας τιμής μέσω του keyword null. (ref) Αξίζει να σημειωθεί πως οι συλλογές με μη τακτοποιημένα ζεύγη κλειδιών-τιμών (unordered key-value collections) και αυτά εσωτερικά μπορούν με τη σειρά τους να αναπαραστήσουν δεδομένα με τους προαναφερθέντες τύπους. (ref)



Εικόνα 2: Δείγμα JSON αρχείου

2.3.2β XML Serialization

Η eXtensible Markup Language (XML) είναι μια απλή, πολύ ευέλικτη μορφή κειμένου που προέρχεται από την Standard Generalized Markup Language (SGML) του ISO 8879 (ref ISO 8879). Αρχικά σχεδιάστηκε από μία ομάδα του World Wide Web Consortium (W3C) το 1998 για να ανταποκριθεί στις προκλήσεις των μεγάλης κλίμακας ηλεκτρονικών εκδοτικών οίκων, όμως πλέον παίζει επίσης ολοένα και πιο σημαντικό ρόλο στην ανταλλαγή μιας μεγάλης ποικιλίας δεδομένων στο Ίντερνετ και αλλού. (ref from W3C).

Η XML, αναπαριστά τα δομημένα δεδομένα με ετικέτες (tags) που περικλείονται σε αγκύλες. Αυτές οι ετικέτες οριοθετούν τη δομή των δεδομένων και μπορούν να περιλαμβάνουν χαρακτηριστικά και nested στοιχεία (elements). Κατά το serialization σε XML, τα αντικείμενα ή οι δομές δεδομένων μετατρέπονται σε μία μορφή XML με βάση το επιλεγμένο schema. Αυτό περιλαμβάνει τη διερεύνηση των ιδιοτήτων (attributes) ή των πεδίων (fields) του αντικειμένου και τη δημιουργία των αντίστοιχων στοιχείων και χαρακτηριστικών XML για την ακριβή αναπαράσταση των δεδομένων. Οι δομές XML Schema Definition (XSD) και Document Type Definition (DTD) προσφέρουν επίσημες προδιαγραφές για τον ορισμό της δομής και των περιορισμών των εγγράφων που δημιουργούνται (ref). Επιπλέον, η XML επιτρέπει στου προγραμματιστές να δημιουργήσουν δικές τους προσαρμοσμένες δομές όπου μερικές από τις αλλαγές που μπορούν να επισημανθούν είναι το τελικό XML format που θα δημιουργηθεί, η διαχείριση ειδικών τύπων δεδομένων, ο καθορισμός συμβάσεων ονοματοδοσίας και η διαχείριση των namespaces. (ref) Σαν serialization format, η XML βρίσκει ευρεία χρήση στις υπηρεσίες ιστού για την ανταλλαγή δεδομένων μεταξύ πελατών και διακομιστών. Το πρωτόκολλο Simple Object Access Protocol (SOAP), για παράδειγμα, αξιοποιεί την XML για τη μορφοποίηση μηνυμάτων που μοιράζονται μεταξύ των κατανεμημένων του συστημάτων (distributed systems). Τέλος, είναι άξιο να σημειωθεί πως η XML μοιράζεται ομοιότητες με την HTML, καθώς και οι δύο είναι γλώσσες σήμανσης (markup languages) που χρησιμοποιούνται για τη δόμηση και την οργάνωση του περιεχομένου. Και οι δύο χρησιμοποιούν ετικέτες (tags) που περικλείονται σε αγκύλες για να ορίσουν τα στοιχεία μέσα σε ένα έγγραφο. Ωστόσο, η XML διαφέρει από την HTML στο ότι είναι πιο ευέλικτη και επεκτάσιμη, επιτρέποντας τη δημιουργία προσαρμοσμένων ετικετών και δομών εγγράφων προσαρμοσμένων σε συγκεκριμένες ανάγκες αναπαράστασης δεδομένων. Ενώ η HTML χρησιμοποιείται κυρίως για την εμφάνιση περιεχομένου στον ιστό, η XML χρησιμοποιείται για την αποθήκευση, τη μετάδοση και την ανταλλαγή δεδομένων σε διαφορετικά συστήματα και πλατφόρμες, επηρεασμένη από την προσέγγιση της HTML για τη σήμανση (markup) και τη δόμηση των δεδομένων. (ref)

Ως προς τους τύπους δεδομένων που υποστηρίζονται από την XML, όπως αυτά αναπαρίστανται στην Εικόνα 3, οι πιο βασικοί είναι το text το οποίο δέχεται από απλό κείμενο μέχρι και HTML markup μέσα σε ενότητες CDATA, αριθμούς όπως integers, floating-point αριθμούς, δεκαδικούς αριθμούς αλλά και επιστημονική σημειογραφία (scientific notation). Επιπλέον, υποστηρίζονται τιμές Boolean είτε σαν True-False είτε με 1 για True και 0 για False αλλά και η δυνατότητα αποθήκευσης δεδομένων σχετικά με την χρονολογία και την ώρα με βάση κάποιο format, όπως για παράδειγμα το "YYYY-MM-DD" για την χρονολογία (ISO 8601) (ref). Παρόλο που η XML είναι ένα format με βάση το κείμενο (text based) υποστηρίζει την αναπαράσταση δυαδικών δεδομένων (binary data) με τη χρήση τεχνικών όπως η

κωδικοποίηση σε Base64 (ref) τα οποία και αποθηκεύονται σαν text μέσα στα XML στοιχεία. Επιπροσθέτως, προσφέρει ιεραρχικές διατάξεις στοιχείων (elements) και χαρακτηριστικών (attributes), διευκολύνοντας τη δημιουργία περίπλοκων δομών δεδομένων. Ακόμη, παρέχει στους χρήστες τη δυνατότητα να προσαρμόζουν την αναπαράσταση των δεδομένων, ορίζοντας προσαρμοσμένους τύπους δεδομένων με τη χρήση elements και attributes, ικανοποιώντας έτσι τις απαιτήσεις συγκεκριμένων τομέων. Τέλος, ενισχύει την ολοκληρωμένη διαχείριση δεδομένων, επιτρέποντας τη συμπερίληψη μεταδεδομένων (metadata ή information about data) στα έγγραφα. Αυτά τα metadata περιλαμβάνουν κρίσιμες λεπτομέρειες όπως το συγγραφικό δικαίωμα, οι ημερομηνίες δημιουργίας, οι πληροφορίες έκδοσης και άλλα συναφή metadata, ενισχύοντας έτσι την κατανόηση του πλαισίου και τη διαχείριση των δεδομένων. (ref)

codeSamples > </> sample.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<data>
    <!-- Textual data -->
    <name>John Doe</name>

    <!-- Numeric data -->
    <age>30</age>

    <!-- Boolean value -->
    <is_student>true</is_student>

    <!-- Date and time -->
    <registration_date>2024-04-05T08:00:00</registration_date>

    <!-- Binary data (Base64 encoded) -->
    <profile_picture>base64_encoded_data_here</profile_picture>

    <!-- Structured data -->
    <address>
        <street>123 Main St</street>
        <city>New York</city>
        <state>NY</state>
        <zip_code>10001</zip_code>
    </address>

    <!-- Custom data type -->
    <product>
        <name>Laptop</name>
        <price>999.99</price>
        <manufacturer>XYZ Electronics</manufacturer>
    </product>

    <!-- Metadata -->
    <metadata>
        <author>John Smith</author>
        <creation_date>2024-04-05</creation_date>
        <version>1.0</version>
    </metadata>
</data>
```

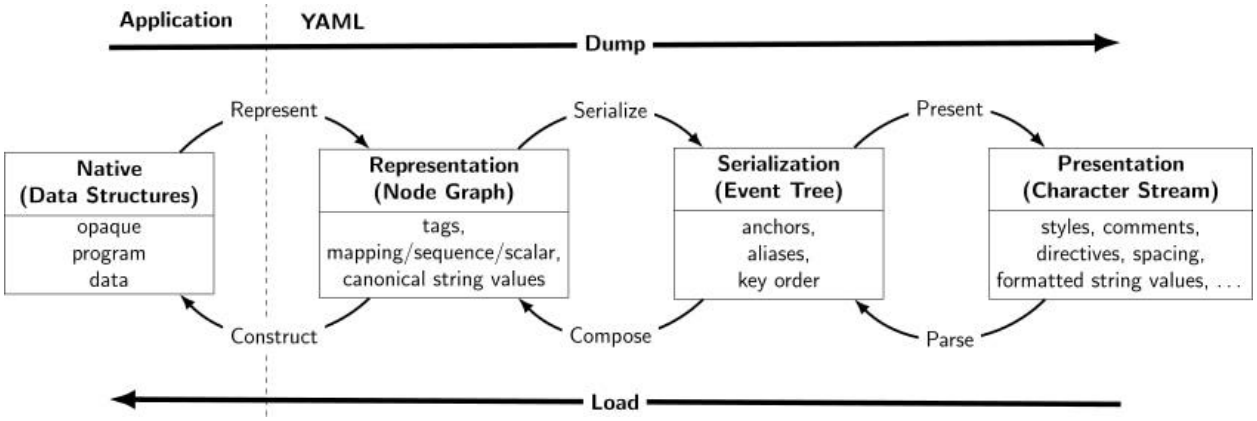
Εικόνα 3: Δείγμα XML αρχείου

2.3.2γ YAML Serialization

Η YAML Ain't Markup Language (YAML) είναι ένα serialization format δεδομένων που χρησιμοποιείται για την αναπαράσταση δομημένων δεδομένων (structured data). Λόγο της εύκολης ανάγνωσης του από τον άνθρωπο χρησιμοποιείται συχνά σε αρχεία ρυθμίσεων, στην ανταλλαγή δεδομένων μεταξύ προγραμμάτων και σε εφαρμογές όπου η αναγνωσιμότητα από τον άνθρωπο αποτελεί προτεραιότητα.

Ιστορικά, η YAML εμπνευσμένη από την XML, δημιουργήθηκε το 2001 από μία ομάδα ερευνητών με κύριους στόχους την εύκολη ανάγνωση της από τον άνθρωπο, την εύκολη μεταφορά και μετάδοση δεδομένων μεταξύ διαφόρων γλωσσών προγραμματισμού και την ευκολία χρήσης της. (ref) Παρόλο που η YAML δεν στηρίζεται σε κάποιο προ υπάρχων standard, όπως η JSON και η XML (ref), μερικά βασικά χαρακτηριστικά του σχεδιασμού της περιέχουν την προσπάθεια ομοιογένειας μεταξύ των τύπων δεδομένων της με τις εγγενείς δομές δεδομένων (native data structures) που έχουν οι δυναμικές γλώσσες προγραμματισμού (dynamic languages), τη σχεδίαση ενός συνεπής μοντέλου το οποίο θα υποστηρίζει γενικά εργαλεία (generic tools) και την έμφαση στην εκφραστικότητα και την επεκτασιμότητα της.

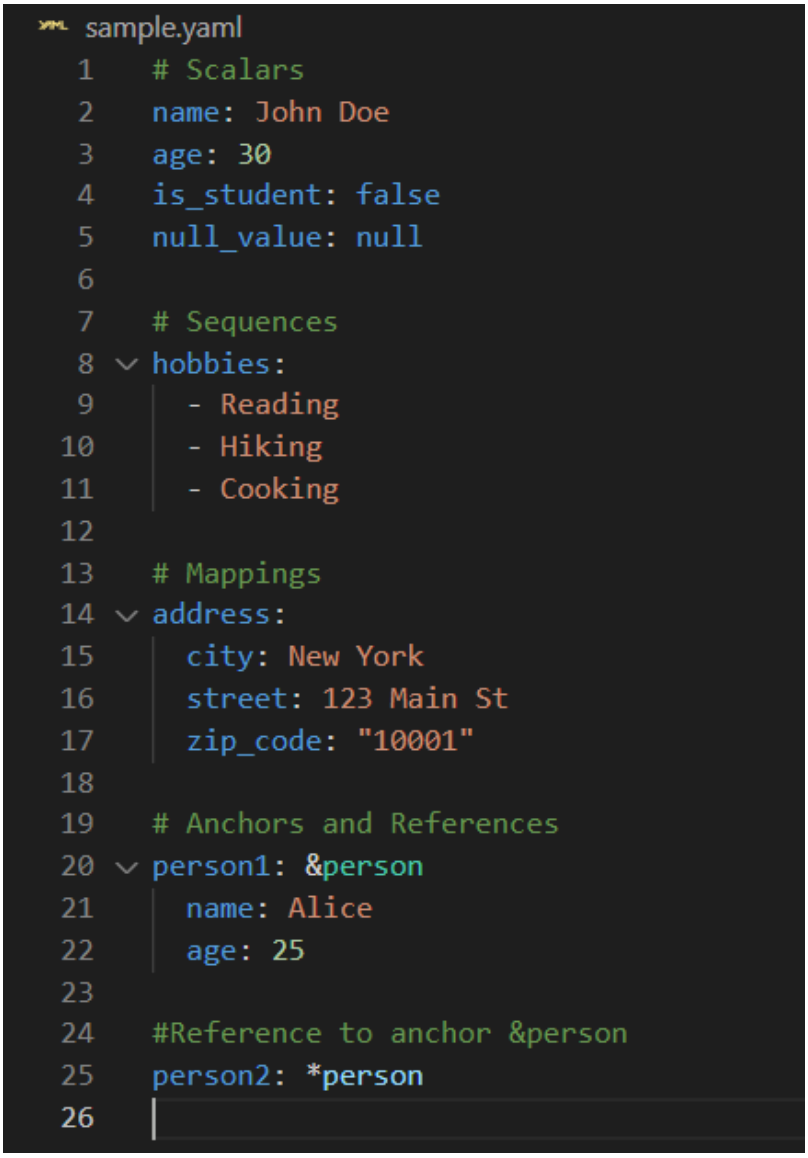
Τέλος, ο μηχανισμός επεξεργασίας της με βάση την επίσημη έρευνα σχεδιασμού της, όπως αυτή αναπαρίσταται στην Εικόνα 4, χαρακτηρίζεται σαν μηχανισμός “one-pass processing” (ref). Αυτό σημαίνει ότι ο αναλυτής (parser) της YAML διαβάζει τα δεδομένα μόνο μία φορά από την αρχή έως το τέλος και τα επεξεργάζεται καθώς τα συναντά. (ref)



Εικόνα 4: Επισκόπηση επεξεργασίας (ref)

Επιγραμματικά, οι τύποι δεδομένων που υποστηρίζει η YAML ως προς την αποθήκευση τους, όπως αυτά αναπαρίστανται στην Εικόνα 5, οι πιο βασικοί είναι οι κλίμακες (scalars) οι οποίες μπορούν να αποθηκεύσουν strings και multi-line strings, integers και floating-point αριθμούς, τιμές Boolean αλλά και την απουσία τιμής είτε με το keyword null ή με το σύμβολο ~ . Επιπλέον, υποστηρίζονται οι διατεταγμένες συλλογές αντικειμένων (ordered item collections) ή όπως τις ονομάζει η YAML, τα Sequences όπου μπορούν να περιέχουν κάθε αναφερόμενο τύπο δεδομένων, τις συλλογές (Mappings) από ζεύγη κλειδιών-τιμών (key-value pairs) όπου το κλειδί πρέπει να είναι μοναδικό μέσα στο σύνολο του Mapping αλλά η τιμή μπορεί να περιέχει ακόμα και nested sequences ή άλλα mappings. Τέλος η YAML περιέχει και τα λεγόμενα Anchors και References. Συγκεκριμένα, το χαρακτηριστικό των anchors και των references που διακρίνεται στη YAML δεν αναφέρεται στα memory address references των αντικειμένων που μόλις έγιναν serialized αλλά χρησιμοποιούνται για τη δημιουργία ψευδώνυμων

(aliases) στο ίδιο το YAML αρχείο. Μέσω της χρήσης τους, αποφεύγεται η διπλοτυπία διατύπωσης των δεδομένων μέσα στο αρχείο και αυξάνεται η ευκολία ανάγνωσης του. Όταν ορίζεται ένα anchor και μετά γίνεται reference αυτού σε κάποιο σημείο μέσα στο YAML αρχείο, ουσιαστικά δίνεται η εντολή στον parser να χειριστεί αυτές τις εμφανίσεις σαν πανομοιότυπες ως προς το περιεχόμενο τους. Εν γένει, είναι ένας μηχανισμός για την επαναχρησιμοποίηση δομών δεδομένων εντός του ίδιου αρχείου YAML, αλλά δεν δημιουργεί καμία σύνδεση με τα αρχικά serialized αντικείμενα ή δομές δεδομένων εκτός του πλαισίου του αρχείου. (ref)



Εικόνα 5: Δείγμα αρχείου YAML

2.3.2δ Binary Serialization

Η δυαδική σειριοποίηση (binary serialization) είναι μια διαδικασία μετατροπής δομών δεδομένων ή αντικειμένων σε δυαδική μορφή (binary), ώστε να μπορούν να αποθηκευτούν, να

μεταδοθούν ή να ανακατασκευαστούν αποτελεσματικά αργότερα. Σε αντίθεση με τις μορφές JSON ή XML και YAML, οι οποίες είναι δυνατό να αναγνωστούν από έναν άνθρωπο, το binary serialization κωδικοποιεί τα δεδομένα σε μια συμπαγή, αναγνώσιμη μόνο από μηχανήματα μορφή, η οποία είναι ιδιαίτερα χρήσιμη σε σενάρια όπου η αποδοτικότητα, η ταχύτητα, το τελικό μέγεθος των serialized δεδομένων ή η μειωμένη χρήση bandwidth είναι σημαντικά ζητήματα.

Αρχικά,